

Oracle® Cloud

Using Processes in Oracle Integration Generation 2



E85489-58
February 2023



Oracle Cloud Using Processes in Oracle Integration Generation 2,

E85489-58

Copyright © 2017, 2023, Oracle and/or its affiliates.

Primary Author: Oracle Corporation

This software and related documentation are provided under a license agreement containing restrictions on use and disclosure and are protected by intellectual property laws. Except as expressly permitted in your license agreement or allowed by law, you may not use, copy, reproduce, translate, broadcast, modify, license, transmit, distribute, exhibit, perform, publish, or display any part, in any form, or by any means. Reverse engineering, disassembly, or decompilation of this software, unless required by law for interoperability, is prohibited.

The information contained herein is subject to change without notice and is not warranted to be error-free. If you find any errors, please report them to us in writing.

If this is software, software documentation, data (as defined in the Federal Acquisition Regulation), or related documentation that is delivered to the U.S. Government or anyone licensing it on behalf of the U.S. Government, then the following notice is applicable:

U.S. GOVERNMENT END USERS: Oracle programs (including any operating system, integrated software, any programs embedded, installed, or activated on delivered hardware, and modifications of such programs) and Oracle computer documentation or other Oracle data delivered to or accessed by U.S. Government end users are "commercial computer software," "commercial computer software documentation," or "limited rights data" pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, the use, reproduction, duplication, release, display, disclosure, modification, preparation of derivative works, and/or adaptation of i) Oracle programs (including any operating system, integrated software, any programs embedded, installed, or activated on delivered hardware, and modifications of such programs), ii) Oracle computer documentation and/or iii) other Oracle data, is subject to the rights and limitations specified in the license contained in the applicable contract. The terms governing the U.S. Government's use of Oracle cloud services are defined by the applicable contract for such services. No other rights are granted to the U.S. Government.

This software or hardware is developed for general use in a variety of information management applications. It is not developed or intended for use in any inherently dangerous applications, including applications that may create a risk of personal injury. If you use this software or hardware in dangerous applications, then you shall be responsible to take all appropriate fail-safe, backup, redundancy, and other measures to ensure its safe use. Oracle Corporation and its affiliates disclaim any liability for any damages caused by use of this software or hardware in dangerous applications.

Oracle®, Java, and MySQL are registered trademarks of Oracle and/or its affiliates. Other names may be trademarks of their respective owners.

Intel and Intel Inside are trademarks or registered trademarks of Intel Corporation. All SPARC trademarks are used under license and are trademarks or registered trademarks of SPARC International, Inc. AMD, Epyc, and the AMD logo are trademarks or registered trademarks of Advanced Micro Devices. UNIX is a registered trademark of The Open Group.

This software or hardware and documentation may provide access to or information about content, products, and services from third parties. Oracle Corporation and its affiliates are not responsible for and expressly disclaim all warranties of any kind with respect to third-party content, products, and services unless otherwise set forth in an applicable agreement between you and Oracle. Oracle Corporation and its affiliates will not be responsible for any loss, costs, or damages incurred due to your access to or use of third-party content, products, or services, except as set forth in an applicable agreement between you and Oracle.

Contents

Preface

| | |
|-----------------------------|-------|
| Audience | xviii |
| Documentation Accessibility | xviii |
| Diversity and Inclusion | xviii |
| Related Resources | xix |
| Conventions | xix |

1 Get Started with Processes

| | |
|--|-----|
| Access Processes for Oracle Integration | 1-1 |
| Learn About Processes | 1-1 |
| Become Familiar with Process Language and Technology | 1-2 |
| Learn About Roles for Processes | 1-2 |
| Tour the My Tasks Page | 1-3 |
| Tour the Process Applications Page | 1-4 |

Part I End User Tasks

2 Work on Tasks

| | |
|---|------|
| What can I do with Process? | 2-1 |
| Start an Application | 2-1 |
| Complete Your Assigned Tasks | 2-2 |
| How do I work on tasks? | 2-2 |
| How do I find, filter, and sort my tasks? | 2-4 |
| Filter the Task List | 2-4 |
| Sort the Task List | 2-6 |
| How do I send email notifications from a comment? | 2-7 |
| How do I add a document to a task? | 2-7 |
| How can I manage documents in folders? | 2-8 |
| Can I reassign my tasks to someone else? | 2-9 |
| Set Your Preferences | 2-10 |

| | |
|--|------|
| How do I set my out-of-office preferences? | 2-10 |
| Can I configure accessibility options? | 2-10 |
| Use the Mobile App | 2-11 |

3 Troubleshoot General Issues

| | |
|---------------------------------------|-----|
| I can't sign in | 3-1 |
| I don't see any applications to start | 3-1 |
| I don't see any assigned tasks | 3-1 |

Part II Developer Tasks

4 Developer Basics

| | |
|--|------|
| Applications, Samples, and QuickStarts | 4-1 |
| Create Your First Application (a Quick Start) | 4-2 |
| Create from a QuickStart App | 4-2 |
| Customize the QuickStart App | 4-3 |
| Test Activate the QuickStart App | 4-5 |
| Try It in Test Mode | 4-5 |
| Activate the QuickStart App | 4-7 |
| Personalize and Promote QuickStart Apps to the Gallery | 4-7 |
| Convert an Application to a QuickStart Master | 4-8 |
| Configure How the QuickStart App Looks in the Gallery | 4-9 |
| Decide What Settings Users Can Customize | 4-10 |
| Promote the Application as QuickStart App to the Gallery | 4-12 |
| Promote Applications as Samples to the Gallery | 4-14 |
| Learn More about the Components in an Application | 4-16 |
| Manage the Development Life Cycle | 4-18 |
| Set Your Preferences for the Design-Time Environment | 4-19 |

5 Create and Manage Applications

| | |
|---|------|
| What You Can Do on the Application Home Tab | 5-1 |
| About Application Sharing and Collaboration | 5-4 |
| Manage Spaces for Sharing and Collaboration | 5-5 |
| About QuickStart Apps | 5-6 |
| Create a New Application | 5-7 |
| Edit Application Properties | 5-8 |
| Validate an Application | 5-10 |
| Save and Publish Changes to an Application | 5-11 |

| | |
|--|------|
| Play Processes and Test Applications | 5-12 |
| Map Process Roles to Users in Your Organization | 5-12 |
| About Testing Processes Using the Application Player | 5-13 |
| Test a Business Process | 5-15 |
| About Testing an Application's Activation | 5-18 |
| Test an Application's Activation | 5-18 |
| Customize Titles | 5-20 |
| Customize Instance Titles Displayed in Runtime | 5-20 |
| Customize Task Titles Displayed in Runtime | 5-21 |
| Work with Application Snapshots | 5-22 |
| View the Change History of an Application | 5-23 |
| Define Application Roles | 5-23 |
| Define Business Parameters | 5-24 |
| Localize Processes | 5-25 |
| Localize Applications | 5-26 |
| Add Locale | 5-27 |
| Work with the Localization Editor | 5-28 |
| Add Dynamic Parameters to Task Titles | 5-29 |
| Import and Export Applications and Snapshots | 5-30 |
| Import an Application from your Local File System | 5-30 |
| Export an Application | 5-31 |
| Export Applications or Snapshots from the Application Home Tab | 5-31 |
| Document Your Applications | 5-31 |
| Add Process-Level Documentation | 5-31 |
| Add Application-Level Documentation | 5-34 |
| Add Activity-Level Documentation | 5-35 |
| Generate Application Reports | 5-37 |

6 Develop Structured Processes

| | |
|--|------|
| About Structured Processes | 6-1 |
| What You Can Do Using the Process Editor | 6-5 |
| Typical Workflow for Creating a Structured Process | 6-8 |
| Create a Structured Process | 6-8 |
| Delete a Structured Process | 6-10 |
| Import a Process Created Externally | 6-10 |
| Clone a Structured Process | 6-11 |
| Work with Process Roles and Swimlanes | 6-12 |
| Add and Edit Swimlanes | 6-13 |
| Work with Elements | 6-14 |
| Add Elements | 6-16 |

| | |
|--|------|
| Edit an Element's Properties | 6-17 |
| Find Elements in the Palette | 6-18 |
| Define Process Start and End | 6-18 |
| About the None Start Event | 6-20 |
| About the Form Start Event | 6-20 |
| About the Document Start Event | 6-21 |
| About the Folder Start Event | 6-21 |
| About the None End Event | 6-22 |
| About the Message End Event | 6-23 |
| About the Error End Event | 6-23 |
| About the Terminate End Event | 6-23 |
| Add Human Interaction | 6-23 |
| Connect to Other Processes and Services | 6-24 |
| Use Insight Models in Processes | 6-26 |
| Use Integrations in Processes | 6-27 |
| Control Process Flow with Gateways | 6-27 |
| Exclusive Gateway: Take Only One Path (Data Condition) | 6-27 |
| Inclusive Gateway: Take One or More Paths | 6-29 |
| Parallel Gateway: Take All Paths Simultaneously | 6-30 |
| Event-Based Gateway: Take Only One Path (Event Occurrence) | 6-31 |
| Create a Gateway | 6-32 |
| Work with Data Mapper Elements | 6-35 |
| Work with Decisions | 6-36 |
| Work with Bot Activities | 6-36 |
| Work with Dynamic Processes | 6-37 |
| Work with Sequence Flows | 6-37 |
| About Unconditional Sequence Flows | 6-38 |
| About Conditional Sequence Flows | 6-38 |
| About Default Sequence Flows | 6-39 |
| Work with Intermediate Events | 6-39 |
| About the Timer Catch Event | 6-39 |
| About the Error Boundary Event | 6-40 |
| Work with Draft Processes | 6-40 |
| About the Abstract Flow Element | 6-41 |
| Work with Subprocesses | 6-41 |
| About Reusable Processes (Reusable Subprocesses) | 6-41 |
| About Embedded Subprocesses (Inline Subprocesses) | 6-42 |
| Configure Multi-Instance Markers in Subprocesses | 6-43 |
| Handle Errors with Event Subprocesses | 6-45 |
| Add an Event Subprocess | 6-46 |
| Configure a Start Error Event | 6-46 |

| | |
|--|------|
| Communicate Between Processes | 6-47 |
| Use Send and Receive to Communicate Between Processes | 6-49 |
| Use Message Throw and Catch to Communicate Between Processes | 6-50 |
| Define Input or Output Arguments | 6-51 |
| Define Input and Output Arguments for a Child Process | 6-52 |
| Call Another Process | 6-52 |
| Use Correlations to Communicate Between Processes | 6-53 |
| Correlations Lead to Process Communication | 6-53 |
| Components of a Correlation | 6-54 |
| Define Correlation Keys and Properties | 6-54 |
| Use Your Correlations in a Process | 6-55 |
| Work with Human Tasks | 6-57 |
| Typical Workflow for Creating a Human Task | 6-57 |
| Create Submit and Approval Tasks | 6-58 |
| Configuring Human Tasks | 6-58 |
| Assign Human Tasks | 6-59 |
| Use Forms to Display Task Information | 6-60 |
| Use an External UI to Display Task Information | 6-61 |
| Define an Approval Pattern | 6-62 |
| Configure the Title and Task Summary | 6-62 |
| Configure the Due Date and Priority | 6-63 |
| Bypass the Approval Chain | 6-64 |
| Configure Reminders | 6-64 |
| Configure Task Expiration, Renewal, or Escalation | 6-65 |
| Override Documents Folder Access | 6-66 |
| Specify Task Actions Shown to Users | 6-68 |
| Assign Custom Properties to Structured Process Activities | 6-68 |
| Customize Notification Emails for Human Tasks | 6-70 |
| About Notification Email and Templates | 6-71 |
| Configure Email Notifications | 6-72 |
| Manage Email Templates | 6-73 |
| Configure Email Templates | 6-75 |
| Use Handlebar Helpers | 6-85 |

7 Develop Dynamic Processes

| | |
|--|-----|
| How do dynamic processes work in process applications? | 7-1 |
| Ready to create a dynamic process? | 7-2 |
| Learn dynamic processing basics | 7-3 |
| Create a Dynamic Process | 7-3 |
| Add Human Task Activities and Stages | 7-3 |

| | |
|--|------|
| Create Web Form Presentations for the Dynamic Process | 7-5 |
| Set the Process Start | 7-8 |
| Set Presentations for the Human Task Activities | 7-9 |
| Configure Data Association for the Human Task Activities | 7-10 |
| Test Activate the Application | 7-11 |
| Try Out the Dynamic Process Application in Runtime | 7-12 |
| Take your dynamic process to the next level | 7-15 |
| Set a Stage's Activation | 7-15 |
| Add a Milestone | 7-17 |
| Control Plan Item Behavior Using Markers | 7-19 |
| Define Roles and Their Permissions | 7-20 |
| Set Assignees for Human Task Activities | 7-22 |
| Set the Process to Complete or Close | 7-22 |
| Explore Embedding Process Runtime Components | 7-23 |
| Explore Advanced Dynamic Process Topics | 7-24 |
| Mix and match processes | 7-25 |
| Use a Structured Process in a Dynamic Process | 7-25 |
| Use a Dynamic Process in a Structured Process | 7-29 |
| Model a Dynamic Process in Design Time | 7-41 |
| Create a Dynamic Process | 7-42 |
| Clone a Dynamic Process | 7-43 |
| Create Activities | 7-43 |
| Create Stages | 7-45 |
| Define Process Input and Output | 7-46 |
| Define Process Data Objects | 7-48 |
| Create Process Roles | 7-48 |
| Define Stage Properties | 7-52 |
| Define General Properties for a Stage | 7-52 |
| Define Conditions for a Stage | 7-55 |
| Create Roles for a Stage | 7-59 |
| Define Activity Properties | 7-61 |
| Define General Properties for an Activity | 7-61 |
| Define Conditions for an Activity | 7-71 |
| Create Roles for an Activity | 7-72 |
| Create Simple Expressions | 7-72 |
| Define Process Completion and Termination | 7-78 |
| Enable the Auto Complete Marker for a Process | 7-79 |
| Define a Condition for Process Termination | 7-79 |
| Edit Process Input or Output | 7-80 |
| About Process and Plan Item Lifecycles | 7-80 |
| Process State Model | 7-80 |

| | |
|--|------|
| Stage or Activity State Model | 7-81 |
| State Transitions – An Illustration | 7-82 |
| Work with Inline Validations | 7-85 |
| Work with Dynamic Processes in Runtime | 7-86 |
| What You Can Do on the Process Instance Details Page | 7-87 |
| Work with Activities | 7-90 |
| View Activity Details | 7-90 |
| View Form Data | 7-91 |
| Upload Process Instance Documents | 7-92 |
| View Process Instance Log | 7-93 |
| Override Roles Assigned to a Process | 7-94 |
| End Dynamic Processes and Activities | 7-98 |

8 Create and Use Micro Processes

| | |
|---|-----|
| Create a Micro Process Link | 8-1 |
| Add Micro Processes to Your Main Workflow | 8-2 |

9 Create Web Forms

| | |
|--|------|
| Work in the Web Forms Editor | 9-1 |
| Use the Simple Editor | 9-3 |
| Create a Simple Form | 9-3 |
| Controls in Simple Forms | 9-4 |
| Preview Simple Forms | 9-4 |
| Localize Simple Forms | 9-5 |
| Ready to create a web form? | 9-5 |
| Create a Simple Application | 9-5 |
| Create a Web Form | 9-6 |
| Add and Configure Controls | 9-7 |
| Add Another Presentation | 9-11 |
| Change the Form's Stylesheet | 9-12 |
| Preview the Form | 9-12 |
| Use Forms and Presentations in a Process | 9-13 |
| Define a Control's Behavior | 9-14 |
| Try the Form in Runtime | 9-16 |
| Explore Advanced Form Options | 9-17 |
| Work with Presentations | 9-17 |
| Change Form Presentations Dynamically | 9-20 |
| Position Controls on Forms | 9-21 |
| Position Controls on Forms | 9-25 |

| | |
|--|------|
| Bind Form Data with Controls | 9-27 |
| Create a Web Form Based on a Business Type | 9-28 |
| Work with Stylesheets and Styling | 9-29 |
| Styling Properties | 9-30 |
| Customize Web Forms Using CSS | 9-31 |
| Style the Form Container | 9-32 |
| Style Form Controls | 9-33 |
| Form Reference Styling | 9-39 |
| Styling for Imported Business Types | 9-40 |
| Manage Style Dynamically | 9-41 |
| Some Useful Styling Tips | 9-43 |
| Configure Basic Controls | 9-43 |
| Configure Text Input and Area Fields | 9-44 |
| Configure Buttons | 9-45 |
| Configure Drop-down Select Fields | 9-45 |
| Configure Check Lists and Check Boxes | 9-46 |
| Select or Unselect All Check List Options | 9-48 |
| Configure Radio Buttons | 9-50 |
| Configure Number Fields | 9-51 |
| Configure Date and Time Fields | 9-52 |
| Configure Email Fields | 9-53 |
| Configure Web Address URL Fields | 9-54 |
| Configure Message Fields | 9-54 |
| Configure Links | 9-55 |
| Configure Simple Text Fields | 9-57 |
| Configure Advanced Controls | 9-57 |
| Configure Currency (Money) Fields | 9-58 |
| Configure Phone Number Fields | 9-59 |
| Include Images | 9-60 |
| Upload and Preview a Base64 Image | 9-60 |
| Include Videos | 9-61 |
| Configure Identity Browser Fields | 9-62 |
| Place Controls in Panels, Sections, or Tabs | 9-64 |
| Format the Title and Description of a Panel | 9-65 |
| Indent Sections | 9-66 |
| Configure Repeatable Sections | 9-66 |
| Configure Tables | 9-68 |
| Configure Rich Text Editor Fields | 9-71 |
| Configure Train Controls | 9-72 |
| Configure Divider Controls | 9-76 |
| Configure Static and Dynamic List of Values Fields | 9-77 |

| | |
|---|-------|
| Add an Option to a List of Values Field | 9-78 |
| Clear All Options From a List of Values Field | 9-79 |
| Specify Filters for Controls | 9-79 |
| Create Computed Controls | 9-82 |
| Localize Web Forms | 9-85 |
| Preview Forms and Their Payload | 9-88 |
| Add Dynamic Behavior to a Form | 9-89 |
| Configure Events | 9-90 |
| Specify Actions | 9-92 |
| Specify Conditions | 9-94 |
| Specify Functions | 9-97 |
| Specify Filters in Events | 9-103 |
| Reuse Event Snippets | 9-107 |
| Extract a Snippet | 9-107 |
| Use a Snippet | 9-108 |
| Specify Custom Outcomes for Forms | 9-108 |
| Execute REST Connector Calls in Events | 9-110 |
| Populate Controls Using REST Calls | 9-112 |
| Link and Refresh List of Value Fields | 9-115 |
| Examples of Loops in Forms | 9-115 |
| Configure a Simple Loop | 9-115 |
| Configure a Loop With an Array | 9-116 |
| Example of Current Logged in User Data Function | 9-118 |
| Reuse Forms | 9-119 |
| Save Web Form Data | 9-121 |
| Configure Web Form Snapshots | 9-121 |
| Create an External UI Connection | 9-124 |

10 Manage Application Data

| | |
|---|-------|
| About Managing Data | 10-1 |
| Define Data | 10-1 |
| Typical Workflow for Defining the Data Used Within a Business Process | 10-2 |
| What You Can Do on the Business Types Page | 10-2 |
| Work with Business Objects | 10-5 |
| Create a Business Object | 10-6 |
| Import a Business Object from XML | 10-8 |
| Import a Business Object from JSON | 10-8 |
| Import XML Schema Files | 10-12 |
| Upload a New Version of an XML Schema File | 10-12 |
| Promote Schema Types and Elements to Business Objects | 10-13 |

| | |
|---|-------|
| Manage Business Types | 10-15 |
| Work with Data Objects | 10-16 |
| Create a Data Object | 10-18 |
| Edit or Delete a Data Object | 10-18 |
| Create a Business Exception | 10-19 |
| Create an Enum Object | 10-20 |
| Work with Business Indicators | 10-21 |
| Create Business Indicators | 10-22 |
| Associate and Manipulate Data | 10-25 |
| Work with Expressions | 10-25 |
| Append to Array | 10-35 |
| Filter Array Data Objects | 10-37 |
| Define Conditions for Data Associations | 10-39 |
| Configure Data Association | 10-40 |
| Data Association Tips | 10-42 |
| Data Association for Enums | 10-42 |
| Work with Transformations | 10-43 |
| Create, Apply, and Edit Transformations | 10-44 |
| Create Transformations Between Enum Items | 10-47 |

11 Create Decisions

| | |
|--|-------|
| How do decisions work in process applications? | 11-1 |
| Ready to create a decision model? | 11-2 |
| Create a Simple Application | 11-2 |
| Create a Decision Model | 11-4 |
| Add Decisions | 11-5 |
| Define Input | 11-6 |
| Model Decision Logic | 11-7 |
| Test Your Decisions | 11-11 |
| Create a Service | 11-12 |
| Activate a Decision Snapshot | 11-13 |
| Use the Decision in Your Application | 11-14 |
| Map Data to and from the Decision | 11-15 |
| Try Your Decision in Runtime | 11-17 |
| Work with Decision Models | 11-17 |
| Create Decision Models | 11-18 |
| Create a Decision Model from a Sample | 11-19 |
| Import and Export Decision Models | 11-20 |
| Import a Decision Model | 11-20 |
| Export a Decision Model | 11-20 |

| | |
|--|-------|
| Create a Decision Model from Scratch | 11-20 |
| Understand Decision Model Views | 11-22 |
| Graph View | 11-23 |
| Toolbar | 11-24 |
| Decision Model Canvas | 11-24 |
| Diagram Palette | 11-25 |
| List View | 11-25 |
| Add and Order Decisions | 11-26 |
| Define Expressions with the Friendly Enough Expression Language (FEEL) | 11-29 |
| Data Types | 11-29 |
| Grammar Rules | 11-30 |
| Built-In Functions | 11-31 |
| Conversion Functions | 11-31 |
| Boolean Functions | 11-32 |
| String Functions | 11-32 |
| List Functions | 11-33 |
| Numeric Functions | 11-34 |
| List Iteration Expressions | 11-34 |
| Date, Time, and Duration Functions | 11-35 |
| Conversion Examples | 11-35 |
| Arithmetic Operation Examples | 11-35 |
| Comparison Operation Examples | 11-36 |
| Define Decision Input and Type | 11-37 |
| Create Input Data | 11-37 |
| Define Custom Data Types | 11-39 |
| Model Decision Logic | 11-41 |
| Create Empty Logic Decisions | 11-41 |
| Create Decision Tables | 11-42 |
| Define Decision Table Input | 11-43 |
| Define Decision Table Output | 11-45 |
| Configure Rules | 11-46 |
| Configure a Hit Policy | 11-47 |
| Create Expressions | 11-53 |
| Create If-Then-Else Statements | 11-54 |
| Create Functions | 11-56 |
| Create Contexts | 11-58 |
| Create Lists | 11-61 |
| Create Relations | 11-63 |
| Create Loops | 11-65 |
| Test Decisions | 11-67 |
| Expose Decisions as Services | 11-67 |

| | |
|---|-------|
| Manage Decision Model Snapshots | 11-68 |
| Add Decisions to Applications and Processes | 11-69 |
| Update Decision Connectors | 11-71 |
| Promote Decision Models as Samples to the Gallery | 11-72 |
| Best Practices for Modeling Decision Logic | 11-74 |
| Paid Time Off (PTO) Calculator Example | 11-74 |
| Add Decisions | 11-75 |
| Define Input | 11-76 |
| Model Logic | 11-77 |
| Test Decisions | 11-80 |
| Create Decision Service | 11-81 |

12 Develop Smart Processes

| | |
|--|------|
| Smart Sentries in Dynamic Processes | 12-1 |
| Configure a Decision Sentry in a Dynamic Process | 12-2 |
| Example of a Decision in a Dynamic Process | 12-3 |
| Configure a REST Sentry in a Dynamic Process | 12-8 |

13 Integrate Documents

| | |
|---|-------|
| Why should I integrate documents? | 13-1 |
| How do I integrate with Oracle Content Management? | 13-2 |
| Access Requirements for a Successful Integration | 13-2 |
| Use Documents or Attachments in a Process Application | 13-2 |
| Configure Settings in Oracle Content Management | 13-3 |
| Configure Documents Settings in Oracle Integration | 13-3 |
| Enable or Disable Documents | 13-4 |
| Access Issues and Configuration Changes | 13-5 |
| Roles and Responsibilities | 13-5 |
| Quick Tour of the Documents Page | 13-6 |
| Edit the Properties of the Documents Root Folders | 13-8 |
| Define Folders and Documents | 13-9 |
| Create a Document- or Folder-Initiated Process | 13-10 |
| Use REST API Calls to Instantiate a Process | 13-15 |
| Get Properties for Documents and Folders | 13-17 |
| Sample Use Case for Documents and Process | 13-18 |

14 Integrate with Applications and Services

| | |
|--|------|
| What You Can Do on the Integrations Page | 14-1 |
| Work with Integrations | 14-3 |

| | |
|---|-------|
| About Integrations | 14-4 |
| Configure Integrations | 14-4 |
| Edit Integrations | 14-5 |
| Update Integrations | 14-6 |
| Create REST and Web Service Connectors | 14-7 |
| About REST and Web Services | 14-7 |
| Create a REST Connector | 14-8 |
| Work with Web Service Definition Files | 14-13 |
| Prepare Local WSDL Files Containing Remote References | 14-15 |
| Use SOAP Headers in Data Associations | 14-16 |
| Create a Web Service Connector | 14-17 |
| Apply Message Security to Integrations | 14-20 |
| Invoke Integrations, REST, and Web Services | 14-21 |
| Embed Process UI Components in Other Applications | 14-21 |
| Process UI Components Available for Embedding | 14-22 |
| Before You Embed Process UI Components | 14-23 |
| Download Process UI Component Files | 14-23 |
| Experiment with Process UI Components | 14-24 |
| Consume the Process UI Component | 14-26 |
| Best Practices for Embedding Process UI Components | 14-27 |
| Use Process UI Composite Components (CCA) | 14-32 |
| Work with Insight Models | 14-32 |
| Integrate with Robotic Process Automation Tools | 14-40 |
| Integrate an External UI with the Process Task List | 14-41 |

15 Troubleshoot Application Development

| | |
|----------------------------------|------|
| Troubleshoot Application Import | 15-1 |
| Troubleshoot Business Processes | 15-2 |
| Troubleshoot Data Association | 15-3 |
| Troubleshoot Application Playing | 15-4 |

Part III Administrator Tasks

16 Administrator Basics

| | |
|--------------------------------|------|
| About Administrator Privileges | 16-1 |
|--------------------------------|------|

17 Manage the Runtime Environment

| | |
|---|-------|
| Task Overview for Administering the Runtime Environment | 17-1 |
| Assign and Manage Roles | 17-1 |
| Monitor and Adjust Processes | 17-3 |
| View Alerts | 17-3 |
| Track Process Instances | 17-3 |
| Alter the Flow of a Process Instance | 17-4 |
| Monitor Key Metrics in Dashboards | 17-5 |
| Create and View Business Analytics Dashboards | 17-6 |
| Select System Indicators | 17-8 |
| View and Resend Email Notifications | 17-10 |
| Configure Application Settings | 17-11 |
| Configure Oracle Content Management | 17-11 |
| Configure Audit and Log Levels | 17-11 |
| Configure Oracle Storage Service | 17-13 |
| Enable Email Notifications | 17-13 |
| Email Notifications from Comments in Tasks | 17-14 |
| Archive and Purge Data | 17-14 |
| Schedule Instances Archive and Purge | 17-15 |
| Schedule Analytics Archive and Purge | 17-16 |
| View Archive Requests | 17-17 |
| Work with Archive Data | 17-18 |
| Explore Archive Structure | 17-18 |
| Components of Archive | 17-19 |
| Configure Credentials for Web Services | 17-32 |
| Manage Security Certificates during Runtime | 17-33 |
| Show Dates in User Time Zone | 17-33 |
| Hide Comments and Attachments | 17-33 |
| Set the Default View for Process History | 17-34 |
| Troubleshoot the Runtime Administration | 17-34 |

18 Manage the Design-Time Environment

| | |
|---|------|
| Administer Spaces, Applications, and the Player | 18-1 |
| Administer Spaces | 18-1 |
| Unlock and Delete Applications | 18-2 |
| Delete Apps from the Gallery | 18-3 |
| Enable the Application Player | 18-3 |
| Manage Environments and Activate Applications | 18-4 |
| Configure the Activation Permissions | 18-5 |
| Activate Applications | 18-5 |

| | |
|---|-------|
| Manage Active Applications | 18-10 |
| Manage Security Certificates During Design Time | 18-12 |

Preface

The information in this guide applies to using Oracle Integration in an environment managed by Oracle or managed by you.

Topics:

- [Audience](#)
- [Documentation Accessibility](#)
- [Diversity and Inclusion](#)
- [Related Resources](#)
- [Conventions](#)

Audience

Using Processes in Oracle Integration Generation 2 is intended for developers who want to create process applications, administrators who want to monitor process instances and assign user roles, and end users who want to work on application tasks from a browser or the mobile app.

Documentation Accessibility

For information about Oracle's commitment to accessibility, visit the Oracle Accessibility Program website at <http://www.oracle.com/pls/topic/lookup?ctx=acc&id=docacc>.

Access to Oracle Support

Oracle customers that have purchased support have access to electronic support through My Oracle Support. For information, visit <http://www.oracle.com/pls/topic/lookup?ctx=acc&id=info> or visit <http://www.oracle.com/pls/topic/lookup?ctx=acc&id=trs> if you are hearing impaired.

Diversity and Inclusion

Oracle is fully committed to diversity and inclusion. Oracle respects and values having a diverse workforce that increases thought leadership and innovation. As part of our initiative to build a more inclusive culture that positively impacts our employees, customers, and partners, we are working to remove insensitive terms from our products and documentation. We are also mindful of the necessity to maintain compatibility with our customers' existing technologies and the need to ensure continuity of service as Oracle's offerings and industry standards evolve. Because of

these technical constraints, our effort to remove insensitive terms is ongoing and will take time and external cooperation.

Related Resources

See these Oracle resources:

- Getting Started with Oracle Cloud
- Known Issues for Oracle Integration
- What's New for Oracle Integration
- Oracle Cloud at <http://cloud.oracle.com>.
- Oracle Integration documentation in the Oracle Cloud Library on the Oracle Help Center.

Conventions

The following text conventions are used in this document.

| Convention | Meaning |
|------------------------|--|
| boldface | Boldface type indicates graphical user interface elements associated with an action, or terms defined in text or the glossary. |
| <i>italic</i> | Italic type indicates book titles, emphasis, or placeholder variables for which you supply particular values. |
| <code>monospace</code> | Monospace type indicates commands within a paragraph, URLs, code in examples, text that appears on the screen, or text that you enter. |

1

Get Started with Processes

Look here for everything you need to know to start using Processes.

Topics:

- [Access Processes for Oracle Integration](#)
- [Learn About Processes](#)
- [Become Familiar with Process Language and Technology](#)
- [Learn About Roles for Processes](#)
- [Tour the My Tasks Page](#)
- [Tour the Process Applications Page](#)

See Oracle Cloud Terminology in *Getting Started with Oracle Cloud* for definitions of terms found in this and other documents in the Oracle Cloud library.

Access Processes for Oracle Integration

Processes is automatically included when provisioning the following Oracle Integration versions.



Note:

While **Processes** is visible in the left navigation pane for Oracle Integration Standard Edition instances, you are not licensed to use this component unless you update your instance to Oracle Integration Enterprise Edition, which provides complete licensing access to all components. See *Editing the Edition, License Type, Message Packs, and Custom Endpoint of an Instance in Provisioning and Administering Oracle Integration Generation 2*.

| Version | Processes in Enterprise Edition | Processes in Standard Edition |
|---|---------------------------------|-------------------------------|
| Oracle Integration Generation 2 | Yes | No |
| Oracle Integration for Oracle SaaS Generation 2 | Yes | No |

Learn About Processes

Use the Processes feature to rapidly design, automate, and manage business processes in the cloud.

There are two working environments: the design-time environment, where you develop and test applications, and the runtime environment, where you use and monitor process applications.

The design-time environment:

- Provides business-friendly graphical tools for designing processes, forms, data, decision models, and metrics from scratch
- Includes QuickStart Apps for fast and easy rollout of custom business applications
- Provides test environments for refining processes before activating them for use in production
- Lets you move process applications (metadata and data) from cloud to on-premises

The runtime environment:

- Makes it easy for you to view, complete, reassign, and delegate tasks
- Lets you stay organized with filters
- Lets you share documents and collaborate with others on your team
- Provides tools to track process flows, view detailed audit trails, troubleshoot, and fix processes

The design-time environment is also called Composer, and the runtime environment is also called Workspace. The terms design time and Composer, and runtime and Workspace are used interchangeably throughout this guide.

Become Familiar with Process Language and Technology

Processes has no prerequisites. However, it's helpful if you're familiar with the following technologies, especially if you plan to integrate with applications outside of Processes:

- Business Process Model and Notation (BPMN), the standard language for process applications.

Processes is based on BPMN, but some of the standards have been simplified for greater usability.

- Web Services Description Language (WSDL), a standard way of describing a web service provider to a web service consumer. You can expose process applications as web services and communicate with web services outside of Processes.

www.w3.org/TR/wsdl

docs.oracle.com/javaee/6/tutorial/doc/gijti.html

- Representational State Transfer (REST), an architecture style for designing lightweight, stateless, networked applications that use Hypertext Transfer Protocol (HTTP). Processes has a REST API that you can use to integrate with other applications.

Learn About Roles for Processes

Users can be assigned various roles that allow them to access, administer, and use Processes.

Want to learn more about how roles work in Oracle Integration and in the Processes feature? See Oracle Integration Service Roles and What Users Can Do in Processes by Role in *Provisioning and Administering Oracle Integration Generation 2*.

| Role | Description |
|--|--|
| ServiceAdministrator | <p>A user with the ServiceAdministrator role can perform the following tasks:</p> <ul style="list-style-type: none"> • Access all Process components • Monitor and manage processes • Manage user accounts and grant roles to users • Configure connections to other services, such as Oracle Content Management |
| ServiceDeveloper | <p>A user with the ServiceDeveloper role can perform the following tasks:</p> <ul style="list-style-type: none"> • Develop and activate process applications • Share applications with other developers • Start applications • Perform process tasks • Reassign tasks to other users <p>Users with the ServiceDeveloper role can't access the administration pages in the design-time or runtime environment. Also, they can't create, modify, or delete users for Processes.</p> |
| ServiceUser and ServiceEndUser | <p>A user with the ServiceUser or ServiceEndUser role perform the following tasks:</p> <ul style="list-style-type: none"> • Start applications • Perform process tasks • Reassign tasks to other users <p>The ServiceEndUser role is available only for new instances created using version 22.2 or later.</p> <p>Users with the ServiceUser or ServiceEndUser role can't access any pages in the Processes design-time environment and can't access the Processes administration pages in the runtime environment. Also, they can't create, modify, or delete users for Processes.</p> |
| <i>service_instance_name</i> . CECIntegrationUser (Oracle Content Management user) | <p>In Oracle Content Management, assign the integrations user to this role. In Processes, use this user's credentials to configure the connection to Oracle Content Management.</p> |

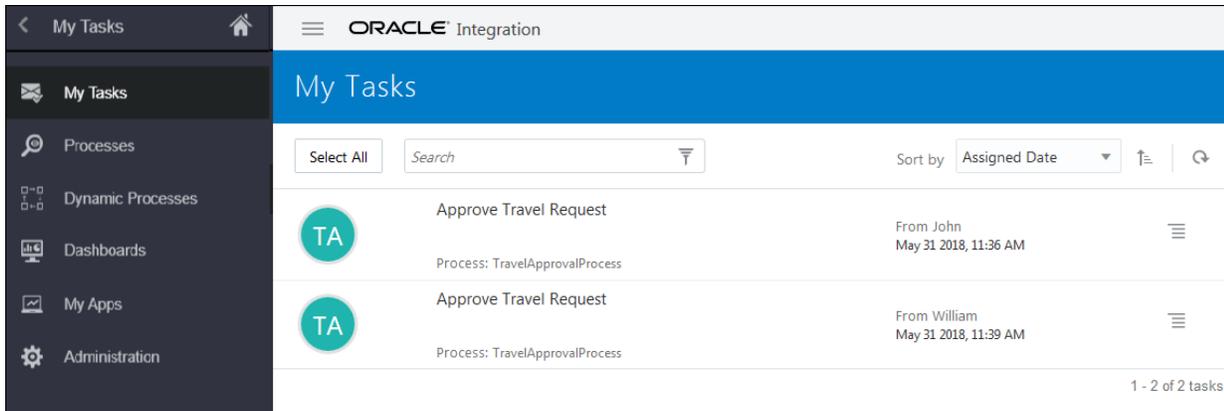
In runtime, each application has process-specific user roles. See [Assign and Manage Roles](#).

In design time, each application has application-specific developer roles. See [About Application Sharing and Collaboration](#).

Tour the My Tasks Page

Depending on your role, use the My Tasks page to work on, monitor, troubleshoot, or administer process tasks.

In the Oracle Integration navigation pane, click **My Tasks** to view your tasks and manage your work.



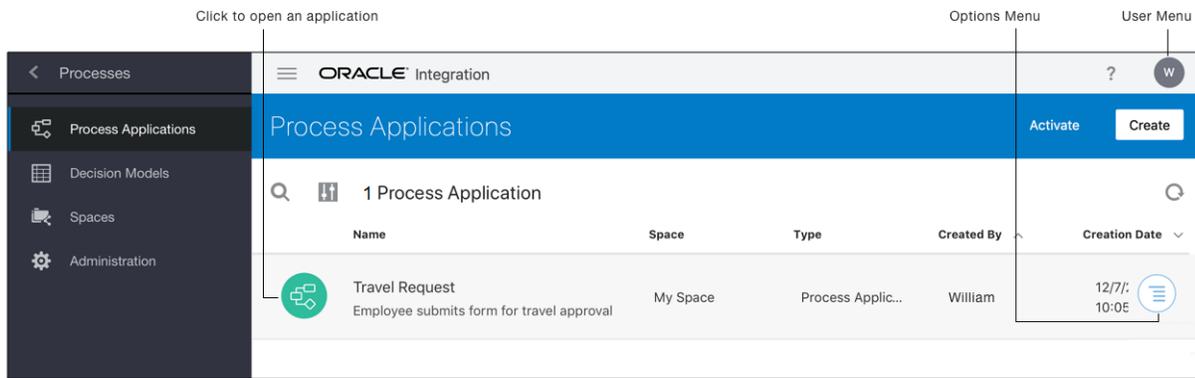
Let's take a closer look at some key functions that you'll find yourself using most often.

| Role | Task |
|---------------|--|
| End User | <ul style="list-style-type: none"> • Start an application, typically by completing and submitting a form. • Work on your tasks. You can work on tasks assigned to you, access detailed information, add information such as comments or documents, and reassign tasks to other participants. • Work on dynamic process. As a knowledge worker, select and complete activities in an unstructured process that depends on expert input or changing circumstances. |
| Developer | Click Processes to create applications, build processes, and manage the application life cycle. |
| Administrator | <p>In the runtime environment, you can work on your tasks, track process instances, view and work with dashboards, start an application, and perform administration tasks. Some of your administration tasks include:</p> <ul style="list-style-type: none"> • Configure the application settings, such as connections to other Oracle Cloud services, roles and assignments, runtime credentials, archive schedules, and notifications. • View dashboards to check the status of activated processes and monitor other key process metrics such as workload and cycle time. • Track running, inactive, and problematic process instances for the applications available to you. • View alerts for suspended, recoverable, and alerted instances. You can go to the instances in these states from each of the alerts. |

Tour the Process Applications Page

A **process application** is the core component of the design-time environment and contains all the required resources of the application, including the business processes.

In the Oracle Integration navigation pane, click **Processes**, and then click **Process Applications**. The Process Applications page provides access to common process application-related features.



Use the Process Applications page to:

- Create applications from scratch, based on a QuickStart App, or by importing
- Manage your applications, including viewing, unlocking, cloning, downloading, and deleting

Be sure to take advantage of the resources and videos available in the Learn More section, and also note the location of the user menu. This menu has options for viewing your language preferences, accessing help (including videos and tutorials), and signing out.

Now that you have an overview of the design-time Home page, let's take a closer look at some key features that you'll find yourself using most often.

| Feature | Description |
|----------|---|
| Activate | Click to activate applications and to manage active applications. |
| Create | Provides options that let you: <ul style="list-style-type: none"> • Create applications from scratch. • Create applications based on a QuickStart App. • Import applications that were previously exported and saved as EXP files. |
| Search | Search applications by name, space, or creator. |
| Filter | Filter process applications by type and owner. Available types are: Process Application , Quickstart App , and Quickstart Master . Available owner options are: Shared with me and Owned by me |
| Update | Click to update the list of applications. If you hover over the icon, the date and time of the last update appear. |

To open an application click its name. For example, click an application name to start adding resources and building your processes. Also, you can open the Options menu to clone, download, or delete applications that are owned or shared by you.

Click to open the application

Application Owner

| Name | Space | Type | Created By ^ | Creation Date v |
|--|----------|-------------------|--------------|--|
|  Travel Request Employee submits form for travel approval | My Space | Process Applic... | William | 12/7/ 10:05  |

Delete

Clone

Download

Options Menu

Part I

End User Tasks

Topics:

- [Work on Tasks](#)
- [Troubleshoot General Issues](#)

2

Work on Tasks

Process enables organizations to automate virtually any business process. Use it to start an application or work on tasks you've been assigned. Use your browser or the Oracle Process Mobile app to work on tasks. Your work automatically syncs between devices.

In Processes, an automated business process is called an **application**, and it contains one or more **tasks** involving humans. Take a travel request process, for example.

- After learning of a travel opportunity, you start the Travel Request application and submit a form.
- The application requires several approvals before you can book travel. Your boss receives an email about the request. To complete the task, your boss clicks the link, signs in, and then approves or rejects the travel request.
- Your request follows a path through each task in the application until complete. Along the way, you or others may perform additional tasks. For example, you may attach supporting documents justifying the travel or you may view and accept a travel policy document.

To use the mobile app, [install and configure](#) Oracle Process Mobile. For browser and mobile access, you'll need the web address to connect to, and your user name and password.

What can I do with Process?

Engage in business processes automated in the cloud.

As a user, you have two main actions:

- Start an application, and then complete and submit a form. For example, you might submit a travel request, a sales opportunity, or an order.
- View and complete your assigned tasks.

The way you and others perform tasks depends entirely on how the process application and tasks are set up. But here are things you typically do:

- [Start an application](#) and complete a form
- [Act on assigned tasks](#) such as Approve, Reject, or Request Info
- [Find tasks](#) and view their details and status
- [Upload documents related to a task](#), download them, and [manage them in folders](#)
- [Set out-of-office routing](#) for assigned tasks

If you're an administrator, learn about additional options in [Task Overview for Administering the Runtime Environment](#). If you're a developer, see [Developer Basics](#).

Start an Application

In the Oracle Integration navigation pane, click **My Tasks** and then click **My Apps**. From the list of available applications, click the one you want to start.



Note:

If you see multiple versions of an application, select the **Show Default Versions Only** field. Selecting this field displays the default (primary) version only of each application.

Complete the form, attach documents if necessary, and click **Submit**. This initiates the first task in the application.

Optionally, click:

- **Discard** to discard the changes and cancel this action.
- **Save** to save your changes and complete the form later. The next time you start this application, the saved form appears.



Note:

If the application you want to start doesn't appear on either page, you may not have the correct role. Contact your administrator to assign you the required role.

Complete Your Assigned Tasks

You can act on tasks assigned to you, access detailed information, add documents or attachments, post comments, and reassign tasks to other participants.

[Find the task](#) you want to work on.

[Work on the task](#) by:

- Modifying the task form
- [Attaching documents](#)
- Taking action on the task, like approving it or [reassigning](#) it to someone else
- Modifying the task priority
- Reviewing the task history



Note:

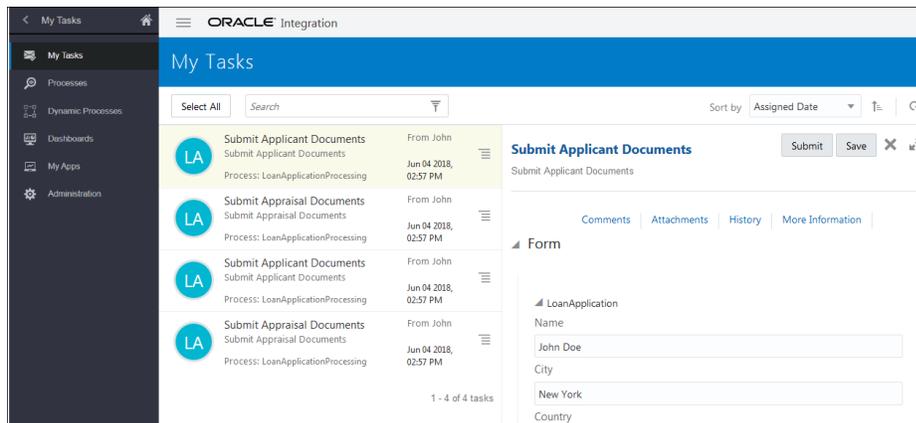
The timeout limit is 60 minutes on inactivity from any page.

How do I work on tasks?

You can act on tasks, view details about the task, modify the task form, and add information to a task. When you take action on a task, it moves to the next step in the process application.

To work on a task:

1. In the Integration Cloud navigation pane, click **My Tasks**.
2. Click the title for the task you want to work on. Now you can see all the details about the task.
3. Click **Resize**  if you want to view both your list of tasks and the details pane for the currently selected task.



4. Optionally, expand the sections you want to review or modify:
 - **Form:** View, print, or modify the task form.
 - **Comments:** View or add a comment explaining your decision or information about the task.
 - **Attachments or Documents:** Depending on how the application was designed, you add attachments in one of two ways. See [How do I add a document to a task?](#)
 - **History:** View the history of the task to understand why it was assigned to you, why it took so long to get to you, and so on.
 - **More Information:** View general information about the task or change the task priority. High priority tasks have a red circle with an exclamation mark.
5. Click the action you want to take, or select an action from the **More** menu if it's available.

The available actions depend on the application. For example, in a travel request application, the actions may be approve or reject. The **More** actions are:

- **Request Info:** You can request more information from the task creator or any of the previous assignees.
- **Escalate:** In case you're unable to complete a task, you can escalate it by adding an optional comment in the Comments section and reassign it to your manager (up one level in a hierarchy).
- **Withdraw:** You can withdraw the task. This results in the process moving to the next activity inline and the task is skipped from the approval process.
- **Reassign:** If you're a manager, you can delegate a task to reportees.
- **Claim:** If a task is assigned to multiple users or a group, you can claim the task to act upon it. After a user claims a task, other assignees won't be able to access the task.

However, after claiming a task if the user doesn't act upon it for a certain period of time (for example, 72 hours), then the task will be automatically available to other users of the group. Any other user of the group can claim the task and perform actions on it. The time period when a claimed but not acted upon task becomes

automatically available to other users in the group is determined by the *task priority* and is not configurable. If the task priority is set to P1, the task becomes available after 24 hours; if priority is P2 after 48 hours, if priority is P3 after 72 hours, and if priority is set to P4 then the task is available after 96 hours.

For a task with multiple assignees, if an assignee performs any action on the task (such as saving the form, adding comments/attachments, or approving/rejecting the form) without claiming the task, then the task is considered as auto-claimed, and all other assignees won't be able to view the task. However, initially, all applicable actions for a task—including Claim—are displayed to all assigned users.

- **Suspend:** You can suspend a non relevant task only if you have been assigned the manager role. A suspension is indefinite. It doesn't expire until **Resume** is used to resume working on the task.

The selected action is performed on the task and it moves to the next step in the application. Unless you're assigned to the next step, the task no longer appears in your task list.

 **Note:**

To perform an action such as Submit or Approve on multiple tasks in the task list, select them by clicking on their icons (for example, ) and choose an action in the toolbar. The color and initials of the icons vary by application.

How do I find, filter, and sort my tasks?

When you select **My Tasks** to view your assigned tasks, you may see many tasks listed.

You can:

- Use the **Search** field to locate a specific task.
- Use the **Filter**  options to change which tasks are displayed. For example, filter by status, assignee, or due date.
- Use the **Sort by** options to change the order of your tasks. For example, sort by assigned date, by priority, or by task title. You can sort the tasks in ascending or descending order.

Filter the Task List

You can apply different filters to the task list to view only tasks that match a certain criteria.

Task List Filters

| Filter | Description |
|----------|---------------------------------|
| My Tasks | Displays tasks assigned to you. |

| Filter | Description |
|--------------------|---|
| High Priority | Displays marked as high priority. |
| Worked on by Me | Displays tasks from process instances that you have started and other tasks that you interacted with in the past, for example, a task that you may have reassigned to someone. |
| Started by Me | Displays tasks from process instances you started. |
| Administered by Me | For users assigned the Process Owner role, displays tasks from applications they administer. |
| Reviewed by Me | For users assigned the Process Reviewer role, displays tasks from applications they can review. |
| Reports to Me | Displays tasks that are assigned to reportees that report to you. For example, if you are a manager who manages ten reportees, you can see the assigned tasks of all the ten reportees. |

Status Filters

You can select one or more status filters.

| Filter | Description |
|----------------|--|
| Assigned | The assigned state indicates a task has been assigned to a user, role, or group and its ready for action by a user. |
| Info Requested | Information is requested on a task. |
| Suspended | Processing has stopped on the task. You can't perform any actions on the task and due dates are suspended while the task remains in this state. |
| Withdrawn | The task is no longer needed. This is one of the possible ways to close a task. |
| Error | The task encountered an unrecoverable error. This is a terminal state. |
| Expired | The task is past its expiration date and has expired. This is a terminal state. |
| Alerted | When there is a recoverable fault that has happened with task assignment, the task is moved to ALERTED state and assigned to the error assignee. |

Assignee Filters

| Filter | Description |
|-------------------|--|
| Me | Displays tasks directly assigned you. |
| Me and My Group | Displays tasks that are assigned to anyone, including you, in the groups you belongs to. |
| Claimed by Others | Displays tasks that are assigned other participants. |

From Filters

You can select one or more participants. When you click the Browse icon, a search dialog box opens. You can search users by first name, last name, email, or ID.

| Filter | Description |
|-----------------|--|
| All | Displays all tasks. |
| [Specific User] | Displays the tasks that were previously processed by this specific user. This option accepts multiple selection. |

Due Date Filters

| Filter | Description |
|------------|---|
| All | Displays all tasks. |
| Today | Displays the tasks that are due today. |
| Tomorrow | Displays the tasks that are due tomorrow. |
| This Week | Displays the tasks that are due this week. |
| This Month | Displays the tasks that are due this month. |

Application Filters

You can select one or more applications. When you click the Browse icon a list with the available applications appears.

| Filter | Description |
|------------------------|--|
| All | Displays all tasks. |
| [Specific Application] | Displays the tasks that belong to this specific application. This option accepts multiple selection. |



Note:

When you click a user or shared filter, you can see the total task count of each filter under **Saved Filters** or **Shared Filters**. Note that a filter's task count is valid only till the due date set on the filter. The task count may not be accurate with respect to future dates (post the due date).

Sort the Task List

You can sort the task list using different options to have a clear view of the work assigned to you.

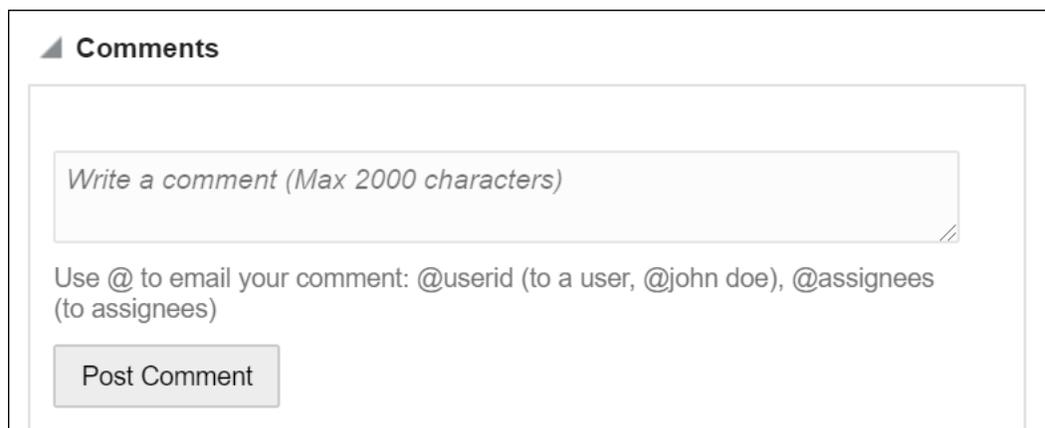
| Sort By | Description |
|---------------|---|
| Due on | Sort the tasks based on their due date. |
| From User | Sort the tasks based on the user that performed the previous task. |
| Application | Sort the tasks based on the application to which the tasks belong. |
| Assigned Date | Sort the tasks based on the date the task was assigned to the current assignee. |
| Updated Date | Sort the task list based on the date it was last updated. |
| Priority | Sort the tasks based on their priority. |
| Title | Sort the tasks based on their title. |
| Ascending | Sort the tasks in an ascending order. This option is used together with another sorting criteria. |
| Descending | Sort the tasks in a descending order. This option is used together with another sorting criteria. |

How do I send email notifications from a comment?

Notify and send emails to a specific user or to all the assignees of a task by tagging them when you enter comments in tasks. This lets you share feedback, ask questions, and notify other task participants immediately on the actions you're performing on the task.

To send email notifications from a comment entered for a task:

1. In the Integration Cloud navigation pane, click **My Tasks**.
2. Click the title of the task you want to work on.
3. Expand the **Comments** field and tag assignees associated with the task in one of the following ways.
 - a. Enter *userid*, for example `@john doe`, to send the email to a specific user.
Enter the first few letters or the whole userid. A pop-up appears with a list of suggested userids depending on what you typed. Select the required userid from the list. Note that you have to always select the userid from the suggested userid list that pops up.
 - b. Enter `@assignees` to send the email to all the assignees of the task.
4. Click **Post Comment**.



The screenshot shows a 'Comments' section with a text input field containing the placeholder text 'Write a comment (Max 2000 characters)'. Below the input field, there is a hint: 'Use @ to email your comment: @userid (to a user, @john doe), @assignees (to assignees)'. At the bottom of the section is a 'Post Comment' button.

An email is sent as you specified. Its subject states that a comment has been added for the particular task. For example - Comment added for task: Submit Loan Application Document.

To view details of the task for which the comment was added, click the task link that is provided in the email.

A message appears in the task if the email is sent successfully. For example, *Notification is sent to @john doe*.

Note that for this feature to work, email notifications must be enabled in the Admin UI. See [Enable Email Notifications](#).

How do I add a document to a task?

Some documents are integral to a task—for example, in a home loan process, you would upload a loan application for approval. Some documents just provide information to other

users—for example, in a travel request process, you might need to upload an agenda that shows why you need to take a more expensive flight at a particular time.

To add a document to a task, click **Documents** , and then depending on how the application was designed perform one of the following steps:

- Click **Upload File** and select a file.
Note that file size of up to 1 MB is supported.
- Select the folder you want to place the document in, click **Upload**, and select a file.
This option is available if the application was configured to use Oracle Content Management. Oracle Content Management lets you manage document files and folders in the cloud. See [How can I manage documents in folders?](#)

After a file is associated with a task, you can select the file and select additional actions.

How can I manage documents in folders?

When Oracle Content Management is integrated with Process, you get more robust file features than you do with the file attachment functionality. You can organize files into folders and subfolders, control folder access, and manage documents.

The Documents folder enables you to upload and manage documents and folders in Process. In the Documents folder, you can perform two types of actions:

- Folder actions
- File actions

The Folder actions are common for all files and folders and the File actions only appear when you select a file.

Folder Actions

The following action items are displayed in the toolbar:

- Share this folder (not available for Process).
- Upload files.
- Create a new folder.
- Open the content in Oracle Content Management. To do so, click **Oracle Documents** .

File Actions

Select a file in the Documents folder to perform any of the following actions:

| Actions | Description |
|----------|--|
| View | View selected file in Oracle Content Management. |
| Download | Download selected file. |
| Move | Move selected items to another location. |
| Copy | Copy selected items to another location. |

| Actions | Description |
|---|--|
| Delete | Delete selected items. |
| Share | Share selected item. This action isn't available for Process. |
| Upload New Version | Upload a new version of the selected file. |
| Version History | View all versions associated with the selected file. |
| Rename | Rename selected item. |
| Properties | View properties of the selected item. |
| Sort by | Sort the selected items using the Last Updated, Name Ascending, and Name Descending filters. Click More Sort Options  to select the filter. |
|  | View file information either in Grid or List format. Click the Views icon, located on the toolbar, to alternate between views. |
|  | Click the Download icon to download files. |

Want to learn more about these folder and file actions? See Manage Your Content in *Collaborating on Documents with Oracle Content Management*.

Can I reassign my tasks to someone else?

You can reassign your tasks to someone else, or delegate them to another user that is helping you with your work.

To reassign or delegate a task:

1. In the Oracle Integration navigation pane, click **My Tasks**.
2. On the My Tasks page, select a task, click **More**, and then click **Reassign**.
3. In the Reassign dialog, select **Reassign** or **Delegate**, specify the user or group you're reassigning the task to, and then click **OK**.
 - **Reassign:** A user can choose to reassign a task to another user/group/role. After reassigning the task, the original user can only view the task but cannot act on it, such as update, submit, or approve it. Only the new user can update, submit or approve the task.

Note:

If the original user is a Process Owner or Admin, then such a user can update, submit, or approve reassigned tasks.

- **Delegate:** A user can choose to delegate a task to another user/group/role. After delegating the task, both the original and new user will be able to update, submit, or approve the task. An example use case for delegate can be - business executives delegating tasks assigned to them to their secretaries for follow up and approvals.

You can use an asterisk (*) as a wildcard when searching for a user or group.

When you reassign a task to multiple users, one of them must claim the task.

Set Your Preferences

You can reassign or delegate your tasks when you're out of the office for a period of time. You can also configure the accessibility options to use with Process.

You can configure the following preferences:

- [Out-of-Office](#)
- [Accessibility](#)

How do I set my out-of-office preferences?

You can configure Process to reassign or delegate your work while you're out of the office.

To enable out-of-office:

1. Click your user name in the top-right corner, and select **Preferences**.
2. Click **Enable Out Of Office** and enter the following details:
 - **Start Date**
 - **End Date**
 - **Reassign To:** The privileges of the new assignee are based on the roles assigned to them.
 - **Delegate To:** The privileges of the new assignee are based on your privileges. Use delegate to assign work to a user that is helping you with your work, for example your assistant.

 **Note:**

A task gets reassigned or delegated only when it's assigned to an individual. If a task is assigned to an application role or to a group with multiple participants or assignees, the task doesn't get reassigned or delegated because any one of assignees can approve the task by following the "Any Single Assignee" approval pattern.

3. Click **OK**.

Can I configure accessibility options?

You can configure Process to use a screen reader and to render in high contrast colors.

To configure accessibility options:

1. Click your user name in the top-right corner, and select **Preferences**.
2. Click **Accessibility**, and then select one or more accessibility options:
 - **Use Screen Reader**
 - **Use High Contrast Colors**

3. Click **OK**.

Use the Mobile App

First download the Oracle Process Mobile app, and then you can use it to start applications and work on tasks from your mobile device.

Download Oracle Process Mobile on Your iOS or Android Device

1. Download the **Oracle Process Mobile** app to your mobile device, or if you're using iTunes download the app and then sync your mobile device.
2. Start Oracle Process Mobile, accept the license agreement, and review or skip the Welcome pages.
3. Configure the following fields with the connection information that your administrator provided, and then tap **Done**.
 - **Host:** The fully qualified server name where Process is running.
 - **SSL Enabled:** Selected by default. Tap to turn off if your organization isn't using SSL.

Note:

If SSL Enabled is turned off, the **Port** field becomes active. In the **Port** field, enter the port number where Process is running.

- **Work Offline:** Off by default. Tap to turn on if you want to work in offline mode.
4. Enter your user name and password to sign in.

Compare Mobile and Browser Features

When it comes to starting an application and working on tasks, you can do many of the same things in the mobile app and the browser. There are a few differences.

| Option | Browser | Phone | Tablet |
|--------------------------------|---------|-------------------------------|--|
| Start an application | ✓ | ✓ | ✓ |
| Work on multiple tasks at once | ✓ | ✓ | ✓ |
| Take action on tasks | ✓ | ✓ | ✓ |
| Filter, search, and sort tasks | ✓ | ✓ | ✓ |
| Create and save filters | | ✓ | ✓ |
| Add attachments | ✓ | ✓ | ✓ (can add from tablet only, not the cloud) |
| Add comments | ✓ | ✓ | ✓ |
| Reassign tasks | ✓ | ✓ (can use phone contacts) | ✓ |
| Track process instances | ✓ | | |
| View dashboards | ✓ | | |

| Option | Browser | Phone | Tablet |
|-------------------|---------|-------|--------|
| Develop processes | ✓ | | |
| Work offline | | ✓ | ✓ |

 **Note:**

With Oracle Process Mobile you can also use the device's native features to take photos, browse pictures and photos, and upload them as task attachments.

3

Troubleshoot General Issues

Topics:

- [I can't sign in](#)
- [I don't see any applications to start](#)
- [I don't see any assigned tasks](#)

I can't sign in

- Try re-entering the user name and password you were provided.
- Your login information is case-sensitive, so make sure the caps lock key isn't on.
- Make sure you have Internet connectivity.

If you still can't sign in, contact your administrator for details about your account or your password.

I don't see any applications to start

If you don't see application icons in the My Apps page, it's likely you're missing the role needed for access. Contact your administrator to assign you the required role.

I don't see any assigned tasks

This may happen because you aren't assigned the role you need to start applications or work on tasks. Contact your administrator to assign you the required role.

Part II

Developer Tasks

Topics:

- [Developer Basics](#)
- [Create and Manage Applications](#)
- [Develop Structured Processes](#)
- [Develop Dynamic Processes](#)
- [Create and Use Micro Processes](#)
- [Create Web Forms](#)
- [Manage Application Data](#)
- [Create Decisions](#)
- [Develop Smart Processes](#)
- [Integrate Documents](#)
- [Integrate with Applications and Services](#)
- [Troubleshoot Application Development](#)

4

Developer Basics

As a developer, you use Process to create, edit, test, and activate applications to a production environment. These applications are high-level wrappers that contain all the resources of a business application, including one or more business processes that determine how the application works.

Topics:

- [Applications, Samples, and QuickStarts](#)
- [Create Your First Application \(a Quick Start\)](#)
- [Personalize and Promote QuickStart Apps to the Gallery](#)
- [Promote Applications as Samples to the Gallery](#)
- [Learn More about the Components in an Application](#)
- [Manage the Development Life Cycle](#)
- [Set Your Preferences for the Design-Time Environment](#)

Applications, Samples, and QuickStarts

Process is packed with smart tools that streamline designing and building applications for business processes. As the developer, you can focus on creating efficient and effective process flows that make it easier for users to complete their everyday business tasks.

Before you jump right in, review the main features of the types of applications you can create and see what each has to offer.

Application and Sample



- Start from scratch or start with a sample
- Model the process from start to end
- Use a sample from the gallery to create a copy of the application
- Work in advanced view
- Access to all process features

QuickStart App



- Ready-to-use app
- Pick one from the gallery
- Users without process knowledge can copy, make certain changes, and activate
- Can use as is or opt to customize

QuickStart Master



- Convert an application
- Configure what items users can personalize (customization view)
- Change process modeling (advanced view)
- Promote to the gallery for others to use to create QuickStart Apps

There's much more, but this gets you started. It's OK to jump in now. Go ahead. Start exploring and discover everything Process can do.

Create Your First Application (a Quick Start)

Ready to create your first application? If so, then here's a good place to start. We'll take you through the main steps—from creating the application to making it available to your end users. Along the way, you'll learn new concepts, pick up some orientation and navigation tips, and complete the entire development life cycle for a travel approval application. And you'll still have plenty to discover.

Topics:

- [Create from a QuickStart App](#)
- [Customize the QuickStart App](#)
- [Test Activate the QuickStart App](#)
- [Try It in Test Mode](#)
- [Activate the QuickStart App](#)

Create from a QuickStart App

The easiest way to create your first application is to start by creating one from a QuickStart App. A **QuickStart App** is a ready-to-use application with all the implementation details included for you to play, test, and activate the application.

To get started:

1. Open the Process Applications page.
 - Go to the Home page and click **Processes**, and then click **Process Applications**.

The Process Applications page opens. In this page, a developer creates, plays, and tests process applications before an administrator activates them for actual end users. The side panel has options related to creating and administering spaces, and managing active applications.

Spaces are containers for applications. They let you easily share applications with other developers. See [Manage Spaces for Sharing and Collaboration](#).

The right panel is your **working canvas**.
2. Create an application.
 - a. Click **Create**, and then click **Start with a QuickStart**. The *Gallery* lists complete applications that you can use as is or adapt to fit your business needs.

Note that both QuickStart Apps and Sample applications are available in the Gallery. Identify and distinguish between the two types of apps with the label (QUICKSTART or SAMPLE) on the top left corner of each application.

Use the filter field on the top right of the Gallery window, and select **QuickStart** from the drop-down menu to show only QuickStart Apps. If you know the name of the QuickStart App, you can type in the name directly in the search field (next to the filter field).
 - b. Take a minute to examine each QuickStart App. The description gives you an idea of what's available. Click **More** to view its process diagram.

- c. Click the **Create** button for the **Travel Approval** application.

In Process, an **application** automates one or more business processes to achieve an outcome. This Travel Approval application enables an employee to submit a travel request, which an approver then approves, rejects, or sends back for resubmission.

In the **Name** field, enter `Request Travel`. Deselect **Open Immediately** for now (we'll show you another way to open your application) and click **Create**.

After your application is created, notice that it's now listed on the Process Applications page.

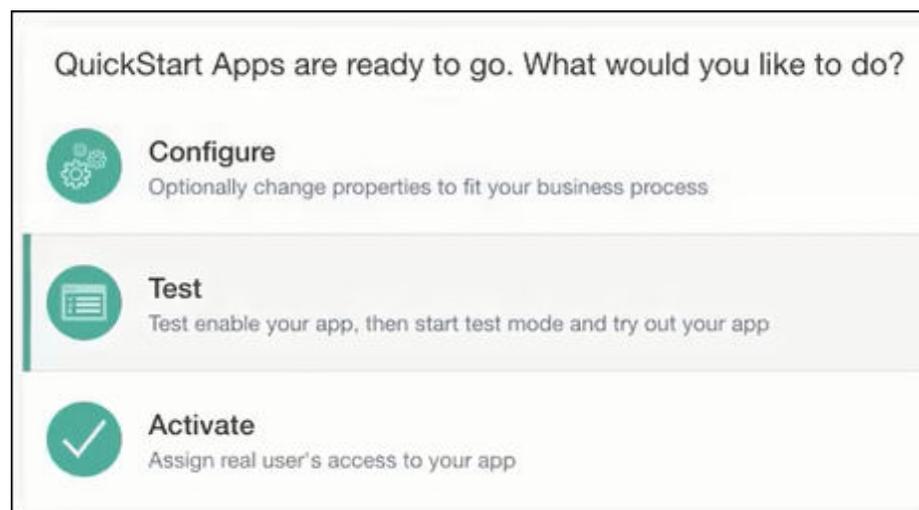
3. Open the application you just created.

- Take a minute to notice a few things on this page.

The page shows **Overview > Configure > Test > Activate** at the top. Overview is selected by default indicating that you're in the application overview page.

The application details are listed. For example, security (locked or lock free), if it's a private or a public application, and language.

Get quick access to configure, test, and activate pages with the three large icons.

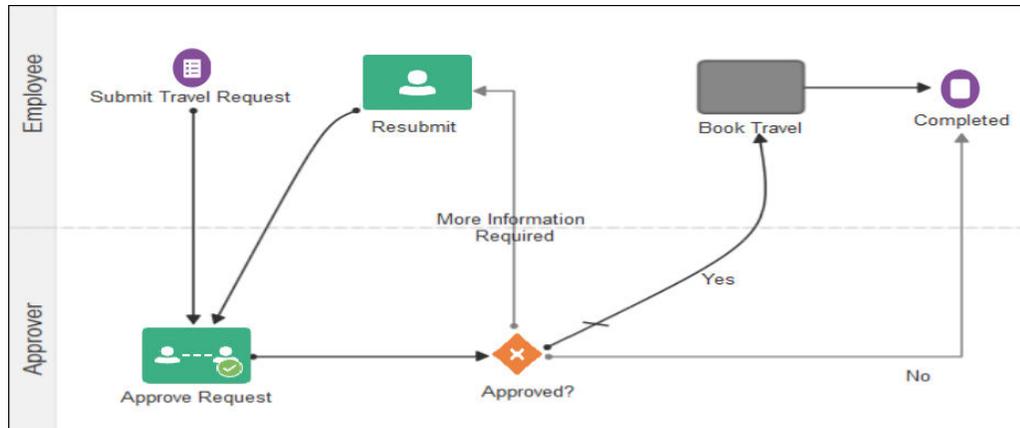


Note that the **Switch to Advanced View** option is at the top right of the page. This means that when the (Travel Approval) QuickStart App was created from the QuickStart Master, the **Switch to Advanced View** option was enabled. It allows users to toggle between the QuickStart and the advanced view. Click **Switch to Advanced View** to access all application features, components, and elements.

Customize the QuickStart App

In the Request Travel application you just created, let's look at how you can customize the application. Although applications created from QuickStart Apps are ready to activate as is, you probably want to change and adapt the application for your business or organization.

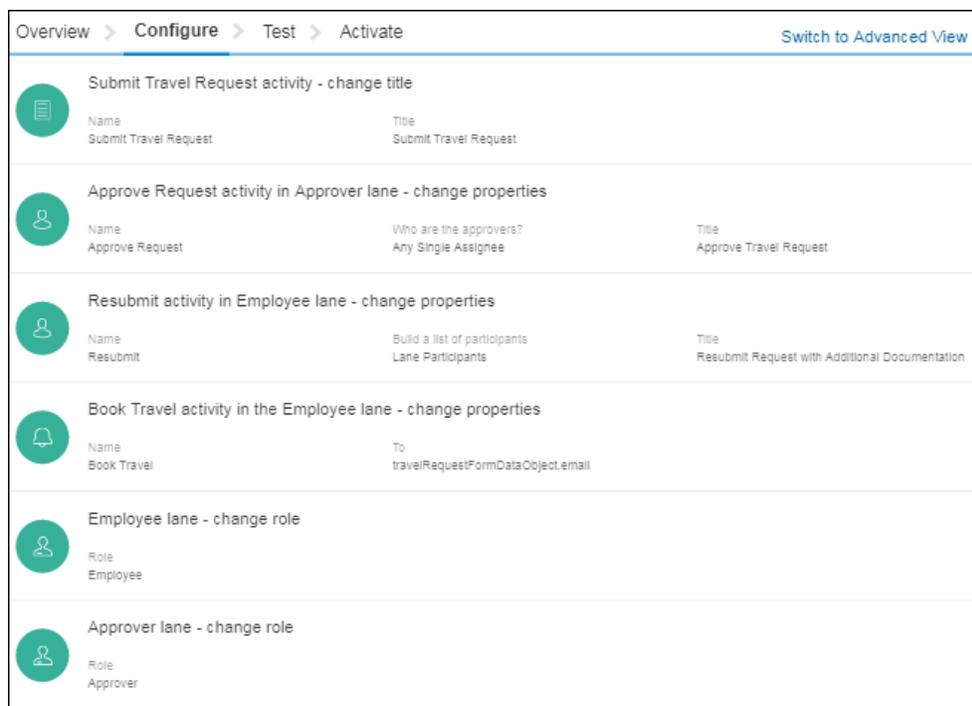
Before you continue, review the process flow for the Request Travel application. An employee completes and submits a travel request form for approval. In the Approve Request task, the designated approver reviews the travel request and takes action. The approver can send the request back to the employee for more information, allow the travel, or deny the travel. Depending on what the approver does, the employee might need to re-submit the form with more information or can book the travel.



To customize the application:

1. Click **Configure**.

Remember that you created this travel application based on an existing QuickStart App. The developer who created the QuickStart Master decides what parts of the application you can customize. For example, in the Request Travel application, you can change settings for the Approve Request task, the Resubmit task, and the Book Travel task.



2. Modify settings associated with the approve task.

- a. Change who can approve the travel request. For example, in the **Build a list of participants** field, select **Names and Expressions**. Then build the list of approvers by clicking **Add** and searching by user name.
- b. Scroll the page and change the priority of this task to **High**.
- c. After you've completed modifying the settings, close the task, and go back to the main Configure page that lists all the activities.

3. Modify settings associated with the resubmit task.
 - a. Click the **Resubmit** activity.
 - b. In the Title field, change the text to `Resubmit request: more info needed`.
 - c. In the Due Date field, enter `10d`. The employee will have ten days to resubmit the travel request.
 - d. In the Reminders section, select **Remind Once**, set the interval to `0M5d0h0m`, and select **Before Due Date**. With these settings, the employee will be reminded 5 days before the due date to resubmit the travel form.

Test Activate the QuickStart App

In this next step, you'll test your application. **Test activating** activates the application to a runtime environment, which lets you play with and refine it. You can test the various user tasks, flows, and outcomes in the process. You don't activate the application to the production environment until it's ready.

To activate your application to a test runtime environment:

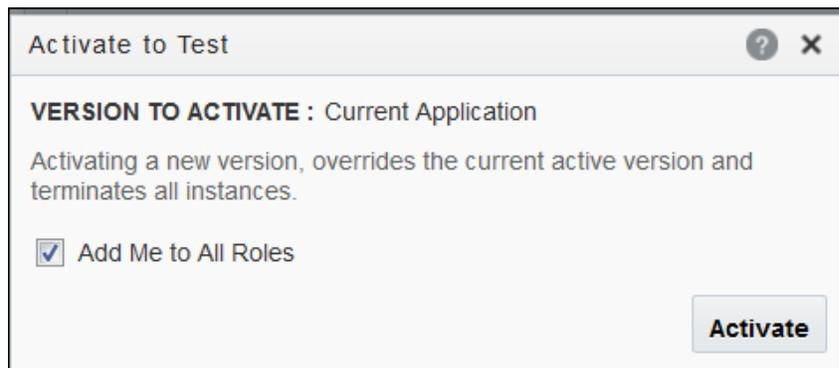
1. Click **Test**.

The Test page opens. Notice that the drop-down list indicates that the current version of the application is selected for test activation.



A screenshot of a web interface showing a dropdown menu with the text 'Current Application' and a small downward arrow. To the right of the dropdown is a blue button with the text 'Activate'.

2. Click **Activate**.
3. Make sure the **Add Me to All Roles** check box is selected (so you'll be able to test any task assigned to any user), and click **Activate**.



A screenshot of a dialog box titled 'Activate to Test'. The dialog has a title bar with a question mark and a close button. The main content area shows 'VERSION TO ACTIVATE : Current Application' followed by the text 'Activating a new version, overrides the current active version and terminates all instances.' Below this is a checked checkbox labeled 'Add Me to All Roles'. At the bottom right of the dialog is a button labeled 'Activate'.

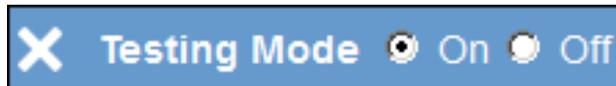
Try It in Test Mode

And now you're ready to try the application in the runtime test mode. Because you added yourself to all roles when test activating the application, you can submit requests in the employee role and also approve or reject requests in the approver role. You're able to test every part of the process.

To test your application:

1. Click **Try in Test Mode** to try out your application as an end user. A new browser tab opens displaying the runtime environment, where end users start applications and complete assigned tasks.

Notice the **Testing Mode** option in the toolbar, indicating test activation:



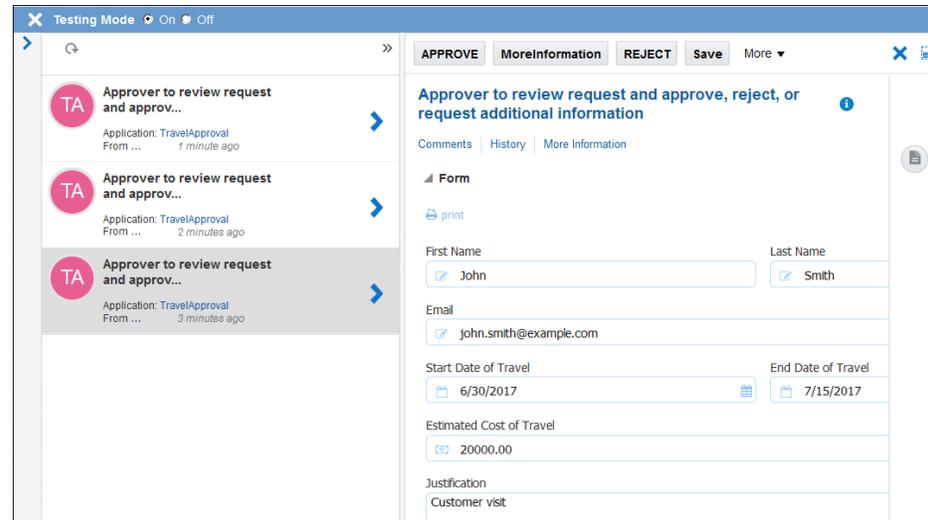
2. Click **My Apps**. Locate your application by its title. Remember, you changed the title to `Request Travel`. Click its badge.
3. Complete the travel form, which starts the process. The rules that were set up for this form require a value in all fields except **Email** and **Justification**. Click **Submit**.

After you submit, a message confirms that you successfully started the application. Click the badge again and submit another request and repeat until you have submitted three requests.

4. Click **Inbox** in the menu bar. Three tasks that begin with **Approver** display.
 - a. Select the first task to view its details, then click **Approve**. In this case, the process is done and removed from the task list.
 - b. Select the next approval task, and this time, click **Reject**. This task is also removed from the list.
 - c. Select the last remaining approval task, scroll the page, and enter this comment:

Please provide details in your justification and submit again.

Click **Post Comment**, and then click **More Information**.



5. Refresh the task list by clicking **Inbox**. Notice that the task now begins with *Employee* rather than *Approver*. Click to open the employee task, enter a justification, and click **Submit**. The task is removed from the list.
6. Refresh the task list once more and notice the task now begins with *Approver*. Select the task and approve the travel request this time.

You successfully tested each possible scenario in your travel request application.

7. Click **Test Mode Close** in the Testing Mode bar to exit.
8. Click **Develop Processes** to return to the Process Applications page.

Activate the QuickStart App

So far, you have created an application from a QuickStart App, viewed and edited the travel approval process, activated the application to a test environment, and then tested each possible task in the process. Your application is good to go. The final step is to make it available to your end users.

Any user who has administrator privileges can activate a QuickStart App to a production environment. As a developer, you may or may not have this privilege.

To activate your QuickStart App to a production environment:

1. Click **Activate**.
2. Click **Activate new version**.

The activate wizard opens. Select which version of your QuickStart App to activate.

Let the wizard guide you through the activation process. Here are a few additional tips:

- You can skip the Customize page. These settings are optional so you can leave all the fields blank.
- On the Activation Options page, you must enter a revision ID. You might also want to select override and default options.

The revision ID is part of the activation. You can enter any number (including sub-revision numbers such as 1.0.1).

- If you enter the same revision number as an existing instance, then you must also select the **Override** check box. Otherwise, the activation will fail because there's already an instance of the application with that number. Also, selecting **Override** causes all existing instances to go stale.
- If you enter a new revision number, then that version is activated separately alongside any other versions. Your new version will be listed on the Manage Active Applications page with the revision number you assigned. Note that if you're activating a new version, then selecting the **Override** check box has no affect.

Optionally, you can specify that this version be marked as the default version. Because applications are activated using a revision number, you can reference an individual version by its revision number. You can also reference the default version. For example, when calling the REST API to initiate a process, you might want to initiate whatever version has been marked as that default instead of initiating a version by its number.

After you successfully activate your application, you—or a user with administrator privileges—must assign end users to the roles defined in the application process. Roles define what end users can do, such as whether they can start the application and what tasks they're assigned. See [Assign and Manage Roles](#).

Personalize and Promote QuickStart Apps to the Gallery

Do you encounter situations at your company where several departments use a business process that is similar, but has slight variations? For example, is the hiring process slightly different for the sales, development, and manufacturing organizations? Or perhaps a travel approval process varies from one department to another.

Why not put the power into the hands of department managers (and others) to easily customize the process to fit their business requirements? As a developer, you can take any existing application and convert it to a QuickStart Master. From there, you decide what settings and details in the process users can customize. With a single click, you promote the application to the QuickStart gallery. Users can then create a new application from the QuickStart App and customize the process for their needs.

Topics:

- [Convert an Application to a QuickStart Master](#)
- [Configure How the QuickStart App Looks in the Gallery](#)
- [Decide What Settings Users Can Customize](#)
- [Promote the Application as QuickStart App to the Gallery](#)

Convert an Application to a QuickStart Master

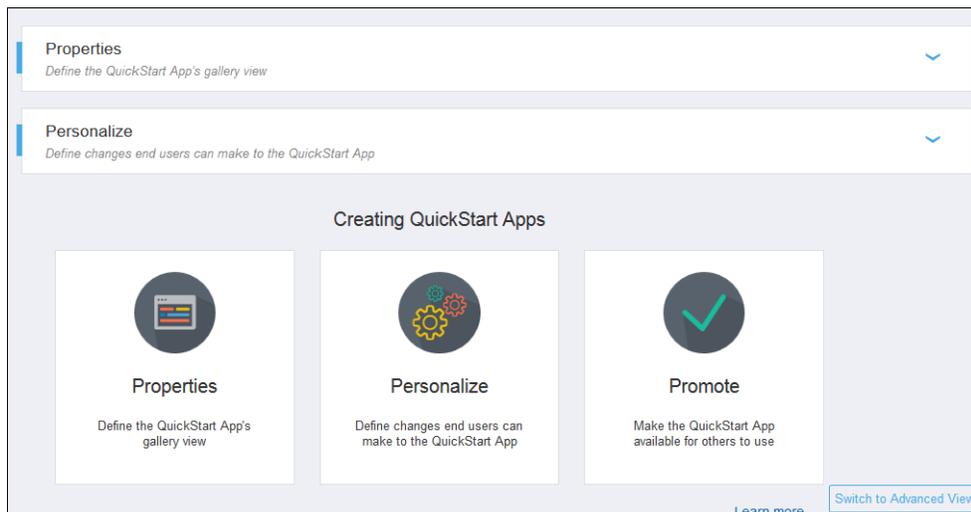
Your first step in promoting a QuickStart App to the gallery is to convert an existing application to a QuickStart Master.

To convert an application to a QuickStart Master:

1. Go to the Home page, and click **Processes**.
2. Use the different icons to distinguish QuickStart Apps, Applications, and QuickStart Masters from each other.



3. Open the application you want to convert.
4. Click **Main menu**  and select **Convert to QuickStart Master**.

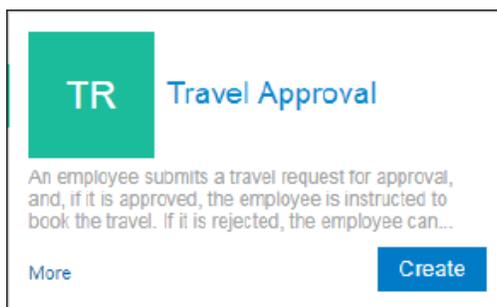


At this point, you're working in what is called **customization view**. From here, you can define how the application looks in the gallery, configure what items users can personalize, and promote the application to the gallery.

Be sure to notice the **Switch to Advanced View** link. If you need to make changes to the process modeling, you can switch to the **advanced view** to access all application features, components, and elements. You can toggle back and forth between the two views at any time.

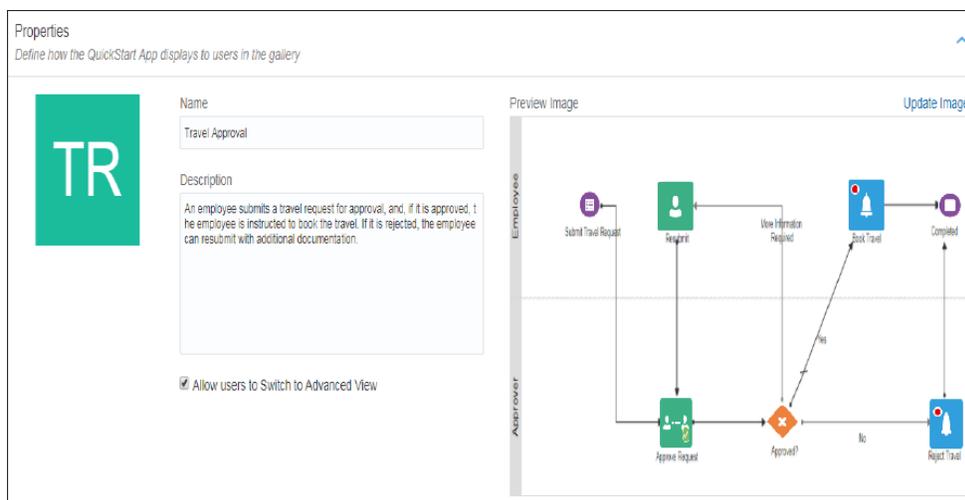
Configure How the QuickStart App Looks in the Gallery

Start by setting the title and description for the application. Users will see this information on the QuickStart App card in the gallery when they go to create their own application.



To set the properties:

1. Click **Properties** to expand the pane.



2. Update the name and description. Be sure to enter meaningful and helpful information.

Also, remember that over time your gallery might grow quite large with hundreds of available applications. Users can search by title so that's one more reason to enter a good one.

3. Click **Update Image** to change the image.

You can use any graphic, illustration, or photo. For example, you might want to include a visual representation of the process but use a more interpretative image instead of the traditional flow diagram. Users will see the image when they click the **More** link on the QuickStart App card in the gallery.

4. Decide whether to allow users to switch back and forth between customization view and advanced view. Advanced view gives users access to the Application Home tab.
5. Click **Collapse**  to close the Properties pane.

Decide What Settings Users Can Customize

Your next step in creating a QuickStart App is to decide what items and properties in the process you'll allow the user to change.

To configure the settings:

1. Click **Personalize**.

2. Take a minute to review the page.
 - The left pane lists items that are defined in the process flow, items that use a form to interact with end users, and the connectors defined within the application. An item may be any of the following:
 - A human task activity or stage in case of a dynamic processes.
 - A swimlane or an activity (such as submit, approve, or notify tasks) in case of a structured process.
 - A human task activity or start form event in case of a form.
 - A REST service, web service, or ICS connector.
 - The right pane lists the elements and properties that you can configure.
3. Select an existing item in the left pane to modify.
 - a. Change the label to be concise and informative. It will appear in the left column. Users will use the label for navigation when customizing the application.
 - b. In the Title field, enter a description that gives the user more information about what the task or activity is about. Users will see this title at the top of the page
 - c. If it's not clear from the label and title what properties the user can change or why, add instructions in the Help field. When customizing their QuickStart App, users can click **Help**  to view the information.

- d. Choose the element that you want to customize within this item. The options available are **Process**, **Connector** and **Form**.
- e. Click **Process** to select the properties that the user can change for an item within a particular process. Specify the process and the item using the following drop-down fields.

All structured and dynamic processes defined within the application are available in the **Process** drop-down field.

For a dynamic process, the **Type** drop-down field contains all stages and human task activities within the process. For a structured process, this field contains all lanes and activities available in the process.

- f. Click **Connector** to select the properties that the user can change with respect to a particular connector. Specify the connector in the following drop-down field.

All REST service, web service, or ICS connectors defined within the application appear in the drop-down field.

- g. Click **Form** to select the form and the presentation that the user can use with respect to a human activity or a start form event. Specify the form and the presentation to display in the **Form** and **Presentation** drop-down fields.

All forms and associated presentations defined within the application are available in the **Form** and **Presentation** drop-down fields. You can see the selected form in preview mode.

- h. After you select an element (either a process item or a connector), click **Choose** to select the properties that the user can change. All properties applicable to that element are displayed; each element has a unique set of properties that can be modified. Don't forget to select the preferred value for each property. Users will see this default value when creating their application, but can change it if they want.

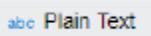
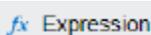
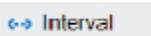
For example, properties for an approve human task within a structured process include name of the task, who can approve the task, how much time they have to approve, what priority is the task, and how often to send reminders about the task deadline.



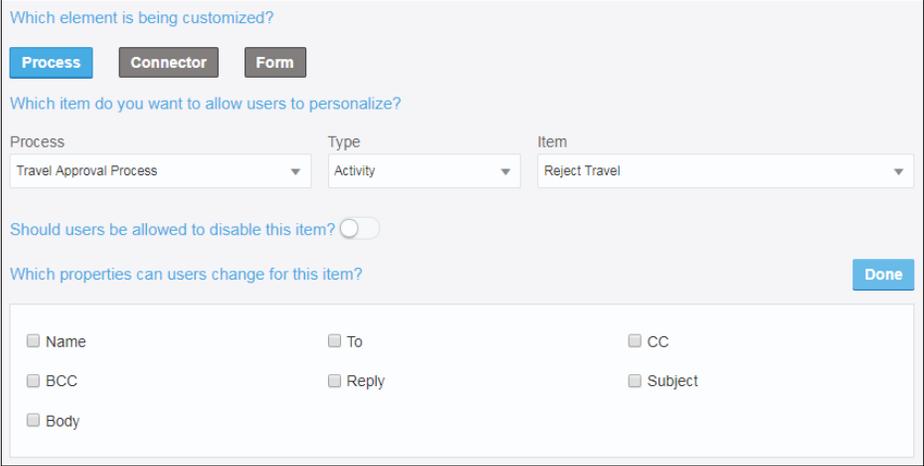
| Which properties can users change for this item? | | | Done |
|--|--|--|------|
| <input checked="" type="checkbox"/> Name | <input checked="" type="checkbox"/> Who are the approvers? | <input type="checkbox"/> Title | |
| <input type="checkbox"/> Task Summary | <input checked="" type="checkbox"/> Due Date | <input checked="" type="checkbox"/> Priority | |
| <input checked="" type="checkbox"/> Reminders | <input type="checkbox"/> Escalation and Expiration | | |

Similar properties are available for a human task activity in a dynamic process. For a REST or web service connector, you can make the following properties available to users: address, read time out, connection time out, and security type.

Notice that for some properties such as Title you can choose either to give a simple string or an expression. Some other properties such as Due Date also has an option to set the time interval. These options are available in a drop-down select field.

-  Plain Text — Click this to define a simple string.
-  Expression — Click this to open a simple expression editor to set the value.
-  Interval — Click this to set the time interval.

4. Add an item to the QuickStart Master that you want to let users customize when they create their own application.
 - a. Click **Add** .
 - b. Skip the Label, Title, and Help Hint fields for the moment. It's easier to enter this information after you know what item you're adding.
 - c. Click the **Process**, **Connector** or **Form** button to choose an element. Here, let's select **Process** to choose an item from a structured process.
 - d. Use the Process, Type, and Item fields to select the item. For example, an activity in which the approver rejects the travel request might be to send an email to the employee stating the reason why the travel request is rejected. You may want to let users modify parts of the email, such as the To, CC, and Subject fields.



- e. Use the check boxes and drop-down list to select the properties the user will be able to change.
 - f. Don't forget to go back to the Label and Title fields and change the default message text.
5. Delete an item that you don't want the user to be able to change.
 - a. Scroll through the items in the left pane.
 - b. Select the item you want to remove.
 - c. Click **Delete** . Don't worry if you inadvertently delete an item. You can always add it back at any time.
6. Publish your changes.

Promote the Application as QuickStart App to the Gallery

At this point, you have converted an application to a QuickStart Master and defined how users can customize the application. Now you just need to promote the application as a QuickStart App to the gallery.

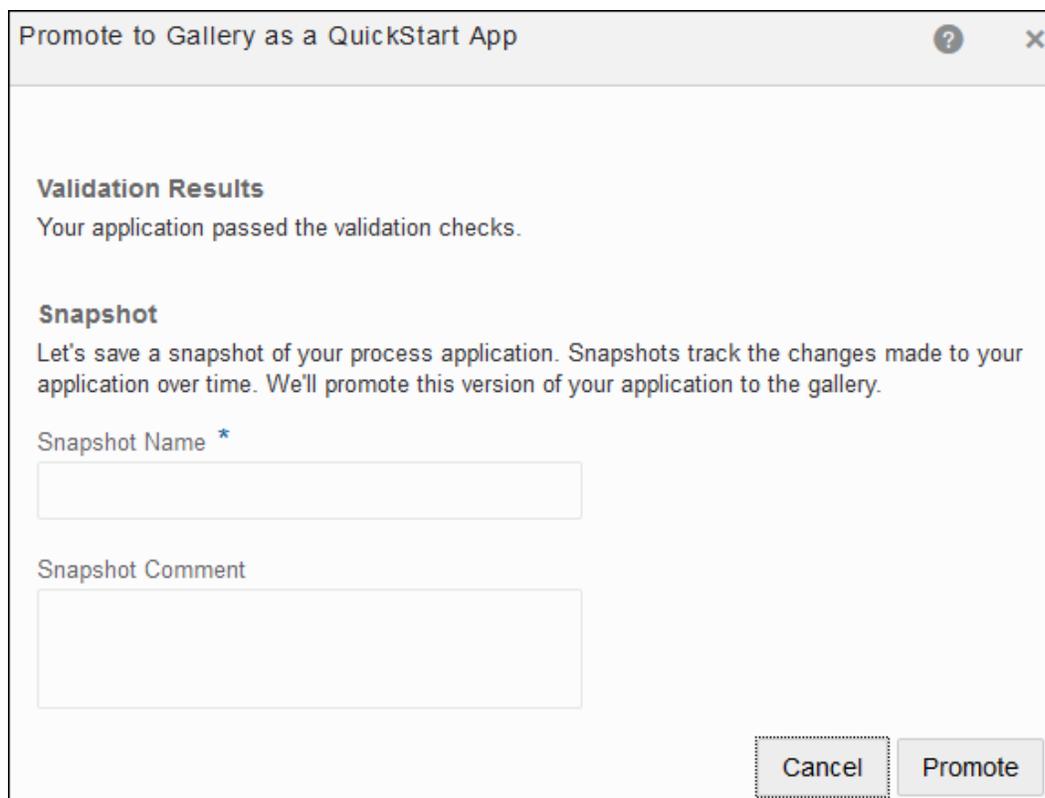
You're promoting a version of your QuickStart App, not the master. You can modify the master at any time and then promote the updated version to the gallery.

To promote the application to the gallery:

1. Test your application.

Even though you converted an existing application, you probably made numerous changes. It's always a good idea to test before promoting an application for others to use.

2. Click **Promote**.



Promote to Gallery as a QuickStart App

Validation Results
Your application passed the validation checks.

Snapshot
Let's save a snapshot of your process application. Snapshots track the changes made to your application over time. We'll promote this version of your application to the gallery.

Snapshot Name *

Snapshot Comment

Cancel Promote

The system validates your application. Be sure to fix any errors before continuing. You can view the validation errors under **Validation Results** in the Promote to Gallery as a QuickStart App dialog box.

3. Enter a name for this snapshot in the **Snapshot Name** field.

Note that this name doesn't appear in the gallery. Instead, the gallery displays the application name you specified in the [Properties pane](#).

You can enter a comment for this snapshot in the **Snapshot Comment** field.

4. Click **Promote** again. You get a confirmation that a snapshot of your application was saved and the application has been promoted as a QuickStart App to the gallery.
5. Click **Close**.

Your QuickStart App is now available in the gallery. Go to the **Process Application** page, and then select **Start with a QuickStart**. Scan or search the gallery for your QuickStart App. Any user can use your QuickStart App to configure, test, and activate their own application.

You can [remove a QuickStart App from the gallery](#) at any time. Users will no longer be able to select the QuickStart App to create an application. However, the source application is still available in the list of applications. You can promote it back into the gallery at any time.

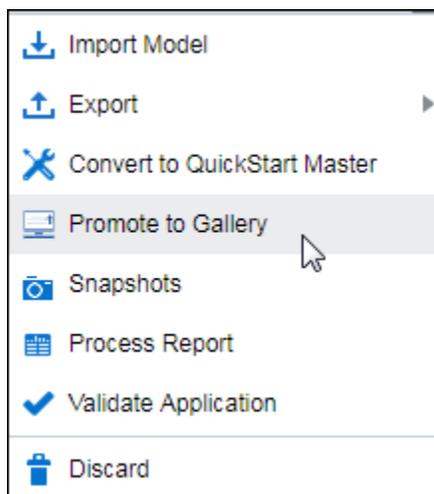
Promote Applications as Samples to the Gallery

If you want to make your process applications available to others, and also allow users to make changes to them in order to suit their business needs, promote them as *Samples* to the gallery.

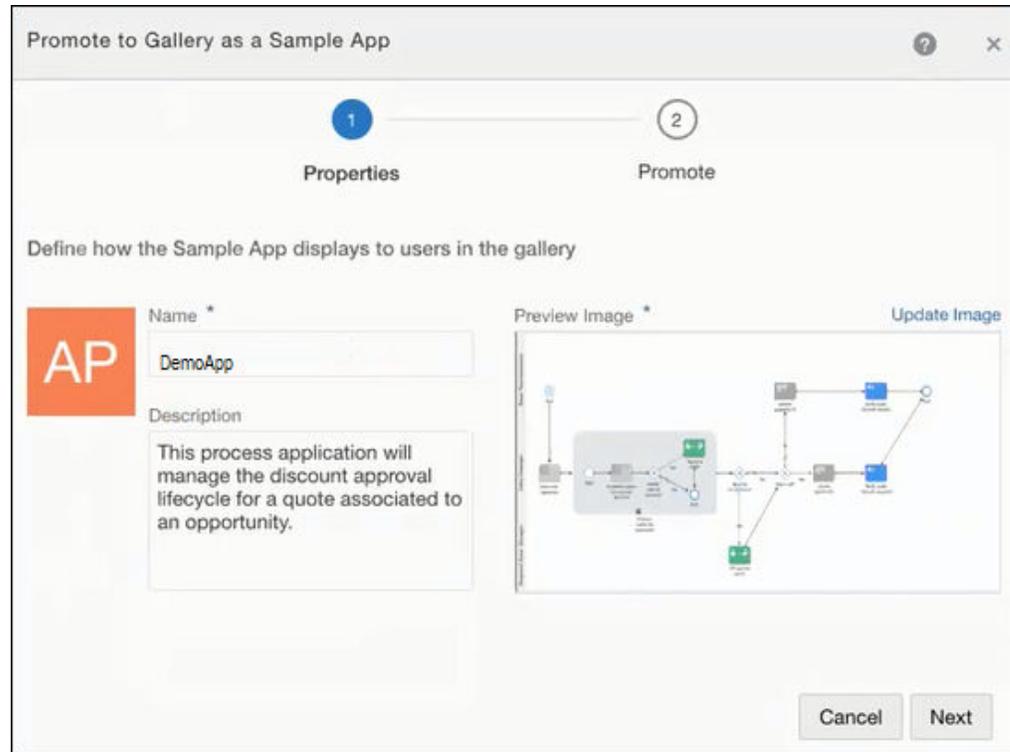
As an advanced user you may like to use sample applications available in the gallery to create a copy of the application.

To promote an application as a sample to the gallery:

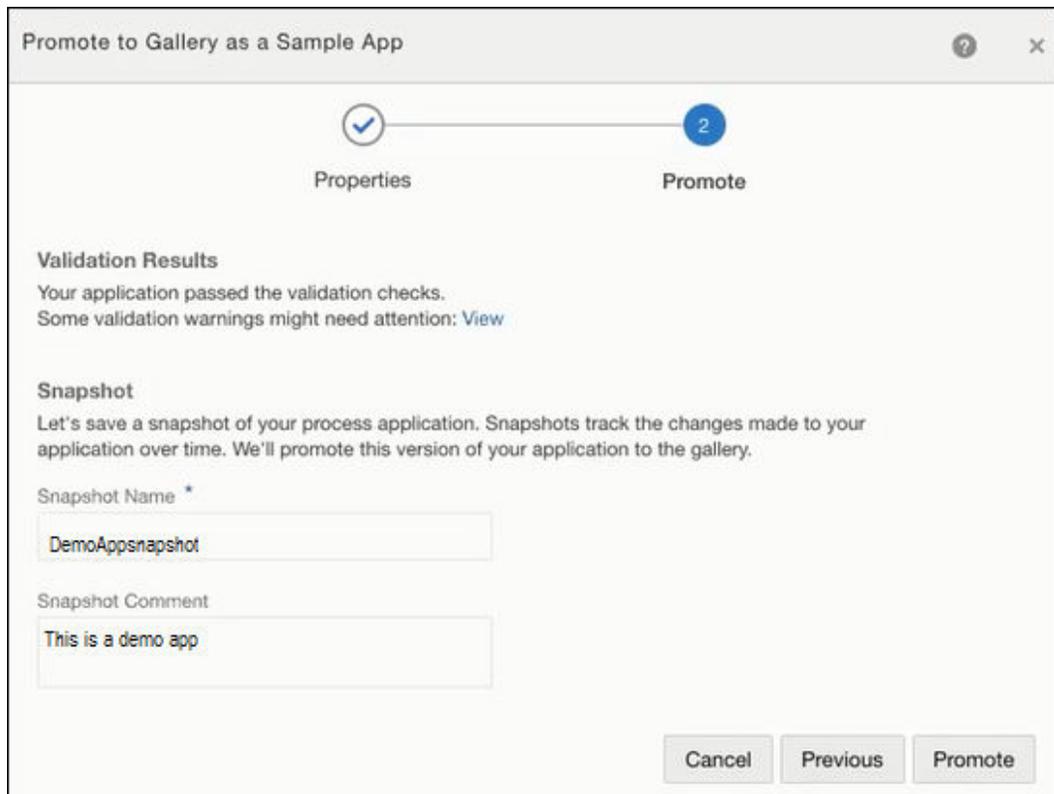
1. Select the application that you want to promote as a sample to the gallery.
2. In the **Application Home** tab, click **Show/Hide Navigation** menu, and select **Promote to Gallery**.



3. In the Properties window of the Promote to Gallery as a Sample App wizard, define how the sample application displays in the gallery.
 - a. Enter a suitable name in the **Name** field.
 - b. Enter a meaningful description in the **Description** field.



4. Click **Next**.
5. In the Promote window, enter a name in the **Snapshot Name** field, and comment in the **Snapshot Comment** field.



6. View and check any warnings under **Validation Results**. You need to fix all major validation errors before promoting the application to the gallery.
7. Click **Promote**. You get a confirmation that a snapshot of your application was saved and the application has been promoted to the gallery.
8. Click **Close**.

Go to the gallery to view the sample application you just created.

The gallery contains QuickStart Apps and Samples. To view all available sample applications, select **Sample** in the filter field. Alternatively, type the name of the sample application in the search field (next to the filter field) to view the application.

Learn More about the Components in an Application

As a developer, it's your job to design and create applications that handle a business process. These applications are made up of components—such as processes, forms, decisions, and documents—that define how the application works.

Processes

A **business process** refers to tasks or activities that result in a well-defined outcome. You can create two types of processes and use both types in a process application:

- **Structured** processes follow a sequential and predictable path. Business Process Model and Notation (BPMN) elements within the process define the flow and behavior of the application.
- **Dynamic** processes automate unpredictable processes that require expert knowledge or changing circumstances.

For most processes, you'll create and edit human tasks. You can configure their duration and deadline, customize the presentation, and add participants and routing. You can also assign users and roles to a human task.

For developing a structured process, here are the topics you'll be most interested in:

- [Develop Structured Processes](#)
- [Work with Human Tasks](#)

For developing a dynamic process, here are the topics you'll be most interested in:

- [Develop Dynamic Processes](#)
- [Ready to create a dynamic process?](#)

Forms

Forms define the interface that your application users see during runtime. Think of it as the user interface for a human task or a start form event that starts an application.

You can create forms from the ground up or you can base them on an existing data structure. You can also add controls to a form, and customize the layout of a form.

Within a form, you can define the behavior of a form, such as showing or hiding certain form controls based on the state of other form controls.

Processes provides the web form editor for creating and editing forms and their behavior. Here are the topics you'll be most interested in:

- [Work in the Web Forms Editor](#)
- [Ready to create a web form?](#)

Business Types

Business types represent real-world concepts or objects, such as a ticket, a request, or an employee. You use business types to create the data structures required in your business application.

Defining how data is stored and manipulated is part of the design and development of an application. You can define complex data types, create data objects, define the expressions used to manipulate the data, define data associations, and define the input and output for your processes.

See [Manage Application Data](#).

Decisions

Decisions are containers for if-then rules and decision tables that use the same input and output data objects. A decision exposes these data objects as a reusable service that multiple business processes can invoke. For example, a decision table can determine whether a manager must approve a travel request based on the travel amount, city, and purpose. See [Create Decisions](#).

Integrations

Integrations define how a business process connects to other processes, systems, integrations, REST services, and web services.

You can create connections to integrations, and exposed REST and web services. Your applications can then communicate and exchange data with these services.

See [Integrate with Applications and Services](#).

Documents

You can create folders to organize and store **documents** in Oracle Content Management. These documents can then be used at Process runtime. You can restrict access to only those folders that are relevant to the objectives of the tasks.

You can also define a document or folder that will be used to start a process. For example, home buyers submitting a loan application starts a mortgage process or job applicants submitting a résumé starts the hiring process.

See [Integrate Documents](#).

Indicators

Business **indicators** enable you to capture and display metrics specific to your process.

You can select data objects as business indicators to capture business metrics, and then display process metrics in custom dashboards.

See [Work with Business Indicators](#).

Manage the Development Life Cycle

By now, you have created your first sample application, activated it to a test environment, and tried it out in the runtime environment. You also explored the different components, such as processes, forms, and decisions, that make up an application. Before you get too involved with creating more applications, let's look at some key ways you can keep your applications in control and under control.

Manage Spaces and Applications

Spaces group related applications and enable users to collaborate when developing applications. You can create additional spaces at any time, add users who can access the space, and specify what access each user has to the space. For example, you decide whether the user can edit applications in the space or only view them. Spaces help with collaboration as well as organization.

As you create and edit applications, you have options to validate, save, and publish your changes. You can also create snapshots of application changes, view the change history, and import and export applications and snapshots.

See [Create and Manage Applications](#).

Document Your Applications

Providing descriptions, notes, and comments about your applications and processes is a good idea. And it's great advice even if it's coming from a writer.

You can add documentation at the application level, the process level, or the activity level. For example, you can use descriptions to help users go to the correct process, explain what the process does, or point to a process that would better serve their needs. Thoughtful details can provide appropriate context for a report about your applications.

If you're collaborating with other developers, then the team can use the documentation fields to share information such as requirements or reminders. When collaborating with others during the creating or editing process, sticky notes are useful. They're highly visible and easily added and removed.

See [Document Your Applications](#).

Play, Test, and Activate Your Applications

As the developer, you can use the application player to test your processes without having to save and activate the application. See [Play Processes and Test Applications](#).

Users with owner or editor permissions can activate the application to the runtime test environment so that they can try it out by simulating the end user experience. And users with administrator privileges can activate applications, and perform specific actions such as retire, activate, or shut down applications. See [Manage Active Applications](#).

Set Your Preferences for the Design-Time Environment

By setting your preferences, you can configure the language locale to the language used by your browser. If you're an administrator, you can also reset your credentials for accessing management options.

To set your preferences:

1. Click the user name located in the menu bar and select **Preferences**.
2. Select the **Use Browser Locale** check box to use your browser's language.

Note that the language currently set is displayed so that you can determine if changing to your browser's language is required.

3. Click **OK**.

If you have administrator privileges, then the Preferences dialog box includes a **Reset Credentials** check box.

As an administrator, you must enter your credentials (user name and password) whenever you click **Management** in the menu bar. You can skip this step if you select the **Remember Me** check box whenever you sign in.

However, there might be times when you want to undo the Remember Me option. You want to clear your saved user name and password, and be forced to reenter them. If that's the case, then simply select the **Reset Credentials** check box and save the change. The next time you click **Management**, the Credentials dialog box prompts for your user name and password.

5

Create and Manage Applications

Let's take a closer look at how you create and manage applications, including sharing spaces to encourage collaboration, testing applications before promoting to production, and generating reports about your applications.

Topics:

- [What You Can Do on the Application Home Tab](#)
- [About Application Sharing and Collaboration](#)
- [About QuickStart Apps](#)
- [Create a New Application](#)
- [Edit Application Properties](#)
- [Validate an Application](#)
- [Save and Publish Changes to an Application](#)
- [Play Processes and Test Applications](#)
- [Work with Application Snapshots](#)
- [View the Change History of an Application](#)
- [Define Application Roles](#)
- [Define Business Parameters](#)
- [Localize Processes](#)
- [Import and Export Applications and Snapshots](#)
- [Document Your Applications](#)
- [Generate Application Reports](#)

What You Can Do on the Application Home Tab

The Application Home tab provides access to information about a Process application and access to common application-related features.

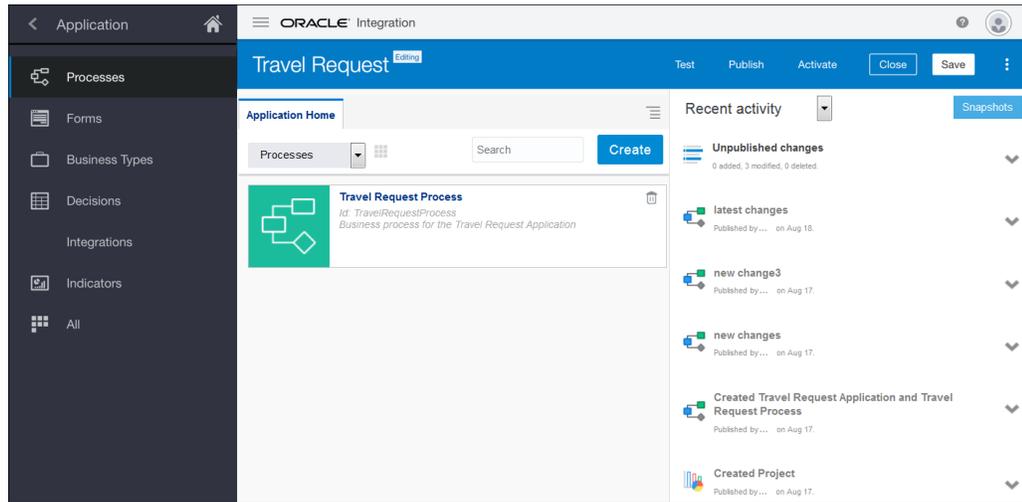
Use the Application Home tab to perform the following types of tasks:

- Edit and view an application's general information.
- Create and manage application components.
- Edit, validate, and publish applications.
- Generate reports.

The Application Home tab is divided into the following sections:

- Application toolbar
- Information pane

- Components pane
- Recent Activity pane



Application Toolbar

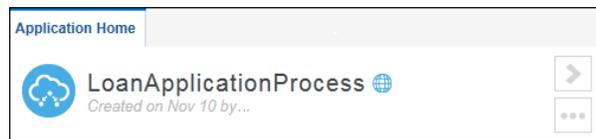
The application toolbar is located across the top of page. It provides access to the main menu as well as to options for saving, publishing, and validating applications.

| Toolbar Icon or Command | Name | Description |
|-------------------------|-----------|---|
| | Main menu | <p>Opens the main menu for the Application Home tab. From this menu, you can:</p> <ul style="list-style-type: none"> • Import a process model • Export the current or last published application • Convert the application to a QuickStart Master or convert a QuickStart Master back to a standard application • Test the activation of an application and try it in runtime • Activate the application to production • View, add, and delete snapshots • Generate a variety of process reports • Save your changes • Discard your changes since the last publication • Validate an application • Close the current application |
| Edit | Edit | <p>Toggles to the Edit mode.</p> <p>When you first open a shared application, it opens in View mode. If the application isn't locked by another participant, you can click this icon to change to Edit mode.</p> |
| Save | Save | Saves the current application. Also, leaves the application open so you can continue editing. |
| Publish | Publish | Publishes the application, makes your recent changes available to other participants who belong to this space. |
| Discard | Discard | Discards changes made to the application since the last publication. |

| Toolbar Icon or Command | Name | Description |
|---|----------------------|--|
| Snapshot | Snapshot | Lets you create, view, and delete snapshots. Take snapshots if you want to track the changes made to the application over time. |
|  | Test Application | Lets you test your application. You can: <ul style="list-style-type: none"> • Use the application player to play your process and try out the flow, including any unpublished changes made to applications. • Activate the application to a test environment so you can try it out and simulate the end-user experience. |
|  | Activate Application | Activates a new version of the application. |
| Validate | Validate | Validates the application. |
| Participants | Participants | Displays the number of participants currently watching this application. This icon is only visible for applications that are shared. |

Information Pane

The Information pane displays some general details and status information about the application.



By default, the Information pane is collapsed when you open an application.

- To expand the pane, click **More details** .
- To collapse the pane, click **Hide details** .

The Information pane also includes a few tools that let you edit the application's description, add languages, convert the application, and change whether this application uses the documents feature. See [Edit Application Properties](#).

Components Pane

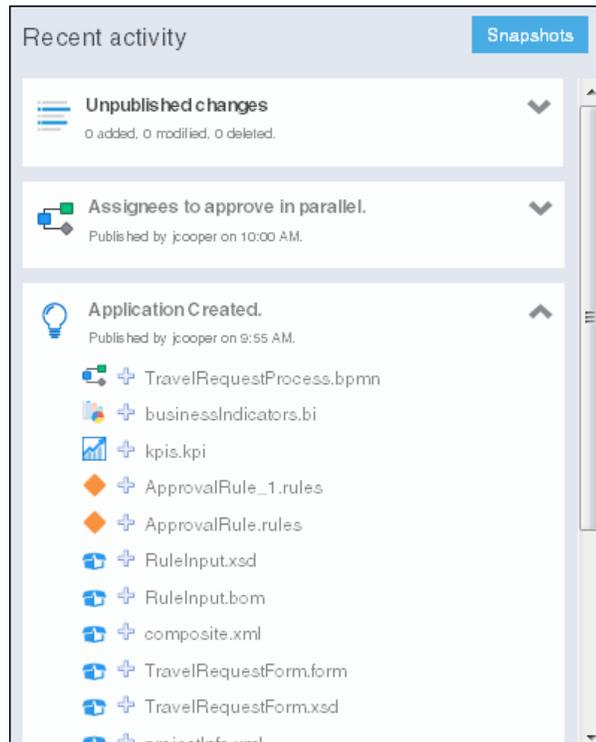
Use the Components pane to view and create the following application components:

- Processes. See [Develop Structured Processes](#).
- Forms. See [Create Web Forms](#).
- Business Types. See [Manage Application Data](#).
- Decisions. See [Create Decisions](#)
- Integrations. See [Integrate with Applications and Services](#).
- Documents. See [Define Folders and Documents](#).
- Indicators. See [Work with Business Indicators](#).

The number next to each component type indicates how many have been created for the current application. The number next to **All** is the total number of components created for the application.

Recent Activity Pane

The Recent Activity pane provides a history of the changes made to the current application. See [View the Change History of an Application](#).



Snapshots are created from the latest published application. Click **Snapshots** to open the Snapshot dialog box where you can create and view the snapshots for this application. Snapshots let you create backups of your applications and view the changes made to an application over time.

Snapshots provide a record of the changes made to an application during the development life cycle. See [Work with Application Snapshots](#).

About Application Sharing and Collaboration

Process provides features for sharing applications among its users. You can control who has access to view or edit applications.

You share applications with other Process users at the space-level. Applications stored in a specific space share the same permissions. Only owners can change permissions at the application-level. Applications are defined as either private or shared. Only the application owner can view or edit private applications. The application owner and other users who have the correct permissions can edit and view shared applications. See [Creating a New Space for Collaboration](#).

View Mode versus Edit Mode

Shared applications have a View mode and an Edit mode, which determines whether you can make changes or not.

- **View (Read-only):** The application is open for viewing only.
- **Edit:** The application is open for editing.

In Edit mode, you can make changes to the application. When an application is in Edit mode, only the user editing the application can make changes because the editor locks the application. Other users with the correct permissions can view the application, but can't make changes.

The application toolbar shows the current mode.

Application Roles

Application roles define who has access to view and make changes to an application. There are three types of application roles defined as follows:

- **Owner:** When you create an application, you're the owner of the application. You can also define other users as the owner of an application. The owner can perform the following task:
 - Edit the application
 - Activate the application
 - Create an application snapshot
 - Share the application with other users
 - Delete the application
- **Editor:** An editor can make changes to an application.

When a user with the Editor role opens an application, the application is in View mode. If no other users are editing the application, the user can switch to Edit mode and begin editing the application.

- **Viewer:** A viewer can view an application, but can't make any changes to it.

Manage Spaces for Sharing and Collaboration

After creating a space in Process, you share the space with other users. Sharing a space also shares all the applications within the space.

To create a space:

1. Go to the Home page, click **Processes**, and then click **Spaces**.
2. Click **Create**.
3. Enter a name for the new space and click **Create**.

Your new space is added to the Spaces list.

To share a space with other users:

1. Click on a space you want to share in the Spaces list. The Edit Space dialog opens.

Edit Space

Name *
Share - HR

Share Space

Specify Users * Select Role
jausten Viewer Share

Space Members

| | | |
|----------|--------|--------|
| CDOYLE | Editor | Remove |
| LTOLSTOY | Editor | Remove |
| JLONDON | Editor | Remove |
| WFAULK | Owner | Remove |

Cancel OK

2. Add users to the space.
 - a. Select a user from the **Specify Users** drop-down list.
Use the search field to find the users you want to add to the space. For example, enter a string in the search field to retrieve a list of all users matching the string.
 - b. Assign a role to the user from the **Select Role** drop-down menu. The role determines the changes a user can make to an application in the space.
 - c. Click **Share**.
 - d. Repeat steps **a-c** to share the space with another user.
 - e. From the Space Members section in the dialog, you can modify a user's role or remove access privileges for a user. Simply select a different role for a user or click **Remove** to stop sharing the space entirely with a particular user.
 - f. Click **OK** to save and close the dialog.

To delete a space and all its applications:

1. Navigate to the Spaces page in **Processes**.
2. Click **Actions**  next to the space you want to delete.
3. Click **Delete**.

The system prompts for confirmation before deleting a space. Be careful. Deleting a space removes the space and all its applications. You can't recover deleted applications.

About QuickStart Apps

Select a QuickStart App to quickly create an application and learn about Process. QuickStart Apps include all the implementation details needed to play and deploy

them. Use these applications as a ready-to-use application to deploy as is or use them as a starting point to adapt the application to fit your business needs.

Oracle provides a set of standard QuickStart Apps. For example, the gallery has QuickStart Apps that you can use to create an application for submitting a travel request, processing a loan application, or getting a business plan approved.

In the gallery, scroll the list and read the brief description of each QuickStart App. Click **More** to display a graphical view and description of the predefined process flow.

The screenshot displays the Oracle QuickStart App Gallery. At the top, it says "Create an app from a QuickStart App or a Sample". Below this, there are six application cards, each with a color-coded icon and a "QUICKSTART" label. The cards are:

- BU Business Plan Approval**: This application contains three levels of approval for a business plan. If any of the three levels reject the plan, it is returned to the owner for resubmission.
- FO Form Approval**: In this application, the process owner submits a request for approval and fulfillment. If the request is denied, the applicant may resubmit with additional...
- LO Loan Application**: This loan application starts with the loan applicant submitting a form. The loan applicant and appraiser upload supporting documentation and then the Loan...
- PA Parallel Request**: In this app request with the process...
- TH Three level approval**: In this app request with the process...
- TR Travel Approval**: An employee submits a travel request for approval, and, if it is approved, the employee is instructed to book the travel. If it is rejected, the employee can...

 The TR card is highlighted with a red border. A red circle highlights the "More" button on the TR card, with a red arrow pointing to a detailed process flow diagram. The diagram shows a process flow between "Employee" and "Approver" roles. The process starts with "Submit Travel Request" (Employee), followed by "Approve Request" (Approver). A decision diamond "Approved?" follows. If "No", it goes to "Book Travel" (Employee) and then "Completed" (Approver). If "Yes", it goes to "Resubmit" (Employee) and then back to "Approve Request" (Approver). A "More Information Required" path also leads from the decision diamond back to "Approve Request".

Developers can create other QuickStart Apps and add them to the gallery. Each QuickStart App will help you rapidly automate your business processes. See [Create Your First Application \(a Quick Start\)](#).

Create a New Application

You can create a new application from a QuickStart App or from scratch.

To create an application:

1. Go to the Process Applications page.
2. Click **Create**, and select one of these options:
 - Click **Start with a QuickStart** to create from a ready-to-use application. Select a QuickStart App from the gallery, click **More** if you want to view the process diagram and additional details, and then click **Create**.

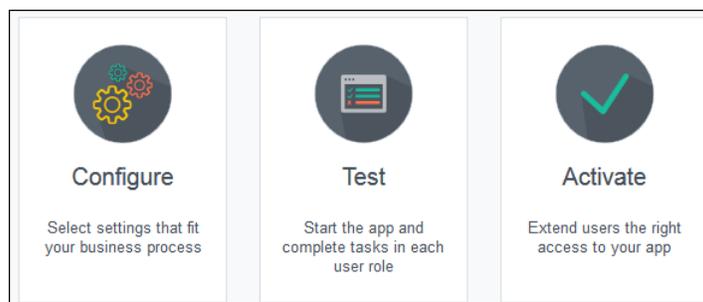
- Click **Create an Application** to create an application from scratch.
3. Enter information into the Create Application dialog box. Here are a few things to note.

| Field | Description |
|-----------------------------|--|
| Name and Description | Make sure the application name and description are useful. Give the user a good idea of what the application is all about and why they might want to use it. Good names and descriptions help users distinguish applications with a similar title or purpose. Also, you can't change the name after the application is created. The first letters of the name are used to create a badge for the application. |
| Space | Select the space where the new application will be stored. All users who are members of the space will have access to the application based on their role. To create a new space, select New Space from the drop-down list. |
| Open Immediately | Most of the time you'll want to leave the Open Immediately check box selected so you can start configuring and testing your application right away. Deselecting the check box is good if you want to create placeholders for several applications at once and modify them at a later time. |

4. Click **Create**.

What happens next and what you do next depends on how you created the application.

- If you created an application from a QuickStart App, the Overview page opens. From here, you can configure the settings to fit your business needs, test each task in the process, and activate your application. And remember, QuickStart Apps are ready to use so you don't need to change any settings. You can skip to test, try out the application, and complete the tasks for each role in the process flow. Adjust any settings if you want, and activate when you're ready.



- If you created an application from scratch, the Application Home tab opens. You can dive right in and start to explore. You'll find smart tools that streamline how you configure settings, model the process, and build your application.

Edit Application Properties

The Information pane displays some general details and status information about the application. It also includes a few tools that let you edit the application's description,

add languages, convert the application, and change whether this application uses the documents feature.

To edit the properties of an application, go to the Process Applications page and open an application. By default, the Information pane is collapsed. To expand the pane, click **show right pane** .

General Properties 



ERApplication 

Created on Jul 13 by...

Add Description

 Jul 14 by...

 Private Application

 Language: English
[Add more languages](#)

 Documents Integration

[Convert to QuickStart Master](#)



Note:

Your changes are saved automatically. So if you make a mistake, just edit the information again.

| Information | Description |
|-------------|--|
| Title | <p>Displays the name of the application, when it was created, and who created it.</p> <p>Click Change Locale Name  to the right of the title to change its localized name.</p> |

| Information | Description |
|-------------------|---|
| Lock / Unlock | Displays the name of user who is currently editing this application and the date and time the application was locked. If the application is shared and no user is currently editing it, then the application is unlocked. You can click the Edit link to lock the application and make changes to it. |
| Type | Displays the application type: Shared or Private . If the application is shared, the number of participants is also displayed. Click the link to see details about the users who can access the application and what their role is. |
| Language | Click Add more languages and use the shuttle controls to add the languages you want to support in this application. Note that this action only adds the selected languages to the current application's set of possible options. After you add languages to the list of options, you can then localize the content. For example, if you want to use additional languages when working with forms, then you must add the languages here first. See Localize Processes . |
| Documents Options | Available only if your administrator integrated Oracle Content Management with Process. After a connection between the two services is established, you can use documents in all your process applications. Use these options to configure whether the documents feature is disabled or enabled for a particular application. See Enable or Disable Documents . |
| Description | Displays the application description, if the optional text has been provided. Descriptions are one or two sentence expansions of the title to help a user distinguish between applications of similar or same titles. You can add or modify a description at any time. |
| Convert Option | Includes a link that lets you convert the current application to a QuickStart App, or vice versa. |

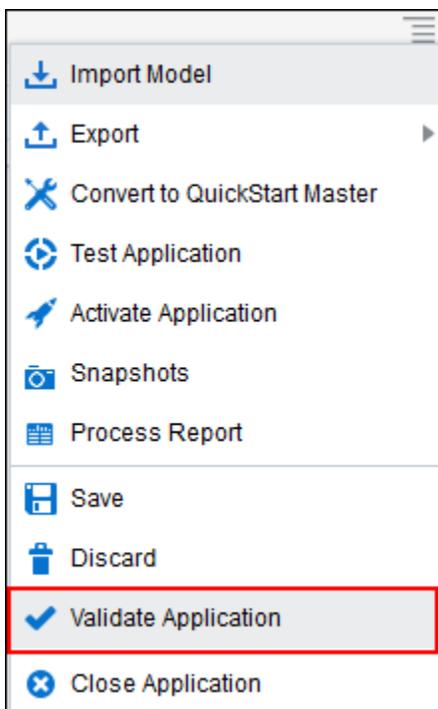
To collapse the Information pane, click **show right pane**  again.

Validate an Application

Validating an application enables you to check your application and processes for errors or warnings. Applications that contain errors can't be deployed.

To validate an application:

1. Go to the Process Applications page.
2. From the Main menu, select **Validate Application** .



3. Click **View** to view the errors based on the following options.

| View | Options |
|-------------|---|
| Scope | <ul style="list-style-type: none"> • Application • Process |
| Severity | <ul style="list-style-type: none"> • Show All • Errors • Warnings • ToDos <p>Note that Errors and Warnings are selected by default.</p> |
| Flow Object | <ul style="list-style-type: none"> • Show All • Activities • Decisions |

4. Click **Export** to export the errors to a Microsoft Office Excel spreadsheet.

Save and Publish Changes to an Application

You can save changes to your application as you're editing application components. You must switch to Edit mode to make changes.

To save changes to your application, click **Save** at any time. All unsaved changes for each application component are saved. You can continue to make changes to the application as necessary. If you're working in a shared application, then the application remains locked and you can continue to edit it.

To make your changes available to other participants, click **Publish**. Published changes are recorded in the Recent Activity pane on the Application Home tab.

To close an application, click **Close**.

If you're sharing an application, then a dialog box prompts for how to handle the current lock on the application. To make the application available for other users to edit, be sure to select the **Release Lock** check box.

 **Caution:**

If the application hasn't been published before you save and release, then it asks for whether you want to publish. If you don't publish before you release, your changes are lost.

While editing application components, you can revert your changes and return to the most recently published version of an application.

To do this, from the Main menu, select **Discard**, and then click **OK**.

 **Note:**

You can't recover changes that you discard.

Play Processes and Test Applications

Use the application player to test the behavior of your business processes at design time. You can test a process using different user IDs without having to activate the Process application.

Topics:

- [Map Process Roles to Users in Your Organization](#)
- [About Testing Processes Using the Application Player](#)
- [Test a Business Process](#)
- [About Testing an Application's Activation](#)
- [Test an Application's Activation](#)
- [Customize Titles](#)

Map Process Roles to Users in Your Organization

Before using the application player, you must map the roles defined in your business process to the users or groups of the organization infrastructure defined in your runtime environment. The application player uses the information of your organization to mimic the behavior of your business processes in real-world situations.

To map roles to users in your organization:

1. Open your application.
2. Start the player either by selecting **Test Application** from the main menu or by clicking the **Player**  icon.
3. Select the role that you want to map from the drop-down list, which displays all the roles defined in your process.

The screenshot shows a web interface for configuring an organization. It has two main sections: 'Add Mappings' and 'Manage Mappings'.

Add Mappings: This section contains a text input field labeled 'Specify Users or Groups' with a 'Choose' button to its right. Below this is a 'Select Role' section with a list box containing 'All' and 'PurchaseRequestWithORDS.ProcessOwner'. There is an 'Add Mapping' button at the bottom of this section.

Manage Mappings: This section features a 'Show Roles:' dropdown menu set to 'All' and a 'Delete' button. Below these is a table with two columns: 'Role' and 'Identity'.

| Role | Identity |
|--|----------|
| PurchaseRequestWithORDS.Process Owner | User PO |
| PurchaseRequestWithORDS.Process Reviewer | User PR |

4. Select the user or group you want to map.
 - a. Click **Choose**.
 - b. Select *User* or *Groups* from the drop-down list.
 - c. Enter the name of the user or group you want to search for, then click **Search**.
To see a list of all users or groups, leave the text area blank, then click **Search**.
 - d. In the table, select the user or group you want to map.
After selecting a user or group, it appears at the bottom of the selected window.
 - e. Click **OK**.
5. Click **Add Mapping**.
The users or groups you mapped to the Process role appear in the mappings table.

 **Note:**

You must map at least one user or group for each role in your process. If the player encounters a user task with an unmapped role, it can't continue running the process beyond the human task.

About Testing Processes Using the Application Player

When testing a business process, the application player deploys a version of the application to runtime using a special runtime partition. This allows the player to run a business process using the same environment as a normally activated application.

The application player provides an efficient way of testing the business processes. It uses a runtime environment, accessible from design time, that emulates the real-world behavior of business processes. As the process runs, the player displays a visual representation of the business process showing the path the process instance follows through the process flow. This allows process designers to easily create, test, and revise business processes without having to save and deploy the application and view it.

As a process instance progresses through a process flow, the player displays an animated view of the behavior of your business process. The outline shows the path the process instance takes through the flow elements and sequence flows of your process. The specific path an instance takes through your process depends on the input data you provide for various flow elements.

 **Note:**

The Instances section on the Application Player tab displays only those instances that are running.

When running the player on a business process, the application is validated and the current version of the application is activated to a player partition of the Process runtime environment. When using the player, you don't have to publish or manually activate the application to view changes while designing a business process.

 **Note:**

Before process modelers can use the application player to test their business processes, an administrator must enable the player. See [Enable the Application Player](#).

Emulating the Runtime Behavior of Flow Elements

As the player runs through a business process, it emulates the runtime behavior of some of the flow elements in your process.

- **Human Tasks**

When the player reaches a human task in a process, it shows the role or user to select on its behalf. It shows all the possible outcomes as actions. If a form is associated with the task, then the player also gives you the option of launching the form or manually selecting the outcome. If you launch the form, the form is activated and displayed in a separate viewer.

If no form is assigned, the player pauses to allow you to select the role you want to perform the task. It prompts you to select one of the outcomes defined for the human task. **Approve** and **Reject** are defined as default outcomes.

However, the list of possible outcomes depends on how outcomes are defined for the human task. After selecting an outcome, the player continues to the next flow element of your business process.

- **Message Send Events and Send Tasks**

When the player reaches a message send event or a send task event within a business process, it performs these events automatically. It then continues to the instance of the process being called and pauses at the corresponding message catch event or receive task.

In both cases, you must manually return to the parent process. For example, if the send and receive pair is creating an instance on a different business process of the same application, then you must return to the Application Player tab, select the new instance for this process, run the child process, and then return to the parent process.

If the send and receive pair calls an external web service, then you must manually enter the required web service message to continue running the process.

- **Timer Events**

When the player reaches a timer event within a business process, it pauses and waits until you click **Run**. The player then moves to the next flow element in the process flow.

- **Call Activities**

When the player reaches a call activity, it calls the child process and creates a new instance of the process. Click the **Drill-Down** icon to view the child process.

- **End Events**

When the player reaches an end event, it pauses and displays the **Drill-Up** icon. Clicking this icon causes the player to return to the parent process. If the current process has no parent, the player returns to the Application Player tab and deletes the process instance.

- **Other Flow Elements**

When the player reaches another flow element that causes the instance to wait for some operation or external event, the player pauses. To continue running the process click **Refresh**, which is located at the top of the Application Player tab.

Test a Business Process

After enabling the application player, you can access the player from the Application Home tab and use it to test the behavior of your business processes. You can access the player from the main menu or the Application toolbar while you're working in Edit mode.

To test the behavior of a business process:

1. Open your process application and access the application player.

You can access the player by selecting **Test Application** from the menu or by clicking **Test** in the toolbar.

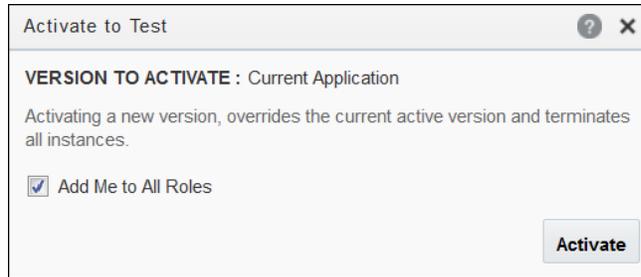
The Test Application tab opens. The application is automatically validated as soon as it's selected.

The screenshot shows the Oracle BPM Studio interface for testing an application. The 'Test Application' tab is active, displaying a 'Current Application' dropdown menu. Below the dropdown, there are two revisions listed: 'Imported Travel Request Application' and 'Initial revision'. A message box on the right indicates 'Current Application validation was successful.' with a green checkmark. At the bottom, there are buttons for 'Play', 'Try in Test Mode', and 'Deactivate'. Below these buttons, there are input fields for 'Active Revision', 'Activation Date', and 'Status'. A list of instructions is provided at the bottom of the interface:

- Select a revision from above and activate it to test
- Manage your active application from here
- Play it or try it in workspace
- Improve the flow and activate it again (only one active version is supported in test)
- Once your application is ready and tested go to management and activate it to a local or remote environment

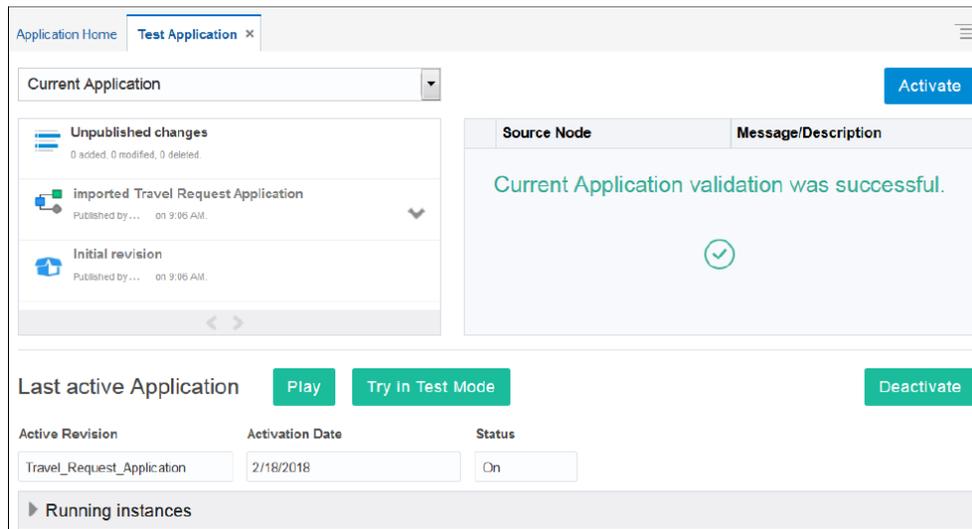
2. Select **Current Application**, **Last Published Version**, or **Snapshot** from the drop-down list.
3. Click **Activate**.

The Activate to Test dialog box opens.



4. Select the **Add Me to All Roles** check box so that you can perform the user tasks and click **Activate**.

A version of the application has now been activated to runtime using a special test partition.



5. Click **Play** and select the business process that you want to test.

The application player begins running the business process. As it passes through each flow element and sequence flow, it outlines the path it takes through the process flow.

As the player continues running through your process, it stops when the process instance reaches one of the following flow elements:

- Form Start Event
- User Task
- Call Activity
- Service Activity
- Message Event
- Timer Event

You must provide input for these types of elements before the player can continue running through the process flow. To emulate the runtime behavior of these flow elements, see [About Testing Processes Using the Application Player](#).

6. If the player pauses on a form start event or user task:

- a. Click **Play** .
- b. Select the user you want to perform the task.

 **Note:**

If the list of users is empty, then verify that you mapped the process roles to your users correctly. See [Mapping Process Roles to Users in Your Organization](#).

- c. Click **Run** .

| Flow Element | Action |
|------------------|--|
| Form Start Event | The form associated with this event is launched. Submit the form. |
| User Task | Select the outcome from the list. The possible outcomes are defined by the human task associated with the current user task. |

The player continues to the next flow element in your process flow.

7. If the player pauses on a message catch event or a receive task, it creates an instance of the child process.

- a. Click **Run**.
- b. Select the **Player** tab.
- c. Go to the **Instances** table and select the newly created instance.

The design-time environment prompts if you want to close the Application Player tab for the original process. Closing this tab has no effect on the process instances.

- d. Click **OK**.

The player opens the new process instance and begins running the business process from the message start event called from the parent business process.

- e. Click **Run** for any flow elements that pause the application player as outlined in previous steps.
- f. When the player reaches the message end event of the child process, click the **Drill-Up** arrow to return to the parent process.

The player closes the tab for this child process and removes the process instance from the list of instances.

- g. From the list of process instances, open the process instance of the parent process.

After reopening the process instance of the parent process, the player continues running through the process from the point where the child process was called.

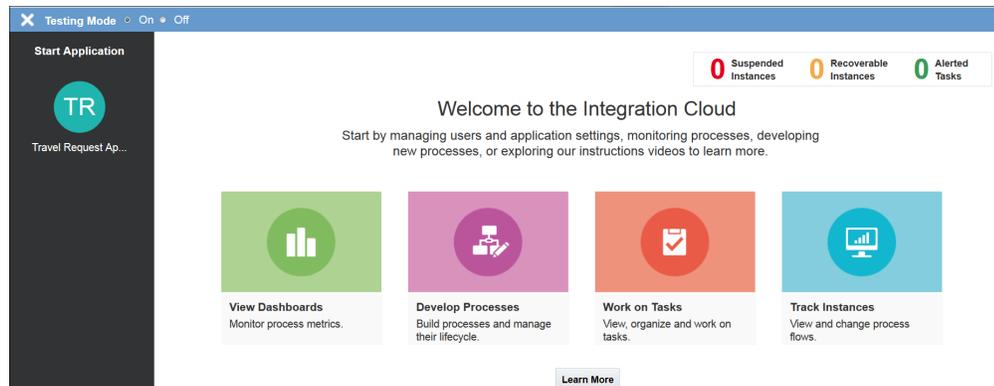
8. When the player reaches an end event in your process, click the **Drill-Up** icon to end the process instance.

The player returns to the Application Player editor and deletes the process instance.

About Testing an Application's Activation

As a developer, you can activate an application from the design-time environment to a test partition in the runtime environment. You can then try out the application and simulate the end user's experience.

Note the **Testing Mode** indicator at the top of the page. When using Testing mode, you can only access the data generated by the applications activated to the test partition. This data is isolated from data in the production environment. You can toggle between Testing mode and Production by selecting either **On** or **Off**.



If you sign out of the runtime environment and want to return to Testing mode, sign in again and then append the following information to the URL:

```
?mode=test&showMode=true
```

Test an Application's Activation

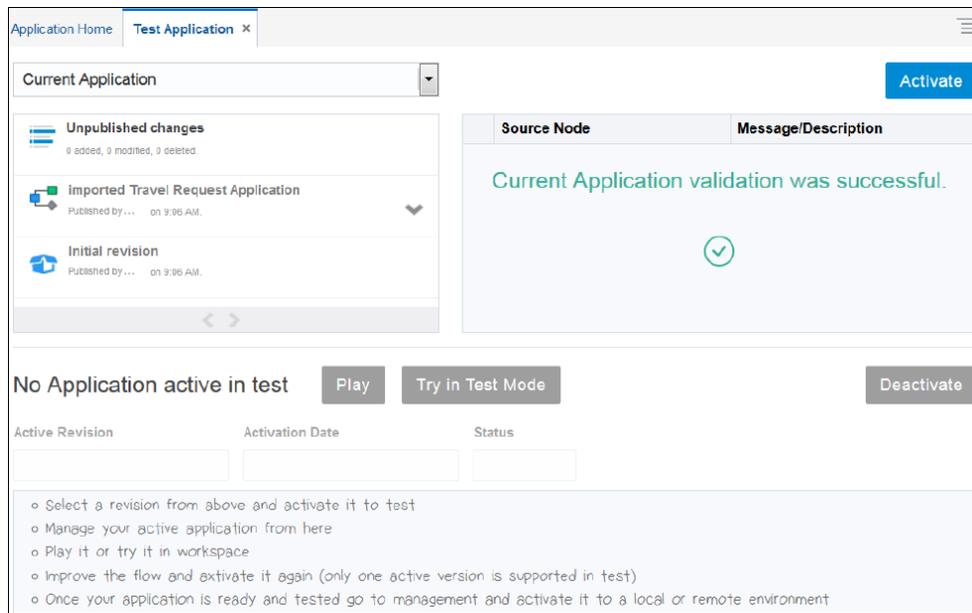
As a developer, you can activate an application to a runtime test environment. You can activate one version of the application at a time.

To perform a test activation:

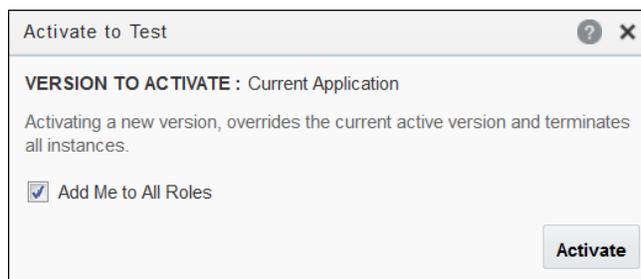
1. Save any changes to your application and publish your application.
2. Click **Test**.

Note:

The application is automatically validated when selected. Validation errors and warnings, if any, are listed on the page. To fix an error, right-click an entry in the list and select **Show in Editor**.

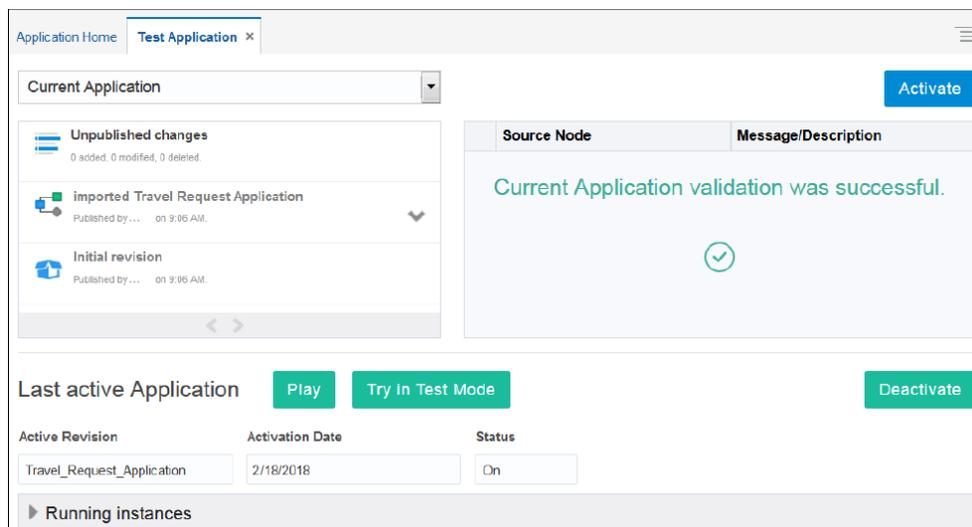


3. Select which version of the application to test. You can test the **Current Application**, the **Last Published Version**, or any available snapshot.
4. Click **Activate**. The Activate to Test dialog box opens.

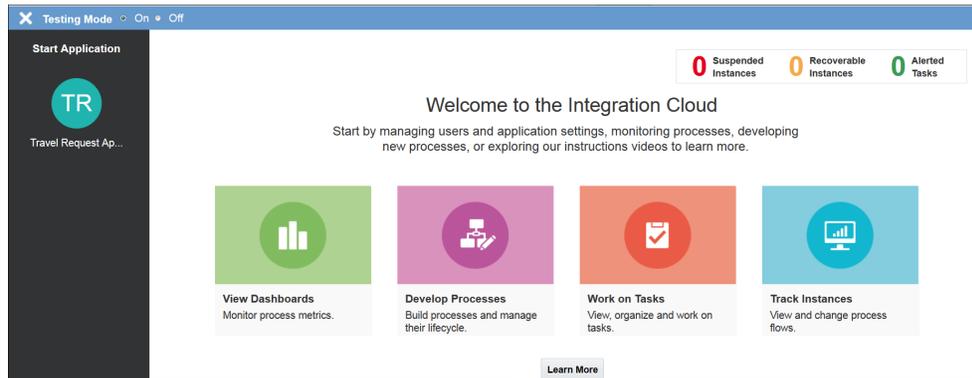


5. Select the **Add Me to All Roles** check box so that you can perform the user tasks.
6. Click **Activate**.

If the application is activated successfully, then the following page is displayed.



- Click **Try in Test Mode** to go to the runtime test environment and test your process as an end user.



The runtime test environment provides the following information:

- List of applications that you can start
- List of process instances related to you
- List of tasks related to you
- Administration actions: Manage Roles and Manage Credentials.

! Important:

Process Developers can only see roles for the processes for which they're the process owner on the Manage Roles page. All other data is protected and available only to administrators.

Only the keys that are created for test data are shown on the Manage Credentials page. Production keys are protected and only available to administrators.

💡 Tip:

If you sign out of the runtime environment and want to return to **Testing** mode, sign in again and append the following information to the URL:
`?mode=test&showMode=true`

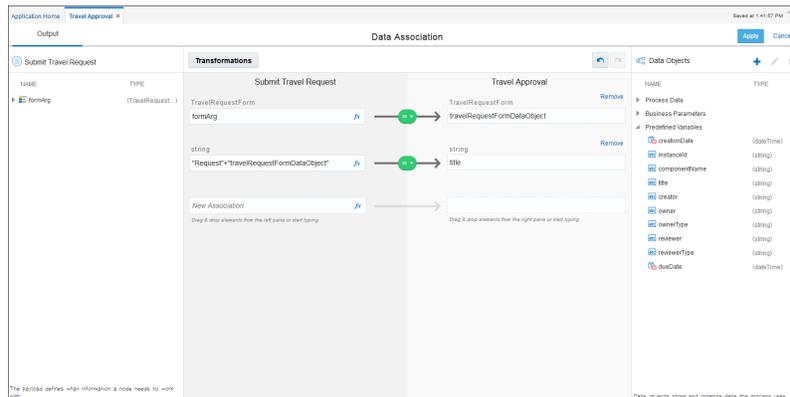
Customize Titles

You can customize the name of the process instances and tasks to ensure a better correlation with the business process. You can customize the title during design time in reference to the business process. Whenever a user submits the request, the customized name can be seen in runtime.

Customize Instance Titles Displayed in Runtime

To customize the title of an instance for display in runtime:

1. Go to the Application Home tab and make sure you're in **Edit** mode if you're editing a shared application.
2. Open the business process where you want to customize.
3. In the process editor, click the **Start Form** activity and then click **Data Association**. See [Configure Data Association](#).
4. On the Data Association page, create a **New Association**.
5. Click the **fx** (expression) icon to create a new expression. The expression should evaluate to a string and should contain a static text and your Form data parameters. See [Work with Expressions](#).
6. In the Data Objects pane, expand **Predefined Variables** and select **title (string)**.



7. Map the **title** data object with the expression you just created, click **Apply**, and then **Activate** the application.
8. Sign in to the Home page, create a new instance, and then go to the **Processes** page. The title of the new instance is customized. See [Track Process Instances](#).

Customize Task Titles Displayed in Runtime

To customize the title of a task for display in runtime:

1. Go to the Application Home tab and make sure you're in **Edit** mode if you're editing a shared application.
2. Open the business process where you want to customize.
3. In the process editor, click the **Human Task** activity and then click **Data Association**. See [Configure Data Association](#).
4. On the Data Association page, create a **New Association**.
5. Click the **fx** (expression) icon to create a new expression. The expression should evaluate to a string and should contain a static text and your Form data parameters. See [Work with Expressions](#).
6. In the Data Objects pane, expand **Predefined Variables** and select **title (string)**.
7. Map the **title** data object with the expression you just created, click **Apply**, and then **Activate** the application.
8. Start the application, create a new task, and then go to the **My Tasks** page. See [Start an Application](#)

The title of the new task is customized. See [Complete Your Assigned Tasks](#).

Alternatively, you can select the **Human Task** activity, click **Open Properties**, and then edit the title using the **Title** field.

Work with Application Snapshots

An application snapshot is a read-only copy of an application at a particular moment. Because snapshots are read-only, you can't open them for editing.

You can:

- Create a snapshot from the last published version of an application
- View the contents of a snapshot
- Export an application based on a snapshot
- Deploy an application based on a snapshot
- Delete a snapshot

Participants with owner or editor permissions can create a snapshot.

On the Recent Activity pane, click **Snapshots** to open the Add Snapshots dialog box.

Click **Add Snapshot** , enter a name and description for your new snapshot, and then click **Add**. The snapshot appears in the list of snapshots defined for this application, including the date the snapshot was created and the user ID of the snapshot creator.

By viewing the contents of a snapshot you can compare previous versions of an application with the current one.

Click **Snapshots**, and then click the name of the snapshot you want to view. Within the snapshot view, you can view the state of the processes, rules, and human tasks associated with the application.

To return to the active version of an application, click its name at the top of the page.



Users with the Editor role can revert back to a previous snapshot, which basically means setting the time that the snapshot was taken as the current time. Snapshots taken after that time are discarded. To revert to a previous snapshot, click **Snapshots**. Right-click and select **Revert to Snapshot** on the snapshot you want to revert back to.

Users with the Editor role can delete snapshots they created. Users with either the Owner role or the Administrator role can delete any snapshot created by any user.

Caution:

You can't recover a deleted application snapshot.

To delete an application snapshot, click **Snapshots**. Select the snapshot you want to delete and click **Delete**.

View the Change History of an Application

To view the history of major published changes made to an application, use the Recent Activity pane. This pane displays these changes, including creating the application, modifying resources, creating processes, and creating human tasks.

Application changes are displayed in the Recent Activity pane. Click **More Details** ▼ next to a specific change to view the details.

Define Application Roles

Use application roles to model the users, groups, or system that performs the work your business process represents. Roles define functional categories that correspond to job functions or responsibilities within your organization. You can create and edit the required roles within your process and assign them to swimlanes.

Several application roles are already defined for you:

- Process Owner
- Process Reviewer
- Analytics Viewer
- Automatic Handler—for tasks automatically handled by the system or by any valid user. This role is already defined for you even though it isn't listed on the Roles tab in the Organization dialog box. However, the Automatic Handler option is available when you assign a role to a swimlane in your process.

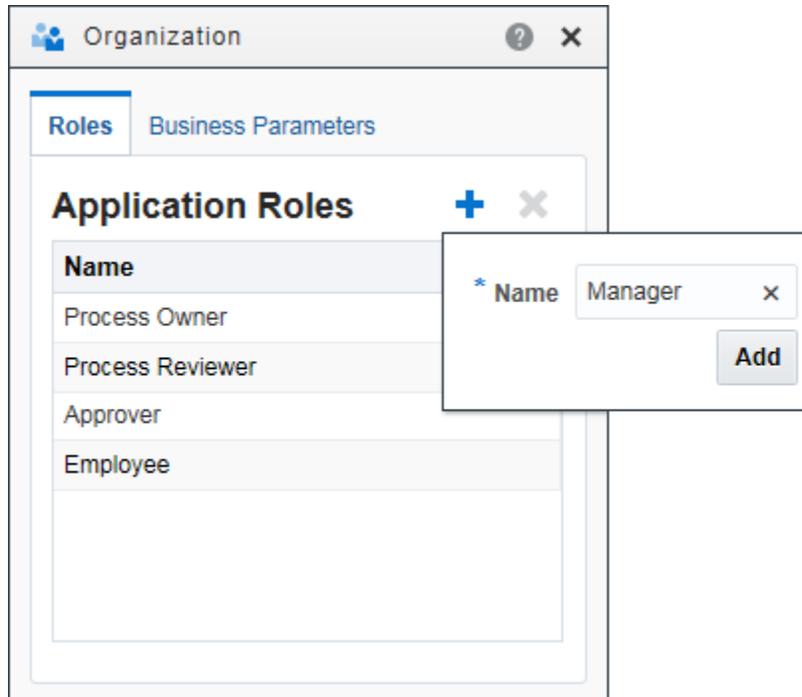
These roles are assigned to the swimlanes in the default pre-defined patterns. You can use these application roles in your processes but they're not required.

You can also [configure roles](#) to create advanced mapping and configuration of application roles, including assigning users to the roles. When an application is activated to runtime, application roles can be mapped to real-world users.

Application roles are defined for the entire application. They can be shared by all the processes in your application. Within a process, roles are assigned to the horizontal swimlanes.

To create an application role:

1. Go to the Application Home tab.
2. On the Recent Activity pane, click **Organization**.
3. Select the **Roles** tab.



4. Click **New +**, provide a name for the role, and click **Add**.

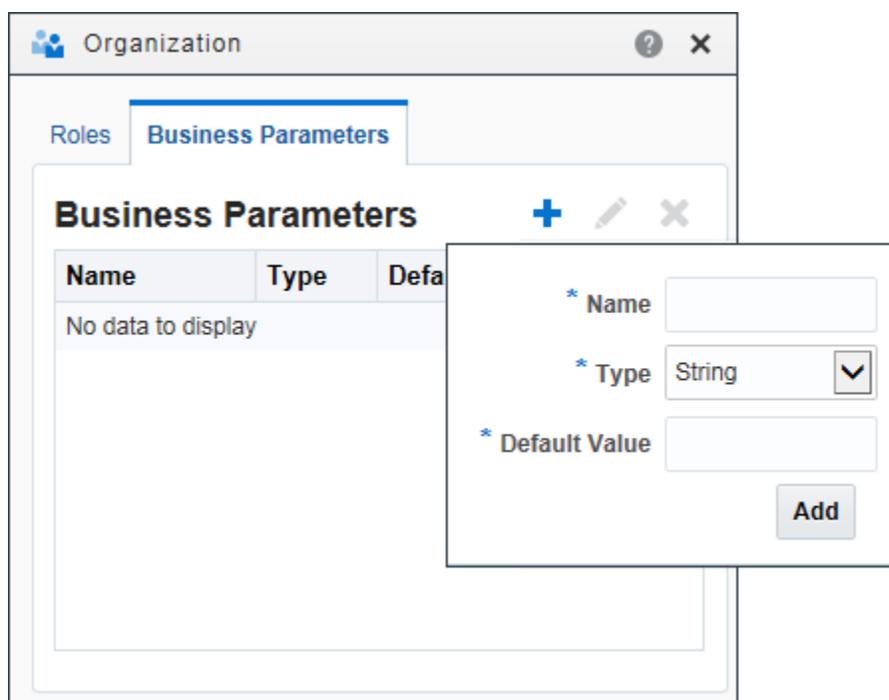
In the Organization dialog box, you can also set a value as a [business parameter](#) to easily modify and use in an application. For example, you might set the interest rate as a business parameter in a mortgage loan application, and use and change as needed.

Define Business Parameters

You can set a value as a business parameter to easily modify and use in an application. For example, you might set the interest rate as a business parameter in a mortgage loan application, and use and change as needed.

To create a business parameter:

1. Go to the Application Home tab.
2. On the Recent Activity pane, click **Organization**.
3. Select the **Business Parameters** tab.



4. Click **New**  to create a business parameter.
5. Enter a name for this parameter, the type of parameter (String, Boolean, Double, or Int), and the default value.
The default value is used if no other value is defined.
6. Click **Add**.

 **Note:**

You can edit the name and default value for an existing parameter at any time. You can't, however, edit the type of parameter. If you need to modify the type, delete the business parameter and then create another parameter with the correct type.

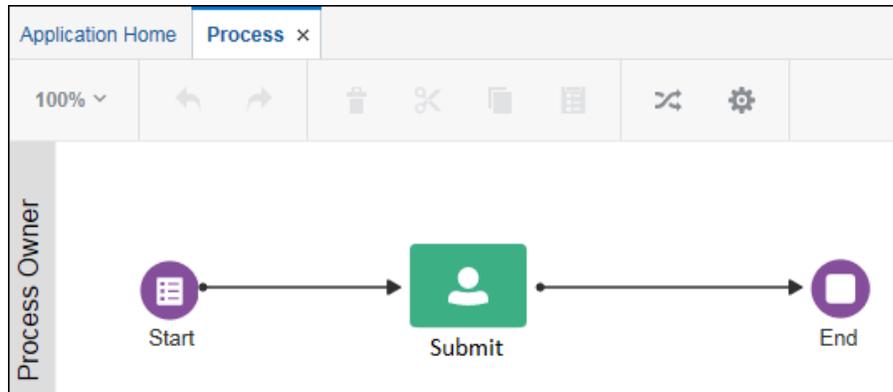
In the Organization dialog box, you can also [define application roles](#). Use application roles to model the users or groups who perform the work your business process represents.

Localize Processes

You can localize process elements and make them adaptable to a selected locale.

To localize processes:

1. Create a process, as shown here.

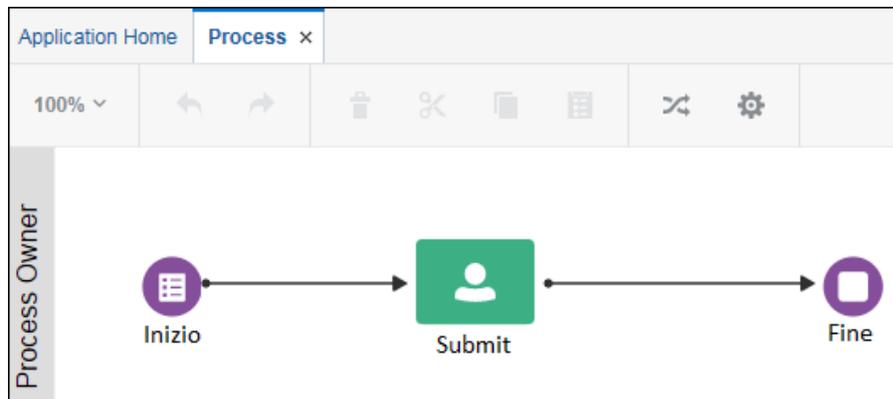


2. On the **Application Home** tab, select the language you want to use from the language options drop-down list.

Return to the process editor page. You'll see that the labels, description, and documentation fields still display in the default language of English.

3. Edit and save the labels and descriptions to ensure that the strings are associated with the selected locale.

For example, after setting the application language to Italian, two process event labels were added and saved, as shown here.



After saving, you can close the process, return to the **Application Home** tab, and reset the language to English. If you reopen the process, the labels are displayed in English.

4. Toggle between the two languages to view the localized process events in the language you select from the **Application Home** tab.

Localize Applications

Apply localization and adapt your applications and their content to a specific locale. You can make your applications usable for a new locale by translating the content from the source language to the language used in the locale.

By default all applications are created in English. You can add language resources (locales) and then localize process and form elements by providing localized strings. When the application is published and viewed by users, the published content displays labels in the localized language based on the users' browser language setting.

You can localize your application components such as process and form elements, human task titles and outcomes from the Localization page. The Localization editor provides a consolidated and editable view of all your default and translated resources. Its interface allows you to manage all your localization resources from one place.

Apart from the Localization editor, you can apply localization to specific application components from within those components. For example, you can also localize form field labels in the Form editor.

See [Localize Web Forms](#).

See [Localize Processes](#).

Topics

- [Add Locale](#)
- [Work with the Localization Editor](#)
- [Add Dynamic Parameters to Task Titles](#)

Add Locale

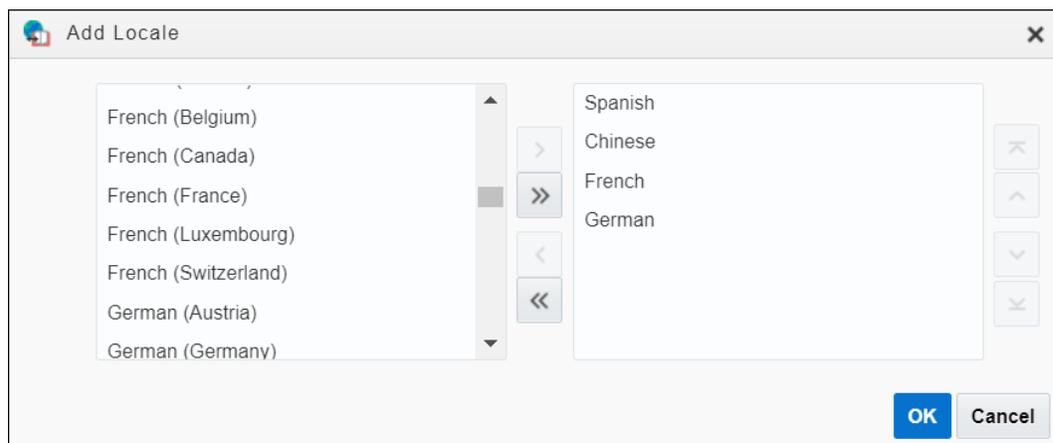
You can add locales to adapt your process applications to specific geographical markets.

1. In the Application home pane, click **Localization**.

The Localization page opens and you see all the available language resources (locales) specified for the application.

Note that initial setting for the locale of any newly created application is English. The initial setting doesn't derive from the browser's locale, nor does adding a language to the application's options change the application's initial language setting.

2. Click **Show/Hide right pane**  (3 dots) to open the General Properties pane.
3. Click the **Add Locales** link.
4. In the Add Locale dialog box use the shuttle controls to add the languages you want to support in the application.



5. Click **OK**.

This action adds the selected languages to the application's set of language resources. Once a language has been added, you can localize content for that language.

Work with the Localization Editor

The localization editor provides a consolidated interface to manage all your default and translated resources.

Click a locale (language resource) in the Localization page, for example English, to open it in the Localization editor. The localization editor's central canvas has the following columns: Component, Element, Property, Default Text, and Translation. There is a search field and a filter field at the top left. You can *filter by component* to display elements related to either Form or Process. Use the search field to find and display a specific element.

| Column | Description |
|--------------|--|
| Component | The component of the application. For example, Form or Process. |
| Element | The element of the particular component. For example, the First Name control of a form. |
| Property | The property of the form or process element. For example, Label of a particular form field or the Title of a process within the application. |
| Default Text | The text that has to be localized. |
| Translation | The localized version of the default text. |

There are four icons (on the top right) that enable you to complete different tasks in the editor.

| Icon | Description |
|---|--|
|  Maximize | Expands the central pane with the five columns (Component, Element, Property, Default Text, and Translation). |
|  Save Changes | Applies and saves the changes done in the Translation column. |
|  Show/Hide Duplicates | Shows or hides duplicate rows. Note that changes done to one row will be applied to all duplicate rows, when they are collapsed. |
|  Show Columns | Displays the columns selected in the drop-down menu. Click the drop-down menu to select the columns that are to be displayed. |

To localize contents of an application:

1. Click **Localization** in the Application home pane.
2. In the Localization page, select a locale in which you want to localize.
If a locale is not available, add it to the page. See [Add Locale](#).
All the application components of the selected locale gets displayed in editor.
3. Enter the translated text in the Translation column of the selected row of the Process or Form component.
4. Click **Save Changes**.

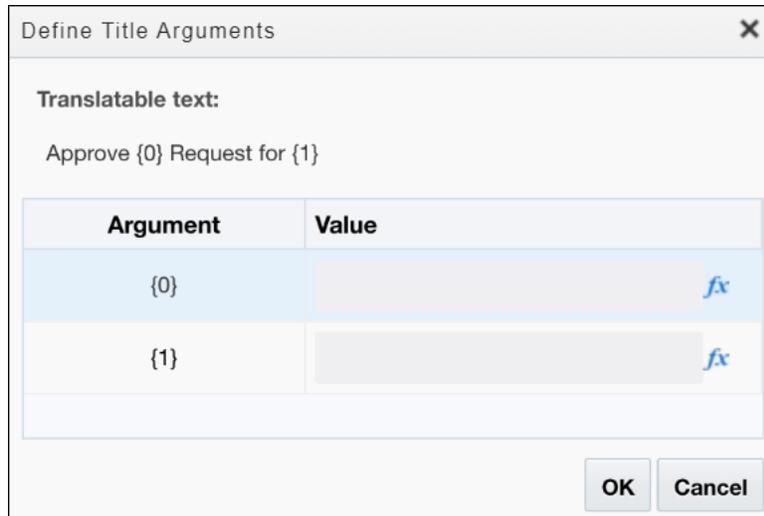
- If you have applied localization to form components, preview the form to see the localized form field labels.

Add Dynamic Parameters to Task Titles

You can add dynamic parameters to task titles and localize them in runtime.

- Select the human task in the process flow. Click the options menu  and then click **Open Properties** from the menu.
- In the implementation pane, add dynamic parameters (for example: {0}, {1}) to the task's title, and then click the editing icon.

- In the Define Title Arguments dialog box, specify the value to be passed. Enter the value directly in the **Value** field.



4. Optionally, click the expression editor icon to open and use the editor to specify an expression. In the Expression Editor, select the expression to insert from Data Objects or Operators. Click **Insert Into Expression**, and then click **Validate** to verify the expression. Click **OK**.
5. Click **OK** to save the arguments and close the Define Title Arguments dialog box.

Import and Export Applications and Snapshots

You can import and export applications and snapshots as EXP files. (Snapshots are exported as normal applications). With this feature, you can share applications and snapshots directly with other users.

Topics:

- [Import an Application from your Local File System](#)
- [Export an Application](#)
- [Export Applications or Snapshots from the Application Home Tab](#)

Import an Application from your Local File System

You can import an application that was previously exported and saved as an EXP file.

To import an application:

1. Go to the Home page, click **Processes**, then click **Create**, select **Import**, and then select **Import Application**.
2. Click **Browse**, then select the application file (.EXP) you want to import.
3. Enter a name for this application.
4. Select the space where you want to store your imported application. Alternatively, you can create a new space for the application.
5. Click **Import**.

Export an Application

Exporting an application to your local file system enables you to share applications.

To export an application to an EXP file:

1. Go to the Home page, click **Processes**, and find the application you want to export.
2. Click **Options** , then select **Download Application**.
3. Select **Save File** and click **OK**.
4. Select a location on your local file system and click **Save**.

Export Applications or Snapshots from the Application Home Tab

Applications or application snapshots in Process can be exported to your local file system.

To export either an application or a snapshot:

1. Open the application or snapshot you want to export.
2. Click **Main menu** , select **Export**, and then select **Last Published Application** or **Export Snapshot**.

The option available is based on whether you open an application or a snapshot. For example, if you open a snapshot, the menu option is **Export Snapshot**.

3. Click **Save File** and then click **OK**.
4. Select a location on your local file system, and then click **Save**.

Your exported application or snapshot is saved as an EXP file on your local file system.

Document Your Applications

Process allows documentation from the application-level, process-level, and activity-level. None of the documentation fields are required; you can document where appropriate.

Topics:

- [Add Process-Level Documentation](#)
- [Add Application-Level Documentation](#)
- [Add Activity-Level Documentation](#)

Add Process-Level Documentation

Process-level documentation is available as process description, process documentation, process-level documentation links, requirements, and process notes.

Process-level documentation can help the end user make decisions because you provide more detail about the process, links, or requirements.

About Process Description

Like the application description, the process description is a one or two sentence expansion of the title. The description helps make a clear distinction between processes with similar titles.

You can enter a process description at the same time that you create a process or you can add, edit, or delete the description from the General tab of the Properties pane.

Where Used

- Detailed Business Process report
- Business Requirements report
- Process Properties report

In a report that contains many processes, the process description can help the reader quickly go to the correct process. The description can explain not only what the process covers, but also what it doesn't cover along with a pointer to another process.

About Process Documentation

Process documentation is rich text documentation that can be added to a process in the Documentation pane. You can access the Documentation pane by clicking the **Restore Pane**  icon located at the bottom right-hand corner of the process editor page.



If the documentation exists elsewhere you should add links so that the information isn't duplicated. If the documentation is more helpful in the activity documentation or description fields, it should be added to where it's most useful.

Note the **Documentation Type** option, which is located at the top right-hand side of the Documentation toolbar. Use the drop-down list to select if you want your documentation to be created for external (**End User**) or **Internal** use. For your documentation to appear in process reports, you must set this option to **End User**.

When Used

Documentation should be added to a process when the information is important for comprehension or following the process, or when the information doesn't exist elsewhere.

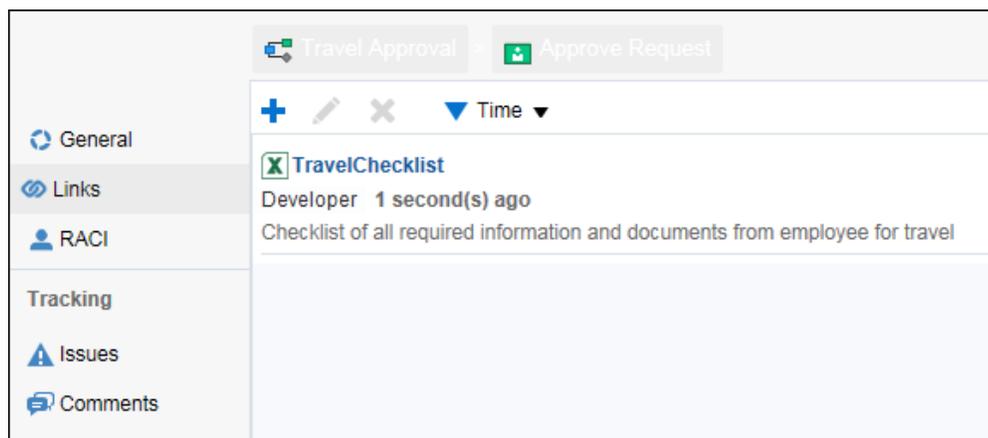
Where Used

- Detailed Business Process report
- Business Requirements report

About Process-Level Documentation Links

Process-level links can be used to link the process to external documentation. However, you must be sure that the link is accessible from Process. If the name of the link is generic or could be confused for another document, use the description to provide more detail. For example, if the sales organization uses several checklists, then the description can help the user determine if the link is to the appropriate

checklist. You can enter a process links description in the Links tab within the Properties pane.



When Used

Process-level documentation links can be added to activities, processes, requirements, activity documentation, and process documentation. The description should be short and expand on the title to help users determine whether the link contains the appropriate document.

Where Used

- Detailed Business Process report
- Business Requirements report

About Requirements

Requirements can be added to a process within the Business Properties pane. There are two plain text fields—notes and solution—where more explanation can be entered. You can also add links to existing documentation.

When Used

The requirements feature tracks the status, priority, and difficulty of the requirements of a process. You can add multiple requirements to a process and they can be sorted by various components, such as date, status, and so on.

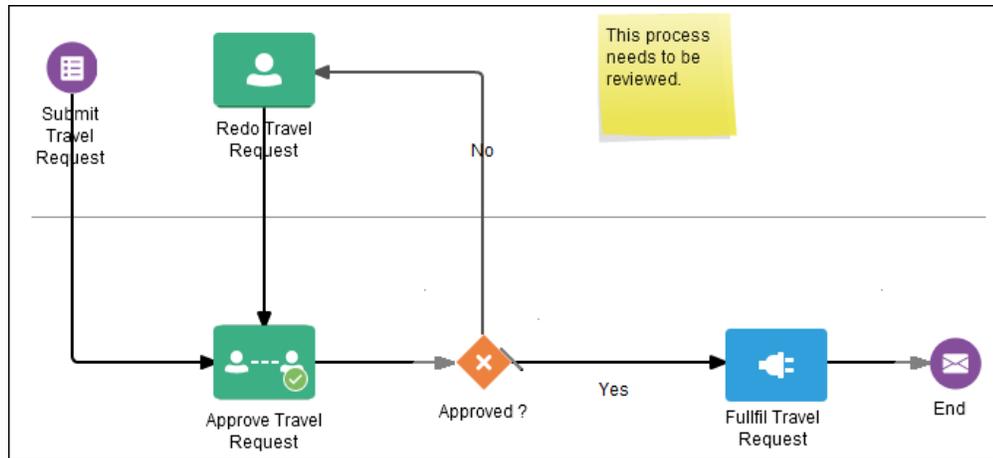
Where Used

- Detailed Business Process report
- Business Requirements report

About Process Notes

A process note is the equivalent of a sticky note—temporary and to be used more as a reminder. They're deleted as soon as the information is used.

To add a process note, drag a Note from the Elements Palette to the process.



When Used

Notes are useful when collaborating with others while creating or editing a process. They're highly visible and can be easily added and removed.

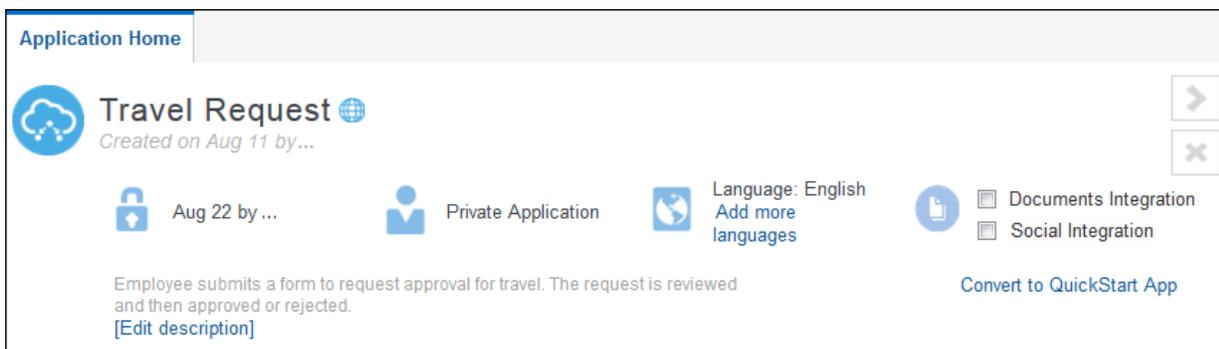
Where Used

Notes don't appear in process reports.

Add Application-Level Documentation

You can create application-level documentation by adding an application description, which is one or two sentences that help you distinguish between applications of similar titles.

You can enter the application description at the same time that you create an application or you can *add*, *edit*, or *delete* the description within an application.



When Used

When you display all applications available in all spaces, one or more applications of the same title may display. For example, in a multi-national corporation, there might be several *Operations* applications for different countries or divisions. The application description helps you select the appropriate application to open.

Where Used

- Detailed Business Process Report

The application description also provides the appropriate context for a report on an application. Reports are often saved and distributed to viewers who don't have access to Process, therefore, it's important to provide detailed information on the application for which the report was created.

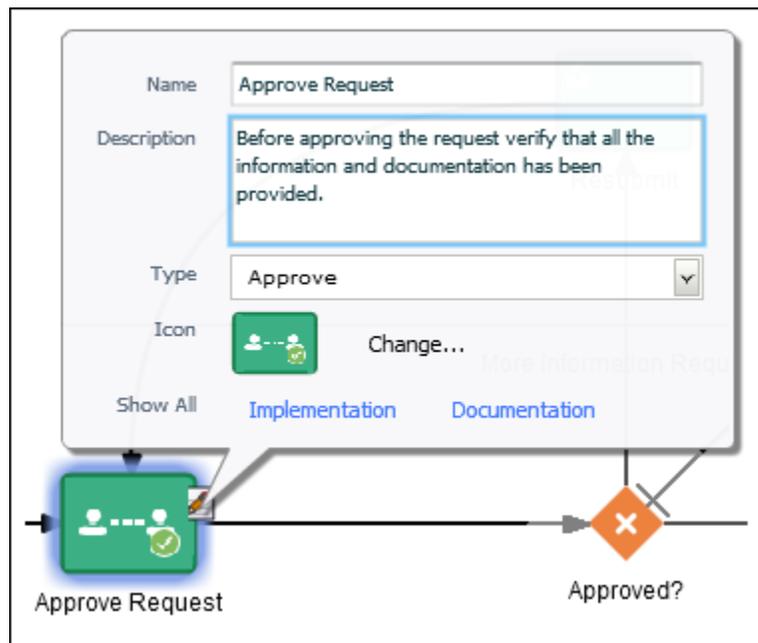
Add Activity-Level Documentation

Activity-level documentation is available as: activity description, activity-level documentation links, activity documentation, activity comments, activity notes, general, issues, and RACI (responsible, accountable, consulted, informed).

You can use activity-level documentation to provide additional information about the activity so that users have a better understanding of the activity, keep records, and determine if the activity is the correct activity when names are similar.

About Activity Description

The activity description should be a brief expansion of the activity name. You can enter the description by editing the activity properties.



When Used

Names of activities tend to be brief. The description is an expansion of the name to help with the understanding.

Where Used

- Detailed Business Process report
- Business Requirements report

About Activity-Level Documentation Links

Links can be added to activities, processes, requirements, activity documentation, and process documentation. The description should be short and expand on the title to help users determine whether the link contains the appropriate document. Activity links are located in the Links tab within the Business Properties pane.

When Used

Customers often have a rich store of existing documentation that can be referenced from the process. You can add multiple links to an activity, and these links are active in the process reports.

Where Used

- Detailed Business Process report
- Business Requirements report

About Activity Documentation

Activity documentation is rich text documentation that can be added to an activity in the Documentation pane. If the documentation exists elsewhere, you should add links so that information isn't duplicated. If the documentation is more explanatory elsewhere, for example, in the description field or as a comment, it should be added there instead.

Note the **Documentation Type** option, which is located at the top right-hand corner of the toolbar. Use the drop-down list to select if you want your documentation to be created for external (*End User*) or *Internal* use. For your documentation to appear in process reports, you must set this option to **End User**.

When Used

Documentation should be added to the activity when the information is important for comprehension or the information doesn't exist elsewhere.

Where Used

- Detailed Business Process report
- Business Requirements report

About Activity Comments

Activity comments can be added in the Comments tab within the Business Properties pane. Once added, a comment can't be edited or deleted.

When Used

Activity comments are like notes in that they're brief, but unlike notes, they're permanently attached to the process and appear in several process reports. You should use comments when its important to keep a record.

Where Used

- Detailed Business Process report
- Business Requirements report
- Issues and Comments report

About Activity Notes

An activity note is the equivalent of a sticky note—temporary and to be used more as a reminder. You can delete them as soon as the information is used. Drag a note from the Elements Palette and drop it on or near the relevant activity.

When Used

Activity notes are useful while collaborating with others when creating or editing a process. They're highly visible and can be easily added and deleted.

Where Used

Activity notes don't appear in any process reports.

About General

The General tab tracks the cost, what application systems the activity interacts with, and time to complete the activity.

When Used

General information is used for process improvement and discovery; it can be important to track the cost of performing the activity as well as how long it takes to complete.

Where Used

- Detailed Business Process report
- Business Requirements report

About Activity Issues

The activity issues should be used to track specific issues including their severity, priority, and resolution status. They can be added in the Issues tab within the Business Properties pane.

When Used

Issues can be used during process improvement, testing, and discovery; it can be important to track the issues that are discovered. Issues can be sorted by severity, priority, resolution status, and date.

Where Used

- Detailed Business Process report
- Business Requirements report
- Issues and Comments report

About RACI

The RACI feature tracks who is responsible, accountable, consulted, and informed regarding an activity. You can add RACI information in the RACI tab within the Business Properties pane. To access, click the relevant activity, click the **Actions** icon, and select **Open Properties**, and then select **Business Properties**.

When Used

The RACI data can be used during process improvement, testing, and discovery to ensure the proper roles are involved in the activity.

Where Used

- RACI report

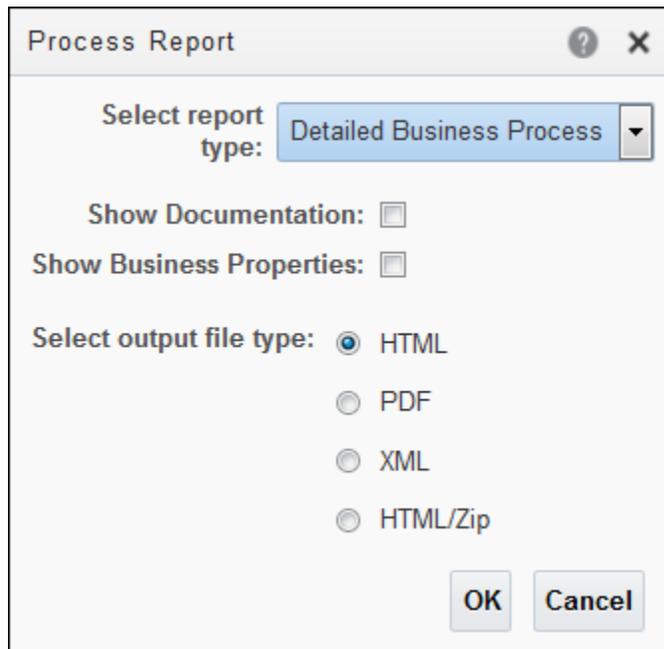
Generate Application Reports

You can generate reports that list each process in your application and show detailed information about each process.

To generate a process report:

1. Go to the Home page and click **Processes**.
2. Open the application that you want to get reports for.

3. Click **Main menu**  and select **Process Report**.



4. Select the report you want to generate. Also, select any optional settings.
 - To include descriptions and other documentation in the report, select the **Show Documentation** check box. The type of documentation that appears on a specific report depends on the report. See [Document Your Applications](#).
 - To include information such as requirements, links to attachments, and tracking of issues and comments, select the **Show Business Properties** check box. Business properties can be at the application-level or the process-level.
 - Be sure to select your preferred file format for the report. The default is HTML.
5. Click **OK**.

6

Develop Structured Processes

A key part of your application is the business process. When developing a structured process, your first step is to determine the people and roles required to complete each task that requires user interaction. You then use the various elements, such as human tasks, system activities, and other events, to design the flow of your process.



Note:

In Oracle Integration, you can create two types of processes: **structured** (described below), and **dynamic**, described in [Develop Dynamic Processes](#).

Topics:

- [About Structured Processes](#)
- [Create a Structured Process](#)
- [Work with Process Roles and Swimlanes](#)
- [Work with Elements](#)
- [Communicate Between Processes](#)
- [Work with Human Tasks](#)

About Structured Processes

A structured process is a sequence of tasks that, after performed, results in a well-defined outcome. A process usually represents work that is performed within the context of a company or organization.

Although applications are higher level wrappers that contain all the resources of a business application, the processes within the application determine how the business application works. Within a structured process, Business Process Model and Notation (BPMN) flow elements define the flow and behavior.



Note:

BPMN is an industry standard notation for defining business processes. Process supports BPMN 2.0.

Business processes are generally created by process analysts who determine the business requirements that must be addressed and define the corresponding process flow.

By default, new structured processes are synchronous. After creating a new process, you can change the type by editing the process.

| Process Type | Description |
|---|--|
| Synchronous Service | Is invoked from another process or service synchronously. In a synchronous service, the calling process waits until the process completes before continuing. |
| Asynchronous Service | Is invoked from another process or service asynchronously. In an asynchronous service, the calling process doesn't wait until the process completes before continuing. |
| Manual Process | Require user interaction. Manual processes can begin with a form, a message, a document, or an empty (none) start event. |
| Reusable Process (Reusable Subprocess) | Can be called by a BPMN process. In BPMN terminology, reusable processes are often called <i>reusable subprocesses</i> . Use the call activity to call reusable subprocesses within your structured process. |

Process Instances

A process instance refers to a specific instance of a process. For example, the Loan Process process example shows the overall definition, or model, of a structured process, including the roles of the process participants who are responsible for performing the work. It defines how a loan application is submitted and approved, and defines the types of people responsible for performing that work. In contrast, a process instance refers to a specific loan application and the specific people responsible for approving it.

The distinction between a process and a process instance is important because Process enables you to model processes, convert them into running business applications, and manage the process instances created within those applications.

Process Tokens

Process token is an abstract concept in BPMN. It refers to the current point of execution within a process. A process can have multiple tokens that indicate that the process is running in multiple paths. For example, gateways are often used to split the path of a process. Splitting a process path creates multiple process tokens.

Flow Elements

Flow elements are the BPMN components that represent the work performed within a structured process.

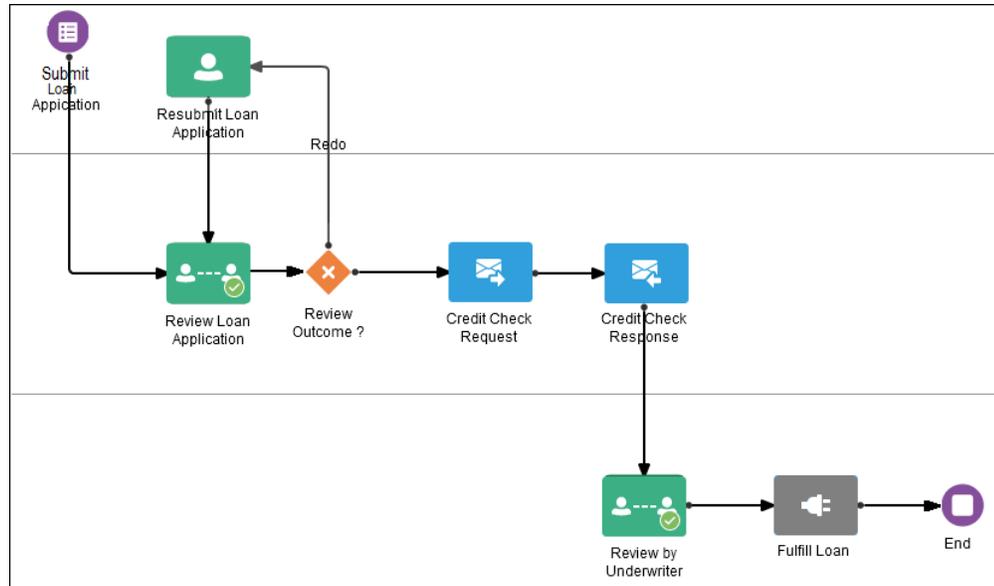
| Flow Element Types | Description |
|--------------------|--|
| Tasks | Represent the work performed by a structured process. |
| Events | Define something that happens during a structured process. |
| Gateways | Determine the flow of your structured process. |
| Sequence Flows | Connect flow elements. |

Data Objects

While flow elements are used to define the behavior of a structured process, data objects are used to define and store the information used by a process. Data objects are variables that are defined during the modeling and implementation of a process. A process instance uses these variables to store specific information. At runtime, the process instance generates and stores specific values for these variables.

Example of a Loan Application Process

Let's look at a sample loan application process. It provides a real-world example of many Process features.

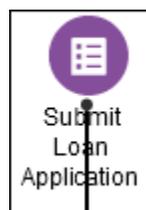


This sample process can be broken down into the following high-level tasks:

- Submit Loan Application
- Determine Application Review
- Check Credit Rating
- Approvals Outcome

Submit Loan Application

The initial form start event is used to trigger a process instance when a user submits a form.



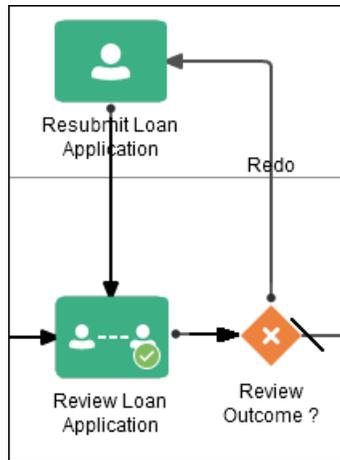
The Submit Loan Application portion performs the following tasks:

- Defines the start point form start event.
- Initiates the process instance when a user submits a form.

The form is specified in the implementation for the form start event.

Determine Application Review

The next set of flow elements determine if the loan application is complete. After the user enters information, the process flow passes the outgoing sequence flow to the approval human task. If the loan application is approved, then the flow passes to the check credit rating service element. If the loan application is rejected, then the flow passes to the re-submit human task for more information from the submitter.



Determining the application review:

1. Determine approval flow (approve task).

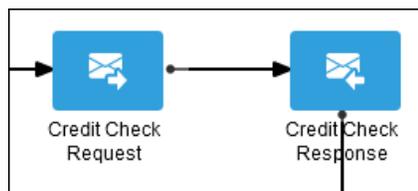
This stage begins with an Approve task, which lets you display a form that the user must review and then perform a certain action. The user might approve or reject the application.

2. Check approval flow (exclusive gateway).

- If *Reject*, resubmit loan application (submit task).
- If *Approve*, proceed directly to the check credit rating stage.

Check Credit Rating

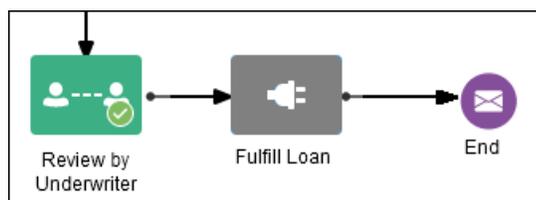
The send task is paired with the Receive task to invoke a process and receive a response in return.



In this example, a message is sent to a process outside of the current process. After this message is sent, the task is complete and the running of the process continues. In contrast to the send task, the receive task waits for a message from the process, which is outside the current process. After this message is received, the task is complete and the running of the process continues to the approvals outcome stage.

Approvals Outcome

The approvals outcome stage represents the final stage of the loan application process. It begins with a review by the underwriter. If the loan is approved, then the process proceeds to the final process flow, which proceeds to the end event.

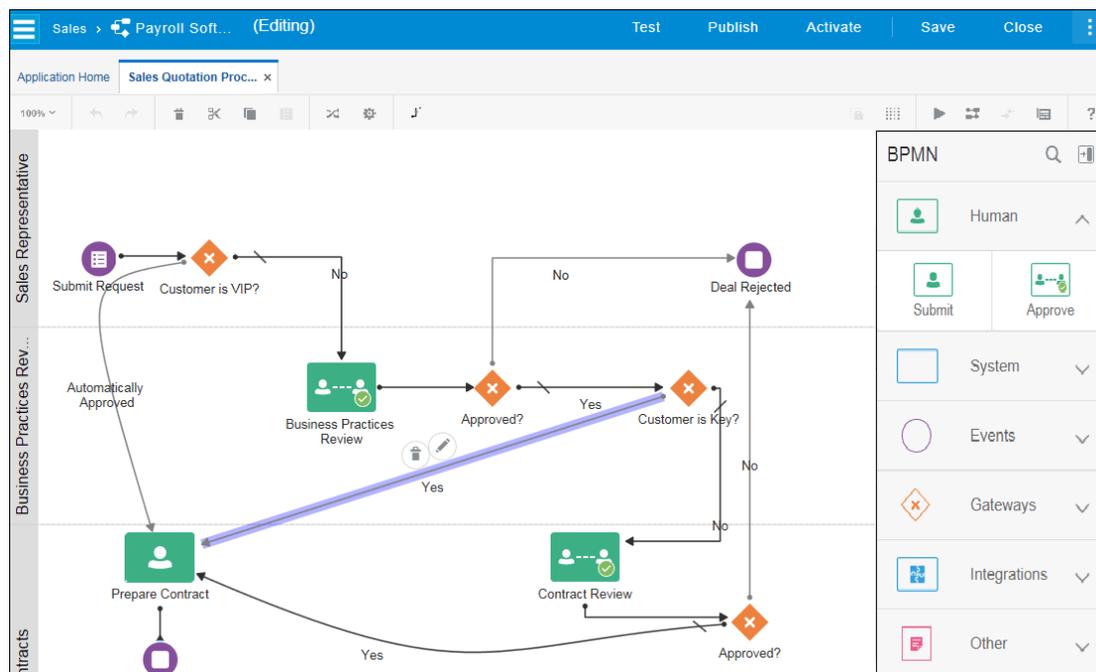


What You Can Do Using the Process Editor

The process editor provides tools for you to easily create and edit business processes.

The process editor opens as a tab and therefore, the toolbar is visible. The options on the toolbar apply to the application as a whole and are available when you're editing a process.

The process editor is divided into three areas: *Toolbar*, *Canvas*, and *Elements Palette*.



Process Editor Toolbar

The process editor toolbar provides quick access to controls related to process design.



| Toolbar Icon | Name | Keyboard Shortcut | Description |
|--------------|----------|-------------------|---|
| | Print | Not available | Print your process using the printer setup of your browser. |
| | Set View | Not available | Set the view size. |
| | Zoom In | Shift + + | Change to a more close-up view of the process flow. |
| | Zoom Out | Shift + - | Change to a more distant view of the process flow. |
| | Undo | Ctrl + Z | Revert to the last change made to your process. |
| | Redo | Ctrl + Y | Reverse the last undo action you performed. |

| Toolbar Icon | Name | Keyboard Shortcut | Description |
|---|----------------|-------------------|---|
|  | Delete | Delete key | Delete the selected items from your process. When you delete a flow element that contains an incoming and outgoing sequence flow, the incoming sequence flow is automatically connected to the outgoing sequence flow. However, you may need to manually re-configure the surrounding flow elements. When you delete a sequence flow from a business process, any implementation details you may have configured for that sequence flow are lost. |
|  | Cut | Ctrl + X | Cut the selected items and copy them to the clipboard. When you cut a flow element that contains an incoming and outgoing sequence flow, the incoming sequence flow is automatically connected to the outgoing sequence flow. However, you may need to manually re-configure the surrounding flow elements. |
|  | Copy | Ctrl + C | Copy the selected items to the clipboard. Note: When you copy a human task, the associated form is not copied. |
|  | Paste | Ctrl + V | Paste the items currently in the clipboard. |
|  | Show/hide Grid | G | Toggle the display of a grid in the process editor window. |
|  | Snap to Grid | S | Center the flow elements in your process on the closest grid axis. Existing flow elements are automatically centered and new flow elements are automatically centered when they're added. Snap to Grid is active only when the grid is shown. |
|  | Find Usages | Not available | Display which other processes within the current application calls your process. |
|  | Play Process | Not available | Test the process using the application player. |
|  | Correlations | Not available | Create correlation keys and properties, and then associate one or more properties with each key. Use the correlations to enable business processes to communicate with each other. |
|  | Change Icon | Not available | Change the icon of the selected item. You can also click it to return to the default. Change Icon is available only when a valid flow element is selected. |
|  | Change Type | Not available | Change the type of the selected item. Change Type is available only when a valid flow element is selected. |

| Toolbar Icon | Name | Keyboard Shortcut | Description |
|---|---------------------------|-------------------|---|
|  | Change Sequence Flow Type | Not available | Change the type of the selected sequence flow. For example, you can change the sequence flow from a straight line to a curved line. Change Sequence Flow Type is available only when a sequence flow element is selected. |
|  | Open Data Objects | D | Define your data. You can create new data objects based on the business types that are defined for your application. |
|  | Open Data Associations | A | Link data to the activity inputs and outputs. |

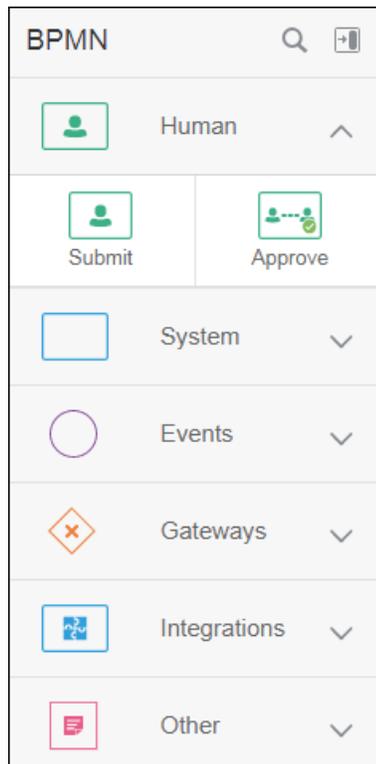
Process Editor Canvas

The process editor canvas is the central area of the process editor window. Use the canvas to draw a diagram that represents your business process using the elements available in the Elements Palette.

The horizontal lines that run across the canvas are swimlanes. Click **New Swimlane**  to add a new swimlane. See [Work with Process Roles and Swimlanes](#).

Elements Palette

Use the Elements Palette to add flow elements and sequence flows to your business process by dragging and dropping elements to the process editor canvas.



Click **Expand**  to expand a specific type of palette item and see all the elements available for that type. Click the icon again to collapse. Click **Show/Hide Palette**  to close and open the palette. See [Work with Elements](#).

Typical Workflow for Creating a Structured Process

Use this typical workflow to create a structured process.

| Task | Description |
|-----------------------------------|--|
| Create a process | Create new business processes from the Application Home tab. |
| Assign roles to the process | Assign roles to swimlanes and determine which members of your business organization are responsible for completing the tasks and activities within the structured process. |
| Design the flow | Drag and drop elements , such as human tasks, system activities, and other events, onto the canvas. |
| Configure flow element properties | Select a flow element in your process diagram, click Menu  , and select Open Properties to open the Implementation pane. Configure properties specific to the flow element. For example, configure a form, service call, or send task and receive task. |
| Define the data | Define the data the activity uses. |
| Associate the data | Link data to the activity inputs and outputs. |

Create a Structured Process

A structured process contains a start event, an end event, and possibly other flow elements. You can start your process from scratch (that is, with an empty start event), with a form, or when a message is received. Alternatively, you can select a pre-defined process pattern to begin creating a new process.

Pre-defined process patterns provide a simplified version of the most commonly-created processes. Because it's easier to edit rather than to create, a pre-defined process supplies you with a shortcut to creating processes, and a way to test other features in Process that require a complete process.

Note:

If you create an application based on a QuickStart App, then a submit and approve structured process is automatically created. You can use this process to either play and activate to a test environment to learn more about the functionality of the design-time environment, or you can use it as a starting point to build your own processes.

To create a structured process:

1. On the **Application Home** tab, click **Processes** in the Components pane.
2. Click **Create a Structured Process**, and then click any option (for example: **Start with a form**) from the list that displays to open the Create Process dialog box.

Optionally, click **Create** and select **New Process** to open the Create Process dialog box.

3. Enter a name for the process.

The field is populated with a default name, such as *Process*, *Process1*, and so on. You can use this name or change it later.

 **Note:**

To rename an existing process, open the process, click **Restore Pane** , click **Business Properties**, select the **General** tab, and then update the process name.

4. Select a pattern from the list. You can start with **None** (empty start event), **Form**, **Message**, or a pre-defined process pattern.
 - If you start your process with a Form, the form isn't created at this point. Instead, you're only setting up the process to expect a form that must be built and associated with the form start event.
 - If you start your process using a pre-defined process pattern, you must still add the implementation details for every flow element that has been automatically added.
 - If you want to start your process with a document or folder, select **None**. You then need to manually adjust the process by replacing the none start event with the **Document Start** or **Folder Start** event. See [Create a Document- or Folder-Initiated Process](#).
5. Review the optional settings and make your changes, if any.
 - In the **Description** field, enter helpful information about this process, what it's all about, and why you would use it.
 - If this process is for descriptive use only, select the **Document-Only** check box.

Business processes often include details that are necessary for the process to run, which can complicate the picture for someone who only wants to understand what the process does at a high-level. A document-only process provides a simplified

process diagram for descriptive use. Document-only processes aren't validated and can coexist with deployable processes in applications.

- If you want to create the process but not edit it right now, deselect the **Open Immediately** check box to return to the **Application Home** tab. For example, you may want to create several processes at once before you edit any of them. You can select and edit your process at any time.

6. Click **Create**.

Delete a Structured Process

You can delete one or more structured processes from your application at any time.

Before you delete a process, make sure the process isn't referenced elsewhere in your application. For example, if the process you want to delete is invoked from another process through a message throw event, then you'll need to re-configure the invoking process to refer to a different process.

An error is displayed during validation if any remaining references to the deleted process exist.

To delete a process:

1. Go to the Process Applications page and open an application.
2. Click **Processes**.
3. Find the process you want to delete and click **Delete** .
4. Click **OK** to confirm you want to delete the process from the application.

Import a Process Created Externally

In Process, you can import external process models created in other programs.

Process supports importing and converting process models in the following formats:

- Visio (vdx)
- XPDL 2.x (xpd)

It may be necessary for you to modify Visio and XPDL processes before conversion to make sure they're converted accurately.



Note:

If the original file contains properties and artifacts that aren't supported, the unsupported elements aren't converted and are omitted from the final process.

To import a process model:

1. On the Application Home tab, select  Import Model from the Main menu  to open the Process Model Converter wizard.
2. Click **Next**.

3. Highlight the format type of the file you want to import and then click **Browse** to find the file.
4. Select one of the following choices if you're importing a Visio or XPDL file:
 - Create a separate model from each pool
 - Merge pools into a single modelThis dialog box opens even if the original file doesn't contain multiple pools.
5. Click **Finish**.

Clone a Structured Process

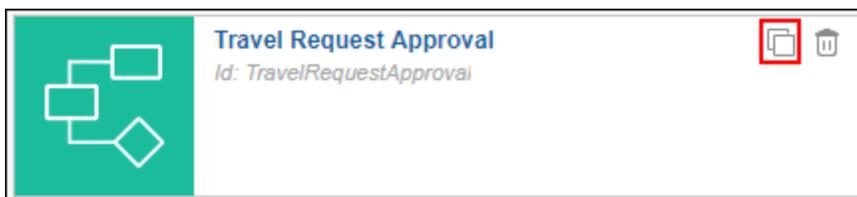
You can clone a structured process to create an exact copy of the process. This is helpful when you need to quickly create and use a replica of an existing process.

When you clone a structured process, the cloned process will have all activities and properties that were configured in the original process. All underlying artifacts (such as forms, business objects, integrations) used by the original process will also be used by the cloned process. Note that the artifacts itself are not cloned; but are just used by both the original and the cloned process.

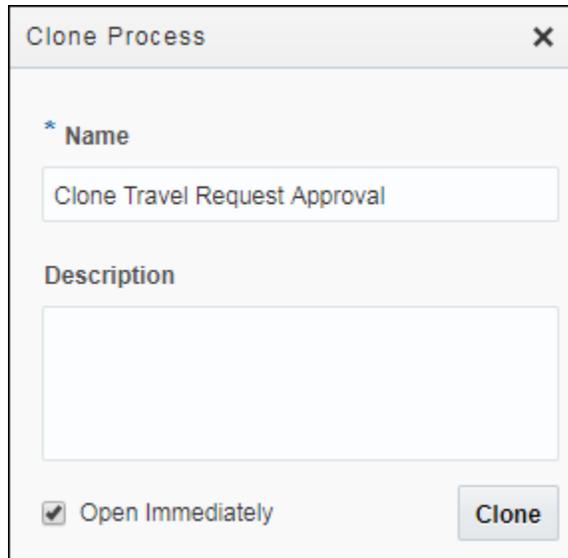
The cloned process will be a standalone process that can be executed and activated independently. You can also modify and customize the cloned process to meet your business needs.

To clone a structured process:

1. Click the **Clone**  icon in the process.



2. In the Clone Process dialog, enter a name in the **Name** field, and a suitable description in the **Description** field.



The screenshot shows a 'Clone Process' dialog box. At the top, there is a title bar with the text 'Clone Process' and a close button (X). Below the title bar, there is a section labeled '* Name' with a text input field containing the text 'Clone Travel Request Approval'. Underneath that is a section labeled 'Description' with a larger, empty text area. At the bottom left, there is a checkbox labeled 'Open Immediately' which is checked. At the bottom right, there is a button labeled 'Clone'.

Note that by default the name of the cloned process will be Clone <original process name>. For example, *Clone Travel Request Approval*. This helps you identify that the process is a clone of the Travel Request Approval process.

3. Click **Clone**.

Work with Process Roles and Swimlanes

Roles are assigned to swimlanes and determine who or what in your business organization is responsible for performing the work of your process-based application.

The key to designing a business process is to determine the people and roles required to complete each task that requires user interaction. Some tasks may be performed by any user or handled automatically by the system. You use swimlanes to group flow elements based on the roles defined within your business process.

Using Roles to Define Responsibilities

Roles are used to decide who or what is responsible for performing the work that is performed within your business processes. Roles allow you to define functional categories that represent job functions or responsibilities within your organization.

The roles defined in your business process are also referred to as logical roles. When your application is deployed to the runtime environment, these roles are mapped to LDAP roles that correspond to the users in your real-world organization.

You can create and edit the required roles responsible for completing activities and tasks within your business process and assign them to the horizontal swimlanes.

As a process analyst, you're responsible for determining what roles are required when designing a business process. For example, for a loan approval business process, you may define the following roles:

- **Loan Officer:** Loan officers are responsible for creating the loan request and forwarding it to the loan processor.
- **Loan Processor:** Loan processors are responsible for reviewing the loan application and checking the credit history of the loan applicant before forwarding the application to the underwriter for approval.

- **Underwriter:** Underwriters are responsible for reviewing and approving the loan application. Additionally, they disburse the loan if it's approved.

Swimlanes and Flow Elements

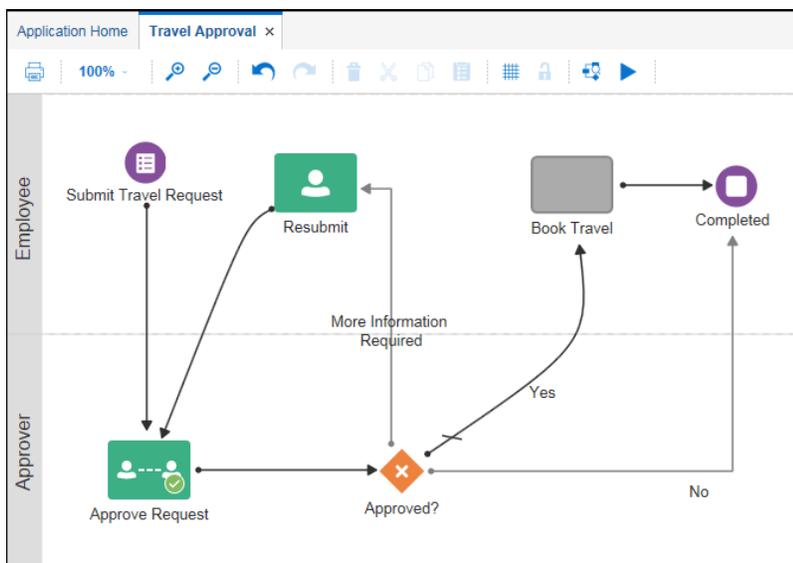
Swimlanes are the horizontal lines that run across the process editor canvas. All flow elements must be placed within a swimlane.

Swimlanes that contain user tasks must have roles assigned to them. Swimlanes visually display the role responsible for performing each flow element within your business process. Additionally, you can have multiple swimlanes that are assigned to the same role. You can also explicitly define a different role for a specific flow element if it needs to be different from the roles assigned to the swimlanes.

Swimlanes can make your business process more readable when you must use the same role in different parts of the same business process.

When you create a new business process, a default swimlane is created. You can add additional swimlanes to your business process as necessary. When adding interactive and manual activities to a business process, you must assign a role to the swimlane.

As shown in the following illustration, the process editor canvas is divided into swimlanes. Each swimlane displays the role of the person responsible for performing the tasks displayed within each swimlane. For example, for a travel request approval business flow, you may have two swimlanes. The *Employee* role is assigned to the first swimlane and the *Approver* role is assigned to the second swimlane. The Approve user tasks appears in the second swimlane so process participants assigned to the *Approver* role are responsible for performing these tasks.



Add and Edit Swimlanes

To add a new swimlane and edit swimlane properties:

1. Open your application and then open the process.
2. Click **Create New Lane**  at the bottom of the page.
3. Click the role name for the swimlane and then click **Edit** .

4. Assign a role to the swimlane.
 - To assign an existing role to the swimlane, select the role from the drop-down list.
 - To assign a new role to the swimlane, click **Add Role**  and enter the name of the new role.
5. Change the background color of the swimlane if you want.

Work with Elements

Use the Elements Palette to add elements to a process by dragging and dropping them onto the process editor canvas. After you add an element, you define its properties, data association, forms, and sequence flows where they apply.

Elements are divided into the following types:

- Human
- System
- Events
- Gateways
- Insight
- Integrations
- Other

Human

Human flow elements represent tasks where a process participant is required to perform the work. The task can be a simple interaction, such as filling out a form, or part of a more complicated workflow that requires input from multiple process participants.

- A **submit task** provides a form or submit action that the user acts on to create a request or provide information about a certain subject.
- An **approve task** provides a form for review or an approve/reject action that the user acts on to approve or reject the request.

See [Add Human Interaction](#).

System

System flow elements allow you to define interactions across business processes. For example, you can use a Service flow element to invoke an external service or process, a Call flow element to call a reusable process from within the current process, or a Send flow element to send a message to a system or process outside the current process.

| System Task Types | Description |
|------------------------------|--|
| Abstract | Use to designate a placeholder for another activity or task. |
| Data Mappers | Use to assign values to data objects and variables within the process. |
| Services | Use to communicate with other processes and services. |

| System Task Types | Description |
|----------------------------------|--|
| Call | Use to call a reusable process from within the current process. |
| Send and Receive | Use to send and receive messages to and from a system or business process outside of the current process. |
| Notify | Use to send an email notification to a user. |
| Dynamic Process | Use to assign a dynamic process within the business process. |
| Bot Activity | Use to integrate external robotic process automation (RPA) applications with your business process. |
| Decisions | Use to incorporate Decision Modeling Notation (DMN) based decision model snapshots within your business process. |
| Micro Process | Use to incorporate micro processes created within different applications in your main business process. |
| Subprocess | Use to group, embed, segment, or reuse processes. |
| Event Subprocess | Use to handle exceptions that occur in the runtime life cycle of a process. |

Events

Event flow elements can be divided into two types:

- Start and End elements that define the starting and ending points of a process
- Intermediate events that can either occur within the typical flow of your process or trigger an interruption with your process

| Event Types | Description |
|---|--|
| Document Start | Use to trigger a process instance when a document is received. |
| Folder Start | Use to trigger a process instance when a folder is received. |
| Start (none) | Use when no instance trigger is defined, such as when a process instance is created by another flow element, or as a placeholder when the start event isn't known. |
| Form Start | Use to trigger a process instance when a user submits a form. |
| Message Start | Use to trigger a process instance when an email message is received. |
| Message Throw and Catch | Use to send and receive messages to and from another business process or service. |
| Timer Catch | Use to control the flow of your business process using a time condition. |
| Error Boundary | Use to handle an error that occurs within a process flow. |
| End Event (none) | Use to mark the end of a process path, or as a placeholder. |
| Message End | Use to send a message to another process or service when the process is completed. |
| Error End | Use when the end of a process is the result of an error condition. |
| Terminate End | Use to immediately stop a process. |

Gateways

Gateway flow elements determine the path a token takes through a process. They define control points within your process by splitting and merging paths.

| Gateway Types | Description |
|-----------------------------|---|
| Exclusive | Splits a process into multiple paths, where process flow continues down only one of the paths. The decision about which path the process should proceed along is based on data-specific conditions. |
| Inclusive | Splits a process into multiple paths, where process flow can continue down multiple paths depending on conditional sequence flow. |
| Parallel | Splits a process into multiple paths, where process flow continues down all paths simultaneously. |
| Event-Based | Splits a process into multiple paths, where process flow continues down only one of the paths. An event-based gateway is similar to an exclusive gateway because both involve one path in the flow. However, for an event-based gateway, decisions about process flow are based on an event taking place rather than a condition being met. |

Insight

Insight flow elements enable you to quickly add integrations to your process. Drag and drop the required [Insight model element](#) to key points in your process application. Select a pertinent milestone at that point, define a process identifier, and define the data to extract. At runtime, you can easily monitor and analyze your business processes in real time using Insight dashboards.

Integrations

Integrations flow elements enable you to quickly add integrations to your process. Drag and drop the required [integration](#) and perform data associations to include a REST connector, web service, or active integration in your process.

Other

Notes are equivalent to sticky notes. They're temporary and you should use them more as a reminder and delete them as soon as the information is used.

See [Add Process-Level Documentation](#).

Add Elements

You can add a flow element to your business process by dragging it from the Elements palette onto the process editor canvas.

To add an element to a process:

1. In the Elements palette, click **Expand** ▼ next to an element type.
2. Click and drag an element onto the process editor canvas where you want to add it.

The cursor displays the icon associated with the element type.

3. Position the cursor at the point in your process where you want to add the flow element.

 **Note:**

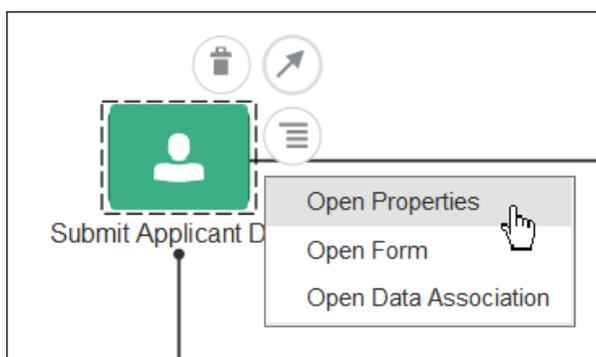
If you position the cursor over a sequence flow, incoming and outgoing sequence flows for the new element are automatically created.

Adding a gateway element, such as an exclusive gateway, requires more steps. See [Creating a Gateway](#).

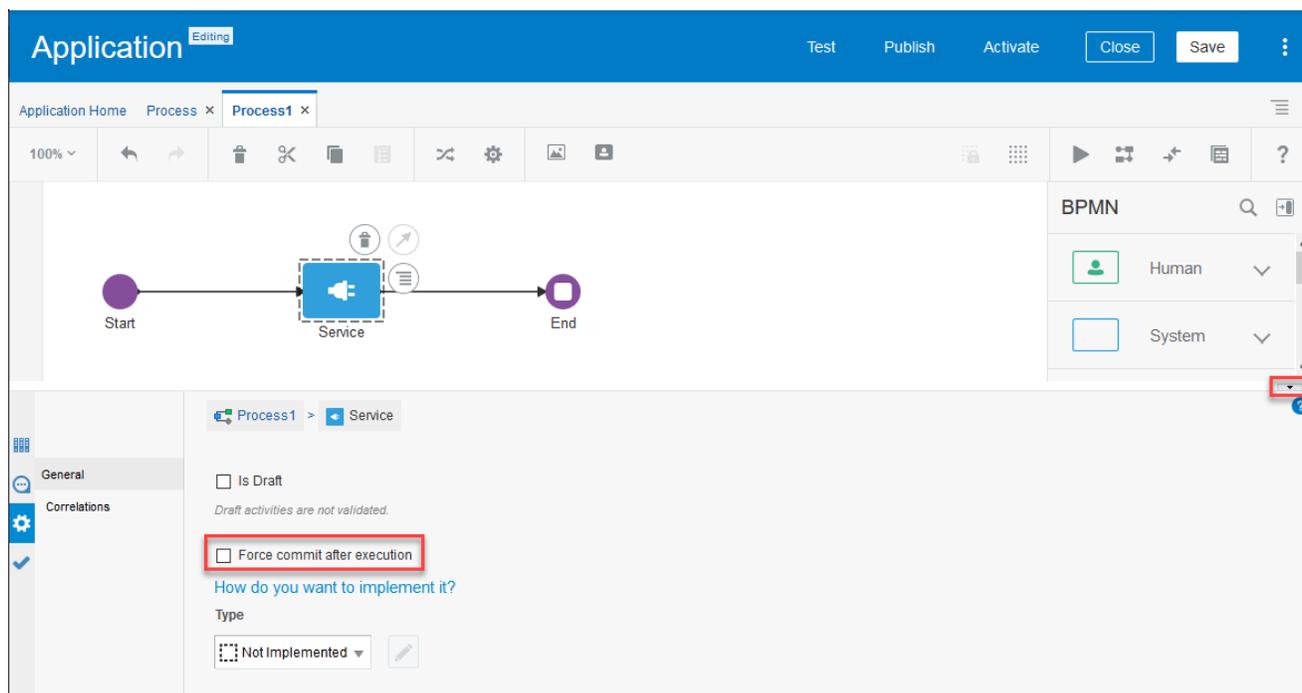
Edit an Element's Properties

You can edit the basic properties of a flow element using the process editor.

In the process editor, select a flow element, click **Menu** , and select **Open Properties**.



The implementation pane expands in the lower portion of the editor to display properties related to the element you selected.



For example, if you select a service, send, decision, and integration activity, a **Force commit after execution** option is available. Selecting the option ensures that the flow is restored to the activity for which it is checked, in case there is a failure afterwards and it retries.

Close the implementation pane by clicking .

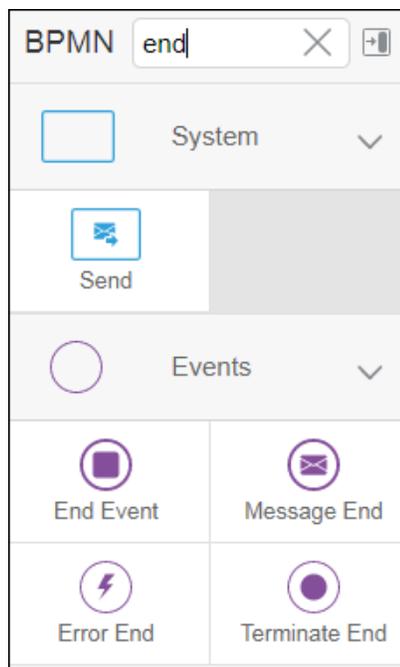
Looking for a keyboard shortcut? After you select a flow element, press **P** to open the element's properties.

Find Elements in the Palette

You can search for an element by entering its name in the **Search** field of the Elements palette.

To search for an element in the palette, enter the name of the element you want to include in your process.

You can also search for an element using a text string in its name. For example, the following search results are displayed when you enter the text string "end" in the **Search** field.



Define Process Start and End

Start events are flow elements that define the starting point of a process. Different types of start events determine how process instances are created. End events, in contrast, define the end point of a process. Different types of end events determine what happens when the process instance is completed.

 **Note:**

In Process, all business processes must have at least one start and one end event.

Because start events define the beginning of a process, they don't have incoming sequence flows. Likewise, end events don't have outgoing sequence flows. However, except for none start and end events, start and end events can have input to and output from other business processes and services.

Specify the Start and End Events for Different Types of Processes

When you create a new business process, a start event and an end event is created by default. The following table shows the default start and end events depending on the type of process you want to create.

| Process Type | Default Start and End Events |
|--------------|-----------------------------------|
| None | None start and end event |
| Form | Form start and None end event |
| Document | Document start and None end event |
| Folder | Folder start and None end event |
| Message | Message start and end event |

To learn more, read about the different types of [business processes](#) supported by Process.

Subprocesses contain none start and end events by default. These are the required start and end events and can't be changed. See [Work with Subprocesses](#).

Use Multiple Start and End Events in a Process

You can define multiple start points in a business process. Multiple start points allow you to specify multiple ways of creating a process instance, depending on which start event is used. Using multiple start events lets you have multiple ways of starting a process without having to create two separate processes.

End events mark the end of a process path. When you have only one end event in your process and the token reaches the end event, the process is stopped.

 **Note:**

Message end events can only be used to terminate processes initiated by a message start event. Additionally, if you have multiple message end events associated with a message start event, each of these message end events must have the same quantity and type of output arguments.

When using multiple end events, it's possible for different tokens to take different paths within a process. In typical cases, all parallel paths must reach an end event before the process is completed.

However, in the following special cases, a process instance can be stopped before all process paths have completed:

- **Error end event:** When an [error end event](#) is reached, all process activity is stopped. Like the error throw event, the error end event stops the flow of a process.
- **Terminate end event:** The terminate end event causes all work on a process to stop immediately. No error handling or other cleanup of the running process is performed.

About the None Start Event

Use the none start event when no instance trigger is specifically defined. The none start event can be used as a placeholder when the required start event of a process is unknown, not yet defined, or implemented later by process developers.



None start events also specify the beginning of a process where the process instance is created by another flow element. In general, the none start event doesn't trigger a new process instance.

However, when used with a receive task, the none start event can trigger a new process instance. The receive task must have the `Create Instance` property set to `True`.

Similar to other start events, the none start event can't have incoming sequence flows. It can only have default out-going sequence flows.



Note:

None start events are always used to define the beginning of subprocesses.

The none start event doesn't accept process input arguments.

About the Form Start Event

The form start event triggers a process instance when a user submits a form. The form is specified in the implementation for the form start event.



The form is designed to get input from the user and present information relevant to the workflow. As you build the form, a business object is created to store the form data.

About the Document Start Event

The document start event triggers a process instance when document details are received.



The document start event has a defined list of parameters, such as the document ID, type (where “d” stands for document), and document name, that you save in data objects to read the document metadata.

```
"id": "document-id",  
"type": "d",  
"docName": "document-name",  
"startedBy": "user Id",  
"role": "contributor"
```

See [Create a Document- or Folder-Initiated Process](#). You can also use REST APIs to [instantiate a process instance](#) and provide all input values.

Similar to other start events, the document start event can't have incoming sequence flows. Document Start events require a default outgoing sequence flow.

About the Folder Start Event

The folder start event triggers a process instance when folder details are received.



The folder start event has a defined list of parameters, such as the folder ID, type (where “f” stands for folder), and folder name, that you save in data objects to read the folder metadata.

```
"id": "folder-id",  
"type": "f",  
"docName": "folder-name",  
"startedBy": "user Id",  
"role": "contributor"
```

See [Create a Document- or Folder-Initiated Process](#). You then use REST APIs to [instantiate a process instance](#) and provide all input values.

Similar to other start events, the folder start event can't have incoming sequence flows. Folder start events require a default outgoing sequence flow.

About the Message Start Event

The message start event triggers a process instance when a message is received. This message can be sent from another business process or from a service.



Messages are types of data used to exchange information between processes. Just as data objects are used to define the data used within an application, messages are used to define the data used between processes or between a process and a service.

Similar to other start events, the message start event can't have incoming sequence flows. Message start events require a default outgoing sequence flow.

You can expose a business process as a service that allows other processes and applications to invoke the process. To expose a process as a service, your process must begin with a message start event. Additionally, you must define the input arguments to the process, which are the data objects passed to the message start event.

The message start event allows you to specify input and output arguments to a process. These arguments define the message that other processes or services must send to the process during invocation. See [Defining Input or Output Arguments](#).

For a business use case, you can invoke a process from any WebService client, such as a SOAP user interface.

To invoke a process using a WSDL:

1. Go to the Home page, click **Processes**, and create an application using the **Start Message Event**.
2. Activate the applications and go to the **Management** page.
3. From the list of activated application, select the application you just deployed.
4. Click **Actions** and select **WebService**.
5. Copy the **url** displayed in the **Exposed Webservice** tab.
6. Open a third party WebService client, for example, the SOAP UI,
7. Create a new project, enter a **Project Name** and enter the copied **url** in the **Initial WSDL** field, and then click **OK**.
8. Click **Start Operations** and create a basic authorization.
9. Enter the **Username** and **Password** for authorization.
10. Click the **Play** button on the top.

The process is invoked.

About the None End Event

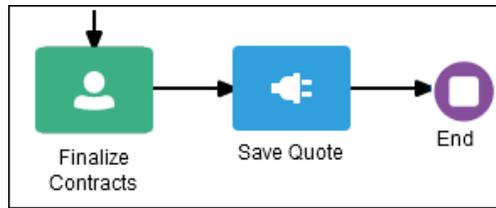
The none end event marks the end of a process path. When a token reaches a none end event, it's consumed. If there are no other tokens within the process instance, then the instance is complete.



Use the none end event:

- When your process isn't required to perform any action after it completes
- To always mark the end of a subprocess
- As a placeholder to be changed later during implementation by a process developer

For example, in a Sales Quote business process, the Sales Quote service task saves information about the sales quote to a database. Because no other work can be performed when the token reaches the end of a process, a none end event is used. After all process tokens reach the none end event, the process instance completes.



About the Message End Event

Use the message end event to send a message to another process or service when the process is completed.



You always use the message end event with either a message start event or message catch event.

When creating a process that has multiple end events, make sure that any tokens that reach a message end event were created by a message start event. For example, you can't use a message end event to end a process instance initiated by a form start event.

Want to learn more about how to configure output arguments using message end events? See [Defining Input or Output Arguments](#).

About the Error End Event

Use the error end event when the end of a process is the result of an error condition.



You typically use the error end event with the error boundary event. The error boundary event is used to change the process flow based on a specific error. This flow usually ends with an error end event. See [About the Error Boundary Event](#).

About the Terminate End Event

Use the terminate end event to immediately stop a process.



The terminate end event causes all work on a process to stop immediately. No error handling or other cleanup of the running process is performed.

Add Human Interaction

Use human tasks to model the user interaction with the application. In Process, process participants interact with your business application during runtime.

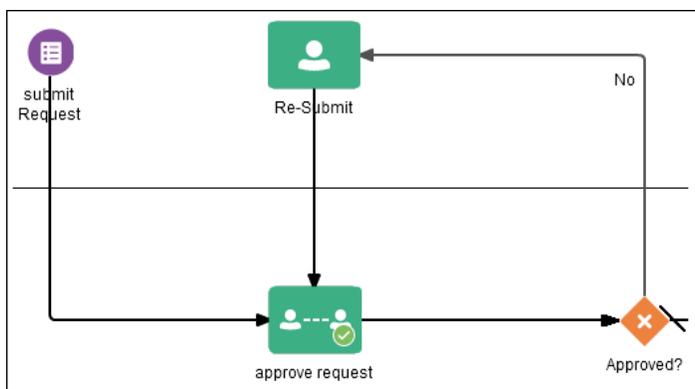
The human task represents a part of your process where a process participant is required to perform an action. The task can be a simple interaction, such as entering a form, or part of a more complicated workflow that requires input from multiple process participants.

After you create the human task, you can configure implementation details such as its assignment, priority, due date, reminders, and deadlines. Human tasks may also contain incoming and default outgoing sequence flows.

Different types of human tasks let you model different types of interactions.

| Human Task | Description |
|--|--|
|  Submit Task | Lets you display a form that the user must submit to create a request or to provide information about a certain subject. |
|  Approve Task | Lets you display a form that the user must review or complete, and then perform a certain action. The user might approve or reject the request. You can also define custom actions for the user to perform. Approval tasks lets you define an approval pattern. Generally, you use the outcome of the approval task to drive the rest of the process flow. |

When a token reaches a human task, the corresponding task is performed. The token waits until the human task is completed before continuing to the next flow element. For example, look at the following process where the loan request is submitted using the form start event:

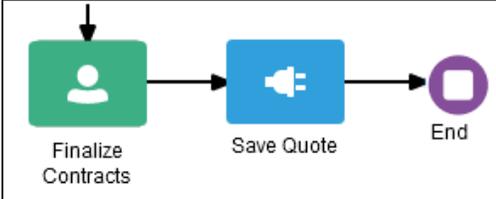


After the user enters information into the form, the process flow passes the outgoing sequence flow to the loan approval human task. If the loan is approved, then the flow passes to the fulfill human task. If the loan application is rejected, then the flow passes to the re-submit human task for more information from the applicant.

See [Work with Human Tasks](#).

Connect to Other Processes and Services

More likely than not, you'll need to define interactions across business processes within a process-oriented application. For example, your business process may need to invoke an external service, call another process from within the current process, or pass a message to a process outside of the current process. Use these system and event flow elements to model communication between processes and services.

| Flow Element | Description |
|---|---|
|  Service Task | <p>Use the service task to communicate with other processes and services. Add the service task when you know that your business process must invoke an external service or process.</p> <p>When the service task invokes a process or service, the token waits at the service task until a response is returned. After the response is received, the token continues to the next sequence flow in the process.</p> <p>For example, the following process uses the service task to save the finalized sales quote to a database.</p> |
| |  <pre>graph LR; Start(()) --> Finalize[Finalize Contracts]; Finalize --> SaveQuote[Save Quote]; SaveQuote --> End((End));</pre> |
|  Notify Task icon | <p>Use the notify task to generate and send notifications. The notify task, which is similar to the service task, uses a predefined service to perform notifications. You use expressions to determine which users receive the notifications generated by the notify task.</p> <p>Currently, email is the only type of notification supported in Process. This type of notification sends an email to the users you specify.</p> |
|  Call Activity | <p>Use the call activity to call a reusable process from within the current process. The process being called becomes a child process of the calling process.</p> <p>When calling a reusable process, the call activity of the parent process waits until the child process completes before continuing.</p> <p>Data objects of the parent process aren't automatically available to the reusable process. Data objects must be passed to and from the child process using argument mapping of the call activity. See About Reusable Processes (Reusable Subprocesses).</p> |

| Flow Element | Description |
|---|---|
|  <p>Send and Receive Tasks</p> | <p>Use the send task to pass a message to a system or business process outside of the current process. After this message is sent, the task is complete and the running of the business process continues to the next task in the process flow.</p> <p>You frequently pair the send task with the receive task to invoke a business process or service, and then receive a response in return. The receive task waits for a message from a system or business process outside the current business process. After this message is received, the task is complete and the running of the business process continues to the next task in the process flow.</p> <p>You can also use the receive task to trigger the start of a business process. This approach is useful when you want to invoke a process from another business process using a send task.</p> <p>To start a business process using the receive task, the following conditions must be met:</p> <ul style="list-style-type: none"> • The receive task is preceded by a none start event. • Your business process doesn't contain any other start events. • The <code>Create Instance</code> property is enabled. <p>The send and receive tasks invoke business processes and services asynchronously. If you need to invoke a business process or service synchronously, then use the service task instead.</p> <p>The send and receive tasks perform functions similar to the message throw and catch events. However, you can't use the send task to invoke a business process that is initiated with a message start event.</p> <p>See Using Send and Receive to Communicate Between Processes.</p> |
|  <p>Message Throw and Catch Events</p> | <p>Use the message throw event to send a message to another business process or service.</p> <p>First, you add a message throw event to a business process to define where that business process must invoke another business process or service. You must also implement the connectivity with other business processes, and create and implement the services invoked by the message throw event. See Communicate Between Processes.</p> <p>Message throw events are frequently used with message catch events to receive a response from the business process or service invoked. After the message throw event sends a message to another business process or service, the token immediately moves to the next flow element of the business process. See Use Message Throw and Catch to Communicate Between Processes.</p> <p>The message throw and catch events invoke business processes and services asynchronously. If your process needs to receive a response synchronously, then use the service task instead.</p> <p>You can use a message throw event to invoke other business processes by calling the message start event of another business process. See About the Message Start Event.</p> |

Use Insight Models in Processes

Integrate with Insight models using the **Insight** element.

The **Insight** element provides you an option to associate points in your process application with milestones defined in an Insight model.

After you link an Insight model to a process application, you can drag the corresponding Insight element in the structured process editor to key points in the process application flow. For each element inserted into the process application, you select the pertinent milestone at that point, define an identifier to correlate the activities

for each instance of the process, and define the data to extract. At runtime, you can easily monitor and analyze your business processes in real time using Insight dashboards that reflect the data you choose to extract, and react quickly to business demands and problems. For example, dashboards can generate graphical visualizations of how many orders have been received, how many had discounts approved or rejected, the details about a single order, or where failures occur in a business transaction.



Want to know more about how to integrate Insight models within your process? See [Work with Insight Models](#).

Use Integrations in Processes

Integrate with other applications and services using the **Integrations** element.

The **Integrations** element provides you an option to quickly integrate your process application with other applications and services.

When you create a REST connector or a web service, or use an active integration in your process, you can specify whether to hide or display these connectors, web services, or integrations in the Elements Palette. If you choose to display them in the palette, these connectors and services show up within the Integrations flow element. You can drag and drop any of these connectors and services and use them within your process. You'll only need to configure data associations for these elements within your process.



Want to know more about how to integrate other applications and services within your process? See [Work with Integrations](#) and [Create REST and Web Service Connectors](#).

Control Process Flow with Gateways

Gateways are flow elements that define the flow of your business process. Gateways determine the path a token takes through a business process. They define control points within your business process by splitting and merging paths. When possible, gateways are used for paths that are exceptions to, or deviate from, the default path of the business process.

Topics:

- [Exclusive Gateway: Take Only One Path \(Data Condition\)](#)
- [Inclusive Gateway: Take One or More Paths](#)
- [Parallel Gateway: Take All Paths Simultaneously](#)
- [Event-Based Gateway: Take Only One Path \(Event Occurrence\)](#)
- [Create a Gateway](#)

Exclusive Gateway: Take Only One Path (Data Condition)

The exclusive gateway lets you split your business process into two or more paths. However, the business process only continues down one of the paths even if multiple outgoing

sequence flows are present. Exclusive gateways can have conditional outgoing sequence flows and must have at least one default outgoing sequence flow.

You can define expressions that are used to determine if your process continues down a conditional sequence flow. See [Work with Expressions](#).

Evaluating the Flows of an Exclusive Gateway

When a process token reaches an exclusive gateway, the outgoing sequence flows are evaluated until one of them evaluates to *True*.



You can define the order in which the flows are evaluated by configuring the properties for the gateway.

- If only one outgoing sequence flow evaluates to *True*, then the process token continues down that outgoing sequence flow to the next flow element.
- If more than one outgoing sequence flow evaluates to *True*, then the process token continues down the first sequence flow according to the order you defined in the gateway properties.
- If none of the outgoing sequence flows evaluates to *True*, then the process token moves down the default outgoing sequence flow. Therefore, you must define a default outgoing sequence flow for the exclusive gateway.

Unlike other gateways, the exclusive gateway doesn't require a corresponding merge to be explicitly defined in your process after splitting.



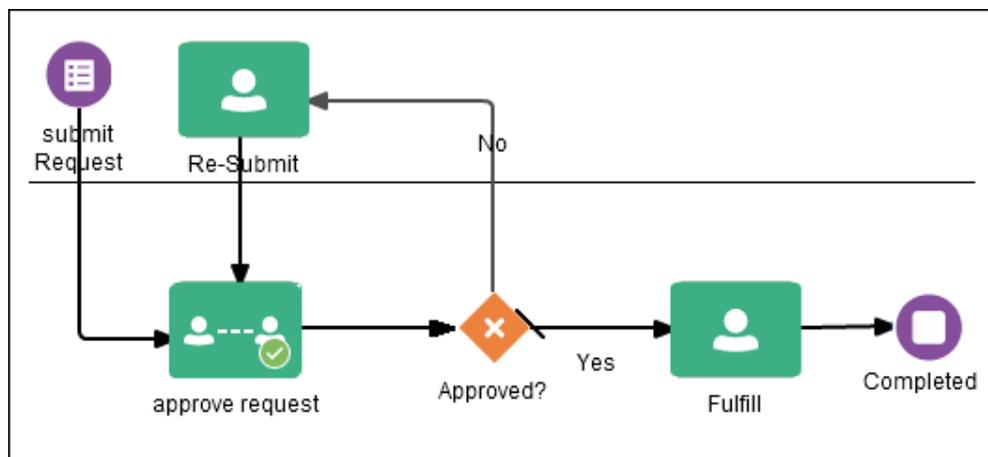
Note:

The exclusive gateway can also merge incoming sequence flows. However, there's no synchronization with other tokens that may be coming from other paths within the process flow. If other tokens arrive at an exclusive gateway merge, then they're passed through as is. If you're synchronizing tokens or performing evaluations on incoming sequence flows, then you should use a different type of gateway.

Example of an Exclusive Gateway

The following diagram shows an example of the exclusive gateway used within a Loan Approval business process. Here, the exclusive gateway is used to evaluate if the loan request is approved or if more information is required.

This evaluation is determined by the expression defined for the outgoing conditional sequence flow. If the expression evaluates to true, then the process flow proceeds down the No path. If it evaluates to false, then the process flow proceeds down the path of the default outgoing sequence flow.



Inclusive Gateway: Take One or More Paths

The inclusive gateway lets you split your business process into two or more paths. Unlike the exclusive gateway, however, a token may flow down one OR more of these paths depending on how the outgoing conditional sequence flows are evaluated.

The inclusive gateway requires a split-merge pair. When you add an inclusive gateway to your business process, the split and merge flow objects are automatically created:



The merge portion of the gateway is required. However, you don't have to ensure that all paths out of the split return to the merge.

Although it's possible to have process paths that split at a gateway without merging through the gateway, it's not a good practice and not recommended.

Note:

If you delete the merge gateway from a business process, the corresponding split gateway is also deleted.

Splitting and Merging Inclusive Gateways

The inclusive gateway splits a business process similar to the exclusive gateway, but allows tokens to proceed down one OR more outgoing sequence flows. You can define any number of outgoing conditional sequence flows for an inclusive gateway split. You must define at least one default sequence flow.

When a token arrives at an inclusive gateway, the expressions of its conditional sequence flows are evaluated.

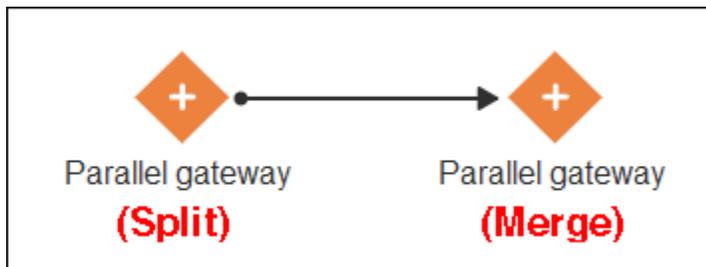
Next, a token is generated for each of the conditional sequence flows that evaluates to *True*. A token is generated for the default sequence flow only if none of the conditional sequence flows evaluate to *True*.

These tokens are joined at the merge of the inclusive gateway. When a token reaches the merge gateway, it waits until all the tokens generated by the split have reached the merge. After all the tokens have reached the merge of the inclusive gateway, the merge is complete, and the token continues to the next sequence flow after the gateway.

Parallel Gateway: Take All Paths Simultaneously

The parallel gateway lets you split your business process into two or more paths when you want your process flow to follow all paths simultaneously. The parallel gateway is useful when your business process must perform multiple tasks in parallel.

The parallel gateway requires a split-merge pair. When you add a parallel gateway to your business process, the split and merge flow objects are automatically created:



The merge portion of the gateway is required. However, you don't have to ensure that all paths out of the split return to the merge.

Although it's possible to have process paths that split at a gateway without merging through the gateway, it's not a good practice and not recommended.

Note:

If you delete the merge gateway from a business process, the corresponding split gateway is also deleted.

Splitting and Merging Parallel Gateways

When a token arrives at a parallel gateway, the parallel gateway creates a token for each outgoing sequence flow. The split of the parallel gateway doesn't evaluate any of the outgoing sequence flows.

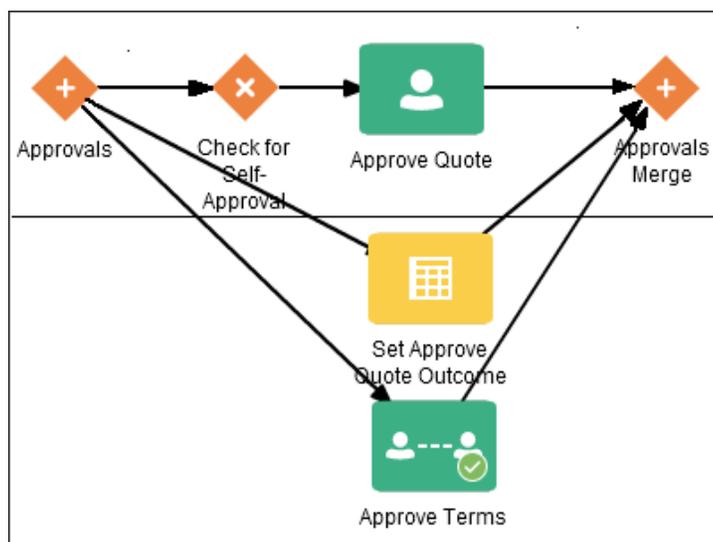
You can also use the parallel gateway to merge process paths split by the parallel gateway. The merge of the parallel gateway waits for a token to arrive from each of the incoming sequence flows. After all tokens arrive, only one token is passed to the outgoing sequence flow.

Note:

You must design your business process so that a token arrives for each incoming sequence flow for the merging parallel gateway. If you don't, your business process can freeze if the merge is expecting tokens that don't arrive.

Example of a Parallel Gateway

In this example, a Sales Quote process uses a parallel gateway at the approval stage. The diagram shows how the parallel gateway is used to execute all process paths at the same time.

**Event-Based Gateway: Take Only One Path (Event Occurrence)**

The event-based gateway allows you to branch your process flow based on the possibility that an event may occur. Depending on the context, the event may be one of several types.



The event-based gateway allows you to anticipate the possibility that several types of events may occur at a specific point in your process. It's similar to the exclusive gateway, but instead of selecting a path based on data-specific conditions (expressions), the event-based gateway selects a path based on the occurrence of an event within your process.

For example, in an order processing process, you may reach a point in your process when no stock is currently available. The process may have to wait until stock is available, but can't wait indefinitely. By using an event-based gateway, your process can wait for a message saying new stock has been received (using a message catch event) or it can continue if no message is received after a certain amount of time has passed (using a timer event).

Target Events for Event-Based Gateways

The event-based gateway is different from other gateways in that decisions about process flow are based on an event rather than data-specific conditions. For an event-based gateway, you define two or more of the following target events:

- **Message catch events:** When initiating a process using a message catch event, the process must be invoked using a message throw event.
- **Timer catch events:** Generally, only one timer event is used following an event-based gateway.
- **Receive tasks:** You can use the receive task to initiate a process instance following an event-based gateway. However, the process must be invoked from a send task within the calling process.

**Note:**

You can't mix message events and receive tasks within the same event-based gateway.

The target elements can only have incoming sequence flows from the event-based gateway. They can't have sequence flows from other parts of the process. Although the event-based gateway allows you to plan that multiple events may occur in your process, only one event is triggered within the process instance. When the first event in the event-based gateway is triggered, the path that comes after that event is followed.

When you add an event-based gateway to a process, no associated target events are created. When you delete an event-based gateway, any outgoing sequence flows are also deleted. The associated events aren't deleted.

Process Start with an Event-Based Gateway

You can also use an event-based gateway at the beginning of a process to create a new process instance. This is similar to having multiple start events within a process.

To allow an event-based gateway to create a new process instance:

- You must enable the **Instantiate** property of the event-based gateway.
- You can't have any incoming sequence flows to the event-based gateway.
- Your process must have at least one other start event, for example a message start event, in addition to the event-based gateway.

Although the event-based gateway can be used to create a new process instance, it doesn't accept data input from another process. Any data that must be passed to the process instance must be configured using the target events.

Create a Gateway

Use gateways to control how the process flows.

To create a gateway:

1. Open your process.
2. In the Elements palette, click **Gateways** to expand the list.

Here are the gateways you can add to a process:

| Gateway Icon | Gateway Type | Description |
|---|--------------|---|
|  | Exclusive | Only one of the paths out of the gateway is taken. The decision about which path the process should proceed along is based on data-specific conditions. For example, an exclusive gateway can specify different paths for the APPROVE and REJECT outcomes of an Approval human task. |
|  | Inclusive | One or more paths out of the gateway can be taken, and the paths must converge later in the process. Use this type to perform several optional or conditional tasks at the same time. |
|  | Parallel | All paths out of the gateway are taken, and the paths must converge later in the process. Use this type to perform several required tasks at the same time. |
|  | Event-Based | Only one of the paths out of the gateway is taken. An event-based gateway is similar to an exclusive gateway because both involve one path in the flow. However, for an event-based gateway, decisions about process flow are based on an event taking place rather than a condition being met. |

3. Drag and drop the gateway element into the process flow.
4. Select the gateway element. Click **Sequence Flow**  and drag the icon to create each path that exits from the gateway.
 - For exclusive or inclusive gateways, make sure the default path connects to the correct flow element. Default sequence flows represent the path your business process takes out of these gateways when none of the data conditions evaluate to true. Default sequence flows are represented by an arrow with a slash mark on one end.
 - For inclusive or parallel gateways, you can drag paths to the side to separate them.
 - For parallel gateways, you must drop at least one activity onto each path before you can create another path.
5. For exclusive or inclusive gateways, create a data object. Data objects store and organize data the process uses.
 - a. Select the gateway and click **Data Objects**  in the toolbar. The Data Objects dialog box opens.
 - b. Click **Add**.
 - c. Type a name for this data object. All names must begin with a lowercase letter.
 - d. Select the data type that matches the data flowing into the gateway from the previous task.

In most cases, the type is **boolean** (true or false) or **string** (text). For example, select **string** for the APPROVE and REJECT outcomes of an Approval human task.
 - e. Click **Create**.

- f. Close the Data Objects dialog box.
6. For exclusive or inclusive gateways, implement each non-default path. You don't need to implement the default path, but you should name it.
 - a. Select a non-default path, click **Edit**  on the path, and click **Implementation**.
 - b. Type a name for this path.
 - c. Click **Edit**  next to the Condition field.
 - d. Type an expression that tests whether the Gateway data object has one of the choices.

For example, type `approvalOutcome == "APPROVE"`. Note the double equal sign.
 - e. Click **OK**.
 - f. Close the Implementation pane.

7. For exclusive or inclusive gateways, associate the output data with the gateway data object.

A parallel gateway doesn't need data association, but some of the activities on the paths might need it.

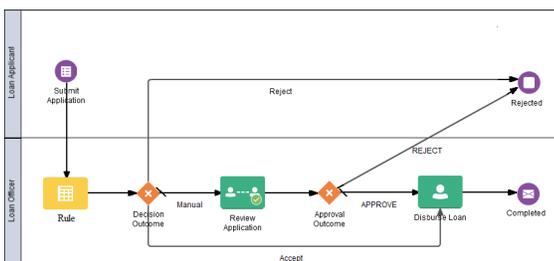
- a. Select the task that precedes and sends data to the gateway, and click **Data Association** in the toolbar.
- b. Expand the Process and Data Objects nodes in the right pane.
- c. Drag and drop the gateway data object into the Outputs field.

You can associate inputs as well, which often come from a form. If the form is referenced in the start event or another process component, then the form data object appears in the right pane. Drag and drop each form field from the right pane to the corresponding Inputs field.

- d. Click **Apply**.

The Data Association page closes.

8. Click **Save**  in the toolbar to save your changes.



Work with Data Mapper Elements

The **Data Mapper** element enables you to assign the value of a process data object, predefined variable, or literal to another data object or variable within the process.

Use this element in scenarios where it is not possible to assign values to certain data attributes through other activities in your process or to override the values assigned to attributes through the defined process flow. To define data associations through other elements, such as human tasks or service tasks, see [Configure Data Association](#).

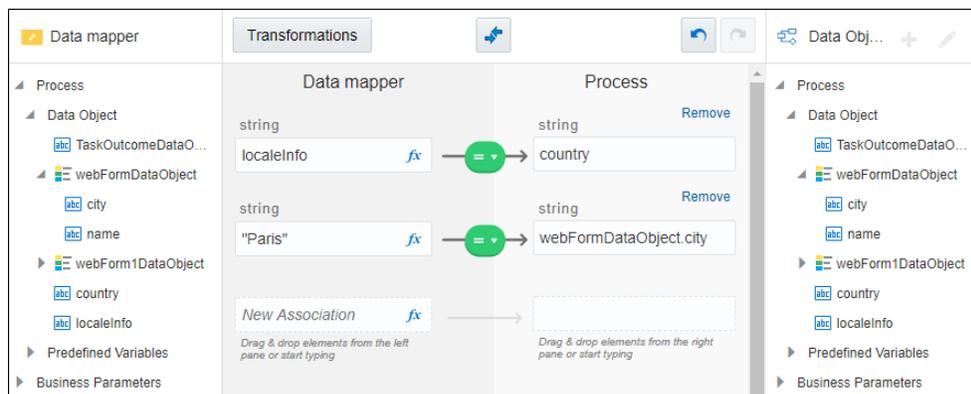
You can use the **Data Mapper** () at any point in your process to perform data mappings or associations. This element is available in the Elements Palette under **System**. Drag and drop the element at the required position on the editor canvas.

To define data mappings through this element:

1. Select the element, click **Menu** (), and select **Open Data Association**.

The Data Association editor opens. Notice that the editor contains only the **Output** tab because this element is just a mapper that serves to associate one process data object or variable with another. Unlike other elements (for example, human tasks), it does not have data objects of its own that require mapping with the process data on the input side.

2. In the editor, drag and drop elements from the left and right panes onto the center pane to map associations. Both the panes display all the available data objects or variables within the process. The object you add on the left serves as the source and the object on the right denotes the target of the mapping. The following figure shows a sample data mapping:



Here, the value of a process data object, *localeInfo* is assigned to another process data object named *country*, and a literal value of “Paris” is assigned to a web form data object, *city*.

Note:

If you chose a predefined variable as the target, ensure that it isn't read only.

3. Optionally, use the Expression Editor to build an input expression using standard functions and operators.

4. If there are validation errors, create transformations to resolve data type mismatches. See [Associate and Manipulate Data](#) for a detailed description on all the aspects of the Data Association editor.
5. Click **Apply** to save the data mappings.

Work with Decisions

Decision models automate operational decisions inherent in your business processes.

Use the **Decision** element to add decision model snapshots to your applications.



You can use different types of decision logic, such as decision tables, simple expressions, and so on, to create an executable model and incorporate it into your business process to facilitate automated decision-making.

Want to know more about working with decision models? See [Create Decisions](#).

Work with Bot Activities

To integrate your business process with applications that don't have an adapter in Oracle Integration and don't expose APIs, you'll typically need human intervention. Using a bot activity, you can replace these human tasks with robotic process automation (RPA) tools and extend workflow automation to disparate or legacy applications.

Configure a **Bot Activity** element , present in the Elements palette under **System**, within your process to call an RPA integration created using one of the available RPA adapters in Oracle Integration. Through an RPA integration, you can access an external RPA tool (based on the adapter you select) and use its capabilities to automate a particular task.

To use a **Bot Activity** element:

1. Drag and drop the element at the required position in your process flow on the editor canvas.
2. Select the element, click **Menu** , and select **Open Properties**.
3. In the **Bot Activity** field, select an active RPA integration.

 **Note:**

The **Bot Activity** field displays only the active integrations created using any of the available RPA adapters in Oracle Integration.

4. Define the required data associations for the element. See [Configure Data Association](#)

Keep in mind the following:

- To use the **Bot Activity** element, you must first create and activate an integration using one of the RPA adapters in Oracle Integration.
- Perform data associations for the bot activity in accordance with the type of data required and returned by the RPA integration.

Want to know more about RPA and how you can employ it in processes? See [Integrate with Robotic Process Automation Tools](#).

Work with Dynamic Processes

If a part of your business process is non-sequential or unpredictable, you can model that part as a dynamic process within your structured process.

Automate unpredictable steps within your business process that require expert knowledge or depend on changing circumstances by creating a dynamic process. Incorporate the dynamic process within your structured process by using a **Dynamic Process** element . This element is available in the Elements palette under **System**. Drag and drop the element onto the required position in your process flow in the editor canvas. Associate the dynamic process that you've created with this element from the properties pane, and define data associations.

Keep in mind the following:

- The dynamic process within a structured process is an *asynchronous* process, that is, it gets started by another process but runs independently.
- As player option is not supported in dynamic processes, so you cannot play a business process which has a part of it configured as a dynamic process.

Want to know more about dynamic processes and how you can use them in structured processes?

See [How do dynamic processes work in process applications?](#)

See [Use a Dynamic Process in a Structured Process](#).

Work with Sequence Flows

Sequence flows define the order or sequence that work is performed within a business process. Sequence flows connect the flow elements within your business process and determine the path a process token follows through your business process.

Incoming sequence flows are the sequence flows that lead into a flow element. **Outgoing sequence flows** are the sequence flows that determine the process path out of a flow element.

Most flow elements contain both incoming and outgoing sequence flows. Exceptions to this are start and end events. Start events can only contain outgoing sequence flows. End events can only contain incoming sequence flows.

To add sequence flows to your business process:

1. In the process editor, click the flow element from where you want to create the outgoing sequence flow.
2. Click **Add Sequence Flow**  and keep the mouse depressed.

This option only appears for flow elements that don't already have outgoing sequence flows.

3. Move the cursor to the flow element you want to connect to, and then release the click.

About Unconditional Sequence Flows

Unconditional sequence flows represent the typical path between two flow elements. Default sequence flows are displayed as arrows:



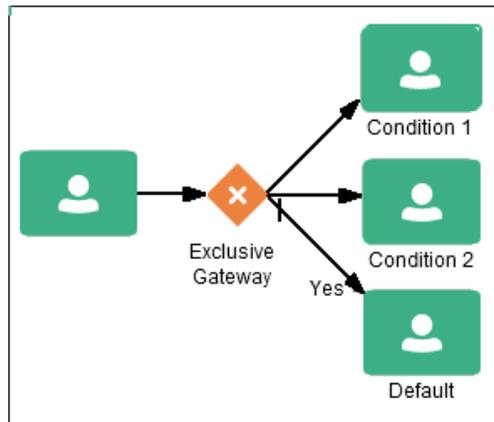
Most flow elements can contain only one default outgoing sequence flow. Only parallel gateways can contain multiple unconditional sequence flows, which represent the parallel paths of your business process.

Exclusive, event-based, inclusive, and conditional gateways can't have unconditional outgoing sequence flows. These gateways use conditional and default sequence flows to determine the flow of your business process.

About Conditional Sequence Flows

Use conditional sequence flows to control the flow of a business process based on certain conditions. Like unconditional sequence flows, conditional sequence flows are represented by an arrow.

Here's an example of two outgoing conditional sequence flows and a default sequence flow (which includes a slash mark at the beginning of its arrow):



Not all flow elements can use outgoing conditional sequence flows. Only the following types of gateways can have outgoing conditional sequence flows:

- Exclusive gateways
- Event-based gateways
- Inclusive gateways (split)

You use [expressions](#) to define the conditions used within a conditional sequence flow.

About Default Sequence Flows

Like conditional sequence flows, default sequence flows are used as outgoing sequence flows to exclusive, event-based, inclusive, and conditional gateways.

Default sequence flows represent the path your business process takes out of these gateways when none of the conditions evaluate to true. Default sequence flows are represented by an arrow with a slash mark on one end.

Work with Intermediate Events

Unlike start and stop events, intermediate events occur during the flow of your business process. You can use them to control the flow and behavior of your business process.

There are two types of intermediate events:

- **Normal flow events:** These events occur within the typical flow of your business process.
- **Boundary events:** These events trigger an interruption with your business process.

Boundary events are associated with flow elements and can be configured to interrupt their usual behavior. They behave similar to sequence flows in that they're used to determine the path a process takes between flow elements.

Boundary events can be divided into two types: interrupting and non-interrupting.

About the Timer Catch Event

Timer catch events lets you control the flow of your business process using a time condition.



Possible uses of the time catch event include:

- Creating a delay before running an activity
- Configuring a deadline for an activity
- Configuring a deadline for a process
- Triggering additional activities after an elapsed time

You can use timer events as boundary events on an activity. Timer events can be defined as either interrupting or non-interrupting boundary events.

When an interrupting timer event fires, the token leaves the main process flow to follow the process flow the timer event defines. The process flow that an interrupting timer event defines can return directly to the main process flow.

When a non interrupting event fires, a copy of the token is created and passes through the process flow the timer event defines. The process flow that a non-interrupting event defines can't return to the main process flow.

About the Error Boundary Event

Error boundary events are intermediate events used to handle an error that occurs within your process flow.

Error boundary events can be attached to the following flow elements:

- Service tasks
- Call activities
- Human tasks
- Send tasks
- Receive tasks
- Script tasks
- Decision tasks
- Subprocesses

Error boundary events are always interrupting, meaning that they interrupt the usual flow of a business process.

The following image shows the default notation for the error boundary event attached to a service task.



When a service or process fails with an error, the error boundary event is triggered. This causes the process flow to follow the path of the outgoing sequence flow of the error boundary event.

You can use this flow to define how to handle the error. This is handled in two ways:

- The process flow returns to the main process flow.

Any work that must be performed is handled within the error process flow before returning to the main flow.

Note:

If the boundary event is non-interrupting, the boundary flow can't return to the main process flow.

- The process flow continues to an end event.

The process is stopped immediately. Process control is returned to the service or process that initiated the process.

Work with Draft Processes

Process enables you to create and deploy draft processes. However, they must be valid before they can be deployed.

A draft process has one or more flow elements, which don't have their implementation defined. By deploying a draft process, you can test the parts of your business process that have been completed without having to wait until all flow elements have been implemented. To create a draft process, mark one or more flow elements within the business process as draft.

When a flow element is marked as draft, you can't configure data associations for it. If you mark a flow element as draft that has previously had data associations configured, these are lost.

You can define the implementation details of a draft flow element. However, it's not required. A draft flow element with no implementation defined doesn't generate errors during validation.

When an application containing a draft flow element is activated, implementation details for those flow elements are ignored. For example, if your business process contains a user task marked as draft, the runtime engine doesn't create instances of the associated human task.

To mark a flow element as draft, select the required flow element, click **Menu** , and select **Open Properties**. Select the **Is Draft** check box.

Marking your process as draft is different than marking it as document-only. Document-only processes are for descriptive use only and aren't used to control the application.

About the Abstract Flow Element

The Abstract flow element is a modeling placeholder for another activity or task.



You can mark the Abstract flow element as a draft element, which means you can activate and test the parts of your process that have been completed without having to wait until all the flow elements have been completed.

Work with Subprocesses

You can use subprocesses to organize your business processes. Subprocesses lets you group functional areas of your business process together and also make your business processes more readable.

Process supports the following types of subprocesses:

- Reusable processes
- Embedded (also called inline) subprocesses

In addition, you can configure multi-instance markers in subprocesses.

About Reusable Processes (Reusable Subprocesses)

Process supports reusable processes, sometimes referred to as reusable subprocesses. Reusable processes let you create business processes that can be called from other business processes.

For example, all your business processes may have to charge a credit card, so you can create a charge credit card reusable subprocess.

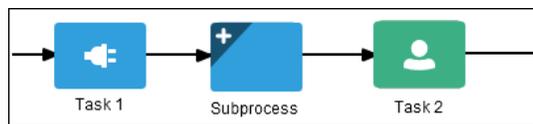
Reusable processes have the following characteristics:

- Must start with one none start event.
- Can contain multiple end events.
- Can only be called by other business processes.

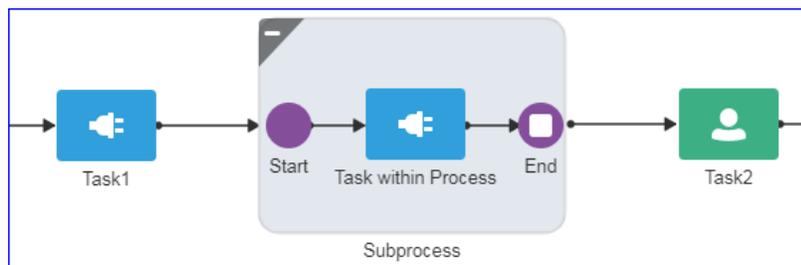
About Embedded Subprocesses (Inline Subprocesses)

Embedded subprocesses let you group flow elements together to make your business process more readable. In Process, subprocesses are embedded subprocesses. Subprocesses are contained as part of the parent subprocess. Embedded subprocesses must begin with a start none event and must end with a none end event.

Embedded subprocesses can be expanded or collapsed. The following is an example of how a collapsed subprocess appears within a process flow.



The following diagram shows how an expanded embedded subprocess appears within a process flow. When an embedded subprocess is expanded, you can edit the flow elements within it. You can also click and drag the edge of the embedded subprocess window to change its size.



Similar to other types of business processes, embedded subprocesses can have start and end events and contain a separate process flow. An embedded subprocess must begin with a none start event and end with a none end event. Embedded subprocesses can't contain swimlanes.

Embedded subprocesses also behave similar to activities. They can have incoming and outgoing sequence flows. They also contain data associations that define the data objects used within the embedded subprocesses.

Embedded subprocesses can also contain timer, message, and error boundary events.

If necessary, your business process can contain nested embedded subprocesses. However, use nested embedded subprocesses only when necessary to make your business process more readable.

Note:

Creating data objects for an embedded subprocess isn't supported.

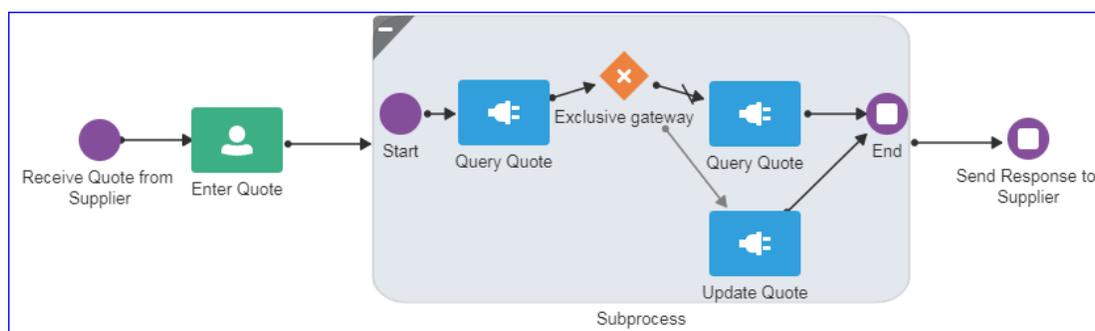
Embedded Subprocesses and Sequence Flows

The flow elements within an embedded subprocess can't have sequence flows that connect to flow elements outside the subprocess.

Similar to other flow elements, embedded subprocesses have incoming and outgoing sequence flows.

Embedded Subprocesses in Context

The following diagram shows an example of an embedded subprocess that groups the service tasks used to process a sales quote.

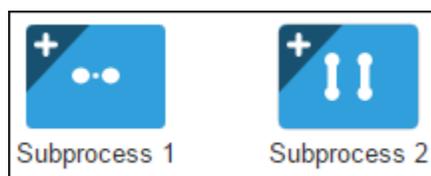


Configure Multi-Instance Markers in Subprocesses

Multi-instance markers enable you to run a subprocess for each of the elements on a set of data. When a subprocess with a multi-instance loop marker runs, it creates a set of instances, one for each element on the set of data. You can configure the multi-instance marker to process these instances in parallel or in a sequence.

For example, you can use multi-instance markers in repetitive processes, such as creating travel requests for a group of employees, ordering office supplies for a team, and so on. Multiple instances are created, one for each employee, which are then passed through the process of individual requests being made and forwarded to the approver. These instances can be configured to run either in sequence or in parallel.

If multi-instance markers are used, an indicator for the same appears on the subprocess; two white circles indicate sequential execution, while two vertical white lines indicate parallel execution.



To configure multi-instance markers in a subprocess:

1. Click the subprocess.
2. Select **Properties**. The Implementation pane appears.
3. Select **Generate multiple instances** under Repetition Cycle.
4. Select the way in which you want to manage your instances:

- a. **In Sequence:** Select to specify that the each instance must complete the subprocess before the next instance starts to run the subprocess.
 - b. **Parallel:** Select to specify that all instances run in parallel.
5. Specify the type of input you want to supply:
 - a. **Condition:** If your input is not an array, then select **Condition** to specify the number of times to execute the sub-process.
 - i. In the adjacent expression area, enter a number to specify the cardinality. You may also enter a data object or a simple expression to specify the same. To launch the Expression Editor window, click the **Edit Expression** button.
 - b. **Array:** Select to enter an array of elements as input.
 - i. In the input expression field, enter the array that is to be consumed by the subprocess. You can select a data object or form a simple expression using the Expression Editor window. Generally, the selected data object is a collection of items.
 - ii. In the output expression field, enter the array to store the result of the subprocess.
6. Optionally, you can specify a completion condition to terminate the execution of instances.
 - a. Select the **Define a condition that terminates execution of subprocess instances** check box..
 - b. In the resulting expression field, enter a completion condition in terms of a simple expression.

**Note:**

- At the subprocess level, you can add data objects (of any type) with their scope limited to an individual subprocess execution. Such data objects help you, especially in parallel execution, to keep local copies of data and merge them after all instances are executed.
- In the XPath expression that is used to fetch data objects for each instance, make sure that the index starts from '1' and not from '0'.

**Important:**

If you have a subprocess that invokes multiple instances of another process, aborting any of these child instances triggered by the parent process will abort all child instances in the same flow.

Predefined Variables

The following predefined variables are available for multi-instance subprocesses. Data communication within a multi-instance subprocess should happen using these variables.

| Name | Data Type | Description | Available for MI Array? | Available for MI Condition? |
|-----------------------------|-----------|--|-------------------------|-----------------------------|
| loopCounter | int | The sequence number of the iteration. | Yes | Yes |
| numberOfInstances | int | The total number of instances. | Yes | Yes |
| numberOfActiveInstances | int | The total number of active instances. | Yes | Yes |
| numberOfCompletedInstances | int | The total number of completed instances. | Yes | Yes |
| numberOfTerminatedInstances | int | The total number of terminated instances. | Yes | Yes |
| loopDataInput | string[] | The reference of the array that is passed as input to the multi-instance subprocess. | Yes | No |
| inputDataItem | string | The handle to an element of the <code>loopDataInput</code> array passed as input to the multi-instance subprocess. | Yes | No |
| loopDataOutput | string[] | The reference of the array that is passed as output of the multi-instance subprocess. | Yes | No |
| outputDataItem | string | The handle to an element of the <code>loopDataOutput</code> array that is passed as output of the multi-instance subprocess. | Yes | No |

**Note:**

Transformations are not supported for predefined variables.

Handle Errors with Event Subprocesses

Use event subprocesses to handle exceptions that occur in the runtime life cycle of a process and cause it to fail.

Generally, a process can encounter two types of exceptions, namely system and business exceptions.

- **System exceptions:** Exceptions related to the underlying software or hardware infrastructure of Processes; for example, connectivity loss, database connection failure, or invoke activity failure.
- **Business exceptions:** Faults related to process applications that occur when there's a problem with information processing; for example, a stock control and inventory service throwing an error when a stock item is not found.

Using an event subprocess element, you can catch all or specific exceptions and recover the original process flow or supply an alternative, exception-handling flow. You can reuse an event subprocess (or the exception-handling flow) with multiple elements in your process. Also, you can define more than one event subprocess to handle different exceptions within a single process.

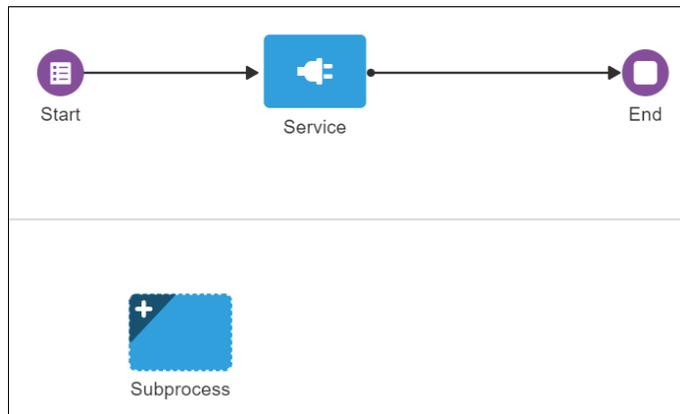
Topics:

- [Add an Event Subprocess](#)
- [Configure a Start Error Event](#)

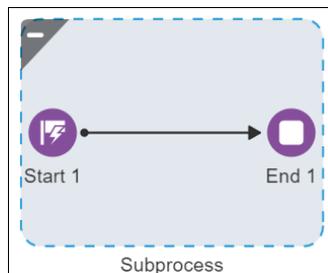
Add an Event Subprocess

To add an event subprocess to your flow:

1. On the Elements palette, expand **System**.
2. Drag the **Event Sub-Process** element onto the process editor canvas, and drop it at the required position. Note that you should add the element outside your process flow as event subprocesses don't support incoming or outgoing flows.



3. Click **Expand**  to expand the element. You'll see a default error-handling flow with a Start Error event and an End event. You can add other elements, such as human tasks, to the flow according to your requirements.



You cannot configure properties or define data association for the element as a whole; however, you can configure the Start Error event to catch and handle exceptions. See [Configure a Start Error Event](#).

Configure a Start Error Event

You can configure the Start Error event within an event subprocess to catch system or business exceptions.

Topics:

- [Catch an Exception](#)

Catch an Exception

Select the Start Error event, click **Menu** , and then click **Open Properties**. In the Properties pane:

1. Select **Catch all Business Exceptions** to capture any business faults in runtime.
2. Select **Catch all System Exceptions** to capture any system faults in runtime.
3. Click **Browse**  next to the **Exception** field to browse for specific exceptions.
 - a. In the exception browser, click **Search** to look for and select a specific business exception you've defined, and then click **OK**. You can define business exceptions for a process application on the **Business Types** tab.
 - b. Select **Show System Faults** to view all predefined system exceptions. Click on an exception to add it to the **Exception** field. The following table describes all available system faults:

| System Fault | Description |
|-----------------------------|---|
| AssertFailure | The specified assertion has failed. |
| BindingFault | The preparation of the operation invoked in a flow object has failed. You cannot retry the invocation after a binding fault as recovering from this error generally requires human intervention. |
| InvalidVariables | The variables used aren't valid. |
| RemoteFault | Problem invoking a service in a flow object. For example, a remote service has returned a SOAP fault. |
| Rollback | The receiver of the exception is enabled to roll back the current Java Transaction API (JTA) transaction from within the process flow. |
| Timeout | The service has exceeded the response time-out period. |
| ConflictingReceive | There are multiple receive activities to respond to the invoked operation. |
| ConflictingRequest | There are multiple requests on the same partner link for the invoked operation. |
| CorrelationViolation | The message doesn't provide the required correlation information. |
| ForcedTermination | The service has been terminated due to a SOAP fault. |
| InvalidReply | The reply doesn't contain the correlation information required by the corresponding receive. |
| MismatchedAssignmentFailure | The assigned types are incompatible. |
| RepeatedCompensation | A compensation handler is invoked multiple times. |
| SelectionFailure | Error running a selection operation. |
| UninitializedVariable | The variable you're trying to access has not been initialized. |

Communicate Between Processes

Processes can interact using message start and message end events, send and receive activities, or message throw and message catch events. Processes can also call other processes or include subprocesses.

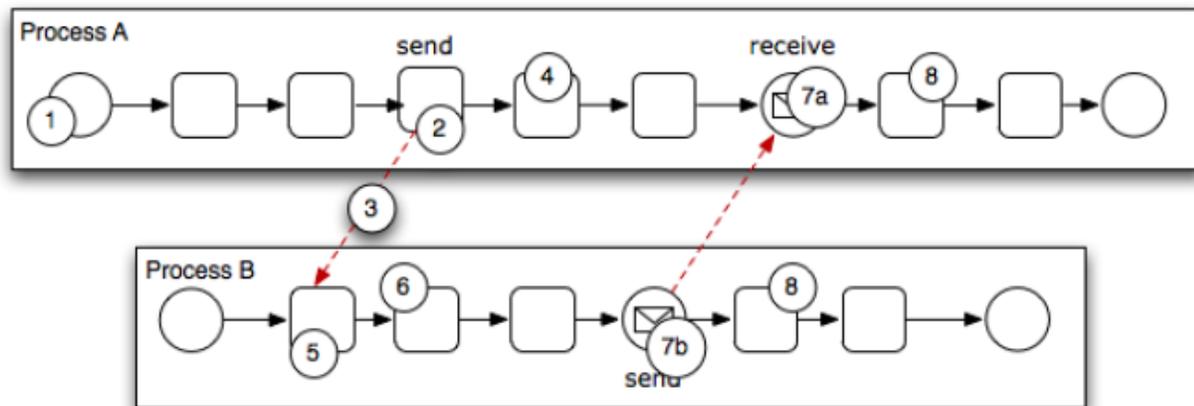
Processes interact *synchronously* when one process starts another and waits for a response. Processes interact *asynchronously* when one process starts another and both processes run independently.

| Interaction Type | Description |
|--|---|
| Message Start and Message End events | <p>If you define input arguments for a process in a message start event, you must supply data for these arguments to start the process.</p> <p>You can asynchronously start a process that begins with a message start event using a message throw event or send activity in another process. Or you can synchronously start such a process using a Service activity.</p> <p>A process that begins with a message start event is exposed as a web service after deployment. Therefore, you can connect with the process as a web service consumer to start it. See Creating a Web Service Connection.</p> <p>If you define output arguments for a process in a message end event, the process supplies data for these arguments when it completes.</p> <p>See Defining Input or Output Arguments.</p> |
| Send and Receive activities | <p>A send activity sends data to and starts another process. A receive activity receives data from another process when it completes.</p> <p>Send and receive activities can have boundary events, unlike message throw and message catch events, which can't.</p> <p>See Using Send and Receive to Communicate Between Processes.</p> <p>For a send activity in a calling process, see Calling Another Process. For a receive activity or a send activity in the called process, see Defining Input or Output Arguments.</p> |
| Message Throw and Message Catch events | <p>A message throw event sends data to and starts another process. A message catch event receives data from another process when it completes.</p> <p>See Using Message Throw and Catch to Communicate Between Processes.</p> <p>For a message throw event, see Calling Another Process. For a message catch event, see Defining Input or Output Arguments.</p> |
| Service activity | <p>A Service activity can call another process or a web service. The send, receive, and call activities and the message throw and message catch events can only call another process asynchronously. The Service activity can call a process asynchronously or synchronously. However, if the called process begins with a message start event, the call must be synchronous.</p> <p>For web service details, see Creating a Web Service Connection.</p> <p>See Calling Another Process.</p> |
| Call activity | <p>A Call activity starts another process as a child process. The parent process waits for the child process to complete before continuing. This lets you reuse a process in multiple processes that call it.</p> <p>See Defining Input and Output Arguments for a Child Process.</p> |
| Subprocess activity | <p>A Subprocess activity contains a subprocess with its own start and end events. A subprocess isn't separate from its parent process. A subprocess activity lets you hide the complex details of a part of a process to make the overall process more readable.</p> <p>Click Expand  to expand the subprocess, then drag and drop activities directly into the subprocess. Click Collapse  to collapse the subprocess.</p> |

Use Send and Receive to Communicate Between Processes

Use the send and receive activities to start another business process and receive messages back from it. Processes that begin with a receive activity and contain a send activity are exposed as services that other processes and services within Process can use.

The diagram shows how a send activity starts another process and a receive activity receives a response.



The following steps outline a possible scenario for using send and receive activities to communicate between processes.

1. *Process A* starts.
2. The flow of *Process A* reaches the send activity.
3. The send activity starts *Process B*.
The send activity implementation specifies which process is started.
4. The flow of *Process A* continues to the next flow element.
5. The receive activity initiates an instance of *Process B*.
The start event in *Process B* must have a Trigger value of None. The receive activity implementation must have **Create Instance** checked.
6. *Process B* runs.
7. Depending on the specific behavior of both processes, the following scenarios may occur:
 - a. If the flow of *Process A* reaches a receive activity paired with a send activity from *Process B*, the flow of *Process A* waits until a response is received.
After the response is received, the flow of *Process A* continues to the next flow element.
 - b. If the flow of *Process B* reaches a send activity paired with a receive activity in *Process A*, *Process B* sends a response to *Process A*.
The flow of *Process B* continues to the next flow element.

8. Both business processes continue running.

You can use subsequent send and receive pairs to define subsequent communication between the two processes.

Use Message Throw and Catch to Communicate Between Processes

Use combinations of message throw and message catch events to start and communicate with other business processes.

When using a message throw event to start another business process, the following conditions must be met:

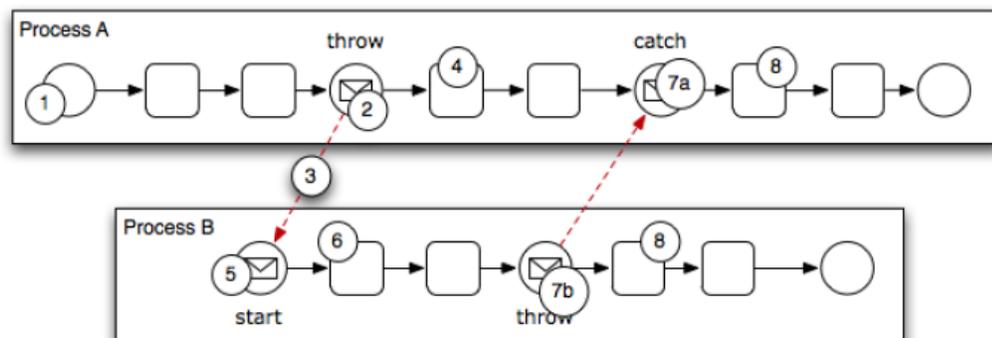
- The business process being started must be an asynchronous process.
Although you can use a message throw event to start a synchronous process, there is no mechanism for catching messages synchronously from the other process. To start a synchronous process, use the Service activity.
- The first time you use a message throw event, it must be paired with the message start event of the process it starts.

This is required to start the process instance. After the instance has been started, you can use subsequent message throw events that are caught by other events in the second process.

Processes that begin with a message start event are exposed as web services. See [Creating a Web Service Connection](#).

Processes started by another process aren't considered child processes. This means that a terminate end event in the calling process doesn't stop a process started with a message throw event.

This diagram shows how a message throw event starts another process and a message catch event receives a response.



The following steps outline a possible scenario for using message throw and catch events to communicate between processes.

1. *Process A* starts.
2. The flow of *Process A* reaches a message throw event that starts *Process B*.
3. The message throw event sends a message to the message start event of *Process B*.

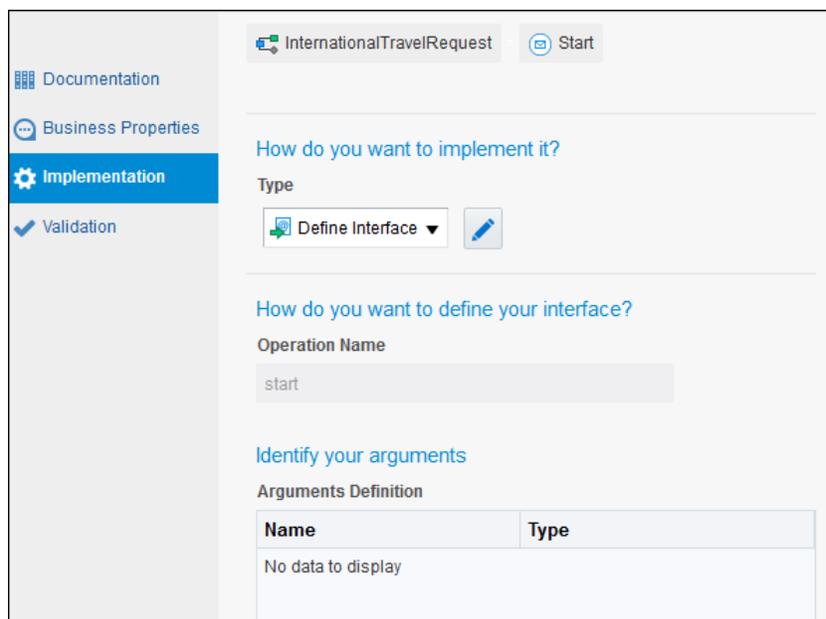
4. The flow of *Process A* proceeds to the next flow element.
5. The message start event starts an instance of *Process B*.
6. *Process B* runs.
7. Depending on the specific behavior of both processes, the following scenarios may occur:
 - a. If the flow of *Process A* reaches a message catch event paired with a message throw event from *Process B*, the flow of *Process A* waits until the message is received.
After the message is received, the flow of *Process A* continues to the next flow element.
 - b. If the flow of *Process B* reaches a message throw event paired with a message catch event in *Process A*, *Process B* throws a message to *Process A*.
The flow of *Process B* continues to the next flow element.
8. Both processes continue running.
You can use subsequent message throw and message catch event pairs to define subsequent communication between the two processes.

Define Input or Output Arguments

You can define input arguments for message start and message catch events and for receive activities. You can define output arguments for message end events and for send activities in processes called by other processes.

To define input or output arguments for a process interaction event or activity:

1. In the process editor's Elements Palette, expand the system or event elements.
2. Drag and drop the element into the process.
3. Select the element, click **Menu** , and then select **Open Properties**. The implementation pane appears.



The screenshot shows the implementation pane for a 'Start' event in a process named 'InternationalTravelRequest'. The left sidebar has tabs for 'Documentation', 'Business Properties', 'Implementation' (selected), and 'Validation'. The main area contains the following sections:

- How do you want to implement it?**
 - Type: 
- How do you want to define your interface?**
 - Operation Name:
- Identify your arguments**
 - Arguments Definition:

| Name | Type |
|--------------------|------|
| No data to display | |

4. For a Receive activity in a process called by another process, check **Create Instance**.
5. Select **Define Interface** from the **Type** drop-down list. Click **Edit**.
6. Click **Add** to add new arguments, then determine the **Name** and **Type** of each argument.
7. For a Send activity in a process called by another process, select **Synchronous** if the process must wait for a reply.
8. Enter an **Operation Name** or accept the default name.
9. Define data associations to pass the input argument data to the next activity in the process. See [Configure Data Association](#).

When you activate a process that begins with a message start event, the process is exposed as a web service. Information about the web service is displayed when you select **View Deployed Services** on the Manage Active Applications page. See [Integrate with Applications and Services](#) and [Administrator Basics](#).

Define Input and Output Arguments for a Child Process

You can define input and output arguments for a child process called by a call activity in the parent process.

The start and end events in the child process must have Trigger values of None. The child process can't contain elements that only make sense in an independent process, such as message events and send, receive, or service activities.

To define input or output arguments for a child process:

1. Open the child process in the process editor.
2. In the process editor, click the **Restore Pane** icon in the lower-right corner.
The Properties pane opens at the bottom of the window.
3. Click the **Add** icon to add new input or output arguments, then determine the Name and Type of each argument.
4. Open the parent process in the process editor, select the call activity, click **Menu** , and select **Open Properties**.
5. Click **Implementation**.
6. Select the child process from the Process drop-down list and close the Properties pane.
7. Define data associations for the call activity to pass the input data to the child process and receive the output data from it. See [Configuring Data Associations for a Process Flow Element](#).

Call Another Process

You call another process using a message throw event or a send, service, or call system activity.

To call another process:

1. In the process editor, expand the **System** or **Events** types in the Elements Palette.
2. Drag and drop a system or event element into the process.

3. Select the element, click **Menu** , and select **Open Properties**.
4. Click **Implementation**.
 - For a call activity, select a process to call from the Process drop-down list. Want to learn more about using a Call activity? See [Define Input and Output Arguments for a Child Process](#).
 - For all other calling components, select **Process Call** from the Type drop-down list. Click the **Browse** icon, select a process, then click **OK**.
5. Select a different Target Node from the drop-down list if necessary.
 - For a message throw event, the message start events in the called process are listed.
 - For a send or service activity, both message start events and receive activities are listed.
6. Define data associations to receive input data from the previous activity in the process. See [Configure Data Association](#).

Use Correlations to Communicate Between Processes

Are you looking for another way to get your processes to talk to one another? Let's look at how you can develop a process that uses correlations to communicate with other processes and services.

Topics:

- [Correlations Lead to Process Communication](#)
- [Components of a Correlation](#)
- [Define Correlation Keys and Properties](#)
- [Use Your Correlations in a Process](#)

Correlations Lead to Process Communication

Correlations enable business processes to communicate with each other based on the state of an instance. The state of all the process data objects in a process defines the state of the instance.

When you define a correlation for a business process, you can identify an instance in another process based on the instance state and then send a message to that specific instance.

For example, you can use correlations to communicate a sales process with the corresponding shipping and mailing processes. When the customer confirms an order, the shipping process sends a message to the shipping and mailing processes using a correlation that defines that it uses the order ID to locate the instances in both processes.

After you initialize a correlation, you can't change its value because the service engine uses this value to locate the instance.

You can define and initialize multiple correlations for a flow element.

- The flow element that sends the message can use just one correlation or all the correlations defined for that flow element. If the flow element uses all the existing correlations, then all the values it sends together with the message must lead to the identification of the same instance.

- Some flow elements are able to initialize a correlation (that is, Start), some to use it (that is, End), and some to both initialize and use (that is, Receive Task).
- For those activities that manage input and output arguments, such as the service task, you can set the property value based on one or the other.

The scope of the correlation is the instance of the process or subprocess where it's defined.

Be careful not to initialize the same correlation more than once. If you do, you'll get an error at runtime.

Components of a Correlation

The main components of a correlation are definition, keys, properties, and property values.

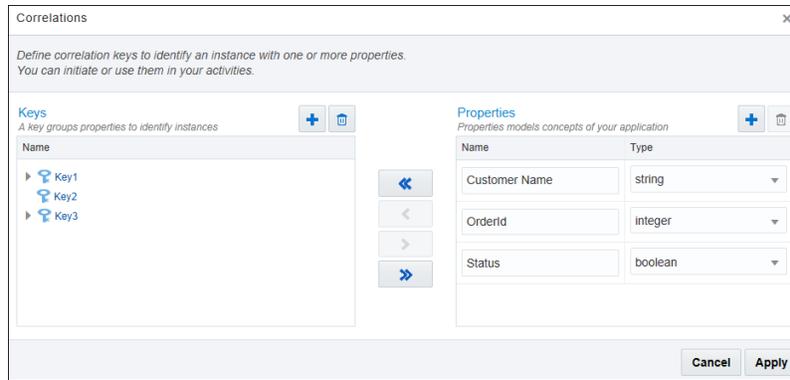
- **Correlation definition** contains the set of correlation keys defined for a flow element.
- **Correlation keys** define a unique name to identify the properties to use in the correlation. When you define a correlation key, you provide a name to identify it. The scope of the correlation key is the process, which means that after you define a correlation key you can use it for the correlation definition of any flow element in that process.
- **Correlation properties** are abstractions for very representative attributes in the process, like the order ID, the customer name, or the social security number. Properties contain a name to identify the attribute and a data type. Properties only support basic data types. The scope of the properties is the application. You can see the property definition in the different processes.
- **Correlation property values** enable you to define how to assign a value to the correlation property using expressions. You can use the arguments and predefined variables of the activity to set the correlation property values.

Define Correlation Keys and Properties

You define correlation keys at the process level, which means you can reuse correlation keys across your process and flow elements.

To define correlations keys and properties:

1. Go to the Process Applications page.
2. Open your application.
3. Open your business process.
4. Click **Correlations**  in the toolbar. The Correlations dialog box opens.



5. Create the correlation keys and properties, and then associate one or more properties with each key you create. Use the arrow keys to move information between the Keys column and the Properties column.

Use Your Correlations in a Process

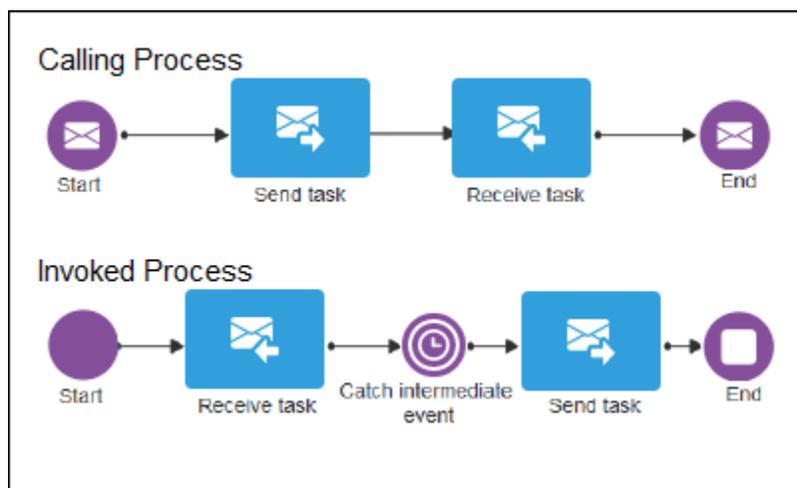
You can define multiple correlations for a single flow element (that is, an activity). The activity that sends the message can use one correlation, several correlations, or all of them. The values used to invoke the correlation lead to the identification of the same instance.

To communicate between processes, you can define correlations for the following activities:

- Message start and message end events
- Message throw and message catch events
- Send and receive activities
- Signal events

To define and use correlations in your processes:

1. Go to the Home page, click **Processes**, and then open your application.
2. Design and build the two processes that will communicate with each other. One process is the **calling process**; the other is the **invoked process**. For example:

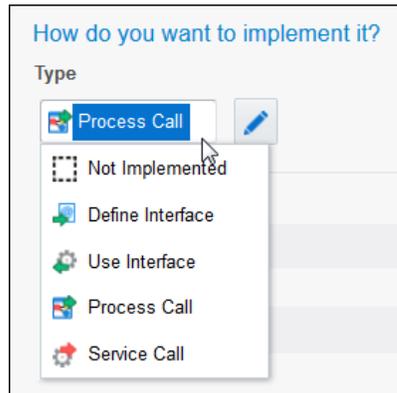


3. Open the properties for an activity.
 - a. Select the activity in your process diagram.

- b. Click **Menu**  and select **Open Properties**.

The Properties pane opens at the bottom of the window.

4. Define the general properties.
 - a. Select the **General** tab.
 - b. Define how you want to implement the activity.

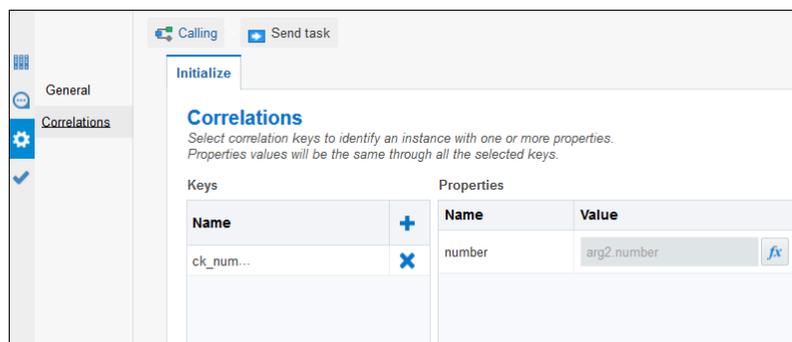


- c. Click **Edit**  to add arguments, and define the name and type of each argument.

Define the input arguments for message start and message catch events and for receive activities. Also, define the output arguments for message end events and for send activities in the invoked process. See [Define Input or Output Arguments](#).

If the required arguments aren't defined, then you can't configure the correlations. The Correlations tab displays a disabled pane.

5. Define the correlation properties.
 - a. Select the **Correlations** tab.



- b. Define a correlation and configure it to initiate the property values.

A few things to note:

- In the Keys box, click **Add**  and select one of the available keys. If no keys are defined, then you need to create them. See [Define Correlation Keys and Properties](#).

- After you select one of the available keys, the contained properties display in the Properties box. To set values for the properties, click **Expression Editor**  to open the editor.
- Some activities are only able to initialize correlations (configured on the Initialize tab) while other activities are able to use the correlations (configured on the Correlate tab). Also, for activities that have input and output arguments, you can select one or both, and then set the property values according to the selected context.

Work with Human Tasks

You can use human tasks to model user interaction with the application. Human tasks enable you to display a form for the user to view or complete, and click an action to perform.

There are different types of human tasks that enable you to model different types of interactions:

- **Submit Tasks:** Enable you to display a form that the user must submit to create a request or to provide information about a certain subject.
- **Approval Tasks:** Enable you to display a form that the user must review or complete and then perform a certain action. The user might approve or reject the request. You can also define custom actions for the user to perform. Approval tasks enable you to define an approval pattern. Generally you use the outcome of the approval task to drive the rest of the process flow.

After you create the human task, you can configure implementation details such as its assignment, priority, due date, reminders and deadlines.

- [Typical Workflow for Creating a Human Task](#)
- [Create Submit and Approval Tasks](#)
- [Configuring Human Tasks](#)
- [Customize Notification Emails for Human Tasks](#)

Typical Workflow for Creating a Human Task

Human tasks enable you to model how a user interacts with the application. When you add a human task to your process, you must also configure its implementation, define a form, and configure the human task data associations.

When you create a human task you must:

1. Add a human task to an existing business process.
2. Configure the human tasks implementation.
3. Create a form to display the human task information.
4. Configure the human task to use the created form.
5. Configure the human task input and output arguments using data associations.

Create Submit and Approval Tasks

You can create human tasks to model the user interaction with the application. You can use submit tasks to display a form for the user to complete and submit. You can use approval tasks to display a form for the user to view and/or complete, and then perform an action such as approving it, or rejecting it, or a custom action that you define.

To create a submit task:

1. Edit an existing business process or create a new one.
2. Click the **Show All** button located at the bottom of the Activities toolbar.
3. Select the **Human** task.

The Submit and Approve tasks appear to the left of the toolbar.

4. Click the task that you want to add. Then drag it to the process editor and drop it on the swimlane that represents role that you want to have access to this human task. If you drop the task over a transition, it gets automatically added to the process flow.

The Submit or Approve task is added to the process.

5. Configure the human task by following one of the procedures described in [Configuring Human Tasks](#).
6. Click the **Save and Continue Editing** button located next to the editing label in the toolbar.

A default process data object with the name `TaskOutcomeDataObject` is automatically created for the human task outcome and is shared by all the human tasks in a process. This default data object is automatically associated with the outcome field in the human task output when you click **Save**.

 **Note:**

`TaskOutcomeDataObject` is a reserved name and therefore, you can't use this name when creating data objects.

Configuring Human Tasks

You can configure the assignment of the human task, the form to use to display its information, the title and summary to identify it, its due date and priority, the number of reminders to send to its assignees, and the action to take when it reaches a certain deadline.

To configure a human task:

1. In the process editor, click the human task, select the **Menu** icon, and then select **Open Properties**.

The Properties pane opens at the bottom of the window. The General tab is selected by default.

2. On the **Properties** pane, you can configure the following:

| Tab | More Information |
|----------------------------------|---|
| General | Work with Draft Processes Assign Human Tasks Use Forms to Display Task Information Use an External UI to Display Task Information Define an Approval Pattern Specify Task Actions Shown to Users Configure the Title and Task Summary Configure the Due Date and Priority Bypass the Approval Chain |
| Reminders | Configure Reminders |
| Escalation and Expiration | Configure Task Expiration, Renewal, or Escalation |
| Notification | Customize Notification Emails for Human Tasks |
| Documents | Override Documents Folder Access |

3. Save your changes by clicking the **Collapse Pane** icon in the top-right corner of the pane.

Assign Human Tasks

You can assign the human task to a specific user, to a group of users, to the users in a certain role, or to the same user that already acted on the instance for a certain role. You can also use expressions to calculate the user, group or shared role.

To assign a human task to a specific user:

1. In the General tab on the implementation pane, click **Edit** next to the Assignee(s) field. The Define Assignees dialog box opens.
2. From the **Build a list of participants using** drop-down list, select one of the following:
 - Lane Participants
 - Names and Expressions
3. If you selected *Lane Participants*, select one of the following options:
 - Exclude the following participants:
 - **Creator:** The user who created the process.
 - **Previous Participants:** Users who have already acted within this task instance.
 - **Expression:** User from an expression.

Selecting this option, activates the expression editor icon. After creating an expression, it appears in the text box located next to the check box.

Note:

If you exclude a user from a task, the user will be able to view the task but not have the permission to act on it.

- Any member of current swimlane role: assigns the human task to any participant granted the role required to run the human task.

4. If you selected *Names and Expressions*,
 - a. Exclude the following participants:
 - **Creator:** The user who created the process.
 - **Previous Participants:** Users who have already acted within this task instance.
 - **Expression:** User from an expression.

Selecting this option, activates the expression editor icon. After creating an expression, it appears in the text box located next to the check box.
 - b. Click **Add**.
 - c. Select one of the available options:
 - Add User
 - Add Group
 - Application Role

A row with the participant type that you selected appears in the List of Assignees table.
 - d. Click the cell for the **Data Type** column to select a data type.

To specify the participant using names, select *By Name*.

To specify the participant using an expression, select *By Expression*.
 - e. If you selected *By Name*, click **Search** to select a participant.
 - f. If you selected *By Expression*, click **Expression** to enter an expression that determines the user or group to assign the human task to.
5. Click **OK**.

The **Define Assignees** dialog box closes and the selected participant or expression appears in the **Assignee(s)** field.
6. Enter the **Percentage Required** and the **Default Outcome**.

These fields only appear if you select *All Assignees in Parallel* option from the **Who are the approvers?** drop-down list. This is the default outcome when the required percentage isn't reached. For example, the required percentage is 51% and the default outcome is *APPROVE*. 55% of assignees vote to reject the task. Therefore, the outcome is *REJECT*.

Use Forms to Display Task Information

You can configure the human task to use a specific form to display the information the user needs to view or complete to perform the task assigned to them. You can use an already existing form, or create a new one.

When you implement a human task with a form, data association is automatically performed when a form is selected:

- If the data object already exists, then that one is used.
- If the data object with the same name doesn't exist, then the first data object of the same type is used.
- If there's no data object of the same type, then a new data object is automatically created. New data objects use this naming convention:

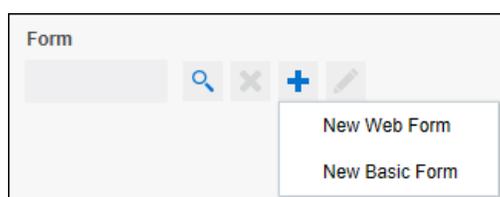
```
<form-name-starting-in-lower-case>DataObject<n>
```

where *n* is a number added to avoid duplicate names.

- Name restrictions for data objects and forms are similar, so no special treatment is required.
- After data association is performed, you're notified with a message below the form input box.

To associate a form with a human task:

1. In the process editor, select a human task, click **Menu**  and select **Open Properties**.
2. In the properties that display, either browse to select a form or create a new one. To create one, click **Add +** next to the **Form** field, and select an option for creating a new form.



3. Enter a form name and click **Create**.

If you select the **Open Immediately** check box, then the form automatically opens in the form editor.

If you're configuring an approval task, you can edit the default actions and add new ones. To add actions, enter their name in uppercase and separate them with commas. You can later access the value of the selected action from the human task data association, using the predefined data object `outcome`.

If you're configuring a submit task, the only allowed action is SUBMIT.

Once a form is associated with a human task, you can open the form by selecting the task, clicking **Menu** , and selecting **Open Form**.

See [Work in the Web Forms Editor](#).

Note:

When you copy and paste a human task anywhere within your BPM process, the associated form is not copied.

Use an External UI to Display Task Information

You can configure a human task to use an external form for displaying the information a user needs to view (or supply) in order to complete the task.

To associate an external UI with a human task:

1. In the process editor, select a human task, click **Menu**  and select **Open Properties**.

- In the Properties pane, click **Browse**  and select an existing external UI connector from the resulting dialog. All external UI connectors within the application appear in the dialog.
 - To add a new external UI connector, click **Add +** next to the Form field, and select **New External UI**. See [Create an External UI Connection](#) for details.
- From the Data Association window, assign necessary values to path parameters, query parameters, and custom payload attributes that you need to pass to the external application.

Define an Approval Pattern

Approval tasks let you define an approval pattern by specifying actions. By default, the actions APPROVE and REJECT are already specified. However, you can also define custom actions, such as HOLD and MOREINFO.

To define an approval pattern:

- Go to the Action field on the General tab in the Properties pane.

The actions APPROVE and REJECT appear by default.

- Enter additional custom actions if required.

Custom actions must be in all uppercase letters. Each action must be separated by a comma.



* Action
APPROVE,REJECT,MOREINFO

Configure the Title and Task Summary

You can specify a title to identify the human task, and a summary to describe it. The title and the summary appear in the user's task list, so that they can easily identify the human task they're looking for without having to view the details. You can specify the title and the summary using plain text, or generate it using expressions.

To configure the title and task summary:

- Select the human task element in your process diagram and open its Properties pane.
- Click the **General** tab.
- Select one of these options for the **Title** and **Task Summary** fields:
 - Plain Text:** Use a text string to enter the title or summary to identify the human task.
 - Expression:** Click the **Expression Editor** to specify an expression that calculates the title or summary of the human task. The Expression Editor dialog box opens. See [About Expressions](#).
 - Reset:** Use this option to clear the value entered in the field and revert to defaults.

 **Note:**

Your plain text string and your expression are maintained so that you can toggle between the two options without losing any data.

Configure the Due Date and Priority

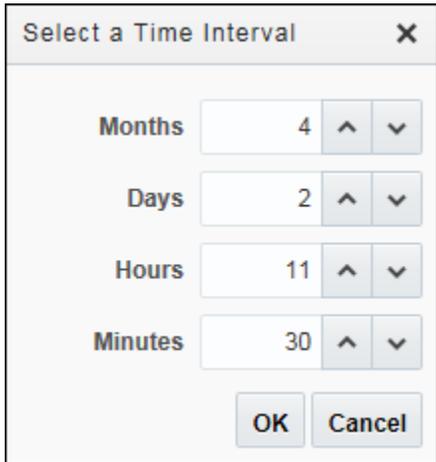
You can specify a due date and a priority for a human task. After the due date is reached, the human task is marked as overdue.

To configure the due date and priority:

1. In the General tab in the implementation pane, locate the due date and priority fields.
2. In the **Due Date** field enter an interval to specify the amount of time the assignee has to complete the instance after the human task is triggered. Select one of the following options:
 - *Manually*: Use the format `##M##d##h##m`. For example:
 - One hour and thirty minutes: 1h30m
 - One day: 1d
 - Four months, two days, eleven hours and thirty minutes: 4M2d11h30m
 - *Expression Editor*: Click the **Expression Editor** to specify an expression to calculate due date of the human task.

The Expression Editor dialog box opens. See [Work with Expressions](#).

- *Interval*: Click **Interval** to specify the due time using the Select a Time Interval dialog box.



- *Reset*: Click **Reset** to clear the value entered in the field and revert to defaults.

 **Note:**

A task due date is different from a process due date. Setting a due date for a process does not automatically set a due date for the task.

3. From the **Priority** drop-down list, select a priority.

Available options are:

- High
- Normal
- Low

These options enable you to sort the Workspace task list based on the task priority. Additionally, in the Workspace task list, high priority tasks are marked with a red exclamation mark.

Bypass the Approval Chain

You can bypass the approval chain for a specified action on approval type human tasks. For example, if you set the Approvers to be *All Assignees in sequence* and the second of four approvers rejects the task, you can use this feature to bypass the remaining two approvers.

To bypass the approval chain:

1. In the General tab on the properties pane, locate the **Skip Approval on** check box.
2. Select the **Skip Approval on** check box to activate the drop down menu.

3. Select the action that you want to skip approval on.

If you add or remove an action in the **Actions** text field, the drop-down menu for skipping approvals automatically updates to reflect the addition or deletion.

Configure Reminders

You can send reminders to the assignees of a human task. You can specify the number of reminders to send, and the event that triggers the reminder.

To configure reminders:

1. Click the **Reminders** tab on the implementation pane.
The Reminders tab appears.
2. From the **Reminder** drop-down list, select a number of times to send reminders to complete the human tasks to its assignees.

Available options are:

- No Reminder
 - Remind Once
 - Remind Twice
 - Remind Three Times
3. If you selected to send multiple reminders, select an interval to wait between reminders.

You can specify this interval either:

- *Manually*: Use the format ##M##d##h##m. For example:
 - One hour and thirty minutes: 1h30m
 - One day: 1d
 - Four months, two days, eleven hours and thirty minutes: 4M2d11h30m
- *Interval*: Click **Interval** to specify the due time using the Select a Time Interval dialog.

4. If you selected to send reminders, select an event to trigger the first reminder from the **When** drop-down list.

Available options are:

- **Before Expiration**: send a reminder before the specified expiration time is reached. After the expiration date is reached the human task doesn't appear on the task list.
- **After Assignment**: send a reminder immediately after assigning the human task to a specific user.
- **Before Due Date**: send a reminder before the specified due date for the human task is reached. After the due date the human task is marked as overdue, but you can still access it from the task list.

Configure Task Expiration, Renewal, or Escalation

You can configure a human task to never expire, to expire after a certain time, to renew the expiration time, or to escalate after a certain time passes.

To configure an action to perform on a specific deadline:

1. Click the **Escalation and Expiration** tab on the implementation pane.
The Escalation and Expiration tab appears.
2. Use the radio buttons to specify if you want the human task to **Never expire**, **Expire**, **Renew**, or **Escalate**.
 - **Never expire**: the human task doesn't expire and if no user completes it, it remains in the users task list for an indeterminate period of time.
 - **Expire**: the human task expires after the specified time and is no longer accessible from the task list.
 - **Renew**: when the specified time passes, the expiration date is extended for one more period until it reaches the specified amount of renewals allowed.

- **Escalate:** when the specified time passes, the human task is escalated to the specified escalation levels.
3. If you chose the human task to expire, renew, or escalate, specify the interval to wait before performing this action.

You can specify the interval to wait using one of the following methods:

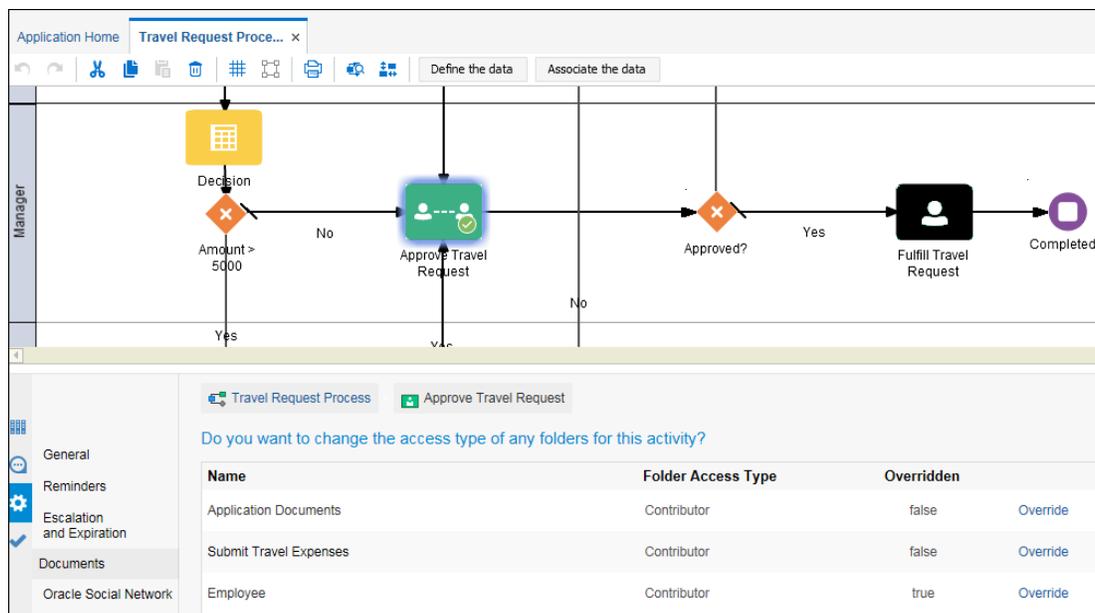
- *Manually:* Use the format ##M##d##h##m. For example:
 - One hour and thirty minutes: 1h30m
 - One day: 1d
 - Four months, two days, eleven hours and thirty minutes: 4M2d11h30m
- *Expression Editor:* Click the **Expression Editor** to specify an expression to calculate due date of the human task.
The Expression Editor dialog box opens. See [Work with Expressions](#).
- *Interval:* Click **Interval** to specify the due time using the Select a Time Interval dialog box.

4. If you renew the human task, in the **Maximum Renewals** field, specify the maximum number of times to renew the human task.
5. If you escalate the human task:
 - a. In the **Maximum Escalation Levels**, specify how high in the management chain you want to escalate the human task.
 - b. In the **Highest Escalation Title**, specify to what job title you want to escalate during repeated escalations. For example, *Manager* or *Director*. This is a free-form text field.

Override Documents Folder Access

You can override the default folder access for each specific human task in a business process.

The Documents tab of the Implementation pane lists the folders that are available for the human task and allows the user to override the default settings, as shown.



To override access to a documents folder for a specific task:

1. Open the process you want to edit and make sure you're in **Edit** mode if you're editing a shared application.
2. In the process editor, click the human task, select the **Menu** icon, and then select **Open Properties**.

The Properties pane opens at the bottom of the window.

3. Click **Documents** to list the available folders for this task.
4. Click the **Override** link that's located next to the folder you want to change access on.

The Override Folder Access dialog box opens.

The 'Override Folder Access' dialog box is shown. It has a title bar with a close button. Inside, there is a 'Folder name' field containing 'Submit Travel Expenses'. Below it is an 'Override' section with a drop-down menu currently set to 'Contributor'. An 'Override' button is located at the bottom right of the dialog.

5. Select one of the following access types from the drop-down list:
 - Contributor
 - Downloader
 - Viewer
 - None
6. Click **Override** to save your changes.
7. Close the implementation pane with your changes saved by clicking the **Collapse Pane** icon on the top right corner.

Specify Task Actions Shown to Users

Specify the task actions available to end users for a human task in runtime.

1. In a structured process, select a human task (approve or submit), and choose **Open Properties**.
2. In the implementation pane, click  under Action.
3. From the Actions Rules dialog, choose actions to display to users completing the task.

Note that administrators and process owners see all options, regardless of these settings.

| Actions Rules | |
|---------------|-------------------------------------|
| Actions | Visible? |
| Delegate | <input checked="" type="checkbox"/> |
| Escalate | <input checked="" type="checkbox"/> |
| Request Info | <input checked="" type="checkbox"/> |
| Reassign | <input checked="" type="checkbox"/> |
| Release | <input checked="" type="checkbox"/> |
| Resume | <input checked="" type="checkbox"/> |
| Suspend | <input checked="" type="checkbox"/> |
| Withdraw | <input checked="" type="checkbox"/> |

Admins and Process Owners will always have the default settings.

OK

The actions you deselect from this list will not be displayed to users. Note that administrators and process owners see all options, regardless of these settings.

4. Click **OK** and verify the new task actions settings in runtime.

Assign Custom Properties to Structured Process Activities

Interested in assigning values to process activities for use in runtime? Begin by defining properties and assigning them fixed values in design time, then use a process metadata REST API endpoint in runtime to access the assigned values.

For example, for an inspection process you might define three custom properties, to track each phase or activity: Started, Submitted, and Complete. In runtime, you might use these values to sort or order activities in a report, or to drive behavior in some way.

1. Open a process application. In the process editor, define custom properties for the process. In design time, you define custom properties at the process level.

Click the **Custom Properties** # on the process editor toolbar and enter property name/description pairs. You can assign any name to a property.

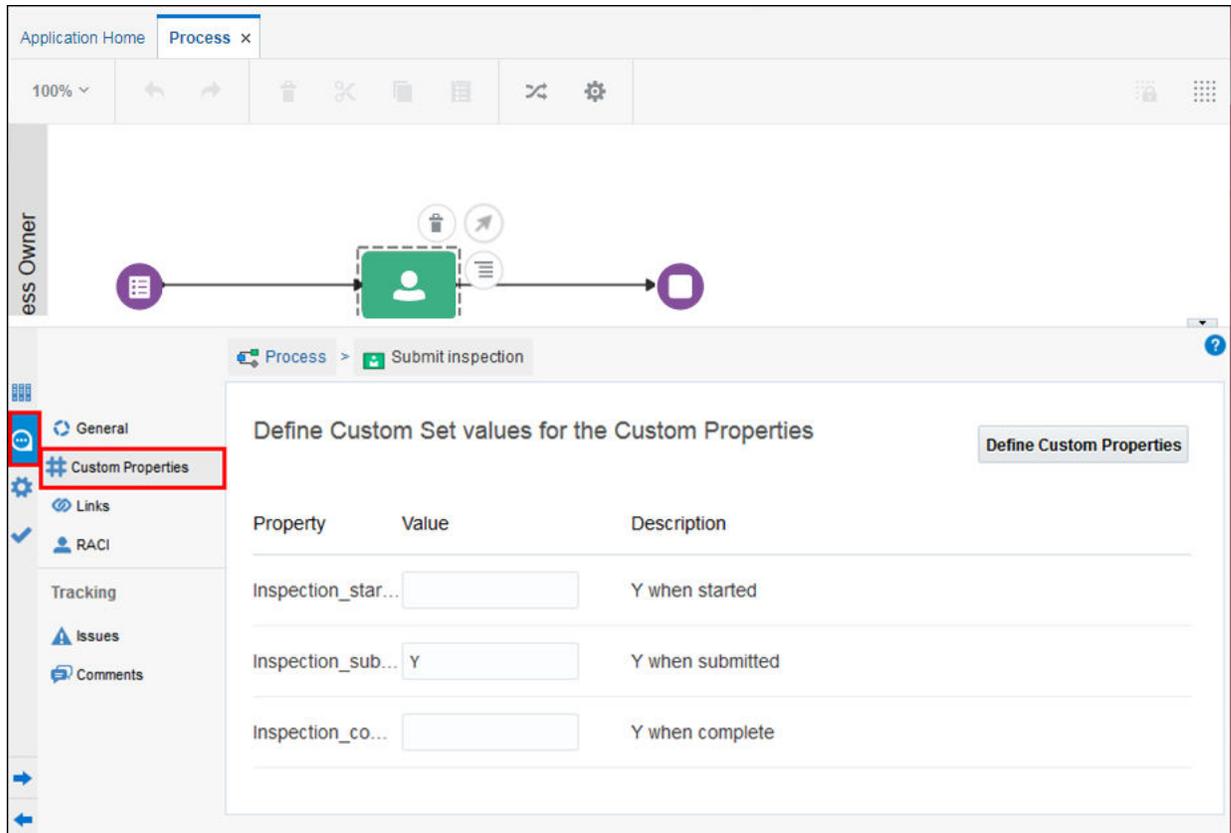
| Property Name | Description | |
|----------------------|------------------|---|
| Inspection_started | Y when started | X |
| Inspection_submitted | Y when submitted | X |
| Inspection_complete | Y when complete | X |
| Property Name | Description | |

2. Assign property values to selected activities.

With properties defined for the process, now assign custom values to activities. Values are fixed and cannot be changed dynamically at runtime

- a. In the property pane's side pane, select Business Properties, then Custom Properties. The properties you defined are displayed.
- b. Enter values for the custom properties. If needed, click Define Custom Properties to add or edit properties.

You can enter string values only in custom properties. You can leaves values empty.



3. In runtime, query process definitions using the REST API and access custom values.
 - After activating the process application and assigning members to its swimlane roles to access the process definitions output, view the custom metadata created in a REST call.
 - See the Process Definition ID/process-metadata endpoint in the Oracle Integration REST API. Along with process application metadata and activities, this endpoint provides metadata for each activity, including assigned custom values. If no value was assigned to a custom property in design time, the property is not displayed in the metadata.
 - Note that you must be assigned a swimlane role that provides access to the process application (Process Owner, for example) to see process definition metadata.

Customize Notification Emails for Human Tasks

Keep your users informed about their task assignments and the progress of a process. You can easily configure notification emails for human tasks and create templates for those notifications.

Topics:

- [About Notification Email and Templates](#)
- [Configure Email Notifications](#)
- [Manage Email Templates](#)

- [Configure Email Templates](#)
- [Use Handlebar Helpers](#)

About Notification Email and Templates

You can configure Process to send a notification email to the task assignee when an event such as assignment, approval, escalation, reminder, and reassignment occurs. A notification email contains outcomes defined for a task, such as Approve or Reject. Assignees can click on these outcomes to complete tasks using their email clients, without having to sign in to the application.

Note:

After you configure the email notifications in design time, you need to enable the email notifications in runtime, see [Enable Email Notifications](#). You can then view the notification logs and also resend email notifications to all or some of the original recipients, see [View and Resend Email Notifications](#).

You can also configure and use customized notification emails for human tasks. You have an option to create customized email templates that include the following information:

- Payload
- Task
- Links to perform an action on a task
- Task comments

About Actionable Emails

You can configure a notification email with or without actions. An email with actions contains outcomes defined for a task, such as Approve or Reject. You can click on these outcomes to act on the task using your email client. In addition, you can add comments and attachments to the task.

WARNING:

Any user who has access to the email can perform the action so be careful who the notification email is sent to.

About Default Email Templates

If you enable email notification for a human task, you can either select a default template or create a new one for your use. By default, the following two email templates are available:

- Default (With Actions) – Contains outcomes defined for a task, such as Approve or Reject. You can click on these outcomes to act on the task using your email client, without having to sign in to the application. In addition, you can add comments and attachments to the task.

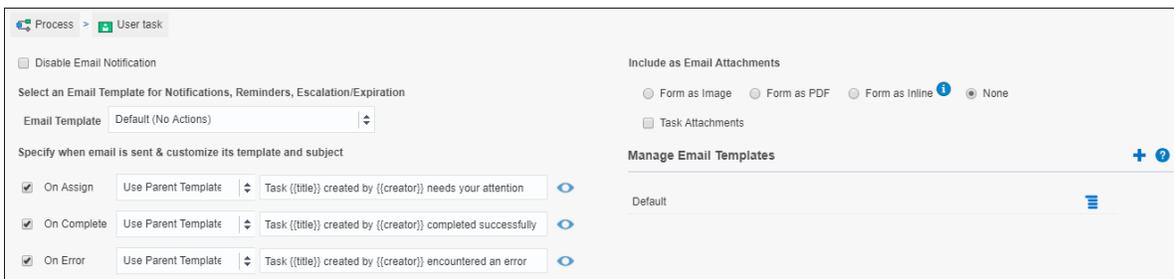
- **Default (No Actions)** – Contains a link that lets the end user sign in to the application, view the task, and complete it. If the end user is viewing the email on a mobile device and clicks the link, then the mobile app launches by default, if it's installed.

Configure Email Notifications

Configure email notifications and specify when an email is sent to the task assignee. You can also customize email notifications so they contain all the information recipients need to easily recognize and act on them.

To configure email notifications:

1. Open a process in the process editor.
2. Select a task, click **Menu** , and select **Open Properties**.
3. Click **Implementation** and select the **Notification** tab.



Note:

If you've configured the human task to use an external UI instead of a web form, the options to include the form as image, PDF, or inline are not available on the **Notification** tab.

4. If needed, disable or re-enable email notifications. Note that notifications are enabled by default. Note that disabling the notification for a task won't disable reminder, escalation, or expiration emails. Instead, the selected email template will be used for those events.
5. In the **Email Template** field, select an email template that you want to use for the task-related emails.

Note:

The email template you select here is also used for reminder, escalation, and expiration emails.

6. Specify when the email notification is sent, for example when a task is assigned or completed, or when an error occurs. By default, **On Assignment**, **On Completion**, and **On Error** are selected. Use the drop down to determine the template for each notification email. The Use Parent Template is selected by default.

The parent template uses the template you selected before. You can either use the same template or select any other template. To manage email templates, see [Manage Email Templates](#).

- Click **Preview**  to see what the subject will look like in your Inbox.
You can edit the subject line of the emails using valid payload attributes. See [Payload Examples](#).
- From the **Include as Email Template** options, optionally specify attachments to include with the email. You can include the form as an image attachment, a PDF attachment, or within the email (Inline).

 **Note:**

If you encounter issues with displaying form as an inline image in your email client, use a custom email template that includes the form payload data. This way, task assignees are able to view the required content or data and act upon the task accordingly, without having to log in. The format for adding payload to your custom template is detailed here: [Payload Examples](#).

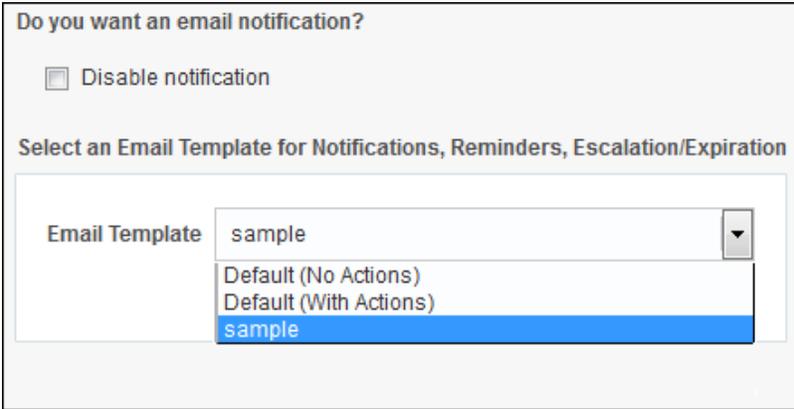
- Optionally, you can select **Task Attachments** to include files that users have uploaded to tasks.

Manage Email Templates

Use a customized notification email for human tasks that includes basic task information, form payload values, and comments, and that optionally lets end users perform a task action without having to sign in to the application. Create customized notification emails using the **Notification** options in the Properties pane.

To assign an email template to a human task:

- Open a process in the process editor.
- Select a human task, click **Menu** , and select **Open Properties**.
- Click **Implementation** and select the **Notification** tab.



Do you want an email notification?

Disable notification

Select an Email Template for Notifications, Reminders, Escalation/Expiration

Email Template: sample

- Default (No Actions)
- Default (With Actions)
- sample

- If needed, disable or re-enable email notifications. Note that notifications are enabled by default. To disable, select the **Disable notification** check box to stop email notifications during task assignment. Note that disabling the notification for a task won't disable

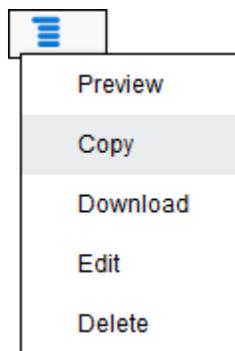
reminder, escalation, or expiration emails. Instead, the selected email template will be used for those events.

- In the Email Template field, select an email template that you want to use for the task-related emails.

Optionally, create a new email template for your use. Click **Add**  to the right of **Manage Email Templates**. See [Configure Email Templates](#).



Optionally, copy, edit, preview, or download an existing email template for your use. Click **Options**  to the right of the email template that you want to use.



Available options include:

| Option | Description |
|-----------------|---|
| Preview | Open the email template in the Preview Email Template dialog. You can view how the email appears to end users. The payload information isn't displayed in the Preview mode. |
| Copy | Copy and save the email template with another name. |
| Download | Download and save the email template in HTML format to your local drive or select an application to open the file. |
| Edit | Open the Edit Email Template dialog. You can edit your template in the code editor or browse for and upload a template from your local drive. You can also preview the template before finalizing it. This option isn't available for the default template. |
| Delete | Delete the email template. This option isn't available for the default template. Important: Deleting a template that is currently in use deletes all references to the template. |

Configure Email Templates

Create customized email templates and use them to send notification emails for human tasks. Upload email templates from your local drive, edit or copy existing email templates, or download email templates for your use.

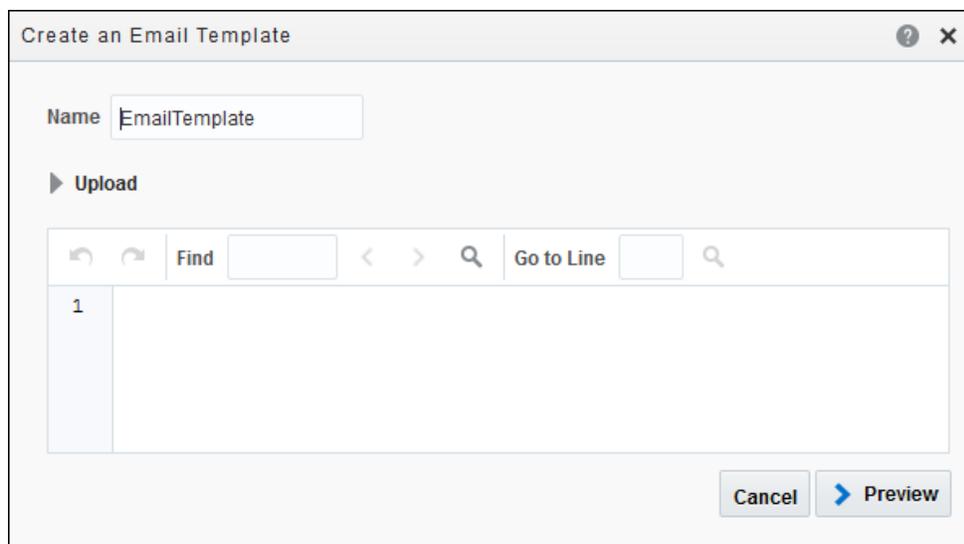
Create an Email Template

You can create email templates by uploading a file with an HTM or HTML extension, or manually entering the HTML markup directly into the code editor provided in the Create an Email Template dialog box.

To create a new email template:

1. Open a process in the process editor.
2. Select a human task, click **Menu** , and select **Open Properties**.
3. Click **Implementation** and select the **Notification** tab.
4. Click **Add**  next to Manage Email Templates.

The Create an Email Template dialog box opens.



5. Enter a name for your email template.
6. Use one of these options to create your email template:
 - You can click **Upload** to browse for and add an existing email template. You can then edit its contents.
 - Alternatively, you can start from scratch by manually entering the HTML markup for your email template. To include data, you'll use mustache templates. Mustache templates are logic-less and work by expanding tags in a template using values provided in an object—the task object in this case. These templates provide an easy way to include data references in an HTML file. The following example shows how to reference the first name and last name using the mustache template:

Template:

```
<p>Hello {{first_name}} {{last_name}}</p>
```

Object:

```
{
  "first_name": "Joe",
  "last_name": "Smith"
}
```

Output:

```
<p>Hello Joe Smith</p>
```

See Mustache and Mustache 5.

The following table provides a description of the **task** related attributes of the task object. Note that the variable names are case-sensitive.

| Variable | Description |
|--------------|---|
| acquiredBy | Name of the user who has acquired the task |
| acquiredById | ID of the user who has acquired the task |
| assignedDate | Date when the task was assigned |
| createdDate | Date of creation of the task |
| creator | Creator of the task |
| creatorId | ID of the creator of the task |
| dueDate | Due date for completing the task |
| endDate | Date when the task must end |
| fromUser | Creator or user who reassigned the task |
| fromUserId | ID of the user from whom the task was acquired |
| longSummary | Detailed description of the task |
| outcome | Task outcome |
| ownerGroup | Group to which the task owner belongs |
| ownerRole | Role of the task owner |
| ownerUser | User who owns the task |
| ownerGroupId | ID of the group to which the task owner belongs |
| ownerRoleId | ID of the role of the task owner |
| ownerUserId | ID of the user who owns the task |
| priority | Task priority |
| payload | Map of task payload |
| priorityNum | Priority number of the task |
| shortSummary | Short summary of the task |

| Variable | Description |
|------------------|---|
| startDate | Start date of the task |
| state | Current status of the task |
| taskId | Task ID |
| taskNumber | Task number |
| taskDefinitionId | Definition ID of the task |
| title | Title of the task |
| updatedBy | Name of the user who has updated the task |
| updatedById | ID of the user who has updated the task |
| updatedAt | Date when the task was updated |

The following table provides a description of the **process** related attributes of the task object:

| Variable | Description |
|----------------|---|
| instanceId | ID of the process instance in which the task is present |
| processId | ID of the process in which the task is present |
| processName | Name of the process in which the task is present |
| processVersion | Version of the process in which the task is present |

The following table provides a description of the **application** related attributes of the task object:

| Variable | Description |
|-------------|---|
| assignee | Full name of the task assignee |
| currentYear | Current year |
| logo | Oracle logo |
| url | URL for accessing the task details in runtime |

The following table provides a description of the **comments** related attributes of the task object:

| Variable | Description |
|------------|--|
| comments | List of comments |
| commentStr | Actual comment |
| updatedBy | Name of the user who updated the comment |
| updatedAt | Date the comment was updated |

The following table provides a description of the **action** related attributes of the task object:

| Variable | Description |
|-------------------|---|
| actionDisplayName | Display name of the action. For example, Approve, Reject. |
| actionName | Name of the action defined in human task action. For example, APPROVE, REJECT. Note that the action name is case sensitive and must match the task outcome. |
| actions | List of actions. |
| url | URL for performing the action. |

Examples:

The following table provides examples of how to define tags using mustache templates:

| To define a... | Example |
|----------------|--|
| Variable | <code>{{title}}, {{assignee}}</code> |
| Section/List | <code>{{#comments}} {{commentStr}} {{updatedBy}} at {{updatedDate}} {{/comments}}</code> |
| Null check | <code>{{#dueDate}} Due Date: {{dueDate}} {{/ dueDate}}</code> |
| HTML string | <code>{{{actions}}}</code> |

| To define a... | Example |
|----------------|---------|
|----------------|---------|

Payload

Use the camelCase naming convention for all business objects and form keys used in the payload to get the desired email template.

Payload Example for a Form with Business Object

```

{{#payload}}
  {{#financialApprovalForm}}
    <td
valign="top">{{contractCategory}}</td>
    <td
valign="top">{{creditLimit}}</td>
      {{#contractDataObject}}
        <td
valign="top">{{customerName}}</td>
        <td
valign="top">{{contractStartDate}}</td>
        <td
valign="top">{{creditLimit}}</td>
      {/
contractDataObject}}
    {/financialApprovalForm}}
  {/payload}}

```

You can view the structure of payload in the Data pane on the web form page.

| Finance Approval | |
|-----------------------|-------------------------|
| NAME | TYPE |
| financialApprovalForm | (FinancialApprovalForm) |
| contractCategory | (string) |
| contractDataObject | (ContractDataObject) |
| customerName | (string) |
| contractStartDate | (dateTime) |
| contractEndDate | (dateTime) |
| creditLimit | (double) |
| approvalNotes | (string) |
| region | (string) |
| creditLimit | (double) |

Payload Example for a Form with Text and Boolean Data

```

{{#payload}}
  {{#travelRequestForm}}
    {{#form}}
      <td valign="top">{{name}}</td>
      <td
valign="top">{{customerName}}</td>

```

| To define a... | Example |
|----------------|---------|
|----------------|---------|

```

        <td
valign="top">{{totalAmount}}</td>
        <td
valign="top">{{country}}</td>
        <td valign="top">{{city}}</td>
        <td
valign="top">{{purposeOfVisit}}</td>

        {{/form}}
    {{/travelRequestForm}}
{{/payload}}

```

To get the structure of the payload, see the incoming webform dataObject in the Data Association page of the human task.

| Submit Request | |
|---------------------|----------------------|
| NAME | TYPE |
| ▲ travelRequestForm | (TravelRequestForm) |
| ▲ form | (TravelRequestForm) |
| abc name | (string) |
| 📅 date | (date) |
| abc totalAmount | (string) |
| ▲ country | (CountryType) |
| USA | (CountryType) |
| INDIA | (CountryType) |
| ▲ city | (CityType) |
| OPTION_3 | (CityType) |
| OPTION_1 | (CityType) |
| OPTION_2 | (CityType) |
| ▲ purposeOfMsit | (PurposeOfVisitType) |
| INTERNAL | (PurposeOfVisitType) |
| CUSTOMER | (PurposeOfVisitType) |
| TRAINING | (PurposeOfVisitType) |
| SEMINAR | (PurposeOfVisitType) |
| abc customerName | (string) |
| abc justification | (string) |
| 99E amount | (double) |

To define a...

Example

Payload Example for Form with Array Data and Business Object

```

{{#payload}}
  {{#basicForm}}
    {{#form}}
      <td
valign="top">{{firstName}}</td>
      <td
valign="top">{{lastName}}</td>
        {{#expensesItem}}
          <td
valign="top">{{description}}</td>
          <td
valign="top">{{amount}}</td>
        {/expensesItem}
      {/form}
    {{#invoice}}
      <td valign="top">{{id}}</td>
    {/invoice}
  {/basicForm}
{{/payload}}

```

| Approve Request | |
|-----------------|----------------------|
| NAME | TYPE |
| basicForm | (BasicForm) |
| form | (BasicForm) |
| firstName | (string) |
| lastName | (string) |
| expensesItem | (ExpensesItemType[]) |
| total | (double) |
| invoice | (Invoice) |
| id | (string) |

| To define a... | Example |
|----------------|---|
| Actions | <p>You can define actions using a string, a list, or a map.</p> <ul style="list-style-type: none"> Actions as a string: In this case, actions are defined as per the default email. You can't modify the look and feel of the links. For example: <code>{{actions}}</code> Actions as a list: Use it if you want to iterate over each action to modify the look and feel of links. The same setting is applied on all the links. For example: <pre> {{#actionsList}} {{actionDisplayName}} {{/actionsList}} </pre> Actions as a map: Use it if you want to customize each action link. For example: <pre> {{#actionsMap}} {{#APPROVE}} {{actionDisplayName}} {{/APPROVE}} {{#REJECT}} {{actionDisplayName}}</ a> {{/REJECT}} {{/actionsMap}} </pre> |

 **Note:**

In Mustache, by default all variables escape in HTML. If you want to render unescaped HTML in an email's subject (that is, with special characters), you need to use triple mustache, for example `{{{title}}}`, or use `&`, for example `{{&title}}`.

7. Click **Preview** to view your template before finalizing it.

The screenshot shows an email interface for Oracle Cloud. At the top left is the Oracle logo and the word "Cloud". Below this, it says "Hello Assignee," followed by "Task Title of the task requires your attention . [View Online](#)".

The main content is divided into three sections, each with a grey header bar:

- Task Information**: A table with two columns. The first column lists labels: "From:", "Summary:", "Priority:", "Created On:", and "Due Date:". The second column lists corresponding values: "From User Name", "Short Summary", "High", "Created Date", and "Due Date" (where "Due Date" is highlighted in red).
- Data**: This section is currently empty.
- Comments**: A list of two comments. The first is "Comment String 1" followed by "By - User1 at Update Date". The second is "Comment String 2" followed by "By - User2 at Update Date".

At the bottom of the email body, there is a footer with copyright information: "Copyright 2016. Oracle and/or its affiliates. All rights reserved." and links for "About Oracle", "Legal Notices and Terms of Use", and "Privacy Statement". Below this is a disclaimer: "This is a system generated message. Do not reply to this message. You are receiving this email as a result of your current relationship with Oracle Cloud. General marketing opt-out preferences have been over-ridden to ensure that you receive this email."

At the bottom right of the email interface, there are three buttons: "Cancel", "Previous" (with a left arrow), and "Finish" (with a checkmark).

 **Note:**

In preview mode, the payload and action map sections don't show the data.

The actual email looks something like this:

Hello Amy Silver ,

Task New Hire Approval requires your attention . [View Online](#)

HOLD REJECT

Task Information

| | |
|-------------|---|
| From: | James Cooper |
| Summary: | Go through the candidate profile to make a decision on hiring |
| Priority: | High |
| Created On: | 2017-02-02 at 08:33:56 |
| Due Date: | 2017-02-05 at 08:33:56 |

Data

Candidate information

| | |
|-------------|------------------|
| Name: | John Smith |
| Department: | Computer Science |

Education

| | |
|-------------|----------------------|
| University: | State University |
| Start Date: | 2010-08-05T07:00:00Z |
| End Date: | 2014-05-23T07:00:00Z |

Comments

Copyright 2016. Oracle and/or its affiliates. All rights reserved. [About Oracle](#) | [Legal Notices and Terms of Use](#) | [Privacy Statement](#)

This is a system generated message. Do not reply to this message. You are receiving this email as a result of your current relationship with Oracle Cloud. General marketing opt-out preferences have been over-ridden to ensure that you receive this email.

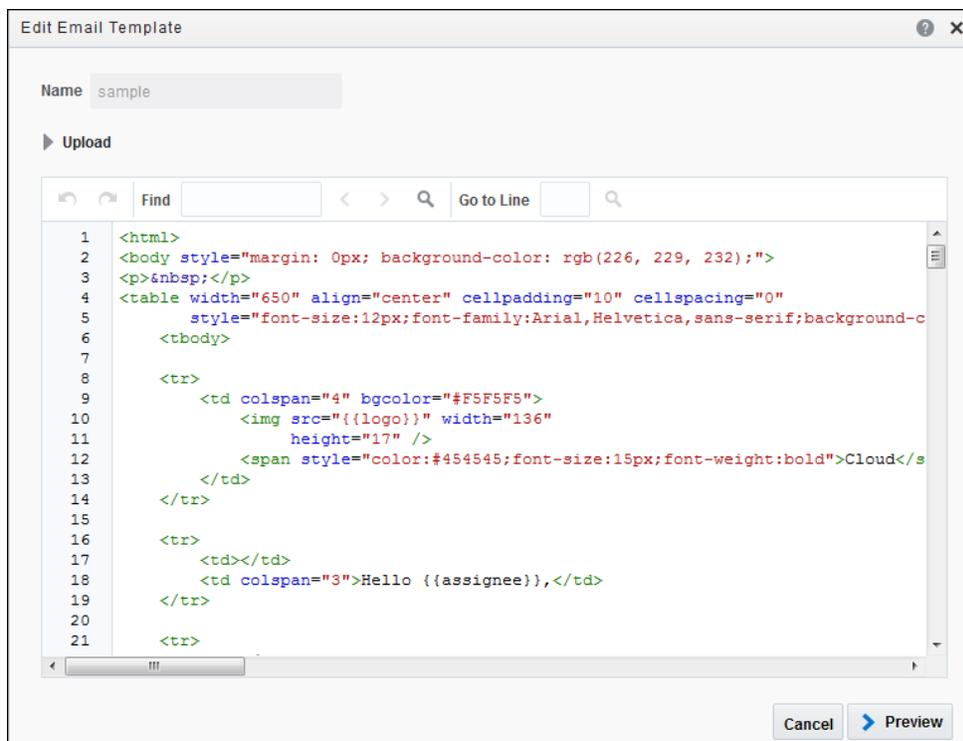
Edit an Existing Template

You can edit an existing template using the Edit Email Template dialog box.

To edit an existing email template:

1. Open a process in the process editor.
2. Select a human task, click **Menu** , and select **Open Properties**.
3. Click **Implementation** and select the **Notification** tab.
4. Click **Options**  next to the email template that you want to edit.

The Edit Email Template dialog box opens.



5. Edit the HTML markup as required.

Alternatively, click **Upload** to browse for and upload an existing email template to replace the current one.

6. Click **Preview** to view your template before finalizing it.

Use Handlebar Helpers

Optimize the configuration of your email templates by using handlebar helpers. Add logic into your email templates by using conditional helpers. You can also use other handlebar helpers such as string, number, and date formats.

Conditional Helpers

Some common conditional function helpers with examples are listed in the table below:

| Function | Description | Example |
|----------|-----------------------------------|--|
| eq | Checks if two elements are equal. | <p>This example shows how to modify an email's subject based on a task's outcome (approve or reject)</p> <pre> Task {{taskNumber}} created by {{creator}} is {{#eq outcome "APPROVE"}}approved successfully{{else}} rejected{{/eq}} </pre> |

| Function | Description | Example |
|----------|---------------------------------------|--|
| neq | Checks if two elements are not equal. | <p>This example shows how to compare a business object's value and modify an email.</p> <pre>{{#eq outcome "APPROVE"}} <p>Hello {{payload.expenseApprovalForm.firstName}}, your expense report for payload.expenseApprovalForm.totalAmount}} was approved!</p> {{else}} <p>Hello {{payload.expenseApprovalForm.firstName}}, Sorry your expense report for {{payload.expenseApprovalForm.totalAmount}} was rejected!</p> {{/eq}}</pre> <pre>{{#neq payload.expenseApprovalForm.team "sales"}} Sorry, this item is only eligible for Sales team {{/neq}}</pre> |

| Function | Description | Example |
|----------|-----------------------|--|
| gt | Greater than operator | <p>The example below shows how to mark an email's subject as <i>Important!</i> if the total amount of an expense report is greater than \$10,000.</p> <pre>{{#gt payload.expenseApprovalForm.totalAmount 10000}}Important!{{/gt}} Expense report by {{creator}} for amount {{payload.expenseApprovalForm.totalAmount}} for approval</pre> <p>The example below compares an array object's size and conditionally shows expense items only if the number of items is greater than 1.</p> <pre>{{#gt payload.expenseApprovalForm.items.length 1}} <h3>Expense Items</h3> <table> <thead> <tr> <th>Item</th> <th>Amount</th> </tr> </thead> <tbody> <tr> <td>item1</td> <td>400</td> </tr> <tr> <td>item2</td> <td>340</td> </tr> </tbody> </table> {{/gt}}</pre> |

| Function | Description | Example |
|----------|-----------------------------------|--|
| and | Checks if two conditions are true | <p>This example shows how to add a special category section into the email's content if the travel category is critical <i>and</i> if the traveler is an executive.</p> <pre> {{#and payload.travelApproval.isCritical payload.travelApproval.traveler.isExecutive}} <h3>Special Category Traveler</h3> {{else}} <h3>Normal Traveler</h3> {{/and}}</pre> |

String, Date, and Number Format Helpers

Some common string, date, and number format helpers with examples are listed in the table below:

| Function | Description | Example |
|-----------------|--|--|
| capitalizeFirst | Capitalizes the first character of the value. | <pre> <pre> {{capitalizeFirst value}} </pre></pre> <p>If value is "string.example", the output will be "String.example".</p> |
| center | Centers the value in a field of a given height. | <pre> <pre> {{center value size=19 [pad="char"] }} </pre></pre> <p>If value is "String.example", the output will be "String.example ".</p> |
| cut | Removes all values of arg from the given string. | <pre> <pre> {{cut value [" "]}} </pre></pre> <p>If value is "String with spaces", the output will be "Stringwithspaces".</p> |

| Function | Description | Example |
|------------|---|---|
| join | Joins an array, iterator or an iterable with a string. | <pre><pre> {{join value " // " [prefix=""] [suffix=""]}} </pre></pre> <p>If value is the list ['a', 'b', 'c'], the output will be the string "a // b // c".</p> |
| lower | Converts a string into all lower case. | <pre><pre> {{lower value}} </pre></pre> <p>If value is 'String Helper Example', the output will be 'string helper example'.</p> |
| upper | Converts a string into all upper case. | If value is 'Hello', the output will be 'HELLO'. |
| replace | Replaces each substring of a string that matches the literal target sequence with the specified literal replacement sequence. | <pre><pre> {{ replace value "... "example" }} </pre></pre> <p>If value is "String ...", the output will be "String example".</p> |
| dateFormat | Returns the date in the specified format. Parameters: <ul style="list-style-type: none"> • full • long • medium • short • pattern | <pre><pre> {{dateFormat date ["format"] [format="format"][tz=timeZone timeZoneId]}} </pre></pre> <p>Examples:</p> <ul style="list-style-type: none"> • {{dateFormat date ["full"]} output: Tuesday, June 19 2012 • {{dateFormat date ["long"]} output: June 19, 2012 • {{dateFormat date ["medium"]} output: Jun 19, 2012 • {{dateFormat date ["short"]} output: 6/19/12 |

| Function | Description | Example |
|--------------|---|---|
| numberFormat | Returns the number in the specified format. Parameters: <ul style="list-style-type: none">• integer• percent• currency• pattern | <pre><pre> {{numberFormat number ["format"] [locale=default]}} </pre></pre> |

**Note:**

Handlebars is a superset of Mustache and Mustache templates are compatible with Handlebars.

See [Handlebars](#).

7

Develop Dynamic Processes

A key part of your application is its business process or processes. When developing a dynamic process, first determine the activities and stages that may occur. You can then build out related elements, such as web forms for human interaction, milestones, and the data flow between elements.



Note:

In Oracle Integration, you can create two types of processes: **dynamic** (described below), and **structured**, described in [Develop Structured Processes](#).

Topics:

- [How do dynamic processes work in process applications?](#)
- [Ready to create a dynamic process?](#)
- [Model a Dynamic Process in Design Time](#)
- [Work with Dynamic Processes in Runtime](#)

How do dynamic processes work in process applications?

Use the dynamic process editor as a canvas to define, configure, and activate dynamic processes for process applications.

Automate unpredictable processes that require expert knowledge or changing circumstances

Many business processes don't follow a structured or sequential path. They might have many possible activities that experts could act on as needed, given the situation. Which tests should a doctor administer? Based on test results, does the patient need surgery? Which documents must be approved before a treatment stage can begin?

If you're new to dynamic processes, [start here to learn the basics](#) by creating, configuring, and activating a dynamic process.

Want to learn more about design time modeling details, such as working with simple expressions or plan item lifecycles? See [Model a Dynamic Process in Design Time](#). Wondering how knowledge workers use activated dynamic processes? See [Work with Dynamic Processes in Runtime](#).

Combine dynamic and structured processes in process applications

Combine dynamic and structured processes in a process application to take advantage of their strengths. For example, orchestrate an overall unpredictable process with a dynamic process that accommodates a wide variety of activities, but use structured processes for activities that follow a structured path.

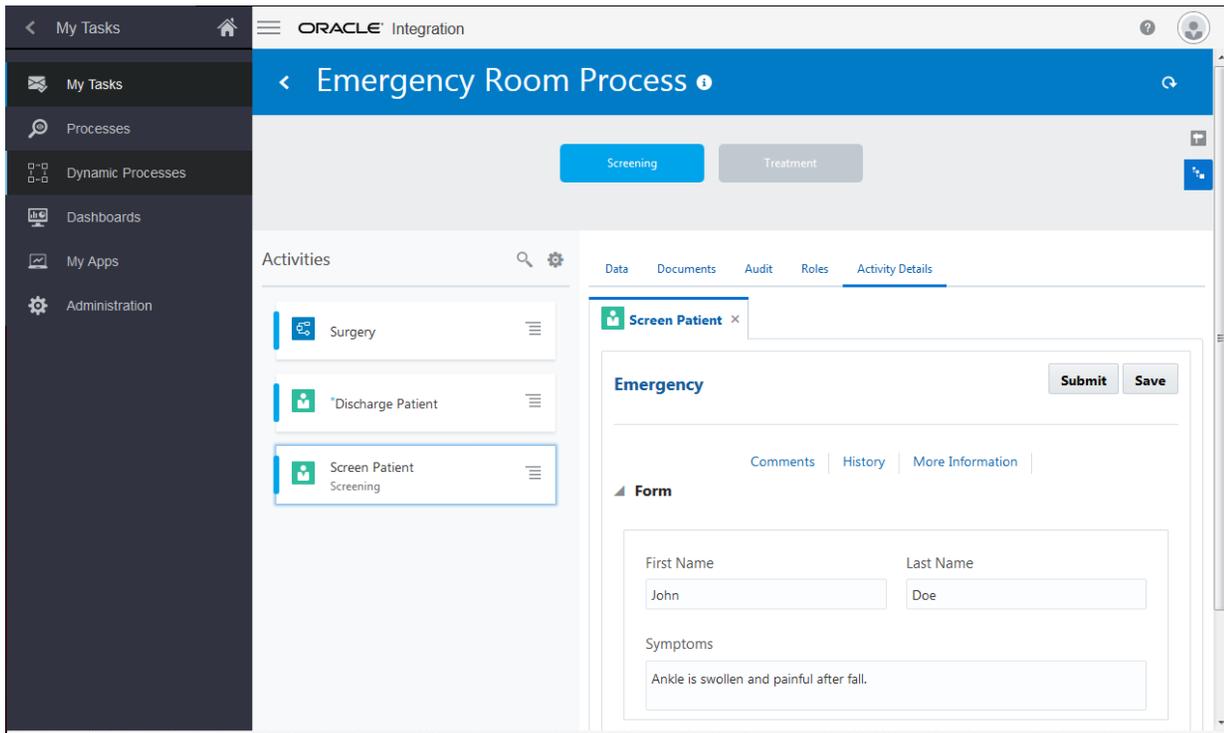
Ready to create a dynamic process?

Imagine you're in charge of an emergency room.

Your staff needs a way to automate the unpredictable process of screening, treating, and discharging patients who enter your doors. Each case is different. Some steps need to happen in order, some depend on expert input or the situation, and others could be performed at any time. You need a dynamic process.

Let's break the example down:

- A nurse starts a new patient by entering name information in a form.
- A nurse performs screening for the patient.
- A doctor diagnoses and treats the patient. If needed, the patient goes to surgery.
- A nurse discharges the patient.



Start here by working through basics and then adding to your dynamic process application.

- [Learn dynamic processing basics](#)
- [Take your dynamic process to the next level](#)

Learn how to use a dynamic process in a structured process and vice versa. See [Mix and match processes](#).

Learn dynamic processing basics

Using an emergency room example, let's explore the entire dynamic process development life cycle—from creating a dynamic process, modeling its activities and properties, activating it on the server, and using it in runtime as a knowledge worker.

- [Create a Dynamic Process](#)
- [Add Human Task Activities and Stages](#)
- [Create Web Form Presentations for the Dynamic Process](#)
- [Set the Process Start](#)
- [Set Presentations for the Human Task Activities](#)
- [Configure Data Association for the Human Task Activities](#)
- [Test Activate the Application](#)
- [Try Out the Dynamic Process Application in Runtime](#)

Create a Dynamic Process

First, let's create a process application (Emergency) to house your dynamic process, then create the dynamic process itself.

1. On the Oracle Integration Home page, click **Processes** in the navigation pane.
2. Select **Process Applications** from the Processes navigation pane. On the Process Applications page, click **Create**.
3. In the Create Process Applications page, select **Create an Application**, and then click **Create**.
4. Enter `Emergency` in the **Name** field, select a space from the **Space** drop-down list, leave **Open Immediately** selected, and click **Create**.

The **Application Home** tab opens, with application components shown in the navigation pane.

5. In the Application Home page, click **Create a Dynamic Process** and then click **Start from scratch**.

Optionally, click **Create**, and then select **New Dynamic Process**.

6. In the Create Dynamic Process dialog box, enter `Emergency Process` in the **Name** field, give a suitable description in the **Description** field, leave the **Open Immediately** check box selected, and click **Create**.

A tab for your Emergency Process opens and the dynamic process introduction page appears.

Add Human Task Activities and Stages

A process features activities that humans perform, such as completing and submitting information or approving documents.

Let's start by adding some human tasks that people typically perform in an emergency room. For simplicity, we'll keep this example brief. But keep in mind that a dynamic process can accommodate the complex scenarios of a real world situation such as an emergency room.

1. Add a human task activity.

In the **Add Activity** field, enter `Treat Patient` and click **Add** . You'll see your new activity in the central editing canvas.



Notice the icons:

- The green icon indicates a human task.
- The **Required**  icon displays when the activity is required.
- The warning  icon displays when there are validation issues.

The number in the icon indicates how many validation issues were found. These validation issues are useful as you get immediate feedback while configuring and you can fix the issues inline.

For now, you can ignore these issues and proceed to the next activity.

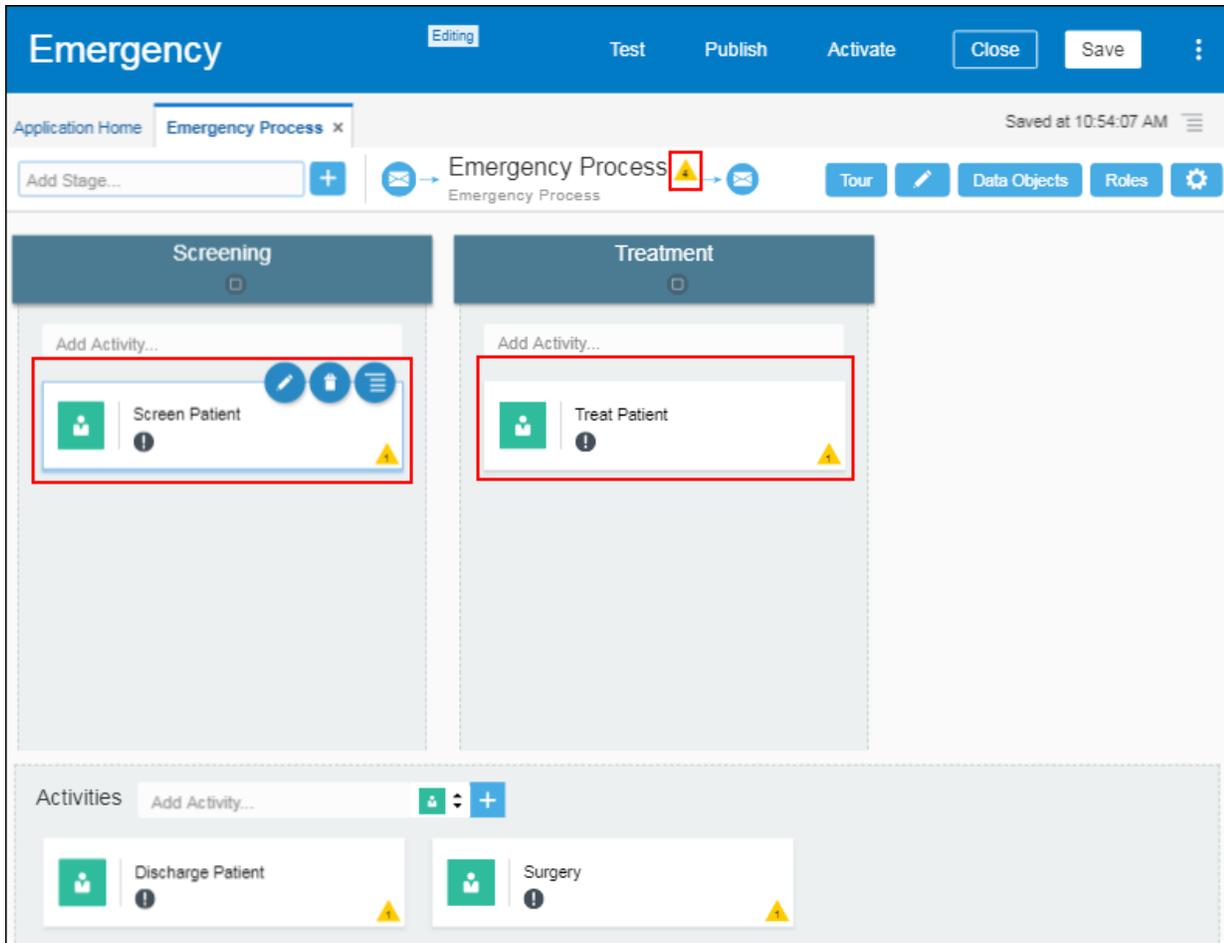
2. Add three more human task activities and name them `Screen Patient`, `Discharge Patient`, and `Surgery`.
3. Add two stages.

In the **Add Stage** field, enter `Screening` and click **Add** . Enter `Treatment` and click **Add**. The two stages open on the central canvas.

Stages enable you to organize activities into phases of a process. (This example is kept simple, but you can include many possible activities in each stage.) Stages can run at the same time or one after another.

4. Drag and drop the screen activity into the Screening stage and the treat activity into the Treatment stage. Leave the discharge and surgery activities where they are.

The two stages are currently set to become available at the same time in runtime. By default, all stages and activities (referred to as *plan items*) become available at the same time in runtime. The other two activities aren't in stages, which means they are available at any point.



Notice the validation icon next to *Emergency Process*. It shows the total number of validation issues in the process. Because you just started to create your dynamic process, and your dynamic process isn't complete, ignore these issues. To hide the validation issues, click **Edit Configuration**  and change the settings for the inline validations.

Important points about human tasks:

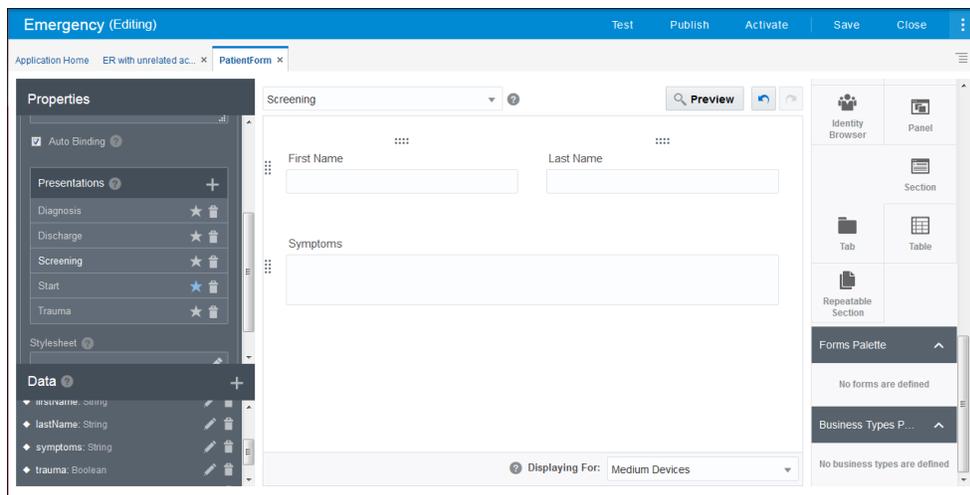
- A human task can be a submit or an approval activity.
- Each human task activity must be associated with a form. The form provides the interface for the task.
- Each human task activity needs an assignee, which could be a user, role, or group.
- Each human task activity has data values that flow in and out of it, referred to as its input and output. For example, a human task's form might display with some fields completed and its output might contain additional or changed fields.

Create Web Form Presentations for the Dynamic Process

Let's create one form with multiple presentations to apply to the human tasks.

1. Create a web form.
 - a. In the Application navigation pane, click **Forms**.

- d. Drag and drop a Text Area control to the form canvas. Select the control, and change its **Name** and **Label** fields to `Symptoms`.



4. Create a Treatment presentation.
 - a. Click outside a form control, and select the **Form** tab.
 - b. Click **Add**  in the Presentations table. In the Select Presentation Type dialog box, select **Clone**.
 - c. In the Create Presentation dialog box, choose **Screening** in the **Select from Previous Presentation** field, and enter `Treatment` in the **Name** field. Leave the **Switch to this presentation** check box selected, and click **Create**.
 - d. Drag and drop a Text Area control to the form canvas. Select the control, and change its **Name** and **Label** fields to `Treatment`.
5. Create a Surgery presentation.
 - a. Repeat the substeps from the previous step to clone a presentation, but this time name the presentation `Surgery` and base it on the `Start` presentation.

Notice **formArg** listed under Interface Argument. It refers to data objects automatically created for the form.

4. Click **Save**.

Set Presentations for the Human Task Activities

Next you'll implement each human task activity with a form presentation.

1. On the **Emergency Process** tab, select the Screen Patient activity and click **Edit Properties** .

The activity's properties pane opens at the side.
2. In the activity's properties pane, click **General** under implementation options.

The properties pane expands.
3. Complete the General implementation fields.
 - By default, the title has the same name as the human task activity. Optionally, enter meaningful text in the **Title** and **Task Summary** fields. Entries you specify appear at the top of the form in runtime. You can enter literal values or expressions. See [Create Simple Expressions](#).
 - In the **Form** field, select `PatientForm`. In the **Presentation** field, select **Screening**.
4. Click **Close**.

The screenshot shows the configuration for a 'Screen Patient' activity. The 'General' tab is selected in the 'Screen Patient Properties' dialog. The 'Title' is 'Screen the patient', 'Form' is 'PatientForm', and 'Presentation' is 'Screening'. The 'Close' button is highlighted with a red box.

5. Select the Treat Patient activity and implement the form and Treatment presentation.
Click **General**, select **PatientForm** again in the **Form** field, but select **Treatment** this time in the **Presentation** field. Click **Close**.
6. Select the Discharge Patient activity and implement the form and Discharge presentation, then click **Close**.
7. Select the Surgery activity and implement the form and Surgery presentation, then click **Close**.

Configure Data Association for the Human Task Activities

Each human task activity needs data input and output defined through data association. Data associations define the information passed between flow elements.

1. Click **Data Objects**. In the Data Objects section, expand **Input**, then **formArg**.
Notice the data objects for the form controls you added. These data objects were automatically added as you created the web form.
2. Select the Screen Patient activity, click **Menu** , then **Data Association**, and then **Input**.
The Data Association editor opens with the **Input** tab selected.
3. In the Data Association editor, click **Auto Mapping** .
Notice that formArg and PatientForm mapping automatically appears. This mapping indicates that values from the form (formArg) get passed to the PatientForm objects.
4. Click the **Output** tab, and then click **Auto Mapping** .

Notice that the same PatientForm and formArg mapping appears, but switched. This mapping allows the data values to change if the user enters or changes values in the human task's web form before submitting.

5. Click **Apply**.
6. Select the Treat Patient activity and repeat the steps to auto map the human task's data association. Be sure to auto map on both **Input** and **Output** tabs, and click **Apply**.
7. Select the Discharge Patient activity and repeat the steps to auto map the human task's data association.
8. Select the Surgery activity and repeat the steps to auto map the human task's data association.

Auto mapping is making a best guess on data values to map, based on names and data types, and works well for this simple example. Keep in mind that you can map and even transform values, including arrays, during data association in complex ways. You can also map and use the outcome of an approval task to drive or affect the process. See [Configure Data Association](#).

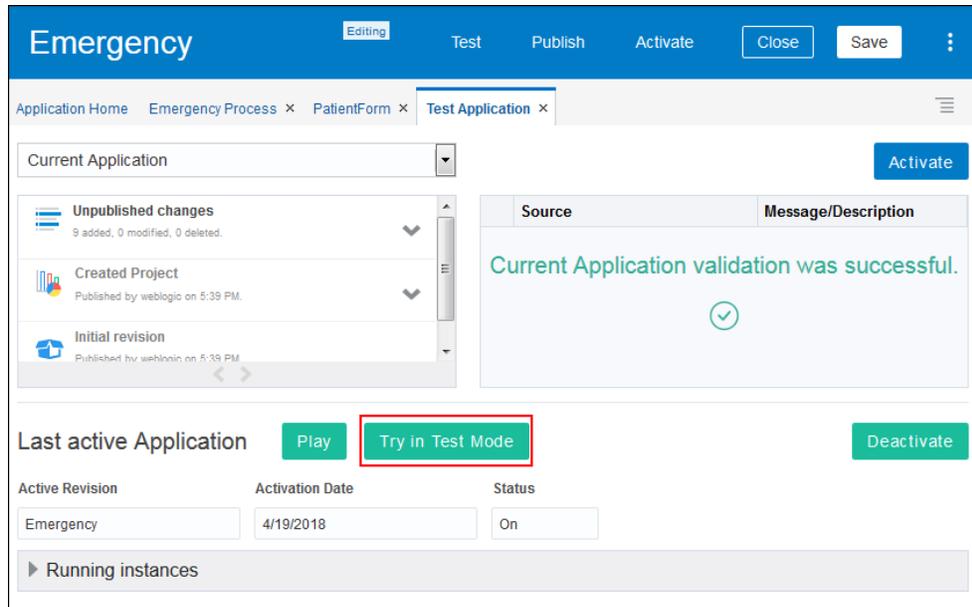
Test Activate the Application

Test activating validates and deploys the application, and then makes it available in runtime test mode. You don't activate the application to the production environment until it's ready.

Before test activating the application, check one last time for any validation issues in your process. Use the configuration panel to display the validation issues. If you see any issues, click the validation icon to view details about the issue. Click **Fix** to open the properties panel related to the issue and easily fix the issue.

With your basic dynamic process configured, it's time now to validate, activate, and make the application available in runtime test mode.

1. Click **Test**.
A **Test Application** tab opens. The upper portion indicates application validation results.
 - If the validation was successful, proceed to the next step.
 - If one or more validation issues are listed, click the **Emergency Process** tab to return to the dynamic process to correct the issues before clicking **Test** again.
2. On the **Test Application** tab, click **Activate**.
3. In the Activate to Test dialog box, leave **Add Me to All Roles** selected, and click **Activate**. A message displays that the application activated successfully.
4. Click **Try in Test Mode**.



A new test mode browser tab opens and the **My Tasks** navigation pane now displays runtime options. You can tell you're in test mode: there's an indicator set to **On** at the top of the screen.

Try Out the Dynamic Process Application in Runtime

The last step in creating a basic dynamic process is to try out its activated process application by starting and running the dynamic process in runtime.

Now that you have created a simple dynamic process, and activated its process application, let's try the process as a knowledge worker would.

1. If needed, click **My Apps** from the runtime options in the navigation pane.

The My Apps page displays automatically after you test activate your application. See [Test Activate the Application](#).

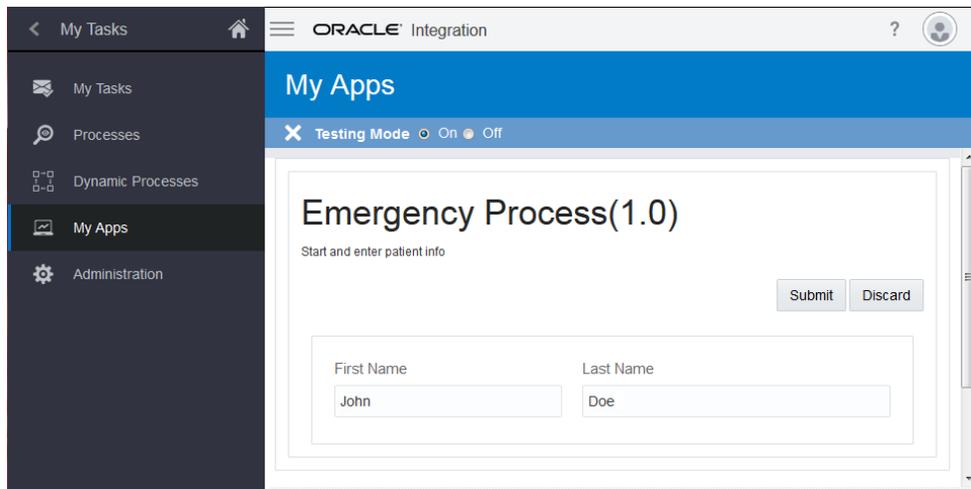
Your activated application appears, showing its icon (with EP for emergency process), along with its revision number and the dynamic process name and start form text you entered during design time.



2. Click the dynamic process application.

The start form you specified for the dynamic process (Start presentation) opens.

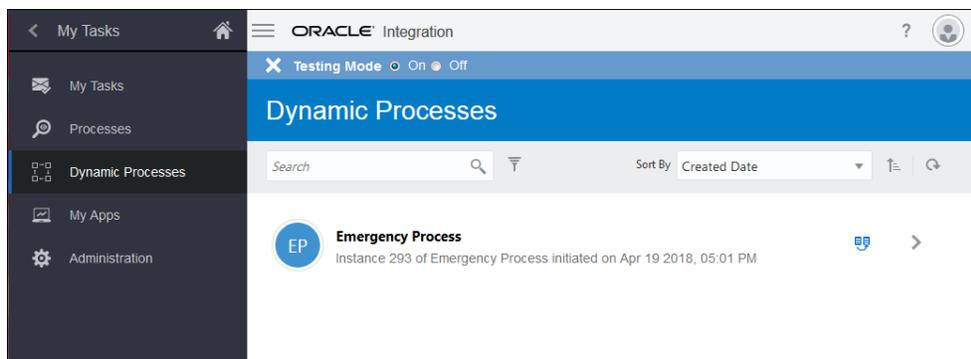
3. Complete the first and last name fields with sample values and click **Submit**.



A message briefly appears to let you know that an instance was created.

4. In the **My Tasks** navigation pane, click **Dynamic Processes**.

The instance you just created is listed.

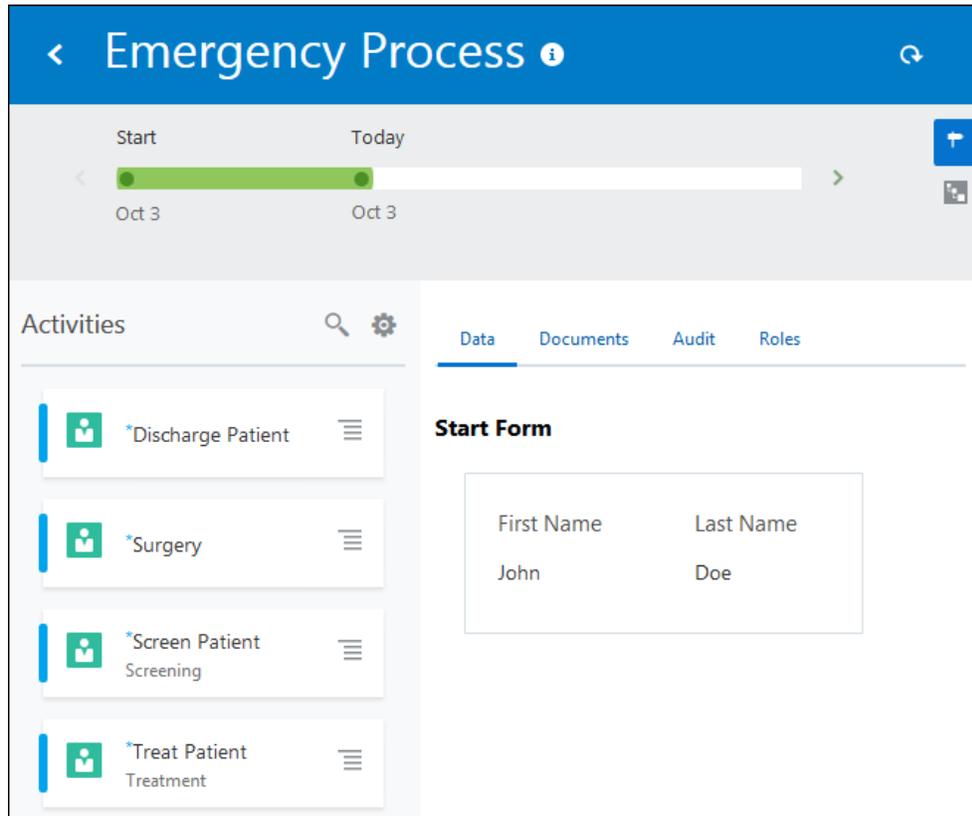


5. Click  to open your Emergency Process instance.

The dynamic process runtime page opens. Take a moment to explore.

- The activities you created are available under **Activities**, and show the same green human task icon as in design time. The surgery and discharge activities that aren't part of a stage are listed first and have no stage listed. An asterisk indicates required activities.
- On the adjacent Details pane, the **Data** tab is selected. The start form includes the values you entered when you started the dynamic process.
- The top bar lets you switch the instance progress view between milestones and stages. You can see that the instance started today. To view the two stages, click the **Stage** icon. Click a stage to see its number of available, active, and completed activities.
- Notice the other tabs. The **Documents** tab displays documents related to the process instance. Oracle Integration must be integrated with Oracle Content Management to work with documents. Your assigned role's document permissions control the actions you can perform with documents.
- The **Audit** tab lists what's happened to the process instance so far. Notice that all plan items (stages and activities) started.

- The **Roles** tab lists roles defined for the process instance, and provides an option to override the assigned role.



6. Select, complete, and submit the Screen Patient activity.
 - a. Under **Activities**, select the activity to work on it (click its title or select **Open**) from its actions. The **Activity Details** tab opens and the screening presentation appears.
 - b. Enter a sample value in the **Symptoms** field and click **Submit**. Notice that the Screening stage at the top turned green, indicating its completion.
7. Select, complete, and submit the Treat Patient activity.

Notice the **Force Complete** option that knowledge workers can access by clicking **Menu** . Selecting this option acts as a cancel for the activity. It withdraws the activity from the Activities list, but doesn't complete the human task or process. A doctor or nurse might force complete an activity that turns out not to be needed.

Also notice that the text you entered in the **Symptom** field doesn't appear in the Treatment activity. That's because all of the activities started at the same time with the same form values (payload) available.

8. Select, complete, and submit the Surgery and Discharge Patient activities.

After you complete the last activity, a message displays that no activities are available to act on. Because all required activities of the process are complete, the process is completed, and a **Close** option appears at the top.
9. Click **Close** and confirm to close the dynamic process and return to the dynamic processes list.

You can't reopen a closed instance. However, you can reopen completed instances to access any remaining open (non-required) activities.

10. Search for the dynamic process you just closed.
 - a. In the **Search** field, enter the first or last name you entered when starting the application.
 - b. Click **Filter** . In the Filter dialog box, select **Closed** in the **State** field and click **OK**. The instance you just closed is listed, but now shows a Closed icon rather than an Active one. You can open it and view its details on the **Audit** tab.
 - c. On the Dynamic Processes page, click **Clear** to clear the search filter.

The searching and filtering options enable knowledge workers to locate and return to instances at any time by entering identifiable information such as name or ID.

Take your dynamic process to the next level

Now that you've got the basics down, let's make some improvements to the dynamic process using milestone and structured process activities, adding conditions and events, and working with plan item lifecycles.

The following topics build on a basic dynamic process application. See [Learn dynamic processing basics](#).

- [Set a Stage's Activation](#)
- [Add a Milestone](#)
- [Control Plan Item Behavior Using Markers](#)
- [Define Roles and Their Permissions](#)
- [Set Assignees for Human Task Activities](#)
- [Set the Process to Complete or Close](#)
- [Explore Embedding Process Runtime Components](#)
- [Explore Advanced Dynamic Process Topics](#)

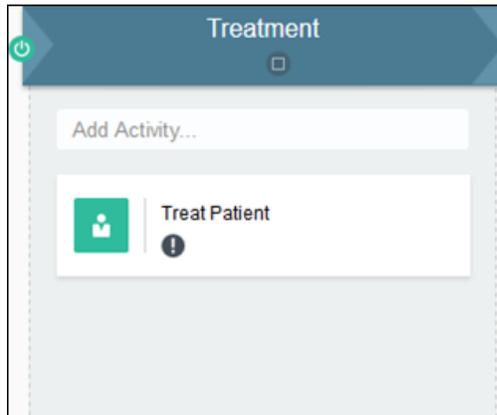
Set a Stage's Activation

By default, stages become available at the same time.

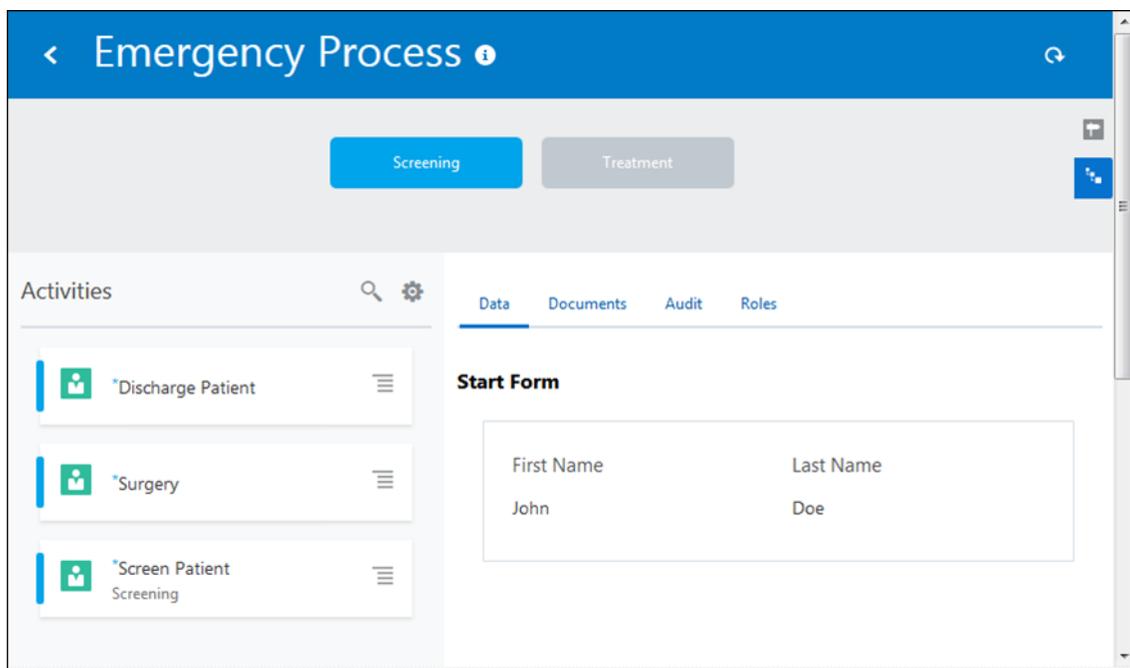
Let's change it so that the Treatment stage isn't available until the Screening stage is complete.

1. Close the test runtime tab, and return to the **Emergency Process** tab in design time.
2. Select the Treatment stage (click where it says "Treatment") and open its properties for editing.
3. Select the **Conditions** tab. Click **Create Condition**  in the **Activation** table.
4. In the expanded Activation condition pane, complete settings.
 - a. Enter `Treatment stage starts when` in the **Label** field.
 - b. Click **Create Event**  in the **Events** table. Leave the default settings of **Previous Stage** and **Complete** selected, and click **Create**.

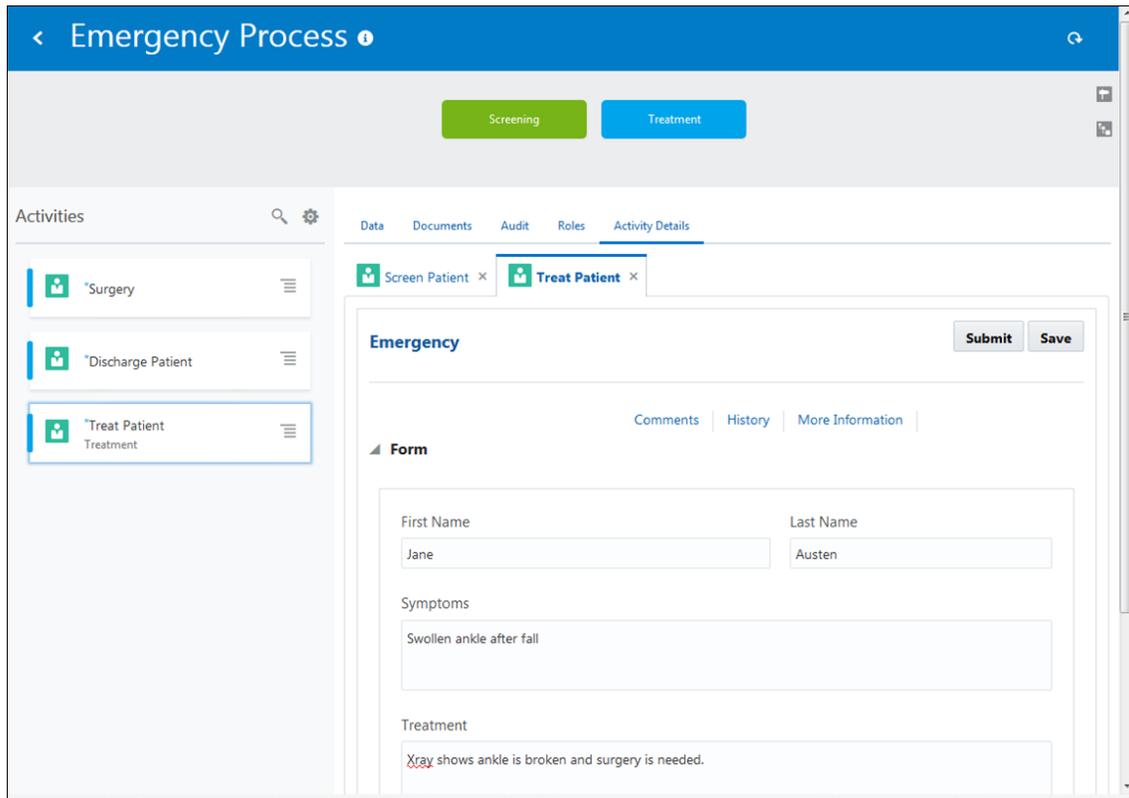
Notice that the Treatment stage now displays an activation symbol indicating an activation condition.



5. Test activate again.
Click **Test**, then **Activate**. Leave **Add Me to All Roles** selected, and click **Activate**. You're test activating a new version, overriding the version you previously activated. After successful activation, click **Try in Test Mode**. See [Test Activate the Application](#).
6. Try out your changes in runtime.
 - a. From **My Apps**, start a new application, enter sample values, and submit. Select **Dynamic Process** in the navigation pane and open the new dynamic process instance.
 - b. Notice that the Treat Patient activity no longer displays, because it doesn't start until the Screening stage is done. Click **Stage**  and notice that the Treatment stage is gray, indicating that it's inactive.



- c. Complete the Screen Patient activity, adding symptom text, and click **Submit**. Notice how the Treatment stage and activity become available.
- d. Complete the Treat Patient activity. Notice how the symptom text was retained. This is because the human task wasn't created until the activity became available (Screening stage was complete), so the activity's data, or payload, included the symptom text you submitted.

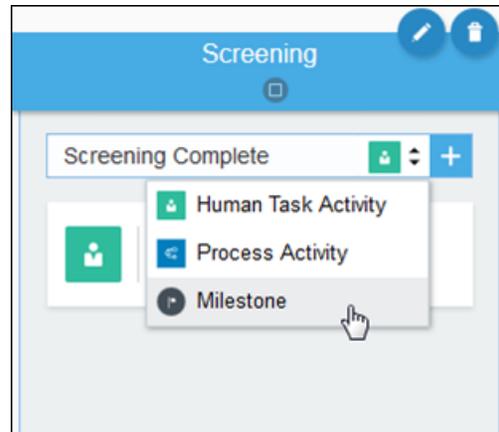


Each plan item has properties, and can have activation, termination, or completion conditions that control its state—for example, when it is active, available, or terminated. Want to learn more about plan item activation? See [About Process and Plan Item Lifecycles](#).

Add a Milestone

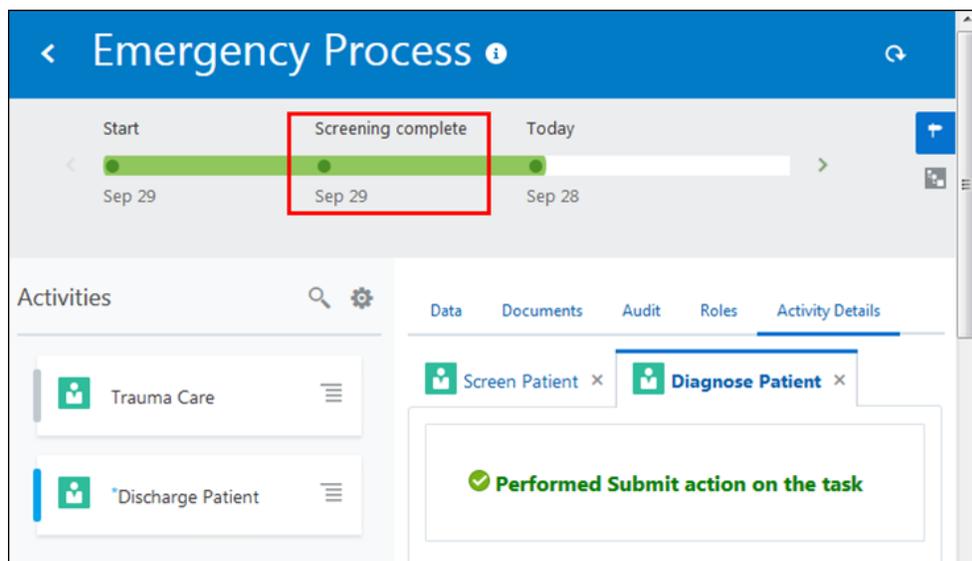
Milestones indicate that something occurred or a condition was met. Use milestones so users get updated on what's transpired in a running process instance.

1. Close the test runtime tab, and return to the **Emergency Process** tab in design time.
2. Add a milestone in the Screening stage.
 - a. In the **Add Activity** field, enter Screening Complete.
 - b. Click the activity icon and choose **Milestone**, and click **Add Activity** .



3. Edit the milestone's properties.
 - a. Click the **Conditions** tab. Click **Create Condition**  to create a Completion condition called *Screening complete* when.
 - b. Click **Create Event**  to add an event and select **Screen Patient** from **Activities** and **Complete**. Click **Create**. (Note that you can add multiple events to a condition and specify an AND or OR condition between them.)
Notice that the milestone now displays an activation symbol indicating an activation condition.
4. Test activate and try out your changes in runtime.

With the **Milestone** view displayed, submit the *Screen Patient* activity, and watch the milestone progress bar show the completed milestone.



Control Plan Item Behavior Using Markers

Markers control the behavior of plan items (activities, stages, and processes), such as whether they're required, repeatable, manually activated, and if they auto complete. Depending on the plan item, different markers are available.

By default when you create an activity, it becomes required (the Required marker is selected by default for the activity). You can deselect Required and make changes to the behavior of the activity, but you have to configure one of the following options. Otherwise, you'll get validation errors.

- The activity has one activation condition based on events.
- The activity uses a condition in its marker.
- The activity has a milestone added to it.

Right now, all activities in the Emergency Room example are required. Let's make some changes.

1. Make the Screen Patient activity not required.

On the **Emergency Process** tab, select the Screen Patient activity and edit its properties. Under **Markers**, deselect **Required**. Notice that the activity no longer displays a **Required**  icon.

Note that the Screen Patient activity has a milestone (Screening Complete) added to it.

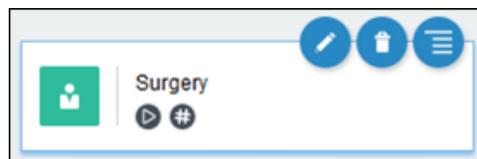
2. Make the Surgery human task activity repeatable, manually activated, and not required.

Edit its properties, select **Repeatable**, deselect **Required**, and select **Manually Activated**.

3. Set conditions to the **Repeatable** and **Manually Activated** markers.

- a. Click **Edit** in the **Repeatable** marker. The Property pane opens.
- b. In the Property pane, click **Create Data Condition** on the Data Driven section.
- c. In the Create Data Condition dialog box, select **Simple** as the condition type and enter `Surgery repeatable condition` in the **Name** field.
- d. Define condition as `formArg` is equal to `formArg` and click **Create**.
- e. In the Property pane, click **Save**.
- f. Repeat the steps above for the **Manually Activated** marker, only this time enter the condition name as `Surgery manually activated condition`.

Notice that the Surgery activity's marker icons now reflect the new marker choices.

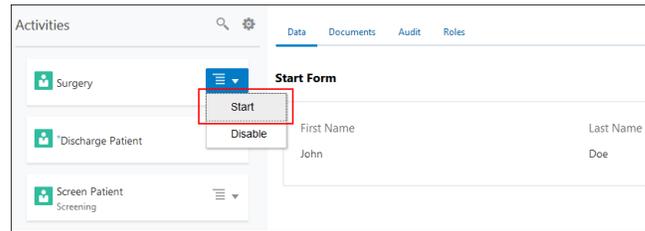


4. Test activate, try in test mode, and view your changes in runtime.

You'll notice that you cannot open the Surgery activity by selecting it in the Activities section of your dynamic process. This is because it is set to manually activated.

5. Start and complete the Surgery activity.

- a. Select its **Actions** menu, and choose **Start**. Click **OK** to confirm.
Note that you can now select and open the Surgery activity.



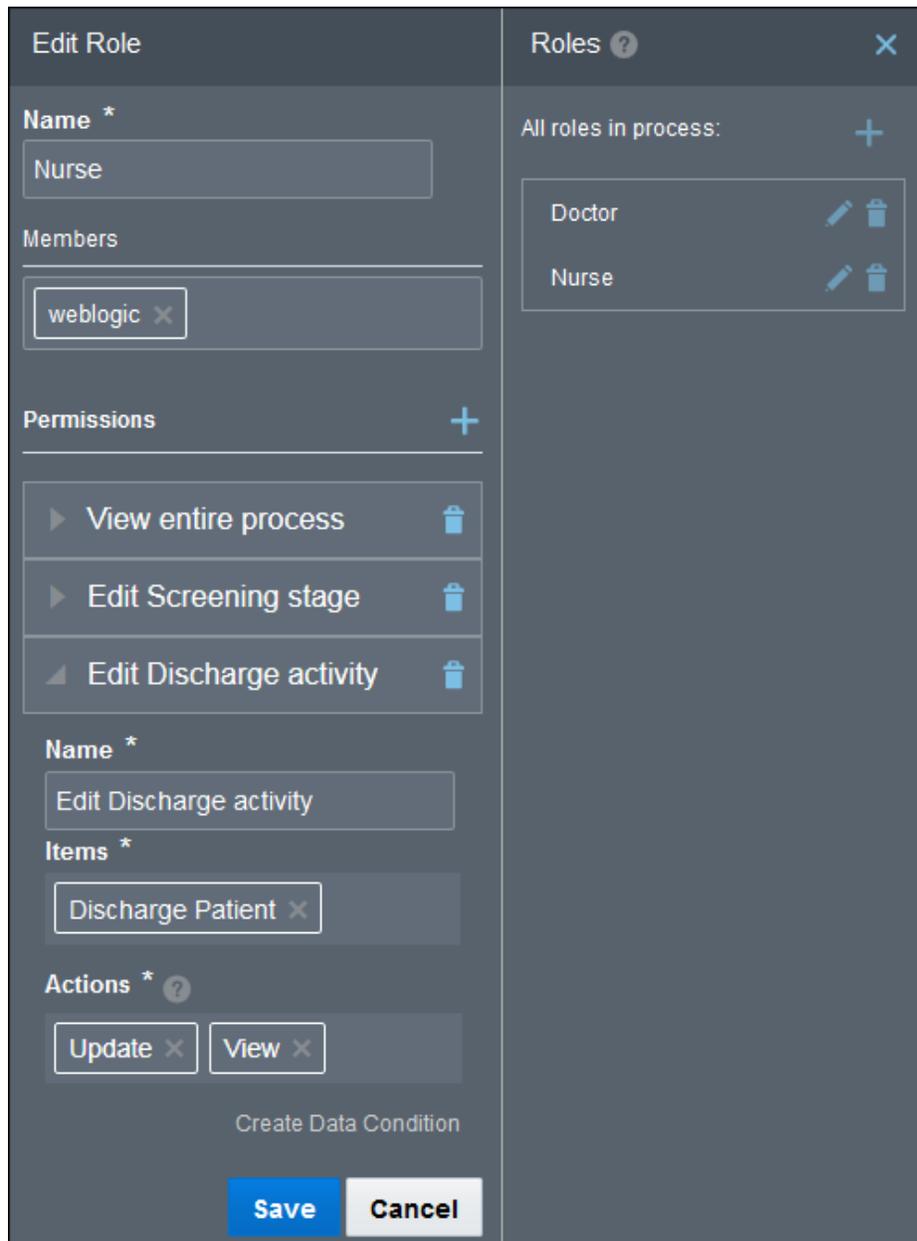
- b. Select, complete, and submit the Surgery activity.
After you submit, notice that Surgery activity displays again in the Activities list because it is set to repeatable.
- c. After completing all required activities, a **Close** option appears at the top of the page for completing the process. Click **Close**.

Define Roles and Their Permissions

Roles in dynamic processes provide maximum flexibility. You create roles and define the process elements they apply to and the permissions granted.

1. Click the **Roles** button.
The **Roles** tab lists roles defined for the dynamic process. Two default roles, Process Owner and Process Viewer, are provided. Let's adapt these default roles for the dynamic process.
2. Click **Edit** for the Process Owner role. Change the **Name** field to `Doctor`.
3. Click the **Members** field and select the user you used to sign in to Oracle Integration.
4. Edit the existing permission for the doctor role.
Notice that the existing permission allows the doctor role to perform all actions in the process. Change the permission's **Name** field to `Perform all actions in process`. Click the triangle to collapse the permission, and click **Save**.
5. Edit the Process Viewer permission for a nurse role.
 - a. Click **Edit** for the Process Viewer role and change the **Name** field to `Nurse`.
 - b. Click the **Members** field and select your sign-in user name. (Normally, the nurse and doctor roles would be assigned to different users.)
 - c. Notice that the existing permission allows the nurse role to view all actions in the process. Change the permission's **Name** field to `View entire process` and collapse it.
6. Add a permission that lets the nurse edit the Screening stage. Click **Add**  in the Permissions table to create a permission, expand it, and complete its settings.
 - a. In the **Name** field, enter `Edit Screening stage`.
 - b. In the **Items** field, select **Screening** from Stages.
 - c. In the **Actions** field, select **Update** and **View**.
 - d. Collapse the new permission.

7. Add a permission that lets the nurse edit the Discharge activity. Click **Add**  to create a permission, expand it, and complete its settings.
 - a. In the **Name** field, enter `Edit Discharge activity`.
 - b. In the **Items** field, select **Discharge Patient** from Activities.
 - c. In the **Actions** field, select **Update** and **View**.



The screenshot shows the 'Edit Role' dialog box. The left pane is titled 'Edit Role' and contains the following fields and sections:

- Name ***: Nurse
- Members**: weblogic
- Permissions**: View entire process, Edit Screening stage, Edit Discharge activity
- Name ***: Edit Discharge activity
- Items ***: Discharge Patient
- Actions * ?**: Update, View
- Create Data Condition**: (link)
- Save** and **Cancel** buttons

The right pane is titled 'Roles' and contains the following sections:

- Roles ?** (with a close icon)
- All roles in process:** (with a plus icon)
- Doctor** (with edit and delete icons)
- Nurse** (with edit and delete icons)

8. Click **Save**.

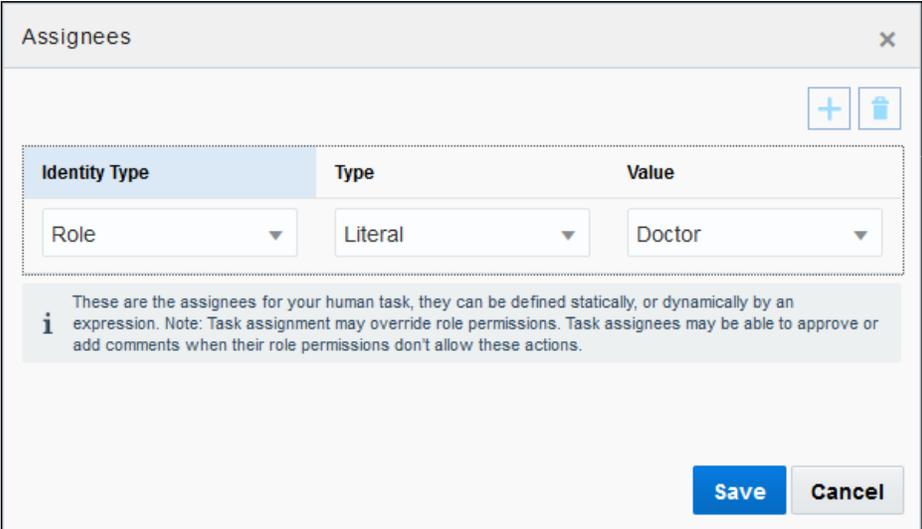
You can temporarily override roles at runtime. But setting roles and permissions in design time ensures they're retained for all deployments.

Want to learn more about roles in dynamic processes? See [Create Process Roles](#) and [Override Roles Assigned to a Process](#).

Set Assignees for Human Task Activities

Each human task needs one or more assignees. Let's assign the roles you created to the human task activities.

1. Assign the Doctor role to the Treat Patient activity.
 - a. Select the activity, click **Edit Properties**, and choose **General** under **Implementation** options.
 - b. In the **Assignees** field, click **Edit**.
 - c. In the Assignees window, click **Add** . In the **Identity Type** field, select **Role**. In the **Value** field, select **Doctor**.
 - d. Click **Save**, then **Close**.



| Identity Type | Type | Value |
|---------------|---------|--------|
| Role | Literal | Doctor |

i These are the assignees for your human task, they can be defined statically, or dynamically by an expression. Note: Task assignment may override role permissions. Task assignees may be able to approve or add comments when their role permissions don't allow these actions.

Save **Cancel**

2. Repeat the step above to assign the nurse role to the Screen Patient and Discharge activities, and the doctor role to the Surgery activity.

Set the Process to Complete or Close

Dynamic processes, like stages and activities, follow a lifecycle where they move through states such as active, completed, and closed. (Note that termination ends the plan item.)

When you start a process instance, the dynamic process is active. All stages and activities are started and available, unless a condition affects their activation. Whether you set the process to auto complete determines whether processes close or complete at the end of their lifecycle. Let's experiment to see the differences.

- By default, dynamic processes do not auto complete, which means that the **Complete** option appears when all stages and required activities contained within them are completed, terminated, or disabled.
- Setting a process to auto complete means that once all required activities are complete, the **Close** option appears and all non-required activities disappears and "No activities available to act on" appears. You can return to the instance, but can't work on any remaining non-required activities.

1. In runtime, create and run a dynamic process instance. On the Dynamic Processes list, notice the **Active**  icon. Open and complete all required activities. (Note that a required activity might not be visible: If you complete the discharge activity but not the screen activity, no **Complete** option appears even with no required activities shown, because the required but not yet started treatment activity in the Treatment stage hasn't completed.)
Notice that after completing all required activities, a **Complete** option appears at the top of the page for completing the process.
2. Instead of clicking **Complete**, click **Exit**  to return to the dynamic processes list.
Notice that the instance is still active, because there are non-required activities.
3. Open the instance again and click **Complete**.
The instance is no longer in the list. If you search for it using the Completed filter, it displays . You can even open it and select **Close**, which changes its icon to .
Now let's see the difference when the process is set to auto complete.
4. Switch to design time and open the process properties at the top of the page by clicking **Edit** .
5. In the process properties pane, select **Auto Complete** under **Markers**.
6. Change the title of the process that displays for instances on the Dynamic Processes page and at the top of instances.
Optionally, enter a new title in the **Title** field to appear for all runtime instances, such as "Emergency Room Process". (Leave the quotation marks.)
7. Test activate and view your changes in runtime.
 - a. Notice that the new title you entered appears in the Dynamic Processes instance list.
 - b. Complete all required activities. Notice that the top link now shows **Close** instead of **Complete**, and the remaining non-required activity disappeared and was replaced with "No activities available to act on."

Want to learn more about process lifecycle? See [Process State Model](#) and [Define Process Completion and Termination](#).

Explore Embedding Process Runtime Components

What if you want users to start and run dynamic processes from another application or service, or display only certain portions of the runtime experience, such as activities and details? Use Embeddable Process UI Components (also called snippets) to embed selected runtime components.

To explore dynamic process runtime components using the cookbook:

1. Copy and paste the design time URL into a new browser tab.
2. Change the end of the URL, starting with `/ic/`, to `/ic/pub/components`.

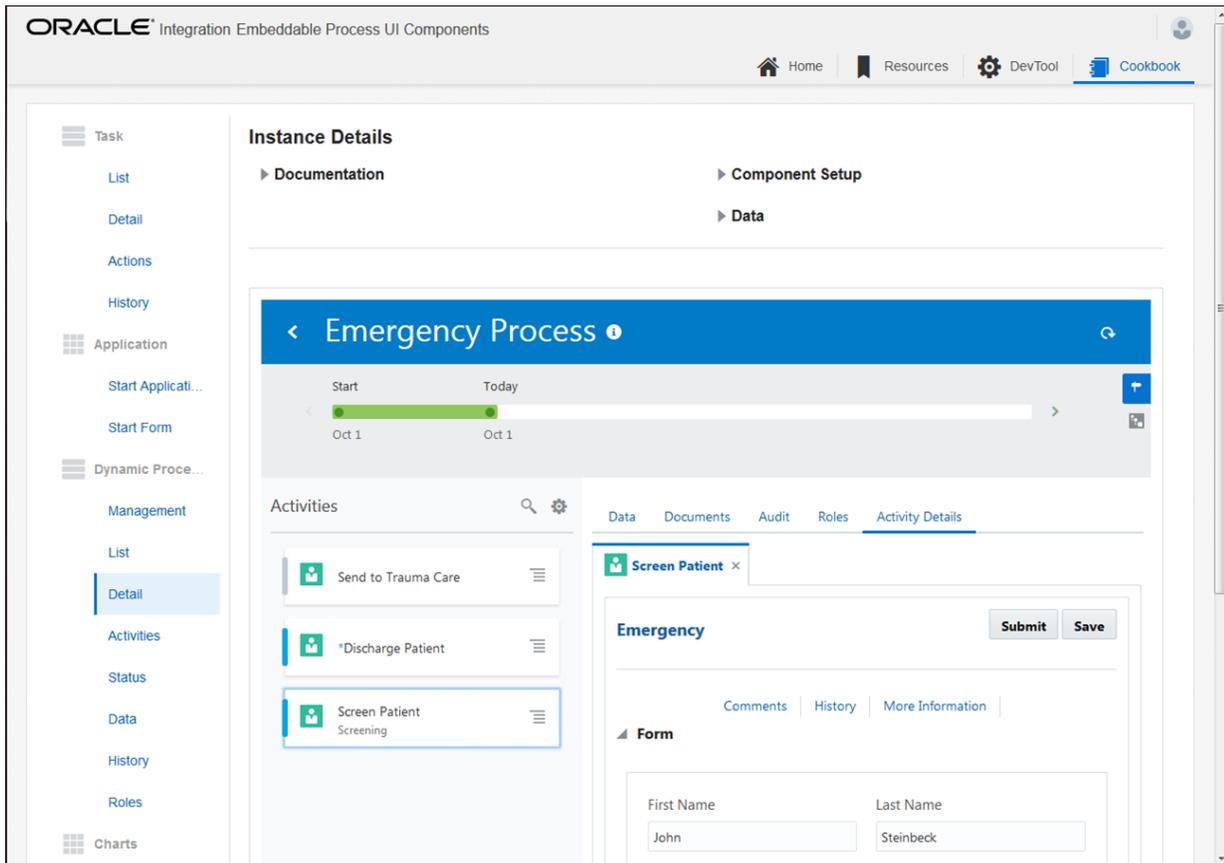
Use this URL format: `https://host:port/ic/pub/components`

The Oracle Integration Embeddable Process UI Components page opens.

3. Connect to your server.

Under **Server Setup**, complete the **username** and **password** fields with the same credentials you use to sign in to Oracle Integration, and click **Apply**. A message displays that the connection was successful.

4. Explore dynamic processes components in the cookbook.
 - a. Select the **Cookbook** tab.
 - b. Select **My Apps** in the side pane and create a dynamic process instance.
 - c. Select components listed under **Dynamic Process**. For example, select **Detail** under Dynamic Process as if in runtime.



Want to learn more about the cookbook and embeddable components? See [Embed Process UI Components in Other Applications](#).

Explore Advanced Dynamic Process Topics

Here are some other areas to explore in the dynamic process editor.

Topics:

- [About Process and Plan Item Lifecycles](#)
- [Create Simple Expressions](#)
- [Create Process Roles](#)
- [Work with Dynamic Processes in Runtime](#)

Mix and match processes

What if your process application needs to address both unpredictable and structured paths? Design processes that fit your needs, then call them from each other: a structured process from a dynamic process, or a dynamic process from a structured process.

The following topics build on a basic dynamic process application taken to the next level. See [Learn dynamic processing basics](#) and [Take your dynamic process to the next level](#).

- [Use a Structured Process in a Dynamic Process](#)
- [Use a Dynamic Process in a Structured Process](#)

Use a Structured Process in a Dynamic Process

What if parts of your business process follow a predictable and sequential path? Model them as structured processes that your dynamic process calls. Structured processes also provide some functionality not currently available in dynamic processes, such as decision models, integrations, and service calls.

In the emergency room example, suppose the surgery process involves a sequential process flow. Let's create and call a structured process.

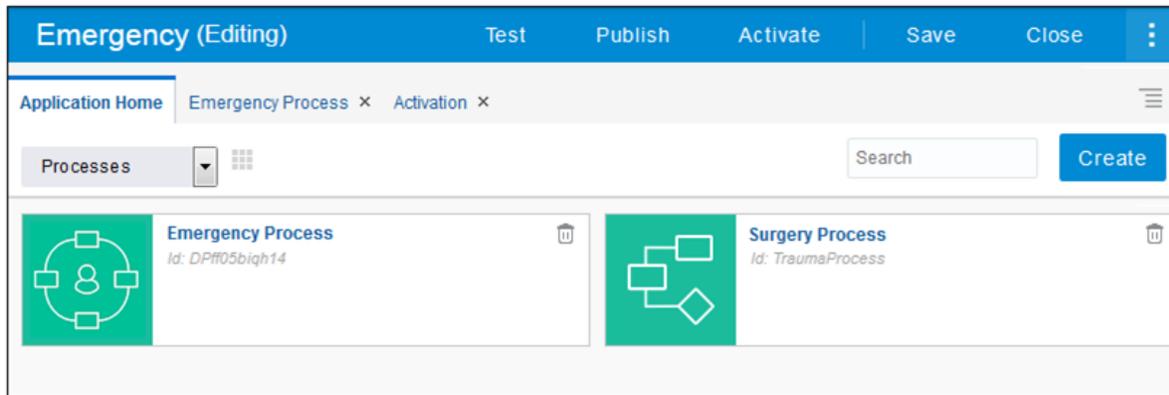
- [Create a Structured Process](#)
- [Add Human Tasks](#)
- [Edit the Process' Swimlane](#)
- [Associate a Web Form with the Approve Human Task](#)
- [Define the Process Start and End Events](#)
- [Configure Data Associations for Process Start and End Events](#)
- [Call the Structured Process in the Dynamic Process](#)
- [Test Activate to Validate the Application](#)
- [Try the Structured Process in Runtime](#)

Create a Structured Process

Let's create a structured process.

1. In design time, click the **Application Home** tab.
2. Click **Create a Structured Process** and then click **Start when a message is received**. Optionally, click **Create**, and then click **New Process**.
3. In the Create Process dialog box, enter `Surgery Process` in the **Name** field. Deselect **Create Immediately** and click **Create**.

Notice how the two processes on the **Application Home** tab have different icons to indicate dynamic versus structured processes.



Note that structured processes must be asynchronous processes using the Message Start pattern.

Add Human Tasks

Let's add human tasks to the structured process we just created.

1. Open the Surgery Process.

2. Insert an Approve human task.

Drag and drop an Approve human task from the Elements Palette onto the process flow. Position it where you want to add it.

Double-click the task's name and enter `Patient Consent`.

3. Insert a Submit human task.

Drag and drop a Submit human task from the Elements palette onto the process flow. Position it where you want to add it.

Double click the task's name and enter `Surgery Tasks`.

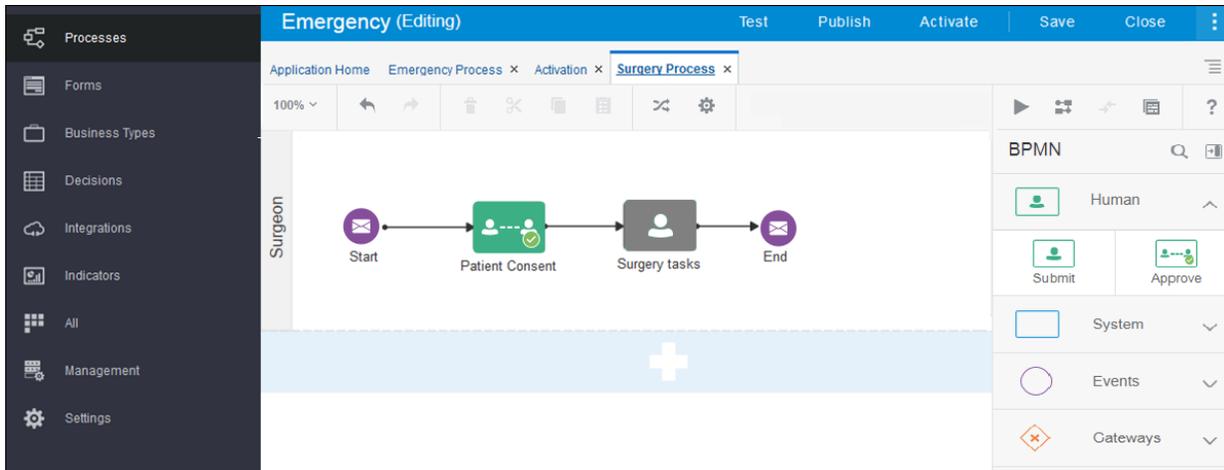
4. Modify properties for the Submit human task.

- a. Select Surgery tasks, click **Menu**  and then select **Open Properties**.

- b. In the Properties pane, select **Is Draft**.

Notice that Surgery Tasks turns gray. This means that it is just a placeholder for the surgery tasks to be configured later.

- c. Click **Collapse Pane**  to collapse the Properties pane.



Edit the Process' Swimlane

Edit the process' swimlane to assign a role responsible for performing each tasks within the process. In the Surgery Process, let us change the process' swimlane to a surgeon role.

1. Select **Process Owner** and click **Edit**.
2. In the implementation pane, click **Add Role**  to add a role and enter *Surgeon*.
3. Click **Collapse Pane**  to collapse the implementation pane.

Associate a Web Form with the Approve Human Task

Let's associate a web form with the Patient Consent human task.

1. Select the Patient Consent human task, click **Menu**  and choose **Open Properties**.
2. Click the browse icon next to the **Form** field to search for PatientForm.
3. Select **PatientForm** and click **OK**.
4. In the **Presentation** field, select **Surgery**. Collapse the implementation pane to save your changes.

Define the Process Start and End Events

Implement the process start and end events to input and output the name fields.

1. Select the Start Message event.
2. Choose **Open Properties**, then click **Edit** for Define Interface.
3. In the Arguments Definition table of the Configure dialog box, click **Add** and enter *FirstName* in the **Name** field. Click **Add** and enter *LastName* as a string type field, and click **OK**
4. Select the End Message event and repeat the steps from 1–3.

| Name | Type | |
|-----------|--------|---|
| FirstName | string | X |
| LastName | string | X |

Configure Data Associations for Process Start and End Events

Define data associations for the Surgery Process' start and end events to define what payload passes onto the process start event and what data passes out of the process end event.

1. Select the Start Message event. Click **Menu**  and choose **Open Data Association** to open the Data Association editor.
2. Map `FirstName` to `patientFormDataObject.firstName`, and `LastName` to `patientFormDataObject.lastName`. Click **Apply**.
3. Select the End Message event, open the Data Association editor, and repeat step 2.

Call the Structured Process in the Dynamic Process

Let's configure the dynamic process to call the structured process.

1. Return to the dynamic process by selecting the **Emergency Process** tab.
2. Select the Surgery activity and click **More**, then **Change Type**, and then **Process Activity**.
Notice that its activity icon now shows a process rather than a human task.
3. Edit its properties (Implementation) and select **SurgeryProcess** in the **Process** field and **Start** in the **Start Event** field.
Leave **Non-blocking** deselected, which means that the dynamic process will wait for the structured process instance to complete to continue.
4. Deselect the **Repeatable** and **Manually Activated** markers.

Note that you can apply these markers to structured processes when they make sense.

5. Set data association now that the activity has changed type.
Use auto mapping for input and output.
6. Click **Apply**.

Test Activate to Validate the Application

Test activate the Emergency application to validate and make it available in runtime.

1. Click **Test**.
 - If the validation was successful, proceed to the next step.
 - If any validation issues are listed, correct the issues and click **Test** again.
2. Click **Activate**.
3. In the Activate to Test dialog box, leave **Add Me to All Roles**, and click **Activate**.
A message displays that the application activated successfully.
4. Click **Try in Test Mode**.

A new test mode browser tab opens. The navigation tab now displays runtime options.

Try the Structured Process in Runtime

Now that you've created a structured process in a dynamic process, and test activated the application, let's try out the Surgery process in runtime to see how it works.

1. Click **My Apps**, then create and open a dynamic process instance. Notice that the Surgery activity shows a structured process icon in the Activities list.
2. Select **Inbox** in the navigation pane. Typically, the surgeon would be accessing an assigned task from a structured process like the Surgery process from the Inbox rather than from within the dynamic process, although knowledge workers can open structured process tasks from either the Inbox or Dynamic Process.
3. Open the Emergency task whose process is SurgeryProcess. Click **Approve** and the task is completed. Notice that this Approval human task displayed **Reject** and **Approve** buttons rather than a **Submit** button.
4. Select **Dynamic Processes** in the navigation pane and reopen the topmost instance. The Surgery task no longer appears, since you completed it through the Inbox. Click the **Audit** tab and you can see (and open) the just completed Surgery process activity.

Use a Dynamic Process in a Structured Process

If your business process is mostly predictable (structured) but part is unpredictable and non-sequential (dynamic), accommodate both by calling a dynamic process from your structured process.

Our previous hospital emergency room example demonstrated calling a structured process (Surgery) from a dynamic process (Emergency Room). Want to learn more? See [Use a Structured Process in a Dynamic Process](#).

But now let's imagine a different twist in our example: Surgery (a structured process) followed by post surgery (a dynamic process).

What's dynamic about the post surgery process?

- The patient may need no additional action, except to be discharged and go home.
- Minor complications could develop, that might require follow-up care or treatment.
- In rare cases, the patient might develop major complications and require extensive treatment.

Let's get started creating:

- A **structured process** for the overall surgery tasks. The doctor begins the surgery process, and the patient gives consent. (For the purposes of our example, we'll imagine that the surgery itself involves additional steps not covered here.)
- A **dynamic process** to model the unpredictable post-surgery tasks. For this example, we'll limit the activities to discharge and, minor and major complications.

To call a dynamic process from a structured process, we'll complete these major steps:

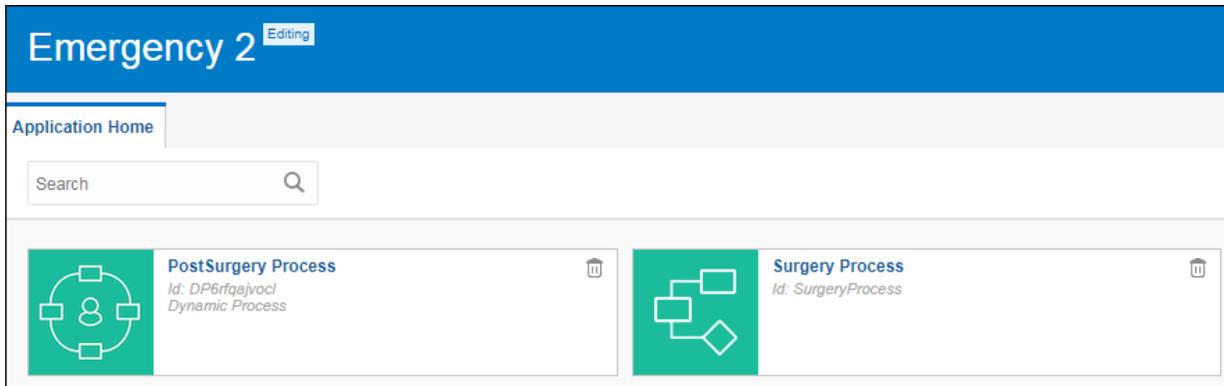
- [Create a Structured and a Dynamic Process](#)
- [Add Human Task Activities to the Structured Process](#)
- [Create a Form and Its Presentations](#)
- [Configure the Dynamic Process Input and Tasks](#)
- [Configure the Structured Process to Call the Dynamic Process](#)
- [Configure Data Association for the Dynamic Process Activities](#)
- [Try Out the Process Application in Test Mode](#)

Create a Structured and a Dynamic Process

We'll start a new simple process application (*Emergency2*) to house the structured and dynamic processes.

1. On the Process Applications page, click **Create**, then **New Application**.
2. In the Create Application dialog box, enter `Emergency2` in the **Name** field, and click **Create**.
The **Application Home** tab opens with application components shown in the navigation pane.
3. Create a structured process.
 - a. Click **Create a Structured Process**, and then click **Start with a form**.
 - b. In the Create Process dialog box, enter `Surgery Process` in the **Name** field. Deselect **Open Immediately** and click **Create**.
4. Create a dynamic process.
 - a. Click **Create a Dynamic Process**, and then click **Start with a todo list**.
 - b. In the Create Process dialog box, enter `PostSurgery Process` in the **Name** field. Deselect **Open Immediately** and click **Create**.

Notice the two processes in the **Application Home** tab. The different icons indicate dynamic versus structured process.



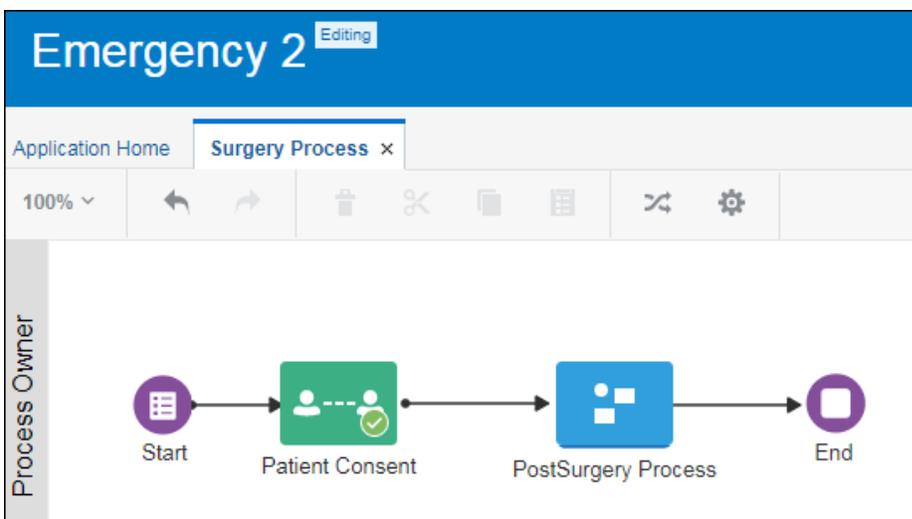
Add Human Task Activities to the Structured Process

Let's add a couple of human task activities that we know will be typically performed in a simple surgery process.

1. Click the Surgery Process to open it.
2. Add an Approve human task.
 - a. Expand **Human** in the Elements palette.
 - b. Drag and drop an approve human task from the palette onto the process editor canvas. Position it in your process where you want to add it.
 - c. Double-click the task's name and enter `Patient Consent`.
3. Add a Dynamic Process element.

This dynamic process element is presently empty, and is just a placeholder. But later, when we associate the PostSurgery dynamic process with this element, then it'll house all the activities of the PostSurgery dynamic process.

- a. Expand **System** in the Elements palette.
- b. Drag and drop a dynamic process element  from the palette onto the process.
- c. Double-click the name and enter `PostSurgery Process`.



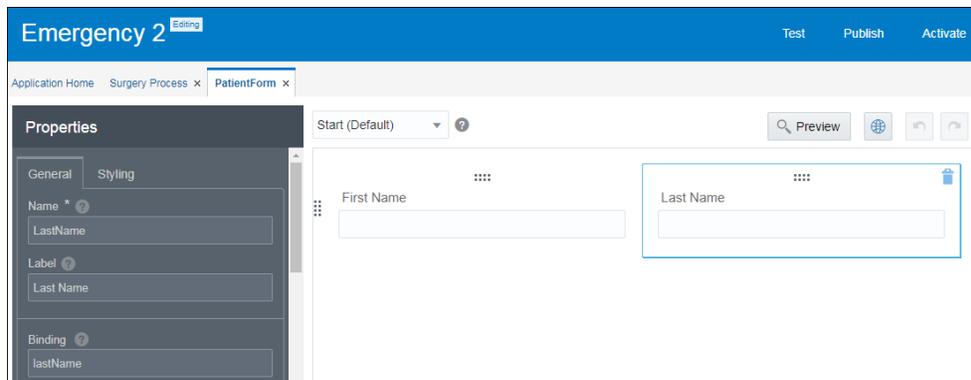
Create a Form and Its Presentations

Let's create a form with multiple presentations that you apply to the human task activities. We'll start by creating a form and a presentation for the structured process' start event. Then we'll create different presentations to customize the form for the various human task activities in our process application.

1. Click **Forms** in the navigation pane. On the Forms page, click **Create**.
2. In the Create New Web Form dialog box, enter `PatientForm` in the **Name** field, leave **Open Immediately** check box selected, and click **Create**.
3. Change the default presentation's name.
 - a. Click the **Presentation** tab on the Properties pane.
 - b. In the presentation table, change the name of the default Main presentation to `Start`.

You'll use this presentation for the structured process' start event.

4. Add input text controls for the patient's first and last name.
 - a. Drag and drop two Input Text controls from the Basic palette onto the form's central canvas side by side.
 - b. Select the first one, and change its **Name** field to `FirstName` and **Label** field to `First Name`.
 - c. Select the second control, and change its **Name** field to `LastName` and **Label** field to `Last Name`.



5. Create a presentation for the Approve human task.
 - a. Click outside a form control and select the **Form** tab in the Properties pane.
 - b. Click **Add**  in the Presentation table. In the Select Presentation Type dialog box, select **Clone**.
 - c. In the Create Presentation dialog box, select **Start** in the **Select from Previous Presentation** field, and enter `Consent` in the **Name** field. Leave **Switch to this presentation** check box selected, and click **Create**.
 - d. Drag and drop a Text Area control onto the form's canvas. Select the control, and change its name and label to `Comment`.
6. Create a Discharge presentation

Let's create presentations for the post-surgery tasks in the dynamic process. Start by creating the first presentation.

- a. Click **Add** in the Presentation table. In the Select Presentation Type dialog box, select **Clone**.
 - b. In the Create Presentation dialog box, select **Start** in the **Select from Previous Presentation** field, and enter `Discharge` in the **Name** field. Leave **Switch to this presentation** check box selected, and click **Create**.
 - c. Drag and drop a Text Area control onto the form's canvas, below the First Name and Last Name controls.
 - d. Enter `DischargeSummary` in its **Name** field and `Discharge Summary` in its **Label** field.
7. Create a Minor presentation.

Create the second presentation for the post surgery tasks.

- a. Click **Add** in the Presentation table. In the Select Presentation Type dialog box, select **Clone**.
 - b. In the Create Presentation dialog box, select **Start** in the **Select from Previous Presentation** field, and enter `Minor` in the **Name** field. Leave **Switch to this presentation** check box selected, and click **Create**.
 - c. Drag and drop a Text Area control onto the form's canvas. Enter `TestReports` in its **Name** field and `Test Reports` in its **Label** field .
 - d. Drag and drop a Radio Button control below the text area control. Enter `TreatmentRequired` in its **Name** field and `Treatment Required` in its **Label** field.
 - e. Change the option names of the radio button control to `Yes` and `No` in the **Option Names** field.
8. Create a Major presentation.

Create the third presentation for the post-surgery tasks.

- a. Click **Add** in the Presentation table. In the Select Presentation Type dialog box, select **Clone**.
- b. In the Create Presentation dialog box, enter `Major` in the **Name** field, and select **Start** in the **Select from Previous Presentation** field. Leave **Switch to this presentation** check box selected, and click **Create**.
- c. Drag and drop a Text Area control onto the form's canvas.
- d. Enter `Treatment` in its **Name** field, and `PostSurgery Treatment` in its **Label** field.

Configure the Dynamic Process Input and Tasks

Define the dynamic process' input to control what type of data starts the dynamic process. Also, implement each human task activity with a form presentation.

In our example, we have to set the input arguments such that the patient's first and last name can be the input data for the PostSurgery dynamic process.

1. In the navigation pane, click **Processes**.

You can see the Surgery Process and the PostSurgery Process that you created in the **Application Home** tab.

2. Click the PostSurgery Process to open it.
3. Select **Process Input** next to the process name at the top of the editor.
4. Configure settings in the Start the Dynamic Process dialog box.

Leave **With Data Only** selected, define the interface arguments and click **Define**.

Note that you cannot start a dynamic process used in a structured process with a form. Always start it with data.

Start the Dynamic Process

With Data Only With Form Only

Interface Arguments

i In this release, you can define process output, but can't use it yet in data associations. Defined output will work in future releases.

| | |
|-----------|--------|
| firstname | String |
| lastname | String |

+ Argument

Define Cancel

5. Select a form and presentation for each human task activity in the To-Do List.

Click **Edit Properties**  to open the related properties pane for the activity.

- Task 1: In the properties pane, enter `Patient Discharge` in the **Name** field. Click **General** under the implementation pane. In the general pane, select `PatientForm` in the **Form** field and select `Discharge` in the **Presentation** field. Click **Close**.

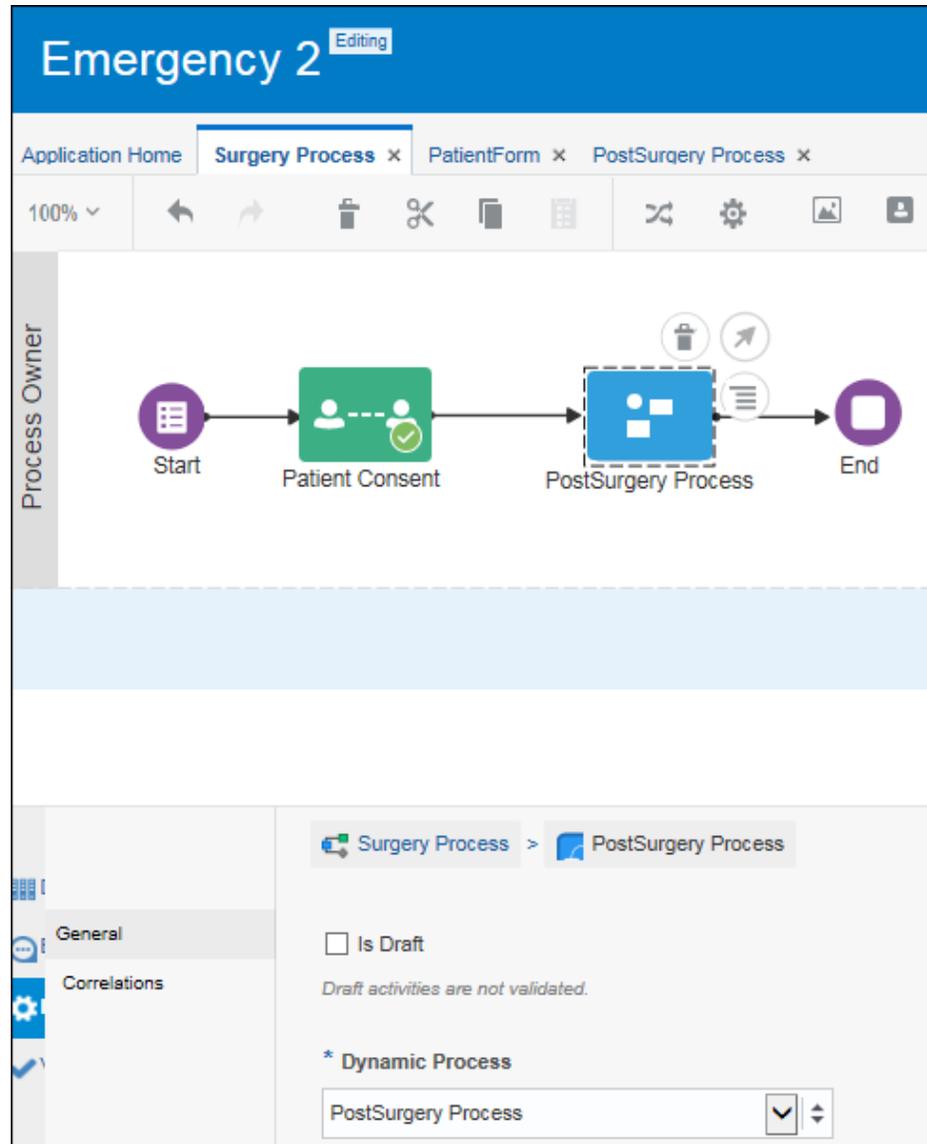
- Task 2: Repeat the steps of Task 1, only this time enter `Minor Complications` in the **Name** field and select `Minor` in the **Presentation** field.
- Task 3: Repeat the steps of Task 1, but enter `Major Complications` in the **Name** field and `Major` in the **Presentation** field.

6. Click **Save**.

Configure the Structured Process to Call the Dynamic Process

Now that you've created the processes and form presentations for users to perform tasks, and also defined the input data for the dynamic process, let's set up the structured process. This includes configuring the structured process to initiate the dynamic process, and configuring data association so that data flows from the structured process into the dynamic process.

1. Click the **Surgery Process** tab to open the structured process.
2. Associate the dynamic process with the dynamic process flow element.
 - a. Select the dynamic process element, click **Menu**  and then select **Open Properties**. The implementation pane opens at the bottom of the window.
 - b. In the implementation pane, select **PostSurgery Process** in the **Dynamic Process** field.

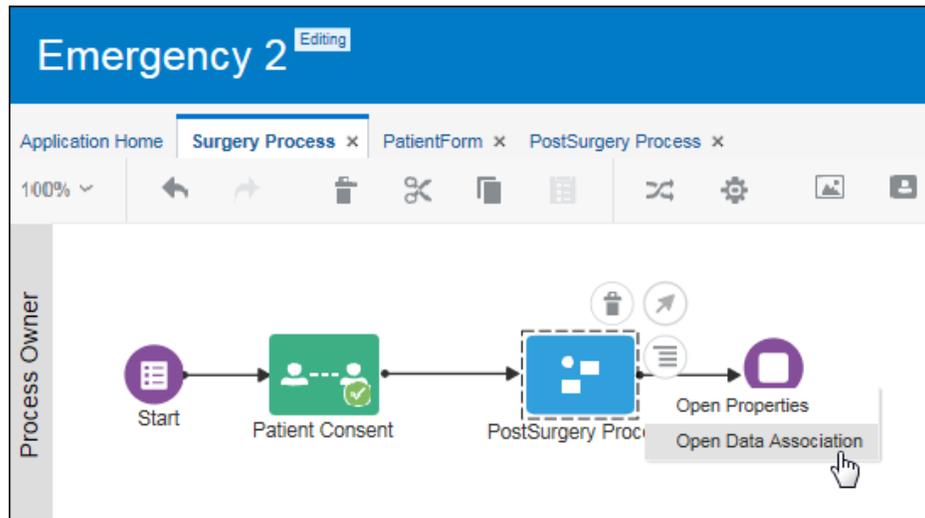


- c. Collapse the implementation pane to save your changes.
3. Configure the structured process' start event.
Now let's configure the Surgery Process' start event.
 - a. Select the Surgery Process' start event and open the related implementation pane.
 - b. Enter `Start the Surgery Process` in the **Title** field.
This title will appear at the process start in runtime.
 - c. Browse and select PatientForm in the **Form** field.
 - d. Select **Start** in the **Presentation** field.
 - e. Collapse the implementation pane to save your changes.
4. Configure the Approve human task.
 - a. Select the Patient Consent task and open the related implementation pane.
 - b. Enter `Patient Consent` in the **Title** field.

- c. Browse and select PatientForm in the **Form** field.
 - d. Select **Consent** in the **Presentation** field.
 - e. Collapse the implementation pane to save your changes.
5. Configure data association to ensure that input data from the structured process flows into the dynamic process.

In our example, we want the PostSurgery dynamic process to receive the patient's first and last name as the input data from the previous activity of the Surgery Process.

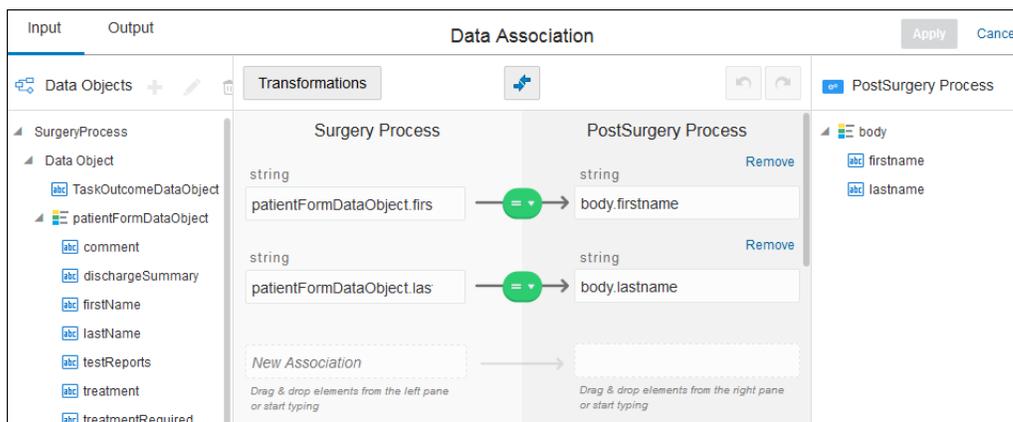
- a. Select the dynamic process element.
- b. Click **Menu**  and then select **Open Data Association**.



- c. In the Data Association editor, map the data between the Surgery Process (structured process) and the PostSurgery Process (dynamic process).

The left pane displays the source (data) objects from the Surgery Process. The right pane displays the payload the PostSurgery Process needs to perform its function.

- Map `patientFormDataObject.firstName` from the Surgery Process to `body.firstname` in the PostSurgery Process.
- Map `patientFormDataObject.lastName` in the Surgery Process to `body.lastname` in the PostSurgery Process.



- d. Click **Apply**.

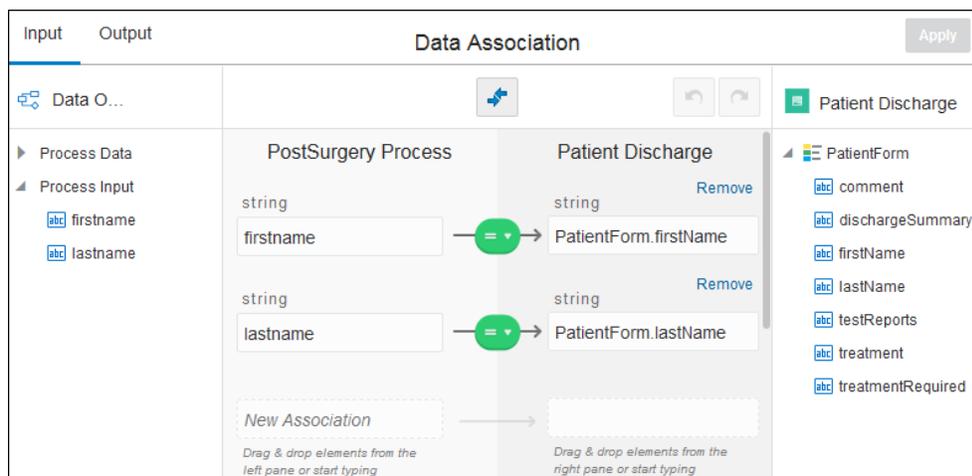
Configure Data Association for the Dynamic Process Activities

Define data input and output for the human task activities in your dynamic process by configuring data associations for them.

In the PostSurgery Process, set data associations for the Patient Discharge, Minor Complications and Major Complications human task activities, so that the patient's first and last name gets auto-populated when these activities are performed in runtime.

1. In the PostSurgery Process, select the Patient Discharge activity and click **More**.
2. In the menu, select **Data Association** and then **Input**.
3. In the Data Association editor, drag and drop data objects into the association fields.

Associate `firstname` and `lastname` from Process Input in the left pane to `firstName` and `lastName` from PatientForm in the right pane.



4. Click the **Output** tab and configure the data association, only this time the mapping will be reverse.

Associate `firstName` and `lastName` from PatientForm in the left pane to `firstname` and `lastname` from Process Input in the right pane.

5. Click **Apply**.
6. Repeat the steps above for the Minor Complications and Major Complications activities.

Try Out the Process Application in Test Mode

Now that you've set up the process application in design time, let's try out the process application in runtime, as users would.

- In the structured process, the patient receives an approve task to provide consent for the post surgery treatment phase, which initiates the post surgery dynamic process.
- A health worker uses the dynamic process to perform activities that fit the patient's post surgery status.

Begin by test activating.

1. Click **Test**.

A **Test Application** tab opens. Notice that a caution is displayed. Output data association isn't currently supported for dynamic process output. But the application can still be activated.

2. Test activate the application.
 - a. Click **Activate**.

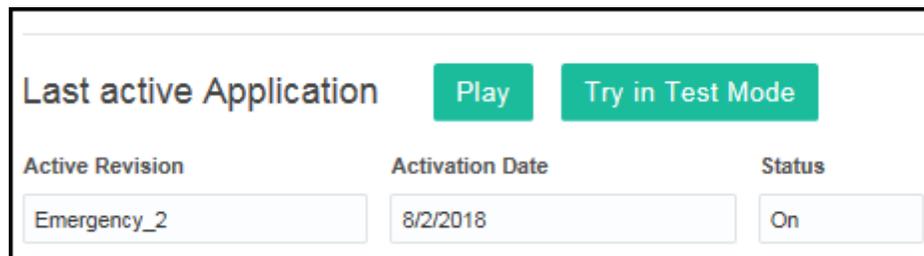
a. Click **Activate**.

- b. In the Activate to Test dialog box that displays, leave **Add Me to All Roles** selected, and click **Activate**.

When activating an application containing a structured process, you must map roles to users. Because you are test activating, you can skip mapping roles.

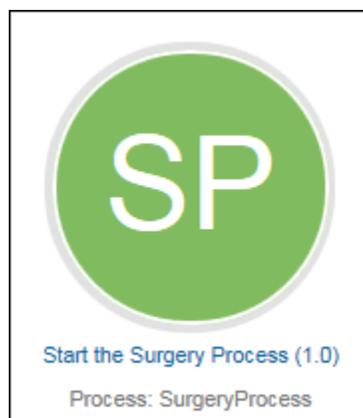
Notice that the **Play** and **Try in Test Mode** buttons became green and active. The **Play** option doesn't apply to process applications that use dynamic processes. While you can initiate the player, it will stop when the dynamic process player is reached, so its use is not recommended with dynamic process applications.

- c. Click **Try in Test Mode**.

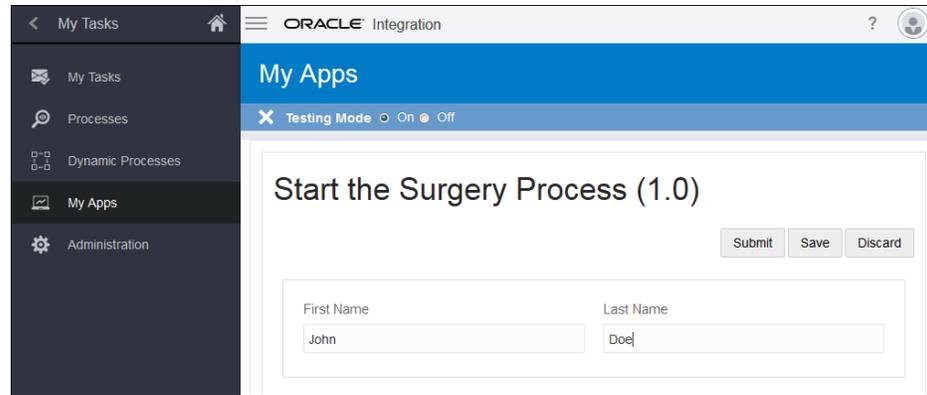


3. As the doctor, start the structured process.
 - a. From **My Tasks** (runtime), click your application on the My Apps page that displays after test activation starts.

- a. From **My Tasks** (runtime), click your application on the My Apps page that displays after test activation starts.



- b. In the start form that displays, enter the patient's first and last name, and click **Submit**.

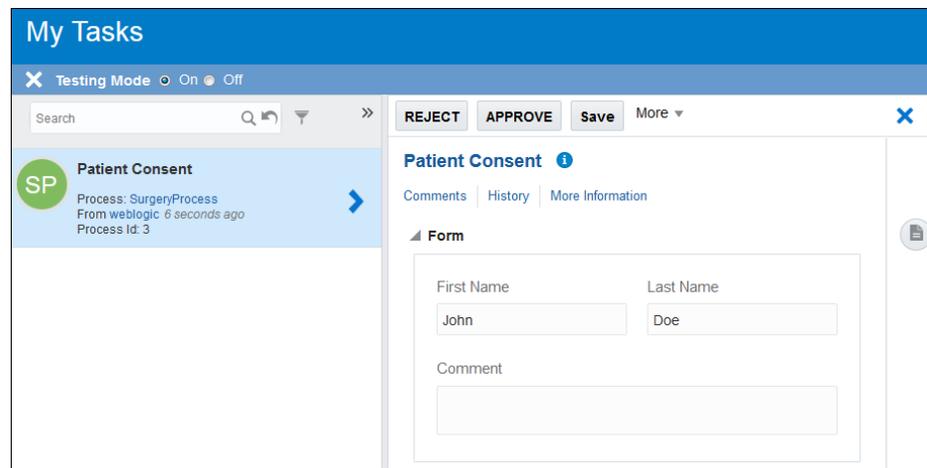


A message briefly displays that a process instance was created.

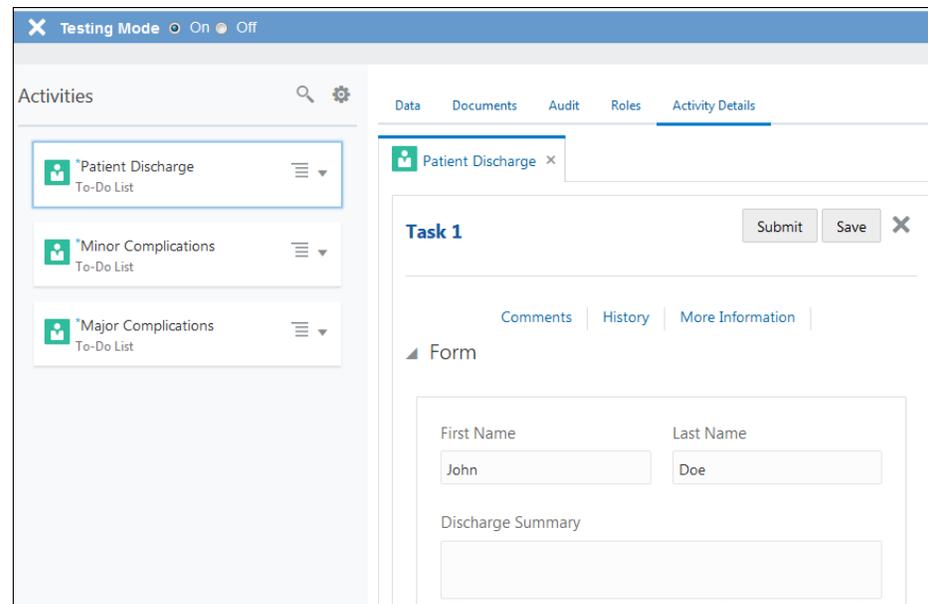
4. As the patient, open and approve your assigned consent task.

The structured process is running and the approve task generated an approval task for the patient.

- a. Click **My Tasks** and open the Patient Consent task from the My Tasks page by clicking the >. Notice that the patient's first and last name is auto-populated in the form.



- b. Click **Approve** to complete the task. A message displays that the action was processed successfully.
5. As a hospital worker, open and perform activities in the dynamic process.
After the approval task completed, the dynamic process was called, starting the dynamic process.
 - a. Click **Dynamic Processes** in the navigation pane.
 - b. On the Dynamic Processes page, open the PostSurgery Process and click each activity.



Notice that the patient's first name and last name was auto-populated and that the presentation you selected for each activity displays correctly.

- c. Complete one of the activities by entering information and clicking **Submit**.

Suppose that the patient experienced minor complications only, then select the minor activity. When you submit the activity, it no longer appears in the Activities list indicating that it is completed.

- d. Click **Close** to complete the PostSurgery dynamic process.

Congratulations! You've just successfully configured a process application that calls a dynamic process from a structured process.

Model a Dynamic Process in Design Time

Within an application, you can create dynamic processes to model flexible and data-intensive business scenarios. For a dynamic process, knowledge workers or process participants can define the process flow in runtime by making real-time decisions and carrying out relevant tasks.

Use the following approach as a general guideline to model a dynamic process; however, you may choose to complete some of these steps in any order. Iteratively refine the dynamic process and its elements as you model them to suit your requirements.

1. To begin with, create a dynamic process with the required details. See [Create a Dynamic Process](#).
2. Within the process, create activities that make up the process. See [Create Activities](#).
3. You can choose to divide the process into several segments by creating stages and group activities into these stages. See [Create Stages](#).
4. Define input and output arguments for the process. See [Define Process Input and Output](#).
5. Define data objects to be used within the process. See [Define Process Data Objects](#).
6. Create process roles to define responsibilities. See [Create Process Roles](#).

7. Define properties for each stage and activity. See [Define Stage Properties](#) and [Define Activity Properties](#).
8. Finally, define conditions for process completion or termination. See [Define Process Completion and Termination](#).

Create a Dynamic Process

A dynamic process is basically a collection of activities or tasks without a predetermined sequence of execution. It provides flexibility to knowledge workers to define the process flow at runtime based on the information available to them.

To create a dynamic process within an application:

1. In the Application Home page, click **Create a Dynamic Process** and then click **Start from scratch** to open the Create Dynamic Process dialog box.

Optionally, click **Create** and select **New Dynamic Process** to open the Create Dynamic Process dialog box.

The screenshot shows the 'Create Process' dialog box. On the left, there is a 'Name' field containing 'Emergency Process' and a 'Description' field. On the right, under 'Select a Pattern', five options are listed: 'None' (selected), 'Todo List', 'Single Stage To-Do List', 'Two Sequential Stage To-Do List', and 'Two Parallel Stage To-Do List'. At the bottom left, there is a checked checkbox for 'Open Immediately'. At the bottom right, there is a 'Create' button.

2. Enter a name for the process. The field is populated with a default name, such as DynamicProcess, DynamicProcess1, and so on. You can use this name or change it.
3. In the **Select a Pattern** field, leave **None** selected to create a dynamic process from scratch, or select a pattern to start with one of the To-Do list scenarios.
4. Review the optional settings and make your changes, if required.
 - a. In the **Description** field, enter helpful information about this process, what it's all about, and why you would use it.
 - b. If you want to create the process but not edit it right now and instead return to the **Application Home** tab, deselect the **Open Immediately** check box. For

example, you may want to create several processes at once before you edit any of them. You can select and edit your process at any time.

5. Click **Create** and the dynamic process is created.

Deleting a Dynamic Process

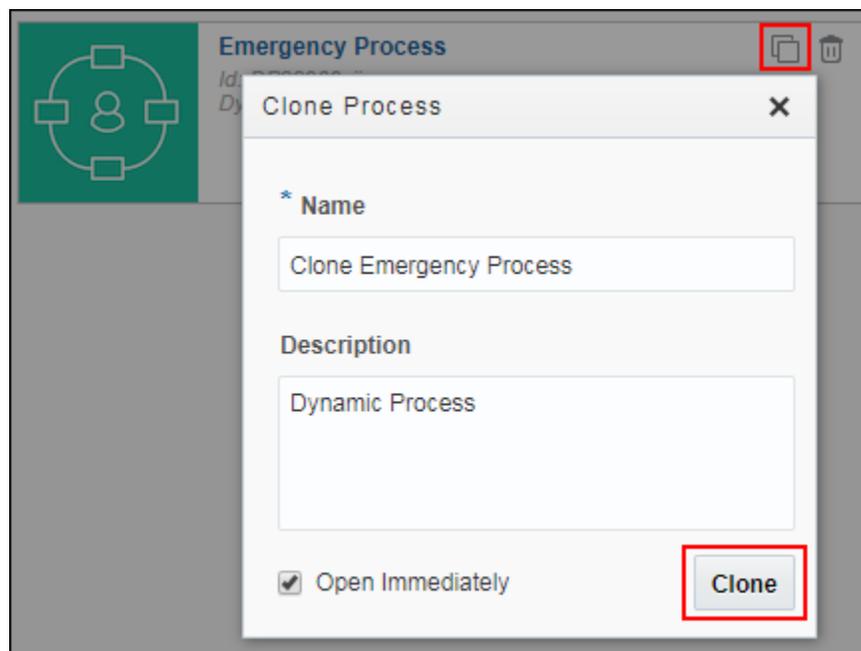
To delete a dynamic process, navigate to the **Processes** tab in Application Home page and click **Delete**  on a process that you wish to delete.

Clone a Dynamic Process

You can clone a dynamic process and create an exact copy of the process. This suits business needs where you have to quickly create a copy of an existing dynamic process.

When you clone a dynamic process, the cloned process will have all the activities and properties that were configured in the original process. Once the process has been cloned, the cloned process will be like a standalone process. You can make changes and customize it to your business needs.

Click the **Clone**  icon to open the Clone Process dialog.



Enter the name in the **Name** field and a suitable description in the **Description** field. By default, the name of the process will be Clone <original process name>. For example: *Clone Emergency Process*. Click **Clone** to create an exact replica of the dynamic process.

Create Activities

Activities represent the actions for a process to execute. The **Activities** drop-down menu on the process canvas contains all the activities that can be used within a dynamic process.

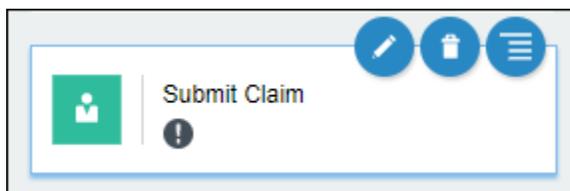
Activities are divided into the following types:

- **Human**  – Represents an activity where a process participant is required to perform the work. The task can be a simple interaction, such as filling out a form, or part of a more complicated workflow that requires input from multiple process participants.
- **Process**  – Represents invoking of a structured process within a dynamic process.
- **Service**  – Represents invoking of an external service, such as an OIC integration or a REST connector.
- **Milestone**  – Represents a sub-goal within a process. Milestones are typically defined to track progress of a process.
- **Micro Process**  – Represents invoking of a process present in another application. See [Create and Use Micro Processes](#).
- **Integrations**  – Represents invoking of an external application or service. The **Integrations** menu displays the following items:
 - REST integrations  : All REST connectors created within the application.
 - OIC integrations  : All OIC integration connectors created within the application.

For a REST or OIC integration connector to be displayed under this menu, you must set its visibility accordingly. See [Create REST and Web Service Connectors](#) and [Work with Integrations](#).

To create an activity, select the type of activity that you want to create from the **Activities** drop-down menu, provide a suitable name, and click **Add Activity** . You can also use the search bar in the **Activities** menu to find a particular type of activity to create.

An example human task activity is shown in the following figure:



Select the activity to perform various actions on it. The actions available for each activity are listed as follows:

- **Edit**  – Click this icon to edit the properties of the activity. See [Define Activity Properties](#).
- **Delete**  – Click this icon to delete the activity.
- **Menu**  – Click this icon to reveal the following additional actions:
 - **Roles**: Use this action to access the activity's **Roles** tab.
 - **Change Type**: Use this action to change the activity type. Select a different activity from the available options.

- **Data Association:** Use this action to define input and output for activities that require them. See [Configure Data Association](#).
- **Open Form:** If a human task activity has a web form associated with it, use this action to edit or view the web form. If no web form is associated, this option is grayed out.
- **Open Process:** If a process activity has a structured process associated with it, use this action to edit or view the structured process. If no structured process is associated, this option is grayed out.

To associate a form or structured process with an activity, see [Define Activity Properties](#).

In addition, you can select an activity and move it within the stage or drag and drop it outside the stage or into another stage.

Create Stages

Stages are segments that a process can be divided into. A stage acts as a container using which you can group and organize activities in a logical way. Typically, activities that must be completed in order for a milestone to be achieved are grouped into a stage.

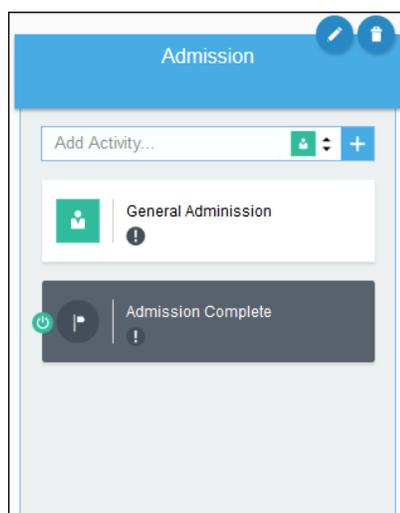
The execution of stages is not necessarily sequential. Stages can be activated in parallel or upon occurrence of an event. See [About Process and Plan Item Lifecycles](#).



Note:

Within a process, activities can exist without being grouped into a stage. Such activities are called *Global* activities.

To create a stage on the process canvas, enter a suitable name in the **Add Activity** box, and click **Add +** next to it. An example stage with some activities grouped into it is shown in the following figure:



Select the stage to perform actions on it. The actions available for each stage are listed as follows:

- **Edit**  – Click this icon to edit properties of the stage. See [Define Stage Properties](#).
- **Delete**  – Click this icon to delete the stage.

In addition, you can select a stage and move it to reorganize your process.

Define Process Input and Output

Define input and output arguments for a dynamic process to communicate or exchange data with other processes and services.

After defining the input arguments for your process, you must map them to data objects of your process or its flow elements. Similarly, data from a flow element in the process can be mapped to output arguments as the output of the process. See [Configure Data Association](#).

In addition, you can use a web form as the input to a process to start process execution when a user submits the form. The form is designed to get input from the user and present information relevant to the workflow.

To define the process input:

1. On the process page, click **Process Input** .
2. Choose one of the following options in the Start the Dynamic Process dialog:
 - **With Data Only**: Select to define input arguments to the process. Use this option to start a process instance with data from other processes or REST services.
 - **With Form**: Use this option to trigger a process instance using a web form or REST service.
3. If you select the **With Data Only** option, click **Add Argument** to define new arguments.
 - a. Provide a name to identify the argument and select a data type for the same. You can use simple or pre-defined business data types.
 - b. Click **Delete**  to remove existing arguments from the list.
4. If you select the **With Form** option:
 - a. Provide a form title to display to the user at runtime.
 - b. In the **Form** field, select a web form that you want to use as input.
 - c. In the **Presentation** field, select a presentation within the form to display to the user.

Note:

When you use a form as input, a business object to store the form data is automatically created and displayed under the Interface Argument section within the dialog. This input business object is also displayed in the Data pane. See [Define Process Data Objects](#).

- d. Edit the name of the business object, if required.

- Click **Define** to save changes and close the window.

The following figure shows a web form defined as process input:

To define the process output:

- On the process page, click **Process Output** .
- In the Start the Dynamic Process dialog, click **Add Argument** to define new arguments.
 - Provide a name to identify the argument and select a data type for the same. You can use simple or pre-defined business data types.
 - Click **Delete**  to remove existing arguments from the list.
- Click **Define** to save changes and close the window.

You can also define or edit the dynamic process input and output from the process' properties pane. Access the dynamic process' properties pane by clicking **Edit Properties**  on the top right corner of the process page.

- On the process page, click **Edit Properties** .

The dynamic process' properties pane opens.
- In the dynamic process' properties pane, click the **Interfaces** tab.
- Under the **Interfaces** tab:
 - Define the dynamic process input or output. Click  in **Input** to open the Start the Dynamic Process dialog box and define the process input. Click  in **Output** to define the process output.
 - Modify existing input or output data of the dynamic process. Use **Edit**  or **Delete**  options in **Input** or **Output** to modify existing input or output data.

Define Process Data Objects

Data objects are variables used to store the information related to your process. They also define the type of information used by the process.

The value of a data object can be different for every instance of a process. However, the structure of the data object is the same for all process instances.

When you define a process you can define data objects to store information. In addition, you can also define in which part of the process you assign a value to these data objects. When you create an instance, the process engine assigns *Null* as the default value for all the data objects defined for that process. Thereafter, the activities in the process assign values to these variables.

You can create two types of data objects within a dynamic process:

- **Simple:** Simple data objects define the basic types of variables that you can use within your process. They can be used separately within your process or they can be combined into complex data objects.
- **Business:** Business objects enable you to create data structures based on simple data objects. For example, you can create a business data object called Patient that contains different data types for *patient name*, *age*, and *pincode*.

To create a new data object:

1. Click **Data Objects** on the process page.
2. In the Data pane, click **Create Data Object** .
3. In the Create Data Object dialog, enter a name to identify the data object.
4. Select a data type to specify the type of data that the variable can store.
 - a. If you select **Simple**, select a simple data type available in the drop-down list.
 - b. If you select **Business**, you can select from a set of pre-defined business data types available in the drop-down list.
5. Click **Create** to save changes and close the dialog.

Use **Edit**  or **Delete**  to modify an existing data object from the Data pane.



Note:

In addition to data objects, the input and output data that you define for the process also show up in the Data pane.

Create Process Roles

Process roles define functional categories that correspond to job functions or responsibilities within your organization.

Create roles and assign process participants to it to define their permissions within the process. For example, you can use roles to authorize participants to perform human tasks, take discretionary actions (such as approve or reject), and influence the process flow during runtime.

While designing a dynamic process, it is essential to determine the type of participants and roles required to complete human tasks. Subsequently, you can define custom roles to suit your requirements. Here are some example process roles:

- **Doctor:** This role can contain one or more participants that are allowed to perform human tasks that require skills of a medical practitioner.
- **Patient:** This role can contain one or more participants that are allowed to provide information about their health through a human task, schedule appointments, and so on.



Note:

Roles defined at the process level apply to all elements within it; however, if a user is assigned a new role at the plan item (stage or activity) level, then the permissions provided at the plan item level override the permissions provided at the process level for that particular user within that plan item.

When you create a new dynamic process, the following two process roles are created by default. You can add users to these roles or create additional roles according to your requirements.

| Role | Permission Granted | Description |
|----------------|--------------------|--|
| Process Owner | All | Process owners typically manage and monitor deployed business processes and alter task flow as needed. For details on permissions, see Process Permissions . |
| Process Viewer | View | Process viewers are usually responsible for reporting on the current process instance status. |

To create a new process role:

1. Click **Roles** on the process page.
2. In the Roles pane, click **Create Role** .
3. In the resulting pane, enter a suitable name for the role, for example, Doctor.
4. Click the **Members** field and select users to add to the role.
5. Click **Add Permissions**  to create a new set of permissions for the role.
 - a. Enter a suitable name for the set in the **Name** field.
 - b. Click the **Items** field, select **Process** for a process-wide role.
 - c. In the **Actions** field, select actions that users assigned to this role can perform on the process. For a detailed description of all actions available for a process, see [Process Permissions](#).
 - d. You can also create a data condition to activate this set of permissions. Click **Add Permission Condition** and choose one of the available condition types from the resulting dialog box.
 - **Simple:** Select to define a simple condition based on a process variable's value in the payload.
 - **Decision:** Select to define a condition based on a decision model. Choose the decision model and the service you want to use (only the models for which

you've created a connector within the application show up in the drop-down menu), specify the process variable you want to pass as input to the model, and, finally, define a condition using the model's output (*body.interpretation*) or problems array (*body.problems*). See [Configure a Decision Sentry in a Dynamic Process](#).

- **REST**: Select to define a condition based on a REST connector. You can also use OIC integrations of the type REST to define conditions. Specify the integration you want to use in the **Integration** field. If you choose a REST connector defined within the process application (that is, a native connector), specify a resource and an operation for it too. Map a process variable as input to the connector's request body or parameter, and then define a condition using the connector's response body.

The following figure shows an example decision condition defined:

The screenshot shows a 'Create Data Condition' dialog box with the following fields and values:

- Name:** Data Condition
- Condition Type:** Decision (selected)
- Decision Model:** Dental
- Decision Service:** GetOutput
- Map Input Values:**

| Input | Variable |
|-------|----------|
| body | employer |
- Condition:** body.interpretation is equal to reportedBy

Note:

- In Decision and REST condition types, you can create multiple data triggers and form a logical expression using AND or OR operators. Click the drop-down arrow to switch between operators.
- A data condition is specific to the set of permissions in which it is defined.

You can create more than one set of permissions for a particular role.

6. Click **Create** to save changes.

Use **Edit** or **Delete** to modify an existing role from the Roles pane. The following figure shows an example of a process role:

 **Note:**

- Adding members and permissions to a role in design time ensures that they're retained for all deployments of the process application in runtime. To temporarily assign users to a role or modify permissions within a particular instance of the process, see [Override Roles Assigned to a Process](#).
- While redeploying an application to production, if you choose to overwrite an existing version, the existing runtime roles and permissions for that version are retained. You cannot overwrite roles and permissions for an existing version of an application deployed to production.
- While deploying an application for test, the existing test-partition roles and permissions for that application are always overwritten.

Process Permissions

The following table defines all permissions that can be granted to a user or group of users over a process.

| Permission | Description |
|------------|--|
| Create | Users granted this permission can: <ul style="list-style-type: none"> • View the process. • Create a new process instance. |

| Permission | Description |
|---------------------|---|
| View | Users granted this permission can: <ul style="list-style-type: none"> View the process. View a process instance. View all stage and activity instances. View data unless specifically revoked. View roles and permissions for the current process instance. |
| Update | Users granted this permission can: <ul style="list-style-type: none"> Update, complete, or terminate a process instance. Add, enable, disable, start, or complete stages and activities for a process instance, unless specifically revoked for a stage or activity. Update data for a process instance unless specifically revoked. Update roles and permissions for the current process instance. |
| Download Document | Users granted this permission can view and download documents from folders, unless specifically revoked for a document folder. |
| Contribute Document | Users granted this permission can view, download, and upload documents to all folders unless specifically revoked for a document folder. |
| Access Document | Users granted this permission can read documents from folders, unless specifically revoked for a document folder. |
| All | Users granted this permission can perform all of the above actions. |
| None | Users assigned this permission cannot perform any actions on the process. |

Define Stage Properties

You can customize a stage to suit your requirements by defining several attributes or properties, such as markers, conditions, or roles and permissions specific to the stage.

Select a stage and click **Edit Properties**  to reveal the Properties pane. Use this pane to define or modify properties of the stage. This pane consists of the following three tabs:

- [General](#)
- [Conditions](#)
- [Roles](#)

Define General Properties for a Stage

In the **General** tab, you can edit the stage name and provide a helpful description about the stage. In addition, you can also enable markers for the stage.

Properties: Diagnosis

General Conditions Roles

id: dp482y6pkvko8n6ac

Name *

Diagnosis

Description

Markers ?

Repeatable

Required

Manually Activated

Auto Complete

Enabling Markers for a Stage

Using markers, you can control transitions of a stage between states. In addition, you can also specify if the execution of a stage is mandatory for the process to complete or if the execution must be repeated under certain conditions.

Select a marker's check box to enable it. When a marker is enabled, a corresponding decorator appears on the stage.

When you enable a marker without conditions, it applies to a stage by default. However, you can also enable markers based on data conditions defined using process variables, decision models, or REST connectors. To enable a marker based on data conditions:

1. Select the marker's check box.
2. Click **Edit Property**  next to the marker.
3. In the resulting pane, click **Create Data Condition**  to create a new condition. The marker is enabled for the stage only when this condition is satisfied.

In the Create Data Condition dialog box, choose one of the available condition types.

- **Simple:** Select to define a simple condition based on a process variable's value in the payload.
- **Decision:** Select to define a condition based on a decision model. Choose the decision model and the service you want to use (only the models for which you've created a connector within the application show up in the drop-down menu), specify the process variable you want to pass as input to the model, and, finally, define a condition using the model's output (*body.interpretation*) or problems array (*body.problems*). See [Configure a Decision Sentry in a Dynamic Process](#).
- **REST:** Select to define a condition based on a REST connector. You can also use OIC integrations of the type REST to define conditions. Specify the integration you want to use in the **Integration** field. If you choose a REST connector defined within the process application (that is, a native connector), specify a resource and an operation for it too. Map a process variable as input to the connector's request body or parameter, and then define a condition using the connector's response body.

The following figure shows an example decision condition defined:

4. Click **Save** to close the pane.

 **Note:**

In Decision and REST condition types, you can create multiple data triggers and form a logical expression using AND or OR operators. Click the drop-down arrow to switch between operators.

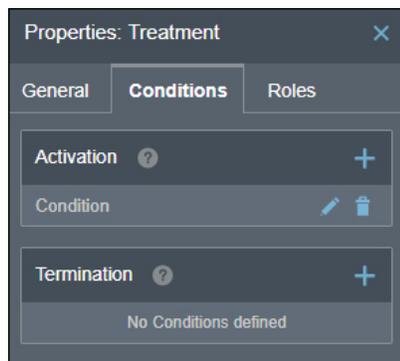
Markers and state transitions of a stage are interlinked. To know how markers effect various transitions, see [About Process and Plan Item Lifecycles](#). The following table lists all markers available for a stage:

| Marker | Description |
|--|---|
| Repeatable  | <p>Enable the Repeatable marker to repeatedly execute an entire stage.</p> <ul style="list-style-type: none"> • If this marker is enabled with no data condition, a new instance of the stage is created each time the execution of the stage is complete. • If this marker is enabled with a data condition, a new instance of the stage is created only when the data condition is fulfilled and the stage instance transitions away from the <i>Available</i> state into the next state. |
| Required  | <p>Enable this marker to make execution of a stage obligatory for the process to complete.</p> |

| Marker | Description |
|--|--|
| Manually Activated  | <p>Enable the Manually Activated marker if you require a stage to be started by a process participant.</p> <ul style="list-style-type: none"> If this marker is enabled, the stage transitions from <i>Available</i> to <i>Enabled</i> (after fulfilling enablement conditions if any). From here, a process participant can move the stage into <i>Active</i> or <i>Disabled</i> states. If this marker is disabled, the stage automatically transitions from <i>Available</i> to <i>Active</i> (after fulfilling activation conditions if any). <p>To define enablement or activation conditions for a stage, see Define Conditions for a Stage.</p> |
| Auto Complete  | <ul style="list-style-type: none"> If the Auto Complete marker is enabled, a stage is automatically marked <i>Complete</i> if none of the activities within it are in the <i>Active</i> state and all required activities are <i>Completed</i>, <i>Terminated</i>, or <i>Disabled</i>. If the Auto Complete marker is disabled, a stage is marked <i>Complete</i> only if all activities are <i>Completed</i>, <i>Terminated</i>, or <i>Disabled</i>. <p>When this marker is disabled, you can also manually complete a stage if none of the activities is in the <i>Active</i> state, and all required activities are <i>Completed</i>, <i>Terminated</i>, or <i>Disabled</i>.</p> <p>Note: For a new stage, the Auto Complete marker is enabled by default.</p> |

Define Conditions for a Stage

In the **Conditions** tab, you can specify conditions to enable, activate, or terminate a stage in runtime.



Enablement

If the Manually Activated marker is enabled, you can define an enablement condition to control transition of a stage from *Available* to *Enabled*.

Activation

If the Manually Activated marker is disabled, you can define an activation condition to control transition of a stage from *Available* to *Active*.

An enablement or activation decorator  appears on the stage (or activity) that has these conditions defined.

Termination

You can define a condition to terminate an *Active* stage without its execution being complete. A termination decorator  appears on the stage (or activity) that has this condition defined.

Want to know more about state transitions of a stage? See [Stage or Activity State Model](#).

Conditions are triggered by events or by data or both. To create a condition:

1. Click the **Create Condition**  icon.
2. In the resulting pane, provide a suitable label for the condition.
3. To create an event trigger, click **Create Event**  and define a trigger from the available drop-down controls.

For example, if you want a stage to be activated only after its previous stage is complete, select **Previous Stage** in the **Stage** or **Activity** drop-down menu and select **Complete** in the **Actions** drop-down menu.

Similarly, you can create event triggers based on various actions on other stages (or activities) within your process. The following table details all the runtime actions or transitions that you can select from the **Actions** drop-down menu. To know when these actions precisely occur in a stage or activity lifecycle, see the figure in the [Stage or Activity State Model](#) section.

| Selection | Triggers the Condition When... |
|--------------|---|
| Complete | The selected plan item completes execution. |
| Auto-Start | The selected plan item is automatically started. (Auto transition between <i>Available</i> to <i>Active</i>) |
| Create | The selected plan item is instantiated. |
| Disable | The disable action occurs on the selected plan item. |
| Enable | The enable action occurs on the selected plan item. |
| Exit | The exit transition occurs on the selected plan item. |
| Fault | The selected plan item encounters a fault. |
| Manual Start | The selected plan item is manually started. |
| Reactivate | The selected plan item is reactivated. |
| Re-Enable | The selected plan item is re-enabled. |
| Terminate | The selected plan item's termination condition is met. |

The following table details the options that you can select from the **Actions** drop-down menu for a milestone activity. To know when these actions precisely occur in a milestone's lifecycle, see the figure in [Milestone State Model](#).

| Selection | Triggers the Condition When... |
|-----------|---|
| Occur | The selected milestone's completion condition is met. |
| Create | The selected milestone is instantiated. |

 **Note:**

- You can create event triggers for a plan item based only on its siblings. That is, for an activity within a stage, you may create an event trigger based on other activities within the same stage. Similarly, you can define event triggers for a stage based on other stages or global activities only.
- When you create multiple event triggers for the same stage, a logical AND expression is formed which includes all triggers.

4. To create a data trigger, click **Create Data Condition**  and choose one of the available condition types in the resulting dialog box.
 - **Simple:** Select to define a simple condition based on a process variable's value in the payload.
 - **Decision:** Select to define a condition based on a decision model. Choose the decision model and the service you want to use (only the models for which you've created a connector within the application show up in the drop-down menu), specify the process variable you want to pass as input to the model, and, finally, define a condition using the model's output (*body.interpretation*) or problems array (*body.problems*). See [Configure a Decision Sentry in a Dynamic Process](#).
 - **REST:** Select to define a condition based on a REST connector. You can also use OIC integrations of the type REST to define conditions. Specify the integration you want to use in the **Integration** field. If you choose a REST connector defined within the process application (that is, a native connector), specify a resource and an operation for it too. Map a process variable as input to the connector's request body or parameter, and then define a condition using the connector's response body.

The following figure shows an example decision condition defined:

Note:

In Decision and REST condition types, you can create multiple data triggers and form a logical expression using AND or OR operators. Click the drop-down arrow to switch between operators.

5. Click **Create** to save the changes.
6. Click **Delete** to delete a condition or triggers within it.

The following figure shows an activation condition defined for a stage:

Create Roles for a Stage

Roles defined at the process level are automatically inherited by a stage. In addition, you can also define roles specific to a stage. Want to know more about roles? See [Create Process Roles](#).

If there are conflicting permissions granted to a user at the process and stage levels, permissions granted at the stage level prevail within that stage.

To create a new role at the stage (or activity) level:

1. Click the **Create Role** link.
2. In the resulting pane, enter a suitable name for the role, for example, Nurse.
3. Click the **Members** field and select users to add to the role.
4. Click **Add Permissions**  to create a new set of permissions for the role.
 - a. Enter a suitable name for the set in the **Name** field.
 - b. Click the **Items** field, select stages (or activities) for which you want the role and associated permissions to apply.
 - c. In the **Actions** field, select actions that users assigned to this role can perform on stages (or activities) selected previously. For a detailed description of all actions available for a stage or activity, see [Stage or Activity Permissions](#).
 - d. You can also create a data condition to activate this set of permissions. Click **Add Permission Condition** and choose one of the available condition types from the resulting dialog box.
 - **Simple**: Select to define a simple condition based on a process variable's value in the payload.
 - **Decision**: Select to define a condition based on a decision model. Choose the decision model and the service you want to use (only the models for which you've created a connector within the application show up in the drop-down menu), specify the process variable you want to pass as input to the model, and, finally, define a condition using the model's output (*body.interpretation*) or problems array (*body.problems*). See [Configure a Decision Sentry in a Dynamic Process](#).
 - **REST**: Select to define a condition based on a REST connector. You can also use OIC integrations of the type REST to define conditions. Specify the integration you want to use in the **Integration** field. If you choose a REST connector defined within the process application (that is, a native connector), specify a resource and an operation for it too. Map a process variable as input to the connector's request body or parameter, and then define a condition using the connector's response body.

The following figure shows an example decision condition defined:

Note:

In Decision and REST condition types, you can create multiple data triggers and form a logical expression using AND or OR operators. Click the drop-down arrow to switch between operators.

You can create more than one set of permissions for a particular role.

5. Click **Create** to save changes.

Use **Edit** or **Delete** to modify an existing role from the **Roles** tab.

The following figure shows a role created for a stage called Diagnosis:

 **Note:**

Adding members and permissions to a role in design time ensures that they're retained for all deployments of the process application in runtime. If you require to temporarily assign users to a role or modify permissions in a particular deployment, you can accomplish this in runtime. See [Override Roles Assigned to a Process](#).

Stage or Activity Permissions

The following table defines all permissions that can be granted to a user or group of users over a stage or activity.

| Permission | Description |
|------------|---|
| View | Users granted this permission can: <ul style="list-style-type: none"> View the stage or activity. View the executing instance of the stage or activity. |
| Update | Users granted this permission can enable, disable, start, and complete the stage or activity for a process instance. |
| All | Users granted this permission can perform all of the above actions. |
| None | Users assigned this permission cannot perform any actions on the stage or activity. |

Define Activity Properties

You can customize an activity to suit your requirements by specifying markers, conditions, and roles.

For a human task activity, you can additionally configure the assignment of the human task, the form to be used to display its information, its due date and priority, among other things. For a process activity, you can select the structured process that you would like to invoke from within the dynamic process.

Select an activity and click **Edit Properties**  to reveal the Properties pane. Use this pane to define or modify properties of the activity. This pane consists of the following three tabs:

- [General](#)
- [Conditions](#)
- [Roles](#)

Define General Properties for an Activity

In the **General** tab, you can edit the activity name and provide a helpful description about the activity. In addition, you can also enable markers for the activity. For human, process, and service tasks you can also specify certain implementation details, see the following topics:

- [Specify Implementation Details for a Human Task Activity](#)
- [Specify Implementation Details for a Process Activity](#)
- [Specify Implementation Details for a Service Task Activity](#)
- [Specify Implementation Details for an Integration Activity](#)

Properties: Request Examination

General Conditions Roles

id: *dp17eiin9hwa7jcy*

Name *
Request Examination

Description
The patient requests an examination.

Markers ?

Repeatable
 Required
 Manually Activated

Implementation

General
Reminders
Escalation and Expiration

Enabling Markers for an Activity

Using markers, you can control transitions of an activity between states. In addition, you can also specify if the execution of an activity is mandatory for the process or stage to complete or if the execution must be repeated under certain conditions.

Select a marker's check box to enable it. When a marker is enabled, a corresponding decorator appears on the activity.

When you enable a marker without conditions, it applies to an activity by default. However, you can also enable markers based on data conditions defined using process variables, decision models, or REST connectors. To enable a marker based on data conditions:

1. Select the marker's check box.
2. Click **Edit Property**  next to the marker.
3. In the resulting pane, click **Create Data Condition**  to create a new condition. The marker is enabled for the activity only when this condition is satisfied.

In the Create Data Condition dialog box, choose one of the available condition types.

- **Simple:** Select to define a simple condition based on a process variable's value in the payload.
- **Decision:** Select to define a condition based on a decision model. Choose the decision model and the service you want to use (only the models for which

you've created a connector within the application show up in the drop-down menu), specify the process variable you want to pass as input to the model, and, finally, define a condition using the model's output (*body.interpretation*) or problems array (*body.problems*). See [Configure a Decision Sentry in a Dynamic Process](#).

- **REST:** Select to define a condition based on a REST connector. You can also use OIC integrations of the type REST to define conditions. Specify the integration you want to use in the **Integration** field. If you choose a REST connector defined within the process application (that is, a native connector), specify a resource and an operation for it too. Map a process variable as input to the connector's request body or parameter, and then define a condition using the connector's response body.

The following figure shows an example decision condition defined:

4. Click **Save** to close the pane.



Note:

In Decision and REST condition types, you can create multiple data triggers and form a logical expression using AND or OR operators. Click the drop-down arrow to switch between operators.

Markers and state transitions of an activity are interlinked. To know how markers effect various transitions, see [About Process and Plan Item Lifecycles](#). The following table lists all markers available for an activity:

| Marker | Description |
|--------------------|--|
| Repeatable | <p>Enable the Repeatable marker to repeatedly execute an activity.</p> <ul style="list-style-type: none"> If this marker is enabled with no data condition, a new instance of the activity is created each time the execution of the activity is complete. In this case, the activity is repeated every time it completes, indefinitely. If this marker is enabled with a data condition, the activity will be repeated (that is, a new instance of the activity will be created) only when the data condition is fulfilled or resolves to True. The activity will not be repeated if the data condition is not fulfilled. |
| Required | <p>Enable this marker to make execution of an activity obligatory for the stage or process to complete.</p> <p>Note: For a new activity, this marker is enabled by default.</p> |
| Manually Activated | <p>Enable the Manually Activated marker if you require an activity to be started by a process participant.</p> <ul style="list-style-type: none"> If this marker is enabled, the activity transitions from <i>Available</i> to <i>Enabled</i> (after fulfilling enablement conditions if any). From here, a process participant can move the activity into <i>Active</i> or <i>Disabled</i> states. If this marker is disabled, the activity automatically transitions from <i>Available</i> to <i>Active</i> (after fulfilling activation conditions if any). <p>To define enablement or activation conditions for an activity, see Define Conditions for an Activity.</p> <p>Note: This marker is not available for Milestone activities.</p> |

Specify Implementation Details for a Human Task Activity

You can configure a human task activity by providing additional details under the Implementation section in the **General** tab.

You can configure the following details under each available option:

| Option | Additional Details |
|---------------------------|--|
| General | <ul style="list-style-type: none"> Select the Task Type Specify the Title and Task Summary Assign Tasks Associate a Web Form Associate an External UI Specify the Due Date and Priority Skip the Approval Chain |
| Reminders | Set Reminders |
| Escalation and Expiration | Schedule Task Expiration, Renewal, or Escalation |
| Notifications | Disable Notification Emails |

Select the Task Type

Create submit tasks to display a form for the user to complete and submit. Create approval tasks to display a form for the user to view and/or complete, and then perform an action such as approving it, or rejecting it, or a custom action that you define.

To select the task type:

1. Click **General** in the **General** tab.
2. Select **Submit** or **Approve** in the General pane according to your requirement.

For a submit task, the only allowed action is SUBMIT and the same is auto-populated in the **Action** field at the bottom of the pane. You can edit or change the name of the action (for example, OK).

If you're configuring an approval task, actions APPROVE and REJECT are populated by default in the **Action** field. You can also edit the default actions and add new ones. To add actions, enter their name in uppercase and separate them with commas, for example, MOREINFO, ONHOLD, and so on. You can later access the value of the selected action from the human task data association, using the predefined data object *outcome*.

Specify the Title and Task Summary

Enter a title to identify the human task, and a summary to describe it.

The title and the summary appear in the user's task list, so that they can easily identify the human task they're looking for without having to view the details. You can enter the title and the summary using plain text (literals) or generate it using expressions.

Assign Tasks

You can assign a human task to a specific user, to a group of users, or to the users in a certain role.

To assign a human task to a specific user or group of users:

1. In the General pane, click **Edit** in the **Assignees** field. The Assignees dialog box opens.
2. Click **Add**  to create new assignees.
 - In the **Identity Type** drop-down list, select one of the following: **User**, **Group**, or **Role**.
3. In the **Type** field, select **Literal** to specify the participant using names and choose an entry from the **Value** drop-down list.
4. To delete an assignee, select the assignee and click **Delete** .
5. Click **Save** to save changes and close the dialog. The number of assignees added is displayed in the **Assignees** field.
6. Click **Exclude the Following Participants** to specify exclusions. You can choose to exclude the following participants by selecting the corresponding check box:
 - **Creator**: The user who created the process.
 - **Previous Participants**: Users who have already acted within this task instance.
 - **Expression**: User specified through an expression.

Note:

If you exclude a user from a task, the user will only be able to view the task but not have the permission to act on it.

For an approval task, you also have to specify the task approvers drawing from the assignees. In the **Approvers** field, select task approvers from the following options:

- **Any Single Assignee:** Any of the assignees can complete the task by performing the actions specified in the **Action** field.
- **All Assignees in Parallel:** All assignees complete the task. When you select this option, the following two fields appear: **Percentage Required** and **Default Outcome**. For example, if you set the required percentage as 51% and the default outcome as APPROVE, and, if 55% of assignees reject the task, the outcome of the task is REJECT.
- **All Assignees in Sequence:** All assignees complete the task, albeit in a sequence. Also see [Skipping the Approval Chain](#).
- **Management Chain in Sequence:** Starting from the assignee, all members in the assignee's management chain complete the task in sequence. In the **Number of Levels** field, specify the level in the assignee's management chain up to which you want the task to hierarchically flow.

Associate a Web Form

You can associate a web form with a human task to display information for users to provide approvals or enter certain data for assigned tasks.

To associate a form:

1. In the General pane, select an existing web form from the **Form** drop-down list. All forms within the application appear in the list.
2. In the **Presentation** field, select a presentation within the form that you want to associate with the activity.

Associate an External UI

You can associate an external UI with a human task to display information to users, so that they can provide approvals or enter data for the task.

To associate an external form:

1. In the General pane, select an existing external UI connector from the **Form** drop-down list. All external UI connectors within the application appear in the list.
2. From the Data Association window, assign necessary values to path parameters, query parameters, and custom payload attributes that you need to pass to the external application.

See [Create an External UI Connection](#) for detailed steps on creating a new connector to an external form.

Specify the Due Date and Priority

You can specify a due date and a priority for a human task. After the due date is reached, the human task is marked as overdue.

To configure the due date and priority:

1. In the General pane, locate the due date and priority fields.

2. In the **Due Date** field, enter an interval to specify the amount of time the assignee has to complete the instance after the human task is triggered. Select one of the following options:
 - **Literal:** Use the format `##M##d##h##m`. For example:
 - One hour and thirty minutes: 1h30m
 - One day: 1d
 - Four months, two days, eleven hours and thirty minutes: 4M2d11h30m
 - **Expression:** Specify an expression to calculate the due date of the human task.
3. From the **Priority** drop-down list, select a priority. Available options are: **High**, **Normal**, and **Low**.

These options enable you to sort the Workspace task list based on the task priority. Additionally, in the Workspace task list, high priority tasks are marked with a red exclamation mark.

Skip the Approval Chain

For an approval task, you can skip the approval chain for a specified action.

For example, if you select approvers to be **All Assignees in Sequence** or **Management Chain in Sequence**, and the second approver out of four approvers rejects the task, you can skip the remaining two approvers.

To skip the approval chain:

1. Select the **Skip Approval** check box to activate the drop-down menu.
2. Select the action that you want to skip approval on.

If you add or remove an action in the **Action** text field, the drop-down menu for skipping approvals automatically updates to reflect the addition or deletion.



Set Reminders

You can send reminders to the assignees of a human task. You can specify the number of reminders to send, and the event that triggers the reminder.

To set reminders:

1. Click **Reminders** in the **General** tab. The Reminders pane appears.
2. From the **Reminder** drop-down list, select the number of times you want to send reminders to assignees to complete the task. The available options are:
 - **No Reminder**

- **Remind Once**
 - **Remind Twice**
 - **Remind Three Times**
3. If you choose to remind once, specify the time to wait before sending the reminder in the **Interval** field. In case of multiple reminders, use this field to specify the time to wait between reminders. You can specify this interval using the following options:
 - **Literal:** Use the format `##M##d##h##m`. For example:
 - One hour and thirty minutes: `1h30m`
 - One day: `1d`
 - Four months, two days, eleven hours and thirty minutes: `4M2d11h30m`
 - **Expression:** Specify an expression to calculate the interval.
 4. If you choose to send reminders, select an event to trigger the first reminder from the **When** drop-down list. The available options are:
 - **Before Expiration:** Send a reminder before the specified expiration time is reached. After the expiration date is reached, the human task doesn't appear on the task list.
 - **After Assignment:** Send a reminder immediately after assigning the human task to a specific user.
 - **Before Due Date:** Send a reminder before the specified due date for the human task is reached. After the due date, the human task is marked as overdue, but you can still access it from the task list.

Schedule Task Expiration, Renewal, or Escalation

You can configure a human task to never expire, to expire after a certain time, to renew the expiration time, or to escalate after a certain time has passed.

To configure an action to perform on a specific deadline:

1. Click **Escalation and Expiration** in the **General** tab. The Escalation and Expiration pane appears.
2. Use the radio buttons to specify if you want the human task to **Never Expire**, **Expire**, **Renew**, or **Escalate**.
 - **Never Expire:** The human task doesn't expire and if no user completes it. It remains in the users' task list for an indeterminate period of time.
 - **Expire:** The human task expires after the specified time and is no longer accessible from the task list.
 - **Renew:** When the specified time passes, the expiration date is extended for one more period until it reaches the specified amount of renewals allowed.
 - **Escalate:** When the specified time has passed, the human task is escalated to the specified escalation levels.
3. If you choose the human task to expire, renew or escalate, specify the interval to wait before performing this action. You can specify the interval to wait using one of the following methods:
 - **Literal:** Use the format `##M##d##h##m`. For example:

- One hour and thirty minutes: 1h30m
 - One day: 1d
 - Four months, two days, eleven hours and thirty minutes: 4M2d11h30m
 - **Expression:** Specify an expression to calculate the interval.
4. If you prefer to renew the human task, in the **Maximum Renewals** field, specify the maximum number of times to renew the human task.
 5. If you prefer to escalate the human task:
 - a. In the **Maximum Escalation Levels** field, specify how high in the management chain you want to escalate the human task.
 - b. In the **Highest Escalation Title** field, specify to what job title you want to escalate during repeated escalations. For example, manager or director. This is a free-form text field.

Disable Notification Emails

Task assignee(s) are notified when an event (such as assignment, approval, escalation, reminder, or reassignment) occurs.

Select the **Disable Notification** check box in the Notifications pane if you prefer not to notify assignees.

Specify Implementation Details for a Process Activity

For a process activity, associate the structured process that you would like to invoke from within the dynamic process. Want to know more about structured processes? See [Develop Structured Processes](#).

To associate a structured process, select the process and its start event in the Implementation section of the **General** tab. In addition, specify if the process is blocking or non blocking using the **Non Blocking** check box.

- **Non Blocking:** If you select the check box, the process activity completes immediately upon its activation after invoking its associated process; it does not wait for the structured process instance to complete.
- **Blocking:** If you set the process as blocking, the process activity remains *Active* until the structured process instance associated with it is completed. The process activity reaches the state *Complete* automatically when the invoked process instance completes.

Note:

- You can only use structured processes with the Message Start and Message End pattern within a dynamic process.
- You can use decision modeling in a dynamic process through a structured process.

! Important:

In runtime, when a user starts a dynamic process containing a process task activity, the structured process corresponding to this activity is automatically instantiated even if the user doesn't have permissions to instantiate it. However, in such cases, the user is only able to view the structured process instance and its associated tasks. To perform actions in the structured process instance, the user must be assigned a role within the process or be an administrator.

Specify Implementation Details for a Service Task Activity

For a service activity, associate the OIC integration or REST connector you'd like to invoke from within the dynamic process. Want to know more about integrations? See [Integrate with Applications and Services](#).

To associate an external service, select the service in the Implementation section of the **General** tab. If you choose a REST connector, specify a resource and an operation for it too.



The screenshot shows a dark-themed configuration panel titled "Implementation". It contains three dropdown menus stacked vertically. The first is labeled "Service" and has "Facebook" selected. The second is labeled "Resource" and has "Friends" selected. The third is labeled "Operation" and has "putFriends" selected.

Note:

- Currently, the service task activity supports only those REST services whose payload attribute names do not start with uppercase letters or contain special characters.
- Also, the service task activity doesn't support SOAP connectors.

Specify Implementation Details for a Micro Process Activity

For a micro process activity, associate the micro process link you'd like to invoke from your dynamic process. Want to know more about micro processes? See [Create and Use Micro Processes](#).

Note that in dynamic processes, you can call only asynchronous micro processes with message starts.

To associate a process link, click the activity and edit its implementation details on the **General** tab.

1. Select an asynchronous link in the **Micro Process** field.
Only the micro processes for which you've created links within the application show up in the **Micro Process** drop-down menu.
2. Select the start event of the micro process in the **Start Event** field.
3. Specify if the activity is blocking or non blocking using the **Non blocking** check box.
 - **Non Blocking:** If you select the check box, the micro process activity completes immediately upon its activation after invoking its associated process; it does not wait for the micro process instance to complete.
 - **Blocking:** If you set the micro process activity as blocking, it remains *Active* until the micro process instance associated with it completes. The micro process activity reaches the state *Complete* automatically when the invoked process instance completes.

Specify Implementation Details for an Integration Activity

After creating an integration activity, you can edit its implementation details on the **General** tab. You may select a different service for the activity, or choose a different resource or operation in case of REST connectors.



The screenshot shows a dark-themed panel titled "Implementation". It contains three dropdown menus stacked vertically. The first is labeled "Service" and has "Facebook" selected. The second is labeled "Resource" and has "Friends" selected. The third is labeled "Operation" and has "putFriends" selected.

An integration activity is non-blocking, that is, the activity completes immediately upon its activation, after invoking its associated external service; it doesn't wait for a response from the external service. Therefore, you'll not be able to specify output data associations for integration activities.



Note:

Currently, SOAP connectors do not show up on the **Integrations** menu.

Define Conditions for an Activity

In the **Conditions** tab, you can specify conditions to enable, activate, or terminate an activity in runtime.

For detailed steps to create a condition, see [Define Conditions for a Stage](#).

For a Milestone activity, you can define only a completion condition. See [Milestone State Model](#).

Create Roles for an Activity

Roles defined at the process and stage levels are automatically inherited by an activity. In addition, you can also define roles specific to an activity. Want to know more about roles? See [Create Process Roles](#).

If there are conflicting permissions granted to a user at process, stage, and activity levels, then the permissions granted at the activity level prevail within that activity.

To create a new role at the activity level, see [Create Roles for a Stage](#).

Create Simple Expressions

Use simple expressions to evaluate and perform calculations on data stored in data objects, using standard operators or functions.

You can create simple expressions while configuring properties for a human task and while configuring data associations for process or human tasks. Use the operators listed in this section to define simple expressions.

Topics:

- [String Operators](#)
- [Arithmetic Operators](#)
- [Date and Time Operators](#)
- [Boolean Operators](#)
- [Duration Operators](#)
- [Base64Binary Operators](#)
- [Array Operators](#)
- [Miscellaneous Operators](#)
- [Special Constants](#)
- [Casting](#)

String Operators

Here is a list of all string operators available to use in simple expressions.

| String Operator | Description | Simple Expression | Result |
|-----------------|------------------------|-----------------------|--------------|
| + | String concatenation | "pine" + "apple" | "pineapple" |
| == | Equals | "apples" == "apples" | <i>true</i> |
| != | Not Equals | "apples" != "oranges" | <i>true</i> |
| > | Greater than | "word" > "work" | <i>false</i> |
| >= | Greater than or Equals | "work" >= "work" | <i>true</i> |
| < | Less than | "word" < "work" | <i>true</i> |

| String Operator | Description | Simple Expression | Result |
|-----------------|--|-------------------------------|-----------|
| <= | Less than or Equals | "work" <= "work" | true |
| contains | Returns true if the first argument string contains the second argument string; otherwise, returns false. | "caramel".contains("ram") | true |
| endsWith | Returns true if the first argument string ends with the second argument string; otherwise, returns false. | "immutable".endsWith("table") | true |
| length | Returns the number of characters in a string. | "house".length() | 5 |
| lowerCase | Returns a string with all the characters in the argument converted to lower-case representation. | "Example".lowerCase() | "example" |
| startsWith | Returns true if the first argument string starts with the second argument string; otherwise, returns false. | "caramel".startsWith("car") | true |
| substring | Returns the substring of the first argument starting at the position specified in the second argument and continuing to the end of the string. | "care".substring(1) | "are" |
| substring | Returns the substring of the first argument starting at the position specified in the second argument and continuing to the end of the string. | "care".substring(1) | "are" |
| | Returns the substring of the first argument starting at the position specified in the second argument with length specified in the third argument. | "care".substring(0,3) | "car" |
| upperCase | Returns a string with all the characters in the argument converted to upper-case representation. | "Example".upperCase() | "EXAMPLE" |

Arithmetic Operators

You can use the following operators for numeric data types, such as byte, short, int, long, double, decimal, and float.

| Arithmetic Operator | Description | Simple Expression | Result |
|---------------------|------------------------|-------------------|--------|
| + | Addition | 2+8 | 10 |
| - | Subtraction | 7-4 | 3 |
| * | Multiplication | 3*4 | 12 |
| / | Division | 3/2 | 1.5 |
| % | Remainder | 3%2 | 1 |
| == | Equals | 12 == 13 | false |
| != | Not Equals | 12 != 13 | true |
| > | Greater than | 15 > 16 | false |
| >= | Greater than or Equals | 15 >= 15 | true |

| Arithmetic Operator | Description | Simple Expression | Result |
|---------------------|--|-------------------|--------------|
| < | Less than | 12 < 10 | <i>false</i> |
| <= | Less than or Equals | 12 <= 12 | <i>true</i> |
| abs | Returns the absolute value of a number | abs(-6) | 6 |

In addition, you can also use the following operators on non-integer data types, such as double, decimal, and float.

| Arithmetic Operator | Description | Simple Expression | Result |
|---------------------|---|-------------------|--------|
| floor | Returns the largest (closest to positive infinity) number that is not greater than the argument and that is an integer. | floor(5.60) | 5 |
| ceil | Returns the smallest (closest to negative infinity) number that is not less than the argument and that is an integer. | ceil(5.60) | 6 |
| round | Returns the number that is closest to the argument and that is an integer. | round(5.60) | 6 |

Date and Time Operators

You can use the following operators for date, time, and dateTime data types.

| Operator | Description |
|----------|---|
| + | Addition (use only when the second argument is a duration) |
| - | Subtraction (use only when the second argument is a duration) |
| == | Equals |
| != | Not Equals |
| > | Greater than |
| >= | Greater than or Equals |
| < | Less than |
| <= | Less than or Equals |
| format | Returns the formatted string of date-time using the provided XSLT 2.0 format picture. Examples: <ul style="list-style-type: none"> travelDate.format("[M01]/[D01]/[Y0001] [H01]:[m01] [z]") returns "02/24/2020 18:37 GMT+00:00" travelDate.format("[h1]:[m01] [P] on [MNn] [D]") returns "6:37 pm on February 24" |
| + | Addition |
| - | Subtraction |
| == | Equals |

In addition, there are some specific operators that you can use on each of date, time, and dateTime data types.

Date Operators

| Date Operator | Description |
|---------------|--|
| year | Returns a number representing the year component of the date-time argument. |
| month | Returns a number representing the month component of the date-time argument. |
| day | Returns a number representing the day component of the date-time argument. |

Time Operators

| Time Operator | Description |
|---------------|--|
| hours | Returns a number between 0 and 23, both inclusive, representing the hours component of the date-time argument. |
| minutes | Returns a number between 0 and 59, both inclusive, representing the minutes component of the date-time argument. |
| seconds | Returns a number between 0 and 59, both inclusive, representing the seconds component of the date-time argument. |
| timezone | Returns an interval value, representing the time offset from UTC. |

Date-Time Operators

| Date-Time Operator | Description |
|--------------------|--|
| year | Returns a number representing the year component of the date-time argument. |
| month | Returns a number representing the month component of the date-time argument. |
| day | Returns a number representing the day component of the date-time argument. |
| hours | Returns a number between 0 and 23, both inclusive, representing the hours component of the date-time argument. |
| minutes | Returns a number between 0 and 59, both inclusive, representing the minutes component of the date-time argument. |
| seconds | Returns a number between 0 and 59, both inclusive, representing the seconds component of the date-time argument. |
| timezone | Returns an interval value, representing the time offset from UTC. |

| Date-Time Operator | Description |
|--------------------|--|
| toTimezone | <p>Returns the date-time expressed in the time offset corresponding to the timezone ID provided.</p> <p>You have the following options for specifying a timezone ID:</p> <ul style="list-style-type: none"> • Provide a fixed offset from UTC/Greenwich, such as "-07:00". For example: <ul style="list-style-type: none"> - <code>dateTimeDO.toTimezone("-07:00")</code> returns <code>2002-11-30T17:20:00-07:00</code> when <code>dateTimeDO</code> is <code>2002-12-01T01:20:00+01:00</code> • Specify a geographical region, which is an area where a specific set of rules for finding the offset from UTC/Greenwich apply. A geographical region is usually represented in the format "{area}/{city}", such as "Europe/Amsterdam" or "America/New_York". For example: <ul style="list-style-type: none"> - <code>dateTimeDO.toTimezone("America/Los_Angeles")</code> returns <code>2002-11-30T16:20:00-08:00</code> when <code>dateTimeDO</code> is <code>2002-12-01T01:20:00+01:00</code> - <code>dateTimeDO.toTimezone("America/Los_Angeles")</code> returns <code>2002-06-30T17:20:00-07:00</code> when <code>dateTimeDO</code> is <code>2002-07-01T01:20:00+01:00</code> <p>Note: You must include double quotation marks around the timezone ID value.</p> <p>For more information about timezone IDs, see Class ZoneId in the Java Platform documentation.</p> |

Boolean Operators

Here is a list of all Boolean operators available to use in simple expressions.

| Boolean Operator | Description | Simple Expression | Result |
|------------------|--|-----------------------------|--------------|
| == | Equals | <code>true == true</code> | <i>true</i> |
| != | Not Equals | <code>true != false</code> | <i>true</i> |
| and | Conditional - And | <code>true and false</code> | <i>false</i> |
| or | Conditional - Or | <code>true or false</code> | <i>true</i> |
| not | Logical complement operator; inverts the value of a Boolean expression | <code>not true</code> | <i>false</i> |

Duration Operators

Here is a list of all duration operators available to use in simple expressions.

| Duration Operator | Description |
|-------------------|--------------|
| == | Equals |
| != | Not Equals |
| > | Greater than |

| Duration Operator | Description |
|-------------------|------------------------|
| >= | Greater than or Equals |

Base64Binary Operators

Here is a list of all Base64Binary operators available to use in simple expressions.

| Base64Binary Operator | Description |
|-----------------------|-------------|
| == | Equals |
| != | Not Equals |

Array Operators

Here is a list of all array operators available to use in simple expressions.

| Array Operator | Description |
|----------------|--|
| [] | Access a particular element in the array. |
| == | Equals |
| != | Not Equals |
| length | Returns the number of elements contained within the array. |

Miscellaneous Operators

Here is a list of all other miscellaneous operators available to use in simple expressions.

| Miscellaneous Operator | Description |
|------------------------|-------------|
| == | Equals |
| != | Not Equals |

Special Constants

Here is a list of all special constants available to use in simple expressions.

| Special Constant | Description |
|------------------|----------------------|
| null | Null value |
| true | Logical true |
| false | Logical false |
| 'now' | The current dateTime |

Casting

Casting refers to bypassing of the type validation and assigning data types that aren't necessarily compatible.

For example, you may want to assign an integer value to a string. In such cases, you can use a conversion operation like the following one:

```
<conversionTypeName> ( <valueToConvert> )
```

where:

`conversionTypeName` is the data type you want to assign to a value.

Here are some examples of casting or conversion operations:

- `string(myIntDO)`
- `int(myStringDO)`
- `duration(myStringDO)`

Note:

You can only cast to primitive data types. Therefore, the `conversionTypeName` variable accepts only primitive types as valid values.

WARNING:

Assigning values that are incompatible results in runtime errors. Even though some preliminary checks are performed on conversions to prevent incompatible castings (for example, a conversion from 'Boolean' to 'date'), all conversions are generally considered unsafe because there's no extra processing performed when applying them.

Define Process Completion and Termination

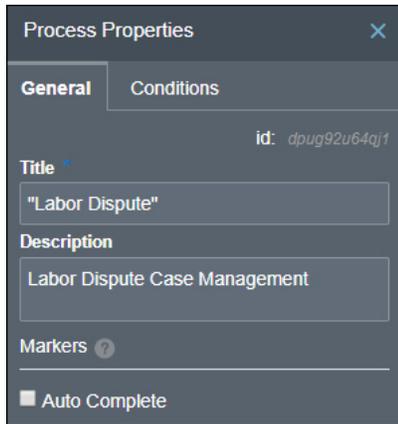
Edit process properties to specify how a process completes (manually or automatically), define a condition for its termination, or edit its input and output.

On the process page, click **Edit Properties**  to open the Process Properties pane. Use this pane to define or modify properties of the process. This pane consists of the following tabs:

- [General](#)
- [Conditions](#)
- [Interfaces](#)

Enable the Auto Complete Marker for a Process

In the **General** tab, you can enable the Auto Complete marker if you require the process to complete automatically under certain conditions. In addition, you can also edit the process title and description.



To enable the marker, select the marker's check box. When the marker is enabled, the Auto Complete decorator appears next to the process name.

When you enable the marker without conditions, it applies to the process by default. However, you can also enable the marker based on data conditions defined using process variables, decision models, or REST connectors. For steps to enable the marker based on data conditions, see [Enabling Markers for a Stage](#).

The following table details the Auto Complete marker for a process.

| Marker | Description |
|---|---|
| Auto Complete <input checked="" type="checkbox"/> | <ul style="list-style-type: none"> If the Auto Complete marker is enabled, a process is automatically marked <i>Complete</i> if none of the plan items (stages or activities) within it are in the <i>Active</i> state and all required plan items are <i>Completed</i>, <i>Terminated</i>, or <i>Disabled</i>. If this marker is disabled, a process is marked <i>Complete</i> only if all plan items are <i>Completed</i>, <i>Terminated</i>, or <i>Disabled</i>. <ul style="list-style-type: none"> When this marker is disabled, you can also manually complete a process if none of the plan items is in the <i>Active</i> state, and all required plan items are <i>Completed</i>, <i>Terminated</i>, or <i>Disabled</i>. |

Define a Condition for Process Termination

In the **Conditions** tab, you can specify a condition to terminate the process without its execution being complete.

You can use events on any stage or activity within the process as a part of your condition. For detailed steps to create a condition, see [Define Conditions for a Stage](#).

When a process is terminated, all *Active* stages and activities within the process are also terminated.

Edit Process Input or Output

In the **Interfaces** tab, you can edit or define process input and output. See [Define Process Input and Output](#).

About Process and Plan Item Lifecycles

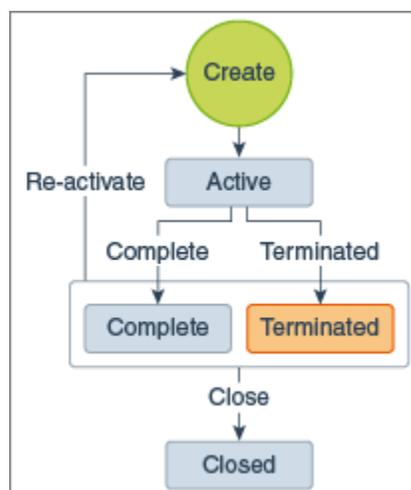
During execution, a dynamic process and its plan items (stages and activities) go through various states from creation to closure.

The following topics detail available states and their flow for a dynamic process and its plan items:

- [Process State Model](#)
- [Stage or Activity State Model](#)
- [State Transitions – An Illustration](#)

Process State Model

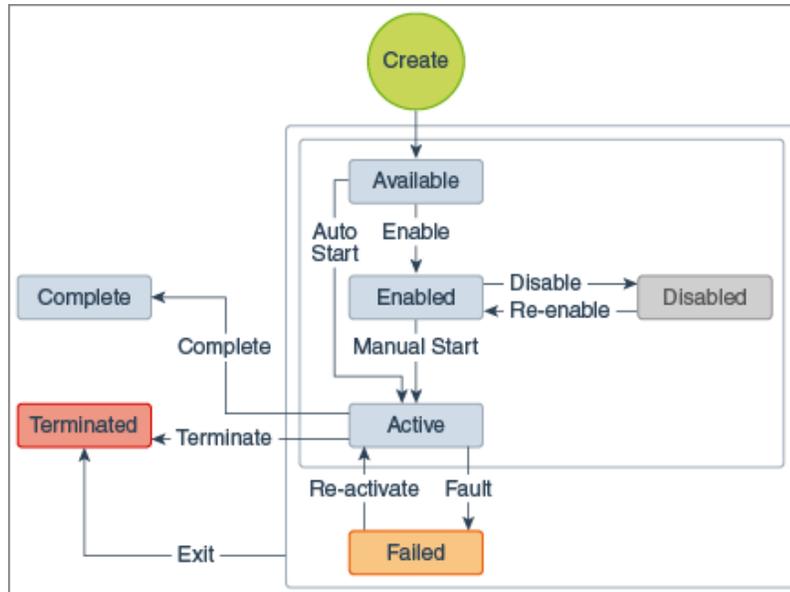
The following diagram shows the lifecycle of a dynamic process instance.



| State | Description |
|------------|--|
| Active | The process state when it is instantiated. When a process is in this state, all stages and global activities defined within it are instantiated and enter the state <i>Available</i> . |
| Complete | A process instance automatically enters this state when all stages and activities contained within it in are <i>Completed</i> , <i>Terminated</i> , or <i>Disabled</i> . You can also manually mark a process <i>Complete</i> or use the Auto Complete marker. See Enable the Auto Complete Marker for a Process . |
| Terminated | A process instance automatically enters this state when its termination condition is fulfilled. |
| Closed | You can manually close a process instance at any time. This removes the process instance from the runtime database. |

Stage or Activity State Model

The following diagram shows the lifecycle of a stage or activity instance.

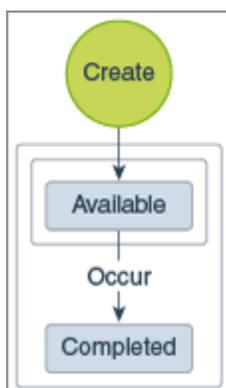


| State | Description |
|-----------|--|
| Available | A stage or global activity becomes <i>Available</i> when the process it is contained in is instantiated or becomes <i>Active</i> . However, an activity within a stage becomes <i>Available</i> when the stage containing it enters the <i>Active</i> state. |
| Enabled | A stage or activity enters this state only if it requires human intervention. If the Manually Activated marker is enabled, then the stage or activity transitions from <i>Available</i> to <i>Enabled</i> (after fulfilling enablement conditions if any). From here, a process participant can move the stage or activity into <i>Active</i> or <i>Disabled</i> states. |
| Disabled | A process participant can move a stage or an activity into the <i>Disabled</i> state to skip its execution in the current process instance. Similarly, a <i>Disabled</i> stage or activity can also be re-enabled. These actions are performed in runtime. |
| Active | When a stage becomes <i>Active</i> , all activities within it are instantiated and become <i>Available</i> . When an activity enters the <i>Active</i> state, the actual execution of that task begins. If the Manually Activated marker is enabled, then the stage or activity enters the <i>Enabled</i> state (after fulfilling enablement conditions if any) and waits for the process participant to activate it. If the Manually Activated marker is disabled, the stage or activity becomes <i>Active</i> as soon as it is instantiated or after fulfilling activation conditions if any. |
| Failed | This state indicates a failure to initiate, activate, or complete the stage or activity. |

| State | Description |
|------------|--|
| Complete | <p>This state indicates normal completion of stage or activity.</p> <p>An activity enters this state when its execution has finished.</p> <p>A stage is automatically marked <i>Complete</i> when <i>all</i> activities contained within it in are <i>Completed</i>, <i>Terminated</i>, or <i>Disabled</i>.</p> <p>You can also manually mark a stage <i>Complete</i> or use the Auto Complete marker. See Enabling Markers for a Stage.</p> <p>A stage or activity in the <i>Complete</i> state is removed from the runtime database.</p> |
| Terminated | <p>A stage or activity enters this state when its termination condition is met or its parent (process or stage) is terminated.</p> <p>A <i>Terminated</i> stage or activity is removed from the runtime database.</p> |

Milestone State Model

The following diagram shows the lifecycle of a milestone activity.

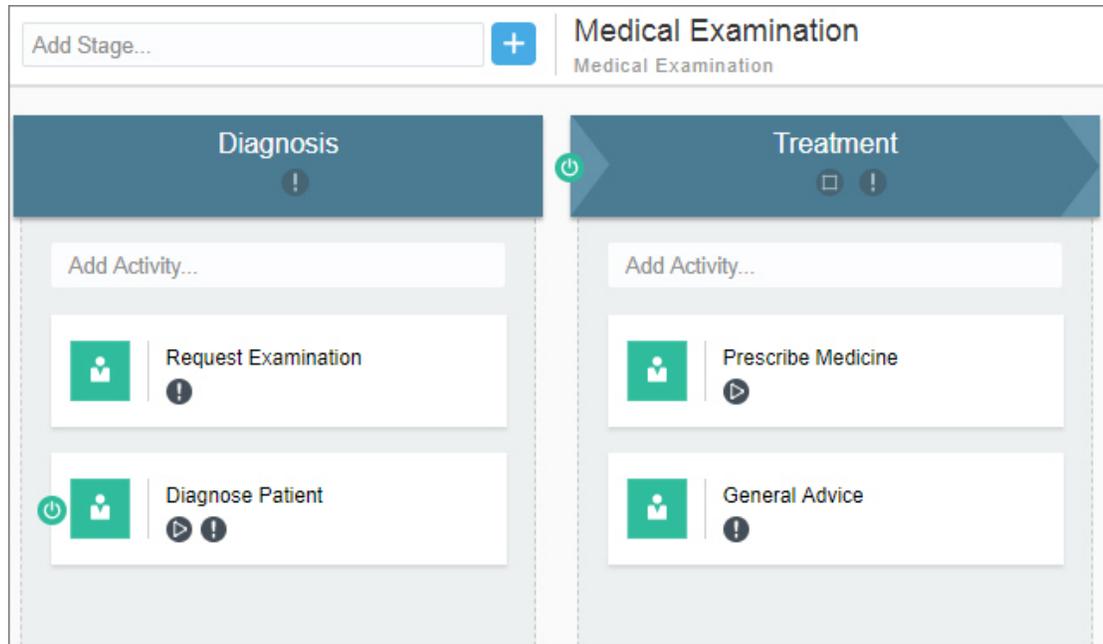


| State | Description |
|-----------|--|
| Available | <p>A milestone becomes <i>Available</i> when the stage it is contained in becomes <i>Active</i>.</p> <p>A global milestone becomes <i>Available</i> when the process it is contained in is instantiated.</p> |
| Completed | <p>A milestone enters this state when its completion condition is met.</p> |

State Transitions – An Illustration

Let's use a simple example in order to understand state transitions of processes, stages, and activities.

Consider a medical exam process with two stages, Diagnosis and Treatment. Both these stages contain two activities each as shown in the following figure:



The following table provides additional details of the plan items:

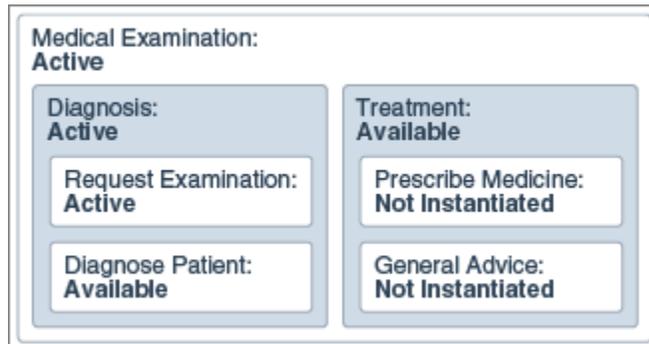
| Plan Item | Markers and Conditions |
|------------------------------|---|
| Diagnosis stage | The Required marker is enabled. |
| Treatment stage | The Required and Auto Complete markers are enabled. In addition, an activation condition is also defined specifying that this stage is activated only after the previous stage is <i>Complete</i> . |
| Request Examination activity | The Required marker is enabled. |
| Diagnose Patient activity | The Required and Manual Activation markers are enabled. In addition, an enablement condition is also defined specifying that this activity is enabled only after the Request Examination activity is <i>Complete</i> . |
| Prescribe Medicine activity | The Manual Activation marker is enabled but no enablement condition is specified. Note: The Required marker is disabled. |
| General Advice activity | The Required marker is enabled. |

When a process participant starts the application and instantiates the process, the following sequence of state transitions take place:

1. The process enters the *Active* state and both stages enter the state *Available*.
2. Because the Diagnosis stage is not manually activated and has no activation conditions, it immediately turns *Active*. However, the Treatment stage remains in the *Available* state until its activation condition is fulfilled. The activities within these stages undergo following transitions:
 - a. The Request Examination activity becomes *Available*, and it subsequently turns *Active* because there are no conditions for its activation.

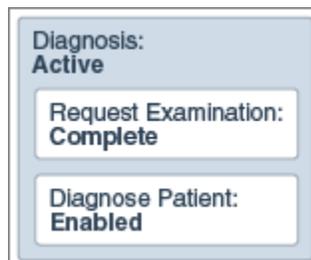
- b. The Diagnose Patient activity remains in the *Available* state until its enablement condition is fulfilled.
- c. Both the activities under the Treatment stage are not instantiated at this point in the process.

Note that all the above transitions occur simultaneously as soon as the process is activated. The following diagram depicts states of the process instance and its plan items at this point in execution:



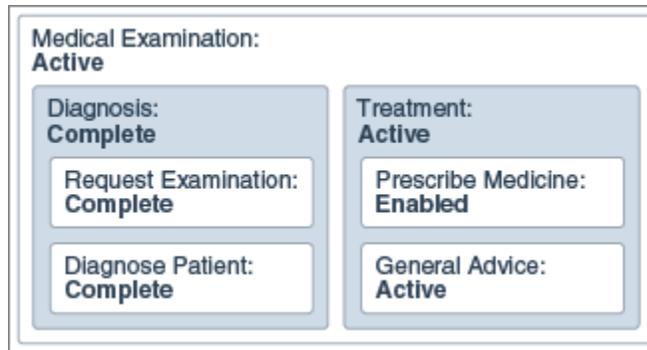
- 3. Now, the patient provides the required information and completes the Request Examination activity.
- 4. This fulfills the enablement condition of the Diagnose Patient activity, and it transitions to the *Enabled* state.

The following diagram depicts the current state of the Diagnosis stage:



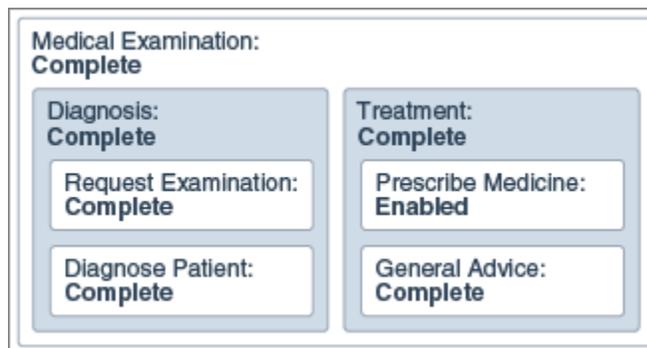
- 5. In the *Enabled* state, the Diagnose Patient activity awaits a relevant process participant to manually activate it. In this case, a medical practitioner starts this activity. The practitioner makes a diagnosis, records observations, and completes the activity.
- 6. With both activities within it being *Complete*, the Diagnosis stage also enters the *Complete* state.
- 7. This fulfills the activation condition for the Treatment stage and it becomes *Active*.
 - a. Subsequently, both activities under this stage become *Available*.
 - b. The Prescribe Medicine activity, in turn, enters the *Enabled* state. The medical practitioner starts this activity only if necessary.
 - c. The General Advice activity, however, becomes *Active* because there are no conditions for its activation.

The following diagram shows states of the process instance and its plan items at this point in execution:



8. Consider this treatment scenario where the practitioner decides not to start the Prescribe Medicine activity and completes only the General Advice activity.
9. Even though the Prescribe Medicine activity is still in the *Enabled* state, the Treatment stage is automatically marked *Complete* because of the following reasons:
 - a. No activity within the stage is *Active*. In addition, all activities which are marked *Required* have been completed. In this case, the General Advice activity is *Complete*.
 - b. The Auto Complete marker is enabled for this stage.
10. With both stages marked *Complete*, the process is marked *Complete* as well.

The following diagram shows final states of the process instance and its plan items:



Work with Inline Validations

Use inline validation to view and fix errors inline as you add and configure elements for your dynamic process. You can see validation issues in the stage, activity and process level. A validation icon in the stage, activity or process level indicates there are validation issues for that level.

- Click the validation icon to see the type of validation issues.
- Red circles  indicate errors. Yellow triangles  indicate warnings.
- If only errors are present, or if both errors and warnings are present, then an error icon displays. A warning icon displays if there are only warnings.
- Click **Fix** next to the error or warning to open the related properties panel where you can fix the issue.
- The validation icon's number indicates the number of issues found.

- Use the configuration panel to hide or display validation issues. You can access the configuration panel from the **Settings**  icon.

Work with Dynamic Processes in Runtime

In Process, you can design and develop ad-hoc processes called dynamic processes. Dynamic processing focuses on unpredictable business processes which rely on worker knowledge and involve human participants. Based on the content and information of the process, knowledge workers make informed business decisions.

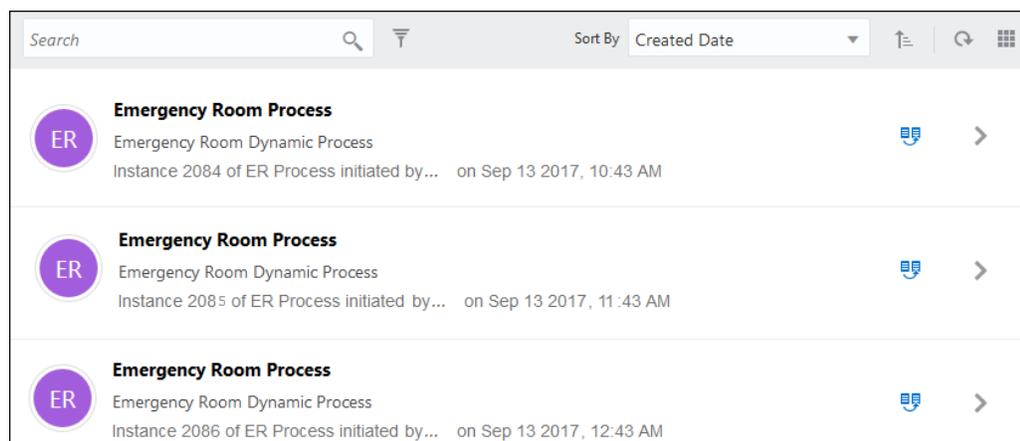
During runtime, knowledge workers decide the course of activities in a dynamic process, start and complete activities, assign activities to roles, and complete and close process instances. As a knowledge worker, you'll work on the following activities:

- Human tasks
- Structured processes

Milestones are sub-goals defined within a process and are used to track progress. You'll complete milestones created during design time as you work on the above mentioned activities. You can track the progress of your process instance in the top bar of the Process Instance Details page.

The Dynamic Processes page displays the different dynamic process in progress in an application. You can view the dynamic process for which you have the necessary permissions to work on based on your role.

To access the Dynamic Processes page, in the Oracle Integration navigation pane, click **My Tasks**, click **Workspace**, and then **Dynamic Processes**. A list of active dynamic processes is displayed.



| Search | Sort By | Created Date | | |
|---|---|--------------------------------|---|---|
|  | Emergency Room Process | Emergency Room Dynamic Process |  |  |
| | Instance 2084 of ER Process initiated by... | on Sep 13 2017, 10:43 AM | | |
|  | Emergency Room Process | Emergency Room Dynamic Process |  |  |
| | Instance 2085 of ER Process initiated by... | on Sep 13 2017, 11:43 AM | | |
|  | Emergency Room Process | Emergency Room Dynamic Process |  |  |
| | Instance 2086 of ER Process initiated by... | on Sep 13 2017, 12:43 AM | | |



Note:

A dynamic process will be active only after the application process instance is started. See [Start an Application](#).

Use this page to perform the following tasks:

| Option | Icon/Field | Description |
|--------------------------|--|--|
| Search |  | Search for an active instance by name |
| Filter |  | Filter by: <ul style="list-style-type: none"> • State: Active, Completed, Terminated, Closed • Process name • Created before or after a certain date • Created by • Closed before or after a certain date |
| Sort |  | Sort by: <ul style="list-style-type: none"> • Instance Id • Created Date • Process • Closed Date • Duration |
| Ascending/ Descending |  | Display the instances list in ascending/ descending order of their names. |
| Grid/List view |  | View dynamic process in either Grid or List format. Click Grid View to alternate between views. |
| Details |  | View the process instance details page. |

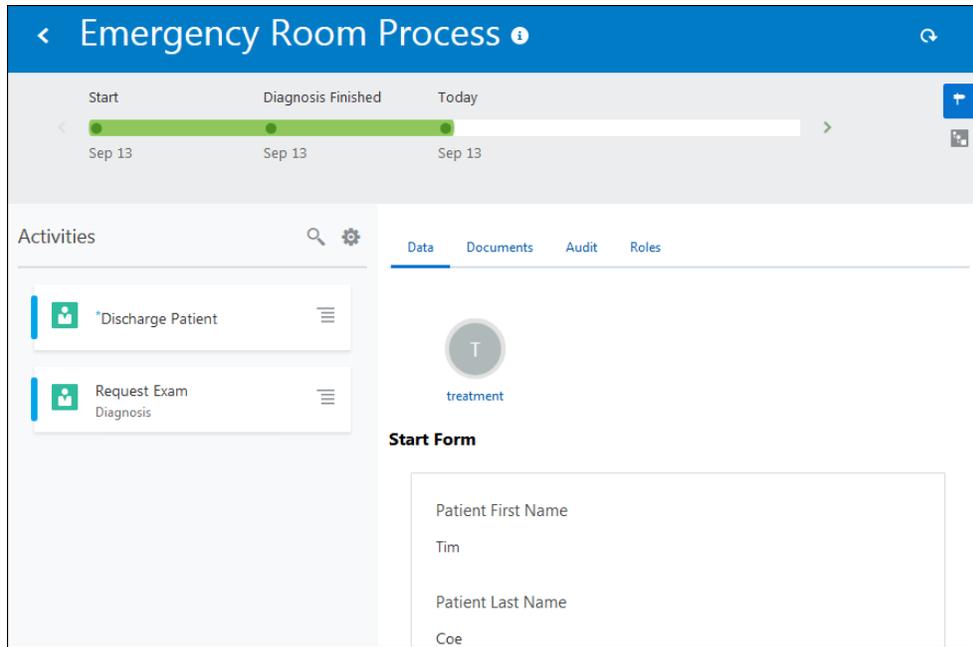
What You Can Do on the Process Instance Details Page

Use the Process Instance Details page to view activities, start, or force complete an activity. You can force complete an activity only if you are the process owner.

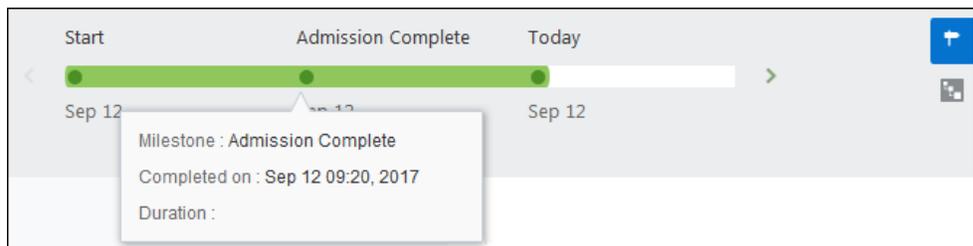
To open the Process Instance Details page:

1. Click **Details**  next to an instance.

The Process Instance Details page displays. The top bar displays the status of the instance. It displays two views, milestones or stages.



2. Click the icons on the right side to alternate between the different status views. In the milestones view, click on a milestone to view its details.



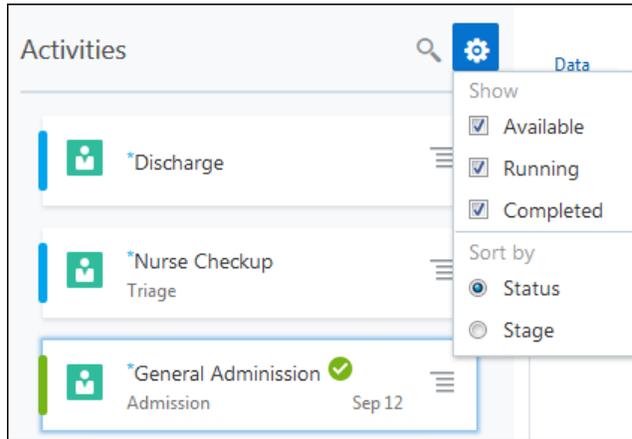
In the stages view, click on a stage to view the number of available, active, and completed activities.



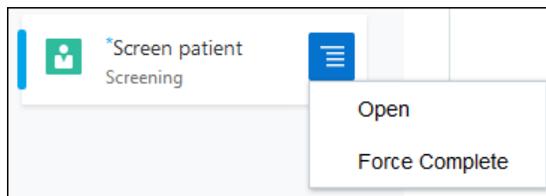
3. The different activities in the process on which you can perform an action are displayed in the left pane. You can search for an activity by entering its name in the **Search** field. You can also choose which activities to display in the left pane.

 **Note:**

The Activities pane lists only the actionable activities of a process instance. To view the status of non-actionable activities, such as services tasks or milestones, click the **Audit** tab next to the Activities pane.



4. Right-click an activity to open it in the right pane or force complete an activity. If you are the process owner, you can force complete an activity. Note that force completing an activity does not complete the underlying structured process or human task, it only withdraws that activity from the dynamic process.



5. Click **Filter/Sort**  to choose the activities to be displayed or sort the activities by stage or status.

The right pane displays the Data tab by default which contains the form data associated with the process instance in a read-only format. The following tabs are displayed at the top in the right pane:

| Tab | Description |
|-----------|---|
| Data | View data related to the dynamic process instance. |
| Documents | View documents related to a process instance. |
| Audit | The audit trail related to the dynamic process instance. It displays the details of different activities related to the process instance. Click the Open icon next to an activity to view the audit details. |
| Roles | Add permissions and membership to this process instance. Click Actions next to a role to edit its permissions and membership. |

The subsequent sections describe how to use these tabs during runtime while working with different activities in the process.

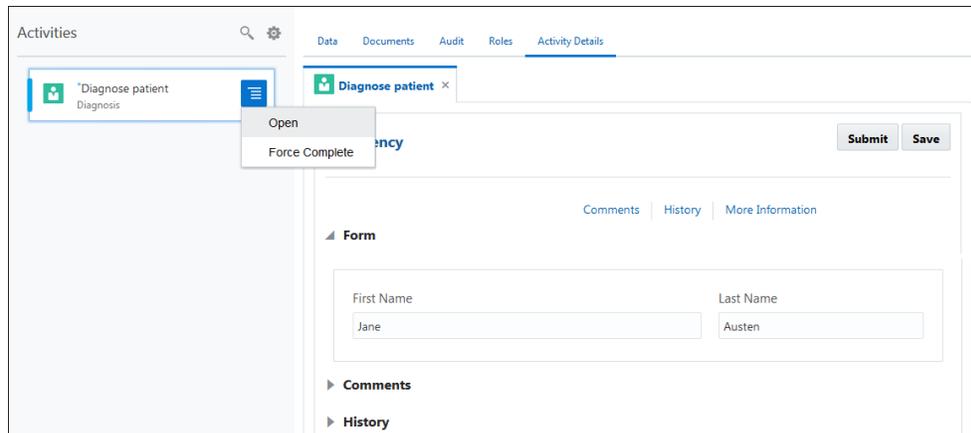
Work with Activities

Activities represent the tasks you execute as part of a dynamic process. As a knowledge worker, you'll work on human tasks and process activities.

To work on an activity:

1. Click the activity name, or click the **Actions** icon next to the activity and select **Open**.

The form related to the activity opens in the right pane in the Activity Details tab.



2. Edit the form as required.
3. Click **Submit** to submit the form. Or click **Save** to save and submit it later.

Note:

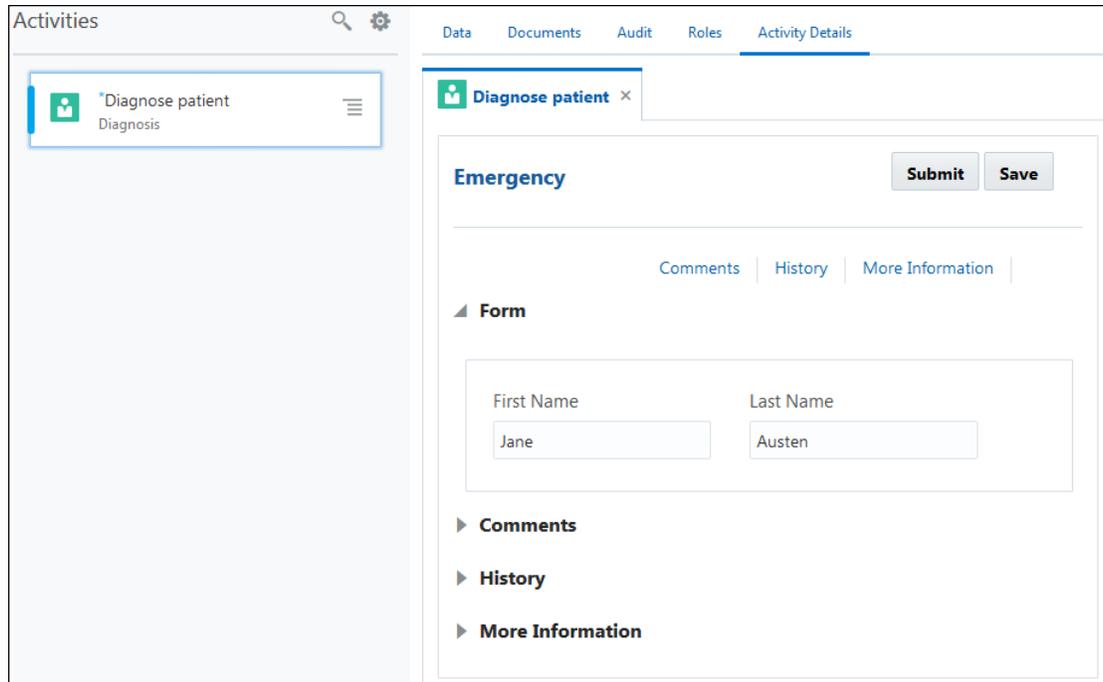
In the current release, a structured process within a process activity can only be designed with a message start event.

You can include integrations within your dynamic process or add rules and decision models to it using the Process activity. When a process activity is assigned to you, it will appear in the list of activities in progress within the dynamic process instance.

View Activity Details

You can view the details of an in-progress activity in the right pane.

To view the activity details, click the activity, or click **Actions**  next to the activity and select **Open**.



View Form Data

You can view the form data for the process instance in the Data tab.

The Data tab is displayed by default when you click **Details** next to a process instance. A read-only version of the form submitted for the process instance is displayed in this tab. This allows you to quickly access any information related to the process instance as you work on the different process activities.

The screenshot shows a web interface for a dynamic process instance. At the top, there are four tabs: 'Data', 'Documents', 'Audit', and 'Roles'. The 'Data' tab is selected and highlighted with a blue underline. Below the tabs is a circular profile picture containing the letter 'T', with the label 'treatment' underneath it. Below the profile picture is the heading 'Start Form'. The form itself is a light gray box containing the following fields and questions:

- Patient First Name: Tim
- Patient Last Name: Coe
- Symptoms: (empty text area)
- Do you feel any trauma?
- Do you agree to an eventual surgery?

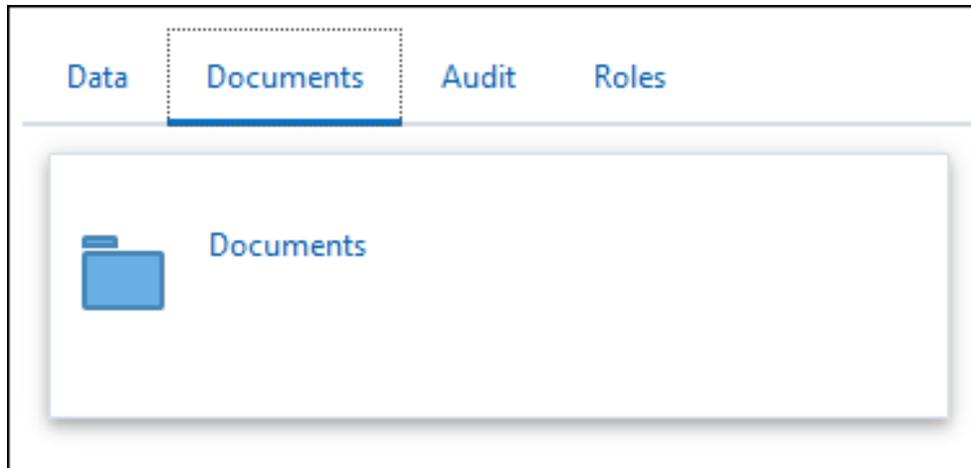
Upload Process Instance Documents

If Oracle Content Management is integrated with Oracle Integration, you can view existing documents and upload additional documents related to a dynamic process instance during runtime. Note that file attachments are not supported with dynamic processes.

To view or upload documents:

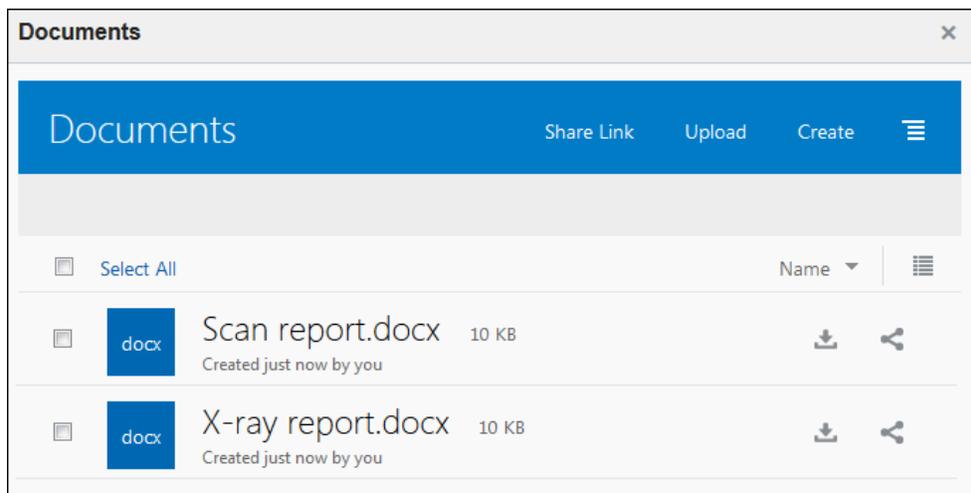
1. Click the **Documents** tab.

A folder containing documents related to the dynamic process instance is displayed.



2. Click the folder.

A window with a list of documents related to the process instance is displayed.



3. Click a document to view it, or use the **Upload** option to upload additional documents related to the process instance.

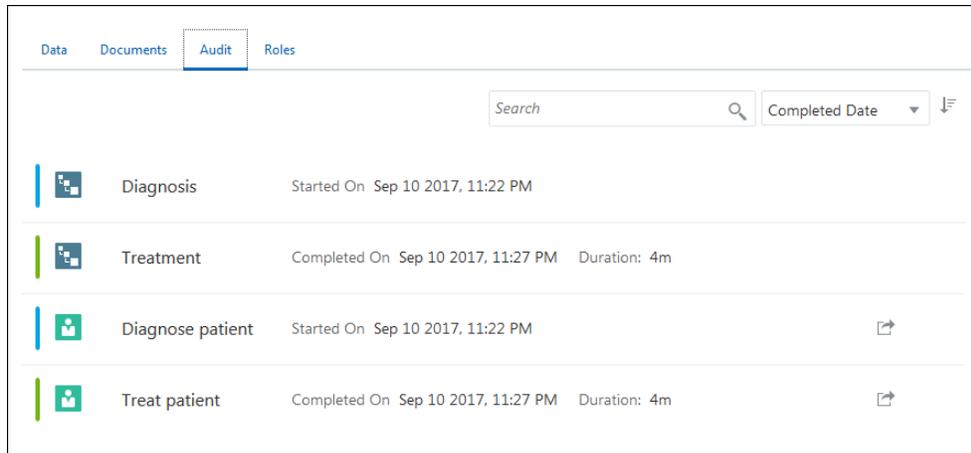
View Process Instance Log

You can view the audit details of the process instance during runtime.

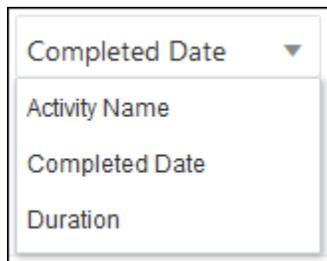
To view the audit details:

1. Click the **Audit** tab.

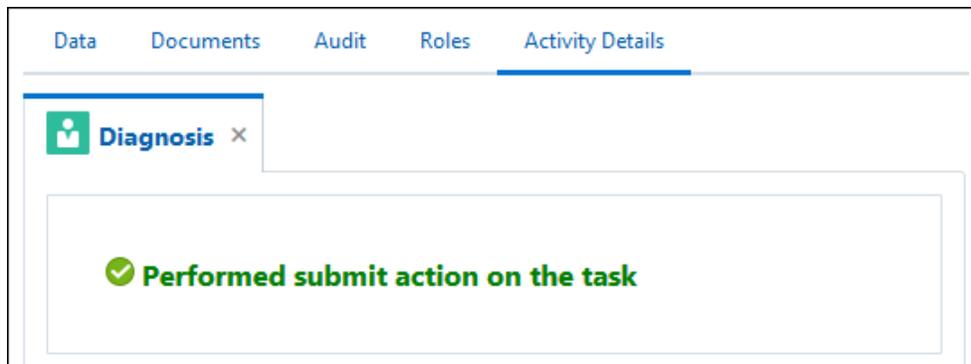
A list of activities and stages is displayed.



2. Search for an activity by entering its name in the search field . You can also sort the list of activities by activity name, completed date, or duration.



3. Click **Open**  next to an activity to view its history.



Override Roles Assigned to a Process

For dynamic processes, the process owner assigns roles to activities and adds members to these roles in design time. However, as a knowledge worker, you can temporarily override these roles—for an instance of the process—during runtime to add or delete members, modify permissions, and so on. This may be particularly

useful when you require to reassign human tasks or process activities to additional members or assign additional permissions to existing members.



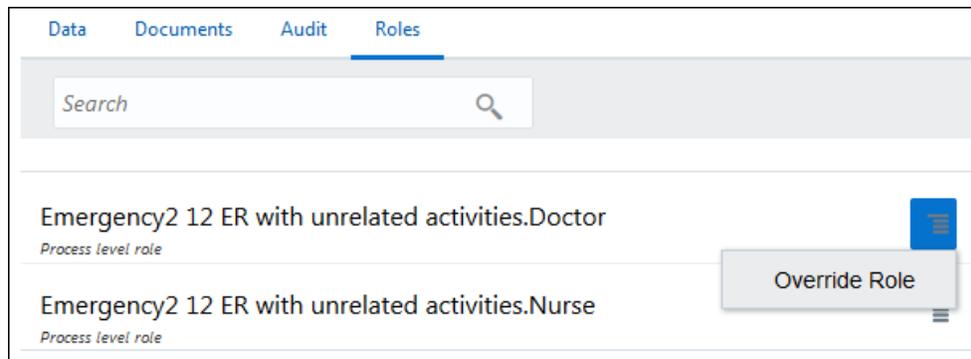
Note:

Unlike structured process applications, if you assign members to a role within a dynamic process instance in runtime, these assignments are retained only for that instance of the process. To retain member assignments for all instances of the process (and all deployments of the application), make these assignments in design time. See [Create Process Roles](#).

To temporarily override a role for an instance or deployment of the process application:

1. Click the **Roles** tab.

The roles currently defined for the dynamic process instance are displayed.



2. Click the **Actions** icon and select **Override Role**.

A list of members assigned to that role along with their permissions is displayed.

The screenshot shows the 'Roles' configuration page in Oracle APEX. The breadcrumb trail is 'Data > Documents > Audit > Roles'. The page title is 'Emergency2 12 ER with unrelated activities.Doctor'. There are three main sections: 'Name *', 'Members', and 'Permissions'. The 'Name' field contains 'Emergency2 12 ER with unrelated activities.Doctor'. The 'Members' field contains 'Doctor2'. The 'Permissions' section has a heading 'What can the members do on the instance?' and a note 'Note: Defaults to None if no action is selected'. There are six permission checkboxes, all of which are checked: 'Start new instance', 'Instance View', 'Instance Update', 'Document View Only', 'Document View and Download', and 'Document View , Upload and Download'. At the bottom of the permissions section is a button labeled 'Setup Advanced Permissions (1)'. At the bottom right of the form are 'Cancel' and 'Save' buttons.

3. Enter the name of a member in the **Members** field or select or deselect the check box next to permission for the dynamic process instance. Note that these permissions are applicable to all activities defined within a process instance.

The permissions defined are:

- Start new instance
 - Instance View
 - Instance Update
 - Document View only
 - Document View and Download
 - Document View, Upload, and Download
4. Optionally, click **Setup Advanced Permissions (0)** to set up additional permissions specific to individual activities, documents or data objects.

The screenshot shows a web interface with a top navigation bar containing 'Data', 'Documents', 'Audit', and 'Roles'. The 'Roles' tab is active. Below the navigation bar, there is a breadcrumb trail: '< Emergency2 12 ER with unrelated activities.Doctor'. A note below the breadcrumb reads: 'Note: Task assignment will automatically override role permissions. Task assignees will be able to approve or add comments in spite of their role permissions (limiting update action)'. To the right of the note is a '+ Add Permission' button. Below this is a section titled 'Permission' with a trash icon. The main content area asks 'What would you like to set this permission on?' and features a dropdown menu with 'Activities' selected. Below the dropdown is a search box containing 'Diagnosis' with a close button (X). Underneath, it asks 'What can the members do with this resource?' and includes a note: 'Note: Defaults to None if no action is selected'. There are two checked checkboxes: 'View' and 'Update'. At the bottom right, there are 'Revert' and 'Done' buttons.

5. Click **Add Permission**. Note that this will override any instance permissions you've set.

 **Note:**

Task assignment automatically overrides role permissions. Task assignees can approve or add comments in spite of their role permissions limiting update action.

You can set permissions on the following resources:

- Activities
- Documents
- Data

The permissions vary according to the resource selected.

| Resource | Permissions |
|------------|---|
| Documents | <ul style="list-style-type: none"> • View • View and Download • View, Download, and Upload |
| Data | <ul style="list-style-type: none"> • View • Update |
| Activities | <ul style="list-style-type: none"> • View • Update |

! Important:

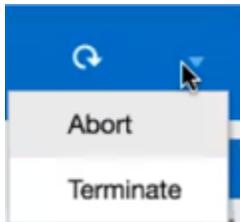
If no permissions are selected for a resource, then the role won't have any actions allowed on that resource.

End Dynamic Processes and Activities

Occasionally, you may need to end a process or activity. Follow these guidelines to choose the best option for your activity or process.

Abort a dynamic process

1. In the top corner, click the option icon and choose **Abort**.



A message asking you to confirm the action appears, with the following results:

Aborting this process will:

- Terminate this dynamic process instances.
- Abort any linked process instances.
- Withdraw any activity tasks triggered from this process.

2. Click **OK** to confirm.

Terminate a dynamic process

1. In the top corner, click the option icon and choose **Terminate**.

A message asking you to confirm the action appears, with the following results:

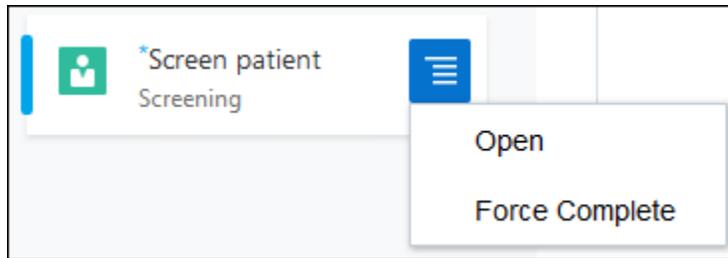
Terminating this process will:

- Terminate this dynamic process instances.
- Cancel any linked process instances.
- Withdraw any activity tasks triggered from this process.

2. Click **OK** to confirm.

Force complete a dynamic process activity

1. Right-click an activity and choose **Force Complete**.



A message asking you to confirm the action appears. It notes that by default, force completing doesn't complete any associated processes, and lets you select a field to complete any associated processes.

2. Click **OK** to confirm.

8

Create and Use Micro Processes

You can divide a large, complex business process into multiple reusable blocks called micro processes, created within separate applications. You can then efficiently manage the overall implementation of your business case by calling these micro processes in a main or parent process application.

Using micro processes, you can develop an application that comprises multiple independent, domain-driven processes. Micro processes are typically smaller processes with quick execution time, with their output feeding into the main process. You can invoke them either synchronously or asynchronously. Within an application, create a link to a deployed micro process present in another application, and use this link in your main process.

As an example, consider an employee travel approval scenario that consists of the following stages: information collection, request approval, booking, and mail delivery. You can implement each of these stages as a micro process (in separate applications), and you can use a main process to orchestrate the execution of all stages.

Topics:

- [Create a Micro Process Link](#)
- [Add Micro Processes to Your Main Workflow](#)

Create a Micro Process Link

If you have a reusable process created and deployed in another application, you can create a link to it in your current application and use it within the main process.

To create a link to a deployed micro process:

1. On the **Application Home** tab, click **Processes**.
2. Click **Create** and select **Use Micro Process**.
3. In the Link Micro Process dialog, select a deployed process, and click **Link**.
4. Complete the properties settings and click **Create**.
 - a. **Operation:** Select an operation (for example, the start event of a micro process) to invoke from the drop-down list.
 - b. **Callback Operation:** In case of asynchronous links, specify a callback operation. For example, specify an element in your current process that'd receive the data from the micro process.
 - c. **Type:** Specify if the micro process link is a synchronous or asynchronous invoke. A synchronous invoke halts the main process until the execution of the micro process element completes. An asynchronous invoke enables parallel execution of both micro process and main process.
 - d. **Security Type:** Select **APP Id - Basic Authentication** to apply basic authentication to the micro process link.

- e. **Keystore Credential:** Complete the fields that display. To create a new key, select **New Key** and enter a key name, user name, and password.

The screenshot shows a 'Link Micro Process' dialog box with the following fields and values:

- Name: SyncProcess1
- Operation: start
- Callback Operation: (empty)
- Type: Synchronous (selected)
- Security Type: APP Id - Basic Authentication
- Keystore Credential: [New Key]
- Key Name: AccessKey1
- Username: WFaulkner
- Password: (masked with dots)

Buttons: Back, Create

A micro process link is now created. You can use this link in any process within the application after performing the required configurations. See [Add Micro Processes to Your Main Workflow](#).

To view and update the security information for a micro process link, click its **Update Security** link on the Processes page. Click the delete icon to remove the process link from your application.

Add Micro Processes to Your Main Workflow

Learn how to add micro processes to the workflow of your main process.

Add a micro process to your structured process workflow

1. On the Process Editor's elements palette, expand the **System** elements.
2. Drag and drop a **Micro Process** element into your process workflow.
3. Implement the micro process element:
 - a. Click the element and select **Open Properties, General**.
 - b. Select a micro process link from the drop-down list. All the links that you've created within the application are listed here.
4. Click **Open Data Association** and configure input and output data mappings for the micro process task.
5. Click **Save**.

 **Note:**

You can also use a process for which you have not created a link previously. Click **Use Micro Process**  next to the **Micro Process** field to display a list of deployed processes on your server. Select a process from the list, and click **Link**. Complete the properties settings and click **Create**. See [Create a Micro Process Link](#). When you add a micro process in this way, a corresponding link is automatically created on the **Processes** tab of the Application Home page.

Add a micro process to your dynamic process canvas

To add a micro process activity in a dynamic process, see [Create Activities](#).

To edit the activity's implementation details, see [Specify Implementation Details for a Micro Process Activity](#).

Finally, perform the necessary data associations for the micro process activity.

 **Note:**

- You can call only message-based processes (Message Start and Message End) using the micro process element or activity. Additionally, you'll need to ensure that the start and end events of the micro processes called are correctly configured, so that the data flow between processes is seamless.
- In dynamic processes, you can call only asynchronous micro process links.
- You can edit the authentication and other settings for a micro process link while activating the main process.

9

Create Web Forms

In Process, you create web forms to interact with end users. As part of creating a web form, add its controls, configure its data, and define form behavior.

Topics:

- [Work in the Web Forms Editor](#)
- [Ready to create a web form?](#)
- [Work with Presentations](#)
- [Position Controls on Forms](#)
- [Bind Form Data with Controls](#)
- [Work with Stylesheets and Styling](#)
- [Configure Basic Controls](#)
- [Configure Advanced Controls](#)
- [Specify Filters for Controls](#)
- [Create Computed Controls](#)
- [Localize Web Forms](#)
- [Preview Forms and Their Payload](#)
- [Add Dynamic Behavior to a Form](#)
- [Reuse Forms](#)
- [Save Web Form Data](#)
- [Create an External UI Connection](#)

Work in the Web Forms Editor

Use the web forms editor to create forms for your human tasks and start form events without scripting rules.

The web forms editor is available in two modes - simple and full.

Create a simple form in two minute using the simple editor. See [Use the Simple Editor](#).

If you want to avail all the features such as multiple presentations, stylesheets, dynamic behavior through Events and REST connectors, then use the full editor. Get started by creating and configuring a form in the full editor, and then try it out in runtime. See [Ready to create a web form?](#)

Want to learn more about configuring a specific control? See [Basic Controls](#) or [Advanced Controls](#).

Accessing the Web Forms Editor

Creating or opening a web form displays the editor. Add, edit, open, or delete web forms using either of these methods:

- **On the Forms page**

On the Application Home tab, select **Forms** in the Components pane. The Forms page displays all forms defined for the application.

Add, edit, or delete a web form on this page. Note that to activate a web form, you must associate it with a flow element, by selecting it in a [human task](#) or [form start event](#) in a process.

- **From the process editor**

On the Application Home tab, select **Processes** in the Components pane and open a process. Select a human task or start form event and open its properties. Add, edit, select, or delete a form on the human task's implementation pane. Adding or selecting a web form in this way associates it with the selected human task or start form event.

Using the Web Forms Editor

In the editor, drag, drop, and [position controls](#) onto the form's central canvas. Select from a wide variety of controls on the [Basic](#) and [Advanced](#) Palettes, ranging from standard text and drop-down fields to tables, list of value (LOV) fields, sections, panels, and special fields such as URLs, videos, and images. Alternatively, [automatically create a form from a business type](#) defined for the application.

After adding some controls, further develop your form by [creating multiple views \(presentations\)](#), [reusing other forms in the form](#), and customizing its styling by [uploading a CSS stylesheet](#) and [applying styling properties](#).

Working with Web Form Data

An integral part of implementing a web form is defining and using its data, both within the form itself and in the human task or start form event.

- Within a web form, you bind form controls to data attributes that hold input values entered by end users. You can [bind data to controls](#) automatically or manually. You can also [automatically create web forms based on business types](#) defined for the application.
- Within a process, you will likely make use of web form data through [data association](#), which defines input and output for flow elements that need them. Process creates data associations for you when you create human tasks with a web form or form start events. See [Configure Data Association](#).

Building Dynamic Forms Using Events and REST Connections

Incorporate dynamic behavior in web forms without scripting by [configuring events on controls](#) that upon firing, trigger conditions, actions, or REST connector calls. Populate controls [from REST or data connections](#).

Associating Forms with Flow Elements

If you've created a form but haven't associated it with a process, select **Processes** in the Components pane of the Application Home tab and open a process. Select a

human task or start form event and open its properties. In the implementation pane, click **Browse** and select a form from those defined for the application. Note that you can select a web form or a basic form. You can use either or both types of forms in processes. If associating a web form with a flow element, also select a [presentation](#) to display. (Basic forms don't use presentations.)

Use the Simple Editor

Create simple forms in minimum time by using the simple editor.

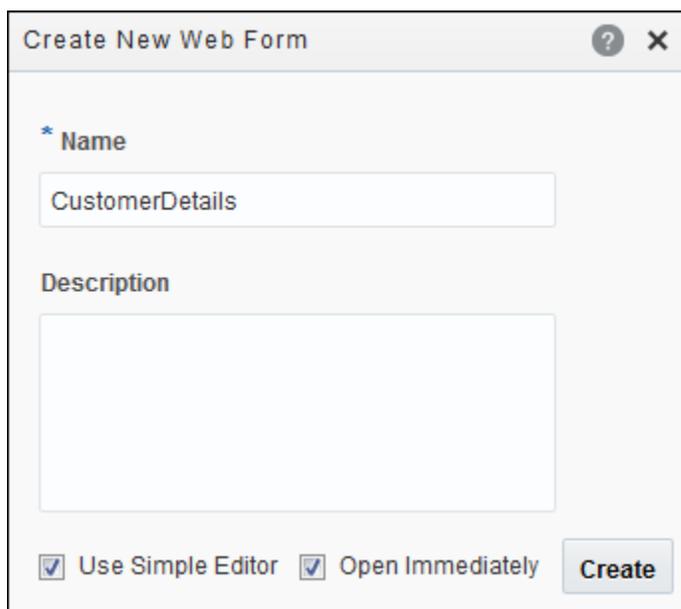
Explore how to use the simple editor in the following topics:

- [Create a Simple Form](#)
- [Controls in Simple Forms](#)
- [Preview Simple Forms](#)
- [Localize Simple Forms](#)

Create a Simple Form

Let's create a new web form using the simple editor.

1. Click **Forms** on the Application Home pane.
2. In the Forms page, click **Create a Form** to open the Create New Web Form dialog box.
3. In the Create New Web Form dialog box, enter the name of your form in the **Name** field. Optionally, give a description in the **Description** field.
4. Select **Use Simple Editor**, keep **Open Immediately** selected, and click **Create**.



The screenshot shows a dialog box titled "Create New Web Form". It has a standard window title bar with a question mark icon and a close button. The main content area includes a text input field labeled "* Name" containing the text "CustomerDetails". Below this is a larger text area labeled "Description". At the bottom of the dialog, there are two checkboxes: "Use Simple Editor" and "Open Immediately", both of which are checked. To the right of these checkboxes is a "Create" button.

5. In the Simple Editor, click **Add New Item**  to add a panel to your form. Panels help you to group items in the form.
6. Customize the panel.

- Change the label of the panel. For example, Customer Details.
- Enter a meaningful description.

You have just created a simple form and added a panel to it.

Controls in Simple Forms

There are different controls that you can use in your simple form. You can access the list of available controls by clicking **Change Type**  within a selected control.

| Control | Description |
|-----------------|--|
| Input Text | Users can enter a short, single line text entry. |
| Number | Users can enter a decimal number. Set the minimum and maximum values to enter with the Min and Max fields. |
| Text Area | Users can enter a longer, multi-line text entry. |
| Message | Users can enter a simple message. |
| URL | Users can enter a web address URL. |
| Email | Users can enter a valid email address. |
| Phone | Users can enter a phone number in International or US format. |
| Money | Users can enter money amounts. Set the minimum and maximum values to enter with the Min and Max fields. Use the Currency field to change the currency type. By default it is set to US Dollar (USD). |
| Date | Users can enter a date in the given format. Use the Format field to set the format of the date. Set the minimum and maximum date values that users can enter with the Min and Max fields. |
| Time | Users can enter a time. Use the Time Step field to set the value by which time increases or decreases when a user clicks the time icon. |
| Image | Include an image in your form by specifying a valid URL of the image in the Image URL field. |
| Video | Include a video in your form by specifying a valid URL of the video in the Source URL field. |
| Radio Buttons | Add mutually exclusive radio buttons to your form. Use the Edit Options to modify and specify settings for the radio buttons. |
| Checklist | Users can select one or more available options. Use the Edit Options to modify and specify settings for your check list. |
| Select | Add a drop-down list to your form. Use Edit Options to modify and specify settings for the options. Toggle Multiple to allow users to select more than one option. |
| IdentityBrowser | Users can search and select individuals to be notified or assigned tasks downstream in the process. Toggle Multiple to allow users to select more than one entry. |

Preview Simple Forms

When you have configured controls and set their specific configurations, you can preview your simple form. Click the **Preview** button to view your form, as it would

appear in runtime. Select Small, Medium, Large or Extra Large Devices to see your form displayed in the selected device.

Want to learn more? See [Preview Forms and Their Payload](#)

Localize Simple Forms

Click **Translate** to provide localized strings corresponding to controls in your form. In the Translate form window, select a language from the **Languages** drop-down field. All languages added to the application are present in the **Languages** drop-down field. Specify the strings for each control in the form and click **OK**.

Want to learn more? See [Localize Web Forms](#)

Ready to create a web form?

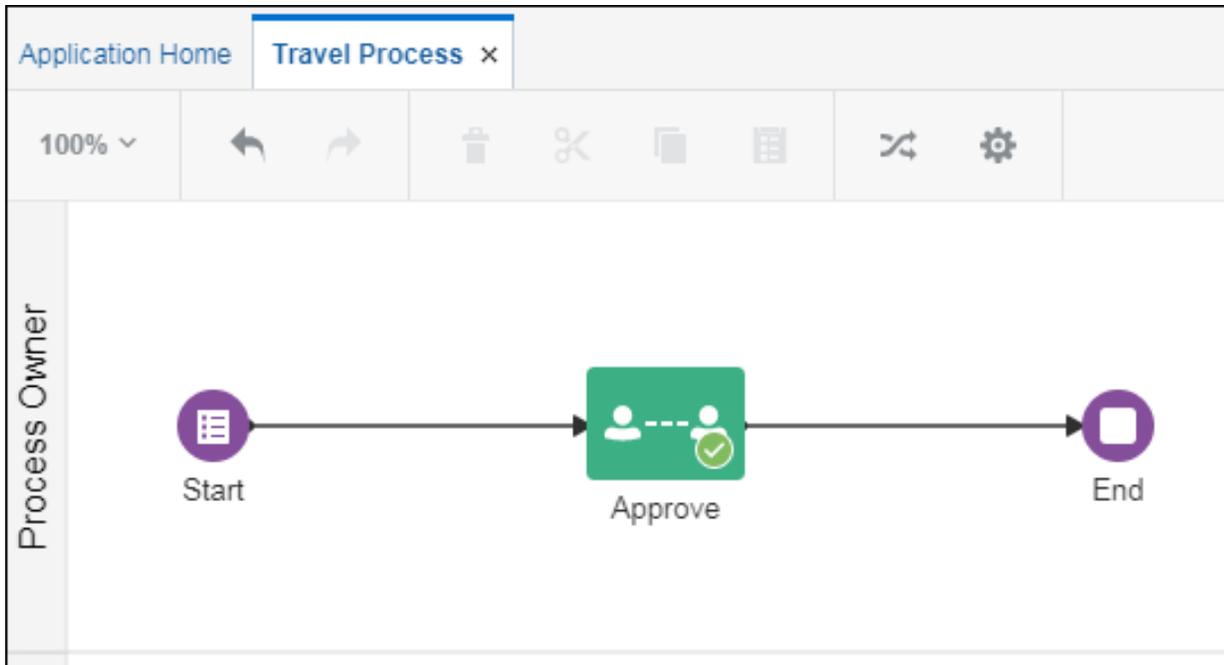
If so, we'll take you through the main steps—from creating a web form to testing it as an end user. You'll learn about forms and their controls, pick up web forms editor tips, and complete the entire web form development life cycle.

- [Create a Simple Application](#)
- [Create a Web Form](#)
- [Add and Configure Controls](#)
- [Add Another Presentation](#)
- [Change the Form's Stylesheet](#)
- [Preview the Form](#)
- [Use Forms and Presentations in a Process](#)
- [Define a Control's Behavior](#)
- [Try the Form in Runtime](#)
- [Explore Advanced Form Options](#)

Create a Simple Application

Web forms enable humans to interact with a business process. For example, a form can start an application or be used in a human task. Let's start by creating a simple application that contains those elements for use in a travel request form.

1. Go to the Home page, click **Processes**, click **Process Applications**, and then click **Create**.
2. Click **Create an Application**.
3. In the Create Application dialog box, enter **Travel**, and click **Create**.
4. In the Create a Process page, select **Start with a form**.
5. In the Create Process window, enter **Travel Process**, and click **Create**.
6. Add an Approve human task to the process and title it **Approve**.

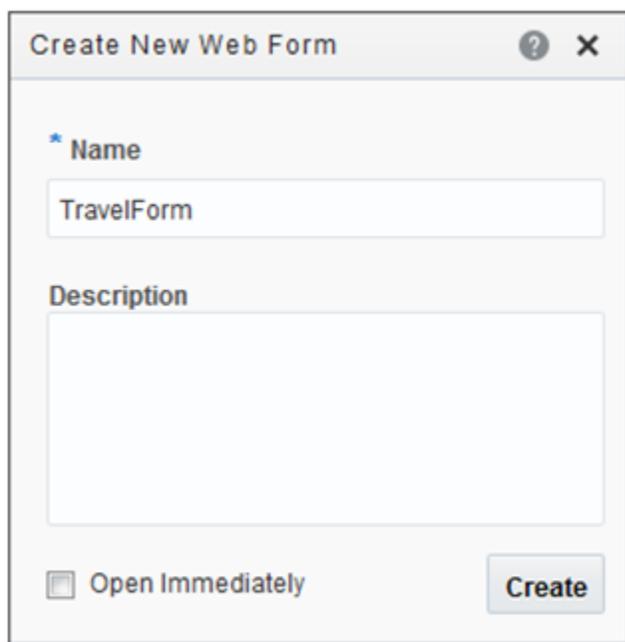


See [Create Your First Application \(a Quick Start\)](#).

Create a Web Form

Let's create a form for employees to complete with travel request details. This form should also display to users responsible for approving or rejecting the request.

1. In the process, open the Start Form's properties. Click the Start icon, and click its actions menu. (Notice how the **Open Form** command is dimmed, because no form is associated with the flow element yet.) Select **Open Properties**.
2. In the **Title** field, enter **Request Travel**.
3. Click **Create New Form +** by the **Form** field, and select **New Web Form**.
4. In the Create New Web Form dialog box, enter **TravelForm** in the **Name** field, and click **Create**.

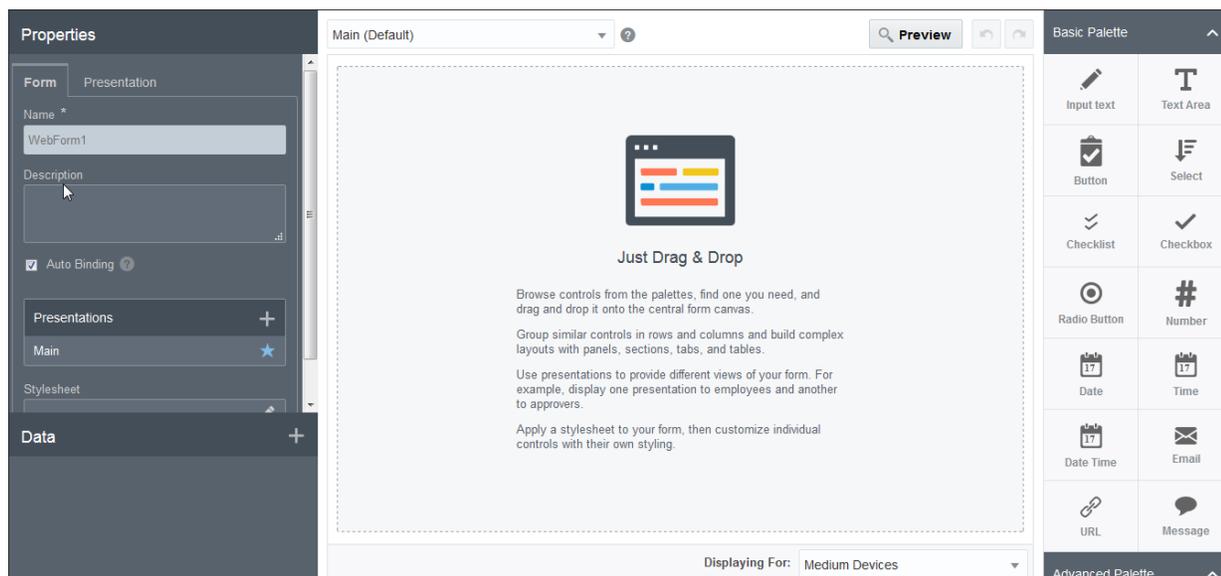


In the Properties pane, notice how the **Form** field identifies the new form.

5. Click the **Edit Form** icon.

The web forms editor opens in a new tab named with the title you entered for the form.

The editor contains a central canvas on which you construct the form by dragging and dropping available controls from the palettes onto the canvas. The **Properties** pane lets you select settings. The **Data** pane lists data attributes for the form's controls.



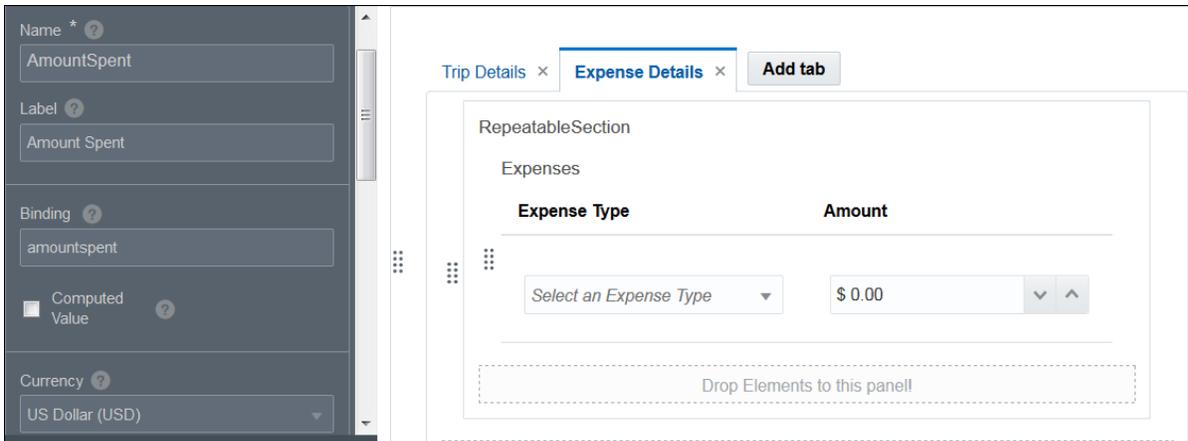
Add and Configure Controls

Let's add controls from the basic palette and the advanced palette to the travel request form.

1. Add and configure name fields onto the form's canvas in a row.
 - a. From the Basic Palette, drag and drop an Input Text control onto the canvas.
 - b. Drag and drop another Input Text control next to the first control.

As you drag, a box with a dotted outline shows where you can place the control, such as next to, between, or below another control. When you drop, controls are adjusted on an invisible grid to make space. (You can place up to 12 controls in a row across the form.)

To reposition a control, drag its dotted handle and drop it in a new location. The controls around it adjust accordingly. As you edit, click **Undo** and **Redo** as needed.
 - c. Select the first text control. Notice how the Properties tabs are **General** and **Styling**. The tab settings apply to the selected control.
 - d. On the General tab, change the **Name** field to `FirstName`, and the **Label** field to `First Name`. (The **Name** field applies to the control itself and the **Label** field determines its display name.)
 - e. Repeat steps c and d to change the second control's name to `LastName` and its label to `Last Name`.
2. Add and configure date fields.
 - a. Drag and drop two Date controls in a row below the name controls.
 - b. Select the first date control and change its name to `StartDate` and its label to `Start Date`. Scroll down on the **General** tab and select a date format in the **Format** field (for example, MM/dd/yy).
 - c. Select the second date control, change its name to `EndDate` and its label to `End Date`, and set its format in the **Format** field.
3. Add a travel justification text control.
 - a. Drag and drop a Text Area control below the date controls.
 - b. Select the control and change its name and label to `Justification`.
4. Add and configure a cost control.
 - a. From the Advanced Palette, drag and drop a Money control above the justification text control.
 - b. Select the control, change its name and label to `Cost`, and select your currency in the **Currency** field.
5. Add and configure an expenses control.
 - a. Position a Repeatable Section control below the justification text control.
 - b. Select the control and change its name to `Expenses`. Scroll down on the **General** tab and select **Users can Add/Remove Rows**.
6. Add and configure an expense details control.
 - a. Drag and drop a Table control on to the expenses section.
 - b. Select the control, change its name to `ExpenseDetails` and its label to `Expense Details`.
 - c. Scroll down on the **General** tab to the Columns section and click **Add**  to add another column. Rename **Column** to `Expense Type` and **Column 1** to `Amount`.

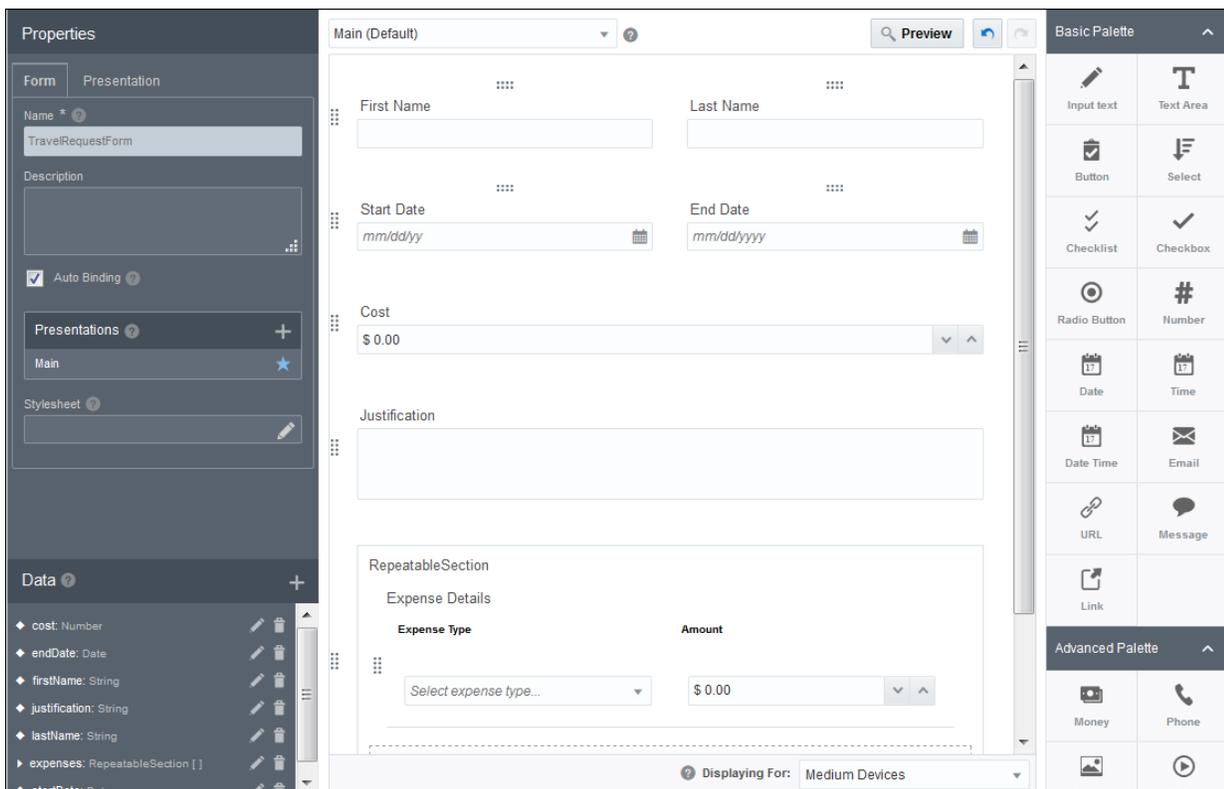


9. Switch between control and form properties.
 - a. Click a blank area of the form canvas.

Notice that the Properties tabs changed to **Form** and **Presentation**, and now apply to the entire form rather than the selected control. Click a control and the tabs change to **General** and **Styling**, and apply to the control.

Also notice that the Data pane lists data attributes with the same names as your controls (but different capitalization). These attributes were automatically created as you added each control. That happens when you have the **Auto Binding** field on the **Form** tab selected. These attributes hold the form's payload (working) data while the process is running.

- b. Click **Save** to save the form.



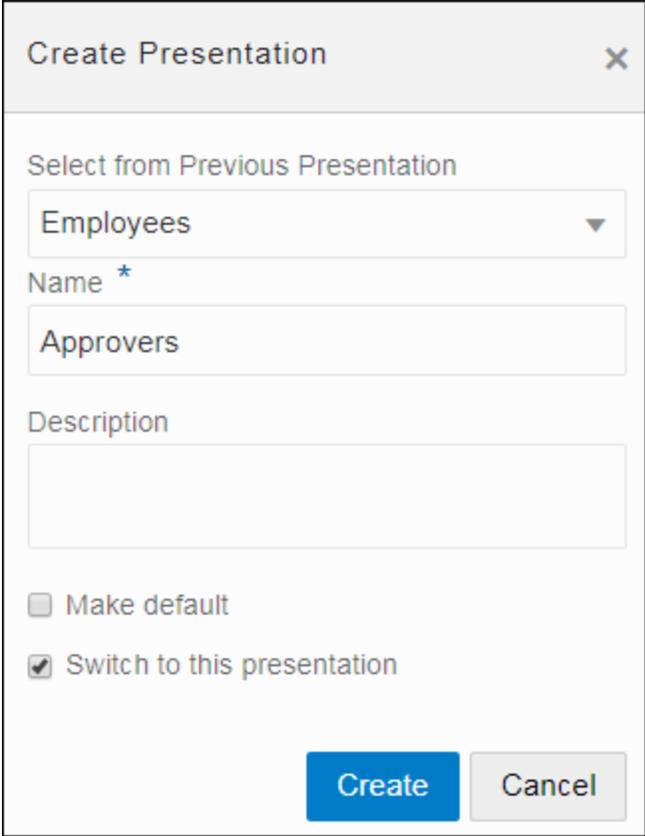
Add Another Presentation

Let's add an alternate view of the form that displays only to the users who review and approve travel requests.

1. Click the **Presentation** tab in the Properties pane. (If you see **General** and **Styling** tabs instead, select a blank area of the canvas first to display the **Presentation** tab.)
2. In the **Name** field, replace **Main** (the default name) with **Employees**. This presentation, which was already created, is the default presentation.

A presentation is a single view of a form. A form can have multiple presentations.

3. Click the **Form** tab. Notice that the Presentations table now lists our presentation as *Employees*. The star indicates that Employees is the default presentation.
4. In the Presentations table, click **Add**  to add a presentation. Instead of building your presentation from scratch, you can clone it from an existing presentation. In the Select Presentation Type dialog box, select **Clone**. In the Create Presentation dialog box, select **Employees** in the **Select from Previous Presentation** field, and enter **Approvers** in the **Name** field. Keep **Switch to this presentation** check box selected, and click **Create**.



The screenshot shows a 'Create Presentation' dialog box. It features a title bar with the text 'Create Presentation' and a close button (X). The main content area includes a dropdown menu labeled 'Select from Previous Presentation' with 'Employees' selected. Below this is a text input field labeled 'Name *' containing the text 'Approvers'. There is also an empty text input field labeled 'Description'. At the bottom of the dialog, there are two checkboxes: 'Make default' (unchecked) and 'Switch to this presentation' (checked). At the very bottom, there are two buttons: 'Create' (highlighted in blue) and 'Cancel'.

Notice that all controls of the Employees presentation get copied to the newly created Approvers presentation. The Approvers presentation is an independent presentation. You can add or modify controls, and customize it to your business needs.

5. To switch between your two presentations, use the **Presentation** field at the top of the canvas.

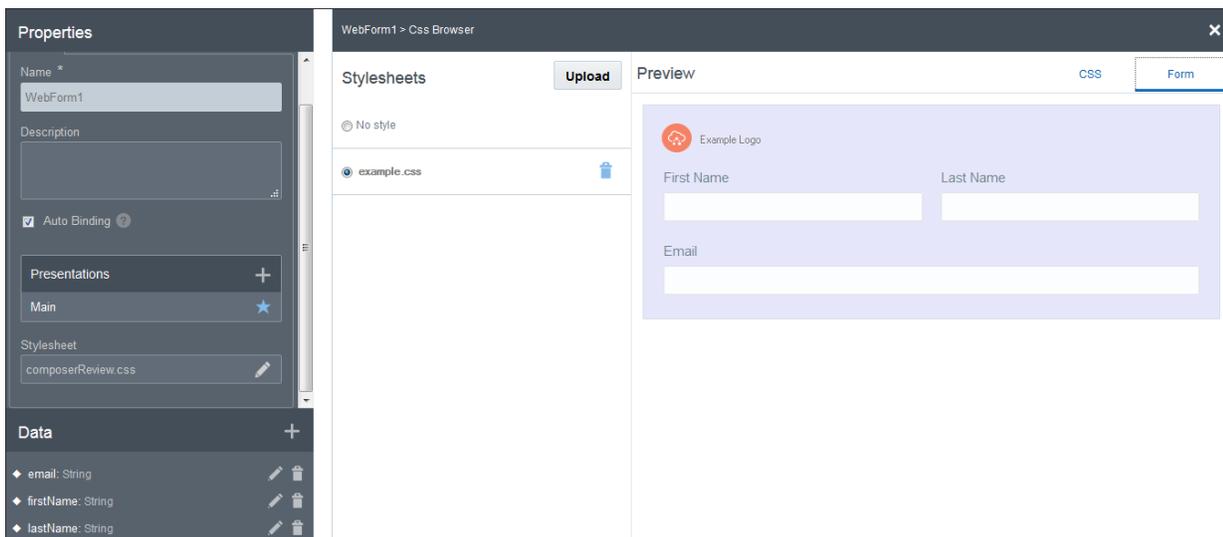
Now let's make the two presentations different.

6. Select different background colors for the presentations.
 - a. Select the Approvers presentation, and click the **Presentation** tab.
 - b. Click the square below **Background Color**, select a light yellow color, add it to custom colors, and click **OK**. Optionally select a border color.
 - c. Repeat the same steps to change the Employees presentation's background color to light green.
7. Make the name fields on the Approvers presentation read-only.
 - a. Select the Approvers presentation.
 - b. Select the **First Name** field. On the **General** tab, scroll down and select **Read Only**. The field turns blank to indicate that its value can't be changed.
 - c. Change the **Last Name** field to read-only.

Change the Form's Stylesheet

To apply your organization's branding, you can upload and apply a stylesheet. You can assign one stylesheet to a form.

1. Click a blank area of the form, then click the **Form** tab on the Properties pane.
2. Scroll down if needed, and click **Edit** in the **Stylesheet** field. The Stylesheets page displays.
3. To upload a stylesheet, click **Upload** and select a stylesheet file (.css or .txt). In the **Stylesheets** area, click an available stylesheet to apply it.
4. In the **Preview** area, click **CSS** to view a stylesheet's format or **Form** to preview it.
5. Click the **X** to close the CSS browser and return to the form.



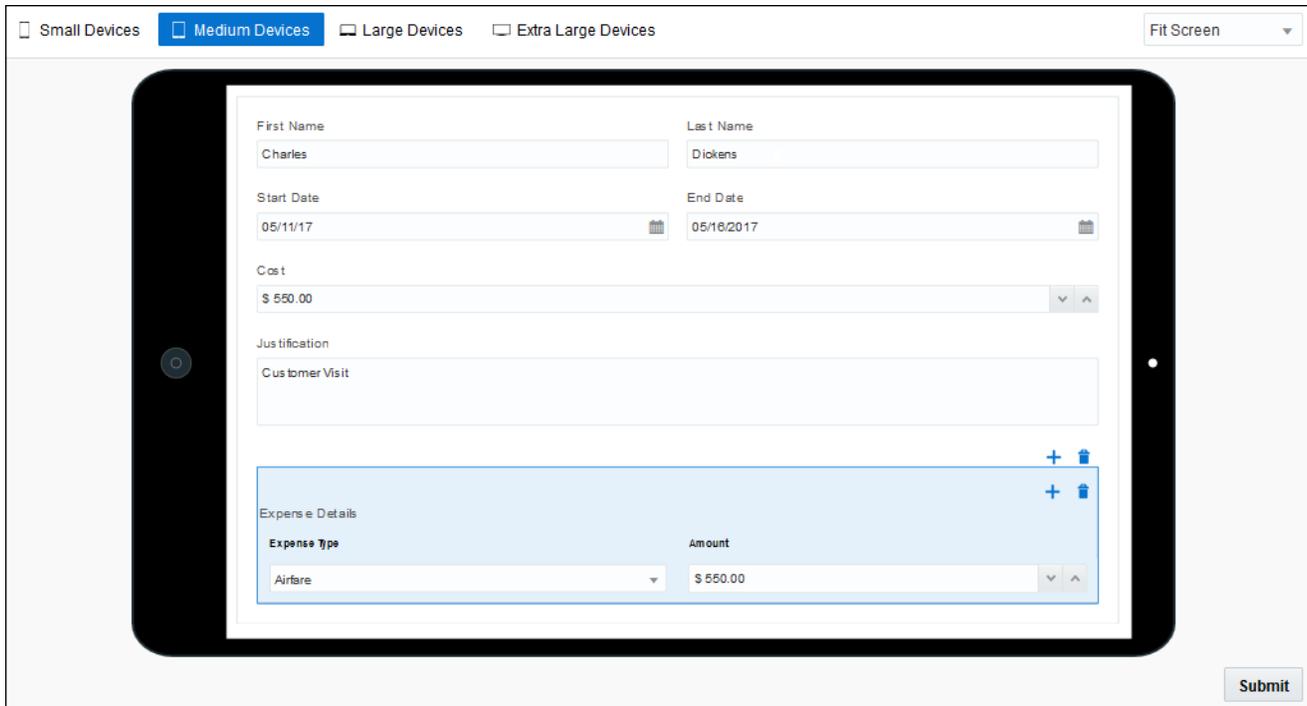
Preview the Form

Forms you create automatically adapt to the screen size, whether end users display them on mobile devices, tablets, or on large screens.

1. In the web forms editor, click **Preview**.
2. In the Preview window, select different device sizes in which to view the form.
3. Enter sample values in the form's controls and click **Submit**.

The field at the bottom of the window shows how form entries (also called *payload values*) are stored as data attributes and used in the process.

4. Click **Close**  to close the Preview window. Click **Save**.



Small Devices Medium Devices Large Devices Extra Large Devices Fit Screen

First Name: Charles Last Name: Dickens

Start Date: 05/11/17 End Date: 05/18/2017

Cost: \$ 550.00

Justification: Customer Visit

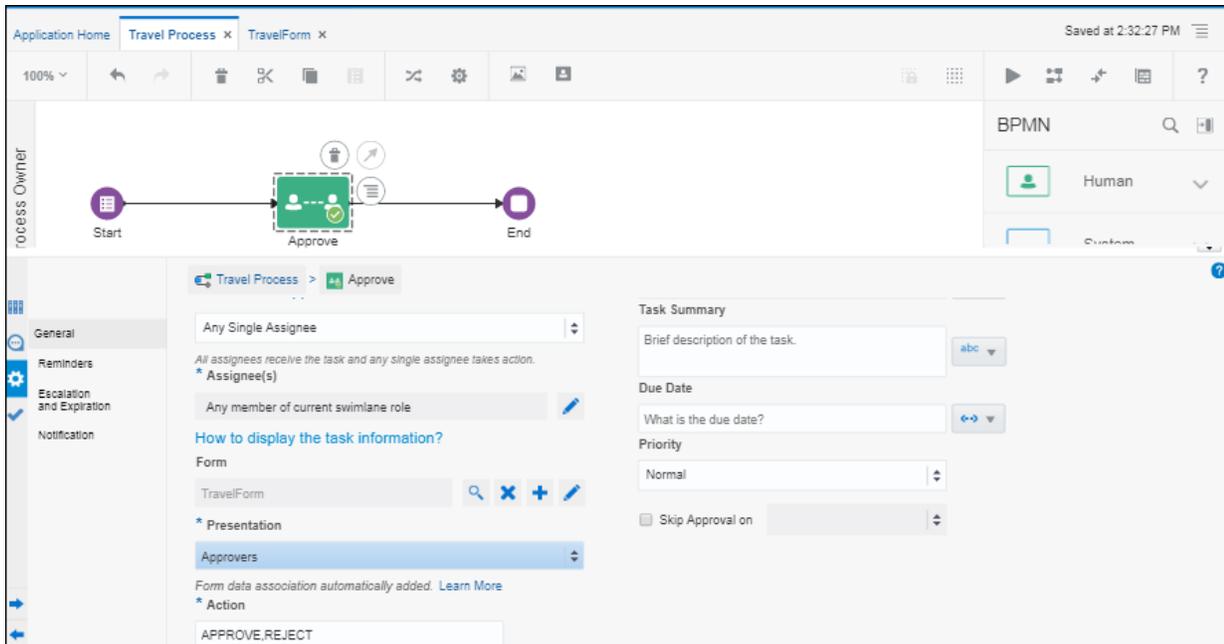
| Expense Type | Amount |
|--------------|-----------|
| Airfare | \$ 550.00 |

Submit

Use Forms and Presentations in a Process

Now let's use the Employees presentation in the start form, and the Approvers presentation in the Approve task.

1. Click the **Application Home** tab, then **Forms**. Notice that your form displays here.
The Forms pane displays all forms created for the application, regardless of whether they're used. Note that you can create and use both web and basic type forms in an application.
2. Click the **Travel Process** tab.
3. Select the Approve human task, and click its action menu. Notice that the **Open Form** command is dimmed, because no form is associated with it yet. Select **Open Properties**. The pane expands to display its properties.
4. In the **Form** field, browse and select **TravelForm**.
5. In the **Presentation** field, select the **Approvers** presentation to display for this task.



Define a Control's Behavior

To change a control's behavior, make use of the web form editor's extensive set of event, action, condition, and connector options. For your travel request, let's apply an event, condition, and action to the Cost control: if the cost is over a certain amount (such as \$500), the Justification text field can't be left blank.

1. From the Approve task's properties, click the **Edit Form** icon for the **Form** field.
2. In the web forms editor canvas, select the Cost control. On the **General** tab, scroll down to the **Events** field and click **Add**  to create an event.

An **event** represents something that may occur, which triggers specific conditions, actions, or REST connector calls.

Each control has possible events you can define for it. The Cost input text field, for example, may fire an event when the form loads (On Load), an end user changes the control's value (On Change), the cursor is in the control (On Focus), the control loses focus (On Blur), or the form is submitted (On Submit).

3. Select **On Change** as the event type and click **Edit**.

The event window displays, with its type listed as *On Change*. Notice how the square buttons contain +s, because you use them to either add an action, if condition, or connector call. Actions display in red, and conditions display in blue.
4. Create a condition and actions for the event.
 - a. Click the blue outlined **+If** button to add a condition. Now solid blue round buttons display to use to configure the condition.
 - b. Below the solid blue **If** button, configure the If condition as follows. Note that related fields display, depending on the selections you make.
 - In the **Control Name** field, select **Cost**.

- In the **Property** field, select **Value**, then select **is greater than** below it.
 - In the **Value** field, enter 500.
- c. Next to the solid blue **Then** button, click the **+Action** button.
- d. Below the solid red Action button that displays, configure the Then condition:
- In the **Control Name** field, select **Justification**.
 - In the **Action** field, select **Required**.
- e. Scroll down. Next to the solid blue **Else** button, click the **+Else Action** button.
- f. Below the solid red Action button, configure the Else condition, which specifies what happens if the condition isn't met:
- In the **Control Name** field, select **Justification**.
 - In the **Action** field, select **Optional**.
- The event is complete: **If** the end user enters a cost over 500, **Then** the Justification field is required, **Else** the Justification field is optional.
- g. Click **OK** to close the event window, then **Save**.

On Change

If + Condition

| Type | Control Name | Property |
|---------|--------------|----------|
| Control | Cost | Value |

is greater than

| Type | Value |
|----------|-------|
| Constant | 500 |

Then + Action

Action

| Control Name | Action |
|---------------|----------|
| Justification | Required |

Else + Else If + Else Action

Action

| Control Name | Action |
|---------------|----------|
| Justification | Optional |

+ Action + If

Cancel OK

Try the Form in Runtime

Let's try using the form as an employee and as an approver.

1. Click **Test**. A Test Application tab opens.
2. Click **Activate**. Leave the **Add Me to All Roles** field checked and click **Activate**. A message displays that the application deployed successfully.
3. Click **Try in Test Mode**.
4. In Runtime, click your Request Travel application under **My Apps**. The Travel form displays. Notice that it's the Employees presentation (green) that you created.

- a. Notice built-in validation for some fields. For example, if you type an invalid character in the date field, an error displays when you change to another field.
 - b. Enter a cost over 500 in the Cost field and change to another field. Notice that an asterisk displays in the Justification field to indicate that it's required.
 - c. Click **Submit**. Now an error indicates that a value is required. Enter a justification and submit again.
5. Click **My Tasks** to view the task as an approver.

Notice that when you open the task, the yellow Approvers presentation displays. Notice that the name fields are read only, as you configured them.

Explore Advanced Form Options

Here are some other areas to explore in the web forms editor:

- **Try out the advanced controls**

Use the Advanced Palette to add advanced items such as panels, sections, tabs, and tables to your form. You can also configure controls that dynamically populate by calling an external service through a connector. See [Configure Advanced Controls](#).

- **Reuse other forms defined in the application**

The Forms Palette lists previously created forms you can reuse in any form. Drag and drop a form onto the canvas and use in one of two ways:

- As a referenced form whose controls can't be edited.
- As single controls that you can edit, reposition, and remove after clicking **Detach**, which removes the link with the original form.

See [Reuse Forms](#).

- **Create a data first form**

This exercise led you through creating a form by adding controls, and data attributes were automatically created through auto binding. You can also select a business type defined for the application, and Process creates a form based on its data attributes. See [Create a Web Form Based on a Business Type](#).

Work with Presentations

A presentation is a single view of the web form.

Presentations are useful:

- When web forms are rendered in different devices
- To present different views of the same data for specific users

When a web form is created, it uses a default presentation. When you add a control to the web form, the new control is added to the currently selected presentation.

Selecting, adding, or deleting a presentation in the web forms editor

1. On the Form tab, notice that the **Presentations** table lists the available presentations. When a web form is created, it uses the **Main** presentation by default.

The **Presentation** tab displays the properties of the currently selected presentation. An asterisk (*) indicates a required property. You can edit the properties of the currently selected presentation and specify new values:

| Field | Description |
|-------------------|---|
| Name | Specify a name for the presentation. |
| Description | Specify a suitable description for the presentation. |
| Border Color | Select a border color from the color palette or enter a hexadecimal value in the adjacent text field. |
| Border Width | Specify a border width in pixels or points. For example, 9px. |
| Border Style | Select a border style (Solid, Dotted, or Dashed) for the presentation. |
| Background Color | Specify a background color for the presentation. Select a background color from the color palette or enter a hexadecimal value in the adjacent text field. |
| Events | Configure events for the presentation. See Add Dynamic Behavior to a Form . |
| Reusable Snippets | <p>Create a global event snippet to reuse across the form's presentation. Use any event block, such as action, condition, or connector, to define a snippet.</p> <p>In addition, all the event snippets you've extracted while defining events for form controls appear here. See Extract a Snippet. Click Edit to view or update a previously extracted snippet. Changes you make at the presentation level reflect in all instances of the snippet across different controls.</p> <p>Note that deleting a global snippet converts all usages of the snippet into local copies.</p> |

| Field | Description |
|-------------------|---|
| Global Connectors | <p>Configure a global connector call to use in one or more controls within a form. Adding a global connector call is similar to adding a connector call within events. See Executing REST Connector Calls in Events.</p> <p>While configuring a global connector, you can also determine when the connector data loads into a control using the Skip Upon Load check box.</p> <ul style="list-style-type: none"> • Deselect the check box (default state) to allow connector data to be populated into a control when the form loads. • Select the check box to prevent connector data being populated into a control when the form loads. When you use this option, you must explicitly execute a connector refresh on the corresponding control for the data to load. <p>While configuring a global connector, specify error handler messages for controls in the Error Handler section and determine how you handle connector error by using the Ignore connector error check box.</p> <ul style="list-style-type: none"> • If you select the check box, the specified error handler messages appear in the form's controls and all other events execute normally when the form loads. • If you deselect the check box, the specified error handler messages appear in the form's controls but all other events fail to execute when the form loads. If multiple global connectors are configured, then you'll see this behavior even if one of the global connectors doesn't have Ignore connector error selected. <p>A global connector call can be reused across different events. Unlike local connectors calls, you don't have to configure it in each event. Instead, you can configure it once at the presentation level and use the data from it in events across different controls.</p> <p>However, a global connector call is not automatically updated. In an event window, click +Action and select the Refresh Global Connector action available for the Presentation control to trigger a new connector call. When you use this action, all other actions present within an event are executed after the global connector update.</p> |

- To add a presentation, click **Create Presentation**  in the **Presentations** table of the **Form** tab, and make an appropriate selection in the Select Presentation Type dialog box.
 - **From Scratch:** Select to create a fresh presentation. In the Create Presentation window, enter a name for the presentation and an optional description. Select **Make default** to make this presentation the default presentation. Select **Switch to this presentation** to switch to this new presentation immediately after creation. Click **Create**.
 - **Clone from Previous Presentation:** Select to create a presentation based on an existing presentation. In the resulting window, select an existing presentation from the list.
 - **Customize from Previous Presentation:** Select to create a customized presentation based on an existing presentation. In the resulting window, select an existing presentation from the list.

 **Note:**

A customized presentation is identical to the base presentation from which it is created. However, unlike a cloned presentation, you cannot add/delete controls or modify control properties within a customized presentation. You can only perform the followings actions on existing controls: hide, show, and modify read-only status. See [Selecting the Right Presentation Type](#) for all the differences between the types of presentations you can create. To edit a customized presentation after creating it, use the **Edit** button next to the presentation switcher.

3. To delete an existing presentation, click the **Delete Presentation** icon in the **Presentations** table. If you try to delete the default presentation, the Delete Default Presentation Confirmation dialog box appears. Select a new default presentation from the list. Click **Delete**.
4. To change the default presentation, in the **Presentations** table, select the **Make this the default presentation** icon next to the presentation you want to change to the default.

Notice that you can switch to a presentation using the drop-down list of available presentations on the top of the canvas.

Selecting the Right Presentation Type

This section lists all the differences between the available presentation types and helps you make the right selection.

| Scratch | Clone | Customize |
|--|---|--|
| Created with no controls. | Created by copying controls from the base presentation. | Created by linking to controls in the base presentation. |
| Adding or editing new controls is allowed. | Adding or editing new controls is allowed. | Adding or editing new controls isn't allowed. Only visibility of the controls can be edited. |
| No base presentation is used. | <ul style="list-style-type: none"> • A base presentation is used; however, deleting the base presentation doesn't affect the cloned presentation. • In addition, a cloned presentation is not in sync with its base presentation. | <ul style="list-style-type: none"> • A base presentation is used and deleting the base presentation deletes all the customized presentations created using it. • A customized presentation is always in sync with its base presentation. |
| Used to design a new presentation. | Used to create an independent presentation with all content from a previous presentation. | Used to create a presentation that is linked and identical to a previous presentation, which only allows for modifying the visibility of controls. |

Some Useful Event Actions for Presentations

This section lists a few useful actions that you can apply to a presentation through the event window. To create an event action, see [specifying actions](#).

- Use the **Set Data** action to set the value of a particular data attribute used in the form. The value is added to the payload. When the event executes, all controls using this data attribute, either directly or indirectly (through the Computed Value option), are updated with this value.
- Use the **Change Presentation** action to switch the presentation of a form or an embedded form on occurrence of an event. See [Change Form Presentations Dynamically](#).
- Use the **Print** action to print a presentation of a form on occurrence of an event. For example, you can define an **On Click** event with the **Print** action for a Button control.
- Use the following actions to manipulate an array or its elements:
 - Use the **Array Value** action to set the value of a particular element in an array on occurrence of an event. Specify the array, index of the element, and value to set—in that order—in the subsequent fields of the event window. The following image shows a value of **option1** set to an element with index **0** in the array named **checklist**.

The screenshot shows the 'Action' configuration window for the 'Array Value' action. The 'Control Name' is 'Presentation', the 'Action' is 'Array Value', and the 'Value' is 'checklist'. The 'Type' is 'Constant', and the 'Value' field contains '0'. The 'Value' field is set to 'option1'.

- Use the **Insert Array Value** action to insert an element into an array. Specify the array, the index at which to insert the element, and the value to insert—in that order—in the subsequent fields of the event window.
- Use the **Append Array Value** action to insert an element at the end of an array. Specify the array and the element to append in the subsequent fields of the event window.
- Use the **Clean Array** action to clear an entire array. Specify the array in the **Value** field.
- Use the **Delete Array Value** action to delete a particular element in an array. Specify the array and the index of the element to delete in the subsequent fields of the event window.

Change Form Presentations Dynamically

Let's consider an example of a travel form to understand how you can use the **Change Presentation** action to dynamically switch a form's presentation in runtime.

Note:

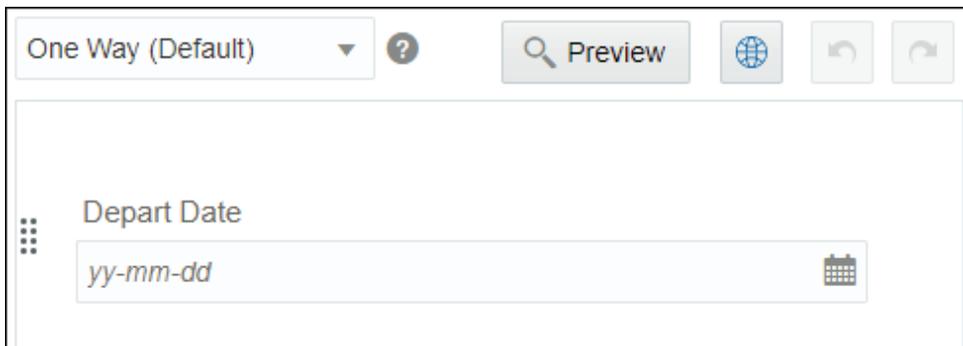
You can also use dynamic data, such as values entered in controls, payload data, or data from connectors to trigger the Change Presentation action.

Here, we create a simple travel form in which a user enters details, such as name, origin, destination, and so on. The user also mentions whether the intended journey is a one-way trip or a round trip using radio buttons, and the date (departure and return) fields are

displayed according to the radio button selection. The date fields are supplied through an embedded web form.

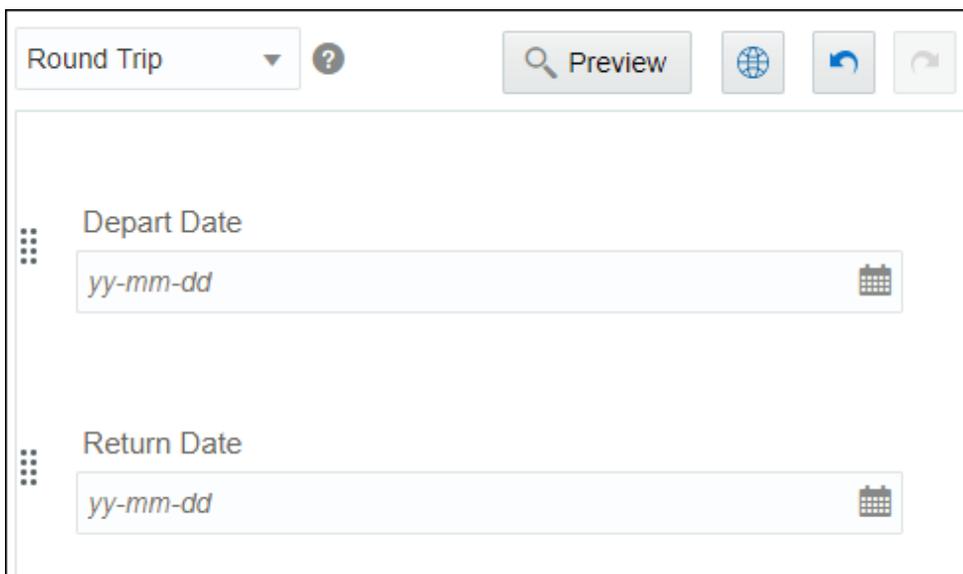
We start by creating this sub-form with date fields:

1. Within the form, create a presentation called **One Way**, make it default, and add a date control named **Depart Date** to it.



The screenshot shows the Oracle APEX form editor interface. At the top, there is a dropdown menu set to "One Way (Default)" with a question mark icon. To the right are buttons for "Preview", a globe icon, and two circular arrows. The main area of the form contains a single date control labeled "Depart Date". The input field has the placeholder text "yy-mm-dd" and a calendar icon on the right side.

2. Create another presentation called **Round Trip** and add two date controls as follows.



The screenshot shows the Oracle APEX form editor interface for the "Round Trip" presentation. At the top, there is a dropdown menu set to "Round Trip" with a question mark icon. To the right are buttons for "Preview", a globe icon, and two circular arrows. The main area of the form contains two date controls. The first is labeled "Depart Date" and the second is labeled "Return Date". Both input fields have the placeholder text "yy-mm-dd" and a calendar icon on the right side.

3. Now, create the main travel form with the following controls:
 - a. Input text controls for name, origin, and destination.
 - b. Radio buttons to specify if it's a one-way journey or round trip.
 - c. Embed the web form created in the previous steps to provide the date controls.

Full Name

Origin City

Destination City

One Way
 Round Trip

Detach

Depart Date

yy-mm-dd

4. Set up option names and values for the radio button control as follows; specify the **One Way** option as default.

Options Source ?

Static From Data Connector

Options Names

Options Values

One Way
Round Trip

One Way
Round Trip

5. Now, define an **On Change** event for the radio button control to change the presentation of the inner web form according to the user's selection. Use the **Change Presentation** action available for the inner form as shown in the following figure:

On Change

If + Condition

Type: Control Control Name: RadioButton Property: Value

is equal to

Type: Constant Value: Round Trip

Then + Action

Action

Control Name: InnerForm Action: Change Presentation Presentation: Round Trip

Else If + Condition

Type: Control Control Name: RadioButton Property: Value

is equal to

Type: Constant Value: One Way

Then + Action

Action

Control Name: InnerForm Action: Change Presentation Presentation: One Way

6. Click **Preview** to test the travel form. When the form loads, the **One Way** option is selected and the default presentation of the inner form loads with only the depart date control.
7. Populate all the controls and select **Round Trip**. The inner form's presentation changes to display two date controls; if you change your selection back to **One Way**, the embedded form displays its default presentation.

The image shows a web form with the following elements:

- Full Name**: A text input field.
- Origin City**: A text input field.
- Destination City**: A text input field.
- One Way**: A radio button.
- Round Trip**: A radio button, which is selected.
- Depart Date**: A date input field with a calendar icon and the placeholder text *yy-mm-dd*.
- Return Date**: A date input field with a calendar icon and the placeholder text *yy-mm-dd*.

Position Controls on Forms

You have many options for positioning controls in your web form.

Rows and Columns

By default, you can drop and position controls onto a web form from any palette in rows and columns. Rows and columns are defined based on where you want to drop your control.

- Each row has a drop target on the top and bottom.
- Each column has a drop target on the left and right.

Each time you drop a control onto the canvas, the control is placed in a new row. If you drop a control beside another control, then that control is added to the same row but in a new column. The column size is based on the number of controls in a row.

The row is based on a 12-column grid system. For example, if the row has four controls, then each uses three columns. When more than 12 controls are added to a row, the remaining controls are displayed below the table.

Panels, Sections, and Tabs

You can group different controls inside a single control by using panels, sections, and tabs.

A **Panel** control allows you to group multiple controls inside it. Drag and drop controls inside a panel and position them as you would in the canvas.

A **Section** control is similar to a panel, but is collapsible. You can use the section control to create groups of controls that users can expand and collapse. To show the section expanded when the form loads, select **Expand** in the General tab. To show the section collapsed when the form loads, deselect **Expand**. You can drag and drop any other controls inside a section control, including panels and other section controls.

A **Tab** control allows you to group controls in different tabs, to create a tabbed view in your web form. By default, when you drag a tab control onto your web form, one tab is created. To add another tab, click **Add tab**. The new tab is added to the right of the existing one. To rearrange the tab order, drag a tab to the right or left of another tab. In small devices, such as mobile phones, tabs are shown as accordions to accommodate the content in the space. You can drag and drop other controls, such as panels, sections, and tables, into an individual tab. When you select a tab, only the controls in the currently selected tab can be seen.

 **Note:**

If you delete a panel, a section, or a tab, all the controls grouped inside the panel, section, or tab also get deleted.

See [Place Controls in Panels, Sections, or Tabs](#).

Tables and Repeatable Sections

Use **Table** controls to conserve space within a web form. The table control allows you to arrange the form controls in a grid pattern. You can add columns into the table and add one control per column. When you delete a column in the table, all the controls in the column get deleted.

If auto binding is selected when you drop a table onto the canvas, then a new entry in the data definition is generated. That attribute is an array of type object. Every time a control is dropped into a column, an attribute inside that list is added.

If auto binding isn't selected, you can select an attribute from the data definition. The controls inside a table can be bound to any attribute in the form data definition. When rendering the form, the data is fetched and shown for that control. However, modifying the value in one row doesn't update every row. Also, when submitting the form, the last control bound to that attribute is the one updating the payload.

You can also select an external source to get the information from the table. In this case, the information from the binding is skipped, and the data is fetched through a connector call. When submitting the form, the payload is modified with the latest data in the table, that is, connector fetched data and user changes.

Use **Repeatable Section** controls to display multiple copies of a set of controls in a web form. Repeatable sections are useful when you want to allow users to enter the same type of input information multiple times in the form; for example, a phone number.

Like tables, you can configure the repeatable section controls such that the information from binding is skipped and data is fetched through a connector call.

Unlike a table, which is structured, a repeatable section is a free-form container. You can add new controls above, below, or between existing controls. You can surround an individual control with a repeatable section control, or it can include an entire section

control. The section control included in the repeatable section must contain all of the web form controls that must be repeated.

Tables and repeatable sections provide a way to add information and content dynamically to web forms. With these controls, you can run **action** and **if** triggers in the form to display dynamically changing information.

See [Configure Tables](#) and [Configure Repeatable Sections](#).

Responsive Layout for Different Devices

The same web form with the same set of controls is rendered in different ways for different devices. For example, if you place controls in your form a certain way for a large device, then the same controls might be rendered in a different way for a small device.

Select the device in which you want the form rendered from the options available in the **Displaying For** field. To see how your form is displayed for the selected device, click **Preview**.

You can also specify absolute column sizes for different devices. To do so, deselect **Automatic column sizes** in the Styling tab, and then enter a number from 1 to 12 in the available fields.

Bind Form Data with Controls

In web forms, data and controls are decoupled. You can define your controls and data independently and then connect the controls to the data using the binding property.

In the web forms editor, on the **Forms** tab, notice that the **Auto Binding** field is selected by default. When the auto binding property is enabled, the web forms editor creates data attributes linking them to the controls as you drag and drop controls from a controls palette onto the canvas. The data attributes are listed in the Data pane. If you disable the **Auto Binding** field, then, you must create data attributes in the Data pane and link them to every unbound control using the **General** tab's **Binding** field.

On the **General** tab, notice that the web forms editor displays the **Binding** field for every control on the canvas. In the **Binding** field, you can specify an attribute for a control by selecting an option from the autocomplete list or entering a valid binding.

Note:

- The name of the binding in the **Binding** field is the same as the name of the control to which it is linked when you use auto-binding.
- When a data attribute is bound—either directly or indirectly (through computed values or events)—to multiple controls, any change to the data attribute is immediately reflected in all associated controls. Repeatable sections or tables bound by the same data attribute are also in sync, that is, addition or deletion of rows in one control is automatically reflected in the other control.

The background colors displayed for the autocomplete options are:

- Green for valid binding names
- Red for invalid binding names

- Blue for complex binding names (their children may contain valid binding names)

If a control isn't linked to an attribute or if a control is linked to an invalid attribute, an error message is displayed in the **Binding** field and an error icon is displayed in the canvas. When a control doesn't have a valid binding, any value entered into the control in runtime is not passed on to the form payload.

To create a new attribute, click the **Add** icon in the Data pane. In the Create Attribute window, enter a name and specify whether the new attribute is a simple type (String, Number, Boolean, Date, Date Time, and Time) or a business type. If you selected **Business** type, then, select an option from the available list of business types. Optionally, enable or disable the list of values. Click **Create**.

 **Note:**

- An attribute name can only start with a lower case letter and can only contain letters, digits, and underscores. Also, an attribute name can't start with XML.
- In the Create Attribute window, if you select **Business** in the **Type** field, then, the business types defined for your application are available in the drop-down list. Notice that the business types defined for your application are listed in the **Business Types Palette**.

To delete an attribute, click the **Delete** icon adjacent to the attribute you want to delete in the Data pane. If the attribute you're trying to delete is bound to at least one control, click **Delete** in the Delete Attribute Confirmation window. Note that the web forms editor clears the binding when you delete an attribute which is bound to a control.

To edit an attribute's name, click the **Edit** icon adjacent to the attribute you want to edit in the Data pane. Enter a new name and click **OK**.

Create a Web Form Based on a Business Type

A business type is used to organize and store related information used in your process, such as employee data.

The web forms editor's Business Types Palette lists all business types defined for your application, including auto-generated ones created from WSDL files or integrations. (Note that any inner types defined in a WSDL file are not displayed.)

Drag a business type from the Business Types Palette onto the canvas, and the web forms editor automatically creates a web form based on the business type's data attributes. You can drop additional controls onto the business type and edit their properties.

A business type of Enum is represented as a customized select control when you drop it onto the form's canvas.

A business type of array is represented as a combination of a customized repeatable section and a suitable additional control, such as input text or number control, depending on whether the array contains strings or numbers. You cannot drop additional controls onto the array business type.

In the Data pane, you can also create business type attributes using the business types defined for your application. See [Bind Form Data with Controls](#) and [Work with Business Objects](#).

Work with Stylesheets and Styling

Using stylesheets and styling, you can customize the appearance of your web form. A stylesheet applies to the entire form. Manage and control the appearance of your web form by applying an available stylesheet to the form or importing one. Styling enables you to change a selected control's appearance. Use styling to define display-specific properties of a currently selected control in your web form.



Note:

Styling applied to an individual control will override the styles applied to the control by the stylesheet.

Changing a Form's Stylesheet

To change a form's stylesheet:

1. In the Properties pane, go to **Stylesheet** and click the Lookup stylesheet icon.
The CSS browser pane opens displaying all the available stylesheets.
2. Select the stylesheet that you want to apply. The selected stylesheet will be applied to your web form.
3. Optionally, import a stylesheet. Click **Upload**. The File Open dialog box opens.
Select the stylesheet you want to import from your local machine and click **Open**. The selected stylesheet will get imported and you can apply it to your form.



Note:

The imported stylesheet will be available in the CSS browser and can be applied to other forms.

4. Preview your form with the selected stylesheet applied to it by selecting the **Form** tab in the Preview pane. Select the **CSS** tab to preview the styles available in the selected stylesheet. Want to know how to customize each control using a style sheet? See [Customizing Web Forms Using CSS](#).

Changing the Style of a Form's Control

To change the styling of a control:

1. Select the control for which you want to change the styling. The Properties pane changes and displays the **General** and **Styling** tabs.
2. Select the **Styling** tab. You can see a number of styling options displayed for that particular control.
3. Select the styling property you want to apply to your control and specify the styling.
See [Styling Properties](#).

Styling Properties

You can change a control's appearance by defining its styling properties on the **Styling** tab in the Properties pane. Use the **Styling** tab to define display-specific properties of the currently selected control.

The following table provides an alphabetical list of the properties available on the **Styling** tab.



Note:

Styling properties are control-specific. Not all of the properties listed below are available for every control.

| Property | Description |
|--------------------------------------|--|
| Automatic column size | Calculates the column size for the control based on the amount of visible controls in the row. Automatic column size is selected by default. Note: In small devices like a phone, each control is displayed in one row when automatic column size is enabled. You can specify absolute column sizes for different device sizes. To do this, deselect Automatic column size and enter a number from 1 to 12 in the four available options: Small, Medium, Large, and Extra large column sizes. |
| Background Color | Specifies the background color of the content area in a control. |
| Border (Color, Width, Style, Radius) | Determines the appearance of the border in the content area of your control. <ul style="list-style-type: none"> • Border Color: Defines the color of the border. • Border Width: Defines the width of the border. Use standard values such as 1in, 5em, or 20px. • Border Style: Defines the style of the border—Solid, Dotted, or Dashed. • Border Radius: Defines the value of rounded border corners. Use standard values such as 1in, 5em, or 20px. |
| Color | Specifies the color of the text in the content area of a control. |
| Control Alignment | Specifies the alignment (left, right, or center) of a control in the form. |
| Control Class Name | Allows customization when you're using a particular Cascading Style Sheet (CSS) in the form. Apply a CSS class to the control by selecting a class name from the properties defined in the form's selected style sheet. |
| Font Size | Defines the font size of the text in the content area of a control. The available values are x-small, small, normal, large, and x-large. |
| Height | Sets an absolute height for the control. Use standard values such as 1in, 5em, or 20px. |
| Label Color | Specifies the color of a control's label. |
| Label Size | Specifies the size of a control's label. The available values are x-small, small, normal, large, and x-large. Note: For a Panel control, using the Standard theme overrides the value in the Label Size property. |

| Property | Description |
|--------------------------------|---|
| Reset Inline Styles to Default | Discards all styling selections made on the Styling tab and restore settings to their default values. |
| Text Alignment | Specifies the alignment (left, right, or center) of the content in a control. This property applies to controls where the user can type in text such as the Input Text control. |
| Theme | Uses a CSS to define the format of a control. This property applies to Panel and Section controls. For a Panel control: <ul style="list-style-type: none">• Standard: Automatically increases the size of the control's label to 24 pixels (font size) and makes the label bold (font weight). The Standard theme overrides the value in the Label Size property.• None: Applies no formatting to the control. None is the default value. For a Section control: <ul style="list-style-type: none">• Indent: Automatically indents the section. With the Indent theme, you can easily nest sections within sections.• None: Applies no formatting to the control. None is the default value. |
| Width | Sets an absolute width for the control. Use standard values such as 5in, 20px, or 5%. |

Customize Web Forms Using CSS

Use cascading style sheets (CSS) to customize the look and feel of your web form elements. Modify visual attributes, such as color, font, or margin, of an element using a custom CSS code.

You can also apply basic styling to form elements using properties available in the forms editor. See [Styling Properties](#). However, styling options available through CSS are far more extensive and specific. The following topics describe all form elements and their corresponding CSS selectors. You can style each of these selectors by modifying several properties, such as color, font, or alignment.

Topics:

- [Style the Form Container](#)
- [Style Form Controls](#)
- [Form Reference Styling](#)
- [Styling for Imported Business Types](#)
- [Manage Style Dynamically](#)
- [Some Useful Styling Tips](#)



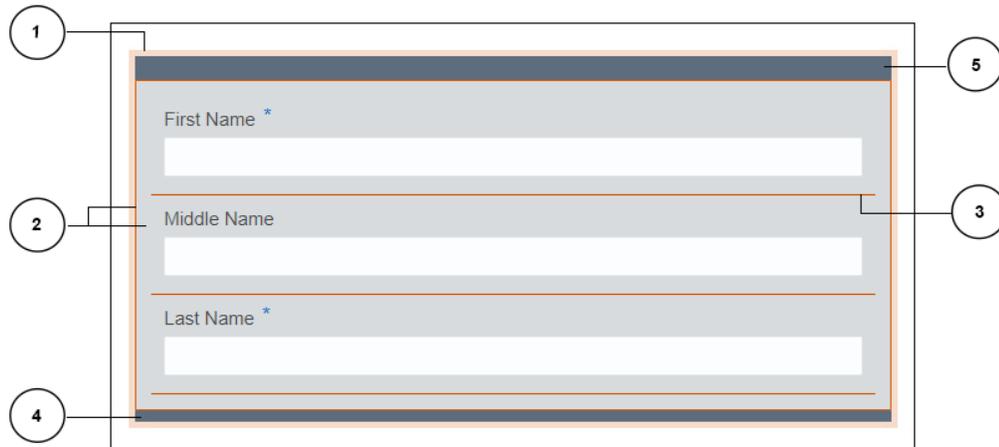
Note:

This section assumes basic working knowledge of CSS.

Style the Form Container

You can style properties that apply to the entire form, such as background or border, using selectors corresponding to the form container.

The following figure shows a sample form container with all its selectors noted:



| No. | Selector Name | Description | Sample CSS Code |
|-----|------------------|---|---|
| 1 | .canvas | This selector corresponds to the outermost container of a form. Use this selector to style properties like form border, padding, and so on. | <pre>.canvas { border: 5px solid #F6DDCC; }</pre> |
| 2 | .canvas__content | Use this selector to style borders and backgrounds for the content area of a form. If you have modified the styling properties of a form through the Properties pane, those styles (backgrounds or borders) are applied to this selector. Tip: To override the applied styles, use !important in your CSS code. | <pre>.canvas__content { background: #D7DBDD !important; border: 1px solid #D35400 !important; }</pre> |
| 3 | .row-control | Use this selector to identify a row in a form. | <pre>.row-control { border-bottom: 1px solid #D35400; }</pre> |
| 4 | .canvas__footer | Use this selector to specify footer for a form. | <pre>.canvas__footer { display: block; background: #5D6D7E; height: 10px; }</pre> |
| 5 | .canvas__header | Use this selector to specify header for a form. Note: Set the <i>display</i> property to <i>block</i> to make headers and footers visible. | <pre>.canvas__header { display: block; background: #5D6D7E; height: 20px; }</pre> |

Style Form Controls

You can style different components of a specific control using corresponding CSS selectors.

The best practice to styling a control is to define a control class name for it. See [Styling Properties](#). The control class name serves to identify the control in the CSS file, and you can use this name as a class name and modify the control's selectors within that class. See [Control Styling Example Using CSS](#).

The following table lists all customizable controls and associated selectors that you can style using CSS.

| UI Control | Selectors | Description |
|--|---|---|
| Label (Corresponds to the title or label of all controls within a form) | .oj-label label | Specify a style for the label. |
| | .oj-label-required-icon | Specify a style for the Required icon. You can apply properties like color, margin, padding, and so on. |
| | .oj-label-required-icon:before | Specify a new signifier instead of the default Required signifier, asterisks (*). |
| | .oj-label-help-icon | Specify a style for the Help icon. |
| | .oj-label-help-icon:before | Specify a new signifier instead of the default Help icon/signifier. |
| Validation Message | .oj-messaging-inline-container .oj-message.oj-message-error | Specify a style for the Validation Message container. |
| | .oj-message-error-icon | Specify a style for the Validation Error Icon container. |
| | .oj-message-error-icon::before | Specify a new signifier instead of the default Error icon/signifier. |
| | .oj-message-content | Specify a style for the validation message content. |
| | .oj-message-summary | Specify a style for the validation message title. |
| | .oj-message-detail | Specify a style for the validation message detail. |
| InputText | .oj-inputtext-input | Specify a style for the Input Text container. |
| | .oj-inputtext.oj-invalid input | Specify a style for the Input Text container when the input is invalid. |
| | .oj-inputtext input[disabled] | Specify a style for the Input Text container when the control is disabled. |
| | .oj-inputtext input[readonly] | Specify a style for the Input Text container when the control is read-only. |
| TextArea | .oj-textarea textarea | Specify a style for the Text Area container. |
| | .oj-textarea.oj-invalid textarea | Specify a style for the Text Area container when the input is invalid. |
| | .oj-textarea textarea[disabled] | Specify a style for the Text Area container when the control is disabled. |
| | .oj-textarea textarea[readonly] | Specify a style for the Text Area container when the control is read-only. |
| Button | .button-control.oj-button | Specify a style for the Button control. |

| UI Control | Selectors | Description |
|------------------|---|---|
| | .button-control.oj-button[disabled] | Specify a style for the button when it is disabled. |
| | .button-control.oj-button .oj-button-text | Specify a style for the button text. |
| | .button-control.oj-button.oj-hover | Specify a style for the button on hover (when you mouse over the button). |
| | .button-control.oj-button.oj-active | Specify a style for the button when it is active. |
| | .button-control.oj-button.oj-focus | Specify a style for the button when it is focused. |
| Select | .oj-select | Specify a style for the Select container. |
| | .oj-select .oj-select-choice | Specify a style for the selected input. |
| | .oj-select.oj-invalid .oj-select-choice | Specify a style for the invalid select input. |
| | .oj-select.oj-disabled .oj-select-choice | Specify a style for the disabled select input. |
| | .oj-select-open-icon | Specify a style for the Drop-Down Icon container. |
| | .oj-select-open-icon:before | Specify a new signifier instead of the default Drop-Down icon/signifier. |
| | .oj-select-choices | Specify a style for the Select container when multiple selection is enabled. |
| | .oj-select-multi .oj-select-selected-choice | Specify a style for selected options when multiple selection is enabled. |
| | .oj-select-multi .oj-select-selected-choice-label | Specify a style for the selected option label. |
| | .oj-select-multi .oj-select-clear-entry:before | Specify a new signifier instead of the default Remove icon/signifier on the selected options. |
| Identify Browser | .oj-identity | Specify a style for the Identity container. |
| | .oj-identity-select | Specify a style for the identity input. |
| | .oj-identity.oj-invalid .oj-select-choice | Specify a style for the invalid identity input. |
| | .oj-identity.oj-disabled .oj-select-choice | Specify a style for the identity input when the control is disabled. |
| | .oj-select-open-icon | Specify a style for the Drop-Down Icon container. |
| | .oj-select-open-icon:before | Specify a new signifier instead of the default Drop-Down icon/signifier. |
| | .oj-select-choices | Specify a style for the Identity container when multiple selection is enabled. |
| | .oj-select-multi .oj-select-selected-choice | Specify a style for selected options when multiple selection is enabled. |
| | .oj-select-multi .oj-select-selected-choice-label | Specify a style for the selected option label. |
| | .oj-select-multi .oj-select-clear-entry:before | Specify a new signifier instead of the default Remove icon/signifier on the selected options. |

| UI Control | Selectors | Description |
|------------------------|---|--|
| Checklist and Checkbox | .oj-checkboxset | Specify a style for the control container. |
| | .oj-choice-row | Specify a style for each check option. |
| | .oj-choice-row input | Specify a style for the check boxes in each row. |
| | .oj-choice-row label | Specify a style for the labels in each row. |
| | .oj-checkboxset.oj-invalid .oj-checkbox:not(.oj-disabled) | Specify a style for the invalid option. |
| | .oj-choice-row input[disabled] | Specify a style for the disabled option. |
| | .oj-choice-row-inline | Specify a style for inline options. |
| Radio Button | .oj-radioset | Specify a style for the control container. |
| | .oj-choice-row | Specify a style for each option. |
| | .oj-choice-row input | Specify a style for the radio button in each row. |
| | .oj-choice-row label | Specify a style for the labels in each row. |
| | .oj-radioset.oj-invalid .oj-radio:not(.oj-disabled) | Specify a style for the invalid option. |
| | .oj-choice-row input[disabled] | Specify a style for the disabled option. |
| | .oj-choice-row-inline | Specify a style for inline options. |
| Number | .oj-inputnumber-wrapper | Specify a style for the Number container. |
| | .oj-inputnumber input | Specify a style for the number input. |
| | .oj-inputnumber.oj-invalid input | Specify a style for the invalid number input. |
| | .oj-inputnumber input[disabled] | Specify a style for the number input when the control is disabled. |
| | .oj-inputnumber input[readonly] | Specify a style for the number input when the control is read-only. |
| Date Time | .oj-inputdatetime-input-container | Specify a style for the entire Date Time control container. |
| | .oj-inputdatetime-input-container input | Specify a style for the input area of the Date Time control. |
| | .oj-inputdatetime-input-container input[disabled] | Specify a style for the Date Time control when it is disabled. |
| | .oj-inputdatetime-input-container input[readonly] | Specify a style for the Date Time control when it is read-only. |
| | .oj-inputdatetime-input-trigger | Specify a style for the Calendar and Time icons container. |
| | .oj-inputdatetime-calendar-icon | Specify a style for the Calendar icon. |
| | .oj-inputdatetime-calendar-icon:before | Specify a new signifier instead of the default Calendar icon/signifier. |
| | .oj-inputdatetime-time-icon | Specify a style for the Time icon. |
| | .oj-inputdatetime-time-icon:before | Specify a new signifier instead of the default Time icon/signifier. |
| Date | .oj-inputdatetime-date-only | Use this selector before the properties below to apply styling specific to the Date control. |

| UI Control | Selectors | Description |
|------------|---|--|
| | .oj-inputdatetime-input-container | Specify a style for the entire Date control container. |
| | .oj-inputdatetime-input-container input | Specify a style for the input area of the Date control. |
| | .oj-inputdatetime-input-container input[disabled] | Specify a style for the Date control when it is disabled. |
| | .oj-inputdatetime-input-container input[readonly] | Specify a style for the Date control when it is read-only. |
| | .oj-inputdatetime-input-trigger | Specify a style for the Calendar icon container. |
| | .oj-inputdatetime-calendar-icon | Specify a style for the Calendar icon. |
| | .oj-inputdatetime-calendar-icon:before | Specify a new signifier instead of the default Calendar icon/signifier. |
| Time | .oj-inputdatetime-time-only | Use this selector before the properties below to apply styling specific to the Time control. |
| | .oj-inputdatetime-input-container | Specify a style for the entire Time control container. |
| | .oj-inputdatetime-input-container input | Specify a style for the input area of the Time control. |
| | .oj-inputdatetime-input-container input[disabled] | Specify a style for the Time control when it is disabled. |
| | .oj-inputdatetime-input-container input[readonly] | Specify a style for the Time control when it is read-only. |
| | .oj-inputdatetime-input-trigger | Specify a style for the Time icon container. |
| | .oj-inputdatetime-time-icon | Specify a style for the Time icon. |
| | .oj-inputdatetime-time-icon:before | Specify a new signifier instead of the default Time icon/signifier. |
| Email | .oj-inputtext-input | Specify a style for the input container. |
| | .oj-inputtext.oj-invalid input | Specify a style for the input container when the input is invalid. |
| | .oj-inputtext input[disabled] | Specify a style for the input container when the control is disabled. |
| | .oj-inputtext input[readonly] | Specify a style for the input container when the control is read-only. |
| URL | .oj-inputtext-input | Specify a style for the input container. |
| | .oj-inputtext.oj-invalid input | Specify a style for the input container when the input is invalid. |
| | .oj-inputtext input[disabled] | Specify a style for the input container when the control is disabled. |
| | .oj-inputtext input[readonly] | Specify a style for the input container when the control is read-only. |
| Message | p, h1, h2, h3, h4, h5, h6 | Use these selectors to style a message based on its type (<i>p</i> or <i>h1</i> to <i>h6</i>). Use the control class name to make it specific. |
| Money | .oj-inputnumber-wrapper | Specify a style for the Money container. |

| UI Control | Selectors | Description |
|------------------------|--|--|
| | .oj-inputnumber input | Specify a style for the money input. |
| | .oj-inputnumber.oj-invalid input | Specify a style for the invalid money input. |
| | .oj-inputnumber input[disabled] | Specify a style for the money input when the control is disabled. |
| | .oj-inputnumber input[readonly] | Specify a style for the money input when the control is read-only. |
| Phone | .oj-inputtext-input | Specify a style for the input container. |
| | .oj-inputtext.oj-invalid input | Specify a style for the input container when the input is invalid. |
| | .oj-inputtext input[disabled] | Specify a style for the input container when the control is disabled. |
| | .oj-inputtext input[readonly] | Specify a style for the input container when the control is read-only. |
| Image | .image-control | Specify a style for the Image container. |
| Video | iframe[name=Video] | Specify a style for the Video container. |
| Link | .anchorLink a | Specify a style for the Link container. |
| Panel | No particular selector | Use the control class name in the Properties pane to build a selector. |
| Section and Mobile Tab | .oj-collapsible-header | Specify a style for the Section Header container. |
| | .oj-collapsible-wrapper | Specify a style for the section header content. |
| | .oj-collapsible-header-icon.oj-collapsible-open-icon | Specify a style for the Section Open icon. |
| | .oj-collapsible-header-icon.oj-collapsible-open-icon:before | Specify a new signifier instead of the default Open icon/signifier. |
| | .oj-collapsible-header-icon.oj-collapsible-close-icon | Specify a style for the Section Close icon. |
| | .oj-collapsible-header-icon.oj-collapsible-close-icon:before | Specify a new signifier instead of the default Close icon/signifier. |
| | .oj-collapsible-header p | Specify a style for the header type: paragraph. |
| | .oj-collapsible-header h1 | Specify a style for the header type: title1 to title6. |
| | .oj-collapsible-header h2 | |
| | .oj-collapsible-header h3 | |
| | .oj-collapsible-header h4 | |
| | .oj-collapsible-header h5 | |
| | .oj-collapsible-header h6 | |
| Tab | .tab-container | Specify a style for the Tabs container. |
| | .oj-tabs-tab | Specify a style for the tab header. |
| | .oj-tabs-tab.oj-selected | Specify a style for the selected tab header. |
| | .oj-tabs-tab.oj-disabled | Specify a style for the disabled tab header. |
| | .oj-tabs-title | Specify a style for the tab title. |
| | .oj-tabs-panel | Specify a style for the tab content. |

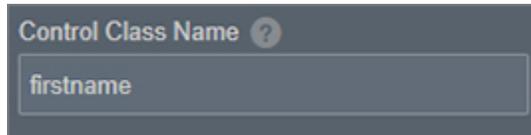
| UI Control | Selectors | Description |
|---------------|------------------------------|--|
| Table | .table-container | Specify a style for the entire Table container (including the label). |
| | .table-control | Specify a style only for the table. |
| | .oj-table-header | Specify a style for the table header section (<i>thead</i>). |
| | .oj-table-header-row | Specify a style for the table header row, containing all table header columns (<i>tr</i>). |
| | .oj-table-column-header-cell | Specify a style for the Table Column Header container (<i>th</i>). |
| | .oj-table-column-header-text | Specify a style for the text in each column. |
| | .oj-table-body | Specify a style for the table body containing all items (<i>tbody</i>). |
| | .oj-table-body-row | Specify a style for each row inside the table body (<i>tr</i>). |
| | .table-cell | Specify a style for a particular cell inside the body. |
| | .table-cell.oj-selected | Specify a style for the selected cell. |
| | button.add | Use this selector to add a button within a table. |
| | button.delete | Use this selector to remove a button from within a table. |
| | Repeatable Section | .repeatable-section |
| button.add | | Use this selector to add a button within a repeatable section. |
| button.delete | | Use this selector to remove a button from within a repeatable section. |

Control Styling Example Using CSS

Let's apply basic styling to a control to understand the usage of control class names and selectors. The following figure shows an Input Text control with no styling applied to it:

To apply styling to this control through CSS:

1. Select the control in the form canvas.
2. In the Properties pane, select the **Styling** tab.
3. In the **Control Class Name** field, enter a name to identify the control in the CSS file as shown in the following figure:



4. Now, use this name in your CSS code to modify the control's properties or selectors. The following sample code shows styling of the control's label:

```
.firstname .oj-label label {  
  font-size: 20px;  
  color: blue;  
}  
.firstname .oj-label-required-icon:before {  
  content: "#";  
}
```

5. Upload this style sheet from the form's Properties pane. The customized Input Text control appears as follows:



6. Similarly, you can customize other selectors associated with this control. You can also apply more than one class names to the control. In addition, you can reuse the same class name with other controls that require similar styling.

 **Note:**

If you do not specify a control class name in the CSS code while defining styles for a selector, the styles will apply to all controls that satisfy the selector.

Form Reference Styling

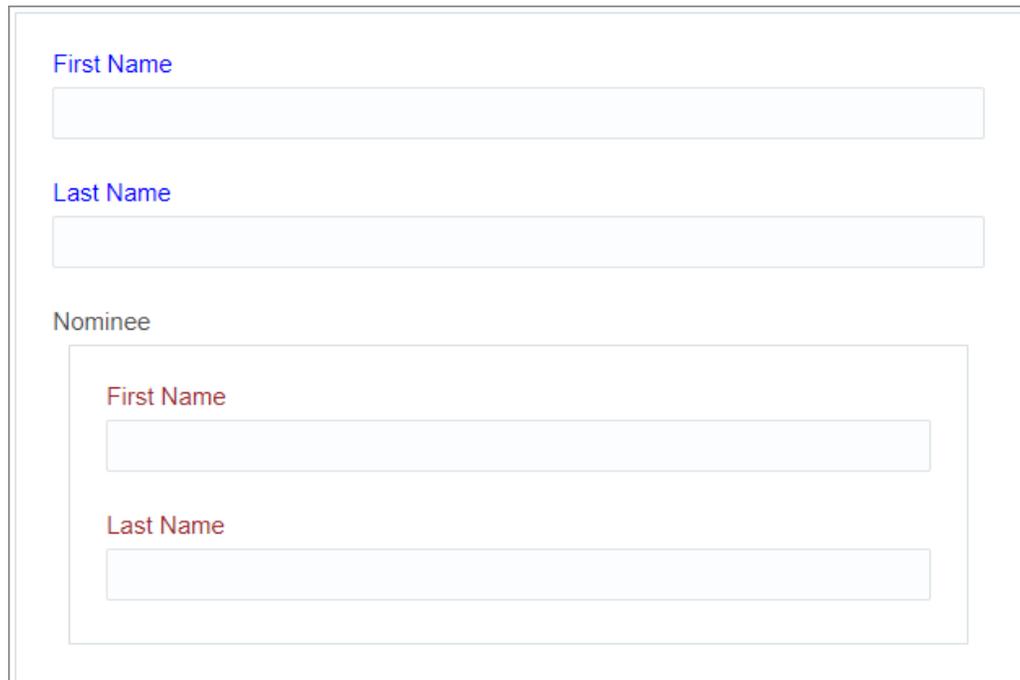
When you import a form into another form by reference, the parent form's styling is automatically applied to the imported form. This behavior ensures that the specified style is enforced for a form, even when there are references.

If you want to retain the styling of the imported form, include the CSS code of the child into the parent form's CSS file. Also, make sure to use the corresponding class names in the code to apply styles specifically to the controls in the imported form.

The following example shows the CSS code of an imported form (along with a control class name) added to its parent's CSS file:

```
.oj-label label {  
    color: blue;  
}  
.nominee .oj-label label {  
    color: brown;  
}
```

The resulting form styling appears as follows:



The screenshot displays a form with three main sections. The first section is labeled 'First Name' in blue text, followed by a text input field. The second section is labeled 'Last Name' in blue text, followed by a text input field. The third section is labeled 'Nominee' in blue text and contains a nested container. Inside this container, there are two sections: 'First Name' and 'Last Name', both labeled in brown text, each followed by a text input field. This visualizes the CSS rules where the parent form's labels are blue, and the child form's labels are brown.

 **Note:**

You must have added the same class name, in this case *nominee*, to the controls of the child form before using it in the parent form's CSS file.

Styling for Imported Business Types

When you drag a business type from the Business Types Palette onto the canvas, a section with a set of controls is created based on the business type's data attributes. All styles defined for the form through CSS are also applied to this auto-generated section of controls.

Manage Style Dynamically

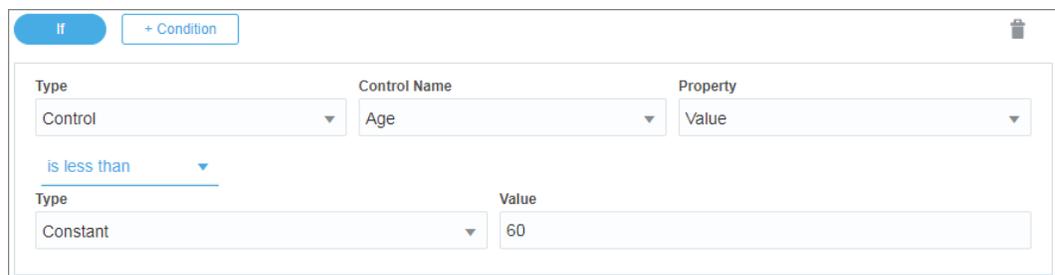
You can manage styling of a control dynamically through the Event window. Want to know more about events? See [Add Dynamic Behavior to a Form](#).

Within the Event window, you can specify actions to add or remove class names to a control when a condition is fulfilled. Based on the class name assigned, the styling of the control changes during execution. To better understand dynamic styling, let us take a look at the following example.

Consider a Number control (Age) for which the allowed value is any number less than 60. Now, let us create an event to style the control based on the value entered by the user during execution.

A screenshot of a web form control labeled "Age". The control is a text input field containing the number "0". To the right of the input field are two small buttons: a downward-pointing arrow and an upward-pointing arrow, indicating a spinner control.

1. In the web forms editor, select the **Age** control, and locate the **Events** option in the **General** tab of the Properties pane.
2. Click the **Add** icon to define an event.
The event field changes to a drop-down menu listing event options available for the control.
3. Select the **On Change** option from the menu.
4. In the Event window, define an If condition to specify allowed values for the control as less than 60.

A screenshot of the "Event window" in a web forms editor. At the top, there is a blue button labeled "if" and a "+ Condition" button. Below this, there is a table-like structure for defining the condition. The first row has three columns: "Type" with a dropdown menu set to "Control", "Control Name" with a dropdown menu set to "Age", and "Property" with a dropdown menu set to "Value". The second row shows the condition "is less than" with a dropdown arrow. The third row has two columns: "Type" with a dropdown menu set to "Constant" and "Value" with a text input field containing "60".

5. Specify actions for Then and Else parts of the condition as shown in the following figure.
The actions defined make sure that the appropriate class name is applied to the control based on the value entered by the user. Subsequently, the styling defined for the class name in CSS is dynamically applied to the control.

The screenshot shows the Oracle APEX configuration interface for a control. It is divided into 'Then' and 'Else' sections. The 'Then' section contains two actions: 'Add Class' (Control Class Name: allowed) and 'Remove Class' (Control Class Name: notallowed). The 'Else' section contains two actions: 'Add Class' (Control Class Name: notallowed) and 'Remove Class' (Control Class Name: allowed). Each action is configured for the 'Age' control.

6. In this case, you can define styling for class names in the CSS file as follows:

```
.allowed .oj-inputnumber input {
color: green;
}
.notallowed .oj-inputnumber input {
color: red;
}
```

7. When the user enters a value into the control, the styling changes as follows:

- If the value entered is less than 60, the following style is rendered.



- If the value entered is greater than or equal to 60, the following style is rendered.



Some Useful Styling Tips

Here are some useful tips and best practices to make forms styling through CSS easy to understand, maintain, and scale.

- Follow a notation for control class names. There are different naming conventions, such as Block Element Modifier (BEM), that you can use to make class names informative and readable.
- Use the *!important* attribute effectively. This attribute allows you to override a style that has a higher specificity or priority.
- Use Scalable Vector Graphics (SVG) images for icons and backgrounds. SVG images scale correctly, avoiding any loss in quality when the form is rendered on different devices.
- Avoid using properties that alter the form layout and behavior, such as *float*, *display*, *flex*, and so on.
- Avoid fixed dimensions while styling controls. The layout of a form changes based on the device, and the content within it adjusts to the available space. If an input text control's width is set to a fixed value of 400 pixels, the control may not display properly on a phone.

Configure Basic Controls

The Basic Palette contains a set of basic controls that you can drag onto the canvas. Note that each time you select a control on the canvas, the **General** and **Styling** tabs become available in the Properties pane to configure settings specific to the selected control. An asterisk (*) indicates a required property.

Topics:

- [Configure Text Input and Area Fields](#)
- [Configure Buttons](#)
- [Configure Drop-down Select Fields](#)
- [Configure Check Lists and Check Boxes](#)
- [Configure Radio Buttons](#)
- [Configure Number Fields](#)
- [Configure Date and Time Fields](#)
- [Configure Email Fields](#)
- [Configure Web Address URL Fields](#)
- [Configure Message Fields](#)
- [Configure Links](#)

Configure Text Input and Area Fields

A text input control allows users to enter short, single line text entries. A text area control allows users to enter longer, multi-line text entries. In a text area control, as a user enters multi-line text, if needed, a scroll bar appears to scroll down and view the entire text.

To configure a text input or a text area control:

1. From the Basic Palette, drag the **Input text** control or the **Text Area** control onto the canvas.
2. Select the control and edit its properties on the Properties pane's **General** tab.

| Field | Description |
|----------------|---|
| Name | An internal identifier that you will use to identify the control. |
| Binding | Defines a link between the control and a data attribute. |
| Computed Value | Set this property to apply a computation to the control. See Creating Computed Controls . |
| Placeholder | Text that will appear in the control until any text is entered. If text is removed from the control, the text specified in this field reappears. |
| Hint | Hint text that will display to users when a user clicks into the control. |
| Help | Help text that will appear for the control. If you specify help text, a help icon appears next to the name of the control and displays the help text when a user hovers over it or clicks it. |
| Min Length | Defines the minimum number of characters a user must enter into the control. Leave this field blank if you don't want to define the minimum length. |
| Max Length | Defines the maximum number of characters a user can enter into the control. |
| Rows | Defines the number of text rows that will be visible to a user. A scroll bar appears automatically when the number of text rows entered by the user is greater than the value specified in this field. |
| Pattern | Allows you to define custom validations on the type of text a user enters into the control. Enter a pattern using regular expressions. When you specify a pattern for the control, you can also specify a message in the Pattern Validation Message field that will display if the validation fails. |
| Password | Allows you to create a secure text field that masks the characters entered into it. |
| Events | Allows you to configure events for the control. You can assign actions or If/Then/Else conditions to the control based on a selected event. See Add Dynamic Behavior to a Form . |

3. On the **Styling** tab, edit the control's [styling properties](#).
4. Click **Preview** to try out using the control.

Configure Buttons

Use a button control to add a button to your web form.

To configure a button control:

1. From the Basic Palette, drag the **Button** control onto the canvas.
2. Select the control and edit its properties on the Properties pane's **General** tab.

| Field | Description |
|--------|--|
| Name | An internal identifier that you will use to identify the control. |
| Events | Allows you to configure events for the control. You can assign actions or If/Then/Else conditions to the control based on a selected event. See Add Dynamic Behavior to a Form . |

3. On the **Styling** tab, edit the control's [styling properties](#).
4. Click **Preview** to try out using the control.

Configure Drop-down Select Fields

Use a drop-down select control to add a drop-down list to your web form.

To configure a drop-down select control:

1. From the Basic Palette, drag the **Select** control onto the canvas.
2. Select the control and edit its properties on the Properties pane's **General** tab.

| Field | Description |
|----------------|---|
| Name | An internal identifier that you will use to identify the control. |
| Binding | Defines a link between the control and a data attribute. |
| Computed Value | Set this property to apply a computation to the control. See Creating Computed Controls . |
| Hint | Hint text that will display to users when a user clicks into the control. |
| Help | Help text that will appear for the control. If you specify help text, a help icon appears next to the name of the control and displays the help text when a user hovers over it or clicks it. |
| Multiple | Determines whether users can select multiple values in the control. If this field is disabled, users can select only one option in the control. |

| Field | Description |
|----------------|--|
| Options Source | Select a source (Static, From Data, and Connector). <ul style="list-style-type: none"> • Static: Specify choices using Options Names and Options Values fields. Use Options Names to specify the label to display for an option and use Options Values to specify an internal value for an option. • From Data: In the Options List field, select a list of values options source from the data definitions available in the web form. If you selected a list of complex elements, then, in the Label Binding field, specify a data attribute that will display as the label and in the Value Binding field, specify a data attribute that will be the value. • Connector: Specify a REST connector, a resource, and an operation to use. Specify parameters to pass to the connector and define how the response should be mapped to the control properties. See Populate Controls Using REST Calls. |
| Filter | Select this check box to apply a filter to the control. See Specifying Filters for Controls . |
| Default Value | If you selected Static in the Options Source field, then, specify a default option in this field. If you selected From Data or Connector in the Options Source field, then, select either the first or the last value as the default value. |
| Events | Allows you to configure events for the control. You can assign actions or If/Then/Else conditions to the control based on a selected event. See Add Dynamic Behavior to a Form . |

3. On the **Styling** tab, edit the control's [styling properties](#).
4. Click **Preview** to try out using the control.

Some Useful Event Actions Related to Drop-down Select Fields

This section lists a few useful actions that you can define using a **Select** control. To create an event action, see [Specify Actions](#).

- You can fetch the selected object label using the **Selected Label** option in the **Property** field.
- When a **Select** control is populated by a connector or data attribute, you can fetch any property of the selected object using the **Selected Property** option in the **Property** field.

Configure Check Lists and Check Boxes

Use a check list control to add a list of options to your web form. A check list allows users to select one or more options. You can add a check box control to your web form to allow users to specify a true or false value.

Note that a check box control is set to false by default. You can edit the **Default Value** property on the **General** tab to change the default value.

To configure a check list or a check box control:

1. From the Basic Palette, drag the **Checklist** or the **Checkbox** control onto the canvas.

2. Select the control and edit its properties on the Properties pane's **General** tab.

| Field | Description |
|----------------|--|
| Name | An internal identifier that you will use to identify the control. |
| Label | Specifies the control name that displays to the user. For check box controls, if you select the Inline HTML check box below the Label field, the value that you enter in the Label field is treated as an inline HTML. |
| Binding | Defines a link between the control and a data attribute. |
| Computed Value | Set this property to apply a computation to the control. See Creating Computed Controls . |
| Help | Help text that will appear for the control. If you specify help text, a help icon appears next to the name of the control and displays the help text when a user hovers over it or clicks it. |
| Inline | Specify the layout of the options defined for the check list control. If enabled, the layout changes from vertical to horizontal. |
| Options Source | Select a source (Static, From Data, and Connector). <ul style="list-style-type: none"> • Static: Specify choices using Options Names and Options Values fields. Use Options Names to specify the label to display for an option and use Options Values to specify an internal value for an option. • From Data: In the Options List field, select a list of values options source from the data definitions available in the web form. If you selected a list of complex elements, then, in the Label Binding field, specify a data attribute that will display as the label and in the Value Binding field, specify a data attribute that will be the value. • Connector: Specify a REST connector, a resource, and an operation to use. Specify parameters to pass to the connector and define how the response should be mapped to the control properties. See Populate Controls Using REST Calls. |
| Filter | Select this check box to apply a filter to the control. See Specifying Filters for Controls . |
| Default Value | If you selected Static in the Options Source field, then, specify a default option in this field. If you selected From Data or Connector in the Options Source field, then, select either the first or the last value as the default value. For the check box control, select either True or False as the default value. |
| Events | Allows you to configure events for the control. You can assign actions or If/Then/Else conditions to the control based on a selected event. See Add Dynamic Behavior to a Form . |

3. On the **Styling** tab, edit the control's [styling properties](#).
4. Click **Preview** to try out using the control.

Select or Unselect All Check List Options

You can select all options in a checklist at the same time by specifying an action **Select All Values**. Similarly, you can unselect all options at the same time in a checklist by specifying an action **Unselect All Values**.

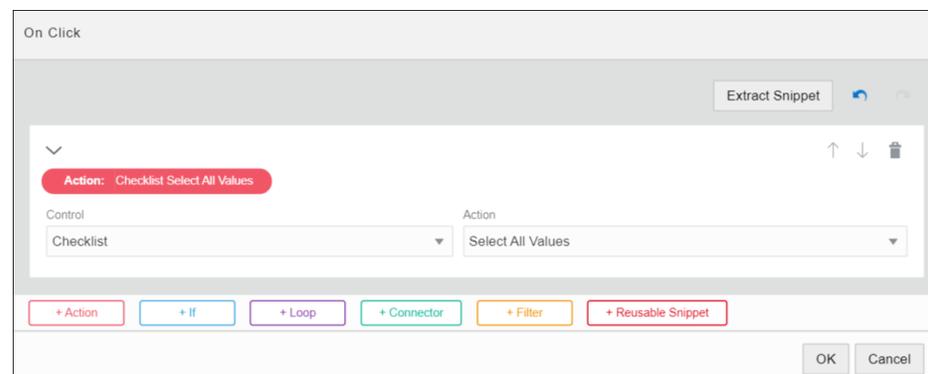
For example, configure a button control with an On Click event and then use it to select all options available in a checklist. Or configure a check box control with an On Change event and then use it to unselect all options in the check list.

See [Specify Actions](#).

See [Configure Events](#).

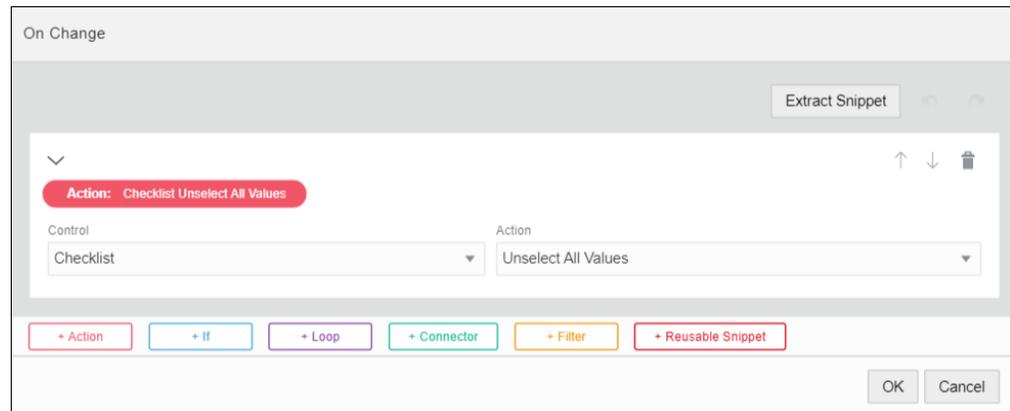
Example of how to configure actions to select or unselect all options in a checklist at the same time:

1. Create a form with a checklist control named **Colors** with three color options - red, blue, and green.
2. From the Basic palette, drag and drop a button control and a checkbox control under the checklist control.
3. Select the button control, and configure the following from the **General** tab of the Properties pane.
 - a. In the **Label** field, enter the name as *Select all colors*.
 - b. Scroll down till you find **Events**. Click  to add and define an event.
 - c. Select **On Click** from the drop-down list and then click the editing icon.
 - d. In the On Click window, click **+Actions** to define an action.
 - e. Select **Checklist** in the **Control** drop-down list and then select the action **Select All Values** from the **Action** drop-down list.

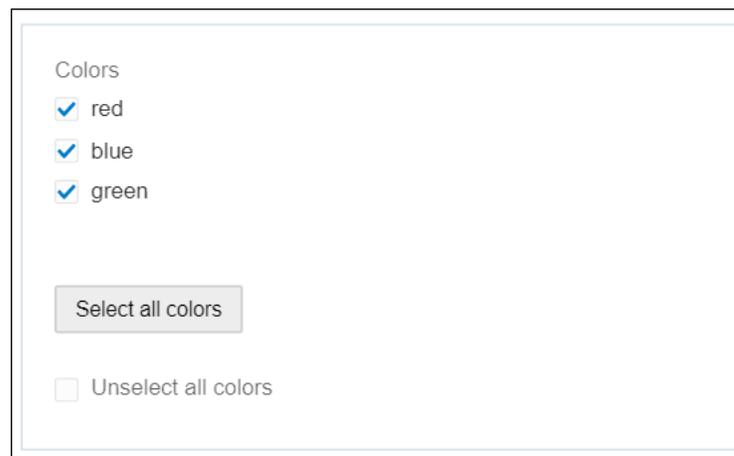


- f. Click **OK**.
4. Select the checkbox control, and configure the following from the **General** tab of the Properties pane.
 - a. In the **Label** field, enter the name as *Unselect all colors*.
 - b. Scroll down till you find **Events**. Click  to add and define an event.
 - c. Select **On Change** from the drop-down list and then click the editing icon.

- d. In the On Change window, click **+Actions** to define an action.
- e. Select **Checklist** in the **Control** drop-down list and then select the action **Unselect All Values** from the **Action** drop down list.



- f. Click **OK**.
5. Preview the form and see how it works.
 - a. Click **Preview**.
 - b. When the form loads, click **Select all colors**. Notice that all the options in the **Colors** checklist gets selected.



- c. Now, select the **Unselect all colors** checkbox. Notice that all the options in the **Colors** checklist gets unselected.

Configure Radio Buttons

Add mutually exclusive radio buttons to your web form. Radio buttons allow users to select an option from a set of available options.

To configure a radio button control:

1. From the Basic Palette, drag the **Radio Button** control onto the canvas.
2. Select the control and edit its properties on the Properties pane's **General** tab.

| Field | Description |
|----------------|--|
| Name | An internal identifier that you will use to identify the control. |
| Binding | Defines a link between the control and a data attribute. |
| Computed Value | Set this property to apply a computation to the control. See Creating Computed Controls . |
| Help | Help text that will appear for the control. If you specify help text, a help icon appears next to the name of the control and displays the help text when a user hovers over it or clicks it. |
| Inline | Specify the layout of the options defined for the control. If enabled, the layout changes from vertical to horizontal. |
| Options Source | Select a source (Static, From Data, and Connector). <ul style="list-style-type: none"> • Static: Specify choices using Options Names and Options Values fields. Use Options Names to specify the label to display for an option and use Options Values to specify an internal value for an option. • From Data: In the Options List field, select a list of values options source from the data definitions available in the web form. If you selected a list of complex elements, then, in the Label Binding field, specify a data attribute that will display as the label and in the Value Binding field, specify a data attribute that will be the value. • Connector: Specify a REST connector, a resource, and an operation to use. Specify parameters to pass to the connector and define how the response should be mapped to the control properties. See Populate Controls Using REST Calls. |
| Filter | Select this check box to apply a filter to the control. See Specifying Filters for Controls . |

| Field | Description |
|---------------|---|
| Default Value | If you selected Static in the Options Source field, then, specify a default option in this field. If you selected From Data or Connector in the Options Source field, then, select either the first or the last value as the default value. |
| Events | Allows you to configure events for the control. You can assign actions or If/Then/Else conditions to the control based on a selected event. See Add Dynamic Behavior to a Form . |

3. On the **Styling** tab, edit the control's [styling properties](#).
4. Click **Preview** to try out using the control.

Configure Number Fields

Add number controls to your web form. Number controls allow users to enter decimal numbers. By default, number controls display 0 but you can set this to any decimal number using the **Default Value** field in the **General** tab.

To configure a number control:

1. From the Basic Palette, drag and drop a **Number** control onto the canvas.
2. Select the control and edit its properties on the Properties pane's **General** tab.

| Field | Description |
|----------------------------------|---|
| Name | An internal identifier that you will use to identify the control. |
| Binding | Defines a link between the control and a data attribute. |
| Computed Value | Set this property to apply a computation to the control. See Creating Computed Controls . |
| Default Value | Sets a value to display to users when the form loads. |
| Placeholder | Hint text that describes the expected value. This hint text will display in the control before users enter a value. |
| Max | Sets a maximum value that users can enter into the control. |
| Min | Sets a minimum value that users can enter into the control |
| Show increment/decrement buttons | Select this check box to display the up and down arrow buttons used to increment or decrement the control value. |

| Field | Description |
|---|--|
| Step | <p>Specifies a value based on which the number will increase or decrease when users increment or decrement the value with the up or down arrow.</p> <p>For example, if the step value is set to 4, and the initial value in the number field is 0, then when users increment the value for the first time, the value will be 4. When users increment the value the second time, it will be 8 and so on.</p> <p>By default the step value is set to 1. Step must be always greater than 0 and you can't enter a value lower than 1.</p> |
| <div style="border-left: 2px solid #0070C0; border-right: 2px solid #0070C0; border-bottom: 2px solid #0070C0; padding: 10px; background-color: #E6F2FF;"> <p> Note:</p> <p>This property is displayed only when you select the Show increment/decrement buttons check box.</p> </div> | |
| Events | <p>Specifies events that trigger actions and conditions. You can customize the control's behavior by configuring events. See Add Dynamic Behavior to a Form.</p> |

3. On the **Styling** tab, edit the control's [styling properties](#).
4. Click **Preview** to try out using the control.

Configure Date and Time Fields

A Date Time control allow users to enter date and time together. Optionally, add a Date and a Time control to your web form to allow users to enter date and time separately. The Date Time, Date and Time controls are available in the Basic Palette.

To configure a Date Time control:

1. From the Basic Palette, drag and drop a **Date Time** control onto the canvas.
2. Select the control and edit its properties on the Properties pane's **General** tab.

| Field | Description |
|----------------|--|
| Name | An internal identifier that you will use to identify the control. |
| Binding | Defines a link between the control and a data attribute. |
| Computed Value | Set this property to apply a computation to the control. See Creating Computed Controls . |
| Placeholder | Hint text that describes the expected value. This hint text will display in the control before users enter a value. |
| Help | Help text to display to users when they hover over or click the control's help icon. |
| Default Value | Sets the default date and time that appears in the control when the form loads and the current control value is empty. |
| Max Time | Sets a maximum date and time users can enter in the control. |
| Min Time | Sets a minimum date and time users can enter in the control. |
| Format | Specifies a date format for the control. You can select from the available date formats such as yy-MM-dd, yyyy-MM-dd, MM/dd/yy, and so on. |

| Field | Description |
|-----------|--|
| Time Step | Sets the value on which the time will increase when users click the control's clock icon. For example, if the time step is set to 30 minutes and the time on the Date Time control is 12:00:00 pm, then when users click the control's clock for the first time, the time changes to 12:30:00 pm, when users click it for the second time, the time changes 01:00:00 pm, and so on. |
| Events | Specifies events that trigger actions and conditions. You can customize the control's behavior by configuring events. See Add Dynamic Behavior to a Form . |

3. On the **Styling** tab, edit the control's [styling properties](#).
4. Click **Preview** to try out using the control.

 **Note:**

When a user enters data into a Date Time control, the timezone information of the user is saved (in UTC). When other users access this data, the date-time information is displayed in timezones specific to them. This timezone conversion capability is available only in the Date Time control and not in the separate Date and Time controls.

Configure Email Fields

Add an email control to allow users to enter a valid email address to your web form.

1. From the Basic Palette, drag and drop an email control onto the canvas.
2. Select the control and edit its properties on the Properties pane's **General** tab.

| Field | Description |
|----------------|--|
| Name | An internal identifier that you will use to identify the control. |
| Binding | Defines a link between the control and a data attribute. |
| Computed Value | Set this property to apply a computation to the control. See Creating Computed Controls . |
| Default Value | Specifies an email to display to users when the form loads and the current control value is empty. You must ensure that you enter a valid email format in the field, otherwise you will get a message indicating that the email format entered isn't valid. |
| Placeholder | Hint text that describes the expected email format. This hint text will display in the control before users enter a value. |
| Hint | Useful hint text that displays to users when they select the control. |
| Help | Help text to display to users when they hover over or click the control's help icon. |
| Max Length | Sets the maximum number of characters users can enter before the @ symbol for the email. |
| Min Length | Sets the minimum number of characters users can enter before the @ symbol for the email. |

| Field | Description |
|--------|--|
| Events | Specifies events that trigger actions and conditions. You can customize the control's behavior by configuring events. See Add Dynamic Behavior to a Form . |

3. On the **Styling** tab, edit the control's [styling properties](#).
4. Click **Preview** to try out using the control.

Configure Web Address URL Fields

Add a URL control to allow users to enter a web address URL to your web form.

1. From the Basic Palette, drag and drop a **URL** control onto the canvas.
2. Select the control and edit its properties on the Properties pane's **General** tab.

| Field | Description |
|----------------|--|
| Name | An internal identifier that you will use to identify the control. |
| Binding | Defines a link between the control and a data attribute. |
| Computed Value | Set this property to apply a computation to the control. See Creating Computed Controls . |
| Default Value | Sets a value to display to users when the form loads and the current control value is empty. You must ensure that you enter a correct web address URL format in the field, otherwise you will get a message indicating that the URL format entered isn't valid. |
| Placeholder | Hint text that describes the expected value. This hint text will display in the control before users enter a value. |
| Hint | Useful hint text to display to users when they select the control. |
| Help | Help text to display to users when they hover over or click the control's help icon. |
| Max Length | Specifies the maximum number of characters users can enter into the control. |
| Min Length | Specifies the minimum number of characters users must enter into the control. |
| Events | Specifies events that trigger actions and conditions. You can customize the control's behavior by configuring events. See Add Dynamic Behavior to a Form . |

3. On the **Styling** tab, edit the control's [styling properties](#).
4. Click **Preview** to try out using the control.

Configure Message Fields

Use a message control to allow users to enter a simple message to your web form.

1. From the Basic Palette, drag and drop a **Message** control onto the canvas.
2. Select the control and edit its properties on the Properties pane's **General** tab.

| Field | Description |
|----------------|--|
| Name | An internal identifier that you will use to identify the control. |
| Binding | Defines a link between the control and a data attribute. |
| Computed Value | Set this property to apply a computation to the control. See Creating Computed Controls . |
| Default Text | The default text message that will appear to users when the form loads. You can use inline HTML tags to format the default text. Before using, you have to select Inline HTML from the Type drop-down list. |
| Type | Sets the style and format in which the message displays. For example, as a bold heading or as a paragraph text. Note that predefined message types such as Error, Info, Success, and Warning are available for selection. These message types have their unique style and format, so that the user can easily identify if the message is an error, info, success, or warning. Select Inline HTML if you want to use inline HTML tags to format the default text message that displays when the form loads. |
| Events | Specifies events that trigger actions and conditions. You can customize the control's behavior by configuring events. See Add Dynamic Behavior to a Form . |

3. On the **Styling** tab, edit the control's [styling properties](#).
4. Click **Preview** to try out using the control.

Configure Links

Use the link control to insert a URL into a form. You can specify the value for the link URL or configure the URL value to change dynamically based on the payload. For example, a URL link that contains an order item could change based on the order ID, which could be in the payload.

1. From the Basic Palette, drag and drop a **Link** control onto the canvas.
2. Select the control and edit its properties on the Properties pane's **General** tab.

| Field | Description |
|--------|---|
| Name | Defines an internal name for the control. It's an internal identifier that you'll use to identify the control. |
| Label | Specifies the control name that displays to a user. |
| Static | Select this option to use the value in the Default Label field as the control name when the form loads. This is the default selection. |

| Field | Description |
|----------------|---|
| Dynamic | Select this option to assign the control name dynamically when the form loads. If you select this option, you must create a data attribute in the Data pane and link it to the control using the Label Binding field. When the form loads, the value of the data attribute is fetched from the payload and assigned as the control name. |
| | <div style="border-left: 2px solid #0070c0; border-right: 2px solid #0070c0; border-bottom: 2px solid #0070c0; padding: 10px; background-color: #e6f2ff;"> <p> Note:</p> <p>If the attribute value is not available in the payload, then the value in the Default Label field is used as the control name.</p> </div> |
| Label Binding | Defines a link between the control's label and a data attribute. Specify an attribute for this field by selecting an option from the autocomplete list or entering a valid binding. |
| Default Label | Sets a label to display to users when the form loads. The value in this field is used as the control name in the following contexts: <ul style="list-style-type: none"> When you select the Static option in the Label field. When you select Dynamic option in the Label field but the binding value is not available. |
| Value Binding | Defines a link between the control and a data attribute. Data attribute bound to the control, either automatically when Auto Binding is enabled or manually using autocompletion. |
| Computed Value | Set this property to apply a computation to the control. See Creating Computed Controls . |
| Default Value | Sets a value to display to users when the form loads and the current control value is empty. |
| Open | Specifies whether you want the link to open in the current tab or in a new tab. |
| Anchor | Select this check box to enable linking to specific controls in the current form. When you select this check box, the Default Value field changes to a drop-down menu that lists names of all controls present in the form. Select a control name to which you want to link. All Basic Palette, Money, Phone, Image, and Video controls, if present in the current form, appear in the Default Value field. However, controls within repeatable sections, tables, or imported from other forms (unless detached) do not appear in the drop-down list. |
| | <div style="border-left: 2px solid #0070c0; border-right: 2px solid #0070c0; border-bottom: 2px solid #0070c0; padding: 10px; background-color: #e6f2ff;"> <p> Note:</p> <p>If the URL value for the link is set from binding or events, the value selected in the Default Value field is overridden.</p> </div> |
| Hide | Select this check box to hide the control. For example, you might hide a control by default, but configure another control that when selected triggers an event that displays the hidden control. |
| Events | Specifies events that trigger actions and conditions. You can customize the control's behavior by configuring events. See Add Dynamic Behavior to a Form . |

3. On the **Styling** tab, edit the control's [styling properties](#).

4. Click **Preview** to try out using the control.

Configure Simple Text Fields

Use a simple text control to specify a label without binding for a form field.

To configure a simple text control:

1. From the Basic Palette, drag the **Simple Text** control onto the canvas.
2. Select the control and edit its properties on the Properties pane's **General** tab.

| Field | Description |
|--------|--|
| Name | An internal identifier that you will use to identify the control. |
| Text | Specifies the simple text value that displays as the field's label. |
| Hide | Select this check box to hide the control. |
| Events | Specifies events that trigger actions. You can change the simple text value of the control by using an action within Events. |

3. On the **Styling** tab, edit the control's [styling properties](#).
4. Click **Preview** to try out using the control.

Configure Advanced Controls

The Advanced Palette contains a set of advanced controls that you can drag onto the canvas. When you select a control on the canvas, the **General** tab and the **Styling** tab become available in the Properties pane so you can configure settings specific to the selected control. An asterisk (*) indicates a required property.

Topics:

- [Configure Currency \(Money\) Fields](#)
- [Configure Phone Number Fields](#)
- [Include Images](#)
- [Include Videos](#)
- [Configure Identity Browser Fields](#)
- [Place Controls in Panels, Sections, or Tabs](#)
- [Format the Title and Description of a Panel](#)
- [Indent Sections](#)
- [Configure Static and Dynamic List of Values Fields](#)
- [Configure Repeatable Sections](#)
- [Configure Tables](#)
- [Configure Rich Text Editor Fields](#)
- [Configure Train Controls](#)

- [Configure Divider Controls](#)

Configure Currency (Money) Fields

Add a money control to your web form to allow users to enter money amounts (USD, EUR, JPY, GBP, and INR). By default, the currency type is set to **USD**. The user can change the currency type using the **Currency** property on the **General** tab. The corresponding currency symbol displays next to the currency amount. Users can use commas or decimal point to enter different amounts into the control, such as £ 30,700.00. If users don't specify commas or decimal point, the web form automatically displays these symbols. The web form rounds all currency amounts to two decimal places.

To configure a money control:

1. From the Advanced Palette, drag the **Money** control onto the canvas.
2. Select the control and edit its properties on the Properties pane's **General** tab.

| Field | Description |
|----------------------------------|--|
| Name | An internal identifier that you will use to identify the control. |
| Binding | Defines a link between the control and a data attribute. |
| Computed Value | Set this property to apply a computation to the control. See Creating Computed Controls . |
| Currency | Allows you to change the currency type. |
| Placeholder | Text that will appear in the control until any text is entered. If text is removed from the control, the text specified in this field reappears. |
| Hint | Hint text that will display to users when a user clicks into the control. |
| Help | Help text that will appear for the control. If you specify help text, a help icon appears next to the name of the control and displays the help text when a user hovers over it or clicks it. |
| Min | Specify the minimum amount that users need to enter into the control. |
| Max | Specify the maximum amount that users can enter into the control. |
| Show increment/decrement buttons | Select this check box to display the up and down arrow buttons used to increment or decrement the control value. |
| Step | Specify a step value based on which the amount will be incremented or decremented correspondingly when a user increments or decrements the amount in the control. By default, the step value in this field is set to 1. For example, if the step value specified is 3 and the initial amount is \$ 0.00, then, when a user increments the amount in the control for the first time, the amount is updated to \$ 3.00. When the user increments the amount again for the second time, the amount is updated to \$ 6.00 and so on. |

Note:

This property is displayed only when you select the **Show increment/decrement buttons** check box.

| Field | Description |
|----------------------------------|--|
| Show increment/decrement buttons | Select this check box to display the up and down arrow buttons used to increment or decrement the control value. |
| Events | Allows you to configure events for the control. You can assign actions or If/Then/Else conditions to the control based on a selected event. See Add Dynamic Behavior to a Form . |

3. On the **Styling** tab, edit the control's [styling properties](#).
4. Click **Preview** to try out using the control.

Configure Phone Number Fields

Add a phone control to your web form to allow users to enter phone numbers in International or US formats. The phone control uses the US format by default and displays the expected phone number pattern (xxx-xxx-xxxx). You can change the format using the **Format** property on the **General** tab.

To configure a phone control:

1. From the Advanced Palette, drag the **Phone** control onto the canvas.
2. Select the control and edit its properties on the Properties pane's **General** tab.

| Field | Description |
|----------------|--|
| Name | An internal identifier that you will use to identify the control. |
| Binding | Defines a link between the control and a data attribute. |
| Computed Value | Set this property to apply a computation to the control. See Creating Computed Controls . |
| Format | Specify a phone number format. By default, the US format is selected. |
| Placeholder | Text that will appear in the control until any text is entered. If text is removed from the control, the text specified in this field reappears. |
| Hint | Hint text that will display to users when a user clicks into the control. |
| Help | Help text that will appear for the control. If you specify help text, a help icon appears next to the name of the control and displays the help text when a user hovers over it or clicks it. |
| Max Length | Defines the maximum number of characters a user can enter into the control. |
| Min Length | Defines the minimum number of characters a user must enter into the control. Leave this field blank if you don't want to define the minimum length. |
| Events | Allows you to configure events for the control. You can assign actions or If/Then/Else conditions to the control based on a selected event. See Add Dynamic Behavior to a Form . |

3. On the **Styling** tab, edit the control's [styling properties](#).
4. Click **Preview** to try out using the control.

Include Images

Use an image control to include an image in your web form. Under image source properties, you can identify the image using an absolute or a relative image URL or a Base64 format string.

To configure an image control:

1. From the Advanced Palette, drag the **Image** control onto the canvas.
2. Select the control and edit its properties on the Properties pane's **General** tab.

| Field | Description |
|------------------|---|
| Name | An internal identifier that you will use to identify the control. |
| Binding | Defines a link between the control and a data attribute. |
| Computed Value | Set this property to apply a computation to the control. See Creating Computed Controls . |
| From URL | To specify either an absolute or relative URL, select this option and enter the image URL in the field below it. |
| From Base64 | To specify an image converted to base64 format, select this option, then identify the image's format in the Image Format field and its binding in the Base64 Binding field. Base64 images can be viewed in preview mode only. |
| Alternative text | Enter text to display if the image can't be loaded. |
| Events | Allows you to configure events for the control. You can assign actions or If/Then/Else conditions to the control based on a selected event. See Add Dynamic Behavior to a Form . |

3. On the **Styling** tab, edit the control's [styling properties](#).
4. Click **Preview** to try out using the control.

Upload and Preview a Base64 Image

You can embed Base64 encoded images in forms. They load quickly and are useful for email signatures.

1. Drag and drop an Image control onto the canvas and select it.
2. On the General tab, specify Base64 settings in the **Image Source** fields:
 - a. Select **From Base64**.
 - b. In the **Image Format** field, specify the format you chose when converting the image to base64.
 - c. In the **Base64 Binding** field, enter a name for the image (for example, a data string called *oracle*).
3. If the encoded string of your base64 image starts with `data:image/jpeg;base64`, remove it.

The base64 input you need begins with: `/9j/4AAQ....`, not `"data:image/jpg;base64,/9j/4AAQ...."`

4. Click **Preview** to try out using the control.

Configure Identity Browser Fields

Add an Identity Browser control to allow users to search and select individuals to be notified or assigned tasks downstream in the process.

Use identities to search for users, groups, or roles. You can add single or multiple entries to the Identity Browser based on the configuration you select. Use options in the search filter to restrict the scope of the search.

To configure an Identity Browser control:

1. From the Advanced Palette, drag the **Identity Browser** control onto the canvas.
2. Select the control and edit its properties on the Properties pane's **General** tab.

| Field | Description |
|---------------|--|
| Name | An internal identifier that you will use to identify the control. |
| Label | Specifies the control name that displays to a user. |
| Binding | Defines a link between the control and a data attribute. |
| Placeholder | Text that will appear in the control until any text is entered. If text is removed from the control, the text specified in this field reappears. |
| Hint | Hint text that will display to users when a user clicks into the control. |
| Help | Help text that will appear for the control. If you specify help text, a help icon appears next to the name of the control and displays the help text when a user hovers over it or clicks it. |
| Default Scope | Specifies the default scope for the identity search. Available options in the drop-down menu are All , User , Group , and Role . The value of this field is set to User by default. |
| Required | Set this property to require users to complete the control in order to successfully submit the form. |
| Multiple | Set this property to allow multiple entries to the control. |
| Disabled | Set this property to display the control as inactive. |
| Hide | Set this property to hide the control. For example, you might hide a control by default, but configure another control that when selected triggers an event that displays the hidden control. |
| Scope Filter | Set this property to allow filtering of results in the control. |
| Auto Focus | Set this property to automatically select the control when the web form loads. |

| Field | Description |
|--------|--|
| Events | Allows you to configure events for the control. You can assign actions or If/Then/Else conditions to the control based on a selected event. See Add Dynamic Behavior to a Form . |

 **Note:**

Except for assigning a constant value, you cannot trigger any changes to the Identity Browser using other controls. Actions such as assigning function results, connector call values, or data from the payload or another control are not applicable in the context of Identity Browser.

3. On the **Styling** tab, edit the control's [styling properties](#).
4. Click **Preview** to try out using the control.

Retrieving a List of User Identities

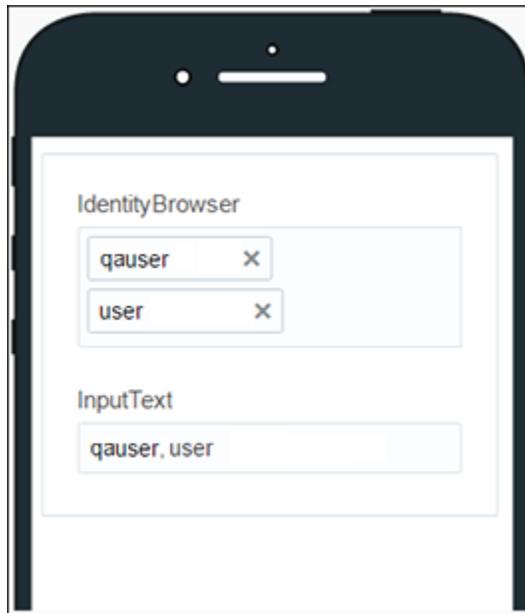
An identity object contains several fields, such as title, ID (user identity), type, e-mail, first name, last name, and contact number.

Using a combination of Identity Browser and Text controls, you can retrieve a comma-separated list of user identities (IDs), which can be used by other process elements.

To fetch a list of IDs:

1. Add an **Identity Browser** control to the form canvas.
2. Edit the control's properties as required. Make sure to select the **Multiple** checkbox to allow multiple entries to the control.
3. Drag the **Input text** control onto the canvas. You may also use the **Text Area** control.
4. Configure an event to populate data in the **Input text** control field based on the data entered into the **Identity Browser** control. See [Add Dynamic Behavior to a Form](#) for details on creating events.
 - a. Select **Identity Browser**, and in the Properties pane's **General** tab, scroll down until you see **Events**.
 - b. Click **Add**  to define an event.
 - c. Select an event option, for example, **On Change**; this event is activated when an end user changes the control's value.
 - d. After selecting the event option, click the editing icon next to the events drop-down menu to specify the action. An event window appears.
 - e. In the event window, click **+Action** to add an action.
A solid red **Action** indicator appears. Below it, configure the action.
 - f. In the **Control Name** field, select the control the action will affect, in this case select **InputText**.
 - g. In the **Action** field, select **Value**.
 - h. In the **Type** field select **Control**, and in the resulting **Control Name** field, select **IdentityBrowser**.

- i. Finally, select **Identities** in the **Property** field. This fetches the data in the ID fields of entries in the **Identity Browser** control to the **Input text** field.
 - j. Click **OK**, then **Save**.
5. Click **Preview** to test the controls.
 6. On the **Preview** page, search for and select multiple identity objects in the **Identity Browser** control.
 7. The **Input text** field is populated automatically with the list of user identities, that is, the data in the ID fields of these entries.



Place Controls in Panels, Sections, or Tabs

You can add panels, sections, and tabs to your web form and use them to group multiple controls under a single control.

To configure a panel, section, or tab control:

1. Expand the Advanced Palette.
2. Drag and drop a **Panel**, **Section**, or **Tab** control onto the canvas.
 - A text *Drop Elements to this panel!* indicates that you can drop controls into the panel.
 - A text *Drop Elements to this section!* indicates that you can drop controls into the section.
 - A text *Drop Elements to this tab!* indicates that you can drop controls into the tab. By default, the tab control displays one tab (Tab1). Click **Add tab** to insert additional tabs into the control.
3. Select the control, and then edit its properties on the **General** tab in the Properties pane.

| Field | Description |
|-------|---|
| Name | An internal identifier that you will use to identify the control. |

| Field | Description |
|--------------|---|
| Label | The title of the panel, section, or tab that the user will see in the form. |
| Description | For a panel control, provides additional information or instructions for the user. |
| Type | For a section control, sets the style and format in which the section label displays. |
| Hide | Set this property to hide the control. |
| Read Only | Set this property to make the (panel, section, or tab) control read-only. Note that when this property is set, all the controls inside a panel, section, or tab control become read-only (irrespective of the individual property of each of the controls). The user can view but is not allowed to edit the control. |
| Lazy Loading | For a section control, implements lazy loading for the controls inside it. The lazy loading controls are collapsed by default. |
| Expanded | For a section control, specifies if the section control is expanded when the form loads. By default, this field is checked. |
| Event | Specifies events that trigger actions and conditions. You can customize the control's behavior by configuring events. See Add Dynamic Behavior to a Form . |

4. On the Styling tab, edit the control's [styling properties](#).
5. Drag and drop individual controls from the Basic Palette or the Advanced Palette into the panel, section, or tab control.
See [Position Controls on Forms](#).
6. Configure general and styling properties for the controls inside your panel, section, or tab control.
See [Configure Basic Controls](#) and [Configure Advanced Controls](#).

 **Note:**

If you delete the panel, section, or tab control, all the controls grouped under it get deleted.

7. Click **Preview** to try out using the control.

 **Note:**

If the data in a field within a section or tab is invalid, a validation error is displayed for the field and the entire section or tab is marked invalid.

Format the Title and Description of a Panel

Use a theme to format the title and description for a panel in your web form.

To select a theme for a panel:

1. From the Advanced Palette, drag and drop the **Panel** control onto the canvas.
2. Select the **Panel** control on the canvas.

3. In the Properties pane, select the **General** tab.
 - In the **Label** field, enter a title for the panel. The user will see this title on the form.
 - In the **Description** field, add information or instructions for the user.
4. In the Properties pane, select the **Styling** tab.
5. In the **Theme** field, select **Standard**.

The Standard theme automatically:

- Increases the font size of the label to 24 pixels (font-size: 24px)
- Applies the bold style to the label (font-weight: bold)

Selecting a theme overrides the value defined in the **Label Size** field on the Styling tab. If you want to use the Label Size field to format the panel label, then make sure the **Theme** field is set to **None**.

You can use the Label Color field to specify a color for the label and the descriptive text in the panel.

Indent Sections

Use a theme to indent the sections in your web form.

To indent one or more sections in a web form:

1. From the Advanced Palette, drag and drop the **Section** control onto the canvas.
2. Select the **Section** control on the canvas.
3. In the **Properties** pane, select the **Styling** tab.
4. In the **Theme** field, select **Indent**.

Configure Repeatable Sections

Use repeatable section controls to display multiple copies of a set of controls in your web form. You can use repeatable section controls to create dynamic content for your web form.

To configure a repeatable section control:

1. From the Advanced Palette, drag and drop a repeatable section control onto the canvas.
2. Select the control and edit its properties on the Properties pane's **General** tab.

| Field | Description |
|---------------------------|--|
| Name | An internal identifier that you will use to identify the control. |
| Label | Specifies the control name that displays to a user. |
| Binding | Defines a link between the control and a data attribute. |
| Label Binding | Allows you to define a dynamic label. To do this, you can use a data attribute listed in the Data definition pane. |
| Users can Add/Remove Rows | Allows users to add or remove rows to the repeatable section. |

| Field | Description |
|-------------------------|---|
| Multiple Selection | <p>Allows users to select multiple rows of the repeatable section.</p> <p>While using a repeatable section with multiple-selection enabled to define Dynamic Behaviors or Computed Values, the options available (specific to multiple selection) in the Event or Computed Value window under the Which? field are listed here:</p> <ul style="list-style-type: none"> • For Each Selected: Use to retrieve a value from another control to apply to each selected row of a repeatable section. • All Selected: Use to retrieve values from all selected rows of a repeatable section to apply to another control. <p>Similarly, in a repeatable section with two controls in a row, you can use these options to apply values from one control to another on occurrence of an event. In this case, when you chose to apply an action to each selected row (by selecting the For Each Selected option), an additional option is available to chose the value source:</p> <ul style="list-style-type: none"> • Current Iteration Row: Use to retrieve the value from one control of the current selected row and apply to another control in the same row. |
| Use Data from Connector | <p>Allows you to populate the repeatable section from a REST connector defined for the application.</p> <p>Specify the connector settings in the Connector, Resource and Operation fields and map response settings. See Populate Controls Using REST Calls.</p> |
| Events | <p>Specifies events that trigger actions and conditions. You can customize the repeatable section control's behavior by configuring events. See Add Dynamic Behavior to a Form.</p> |

 **Note:**

While configuring events for a repeatable section, you can now add a condition based on number of rows present within the section using the **Row Count** property.

3. On the **Styling** tab, edit the control's [styling properties](#).
4. Drag and drop individual controls from the Basic or Advanced Palette into the repeatable section control.
See [Position Controls on Forms](#).
5. Configure general and styling properties for the controls inside your repeatable section control.
See [Configure Basic Controls](#) and [Configure Advanced Controls](#).

 **Note:**

If you delete the repeatable section control, all the controls grouped under it get deleted.

6. Click **Preview** to try out using the control.

Some Useful Event Actions for Repeatable Sections

This section lists a few useful actions that you can apply to a repeatable section through the event window. To create an event action, see [Specify Actions](#).

- To retrieve a value from another control and apply it to each row of the repeatable section, use the **For Each** option under the **Which?** field.
- In a repeatable section with two controls in a row, to copy values from one control to another (for each row) on occurrence of an event, use the **For Each** option along with an additional option called **Current Iteration Row** that is available to choose the value source. The **Current Iteration Row** option allows you to retrieve the value from one control of a row and apply to another control in the same row. The following figure shows an event action configuration for a repeatable section with two controls in a row. The value of **control2** is copied into **control1** for each row of the repeatable section on occurrence of the specified event:

Configure Tables

Use table controls to group multiple controls in a grid pattern into your web form. You can use table controls to create dynamic content for your web form.

To configure a table control:

1. From the Advanced Palette, drag and drop a table control onto the canvas.
By default the table control contains one column.
2. Select the control and edit its properties on the Properties pane's **General** tab.

| Field | Description |
|---------|---|
| Name | An internal identifier that you will use to identify the control. |
| Binding | Defines a link between the control and a data attribute. |

| Field | Description |
|---------------------------|--|
| Columns | In the Columns field, click Add  to add columns. Edit the labels of each column in their respective label fields. To hide a particular column from users, click Hide  beside its label field; you may require to hide a column initially and show it on occurrence of an event, see Some Useful Event Actions for Tables . Click the Delete icon to delete a column. |
| Users can Add/Remove Rows | Allows users to add or remove rows to the table. |
| Multiple Selection | Allows users to select multiple rows of the table. While using a table with multiple-selection enabled to define Dynamic Behaviors or Computed Values , the options available (specific to multiple selection) in the Event or Computed Value window under the Which? field are listed here: <ul style="list-style-type: none"> • For Each Selected: Use to retrieve a value from another control to apply to each selected row of a table. • All Selected: Use to retrieve values from all selected rows of a table to apply to another control. Similarly, in a table with two columns, you can use these options to apply values from one column to another on occurrence of an event. In this case, when you chose to apply an action to each selected row (by selecting the For Each Selected option), an additional option is available to chose the value source: <ul style="list-style-type: none"> • Current Iteration Row: Use to retrieve the value from one cell of the current selected row and apply to another cell in the same row. |
| Use Data from Connector | Allows you to populate the table from a REST connector defined for the application. Specify the connector settings in the Connector, Resource and Operation fields and map response settings. See Populate Controls Using REST Calls . |
| Events | Specifies events that trigger actions and conditions. You can customize the control's behavior by configuring events. See Add Dynamic Behavior to a Form . |

 **Note:**

While configuring events for a table, you can now add a condition based on number of rows present within the table using the **Row Count** property.

- On the **Styling** tab, edit the control's [styling properties](#). You can specify column width in several ways:
 - Leave the **Automatic column size** checkbox selected so the table's columns automatically resize to fit the device on which the form is viewed.
 - Deselect the **Automatic column size** checkbox and specify column sizes for different device sizes by entering a number from 1 to 12 in the column size fields that display for different devices.
 - In the **Table Columns Width** fields, specify an absolute width for each column such as 2in (inches), 5cm (centimeters), 100px (pixels), or 25% (percent). If the column

widths together exceed the table width, a scroll bar displays, unless columns are defined with percentages. If percentages are specified but exceed 100%, column widths are displayed proportionally across the table.

4. Drag and drop individual controls from the Basic or Advanced Palette into the columns.

Each column can have one control. See [Position Controls on Forms](#).

5. Configure general and styling properties for the controls inside your table control. See [Configure Basic Controls](#) and [Configure Advanced Controls](#).

 **Note:**

If you delete the table control, all the controls grouped under it get deleted.

6. Click **Preview** to try out using the control.

Some Useful Event Actions for Tables

This section lists a few useful actions that you can apply to a table through the event window. To create an event action, see [Specify Actions](#).

- To retrieve a value from another control and apply it to each row of the table, use the **For Each** option under the **Which?** field.
- In a table with two columns, to copy values from one column to another (for each row) on occurrence of an event, use the **For Each** option along with an additional option called **Current Iteration Row** that is available to choose the value source. The **Current Iteration Row** option allows you to retrieve the value from one cell of a row and apply to another cell in the same row. The following figure shows an event action configuration for a table with two columns. The value of **control2** (the control in table column 2) is copied into **control1** (the control in table column 1) for each row of the table on occurrence of the specified event:

- To hide a column within a table on occurrence of an event, select **Self** under the **Which?** field, choose **Hide Column** under **Action**, and specify the column to hide. The following figure shows an event action configuration to hide a table column:

- Similarly, to show a previously hidden column—hidden either through Properties pane's **General** tab or through an event action, select **Self** under the **Which?** field, choose **Show Column** under **Action**, and specify the column to show.

Configure Rich Text Editor Fields

Add a rich text editor control to allow users to enter various types of content, like multimedia, web links, formatted texts, and more into the web form.

This control supports most of the word processing features found in desktop applications such as Microsoft Word. Use the standard editing buttons to insert content or to apply styles to the selected content.

To configure a rich text editor control:

1. From the Advanced Palette, drag the **Rich Text Editor** control onto the canvas.
2. Select the control and edit its properties on the Properties pane's **General** tab.

| Field | Description |
|----------------|--|
| Name | An internal identifier that you will use to identify the control. |
| Label | Specifies the control name that displays to a user. |
| Binding | Defines a link between the control and a data attribute. |
| Computed Value | Set this property to apply a computation to the control. See Creating Computed Controls . |
| Default Value | Sets a value to display to users when the form loads. |
| Help | Help text that will appear for the control. If you specify help text, a help icon appears next to the name of the control and displays the help text when a user hovers over it or clicks it. |
| Read Only | Set this property to make the control read-only, that is, to display the control (and its contents if any) but not allow users to edit it. |
| Hide | Set this property to hide the control. For example, you might hide a control by default, but configure another control that, when selected, triggers an event that displays the hidden control. |
| Events | Allows you to configure events for the control. You can assign actions or If/Then/Else conditions to the control based on a selected event. See Add Dynamic Behavior to a Form . |

3. On the **Styling** tab, edit the control's [styling properties](#).
4. Click **Preview** to try out using the control.

**Note:**

Using computed values or events, you can populate another control (for example, text area) with the contents of a rich text editor control along with all the HTML tags.

Configure Train Controls

If you have a multistep web form activity that users should approach in a particular sequence, then add a Train control to guide them through this activity. The Train control displays the number of steps a user must complete before submitting the form, and it also indicates the user's current place within a multistep activity.

In a multistep form (form with a set of presentations), you can represent each step using a train stop. In other words, you can map each presentation to a stop. Users can navigate between steps by clicking on the train stops; the current stop of the user is highlighted in blue and the visited stops bear a check mark. However, to display a different presentation for each stop, you must define an event with Change Presentation actions for the Train control. See [Setting Up Navigation for a Multistep Web Form](#).

To configure a Train control:

1. From the Advanced Palette, drag the **Train** control onto the canvas.
2. Select the control and edit its properties on the Properties pane's **General** tab.

| Field | Description |
|----------------|--|
| Name | An internal identifier that you will use to identify the control. |
| Label | Specifies the control name that displays to a user. |
| Binding | Defines a link between the control and a data attribute. |
| Computed Value | Set this property to apply a computation to the control. See Creating Computed Controls . |
| Options Source | Select a source (Static, From Data, and Connector). <ul style="list-style-type: none"> • Static: Specify choices using Options Names and Options Values fields. Use Options Names to specify the label to display for an option and use Options Values to specify an internal value for an option. • From Data: In the Options List field, select a list of values options source from the data definitions available in the web form. If you selected a list of complex elements, then, in the Label Binding field, specify a data attribute that will display as the label and in the Value Binding field, specify a data attribute that will be the value. • Connector: Specify a REST connector, a resource, and an operation to use. Specify parameters to pass to the connector and define how the response should be mapped to the control properties. See Populate Controls Using REST Calls. |

| Field | Description |
|---------------|---|
| Default Value | If you selected Static in the Options Source field, then, specify a default option in this field. If you selected From Data or Connector in the Options Source field, then, select either the first or the last value as the default value. |
| Events | Allows you to configure events for the control. You can assign actions or If/Then/Else conditions to the control based on a selected event. See Add Dynamic Behavior to a Form . |

3. On the **Styling** tab, edit the control's [styling properties](#).
4. Click **Preview** to try out using the control.

 **Note:**

- You can add a maximum of 15 stops for your form using the Train control. When the form loads, users can scroll to navigate the stops or the control can be designed to fit into a single view by selecting **Stretch** from the Properties pane's **General** tab.
- To avoid redesigning the Train control each time you make changes (especially, addition or deletion of presentations) to the multistep form, add the control after you've completely configured the web form and all of its steps.

Setting Up Navigation for a Multistep Web Form

Let's set up navigation for a multistep web form using the Train control. To create a web form from scratch, see [Creating Web Forms](#).

In this example, we'll use a simple, two-step web form designed to receive feedback from users. When the form loads, you enter your basic details to begin with and then navigate to the next step to provide your feedback.

Now, let's create and configure forms and controls to suit this example.

1. Create a web form with two presentations, one to obtain basic details of the user and the other to receive feedback. Name these presentations, **BasicDetails** and **Feedback**.
 - a. To the BasicDetails presentation, add three **Input text** controls and rename their labels to **Name**, **Company**, and **Email**.
 - b. To the Feedback presentation, add a **Text Area** control and rename its label to **Enter Feedback**.
 - c. Click **Save**.
2. Now, create another web form, then drag and drop the **Train** control onto the form's canvas. On the **General** tab, rename options names and values for the control as follows:

| Options Names | Options Values |
|------------------------|-----------------------|
| Basic Details Feedback | basicdetails feedback |

3. From the Forms Palette, drag and drop the web form you created previously.
4. Then, select the **Train** control to add an event.
 - a. On the **General** tab, locate **Events** and click **Add**.
 - b. Select **On Change** from the drop-down menu and click **Edit**.
5. As illustrated in the following figure, define If-Else conditions in the event window to switch presentations of the embedded form according to train stop selections.

On Change

If
+ Condition
↑ ↓ 🗑️

| | | |
|--------------------|-------------------------|---------------------|
| Type Control ▼ | Control Name Train ▼ | Property Value ▼ |
| is equal to ▼ | | |
| Type Constant ▼ | Value basicdetails | |

Then
+ Action

Action
🗑️

| | | |
|----------------------------|---------------------------------|--------------------------------|
| Control Name WebForm1 ▼ | Action Change Presentation ▼ | Presentation BasicDetails ▼ |
|----------------------------|---------------------------------|--------------------------------|

Else If
+ Condition
🗑️

| | | |
|--------------------|-------------------------|---------------------|
| Type Control ▼ | Control Name Train ▼ | Property Value ▼ |
| is equal to ▼ | | |
| Type Constant ▼ | Value feedback | |

Then
+ Action

Action
🗑️

| | | |
|----------------------------|---------------------------------|----------------------------|
| Control Name WebForm1 ▼ | Action Change Presentation ▼ | Presentation Feedback ▼ |
|----------------------------|---------------------------------|----------------------------|

Click **OK** to close the window.

6. Click **Preview** to test the form.
 - a. At the first train stop, you are presented with the fields to enter your details.

Feedback Form

1 — 2

Basic Details Feedback

Name
Beckett Sam

Company
XYZ Corp.

Email
Sam@xyz.com

- b. Clicking on the second stop provides you with a space to enter your feedback and submit the form.

Feedback Form

✓ — 2

Basic Details Feedback

Enter Feedback
Thank you for an excellent service.

Configure Divider Controls

Use a divider control to separate related content within your form or to create logical breaks in your form's layout. For example, you might want to break up your form into sections or separate a related group of controls from the rest in the form.

To configure a divider control:

1. From the Advanced Palette, drag the **Divider** control onto the canvas.
2. Select the control.

3. In the Properties pane, select the **General** tab and edit the properties.

| Field | Description |
|--------|--|
| Name | An internal identifier that you use to identify the control. |
| Hide | Set this property to hide the control. For example, you might hide a control by default, but configure another control that, when selected, triggers an event that displays the hidden control. |
| Events | Allows you to configure events for the control. You can assign actions or If/Then/Else conditions to the control based on a selected event. See Add Dynamic Behavior to a Form . |

4. On the **Styling** tab, edit the control's styling properties.
 - To specify the color of the divider control, use the **Stroke Color** property.
 - To specify the width of the divider control, use the **Stroke Width** property. You can enter standard values such as 0.05in, 0.2em, or 5px.
 See [Styling Properties](#).
5. To try out the control, click **Preview**.

Configure Static and Dynamic List of Values Fields

Use checklists, radio button or select controls to create static or dynamic list of values in your web form. Static list of values are defined in the control properties. Dynamic list of values are defined from a collection of available attributes or are based on a REST call.

You can configure static and dynamic list of values for checklists, radio buttons and select controls from **Option Source** in the **General** tab. In **Option Source** specify whether the control's options should come from static values you enter, from a list of values data attribute, or externally from a REST connector created for the application.

To configure static and dynamic list of values (for checklist, radio button and select controls):

1. From the Basic Palette, drag and drop a checklist, radio button or select control onto the canvas.
2. Select the control and configure its static list of values on the Properties pane **General** tab.
 - a. Go to **Option Source** and click **Static**.
 - b. In the **Option Names** field, enter a label to display for each option.
 - c. In the **Option Values** field, enter an internal value for each option.
 - d. Specify a default value from the option values available in the **Default Value** field.
 - e. Select an option in the **Autofocus** field to make that option the selected option when the form loads.
3. Optionally, configure dynamic list of values for the checklist, radio button or select control. There are two options to configure dynamic list of values.
 - Specify that the control's options should come from a list of value data attributes defined in the Data definition pane.
 - a. Click **From Data**.
 - b. Select the list of values options source from available attributes in the **Options List** field.

- c. Specify a default value from the options available in the **Default Value** field.
 - d. Select an option in the **Autofocus** field to make that option the selected option when the form loads.
 - Specify that the control's options should come from a REST connector created for the application.
 - a. Click **Connector**.
 - b. Specify the connector settings in the Connector, Resource and Operation fields and map response settings. See [Populate Controls Using REST Calls](#).
 - c. Select an option in the **Autofocus** field to make that option the selected option when the form loads.
4. Click **Preview** to try out using the control.

Add an Option to a List of Values Field

You can add an option to a list of values field (checklist, radio button, or select) by specifying an action **Append Option** to it. The new option gets added to already present static or dynamic values in the list of values field.

See [Specify Actions](#).

Configure a control in your form such that you can use it for adding an option to a list of values field. For example, you can configure a button control with an On Click event and use it to add an option to a list of values field.

To add an option to a list of values (LOV) control:

1. Drag and drop a control, such as a button control, from the Basic Palette onto your form with the LOV control. Change the label and name of the control, for example **Add**.
2. Configure the control by specifying an event and defining an action for the event.
 - a. Select the control, and scroll down the **General** tab of the Properties pane until you find **Events**. Click  to add and define an event.
 - b. Select **On Click** from the drop-down list and then click the editing icon to specify the action for the event.
 - c. In the On Click event window, click **+Actions** to define an action.
 - d. Specify the **Append Option** action to the LOV control. Select the LOV control from the **Control Name** drop-down list, and then select **Append Option** from the **Action** drop-down list.
 - e. In the **Option Name Type** and **Option Value Type** drop-down lists, select a type, for example **Constant**.
 - f. Enter the value that you want to add to the default options for the LOV control in the **Value** fields.
 - g. Click **OK**.
3. Preview the form and see how it works.

Click **Preview**. When the form loads, view the options in the LOV field.

Click the control for which you configured an On Click event, and view the options in the LOV field again. Note that the value you specified in step 2f gets added to the list of values.

Clear All Options From a List of Values Field

You can delete all options from a list of values field (checklist, radio button, or select) by specifying an action **Clear Options** to it.

See [Specify Actions](#).

Configure a control in your form such that you can use it for clearing all options from a list of values field. For example, you can configure a button control with an On Click event and use it to clear all options from a list of values field.

To delete all options from a list of value (LOV) control:

1. Drag and drop a control, such as a button control, from the Basic Palette onto your form with the LOV control. Change the name and label of the control, for example **Clear**.
2. Configure the control by specifying an event and defining an action for the event.
 - a. Select the control, and scroll down the **General** tab of the Properties pane until you find **Events**. Click  to add and define an event.
 - b. Select **On Click** from the drop-down list and then click the editing icon to specify the action for the event.
 - c. In the On Click event window, click **+Actions** to define an action.
 - d. Specify the **Clear Options** action to the LOV control. Select the LOV control from the **Control Name** drop-down list, and then select **Clear Options** from the **Action** drop-down list.
 - e. Click **OK**.
3. Preview the form and see how it works.

Click **Preview**. When the form loads, view the options in the LOV field. Now click the control for which you configured an On Click event, and view the LOV field again.

Notice that the LOV field is empty and all options have been cleared.

Specify Filters for Controls

You can specify filters for controls that accept list of values (LOV) as input, such as drop-down select, checklist, checkbox, and radio button controls. You can use a filter to selectively extract data from a data source (such as payload or an external connector) and display the filtered data as options of the control.

To specify a filter for an LOV control (for example, drop-down select):

1. Click on the control in the form's canvas, and in the Properties pane's **General** tab, navigate to the Option Source section.
2. Select the data source as either **From Data** or **Connector**. Configure subsequent fields to match your selection.
3. A **Filter** check box appears after you complete configuring the Option Source section.
4. Select the check box, and click the **Edit** button that appears next to it.

5. In the Filter window, specify how the data is to be filtered from the source. You can also use an If block to specify the same.

 **Note:**

If you specify more than one way to filter data from the source, the results are ORed.

6. Click **OK**, then **Save**.

Filtering in Controls – An Example

This section demonstrates filtering in controls using a web form with a drop-down select control.

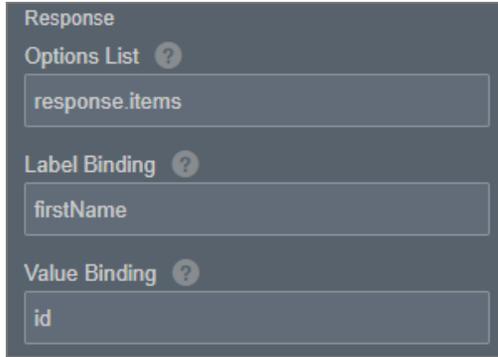
Based on the filtering condition specified for the control, you will be able to see relevant data as options of the control when the form loads.

In this example, we use a previously created REST connector to fetch data from a web server, which contains a list of items. Each item, in this case, is a complex data object that contains details of employees of an organization, such as first name, last name, unique ID, phone number, isActive (this Boolean attribute specifies if the individual is a current employee or not), city, and so on. To create a REST connector from scratch, see [Creating a REST Connector](#).

We use this REST connector as the input to the select control.

Now, let's configure the select control to suit this example.

1. Click on the control, and on the **General** tab, change the **Label** field to *First Name*.
2. Under Options Source, select the **Connector** option and populate the **Resource** and **Operation** fields with respect to the REST connector defined within the application. See [Populating Controls Using REST Calls](#).
3. Under Response, specify the items list to display as options of the control.
 - a. For this particular example, we map the first name attribute to the **Label Binding** field, so that only first names of employees appear as options in the select control. Also, the unique ID attribute is mapped to the **Value Binding** field as follows:



Response

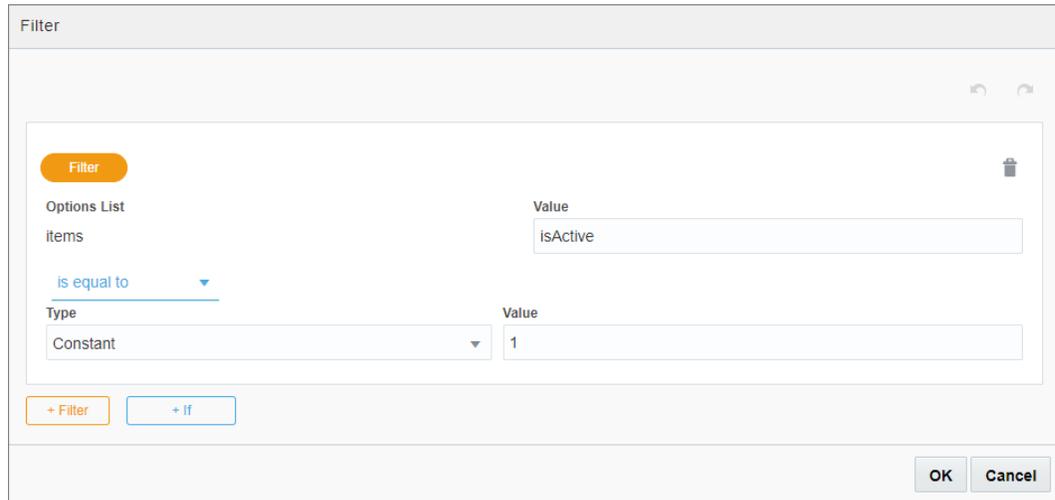
Options List ?
response.items

Label Binding ?
firstName

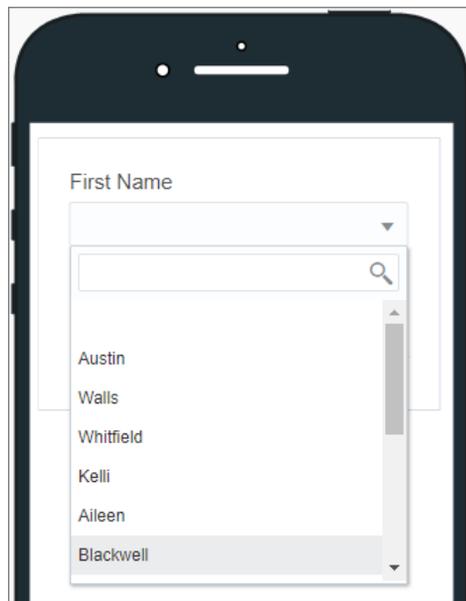
Value Binding ?
id

4. Select the **Filter** check box, and click the **Edit** button that appears next to it.

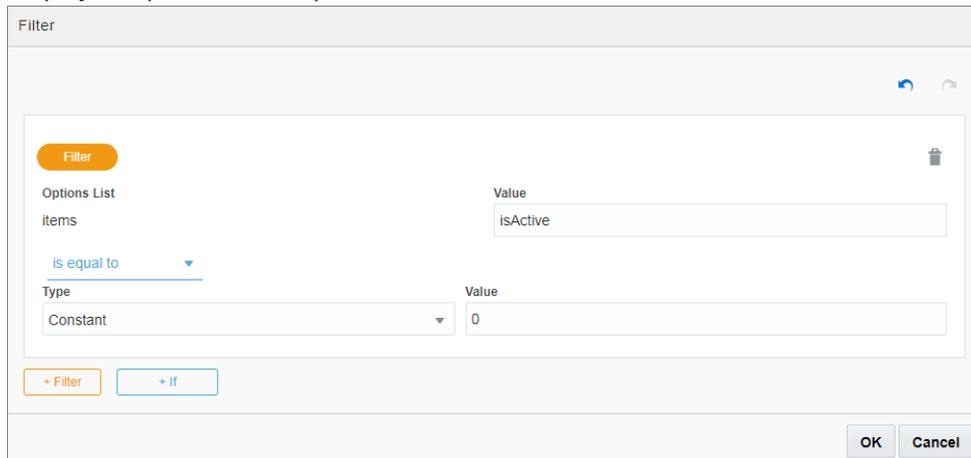
5. In the Filter window, specify a condition to filter specific data from the source. The filter configuration in the following figure extracts only active items (that is, current employees of the organization) from the data source to display as options of the select control:



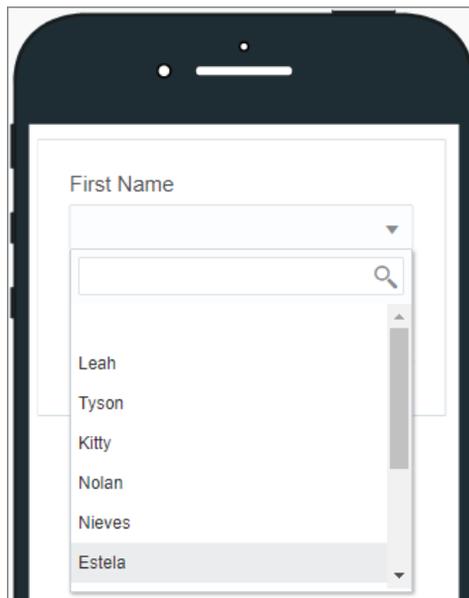
6. When the form loads, the control displays first names of all current employees as follows:



- You can change the filter configuration on the select control to display only former employees (*isActive=false*) as follows:



- The options in the drop-down select change accordingly:



Create Computed Controls

Within a web form, you can define a control as a computed field that uses a data definition, value of an existing control, result of a custom formula, or connector data as its value.

You can define complex formulas using different functions based on data definitions, connector data, and existing controls to compute the control value. All controls that support binding also support computed values.

To convert an editable control to a computed control in a web form:

- Click on the control in the form's canvas and select the **Computed Value** check box on the Properties pane's **General** tab. An **Edit** button appears.

2. Click **Edit** to define how the control's value is calculated.
3. In the Configure Computed Value window, select from the following options in the **Type** field:
 - a. **Constant** lets you specify a value for the selected control. Be sure to specify the corresponding value in the **Value** field.
 - b. **Data Definition** lets you use the data from the payload. When you click the adjacent **Value** field that displays, available items are listed.
 - c. **Control** lets you use the data from a control. Be sure to select the control name.
 - d. **Function** lets you define custom formulas using common operations with strings, values and arrays, as described in [Specifying Functions](#).
 - e. **Connector Data** lets you assign a value from a global connector call.
4. After configuring the value, click **OK**, then **Save**.

 **Note:**

- If a control has a computed value and also has a value assigned through payload, the computed value has the priority.
- If a control has a computed value but is edited, either through user input or event actions, the new value remains. However, when a dependency (data definition, control value, or connector data) is updated, the new computed value is set.

Example of a Computed Control

Let's create a web form to collect sales and costs data through user input and calculate total profits. To create a web form from scratch, see [Creating Web Forms](#).

When the form loads, you should be able to enter sales and costs data in editable controls, and a computed control should calculate the difference between data entered to arrive at total profits.

Now, let's add and configure controls in the web form to suit this example.

1. Add and configure two editable number fields onto the form's canvas.
 - a. From the Basic Palette, drag and drop a **Number** control onto the canvas.
 - b. Drag and drop another **Number** control below the first control.
 - c. Select the first **Number** control. On the **General** tab, change the **Name** field to *TotalSales*, and the **Label** field to *Total Sales (in USD)*.
 - d. Repeat the previous step to change the second control's name to *TotalCosts* and its label to *Total Costs (in USD)*.
2. Add a third number field and configure it as a computed control.
 - a. Drag and drop a **Number** control below the previously added controls.
 - b. Select the control. On the **General** tab, change the **Name** field to *TotalProfits*, and the **Label** field to *Total Profits*. Select the **Computed Value** check box and click the **Edit** button.

- c. In the Configure Computed Value window, define the formula to calculate profits.

Select **Function** in the **Type** field, and choose the **Subtract (-)** function in the **Function** field.

As illustrated in the following figure, specify parameters for additional fields to use values from editable controls.

Configure Computed Value

Type: Function, Function: -, Type: (Control

Control Name: TotalSales, Type: Control, Control Name: TotalCosts

OK Cancel

Click **OK** to close the window.

- 3. Click **Preview** to test the computed control. The following figure shows sample data entered and profit calculated automatically.

Total Sales (in USD)
1,425

Total Costs (in USD)
315

Total Profits
1,110

Localize Web Forms

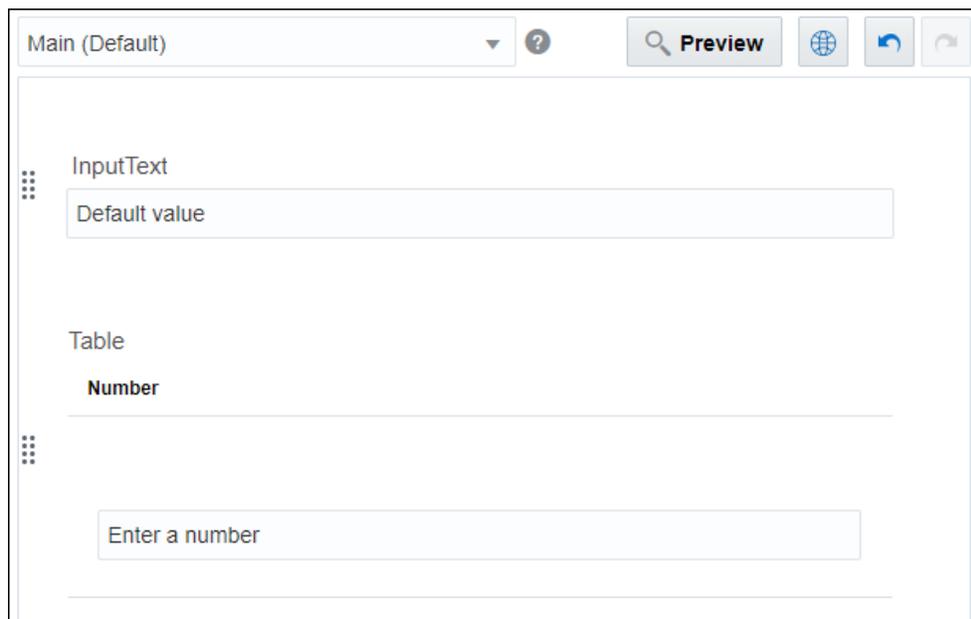
If you have added a language to localize an application's content, you can supply localized strings for control labels and control fields that take constants for input (default value, hint text, and so on) in the forms editor.

When the application is published, users can view the localized version of the form in runtime by choosing their preferred language.

To add a new language to an application, see [Localize Processes](#). When you have added a language in addition to the default (English) language, the **Translate** (🌐) button appears in the forms editor. Click this button to supply localized strings for controls present within the form.

Let's consider a simple example to understand this feature in detail.

1. Create a web form with a couple of controls as shown in the following figure:



2. Click **Translate** (🌐) to provide localized strings corresponding to these controls.
3. In the Translate form window, select a language for which you want supply localized strings from the **Languages** drop-down field.

All languages added to the application are present in the **Languages** drop-down field.

4. Specify the strings for each control in the form and click **OK**. The following image shows strings provided for the Spanish language:

Note:

In addition to control fields, you can also provide localized strings for constant values that you specify through events or as computed values.

| Translate form | | |
|----------------------------|----------------|---|
| | | Spanish <input type="button" value="Clean"/> |
| ID | Default Value | Current Language |
| InputText - Label | InputText | Texto |
| InputText - Default Value | Default value | Valor por defecto |
| Table - Label | Table | Tabla |
| Column - Label | Number | Número |
| InputText1 - Label | InputText1 | Texto1 |
| InputText1 - Default Value | Enter a number | Ingrese un número |
| | | <input type="button" value="OK"/> <input type="button" value="Cancel"/> |

- If a language has a sub-locale corresponding to a country, for example, Spanish (Argentina), you can specify strings that differ for this sub-locale. The remaining strings are picked up from the parent language. The following figure shows example strings specified for a sub-locale:

| Translate form | | |
|----------------------------|----------------|--|
| | | Spanish (Argentina) <input type="button" value="Clean"/> |
| ID | Default Value | Current Language |
| InputText - Label | InputText | Ingrese Texto |
| InputText - Default Value | Default value | |
| Table - Label | Table | |
| Column - Label | Number | |
| InputText1 - Label | InputText1 | Ingrese Texto1 |
| InputText1 - Default Value | Enter a number | |

 **Note:**

- If you have added a new language to the application but not specified localized strings for it in the forms editor, the default strings (in English) for each control are displayed when a user selects this language in runtime.
- If you have added a new language with a sub-locale to the application but not specified localized strings for it in the forms editor, the parent language strings (if specified) or default strings for each control are displayed when a user selects this sub-locale in runtime.

6. In the forms editor, you can also specify strings for languages that have unique scripts, such as, Chinese, Arabic, or Hindi.

| ID | Default Value | Current Language |
|-------------------|---------------|------------------|
| InputText - Label | InputText | इनपुट टेक्स्ट |

7. If you add a new control to the forms editor, the Translate form window is automatically updated to show this control's label. However, other fields that take constants for input are displayed only if you have entered a value in these fields.
8. If you delete a control from the forms editor, the localized strings that were specified for the control show up as grayed out rows in the Translate form window. To completely remove a string of a deleted control, click the corresponding **Delete** button or click **Clean** to clean up the entire window.

| Translate form | | |
|----------------------------|----------------|---|
| ID | Default Value | Current Language |
| | | Spanish <input type="button" value="Clean"/> |
| Table - Label | Table | Tabla |
| Column - Label | Number | Número |
| InputText1 - Label | InputText1 | Texto1 |
| InputText1 - Default Value | Enter a number | Ingrese un número |
| Unknown | Unknown | Texto  |
| Unknown | Unknown | Valor por defecto  |

9. Click **Preview** to see how the form is displayed for different language selections.



10. Click **Save** to save your changes to the form.

Preview Forms and Their Payload

At any point in configuring a web form, preview it to see its appearance at different device sizes, test its behavior, and view its payload.

1. In the web form editor, click **Preview**.
2. In the Preview window, switch to device sizes users might use to see your web form's layout. The web forms automatically adjust their layout according to the screen size of the device that users use to display them. Use the drop-down fields to change the language and display percent (for example, choose 150% to zoom in on a form).
3. Enter sample values in the form's controls, and click **Submit**.

Notice that the field at the bottom of the Preview window displays the payload values. The payload values are the values that you specified in the web form's controls. These values are stored as data attributes for use in a process.

Dynamically populated controls, such as controls that call a REST connector, also function in preview mode.

4. Click the **Trigger Custom Outcome** button to test the form with different outcome values. In the resulting window, enter a custom outcome value and click **Submit**. Observe the form's behavior in the payload field.
5. Click **Close**.

Monitoring Form Logs

Click **Log** at the bottom of the Preview window to view activity logs for a form. Monitoring these logs helps you identify and address issues with your form.

Under the **Log** tab, all activities (such as events, actions, connector calls, computed value population, and errors) occurring in your form are recorded chronologically.

Expand an activity log to view the details under it, including the control on which the activity occurred, the time at which the activity occurred, and so on.

Narrow down to specific activity logs using the **Filter Logs** button. You can filter based on actions, events, errors, and so on. You can use more than one filter at a time.

Click **Clear Logs** to delete all log messages.

Reloading Form with Custom Payload

Click the **Reload With Payload** button to test your form with custom payload. In the resulting dialog box, enter the payload JSON. Edit the JSON to include your custom data and click **OK**. The form loads with the custom data.



Note:

You can obtain the payload JSON from the **Outcome** tab.

Add Dynamic Behavior to a Form

Use events to introduce dynamic behaviors into your web forms, and combine them with actions, conditions, functions, and REST connector calls.

For example, you can introduce the following behaviors into your forms:

- Populate data in a control field based on another control field in the form. For example, a Country select field will impact the State select field and the State select field will impact the City select field.
- Enable control field validation based on another form control field. For example, if Start Date is given then End Date is mandatory or a Full Name gets its value from the First Name and Last Name.
- Make a REST call on demand, store the call's response, and use response data in an event action or condition.



Note:

To use logged-in user's credentials when loading a form to execute a REST operation, define a REST connector without credentials. To use the same operation as a service call in the process, define another identical REST connector with credentials. This applies to internal REST API calls only.

Topics:

- [Configure Events](#)
- [Specify Actions](#)
- [Specify Conditions](#)
- [Specify Functions](#)
- [Specify Filters in Events](#)
- [Reuse Event Snippets](#)

- [Specify Custom Outcomes for Forms](#)
- [Execute REST Connector Calls in Events](#)
- [Populate Controls Using REST Calls](#)
- [Link and Refresh List of Value Fields](#)
- [Examples of Loops in Forms](#)
- [Example of Current Logged in User Data Function](#)

Configure Events

By configuring one or more events on a control or presentation, you change the control or presentation's behavior. Configuring events in forms enables you to trigger connector calls, actions, conditions, and functions.

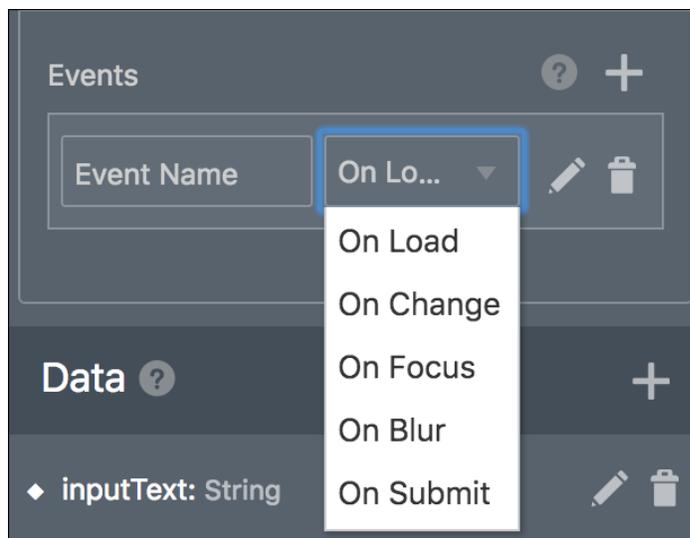
To configure events for a control or presentation:

1. In the web forms editor, select a control or presentation, and locate the **Events** options.
 - a. For a control, select the control and scroll down the **General** tab of the Properties pane until you see **Events**.
 - b. For a presentation, click a blank area of the form so that the **Form** and **Presentation** tabs display in the Properties pane. On the **Form** tab, select a presentation, then select the **Presentation** tab. Scroll down until you see **Events**.

Notice placeholder text displayed in the Events field *No events defined*.

2. Click **Add**  to define an event.

The Events section now displays two fields; an **Event Name** field and a field with a drop-down menu listing event options available for the control or presentation.



3. Select an event option from the list and, optionally, enter a name for the event. Available event options are specific to the selected control or presentation.

| Event | Description | Control or Presentation |
|---------------------|--|---|
| On Load | Fires when a form or presentation loads. | Input Text, Text Area, Date, Time, Date Time, Email, URL, Money, Phone, Checklist, Checkbox, Radio Button, Button, Select, Section, Table, Repeatable Section, Message, Image, Video, Tab and Panel. Presentation |
| On Change | Fires when an end user changes the control's value. | Input Text, Text Area, Date, Time, Date Time, Email, URL, Money, Phone, Checklist, Checkbox, Radio Button and Select. |
| On Focus | Fires when the cursor is in the control. | Input text, Text Area, Number, Date, Time, Date Time, Email, URL, Money and Phone. |
| On Blur | Fires when the control loses focus. | Input text, Text Area, Number, Date, Time, Date Time, Email, URL, Money and Phone. |
| On Submit | Fires before the form or presentation is submitted. | Input Text, Text Area, Date, Time, Date Time, Email, URL, Money, Phone, Checklist, Checkbox, Radio Button, Button and Select. Presentation |
| On Click | Fires when an end user clicks the button. | Button and Link |
| On Expand | Fires when a section is expanded. | Section |
| On Collapse | Fires when a section is collapsed. | Section |
| On Expand Toggle | Fires when an expanded section is toggled. | Section |
| On Row Add | Fires when an end user adds a row to a table or repeatable section. | Table and Repeatable Section |
| On Row Remove | Fires when an end user removes a row from a table or repeatable section. | Table and Repeatable Section |
| On Selection Change | Fires when an end user changes the row selection in a table or repeatable section. | Table and Repeatable Section |

- After selecting an event option, click the editing icon next to the events drop-down menu to specify the actions, conditions, connectors, or functions for the event. Click **OK** to complete configuring the event.

See [Specify Actions](#), [Specify Conditions](#), and [Execute REST Connector Calls in Events](#).

 **Note:**

To cancel an event, click its delete icon. Click **Add**  to define an additional event. You can define multiple events for a control.

You can configure multiple events in a control. The configured events are executed sequentially in runtime. Note that, by default, the newest configured event is added last in the event sequence, and subsequently gets executed last in runtime. But you can re-order events to control their execution sequence. Use the event re-order handler  to re-order events and ensure that they get executed in the correct sequence in runtime. The event re-order handler

gets displayed in front of each event name field only when there is more than one event configured on a control.

If you want to disable all events configured for a control, select the control, and specify the action **Disable Events**. See [Specify Actions](#). This disables all events configured for the control and the events do not get executed in runtime. But note that On Load type of events are exceptions. On Load events get executed even if you specify the **Disable Events** action.

To enable events in controls where they have been disabled by the **Disable Events** action, configure the **Enable Events** action. This ensures that all those events get executed in runtime.

If the **Disable Events** action is configured for container controls such as tables, repeatable sections, sections, panels, and tabs, then all events in child controls that reside inside them are disabled. This also applies to any rows that you add to container controls after the **Disable Events** action has been configured in the container controls. To enable events in container controls, and also in all child controls inside the container controls, configure the **Enable Events** action in the container controls.

Specify Actions

Actions let you trigger changes to a control. You can choose from a variety of control and style actions. For example, you might configure a Clear button to clear the values of other form controls.

To specify actions for a control:

1. Select a control on the form canvas and specify an event for it. For example, choose **On Click** for a button control. See [Configure Events](#).
2. Click the **Event** editing icon adjacent to the **Event** field.

The event window displays with the selected event option (for example, On Load) at the top, and color coded buttons for adding actions, conditions, or connectors to the event.

3. Click **+Action** to add an action.

A solid red **Action** indicator appears. The **Summary** field below it displays a compact version of the action you'll define. Use the button at the end of this field to collapse or expand the actions editor.

4. In the **Control Name** field, select the control the action will affect.
5. In the **Action** field, select the action to take place. For example, select **Clear Value** to clear the selected control's value.

You can select from a variety of actions grouped in **Control** and **Style** categories. The options vary based on the control.

6. If configuring a control action, complete any additional fields that display. Control actions such as Value, Label, Help, Hint, Placeholder, Min Value, Max Value, and Pattern require further configuration for **Type** and **Value** fields.

If a **Type** field displays, select from these options:

- **Constant** lets you specify a value for the selected control. Be sure to specify the corresponding value in the **Value** field.
- **Data Definition** lets you use the data from the payload. When you click the adjacent **Value** field that displays, available items are listed.
- **Control** lets you use the data from a control. Be sure to select the control name and property to use.

Properties are control specific. For example, some controls, such as the number control, have many properties as options. Other controls, such as the image control, only have one property: Hidden. Email and phone controls have a property Placeholder but don't have the Min Value and Max Value properties.

- **Function** lets you perform common operations with strings, values and arrays, as described in [Specify Functions](#).
- **Connector Data** lets you assign a value from a connector call made from the same event.

Note:

You can modify multiple properties of controls such as their value or label. You can also use the data, such as connector data, to modify the control, but you cannot modify data.

7. If configuring a style action, complete additional fields that display for the selected action. For example, you might combine a condition with a style action so that if a certain value is exceeded, a control's color, label, or class changes to alert users.
8. If needed, use the buttons at the bottom of the event window to configure additional actions or specify a condition or connector call. See [Specify Conditions](#) and [Execute REST Connector Calls in Events](#).

Connector calls cannot be executed based on a condition. Connectors are always executed when their event is executed. You can later use the connector response values in any condition or action.

 **Note:**

In the event window, use the **Undo** or **Redo** buttons to remove or restore recent changes to actions. Use the delete icon to delete an action. Click **Cancel** to exit the event window without saving changes.

9. After completing the event, click **OK**, then **Save**.

Specify Conditions

Conditions let you configure an If/Then/Else condition to trigger an action or connector call for a control's selected event or a control's specified action.

To specify conditions for a control:

1. Select a control on the form canvas and specify an event for it. See [Configure Events](#).
2. Click the event editing icon adjacent to the **Event** field.

The event window displays with the selected event option (for example, On Load) at the top, and color coded buttons for adding actions, conditions, or connectors to the event.

3. Click **+If** to add a condition.

Multiple items display to help you construct and complete the condition.

- Solid blue **If**, **Then**, and **Else** indicators signal each portion of the condition to complete.

The **Summary** field below the **If** indicator displays a compact version of the condition you'll define. Use the button at the end of this field to collapse or expand the conditions editor.

- Use the **+Condition**, **+Action**, **+Else If**, and **+Else Action** buttons to add additional conditions or actions to the condition.
 - Use the fields displayed under the **If** or **Else If** indicators to define conditional behavior.
4. Complete the If portion of the condition.

- a. Select a type from the **Type** field and complete fields that display. Select from these options:

- **Constant** lets you specify a value for the selected control. Be sure to specify the corresponding value in the **Value** field.
- **Data Definition** lets you use the data from the payload. When you click the adjacent **Value** field that displays, available items are listed.
- **Control** lets you use the data from a control. Be sure to select the control name and property to use.

Properties are control specific. For example, some controls, such as the number control, have many properties as options. Other controls, such as the image control, only have one property: Hidden. Email and phone controls have a property Placeholder but don't have the Min Value and Max Value properties.

- **Function** lets you perform common operations with strings, values and arrays, as described in [Specify Functions](#).
 - **Connector Data** lets you assign a value from a connector call made from the same event.
- b. Select an operation. For example, choose **is True** from the drop-down field.

 **Tip:**

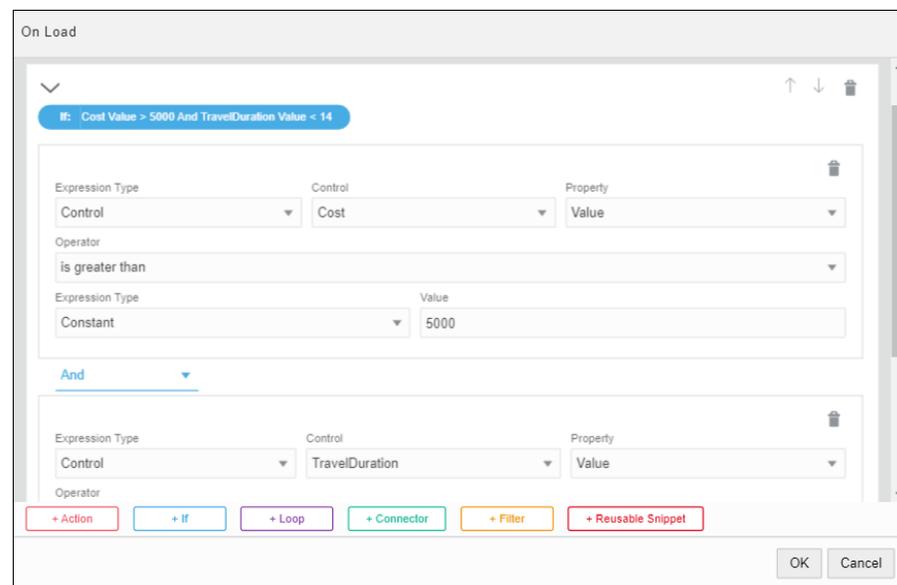
To check for a null value, choose **is False**; if a control has no value, the condition result returns true. Conversely, choose **is True** to check that a property is present.

 **Note:**

You can also check if a particular property, such as value, of a control is empty or not using the **is empty** operation. Controls of array type (repeatable sections or tables) are considered empty if the array is empty. A check box or a checklist is considered empty if no selection is made.

- c. Based on the selected operation, complete a second **Type** field and additional fields that display. You must specify a type and additional fields for all operations except when **is True** or **is False** are selected.
- d. To add another condition, click **+Condition**, specify if it's an **And** or an **Or** condition and configure it.

For example, as shown in the figure, you can configure a condition for the Cost control field of a travel request form such that if cost is greater than 5000 and the duration of travel is less than 14 days then you can set further Then or Else conditions to be triggered.



The screenshot shows a configuration window titled "On Load" with a list of conditions. The first condition is highlighted in blue and reads "If: Cost Value > 5000 And TravelDuration Value < 14". Below this, the configuration for the first condition is shown: Expression Type is "Control", Control is "Cost", Property is "Value", and Operator is "is greater than". The configuration for the second condition is shown below: Expression Type is "Constant", Value is "5000", and Operator is "And". At the bottom of the window, there are buttons for "+ Action", "+ If", "+ Loop", "+ Connector", "+ Filter", and "+ Reusable Snippet", along with "OK" and "Cancel" buttons.

- Complete the Then portion of the condition by clicking **+Action** and completing the action. See [Specify Actions](#).

For example, for the Cost control of a travel request form if the cost is greater than 5000 and the number of days of travel is less than 14 then the Justification field is required in the form.

The screenshot shows the configuration of a 'Then' condition. At the top, there are three dropdown menus: 'Expression Type' set to 'Control', 'Control' set to 'TravelDuration', and 'Property' set to 'Value'. Below these is an 'Operator' dropdown set to 'is less than'. Underneath is another set of dropdowns: 'Expression Type' set to 'Constant' and 'Value' set to '14'. A blue 'Then' button is visible. Below the condition configuration, there is a red pill-shaped button labeled 'Action: Justification Required'. At the bottom, there are two dropdowns: 'Control' set to 'Justification' and 'Action' set to 'Required'.

- Complete the Else portion of the condition by clicking **+Else Action** and completing the action.

For example, for the Cost control of a travel request form if the cost is greater than 5000 and the number of days of travel is less than 14 then the Justification field is required in the form else it's optional.

The screenshot shows the configuration of both 'Then' and 'Else' conditions. The 'Then' section is at the top, with a blue 'Then' button and a red pill-shaped button labeled 'Action: Justification Required'. Below it are dropdowns for 'Control' (Justification) and 'Action' (Required). The 'Else' section is below, with a blue 'Else' button and a red pill-shaped button labeled 'Action: Justification Optional'. Below it are dropdowns for 'Control' (Justification) and 'Action' (Optional). Buttons for '+ Then if', '+ Action', '+ Else If', and '+ Else Action' are visible at the top of their respective sections.

- If needed, add and configure additional If/Then/Else conditions.

Note:

In the event window, use the **Undo** or **Redo** buttons to remove or restore recent changes to conditions. Use the delete icon to delete a condition. Use up and down arrows to reorder conditions within the window. Click **Cancel** to exit the event window without saving changes.

- After completing the event, click **OK**, then **Save**.

Specify Functions

Use functions in event actions and conditions to perform common operations with strings, values, and arrays. For example, use a function to add two values, concatenate two strings, or sum items in table rows.

Important points about functions:

- You can specify parameters for selected functions. The parameter value can be a constant, data definition value, control value, other function, or connector data value.
- Some functions support selecting arrays of data or repeatable controls.
- You can nest functions, such as concatenate multiple strings or the results of multiple rows. You might select a concatenate function and concatenate a data value, connector data value, or control value with another value such as a constant.

Choose from the following mathematical, aggregation, and text functions.

| Function Category | Function Name | Parameters | Description |
|-------------------|--------------------------|------------|---|
| Other | Create UUID | None | Generates a universally unique identifier. |
| Other | Current browser language | None | Returns the locale that the user has specified in the browser settings. For example: en, en-US, de, fr |
| Other | Get application name | None | Returns the current application name. |
| Other | Get Geolocation | (Number) | Returns a String with the JSON representation of an object that contains the current position of the device. You must specify a timeout value in milliseconds to wait for the browser's response. A timeout of 10000 milliseconds works well in most situations. The returned String with the JSON representation is similar to: <pre>{"latitude":37.556685099999996, "longitude":-122.2777825, "altitude":null, "accuracy":45, "altitudeAccuracy":null, "heading":null, "speed":null}</pre> You can use the function <code>Parse from JSON</code> to convert the returned String to an object, and then the function <code>Get Property</code> to access the attributes in the object. For example, to access the latitude use: <pre>Get Property(Parse from JSON(Get Geolocation(10000)), "latitude")</pre> |
| Date | Current Date | None | Generates the date in yyyy-mm-dd format. |

| Function Category | Function Name | Parameters | Description |
|-------------------|-------------------------------|--|---|
| Date | Current Time | None | Generates the time in 24 hour T00:00:00 format (for example, T23:59:59). |
| Date | Current Date Time | None | Generates the date/time in yyyy-mm-dd T00:00:00 format. |
| Date | Add Seconds | (Date, Number) | Adds the number of seconds to the given date, time, or date/time. |
| Date | Add Minutes | (Date, Number) | Adds the number of minutes to the given date, time, or date/time. |
| Date | Add Hours | (Date, Number) | Adds the number of hours to the given date, time, or date/time. |
| Date | Add Days | (Date, Number) | Adds the number of days to the date. |
| Date | Add Months | (Date, Number) | Adds the number of months to the date. |
| Date | Add Years | (Date, Number) | Adds the number of years to the date. |
| Logic | And | (Input1, Input2, Input3...) | Checks if Input1 && Input2 && Input3... is <i>true</i> or <i>false</i> . |
| Logic | Or | (Input1, Input2, Input3...) | Checks if Input1 Input2 Input3... is <i>true</i> or <i>false</i> . |
| Logic | Not | (Condition) | Negates the value of the specified condition. |
| Logic | Inline If | (Condition, ValueIfTrue, ValueIfFalse) | Implements an If...Else statement. If the condition specified is <i>true</i> , the first value is returned as the result, else the second value is returned. Example: Inline If (N=5, 60, 70) returns 60 if N=5 else returns 70. |
| Relational | Equal (=) | (Input1, Input2) | Checks if Input1 equals Input2 and returns a Boolean value. |
| Relational | Greater than (>) | (Input1, Input2) | Checks if Input1 is greater than Input2 and returns a Boolean value. |
| Relational | Less than (<) | (Input1, Input2) | Checks if Input1 is less than Input2 and returns a Boolean value. |
| Relational | Greater than or equal to (>=) | (Input1, Input2) | Checks if Input1 is greater than or equal to Input2 and returns a Boolean value. |
| Relational | Less than or equal to (<=) | (Input1, Input2) | Checks if Input1 is less than or equal to Input2 and returns a Boolean value. |
| Math | Sum (+) | (Number, Number) | Adds two numbers. |
| Math | Summation | ([Number]) | Adds arrays of numbers. |
| Math | Subtract (-) | (Number, Number) | Subtracts numbers (for example, 10–5). |
| Math | Multiply (*) | ([Number]) | Multiplies numbers. For example, multiply all values in a column. |
| Math | Divide (/) | (Number, Number) | Divides numbers and includes the decimal portion, up to 10 decimal places (for example, 4/3=1.3333333333). |

| Function Category | Function Name | Parameters | Description |
|-------------------|------------------|----------------------------|--|
| Math | Integer Division | (Number, Number) | Divides numbers and truncates the result (for example, $5/2=2$, $-5/2=-2$). |
| Math | Modulo (%) | (Number, Number) | Finds the remainder after division of one number by another. |
| Array | Min | ([Number]) | Finds the minimum value in an array. |
| Array | Max | ([Number]) | Finds the maximum value in an array. |
| Array | At Index | ([Any], Index) | Finds the value at a particular index of an array or string. |
| Array | Count | ([Any]) | Finds the count value in an array. |
| Array | Avg | ([Number]) | Finds the average value in an array. |
| Array | Concat | ([Any]) | Joins array values. |
| Array | IndexOf | ([Array], Element) | Returns the index of an element in an array. For example, <code>IndexOf ([1,2,3], 1)</code> returns 0. Returns -1 if the element specified is not found in the array. |
| Text | Concat | ([String]) | Joins a text string. |
| Text | Split | (String, String) | Splits a string into an array using the second parameter as a separator. For example, you can split a series of numbers in a text field into a checklist. |
| Text | Join | ([String], String) | Joins an array into a string using the second parameter as a separator. For example, you can fetch values of all rows within a table column and create a series. |
| Text | Trim | (String) | Removes leading or trailing spaces. |
| Text | Contains | (String or Array, Element) | Checks if a string or array (within a control) contains a specific element and returns a Boolean value. |

| Function Category | Function Name | Parameters | Description |
|-------------------|---------------|--------------------------|---|
| Text | Replace | (String, String, String) | <p>Replaces a text string. Uses three parameters, where:</p> <ul style="list-style-type: none"> • The first string is the original text • The second string is the value or regular expression that will be replaced by the new value • The third text is the replacement <p>For example, REPLACE("Hello World!", "Hello", "World") returns "World World!"</p> |

 **Note:**

In regular expressions there are 12 characters with special meanings: backslash \, caret ^, dollar sign \$, period or dot ., vertical bar or pipe symbol |, question mark ?, asterisk or star *, plus sign +, opening parenthesis (, the closing parenthesis), opening square bracket [, and opening curly brace {. If you want to use any of them as regular characters, you will

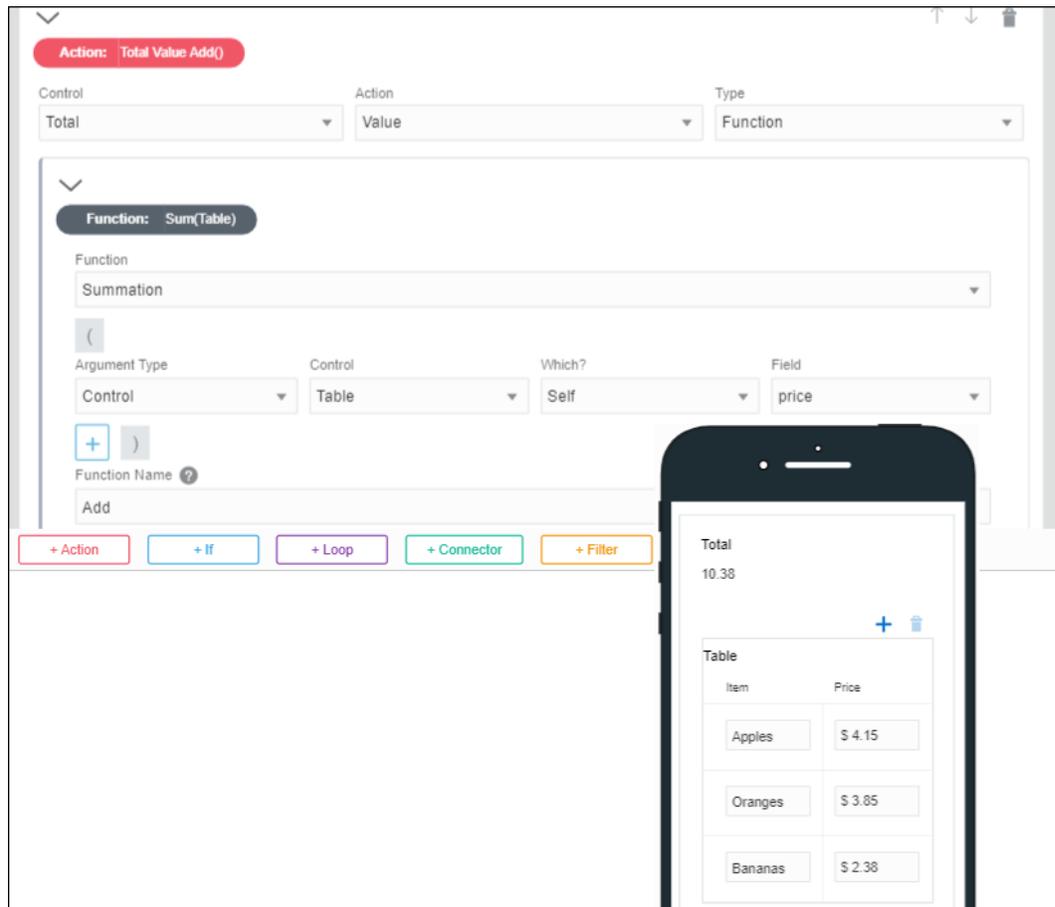
| Function Category | Function Name | Parameters | Description |
|-----------------------------|----------------------------|---------------------------------|--|
| | | | need to escape them with backslash \. |
| Text | Matches | (String, String) | Checks if two strings or expressions match and returns a Boolean value. |
| Text | Substring | (String, StartingIndex, Length) | Returns a substring of a string, beginning at the specified starting index up to the length indicated. For example, Substring ('apple', 0, 3) returns <i>app</i> . |
| Interval | Duration in days | (Date, Date) | <p>Calculates the difference between two dates and returns the result in number of days.</p> <p>To calculate the interval, you can manually enter start and end dates in the event window or supply them through date controls present within the form.</p> <p>If you choose to manually provide dates for calculation, use the following valid formats:</p> <ul style="list-style-type: none"> • mm-dd-yyyy • yyyy-mm-dd • 7 October 2017 • 7 Oct 2017 • December 25, 2017 <p>You can use the following separators interchangeably:</p> <ul style="list-style-type: none"> • - • / • , • space |
| Interval | Duration in months | (Date, Date) | Calculates the difference between two dates and returns the result in number of months. |
| Interval | Duration in years | (Date, Date) | Calculates the difference between two dates and returns the result in number of years. |
| Current Logged in User Data | Current user's ID | None | Returns the current logged in user's ID. |
| Current Logged in User Data | Current user's first name | None | Returns the current logged in user's first name. |
| Current Logged in User Data | Current user's middle name | None | Returns the current logged in user's middle name. |
| Current Logged in User Data | Current user's last name | None | Returns the current logged in user's last name. |

| Function Category | Function Name | Parameters | Description |
|-----------------------------|--------------------------------------|------------|---|
| Current Logged in User Data | Current user's email | None | Returns the current logged in user's email. |
| Current Logged in User Data | Current user's manager's ID | None | Returns the current logged in user's manager's ID. |
| Current Logged in User Data | Current user's manager's first name | None | Returns the current logged in user's manager's first name. |
| Current Logged in User Data | Current user's manager's middle name | None | Returns the current logged in user's manager's middle name. |
| Current Logged in User Data | Current user's manager's last name | None | Returns the current logged in user's manager's last name. |
| Current Logged in User Data | Current user's manager's email | None | Returns the current logged in user's manager's email. |

To use a function:

1. In the web forms editor, select a control on the canvas. Note that you can also apply events and functions to presentations, as described in [Configure Events](#).
2. On the Properties pane, scroll down to the **Events** field. Click **Add**  to add an event and select its type in the drop-down field (for example, On Change).
3. Click **Edit**  next to the event you just added.
4. In the event window, add an action or condition by clicking the **+Action** or **+If** button.
5. Configure the action or condition to use a function.
 - For an action, select a control on which to apply the function in the **Control Name** field. In the **Action** field that displays, select **Value** in the **Action** field. In the **Type** field that displays, select **Function**, then choose a function from the functions listed by category in the **Function** field. Optionally, provide a name for the function in the **Function Name** field to easily identify it (this name displays in the **Action Summary** field).
 - For an if condition, select **Function** in the **Type** field that displays, and choose a function from the functions listed by category in the **Function** field. Optionally, provide a name for the function in the **Function Name** field to easily identify it (this name displays in the **Condition Summary** field). Complete the condition in the remaining condition fields.
6. Specify parameters within parentheses for functions that allow them.
Additional fields display within parentheses for functions that include parameters, such as math functions.
The **Compact Function** field displays a summary of the function you define. Use the button at the end of this field to collapse or expand the functions editor.
7. Click **OK** to close the event window. Click **Preview** to test the function. For example, a simple application uses a function set on the Value field to sum values

each time a user enters or changes a value in the table, and displays the calculated value in the Total field. If the total exceeds a set amount (constant), a style action displays the total in red letters.



Specify Filters in Events

Filters let you selectively use a subset of data from a larger set. For a form control, you can define a filter within an event and use the filter data in other controls through event actions and conditions.

To specify filters within an event:

1. Select a control on the form canvas and specify an event for it. See [Configuring Events](#).
2. Click the event editing icon adjacent to the **Event** field.

The event window displays with the selected event option (for example, On Change) at the top and color coded buttons for adding actions, conditions, connectors, or filters to the event.

3. Click **+Filter** to add a filter, and specify the data source from which you want to filter specific information. You can choose data attributes, another control, or a REST connector.

- If the data source selected is valid (of the type: array), additional **+Criteria** and **+If** buttons appear as shown in the following figure:

- Click **+Criteria** to specify how data is filtered from the data source. You can also use an If condition to do the same.

 **Note:**

You can specify more than one way to filter data from the source. If you specify an Or condition, the results are combined or unioned. If you specify an And condition, the result is the intersection of that filter condition with the other filter conditions.

You can also specify another Filter (specified before) as the source type for your filter. This gives you the ability to filter data from another filter.

- Provide a suitable name for the filter; the information extracted from the data source is stored within this filter.
- In the same event window, you can use the data in the filter to populate other controls dynamically through actions or conditions.
- After completing the event, click **OK**, then **Save**.

Filtering Using Events – An Example

This section demonstrates filtering through events using a web form with a drop-down select control and a table.

Based on the option you choose in the select control, you will be able to filter data from a data source and selectively use the extracted data to populate the table.

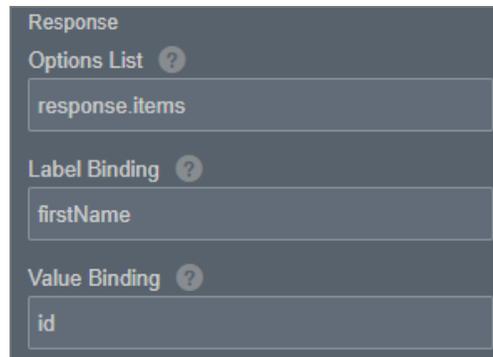
In this example, we use a previously created REST connector to fetch data from a web server, which contains a list of items. Each item, in this case, is a complex data object that contains personal details of individuals, such as first name, last name, unique ID, company name, email, phone number, city, and so on. To create a REST connector from scratch, see [Creating a REST Connector](#).

We use this REST connector to define a global connector call, *globalResponse*, under Presentation properties of the form. Want to know more about global connectors? See [Working with Presentations](#). In addition, this connector also serves as the input to the select control.

Now, let's configure controls and events in the web form to suit this example.

- Add and configure a drop-down select field on the form's canvas.
 - From the Basic Palette, drag and drop a **Select** control onto the canvas.

- b. Select the control, and on the **General** tab, change the **Label** field to *First Name*.
- c. Under Options Source, select the **Connector** option and populate the **Resource** and **Operation** fields with respect to the REST connector defined within the application. See [Populating Controls Using REST Calls](#).
- d. Under Response, specify the items list to display as options of the control.
 - i. For this particular example, we map the first name attribute to the **Label Binding** field, so that only first names of all individuals appear as options in the select control. Also, the unique ID attribute is mapped to the **Value Binding** field as follows:



2. Add and configure a table control on the form's canvas.
 - a. From the Advanced Palette, drag and drop a **Table** control onto the canvas.
 - b. Add two additional columns, and edit column labels as *Last Name*, *Organization*, and *Email*.
 - c. Drag and drop an input text control into each column.
3. Click on the select control, and define an **On Change** event for it in the **General** tab.
4. Click the event editing icon to open the event window. In this window, click **+Filter** to add a filter.
5. The following figure shows the event configuration for this example:
 - a. Connector data is set as the data source and its value comes from the global connector call, *globalResponse*, defined previously.
 - b. The subsequent Filter section shows how the data is filtered from the source, that is, when the ID of an item in the source list matches with the ID of the option selected by the user, all data associated with that item is stored in the filter, *filter*.
 - c. An event action selectively uses the data contained in this filter to populate the table. Although each item may have several attributes associated with it, here we use only last name, company, and email attributes. We map each of these attributes a corresponding input text control or table column.

The screenshot shows the 'On Change' configuration window with the following settings:

- Filter:** Source Type: Connector Data, Value: globalResponse, Store result in: filter.
- Criteria:** List to Filter: globalResponse, Value: id, Operator: is equal to, Type: Control, Control: Select.
- Action: Table Value filter:** Control: Table, Which?: Self, Action: Value, Type: Filter. Value: InputText1, InputText2, InputText3. filter: lastName, company, email.

Note:

While defining event actions, if you choose to map the filter data to a control of simple data type (for example, input text), you have to specify an index value in addition to the attribute that you would like to map to the control.

- When the form loads, the select control displays first names of all individuals as follows:

The screenshot shows a select control with the title 'First Name'. The dropdown menu is open, displaying a search bar and a list of names: Leah, Tyson, Austin, Walls, Whitfield, and Kitty.

- After you make a selection, all data associated with the selected option is stored within the filter, and this data is selectively used to populate the table as shown in the following figure:

| Table | | |
|-----------|--------------|-----------------------|
| Last Name | Organization | Email |
| Welch | CUBIX | aileenwelch@cubix.com |

Reuse Event Snippets

You can extract event snippets from an event editing window and reuse them across the form's presentation.

Define a complex event snippet once and reuse it across different events, thereby saving time and minimizing errors. You can manage all the extracted snippets from the web form's Presentation tab. Additionally, updating a global snippet definition updates all instances of the snippet within other events.

Topics:

- [Extract a Snippet](#)
- [Use a Snippet](#)

Extract a Snippet

Extract any event block, such as action, condition, loop, connector, or filter, as a global snippet and reuse it in other event definitions.

To extract an event snippet:

1. In an event definition window, click **Extract Snippet**.
A name field appears at the top of the window and all the available blocks are selected for extraction.

2. Turn a block's toggle button off to exclude it from extraction.
3. Enter a suitable name for the snippet and click **OK**.

The snippet is now available for reuse in any event within the same presentation. You can access and edit the extracted snippet from the form's **Presentation** tab. See [Work with Presentations](#).

Use a Snippet

Use previously extracted event snippets in any event definition.

1. In the event definition window, select **+ Reusable Snippet** at any point in your definition.
2. Select an extracted snippet from the drop-down list to add it to the window.
3. Additionally, you can click **Detach**  to independently retain the blocks within a global snippet, but not the snippet as a whole. Note that changes made to the global snippet won't reflect in the blocks you've retained.
4. Click **Delete**  to remove the snippet and all its constituents.

Specify Custom Outcomes for Forms

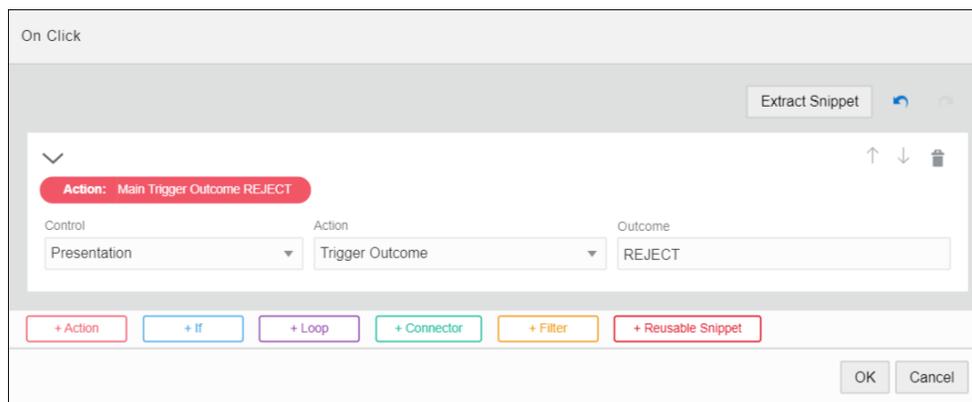
In addition to Submit, you can specify custom outcomes (such as Approve, Reject, On Hold, and so on) for a form according to your requirements and test the same in the Composer. Also, you can specify conditions to prevent users from submitting a form.

You can trigger submit of a form with a custom outcome using any event on a control or presentation. Want to know how to define events? See [Configuring Events](#).

For example, let us specify a form's outcome value through an **On Click** event on a button control. In the event window:

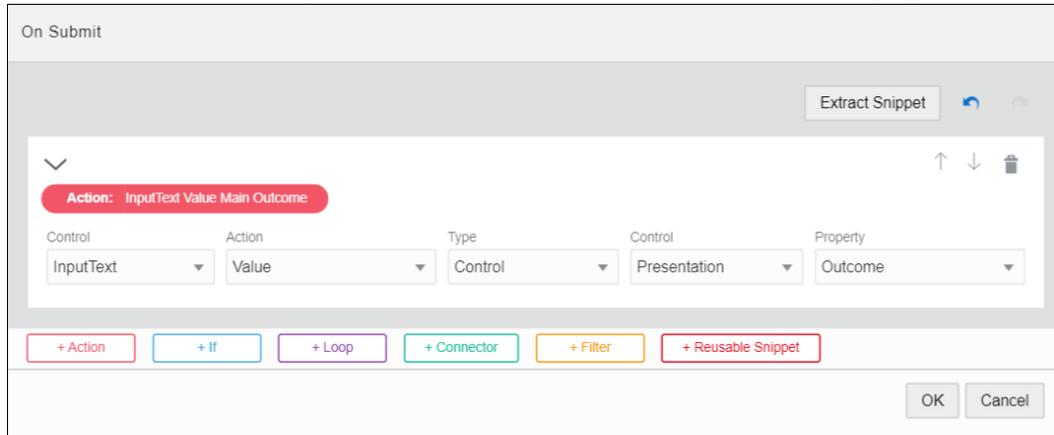
1. Click **+Action** and select the **Trigger Outcome** action available for the Presentation control.
2. Specify a custom value in the **Outcome** field, for example, REJECT.

When a user clicks this button in runtime, the form is submitted with the outcome value as REJECT.

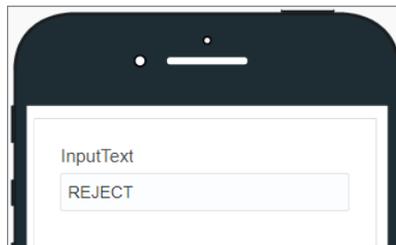


You can also use the outcome value of a form in any control using an **On Submit** event. As an illustration, let us define an **On Submit** event for an input text control to display the form's outcome value. In the event window:

1. Click **+Action** and select parameters for all fields as shown in the following figure.



2. When a user submits the form in runtime, the input text control is populated with the form's outcome value.



You can input and test different custom outcomes using the **Preview** feature, see [Previewing Forms and Their Payload](#).

Preventing Form Submit

By default, the form submission is prevented when the form is incomplete or invalid. You can also define custom conditions within an **On Submit** event to prevent users from submitting a form. For example, you can prevent submit based on a data attribute's or control's value.

The following example shows an **On Submit** event defined for a Presentation. Here, the form submission is prevented if the outcome variable of the form contains a specific value, that is, PREVENT.

The screenshot shows the configuration for an 'On Submit' event. Under the 'If' section, the condition is 'Main Outcome == PREVENT'. The configuration includes three dropdown menus: 'Expression Type' set to 'Control', 'Control' set to 'Presentation', and 'Property' set to 'Outcome'. The 'Operator' is 'is equal to'. The second 'Expression Type' is 'Constant' and the 'Value' is 'PREVENT'. Under the 'Then' section, the action is 'Main Prevent Submit' with the message 'The form couldn't be submitted because an event prevented the submit.' The configuration also shows 'Control' set to 'Presentation', 'Action' set to 'Prevent Submit', and 'Message' set to 'The form couldn't be submitted because an eve'.

Execute REST Connector Calls in Events

Executing a REST call in an event enables you to store the call's response and use it in an event action or condition.

Examples

- Configure a web form that prompts users to enter a zip code. Add an event that calls a weather site to query weather values. Store the response data, then add actions to the event that display temperature-related values.
- Configure a web form that prompts users to enter a company name and click a **Get Quote** button. Add an event to the button that calls a stock service site to query stock values for the specified company. Store the response data, then add event actions that display read only stock value fields.

The screenshot shows a web form with a 'Company' input field and a 'Get Quote' button. Below the button are three stacked read-only fields: 'Last Price' (green background), 'Low' (yellow background), and 'High' (blue background).

Before configuring a REST connector call, [create the REST connector](#) you want the event to call.

To add and configure the connector call:

1. In the web form editor, select a control, add an event, and select its type.
For example, include an On Change event for a text input control or an On Click event for a button control. See [Configure Events](#).
2. Click the event's **Edit** icon to display the **Events** window.
3. Click **+Connector** to add the connector call.
4. Configure the connector call.
 - a. In the **Connector** field, select a REST connector from those defined for the application.
 - b. In the **Resource** and **Operation** fields, select the resource to access and operation to perform as configured in the REST connection.

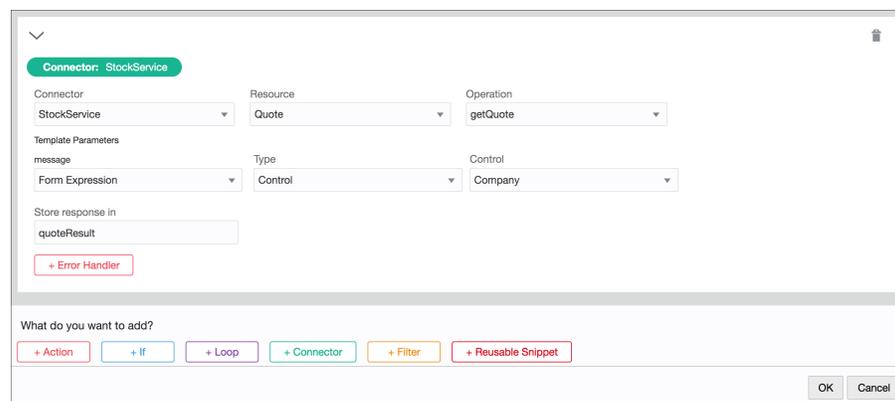
 **Note:**

You can also perform a POST operation from the event window. Currently, for POST operations, arrays aren't supported for requests but can be successfully received and handled in responses.

- c. Complete the connector call parameters (query, header, or body) that display based on the operation you choose. For example, the illustration below shows a connector call with a GET operation and a template parameter called `message`.

You can configure query parameters either from static texts or from form expressions (as shown in the example). Use **Form Expressions** to specify complex values in query parameters based on functions, payload data, and control values. Get values from payload data directly without dependency on controls by using Data Definition type. Configure functions by using the Function type, for example, specify a concat function to combine two or more values.

- d. Enter a variable name in which to store the response data (for example, `quoteResult` or `weatherInfo`).



5. Optionally, configure error handling. You can set actions to address all errors or a selected error detected in the call.
 - a. Click **+Error Handler** to add an error handler.
 - b. In the **Run on Error** field, select an error (or all errors) to detect.
 - c. In the **Control Name** field, select the control on which to identify the error. In the **Action** field, select the control or style action to take.

6. Collapse the connector call.
7. Below the connector call, configure actions, conditions, or additional connectors, and click **OK**.

A common use case is to add an action to display response data from the connector call. Click **+Action**, select a control, specify **Value** as its action, **Connector Data** as its type, and the response variable data to display in the **Value** field.

The screenshot shows the configuration for a connector call. The connector is 'StockService' and the response is stored in 'quoteResult'. Three actions are configured:

- Action: LastPrice Value quoteResult.lastPrice**
Control: LastPrice, Action: Value, Type: Connector Data, Value: quoteResult.lastPrice
- Action: LowPrice Value quoteResult.low**
Control: LowPrice, Action: Value, Type: Connector Data, Value: quoteResult.low
- Action: High Value quoteResult.high**
Control: High, Action: Value, Type: Connector Data, Value: quoteResult.high

Buttons at the bottom: + Action, + If, + Loop, + Connector, + Filter, + Reusable Snippet, OK, Cancel.

Note:

- If the connector data is an array of elements, you can specify a particular array item in the response **Value** field by entering the corresponding array index, for example, `quoteResult[0].lastPrice[1].price`.
- If the REST response consists of a complex data object, you cannot map the data object directly into a control (such as Input Text or Text Area) through an event. Instead, you must specify which details of the complex data object should be mapped to the control, along with an appropriate function such as *Concat* (if required), in the event window. For example in a REST response, if a complex data object called **name** contains two elements, **firstname** and **lastname**, you must specify which one of these elements should be populated into the control. If you require both of these elements to be populated, use a function such as *Concat* with a suitable separator in the event window.

Populate Controls Using REST Calls

Dynamically populate controls such as drop-down select, check list, radio button, table, and repeatable section controls with data using REST connectors.

1. Drag and drop a drop-down select, a check list, a radio button, a table, or a repeatable section control onto the canvas to populate it with data using a REST call.
2. Select the control and optionally, on the **General** tab in the Properties pane, edit the name, binding, label, and other fields.
3. For a drop-down select, check list, or radio button control, in the **Options Source** field, select **Connector** and then configure the fields under the **Options Source** field to define the values based on a REST call. For a repeatable section or table control, select the **Use Data From Connector** field and configure the fields under this field.

- a. **Connector:** Select a REST connector from the defined list of connectors for the application.

A REST connector lets you fetch data and perform tasks based on that data. See [Create REST and Web Service Connectors](#).

- b. **Resource:** Select a resource from which data should be fetched.

- c. **Operation:** Specify an operation to call.

An operation indicates the task you want to perform. For example, a user may use an operation to fetch all the countries or to fetch a list of states based on a country code.

Options Source

Static From Data Connector

Connector

Countries

Resource

States

Operation

getForCountry

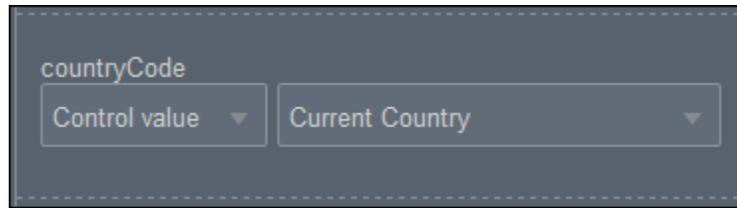
4. Based on the call, the system will display a list of required parameters (*payload values*) and a response below the **Options Source** field or the **Use Data From Connector** field.

- a. **Payload values:** Specify the information to pass to the REST connector (parameters). You can specify query, header, or template parameters.

If you selected **Text**, enter the parameter information. Or, if you selected **Control value**, select a control value from the available options.

 **Note:**

Text parameters are secure and remain on the server.



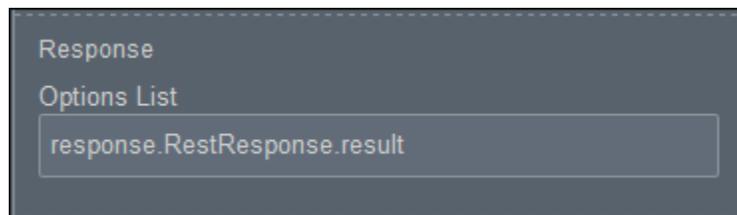
b. **Response:** Define how the response will be mapped to the control properties:

- **Options List:** Define the mapping by specifying an attribute list from the response that contains the items to display as the options in the control.
- If the list is a simple type of list such as a list of strings or numbers, then no label and value mappings are needed. The value of the item in the list will be used as both the label and the value of the option.

If the list consists of complex elements, then you need to specify a mapping using the **Label Binding** and **Value Binding** fields to identify the label and value.

 **Note:**

- For a table control, define how the response will be mapped to each column in the table. For a repeatable section, define which property from the response will be mapped to which control inside the repeatable section.
- If the connector data is an array of elements, you can map a particular array item to a **Response** field by entering the corresponding array index, for example, *response[0].RestResponse[1].result*.



5. You may also use the **Skip Upon Load** property to determine when the connector data loads into the control.
 - Deselect the checkbox (default state) to allow connector data to be populated into the control when the form loads.
 - Select the checkbox to prevent connector data being populated into the control when the form loads. When you use this option, you must explicitly execute a connector refresh for the data to load into the control.
6. Optionally, in the **Events** field, configure the events for the control. See [Add Dynamic Behavior to a Form](#).

Test the dynamic fetching of data by testing the application. And, verify if the mappings you defined work correctly as expected.

Link and Refresh List of Value Fields

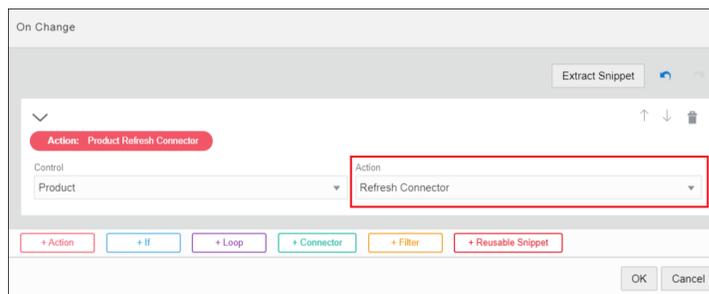
Use the Refresh Connector control action in an event so that each time an end user makes a selection in a LOV field, REST call data is refreshed and reflected in another LOV field. For example, in an order application form, after a user selects from a Category drop-down field, the list of products for that category only displays in a Product drop-down field.

To refresh an LOV field:

1. In a web form, configure two LOV controls (select, checklist, or checkbox) to use the same REST connector to populate and return values such as categories and products. See [Populate Controls Using REST Calls](#).

In this LOV use case, the Product control has query parameters set to search for the Category control's value.

2. Select the first control (for example, Category field). On the Properties pane, scroll down to the **Events** field and add or edit an event. For example, add an **On Change** event and click **Edit**  to edit the event.
3. In the event window, add an action by clicking the **+Action** button.
4. In the **Action** field that displays, select **Refresh Connector**. This setting means that each time users make a selection in the Category field, the Product field's REST call is refreshed to return product values for the selected category.



5. Click **OK** to close the event window. Click **Preview** to test the function.

Examples of Loops in Forms

You can use loops in form controls to execute actions that need to iterate. The iterator can be a number or an array. Loops are a way to execute the same action multiple times in a form.

Let us explore with some examples how loops work in forms.

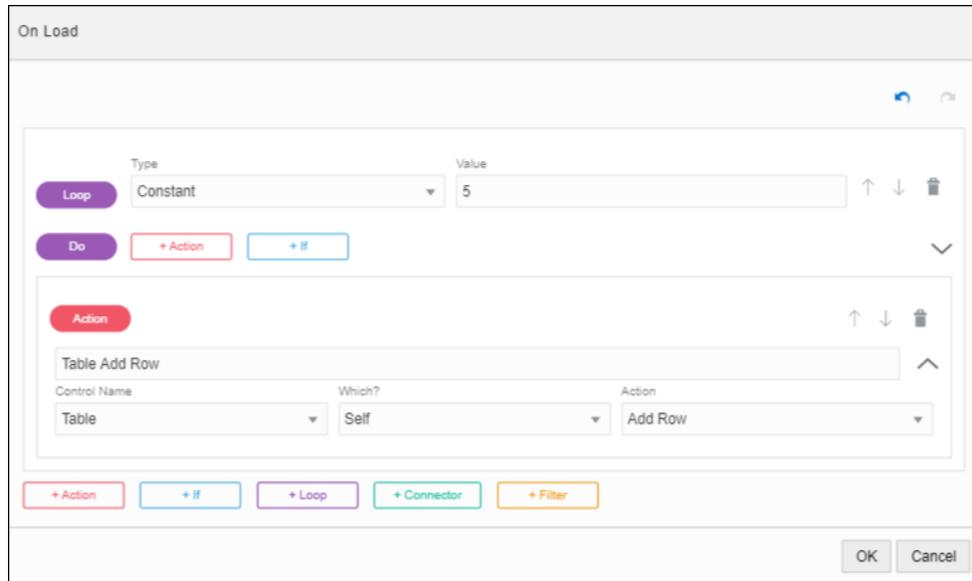
- [Configure a Simple Loop](#)
- [Configure a Loop With an Array](#)

Configure a Simple Loop

In this example, let us add multiple rows to a table by configuring a simple loop with a number iterator.

1. Drag and drop a table control onto your form's canvas. Then drag and drop an InputText control in the table's column.

2. Select the table control, and in the properties pane that displays scroll down till you find **Events**. Click  to configure an event for the control.
3. Select **On Load** from the drop-down list, and then click the editing icon.
4. In the event On Load window, click **Loop**. Select **Constant** in the **Type** drop-down list, and enter a number in the **Value** field, for example: 5. This value determines the number of times looping occurs when the form loads.
5. Click +Actions. In the **Control** drop-down list select **Table**, and in the **Action** drop-down list select **Add Row**.



When the loop executes (on form load), the Add Row action gets executed 5 times.

6. Click **OK**.
7. Preview the form and see how it works.

Note that 5 rows get added to the table as the form loads.

To see how the loop executed, click **Log** in the Preview window.

Configure a Loop With an Array

In this example, let us configure a loop with an array of strings.

1. Drag and drop a table control onto your form's canvas. Then drag and drop an InputText control into the table's column.
2. Select the table control, and in the properties pane scroll down till you find **Events**. Click  to configure an event.
3. Select **On Load** from the drop-down list, and then click the editing icon.
4. In the event On Load window, click **Loop**. Select the type as **Data Definition** in the **Type** drop-down list, and enter the value as `list` in the **Value** field.

Note that *list* is already defined as an array of strings.

5. Click **+Actions**. From the **Control** drop-down list, select **Table**. From the **Action** drop-down list, select **Add Row**.
6. Configure another action. Click **+Actions**.
 - a. Select **Table** in the **Control Name** drop-down list, and specify **Last** in the **Which?** field.
 - b. In the **Control Name** that displays, select **InputText**, and then select **Value** in the **Action** drop-down list.
 - c. In the **Type** field that displays, select **Loop**, and then select **Value** from the **Value** drop-down list.

The screenshot shows the 'On Load' configuration window. At the top, there is a 'Type' dropdown set to 'Loop' and a 'Value' field containing 'list'. Below this, there are buttons for '+ Action' and '+ If'. A list of actions is shown, with 'Table Add Row' selected. Below that, another 'Table Last InputText Value Loop's Value' action is configured. The configuration table is as follows:

| Control Name | Which? | Control Name | Action |
|--------------|--------|--------------|--------|
| Table | Last | InputText | Value |
| Type | Value | | |
| Loop | Value | | |

Note that the Loop type can have three options as value.

- Index: 0-based index of the iteration (0, 1, 2, 3...)
 - Position: 1-based index of the iteration (1, 2, 3, 4...)
 - Value: The value of the current iteration. If the iterator is a number, this value will be same as Index.
7. Click **OK**.
 8. Preview the form and see how the loop works.
 - a. In the Preview window, click **Reload with Payload**.
 - b. In the Custom Payload window, specify an array, and click **OK**.

```

Payload JSON
{
  "payload": {
    "list": ["red", "blue"]
  }
}

```

- c. You can see that the loop executes with the specified array.

On form load, the table displays two rows with the values (red, blue) specified in the array.

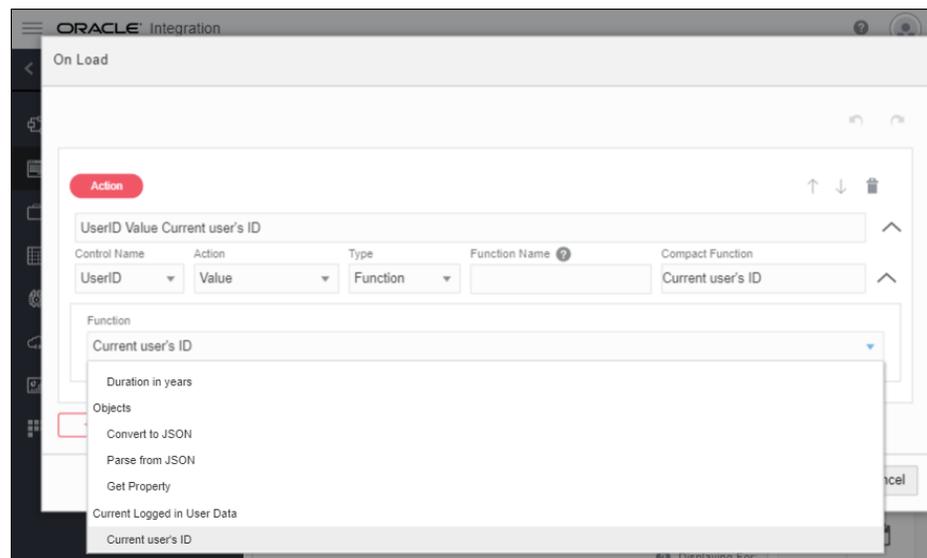
Example of Current Logged in User Data Function

Use the Current Logged in User Data function within actions in form controls to get information about users in forms.

The *Current Logged in User Data* function allows you to dynamically populate information about the current user's id, email, and the first, middle, and last name in forms. Additionally, you can also get the current user's manager's id, email, and the first, middle, and last name.

Let us explore how to use the Current Logged in User Data function with an example. In this example, let us configure a form with three fields (user's id, first, and last name) such that they automatically populate when the form loads. Then configure another field (manager's email) such that it gets populated when the user clicks a button GetManagerEmail.

1. Drag and drop an Input Text control onto the form's canvas. Then drag and drop two more Input Text controls and place them side by side.
 - a. Change the name of the first control to UserID and its label to User ID.
 - b. Change the names of the controls that you placed side-by-side to FirstName and LastName. Change their labels to First Name and Last Name.
2. Configure the User ID control.
 - a. Select the control, and in the properties pane scroll down till you find **Events**.
 - b. Click  to configure an event. Select **On Load** from the drop-down list and then click the editing icon.
 - c. In the event On Load window, click **+ Action**. In the **Control Name** field select **UserID** and then select **Value** from the **Action** drop-down list. In the **Type** field that displays, select **Function**.
 - d. In the **Function** drop-down list that displays, scroll down till you find *Current Logged in User Data*, and then select **Current user's ID**.

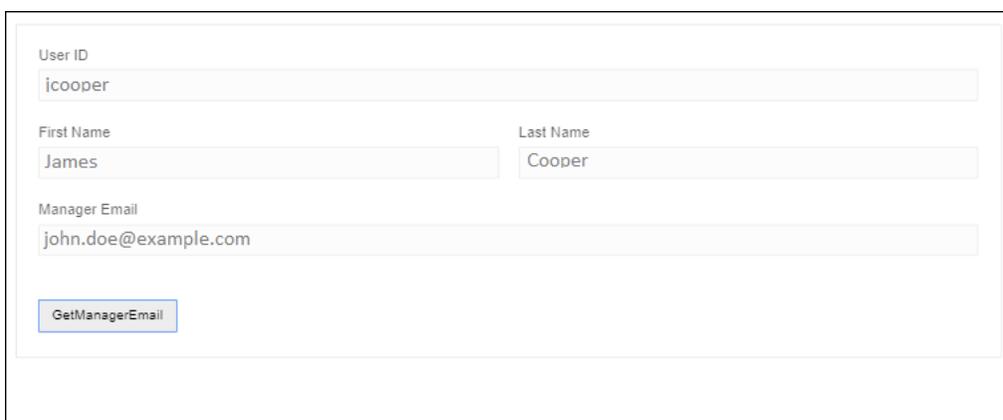


- e. Click **OK**.

3. Configure the First Name and Last Name controls.
 - a. For each control, select the control and repeat steps 2b to 2d. Ensure to select the correct control (First Name or Last Name) in the **Control Name** field.
 - b. For the First Name control, in the **Function** drop-down list, select **Current user's first name** under *Current Logged in User Data*.
 - c. For the Last Name control, in the **Function** drop-down list, select **Current user's last name** under *Current Logged in User Data*.
4. Drag and drop another Input Text control, and change its name to ManagerEmail and its label to Manager Email. Then drag and drop a Button control under it. Change the name and label of the button control to GetManagerEmail.
5. Configure the GetManagerEmail button.
 - a. Select the button, and in the properties pane scroll down till you find **Events**.
 - b. Click  to configure an event. Select **On Click** from the drop-down list and then click the editing icon.
 - c. In the event On Click window, select **ManagerEmail** from the **Control Name** drop-down list, and then select **Value** from the **Action** drop-down list. In the **Type** drop-down list that displays, select **Function**.
 - d. In the **Function** drop-down list that displays, scroll down till you find *Current Logged in User Data*, and then select **Current user's manager's email**.
 - e. Click **OK**.
6. Click **Preview**.

When the form loads, the user's id, first, and last name gets auto populated into the User ID, First Name and Last Name fields. The Get Manager Email field is empty.

Click the **GetManagerEmail** button. Note that the manager's email gets loaded into the **Manager Email** field.



The screenshot shows a form with the following fields and values:

- User ID: jcooper
- First Name: James
- Last Name: Cooper
- Manager Email: john.doe@example.com

Below the fields is a button labeled "GetManagerEmail".

Reuse Forms

You can reuse forms from the Forms Palette and use them either as reference or as copy in the forms that you build in Process. When you reuse a form as a reference, you can't change the properties or layout of the controls inside the form. When you reuse a form as a copy, all

the controls inside the reused form get converted into single controls. You can modify and update these controls like other controls in your web form.

To reuse forms:

1. Go to the Forms Palette. You can see the forms loaded into it.

If the form that you want to reuse isn't available in the Forms Palette, search for it. Click the search icon in the Forms Palette and type the form's name in the search field. If no forms in your application match the search, you will get a message stating this. Otherwise, the form gets loaded into the Forms Palette.

2. From the Forms Palette, drag and drop the form onto the canvas.

You can drop the form onto any part of the canvas. The layout of the reused form is preserved but depending on where you drop the form some panels may get added to it to maintain the form's layout.

3. By default, use the dropped form as a reference.

The reused form becomes a single control in the canvas and you can't change the properties of the controls inside the reused form.

- If **Auto Binding** is enabled, then a new type will be created in the form data definition. The data definition structure will be same as that of the reused form but it will be bound to the form that is built by you.
- If **Auto Binding** is disabled, then a warning icon will be displayed and it will indicate that the control is unbound. Define binding for the referenced form in the **Binding** field under **General** tab of the Properties pane. See [Bind Form Data with Controls](#).

4. Optionally, click **Detach** on top of the referenced form.

This will convert all the controls of the referenced form into single controls. You can modify and update these controls like other controls in your form.

This can be helpful when you want only certain parts and don't want some parts of a reused form. You can delete the unwanted controls and reuse only the controls that you want in building your web form.

 **Note:**

- When you detach an embedded (referenced) form on another form's UI, the controls corresponding to the embedded form appear as separate entities on the UI, but the data attributes related to these controls continue to be mapped under the data attribute of the embedded form. However, if you add a new control to the detached embedded form's section, the new control and its data attribute are treated as a part of the referencing form.
- Additionally, once an embedded form is detached on another form's UI, new controls added at the source of the embedded form aren't reflected on the referencing form's UI. However, the data attribute of the embedded form reflects this change in the referencing form's data definition section. The referencing form and its associated process task use this updated data attribute.

Save Web Form Data

You can capture web form data at multiple points in a process, and store it in a database or file system for archiving or auditing purposes. For example, capture a web form snapshot after a user submits a form and then again after a manager approves the form.

The save process isn't visible to the end user.

How it Works

- Every web form has a data object associated with it that gets created when you create a form.
- You use input/output data association to create a snapshot of the web form's data object as a binary system. Create a simple expression that uses a `Form.getWebForm` function to access the data. You can store the form's data as either a PDF (default) or an image (PNG, JPEG or JPG) file.
- You configure a service task to consume the data stream. In the service task's input data association, associate the stream data to your service method input parameter.
- Configure the service task's web service method to write the object's data to a location you select, such as a database or file system.

Form.getWebForm Function

Use the Form function in a simple expression to access the web form's data stream:

```
Form.getWebForm(webFormDataObject,[format],[formName],[presentationId])
```

- `webFormDataObject` - Can be a data object or task payload object
- `format` (String) - Valid values are PDF or PNG/JPEG/JPG (for image). Defaults to PDF
- `formName` (String) - Form associated with the data object. Form name is case sensitive
- `presentationName` (String) - optional - Presentation Id to be rendered for the form used

Function Examples

- `Form.getWebForm(taskWebFormDataObject, "PDF", "startWebForm")`
- `Form.getWebForm(taskWebFormDataObject, "PDF", "startWebForm", "8d4913a0-e237-dfc5-0eb5-831ddfb9e543")`

Configure Web Form Snapshots

You can configure service tasks and capture snapshots of the web form data, and save them locally as PDF or image files.

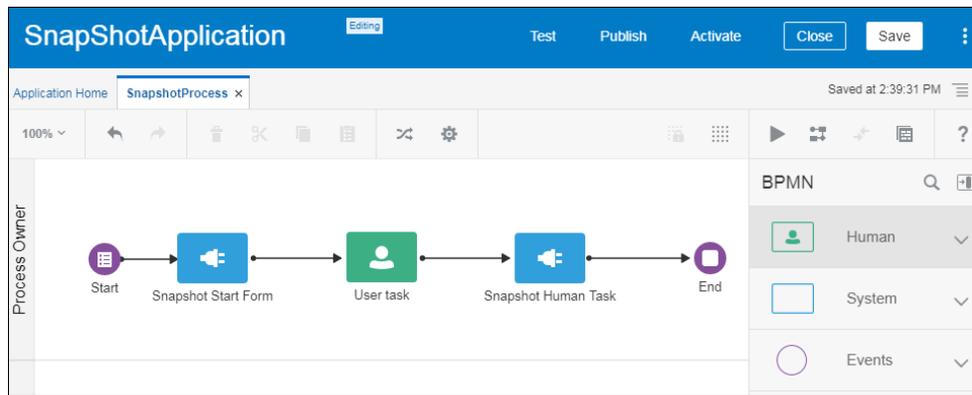
To configure web form snapshots:

1. Configure a process that includes a web form and a service task.

You can create snapshots for a form start event or a human task that uses a web form. For example, in the process shown below, each service task takes a web form snapshot:

- First, when the user clicks Submit for the start form

- Second, when the user clicks Approve for the human task

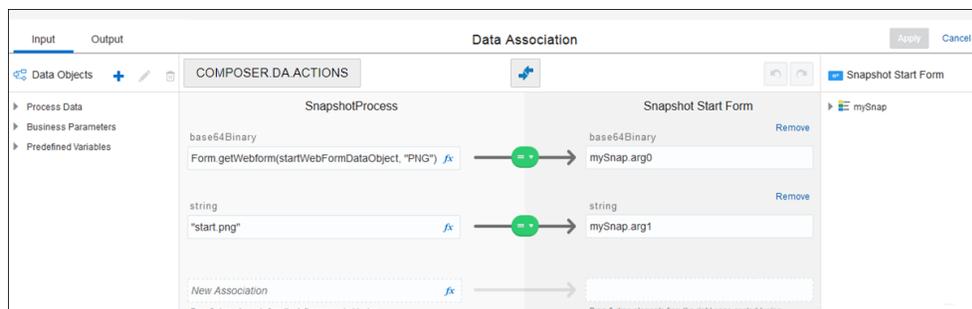


2. Set up an expression for the service task that performs the web form snapshot.
 - a. Select the service task, and select **Data Associations**.
 - b. On the input side of the Data Association editor, click the **fx** (Expressions) icon.
 - c. In the Expression Editor, click the **Operators** tab, expand **Form**, select the **Get WebForm** function, and click **Insert Into Expression**. Note that you can enter Form. in the expression field to trigger an autocompletion that lists the three available formats for the function.
 - d. In the expression, insert the data object you want to snapshot, and identify an output format of PDF or image (PNG/JPEG/JPG), and other optional parameters.

PDF is faster than image generation.

Note that PDF and images are static (read only), so radio button and video controls will not be rendered. Also, events will not work.
 - e. Click **Validate**.

This expression creates a binary stream of data you can associate with your service method input parameter.
3. Optionally, set up a second data association that stores a snapshot file name, which you can use to identify the file in your service method.

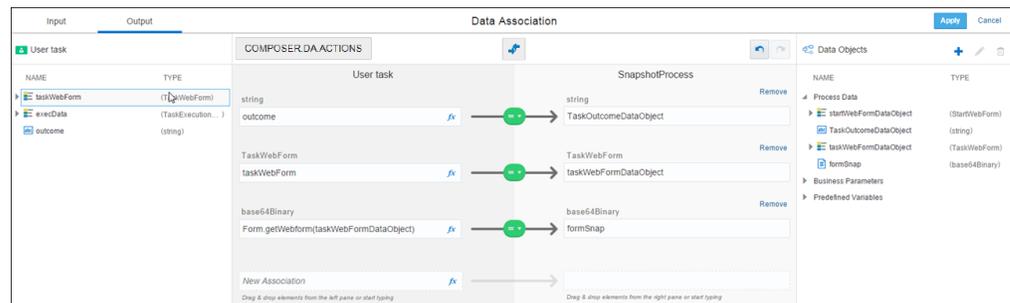


4. Optionally, store the snapshot stream in a base64Binary data object for later use as input in your service task data association.

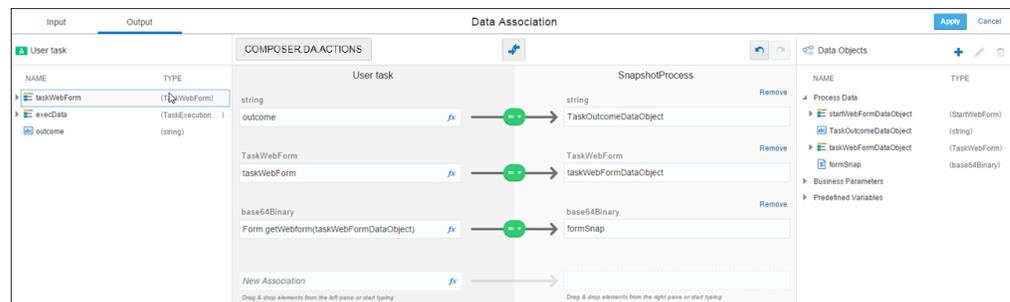
This step enables you to store the snapshot in the data object and then save all the form snapshots at once at the end of the process.

 - a. Create a base64binary process data object to store the captured data stream.

- b. Populate the data object using the output data association of the start form or human task. This method is executed when the start form is submitted or the human task is acted upon.



- c. Use this process data object in your service task's input data association.



The web form's data object always contains the form's latest data. If the same form is editable at multiple process stages, it is useful to store the form data at each level in a different process data object as described in step 4.

5. Create a web service for the service task that calls a method to write the file.

Below is an example of the service class implementation with a method that takes the web form data stream and stores the file in a local file system.

Example Method

```
public String mySnap(byte[] content,String name) {
    try {
        String fileName = "/scratch/"+name;
        File file = new File(fileName);
        BufferedOutputStream writer = new BufferedOutputStream(new
        FileOutputStream(file));
        writer.write(content);
        writer.flush();
        writer.close();
    }catch (Exception ex) {
    }
    return "Done";
}
```

6. Deploy and test the application, submitting and approving the form as needed. Check the location where the web service stored the form data and view the PDF or image files.

Create an External UI Connection

In a process application, you can configure human tasks to use an external UI—instead of a web form—for interacting with end users.

You can create and use external UI connections within dynamic as well as structured process tasks.

While using an external UI with a human task, note that you can only associate interfaces of applications that share the same identity domain as Oracle Integration.

To create a new external UI connection:

1. Within a process application, select **Forms** on the **Application Home** tab.
2. Click **Create**, and then select **New External UI**.
3. In the resulting dialog, enter a name for the external UI connection, provide the destination URL, and click **Next**.

 **Note:**

You can also provide path or template parameters while specifying the external URL, for example, `https://www.externalui.com/{instanceId}`.

4. Now, add the required query parameters for the URL. The following figure shows a few example parameters added:

Create connection to External UI

Basic Parameters Data

Parameter name Add

instanceId TEMPLATE

permitId QUERY

Previous Next

In addition to these parameters, the `task ID` parameter is passed with the value corresponding to the task ID. Click **Next**.

- In the **Data** tab, add custom payload attributes that you'll use to send/receive data from the external application. Unlike query parameters, here you can add complex business data types as well.

Note:

Through these attributes, you can use the data from an external UI in any task within a process. However, you must perform a POST operation using the REST API to save the changes made to the payload attributes in the external UI. See [Update the Task Payload](#).

- Click **Create**.
Now, perform the necessary data associations and assign values to path parameters, query parameters, and payload attributes from within a process. The final URL invoked will have the following format:

```
https://www.externalui.com/{instanceId}?permitId={permitid}&taskId={taskid}
```

Note:

You can edit the base URL of the external UI while activating your application.

To configure a human task with an external UI, see:

- [Use an External UI to Display Task Information](#) for structured processes.
- [Associate an External UI](#) for dynamic processes.

To integrate an external UI with the runtime process task list, see:

- [Integrate an External UI with the Process Task List](#) .

10

Manage Application Data

Almost all business processes require some type of data. For example, a purchase order application requires customer name, contact information, order number, items purchased, and payment details. Before you can use data in your application, you must define how it's structured and stored.

Topics:

- [About Managing Data](#)
- [Define Data](#)
- [Associate and Manipulate Data](#)

About Managing Data

Most business applications require users to create and manipulate data. In a travel request application, for example, a user enters data related to the request which includes information about the employee, to and from destinations, and other types of data. Additionally, an application may have to create and manipulate other data that is only used internally as part of the overall function of the application.

When creating business processes you must define the data that the Process application uses. Data is stored within a data object. Data objects are defined based on complex data types.

In Process, business types consist of data definitions referred to as business objects and business exceptions. A business object is a complex data type that groups together related data. For example, if you're creating an application that requires information about an employee, you may want to create a business object that stores the name, address, salary, and other information about the employee. Business objects are similar to the concept of classes used in object-oriented programming languages like Java. Create business objects from scratch using the Business Object Editor or automatically by importing an XSD. Business objects are also auto-generated when designing forms using the form-first design methodology and when importing services into an application. In these situations, business objects are referenced by forms and services and can only be deleted if the artifacts referring them are deleted. Business objects can be hierarchical - attributes can be simple or point to other business objects.

Define Data

To use data in your application, you must first define how it's structured and stored.

Topics:

- [Typical Workflow for Defining the Data Used Within a Business Process](#)
- [What You Can Do on the Business Types Page](#)
- [Work with Business Objects](#)

- [Work with Data Objects](#)
- [Create a Business Exception](#)
- [Create an Enum Object](#)
- [Work with Business Indicators](#)

Typical Workflow for Defining the Data Used Within a Business Process

Defining how data is stored and manipulated is part of the overall design and development of an application.

Use the following typical workflow to define the data used within your process application. The first three tasks are required; complete the remaining tasks as needed.

| Task | Description |
|---|---|
| Define the business types | Business types, such as business objects, business exceptions, and Enum objects, define the data structures used within your application. |
| Create data objects | Data objects store the information used within your business process. Associate each activity with your business process with the data objects it requires. |
| Configure data associations and transformations | Data associations define how information is passed between the flow elements in your business process. |
| Define expressions | Expressions manipulate the data used in your business process. |
| Create business indicators | Business indicators store the key performance information used in your business process. |
| Define input and output arguments | The flow events that you use to define your business process operations allow you to define input and output arguments. These input and output arguments define the process input and output. |

What You Can Do on the Business Types Page

The Business Types page provides the tools required to create and manage business types (business objects, enumeration (enum) objects, business exceptions), and also separately manage the life cycle of XML Schema Definition (XSD) files. You can use the business types in any process created for the selected application.

The Business Types page is divided into two sub views:

- Business Types
- Schema Files

Use the view option to alternate between views.



Business Types View

Use the Business Types view to perform the following tasks:

- [Create business objects.](#)
- [Create business exceptions.](#)
- [Create enum objects.](#)
- [Import business objects.](#)
- Click **Views**  to view the business types in Grid or List format. Grid view is the default.

Each business type displays: its name, type (*Business Object*, *Enum Object*, *Business Exception*), schema, and file name.



The business type is also identified by an image:

-  Manually created business objects
-  Form business objects
-  Imported business objects
-  Auto-generated business objects
-  Enum objects
-  Business exceptions
- Click the business type name to edit it in the [Business Types Editor](#). Click the file name to view the file details.
- Select **Include Auto-Generated** to view the auto-generated business types.

The number of auto-generated business objects that exists in the application display in parentheses.

Include Auto-generated (11)

- Delete manually created business types.

Note:

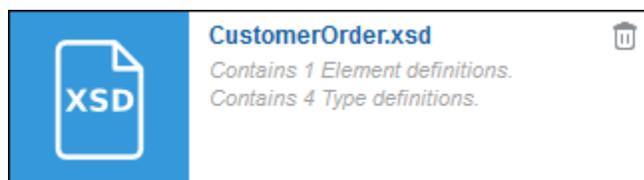
The **Delete** icon only appears for manually created business types. You can't delete business types that are being referenced by other artifacts.

Schema Files View

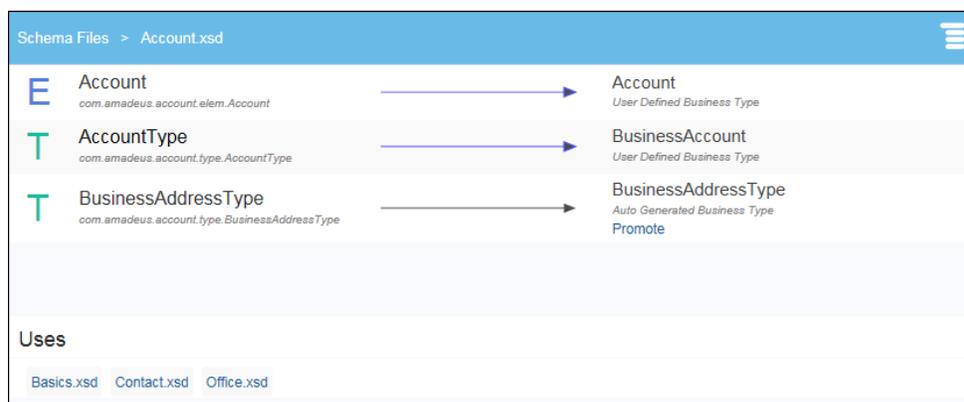
Use Schema Files view to perform the following tasks:

- [Import schema files.](#)
- Click **Views**  to view the schema files in Grid or List format. Grid view is the default.

Each schema file displays its name, the number of elements and types it contains, and an XSD image.



- Click the schema file name to view its details, and perform the following tasks:
 - [Promote auto-generated business types to business objects.](#)
This is allowed as auto-generated business types aren't explicitly created by the user.
 - [Upload a new version of the schema file.](#)
 - View the details of other schema files that are imported from the current one. (You can navigate from one to the other)



- Delete schema files.

Note:

You can't delete schema files that are being used by other business objects.

Work with Business Objects

Business objects let you group related types of data together to define the data structure required for your process applications. Create business objects manually or base them on a XML Schema Definition (XSD). After defining business objects, you can use them to define data objects to store the data within your application.

When manually creating business objects, you can define the hierarchical relationship between modules and objects, and the attributes used in the business object.

Note that you can't modify XSD-based business object properties. However, if you manually create a business object with an attribute pointing to another XSD-based business object, then you can modify the property values of that attribute.

Note:

If you create a business object based on a XSD, any changes made to the XSD (modify, add, delete elements) aren't reflected in the business object. After importing the modified XSD, you must delete the existing business object and then create a new business object on the modified XSD.

You can't create business objects based on in-line types defined in a WSDL file. However, read-only business objects are automatically created from the WSDL file when you create a web service connection. You can create data objects based on these read-only business objects.

Want to learn more about WSDL files and web service connections? See [Creating a Web Service Connection](#).

- **Modules**

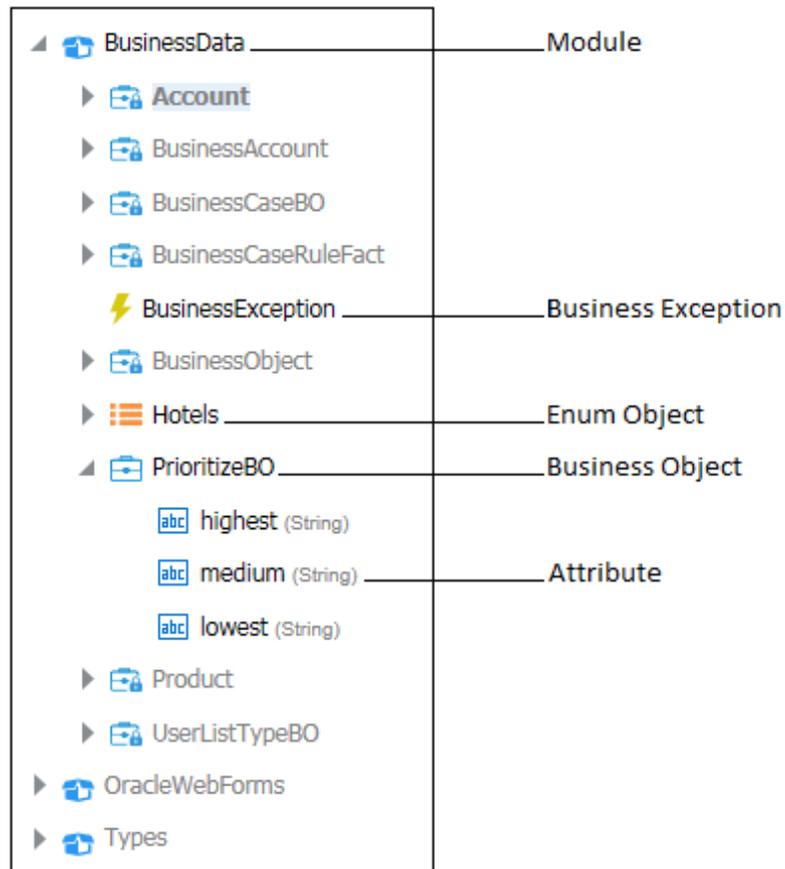
Modules are containers that enable you to create a hierarchical structure. Each business object must be contained in a module. When you create a new business object, it's automatically created within the `BusinessData` module. You can create additional business objects within a module or create additional modules.
- **Business objects**

Within a module you can define one or more business objects. A business object can contain other business objects.
- **Attributes**

Attributes define particular pieces of process data that are stored and can be shared among process activities. Attributes represent a characteristic of a real-world concept. For example, a business object that represents a person typically has `firstname` and `lastname` attributes.

You can also add documentation to your business objects and attributes. Adding documentation makes your data structures more understandable to other users who are collaborating on your application.

The following image shows an example of the hierarchical structure for the modules, business objects, and the business object attributes defined for an application.



An Enum object (enumeration) is a special type of business object that enables a data object to contain a set of predefined constants, such as one that includes the names of the days of the week (Sunday, Monday, Tuesday, and so on). See [Creating an Enum Object](#).

Business exceptions convey unexpected situations that can occur while running a business process. You can use Process to define business exceptions for your application. See [Creating a Business Exception](#).

Create a Business Object

Create new business objects (complex data types) manually. When creating your business objects you define the hierarchical relationship between modules and business objects, and the attributes used in the business object.

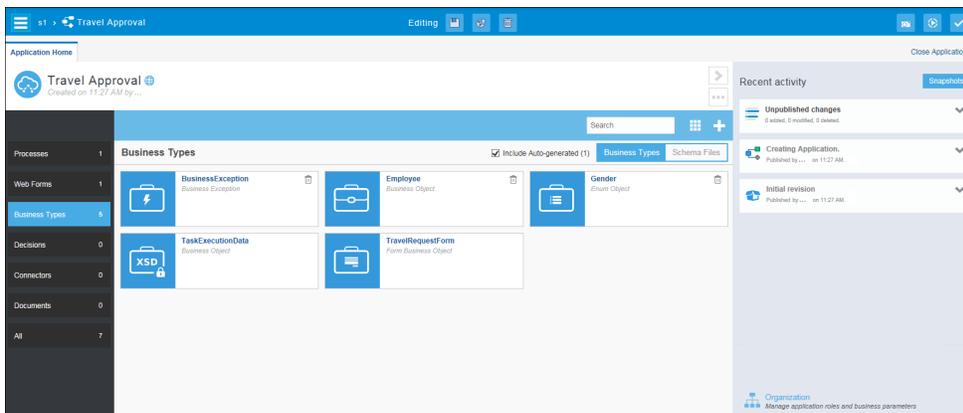


Note:

You can also import business objects based on XML schema (XSD) files. See [Import a Business Object from XML](#).

To create a business object:

1. On the Application Home tab, click **Business Types**.



2. Click **New Business Type** , then select **New Business Object** to manually create a business object.
3. In the Create Business Object dialog box, enter a name for the business object or use the default name, select or add a parent module, and then:
 - Click **Finish** to create your new business object.
 - Click **Next** to add attributes.

Create Business Object ? X

Business objects are complex data types that allow you to group related data.

* Name X I

* Parent Module V +

4. Click **Add Attribute**  to add attributes. Enter a name, click **Change Type** to change the data type from the default value *String*, click **Add**. Click **Finish** after you add all your attributes.

Create Business Object ? X

Attributes define particular pieces of process data that are stored and can be shared among process activities.

+ X

| Attribute name | Data Type |
|----------------|-----------|
| sku | int |
| brand | string |
| type | int |
| price | decimal |

Import a Business Object from XML

You can import business objects (complex data types) that are based on an XML Schema (XSD) file. During the process you define the hierarchical relationship between modules and business objects.

To import a business object:

1. On the Application Home tab, click **Business Types**.
2. Click **New Business Type** , select **Import Business Object**, and then select **From XML Schema**.
3. In the Import Business Object from XML Schema dialog box, enter a name for the business object (or use the default), select or add a parent module, and select an element or type defined in a schema file, or click **Import Schemas from File**  to import an XSD or ZIP file. You can then highlight the required element or type from the list of elements and types.

 **Note:**

Primitive types as top level elements is not supported.

4. Click **Import** to create a business object based on an XSD file.

Import a Business Object from JSON

You can import business objects (complex data types) that are based on a JSON instance or schema. During the process you define the hierarchical relationship between modules and business objects.

To import a business object from JSON:

1. On the Application Home tab, click **Business Types**.
2. On the Business Types page, click **Create** , select **Import Business Object**, then **From JSON**.
3. In the Import Business Object from JSON dialog box, enter a name and JSON for the business object, or click **Import from File**  to import a JSON file and the definitions contained within it.
4. Click **Next** to analyze and generate the JSON schema or instance, then make changes to it as needed. (Note that generation makes best guesses based on JSON contents.)
5. Click **Import** to create a business object based on the JSON instance or schema.

The following are the limitations while creating business objects using JSON schemas:

- Imported JSON Schemas can contain references only to the definitions contained within it. External references aren't allowed.
- Only one file or JSON schema is accepted at a time.
- The use of *required* in the JSON schema doesn't impact the *Not Null* property of the resulting business object.

- Imported JSON cannot have fields that are not NCName compliant, for example, they cannot contain special characters like @ \$ % & / + , ; , whitespace characters, or different parenthesis. Furthermore a JSON field cannot begin with a number, a dot, or a minus character although they can appear later in the field.
- Process doesn't currently support JSON natively. The JSON schema is converted to an XML Schema before generating the business object.

 **Note:**

Because JSON isn't supported natively, the conversion to XML Schema must be valid. The field names (which are translated to XML Schema element names) must be compliant with XML Schema. For example, you can't use names like `some:name` because the colon conflicts with the namespace prefix declaration.

When converting a JSON schema to an XML schema, the following conventions are used:

- All valid JSON schema blocks must define either a `type`, `enum`, or `$ref` keywords. JSON schema blocks not identified as `type`, `enum` or `$ref` are ignored.
- The root JSON schema block can only be `object` type, or an `enum` block.
- Only root definitions are taken into account.
- Local definitions must be referenced by `#/definitions/<local_name>`.
- Keywords `allOf`, `anyOf`, `oneOf` in the JSON schema map to `xs:anyType` in the resulting XML schema.
- The `not` keyword isn't supported.

The following is an example of a JSON schema:

```
{
  "id": "http://some.site.somewhere/entry-schema#",
  "$schema": "http://json-schema.org/draft-04/schema#",
  "description": "schema for an fstab entry",
  "type": "object",
  "required": [ "storage" ],
  "properties": {
    "storage": {
      "type": "object",
      "oneOf": [
        { "$ref": "#/definitions/diskDevice" },
        { "$ref": "#/definitions/diskUUID" },
        { "$ref": "#/definitions/nfs" },
        { "$ref": "#/definitions/tmpfs" }
      ]
    },
    "fstype": {
      "enum": [ "ext3", "ext4", "btrfs" ]
    },
    "options": {
      "type": "array",
      "minItems": 1,

```

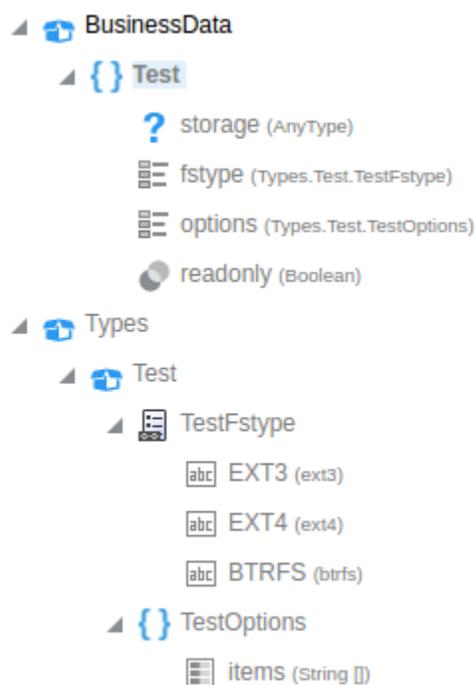
```
        "items": { "type": "string" },
        "uniqueItems": true
    },
    "readonly": { "type": "boolean" }
},
"definitions": {
    "diskDevice": {
        "properties": {
            "type": { "enum": [ "disk" ] },
            "device": {
                "type": "string",
                "pattern": "^/dev/[^/]+(/[^/]+)*$"
            }
        },
        "required": [ "type", "device" ],
        "additionalProperties": false
    },
    "diskUUID": {
        "properties": {
            "type": { "enum": [ "disk" ] },
            "label": {
                "type": "string",
                "pattern": "^[a-fA-F0-9]{8}-[a-fA-F0-9]{4}-[a-fA-F0-9]{4}-[a-fA-F0-9]{4}-[a-fA-F0-9]{12}$"
            }
        },
        "required": [ "type", "label" ],
        "additionalProperties": false
    },
    "nfs": {
        "properties": {
            "type": { "enum": [ "nfs" ] },
            "remotePath": {
                "type": "string",
                "pattern": "^(/[^/]+)$"
            },
            "server": {
                "type": "string",
                "oneOf": [
                    { "format": "host-name" },
                    { "format": "ipv4" },
                    { "format": "ipv6" }
                ]
            }
        },
        "required": [ "type", "server", "remotePath" ],
        "additionalProperties": false
    },
    "tmpfs": {
        "properties": {
            "type": { "enum": [ "tmpfs" ] },
            "sizeInMB": {
                "type": "integer",
                "minimum": 16,
                "maximum": 512
            }
        }
    }
}
```

```

    }
  },
  "required": [ "type", "sizeInMB" ],
  "additionalProperties": false
}
}
}

```

The resulting business object in Process appears as follows:



In certain business cases, you may need to update a JSON (either by updating it directly or importing a newer version) with changes that are *incompatible* with the current JSON. An incompatible JSON can include changes like:

- some elements or types have been removed
- some attributes in complex type have been updated to new type
- some attributes in complex type have been removed

While updating a JSON with incompatible changes, warning messages are displayed indicating the impact of the changes. For example, you may be notified that some business objects have to be modified or removed from the application to support the changes.

Import XML Schema Files

Import and store XML Schema (XSD) files, and then use its complex types or elements to create business objects. You can also update the root XSD and all its dependencies.

Note:

Updating your schema files using this import feature is different from the update feature in the XSD details view where you can only update the current schema file. Want to learn more about how to update the current schema file? See [Upload a New Version of an XML Schema File](#).

To import a XSD file:

1. On the Application Home tab, click **Business Types**.
2. On the Business Types page, click the **Schema Files** option, and then click **Import Schema Files** .
3. In the Upload Schema File dialog box, select one of the following, and then click **Validate**.
 - **Upload from file:** Click **Browse** to browse for a XSD or ZIP file.
 - **Use URL:** Provide the URL for your XSD or Zip file.

A summary of the files to be added are shown if the validation is successful.

Note:

If validation is unsuccessful, the validation errors or warnings are displayed. Validation warnings occur if some of the files that you want to upload override existing ones. In this case, you can upload and override existing files. Whenever an unexpected error occurs, the error is logged in the server log and the upload is cancelled. You can't import files that fail validation.

4. Optionally, click **Update** to update the root XSD and all its dependencies.
5. Click **Upload** to import your XSD file.

Upload a New Version of an XML Schema File

Process lets you upload and validate new versions of existing XML schema (XSD) files.

To upload a new version of an XSD file:

1. On the Application Home tab, click **Business Types**.
2. Click **Schema Files**.
3. Click the name of the schema file to open it.

4. Click **Menu**  and select **Upload New Version**.
5. In the Upload New Version dialog box, select one of the following options:
 - **Upload from file:** Click **Browse** to browse for your XSD file.
 - **Use URL:** Provide the URL for your XSD file.
6. Click **Validate**.

A summary of the files to be added are shown if the validation is successful.

 **Note:**

If validation is unsuccessful, then validation errors or warnings are displayed. Validation warnings occur if some of the files that you want to upload override existing ones. In this case, you can upload and override existing files. Whenever an unexpected error occurs, the error is logged in the server log and the upload is cancelled. You can't upload files that fail validation.

7. Click **Upload** to update your XSD file.

In certain business cases, the XSD you're trying to upload may contain changes that are *incompatible* with the current XSD. An incompatible XSD can include changes like:

- some elements or types have been removed
- some attributes in complex type have been updated to new type
- some attributes in complex type have been removed

While uploading an XSD with incompatible changes, you get warning messages in the Upload New Version dialog box to indicate the impact of the changes. For example, you may be notified that some business objects in the application were deleted or modified to support the incompatible changes.

Promote Schema Types and Elements to Business Objects

You can quickly create business objects (complex data types) that are based on specific XML Schema (XSD) types or elements by using the Promote feature available in the Schema Files — Details view.

To promote a schema type or element to a business object:

1. On the Application Home tab, click **Business Types**.
2. Click **Schema Files** to switch to the Schema Files view, which lists all schema files in the application.



3. Click the name of the required schema file to view its details.

| Schema Files > Basics.xsd | | |
|---------------------------|---|---|
| E | NotificationCriteria <small>com.amadeus.basics.elem.NotificationCriteria</small> | No Business Object Defined. Promote |
| E | NotificationCriteriaInstance <small>com.amadeus.basics.elem.NotificationCriteriaInstance</small> | No Business Object Defined. Promote |
| T | NotificationCriteriaInstanceType <small>com.amadeus.basics.type.NotificationCriteriaInstanceType</small> | No Business Object Defined. Promote |
| T | EmailType <small>com.amadeus.basics.type.EmailType</small> | EmailType Auto Generated Business Type Promote |
| T | NotificationCriteriaType <small>com.amadeus.basics.type.NotificationCriteriaType</small> | NotificationCriteriaType Auto Generated Business Type Promote |
| T | PhoneType <small>com.amadeus.basics.type.PhoneType</small> | PhoneType Auto Generated Business Type Promote |
| T | SystemIdentifierType <small>com.amadeus.basics.type.SystemIdentifierType</small> | SystemIdentifierType Auto Generated Business Type Promote |
| T | UserListType <small>com.amadeus.basics.type.UserListType</small> | UserListTypeBO User Defined Business Type |
| T | UserType | UserType Auto Generated Business Type |

This view shows the details of the schema file including:

- User-defined business types
- Auto-generated business types
- Types and Elements with no business object defined

You can create business objects based on the types and elements that aren't already used for other business objects, as well as on the auto-generated business types because they're not explicitly created by the user.

4. Click **Promote** for the type or element you want to base your new business object.

The Promote to Business Object dialog box opens.

Promote to Business Object ✕

* Name
 ✕

Schema Type

* Parent Module
 ▼ ▴ +

OK Cancel

5. Enter a name for the new business object.

The field is populated with the name of the selected business type or element, such as *BusinessAddressType*. You can use this name or change it.

6. Select a parent module from the drop-down list or click **New** to create a new module and then click **OK** to create your business object.

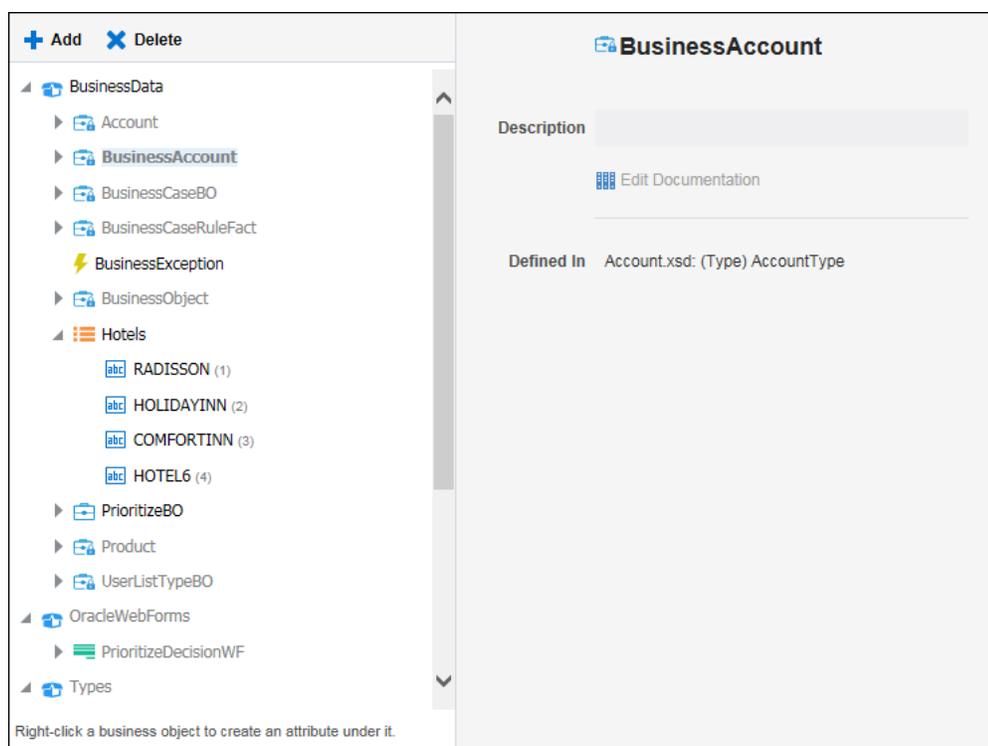
Manage Business Types

After you create a business object, click its name to go to the Business Types — Data Structure page. From this page, you can add or change the type of a business object's attributes or add documentation. You can also add modules, business objects, business exceptions, Enum objects and Enum items, and import business objects to the data structure.

The business types and related items you can add are based on what you have highlighted in the hierarchy.



When you highlight a business type or related item in the hierarchy, its details appear in the pane on the right. You can edit the details or add documentation. For imported business objects, the type behind the business object and the file where that type is defined is also displayed.



Right-click a business type or related item in the hierarchy to delete it, move it, or collapse a section of the hierarchy.

Think before you delete. You can't recover a business object, module, or attribute that has been deleted. When you delete a business object or module that contains other related items, they're also deleted. However, if the deleted business object or module contains an attribute based on a business object, that business object isn't deleted.

Work with Data Objects

Data objects are, in general, the variables used to store the information used by your business processes. They're defined during the design and implementation stage of a process. To complete a process application, you must ensure that you have associated each activity with the data objects it requires.

At runtime, data objects contain information that might be altered as users interact with your business process. Running processes can store, access, and manipulate data. The values stored within a data object can also be used to determine the branch that a process takes.

Process supports data objects that are based on either simple or complex data types. Which type of data type you base your data object on depends on the type of data it must handle.

- **Based on Simple Data Types**

Data objects can be based on simple data types. These are the core data types common in most programming languages. The following table lists the basic data types supported by Process.

| Type | Description |
|----------|--|
| Bool | Represents the logical values <i>True</i> or <i>False</i> . |
| Int | Represents an integer. For example: <i>23</i> , <i>-10</i> , <i>0</i> . |
| Decimal | Represents a number that can be expressed in decimal notation. For example: <i>3.14</i> , <i>62</i> , <i>0.023</i> . |
| Real | Represents a floating-point numeric value. For example: <i>2e-1</i> , <i>2.3E8</i> . |
| String | Represents a sequence of characters. For example: <i>This text is a string</i> . |
| Time | Represents a specific time expressed as: year-month-day hour:minute:second. For example: <i>2014-12-14 13:20:28</i> . |
| Interval | Represents a duration of time expressed as a number in years, months, days, hours, minutes, and seconds. For example: <i>1d3h30m</i> . |
| Binary | Used to store binary data, including images or videos. |

- **Based on Complex Data Types (Business Objects)**

Data objects based on complex data types (business objects) lets you create data structures that group different types of data. Business objects allow you to create data structures from data objects based on simple data types.

For example, you can create a business object called *employee* that contains different types of data for employee such as *id*, *name*, and *age*. The relationship between data objects based on simple data types and data objects based business objects is similar to the relationship between classes and instances in the Java programming language. The following tables show this relationship.

| Data Object | Simple Data Type |
|--------------|------------------|
| <i>id</i> | Int |
| <i>name</i> | String |
| <i>age</i> | Int |
| <i>sku</i> | Int |
| <i>price</i> | Decimal |
| <i>brand</i> | String |
| <i>type</i> | Int |

| Data Object | Structure |
|-----------------|--|
| <i>Employee</i> | <ul style="list-style-type: none"> – <i>id</i> (Int) – <i>name</i> (String) – <i>age</i> (Int) |
| <i>Product</i> | <ul style="list-style-type: none"> – <i>sku</i> (Int) – <i>brand</i> (String) – <i>type</i> (Int) – <i>price</i> (Decimal) |

Before creating a complex data object, you must first define the business object that defines the data structure.

Create a Data Object

You can create a data object based on a simple data object type or on a business object (complex data type). If you're creating a data object on a business object, you must create the business object first.

To create a data object:

1. On the Application Home tab, select a process, and then click **Data Objects**.
2. In the Data Objects dialog box, click **Add**.
3. Enter a name for your data object or use the default name, select a simple data object type or a data object based on a business object, click **Create**, then click **Close**.

 **Note:**

A default process data object with the name `TaskOutcomeDataObject` is automatically created for the human task outcome when a human task is created. The **Edit** and **Delete** links are disabled for this data object.

In the Data Objects dialog box, you can also edit or delete a data object. After editing or deleting a data object, validate your application to verify that there are no references to the changed or deleted data object.

See [Work with Data Objects](#).

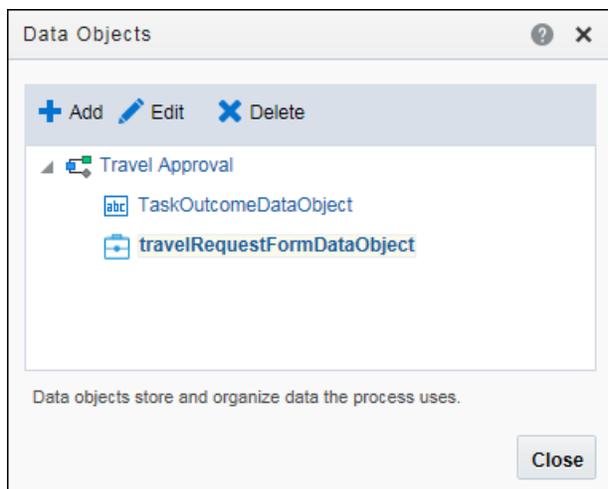
Edit or Delete a Data Object

You can edit or delete data objects.

To edit or delete a data object:

1. Go to the Application Home tab and make sure you're in **Edit** mode if you're editing a shared application.
2. Open the business process where you want to edit or delete a data object.
3. Click **Data Objects** and expand the list of data objects for your business process.
4. Select the specific data object.

This step activates the **Edit** and **Delete** icons.



- To edit the data object, click **Edit**, then change the name or data type as necessary.
- To delete the data object, click **Delete**.

5. Click **Close**.

After editing or deleting data objects, validate your application to verify that there are no references to the changed or deleted data objects.

After editing a data object, you must ensure that all references to it are still valid. For example, if you change a data object type from an integer to a string, you must verify that all of the expressions that use the data object still function correctly. If they don't function correctly, the application doesn't validate.

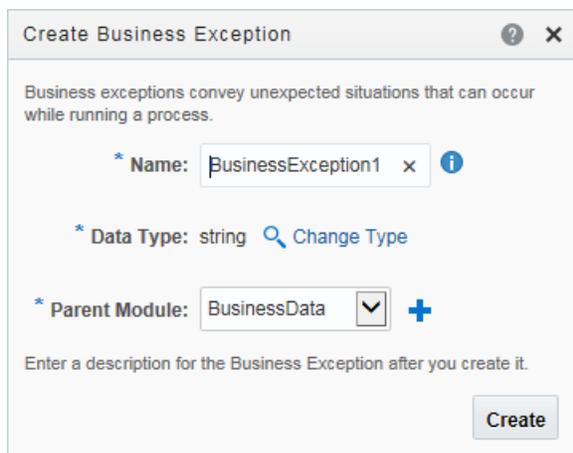
After deleting a data object, you must ensure that all references to it are removed. This includes any data associations and expressions that use the data object. If you don't remove references to the deleted data object, the application doesn't validate.

Create a Business Exception

Business exceptions are used to convey unexpected situations that can occur while running a business process. Define business exceptions for your application and use them in an error end event that is triggered under a certain condition, such as an invalid Id or poor credit rating.

To create a business exception:

1. On the Application Home tab, click **Business Types** to open the Business Types page.
2. Click **New**, then select **New Business Exception** to open the Create Business Exception dialog.



3. Enter a name for the new business exception.
The field is populated with a default name, such as *BusinessException*, *BusinessException1*, and so on. You can use this name or change it.
4. Click **Change Type** if you want to change the type. The default value is *String*.
5. Select a parent module from the drop-down list, and then click **Create** to create your new business exception.

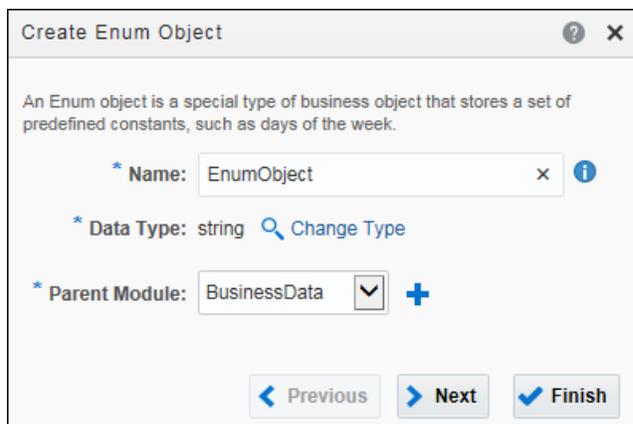
The parent module is *BusinessData* in most cases. If you're an advanced user, you can click **New** and create a new parent module.

Create an Enum Object

You can manually create Enum objects for your application. An Enum object (enumeration) is a special type of business object that enables a data object to contain a set of predefined constants, such as one that includes the names of the days of the week (Sunday, Monday, Tuesday, and so on).

To create an Enum object:

1. On the Application Home tab, click **Business Types** to open the Business Types page.
2. Click **New**, then select **New Enum Object** to open the Create Enum Object wizard.



3. Enter a name for the new Enum object.

The field is populated with a default name, such as *EnumObject*, *EnumObject1*, and so on. You can use this name or change it.

4. Click **Change Type** if you want to change the data type of the Enum items. The default value is *String*.
5. Select a parent module from the drop-down list.

The parent module is *BusinessData* in most cases. If you're an advanced user, you can click **New** and create a new parent module.

You now have two options:

- Click **Finish** to create your new Enum object.
- Continue to the next step to add Enum items.

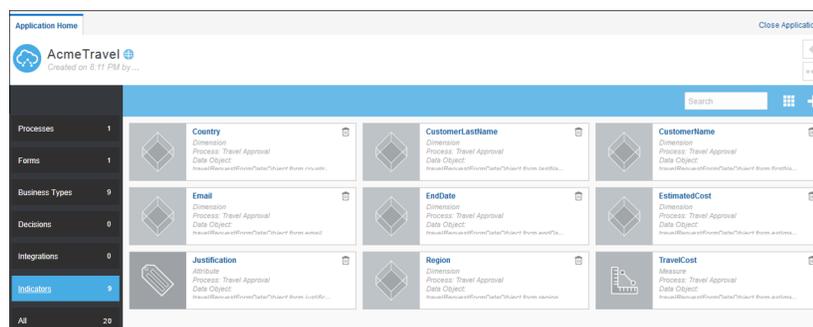
Perform the following steps to add Enum items.

6. Click **Next** to open the Enum Items dialog.
7. Click **New** to add an Enum item. Enter the name and value, then click **OK**.
Continue adding as many new Enum items as required.
8. Click **Finish** to create your new Enum object.

Work with Business Indicators

Business indicators provide a way to capture and display business metrics specific to your process. You designate selected data objects as business indicators, then plot them in business analytics dashboards. For example, for a sales quotation application, you might track key values that end users enter into a form, such as opportunity amount and region, then plot them in a chart displaying total opportunity amount by region.

The following image shows sample business indicators defined for an application:



You can create the following types of business indicators. Consider a data object's type when creating its business indicator.

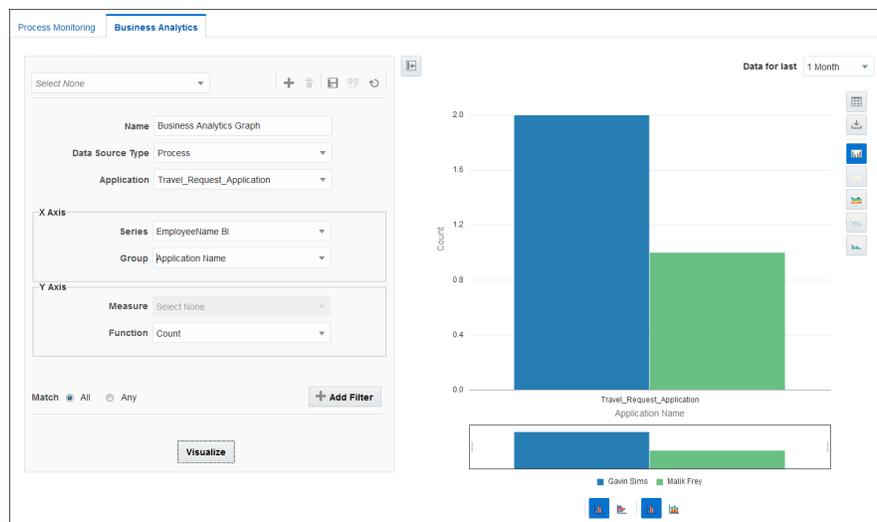
- Dimension indicators are available to plot as the X axis in charts, as a series and a group. Available dimension types include string, integer, decimal, Boolean, or date/time.
- Measure indicators are available to plot as the Y axis in charts, along with a function such as COUNT, SUM or AVG. They must be numeric (integer or decimal). You can set ranges for your numeric measure indicators.
- Attribute indicators are available to plot as filters in charts. Available attribute types include string, integer, decimal, Boolean, or date/time.

 **Note:**

The X axis, the Y axis and the filters are available in the business analytics section in Dashboards.

After activating the application, you can select business indicators to plot charts or graphs in Dashboards using business analytics options as described in [Create and View Business Analytics Dashboards](#). You must be assigned the Process Owner role, the Administrator role, or the Analytics Viewer role to create and view business analytics in Dashboards. Business analytics options are available in production deployments.

The following image shows a chart plotted using the business indicators defined in the application.

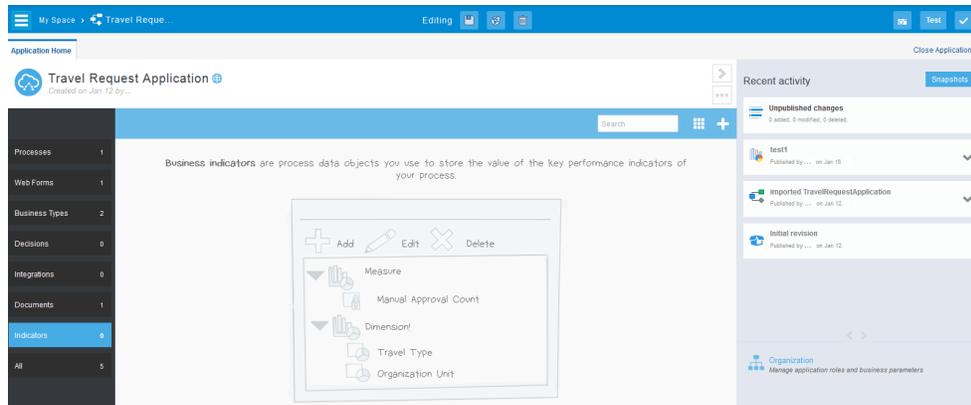


Create Business Indicators

You can create business indicators to capture key metrics for use in custom visualizations, as described in [Creating and Viewing Business Analytics Dashboard](#). You can create them using the Indicators pane or using data objects options in the process editor.

To create a business indicator from the Indicators pane:

1. Go to the Application Home tab and select **Indicators** on the left pane.



2. Click **Create** and select a business indicator type to add.

- **Attribute:** acts as a filter
- **Dimension:** acts as the X axis
- **Measure:** acts as the Y axis

 **Note:**

The X axis, the Y axis and the filters are available in the business analytics section in Dashboards.

3. Complete fields in the [Attribute](#), [Dimension](#), or [Measure](#) dialog box. In addition to entering a name for the indicator that will display in charts, identify the data object whose value will be captured.

Create Business Indicators from the Process Editor

To add business indicators from the process editor:

1. Go to the Application Home tab, click **Processes** on the left pane, and then select a process.
2. On the process editor, click **Data Objects**.
3. In the Data Objects dialog box, expand the process and select a data object. Click **Indicator** and select a business indicator type to add.
4. Select the business indicator type, and enter a name (without spaces) in the **Name** field for the indicator, or leave the default name, and then click **OK**. The **Type**, **Process**, and **Data Object** fields are filled in by default. Click **Close** to close the data object dialog.

Deploy and Test the Indicators

To deploy and test the business indicators in runtime:

1. [Deploy the application](#).
You must deploy the application to a production environment to create business analytics reports. Business analytics options aren't available in test deployments.
2. Run multiple process instances defined in your application.
3. Sign in as an administrator or process owner or the analytics viewer.

4. Click **Dashboards** in the navigation pane, then select the **Business Analytics** tab.
5. Create charts or graphs that use the business indicators, as described in [Create and View Business Analytics Dashboards](#).

Add Dimension Business Indicators

Create dimension indicators to identify data objects whose values you want to display in business analytics dashboards.

Use dimension-type [business indicators](#) to capture values that can be grouped for display. Dimension indicators are plotted as the X axis in business analytics charts. You apply them as a series and a group in charts. You can also define ranges for numeric dimensions with continuous values.

1. In the Dimension dialog box, enter a name (without spaces) in the **Name** field for the indicator, or leave the default name. This name displays in charts and graphs, appended with BI.

The **Type** and the **Process** gets selected by default. Unless the application contains multiple processes, only one process is available.

2. In the **Data Object** field, browse and select the object whose data you want to capture in the indicator, and click **OK**. Note that only when you select the data objects, the type gets displayed.
3. Optionally, define range entries for a dimension indicator with a type of decimal or integer.

For example, for a dimension indicator that displays loan amounts, you might create multiple ranges such as the following:

| Name | StartPoint | Range |
|--------|------------|-------------------|
| High | 50,000 | >=50,000 |
| Medium | 25,000 | 25,000 ... 50,000 |
| Low | 0 | 0 ... 25,000 |

4. Deploy and test indicators, as described in [Create Business Indicators](#).

Add Measure Business Indicators

Create measure indicators to identify numeric data objects whose values you want to display in business analytics dashboards.

Use measure-type [business indicators](#) to capture numeric values that can be measured, such as amounts or counts in a loan approval process. Measure indicators are plotted as the Y axis in business analytics charts. You apply standard aggregation functions to them, such as COUNT, SUM, and AVG. They must be integers or decimals.

1. In the Measure dialog box, enter a name (without spaces) in the **Name** field for the indicator, or leave the default name. This name displays in charts and graphs, appended with BI.

The **Type** and the **Process** gets selected by default. Unless the application contains multiple processes, only one process is available.

2. In the **Data Object** field, browse and select the object whose data you want to capture in the indicator, and click **OK**. Note that only when you select the data objects, the type gets displayed.
3. Deploy and test indicators, as described in [Create Business Indicators](#).

Add Attribute Business Indicators

Create attribute indicators to identify data objects whose values you want to use as filters in business analytics dashboards.

Use attribute-type [business indicators](#) to capture values that aren't suited as measure or dimension indicators, such as transaction IDs. You can use them to filter information or refer to other information in the process.

1. In the Attribute dialog box, enter a name (without spaces) in the **Name** field for the indicator, or leave the default name. This name displays in charts and graphs, appended with BI.

The **Type** and the **Process** gets selected by default. Unless the application contains multiple processes, only one process is available.
2. In the **Data Object** field, browse and select the object whose data you want to capture in the indicator, and click **OK**. Note that only when you select the data objects, the type gets displayed.
3. Deploy and test indicators, as described in [Create Business Indicators](#).

Associate and Manipulate Data

After you define business types and objects to store your application's data, you need to manage their use within flow elements.

Topics:

- [Configure Data Association](#)
- [Work with Transformations](#)
- [Work with Expressions](#)
- [Append to Array](#)
- [Filter Array Data Objects](#)

Work with Expressions

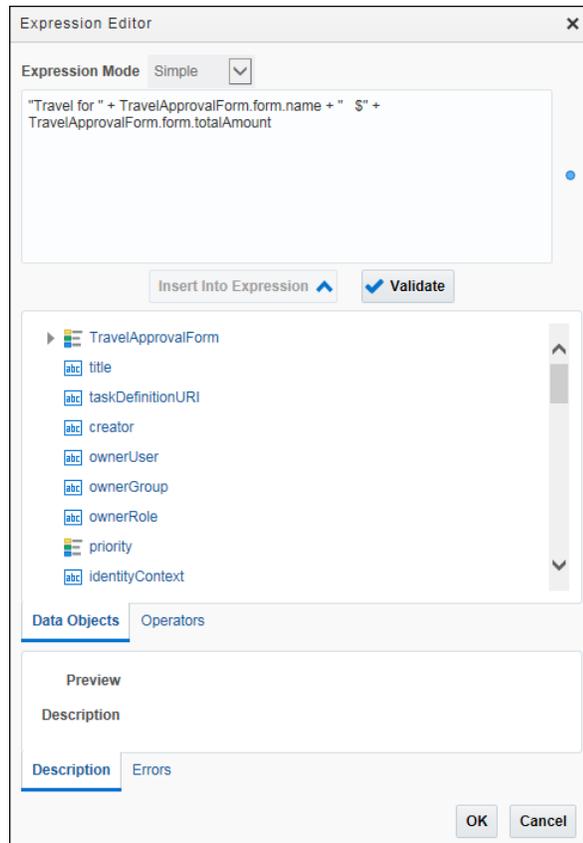
Use expressions to evaluate and perform calculations on data stored in data objects, using standard operators and functions.

Use expressions to change values passed to and from a sequence flow. You can create expressions when [configuring data associations](#) and [transformations](#), and when configuring properties for these flow elements:

- Human tasks: Use an expression to dynamically determine assignees
- Conditional sequence flows: Use an expression to define its condition
- Timer catch events: Use an expression to configure its time condition
- Notification tasks: Use expressions to define dynamic e-mail notification

- Sub-process tasks: Use an expression to conditionally call a subprocess

To create an expression, click the **Expression Editor**  icon or **Expressions Mode** field during [data association](#) or from the flow element's properties. The Expression Editor dialog box opens.



- In most cases, the **Expression Mode** field is set to Simple, but for some flow elements, Plain Text and XML Literal options are listed. The field below it displays the expression as you configure it.
- Create an expression by selecting the **Data Objects** or **Operators** tab, and selecting data objects and operators for your expression. Click **Insert Into Expression** to add a selected object or operator, or manually enter them.
- In the expressions field, enter a . (period) after a data object to view options available for its type (trigger a completion action).
- Click **Validate**, and results display in the expression field. The **Error** tab lists any errors found.

Using Functions

You can include the following functions in expressions, depending on data type. For example, you might use the string function to pass an integer value.

- **String** — text
- **int** — whole numbers

- **double** — decimal numbers
- **Boolean** — true or false
- **char** — single characters
- **byte** — eight bits, the smallest unit of data
- **short** — small whole numbers
- **long** — large whole numbers
- **float** — large decimal numbers
- **Date** — dates only
- **Time** — times only
- **DateTime** — dates and times

Function Guidelines and Examples

- If the field name is `inputDataObject`, and the type must be string, enter `string(inputDataObject)`.
- A field with a data type of `int` can contain integers only, or whole numbers. A field of type `double` or `float` can contain decimal numbers.
- In data association, if the input type is `int`, the output type can be any numeric data type, such as `int`, `double`, or `float`. However, if the output type is `int`, the input type must also be `int`, or an error occurs. Depending on needs, use the `round()`, `int()`, `floor()`, or `ceil()` functions.
- If the field name is `loanAppDataObject.form.income`, for basic forms, change it to `round(loanAppDataObject.form.income)` or `loanAppDataObject.form.income.round()` and for web forms, change it to `round(loanAppDataObject.income)` or `loanAppDataObject.income.round()`.

About Simple Expressions

Simple expressions are defined using a basic expression language and support. You can use these operators to write expressions and conditions to define your process flow. Generally these expressions perform their calculations based on the data objects in your business process. You can write expressions and conditions using the value of the data objects, but you can't explicitly modify the value within the data object.

Here are some examples of expressions using operators:

- `totalAmount - discount`
- `activationCount > 3`
- `unitsSold <= 1200`

Operator Precedence

Operator precedence defines the order in which the compiler evaluates operators. You can change operator precedence in an expression by using parentheses. In Process, the operator precedence is:

- Addition, Subtraction
- Multiplication, Division, Remainder
- Plus, Minus

- Less than, Greater than, Less than or equal to, Greater than or equal to
- Equals, Not equals
- Not
- Conditional And
- Conditional Or

Arithmetic Operators

| Operator | Name | Description |
|----------|----------------|---|
| + | Addition | Adds numeric data types; also concatenates strings |
| - | Subtraction | Subtracts numeric data types |
| * | Multiplication | Multiplies numeric data types |
| / | Division | Divides numeric data types |
| % | Remainder | Calculates the remainder of a division in which the divisor doesn't exactly divide the dividend |
| () | Precedence | Indicates the order of evaluation of an arithmetic expression |

Unary Operators

| Operator | Name | Description |
|----------|-------|--|
| + | Plus | Has no effect on the value of the numeric operand. Use it to explicitly indicate that a certain value is positive. |
| - | Minus | Negates an arithmetic expression. |
| ! | Not | Logical complement operator. Negates the value of a Boolean expression. |

Equality and Relational Operators

| Operator | Name | Description |
|----------|--------------------------|--|
| = or == | Equal to | Returns true if the first operand is equal to the second operand |
| != | Not equal to | Returns true if the first operand isn't equal to the second operand |
| > | Greater than | Returns true if the first operand is greater than the second operand |
| >= | Greater than or equal to | Returns true if the first operand is greater than or equal to the second operand |
| < | Less than | Returns true if the first operand is less than the second operand |
| <= | Less than or equal to | Returns true if the first operand is less than or equal to the second operand |

Conditional Operators

| Operator | Name | Description |
|----------|-----------------|--|
| and | Conditional And | Returns true if both operands evaluate to true |
| or | Conditional Or | Returns true if either operand evaluates to true |

String Operators

| Operator | Description | Usage Expression | Usage Result |
|------------|---|-------------------------------|--------------|
| + | String concatenation | "pine" + "apple" | "pineapple" |
| == | Equals | "apples" == "apples" | true |
| != | Not equals | "apples" != "oranges" | true |
| > | Greater than | "word" > "work" | false |
| >= | Greater than or equals | "work" >= "work" | true |
| < | Less than | "word" < "work" | true |
| <= | Less than or equals | "work" <= "work" | true |
| contains | Returns true if the first argument string contains the second argument string; otherwise returns false | "caramel".contains("ram") | true |
| endsWith | Returns true if the first argument string ends with the second argument string; otherwise returns false | "immutable".endsWith("table") | true |
| length | Returns the number of characters in a string | "house".length() | 5 |
| lowerCase | Returns a string with all the characters in the argument converted to lower-case representation | "Example".lowerCase() | "example" |
| startsWith | Returns true if the first argument string starts with the second argument string, otherwise returns false | "caramel".startsWith("car") | true |
| substring | Returns the substring of the first argument starting at the position specified in the second argument and continuing to the end of the string | "care".substring(1) | "are" |
| substring | Returns the substring of the first argument starting at the position specified in the second argument with length specified in the third argument | "care".substring(0,3) | "car" |
| upperCase | Returns a string with all the characters in the argument converted to upper-case representation | "Example".upperCase() | EXAMPLE |

Integer Operators

| Operator | Description | Usage Expression | Usage Result |
|----------|----------------|------------------|--------------|
| + | Addition | 2 + 8 | 10 |
| - | Subtraction | 7 - 4 | 3 |
| * | Multiplication | 3 * 4 | 12 |

| Operator | Description | Usage Expression | Usage Result |
|----------|--|------------------|--------------|
| / | Division | 3 / 2 | 1.5 |
| % | Remainder | 3 % 2 | 1 |
| == | Equals | 12 == 13 | false |
| != | Not equals | 12 != 13 | true |
| > | Greater than | 15 > 16 | false |
| >= | Greater than or equals | 15 >= 15 | true |
| < | Less than | 12 < 10 | false |
| <= | Less than or equals | 12 <= 12 | true |
| abs | Returns the absolute value of a number | - 6 | 6 |

Non-integer Operators

| Operator | Description | Usage Expression | Usage Result |
|----------|--|------------------|--------------|
| floor | Returns the largest (closest to positive infinity) number that isn't greater than the argument and is an integer | floor(5.60) | 5 |
| ceil | Returns the smallest (closest to negative infinity) number that isn't less than the argument and is an integer | ceil(5.60) | 6 |
| round | Returns the number that is closest to the argument and is an integer | round(5.60) | 6 |

Date and Time Operators

| Operator | Description |
|----------|---|
| + | Addition (valid only when the second argument is a duration) |
| - | Subtraction (valid only when the second argument is a duration) |
| == | Equals |
| != | Not equals |
| > | Greater than |
| >= | Greater than or equals |
| < | Less than |
| <= | Less than or equals |
| format | Returns the formatted string of date-time using the provided format picture |
| year | Returns a number representing the year component of the date-time argument |
| month | Returns a number representing the month component of the date-time argument |
| day | Returns a number representing the day component of the date-time argument |
| hours | Returns a number between 0 and 23, both inclusive, representing the hours component of the date-time argument |
| minutes | Returns a number between 0 and 59, both inclusive, representing the minutes component of the date-time argument |

| Operator | Description |
|----------|---|
| seconds | Returns a number between 0 and 59, both inclusive, representing the seconds component of the date-time argument |
| timezone | Returns an interval value, representing the time offset from UTC |

Boolean Operators

| Operator | Description | Usage Expression | Usage Result |
|----------|---|------------------|--------------|
| == | Equals | true == true | true |
| != | Not equals | true != false | true |
| and | Conditional — And | true and false | false |
| or | Conditional — Or | true or false | true |
| not | Logical complement operator, inverts the value of a Boolean expression. | not true | false |

Duration Operators

| Operator | Description |
|----------|------------------------|
| == | Equals |
| != | Not equals |
| > | Greater than |
| >= | Greater than or equals |
| < | Less than |
| <= | Less than or equals |

Base64Binary Operators

| Operator | Description |
|----------|-------------|
| == | Equals |
| != | Not equals |

Array Operators

| Operator | Description |
|----------|---|
| [] | Access a particular element into the array |
| == | Equals |
| != | Not equals |
| length | Returns the number of elements contained within the array |

Other Operators

| Operator | Description |
|----------|-------------|
| == | Equals |
| != | Not equals |

Special Constants

| Constants | Description |
|-----------|------------------|
| null | Null value |
| true | Logical true |
| false | Logical false |
| 'now' | Current dateTime |

Casting

In some cases, it could be desirable to bypass the type validation in order to assign types that aren't necessarily compatible. For example, you may want to assign an 'int' value to a 'string' one and in order to do that you can use the conversion operation like this:

<conversionTypeName> (<valueToConvert>)

where the 'conversionTypeName' is the type you want to see as the value.

Here are some conversion examples:

- string(myIntDO)
- int(myStringDO)
- duration(mystringDO)



Note:

You can only cast to primitive types, so the 'conversionTypeName' will only accept those that have a valid value.

Assigning two values that are incompatible will result in a runtime error.

Identity Service

| Function | Description | Available | Usage Function Prototype | Usage Example |
|------------|--|--------------|--|--|
| getManager | Returns a string containing the manager of the specific user | Process wide | IdentityService.getManager(<userName:string>): string | IdentityService.getManager("wfaulkner") |
| getManager | Returns a string containing the manager of a certain user in the realm specified | Process wide | IdentityService.getManager(<userName:string>,<realm:string>): string | IdentityService.getManager("wfaulkner", "myRealm") |

Human Task

| Function | Description | Available | Usage Function Prototype | Usage Example |
|------------------|-------------|--------------------|--------------------------------------|------------------------------|
| getPerformer | | Assignee Selection | HumanTask.getPerformer(): string | HumanTask.getPerformer() |
| getLastPerformer | | Assignee Selection | HumanTask.getLastPerformer(): string | HumanTask.getLastPerformer() |

Form

| Function | Description | Available | Usage Function Prototype | Usage Example |
|------------|---|--------------|---|--|
| getWebform | Returns a base64 encoded value representing the PDF image of the specified webform data object | Process wide | Form.getWebform(<webFormDataObject:catalogObject>): base64 | Form.getWebform(myWebformDO) |
| getWebform | Returns a base64 encoded value representing the image of the specified webform data object in the format passed as the second argument (valid values are "PDF" or "PNG") | Process wide | Form.getWebform(<webFormDataObject:catalogObject>,<format:string>): base64 | Form.getWebform(myWebformDO, "PNG") |
| getWebform | Returns a base64 encoded value representing the image of the specified webform data object having the name determined in the third argument in the format passed as the second argument (valid values are "PDF" or "PNG") | Process wide | Form.getWebform(<webFormDataObject:catalogObject>,<format:string>,<webFormName:string>): base64 | Form.getWebform(myWebformDO, "PNG", "MyWebformDO") |

Document Service

| Function | Description | Available | Usage Function Prototype | Usage Example |
|--------------------------|--|--------------|--|--|
| getDocumentAssetProperty | Get some property for a specific document asset, which can be one of the following: Id, Type | Process wide | DocumentService.getDocumentAssetProperty(<propertyName:string>,<documentAssetName:string>): string | DocumentService.getDocumentAssetProperty("Id", "myFolder") |

Get or Else

| Function | Description | Available | Usage | Function Prototype | Usage Example |
|----------|---|--------------|--|---|---------------|
| boolean | Request an object property along with a backup value. The backup value is used if the first argument corresponds to a missing node or an uninitialized value. This function set is useful while dealing with an untrustworthy data source, where initialization isn't guaranteed. | Process wide | GetOrElse.boolean(<expression:boolean>,<fallbackValue:boolean>): boolean | GetOrElse.boolean(myDO.boolAtt, true) | |
| string | | Process wide | GetOrElse.string(<expression:string>,<fallbackValue:string>): string | GetOrElse.string(myDO.stringAtt, "User") | |
| decimal | | Process wide | GetOrElse.decimal(<expression:decimal>,<fallbackValue:decimal>): decimal | GetOrElse.decimal(myDO.decimalAtt, 25.9) | |
| byte | | Process wide | GetOrElse.byte(<expression:byte>,<fallbackValue:byte>): byte | GetOrElse.byte(myDO.byteAtt, byteVar) | |
| short | | Process wide | GetOrElse.short(<expression:short>,<fallbackValue:short>): short | GetOrElse.short(myDO.shortAtt, shortVar) | |
| int | | Process wide | GetOrElse.int(<expression:int>,<fallbackValue:int>): int | GetOrElse.int(myDO.intAtt, 12) | |
| long | | Process wide | GetOrElse.long(<expression:long>,<fallbackValue:long>): long | GetOrElse.long(myDO.longAtt, 1000) | |
| double | | Process wide | GetOrElse.double(<expression:double>,<fallbackValue:double>): double | GetOrElse.double(myDO.doubleAtt, 25.3d) | |
| float | | Process wide | GetOrElse.float(<expression:float>,<fallbackValue:float>): float | GetOrElse.float(myDO.floatAtt, floatVar) | |
| binary | | Process wide | GetOrElse.binary(<expression:base64Binary>,<fallbackValue:base64Binary>): base64Binary | GetOrElse.binary(myDO.binaryAtt, binaryVar) | |
| integer | | Process wide | GetOrElse.integer(<expression:integer>,<fallbackValue:integer>): integer | GetOrElse.integer(myDO.integerAtt, 12) | |
| time | | Process wide | GetOrElse.time(<expression:time>,<fallbackValue:time>): time | GetOrElse.time(myDO.timeAtt, '9:20') | |
| date | | Process wide | GetOrElse.date(<expression:date>,<fallbackValue:date>): date | GetOrElse.date(myDO.dateAtt, '1988/08/08') | |
| dateTime | | Process wide | GetOrElse.dateTime(<expression:dateTime>,<fallbackValue:dateTime>): dateTime | GetOrElse.dateTime(myDO.dateTimeAtt, '1988/08/08 9:20') | |
| duration | | Process wide | GetOrElse.duration(<expression:duration>,<fallbackValue:duration>): duration | GetOrElse.duration(myDO.durationAtt, '5s') | |

 **Note:**

The Get or Else function set is available only for regular data associations within a structured process. It isn't available either for data transformations or dynamic processes.

Append to Array

An array is a series or list of data elements of the same type. You can add a single data element to an array with the **Append** action available in the Data mapper.

Append is available in the Data Association editor for both structured and dynamic processes.

Keep in mind the following:

- The target expression type must be always an array.
- The source expression type must be of the same data type as the target array.

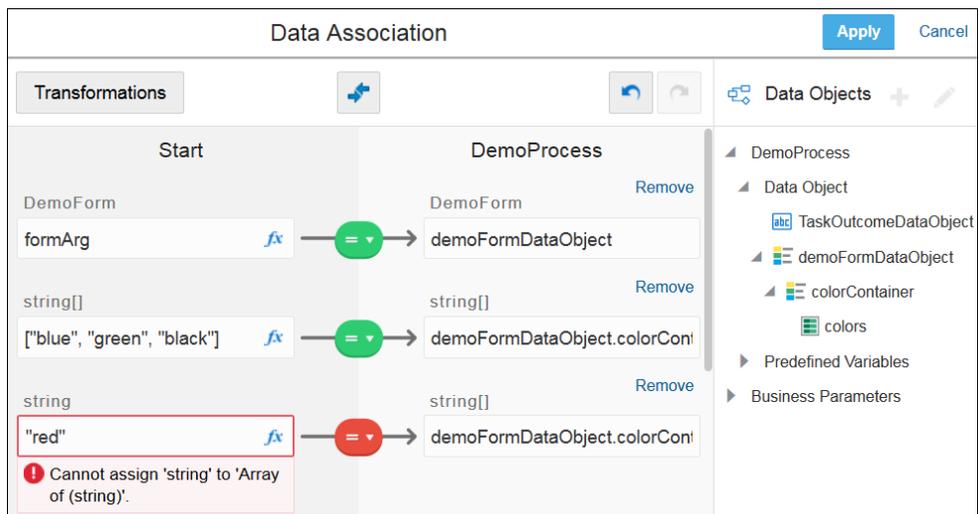
The only exception is when the source data type can be assigned to the target array. For example, you can append an integer to an array of type double since you can assign an integer to a double.

- Both the source and target expressions have to be valid.

Suppose you have a list of colors: blue, green, and black in a form. Now you want to add another color red to the list. Let's append "red" (string) to the list of colors (string of type array) in design time, validate and test the application, and finally check if the append action is successful in runtime.

1. In design time, open the Data Association editor. Expand Data Object in the right pane and then expand **colorContainer**.
2. Drag and drop **colors** (array) to the target field in the Data Association editor.
3. Enter "red" in the source field.

Notice that the association icon turns red and you get an error message.

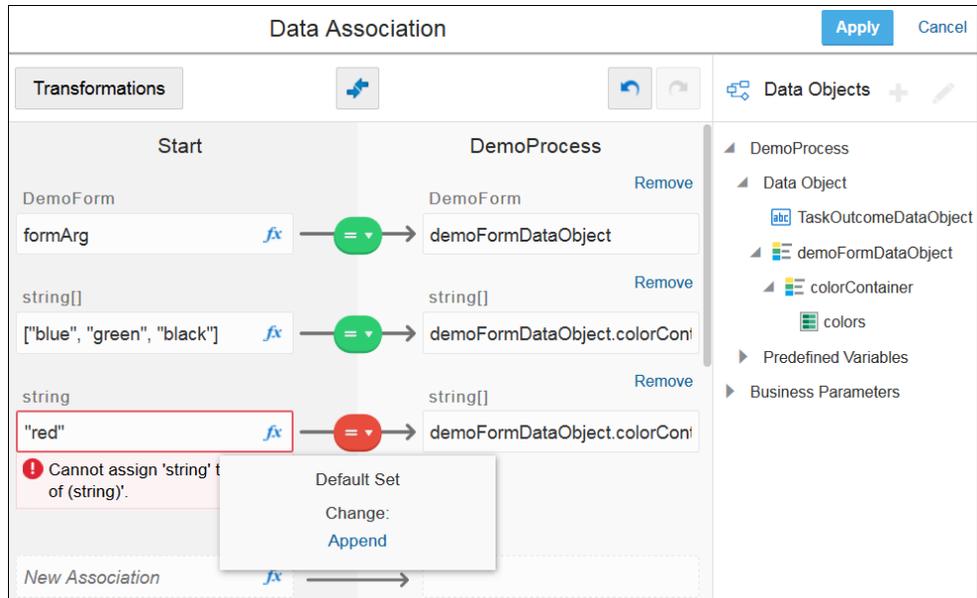


The screenshot shows the Data Association editor interface. It features a 'Transformations' tab and a 'Data Objects' pane on the right. The main workspace is divided into 'Start' and 'DemoProcess' sections. In the 'Start' section, there are three data associations:

- 1. A 'formArg' (type 'fx') associated with 'demoFormDataObject' (type 'string[]').
- 2. A '["blue", "green", "black"]' (type 'fx') associated with 'demoFormDataObject.colorConl' (type 'string[]').
- 3. A '"red"' (type 'fx') associated with 'demoFormDataObject.colorConl' (type 'string[]').

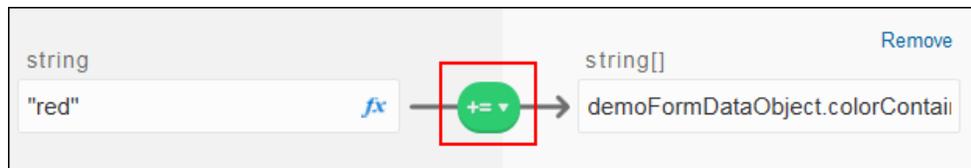
 The third association icon is red, and a red error message box is displayed below it: "Cannot assign 'string' to 'Array of (string)'". The 'Data Objects' pane on the right shows a tree view with 'DemoProcess' expanded to 'Data Object', then 'demoFormDataObject', and finally 'colorContainer' expanded to show 'colors'.

4. Click the association icon. A drop-down menu appears.



5. Click **Append**.

The association icon turns green indicating that the append action was successful.



6. Click **Apply**.
7. Test activate and try the application in test mode.
8. In runtime, open, complete, and submit the start task.
You get a message that an instance has been created.
9. Click **My Tasks** and open the task that has the color list.
Notice that red has been added to the list of colors in the form.

The screenshot shows a web form with a header containing 'Submit', 'Save', and 'More' buttons. Below the header is a section titled 'Form'. Inside the form, there is a container labeled 'ColorContainer' which contains an array of items. The array is labeled 'colors' and has a trash icon and a plus sign to its right. The array contains four items, each with a checkbox and the label 'itemType'. The first item has the value 'red' and is highlighted with a red box. The second item has the value 'black', the third has 'green', and the fourth has 'blue'.

You have successfully appended a string to an array in this example.

Filter Array Data Objects

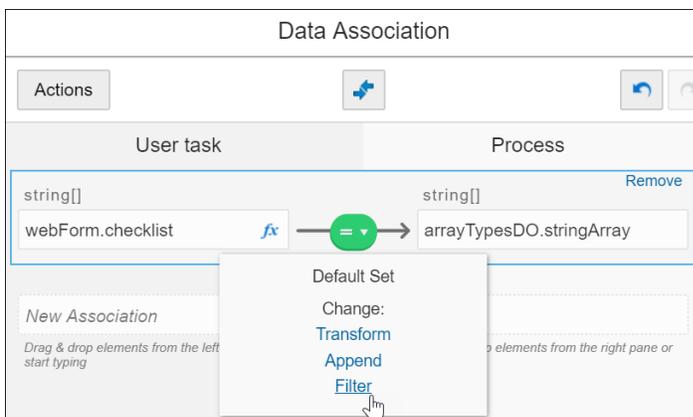
While defining data associations between process components, you can selectively filter elements from a source array object and assign them to a target array.

In the Data Association editor of a structured process, a dynamic process, or a transformation, an option to define a filter appears if the following conditions are met:

- The target expression type is an array.
- The source expression type is an array. The base type of the source is assignable to the base type of the target (for example, integer to double assignments).
- Both the source and target expressions are valid.

To define a new filter:

1. Click the **Data Association** icon and select **Filter**.

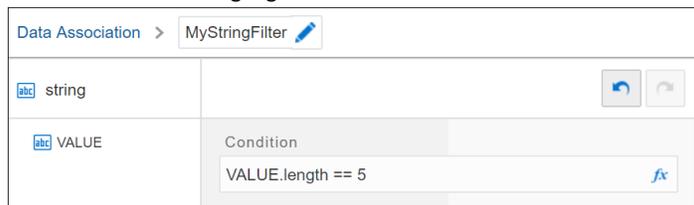


2. Select the **Create** action in the resulting window and provide a suitable name for the filter, for example, MyStringFilter, and click **Create** again. A window to define the filter condition appears.

Note:

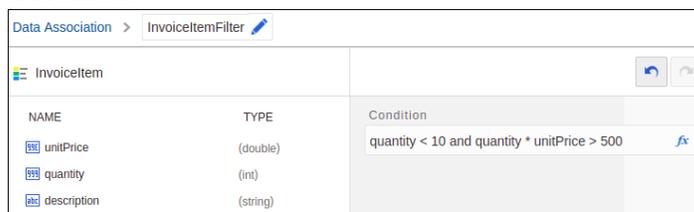
If a suitable filter already exists, you can select or edit it from the Create Filter window.

3. Based on the type of your array, define a filter condition using the available operators. For a simple array object (of the type integer, string, or double), an element named **VALUE** is present in the window, which represents each element of the source array.
 - a. Drag and drop the **VALUE** element into the **Condition** field to define an expression. If needed, click the **fx** (Expression) icon and use the Expression editor. The following figure shows a filter condition defined for a string array:



In this case, only the elements of the source array of length 5 are assigned to the target array.

- b. For a complex array object, you can define a condition comprising all attributes it contains:



 **Note:**

Currently, while defining a filter condition for a complex array object, you must type in each array element and its associated condition into the **Condition** field. Drag and drop functionality is supported only for a single object or element.

4. Save the filter condition.
5. To view or edit any filter defined within the process, click **Actions** in the Data Association editor.
6. Finally, click **Apply** to save all data associations and filters.
In runtime, target arrays are populated according to the filter conditions you've defined.

Define Conditions for Data Associations

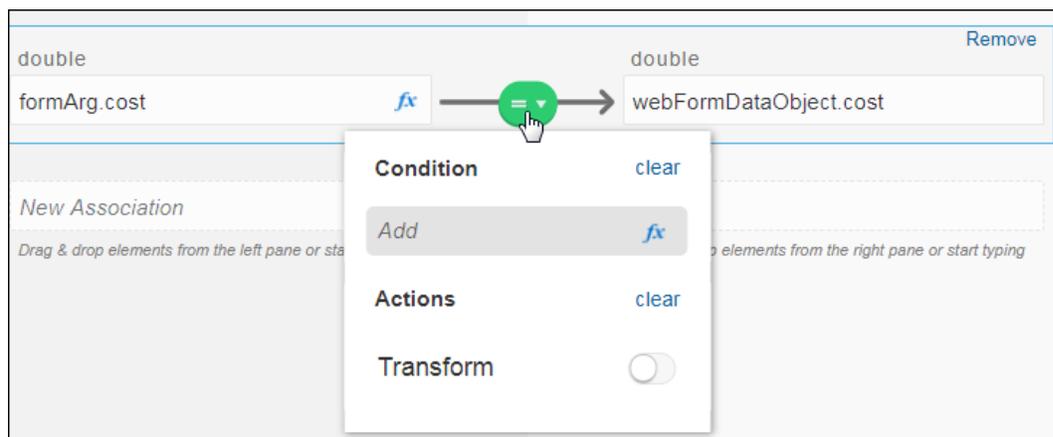
There may be a business scenario where you want a data association to execute in runtime only when a particular condition fulfills. Control the execution of data associations in runtime by adding conditions to them in the Data Association editor.

A data association that has been configured with conditional mapping executes in runtime only when the defined condition fulfills. Otherwise, it fails to execute. All other data associations that do not have conditional mapping defined execute normally in runtime.

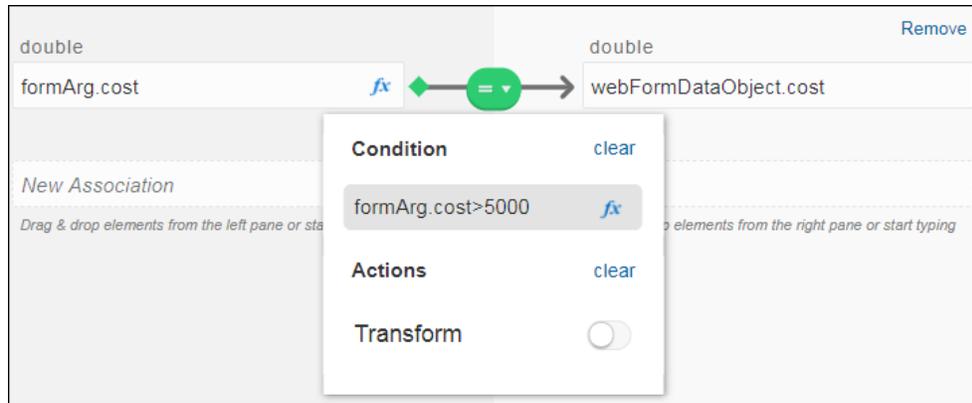
Note that you can set conditions for data mapping in both structured and dynamic processes.

To define a new condition:

1. Click the **Data Association** icon in the data association for which you want to configure a condition. You can define the new condition in the resulting dialog box.



2. Define the new condition by typing directly in the **Add** field. For example, `form.Arg.cost>5000`.
3. Optionally, click the **Expression Editor** icon to use the Expression Editor to define a condition using standard functions and operators.
See [Work with Expressions](#).
4. If the condition is valid a green icon  is displayed next to the **Data Association** icon.



5. If the configured condition is invalid, the icon turns red . To know more about the error, click the **Data Association** icon. An error message with details about the error is displayed below the condition field.
6. Finally, click **Apply** to save all data associations including the data associations with conditions set on them.

In runtime, the data associations for which you have defined conditions execute only if the conditions fulfill. In case of our example, the payload from the **Cost** field in the process Start activity passes on to the next activity of the process, only if its value is more than 5000.

Configure Data Association

Data association refers to the flow of data within a process. Use the Data Association editor to define input and output for flow elements that need them.

A data association involves a source and a target, where the source provides a value or an expression to be assigned to the target. An approval human task, for example, needs both input and output data association.

- On the input side, it needs data input into the activity (referred to as its payload).
- On the output side, after the activity has just finished, it needs output from the activity to data objects, to store results for use elsewhere in the process.

To configure data association:

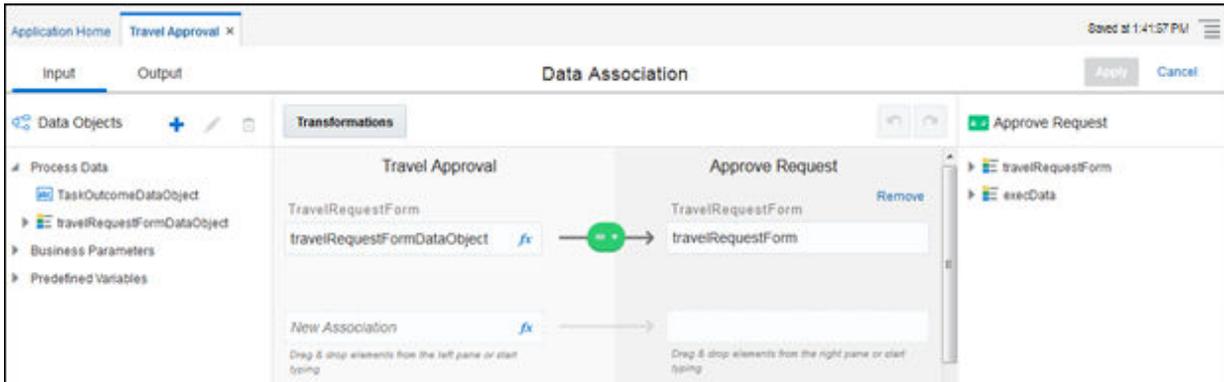
1. In the process editor, select a flow element that needs data association, such as a human task or service task, and click the **Data Association** button in the upper right. (This button is active when the selected flow element needs association. See [Data Association Tips](#).)

The Data Association editor opens. Use its center pane as a canvas for mapping associations, by dragging and dropping elements from the left and right panes. Any data associations already configured display below **Data Association**. You use the empty pair of fields to create a new association.

Note:

Process creates data associations for you in some cases, such as when you create a human task or start event with a form.

- a. Notice that the **Input** tab is selected. The left pane displays source objects (**Data Objects**) in an expandable tree. The right pane (indicated by the task's icon) displays the payload, or entry parameters the activity needs to perform its function.
- b. Click the **Output** tab and notice how the left and right panes change. Now the left pane displays the task's payload, and the right pane displays the variables available in the process.



2. Click the **Input** tab again, and begin creating a new association by dragging an input object from the Data Objects tree and dropping it in the input field titled *New Association*. You can also begin typing and select from autocomplete options that appear. To specify attributes within objects, enter a . (period) and select from the list that displays.

The input source provides the value or expression to assign to the target. You select variables related to the process (such as data objects or business parameters) to map to the specific parameters the activity needs to perform its function (its entry parameters).

3. If needed, click the **fx** (Expression) icon and use the Expression Editor to create an input expression using standard functions and operators. See [Work with Expressions](#).
4. Complete the association by dragging an output object from the payload tree and dropping it in the output field.

Process validates the new association and displays a green association icon if valid, or a red association icon and error if invalid:

 Valid data association

 Invalid data association

See [Data Association Tips](#) for additional information about configuring associations.

5. If the association is invalid, optionally create a transformation to convert its associated attributes.

A transformation maps mismatching data types. See [Work with Transformations](#).

6. If needed, create additional associations and reorder them.

Drag and drop an association to move it up or down in the associations list. Associations are executed in the order in which you position them. For example, if multiple associations assign a value to the same object, the last assigned value is used.

As you edit, click the **Undo** or **Redo** buttons as needed.

7. Click the **Output** tab, and create output data associations. You can think of the output as though the activity has just finished, and likely contains results that you'd like to store for use elsewhere in the process.
8. Click **Apply** to save the data associations.

You can save invalid associations and fix them later. Note that you can't play or deploy applications that contain validation errors.

Data Association Tips

Use the Data Association Editor to add, edit, delete, and reorder data associations.

Below are tips for [configuring data associations](#).

- The following flow elements need data association. Note that some flow elements have only output, such as a start form event, which captures the data end users enter into the form.
 - Human tasks
 - System tasks, except abstract and notification tasks
 - Events, except timer catch, error boundary, and terminate end events
- You can access data association in any of these ways from the process editor:
 - By clicking the **Data Association** button in the upper right.
 - By clicking a flow element on the process, clicking the menu icon, and choosing **Open Data Association**.
 - By opening the Properties pane and clicking the input (right) or output (left) arrow in the lower left of the pane.
- In the Data Association Editor, click the **Input** tab to define data input into the activity, and the **Output** tab to define data output resulting from the activity.
- Under **Data Objects**, you can create, edit, and delete data objects if needed without exiting the data association editor.
- Dropping an object into an association field that already contains a value replaces its value. If you drop an object into an input field that ends with an operator, the object name is appended to the field's contents.

Data Association for Enums

The Data Association Editor provides flexibility in associating enums (enumerations) and primitive types.



Note:

You can also transform enum items. See [Create Transformations Between Enum Items](#).

Follow these guidelines for associating [enum objects](#).

- **You can assign an enum type to a primitive type.**

In this case, you're assigning a source that includes a limited set of values to a target that includes a full set of values. Note that the types must be compatible; for example, you can't assign an integer enum data object to a string data object.



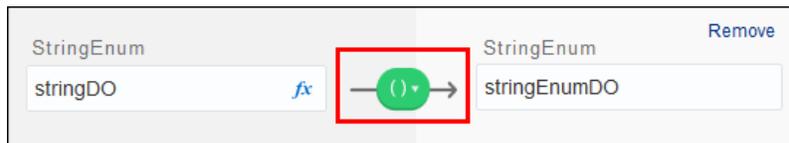
- **Use caution when assigning a primitive type to an enum type.**

In this case, you're assigning a source that includes a full set of values to a target that includes a limited set of values. The data association editor allows the assignment, but displays an **Automatic Casting** icon as a caution that runtime issues may result.



Automatic Casting association icon

For example, an error might be triggered when data from an automatically cast association is passed to an external service. This same behavior occurs when using explicit casting, as described in [Work with Expressions](#).

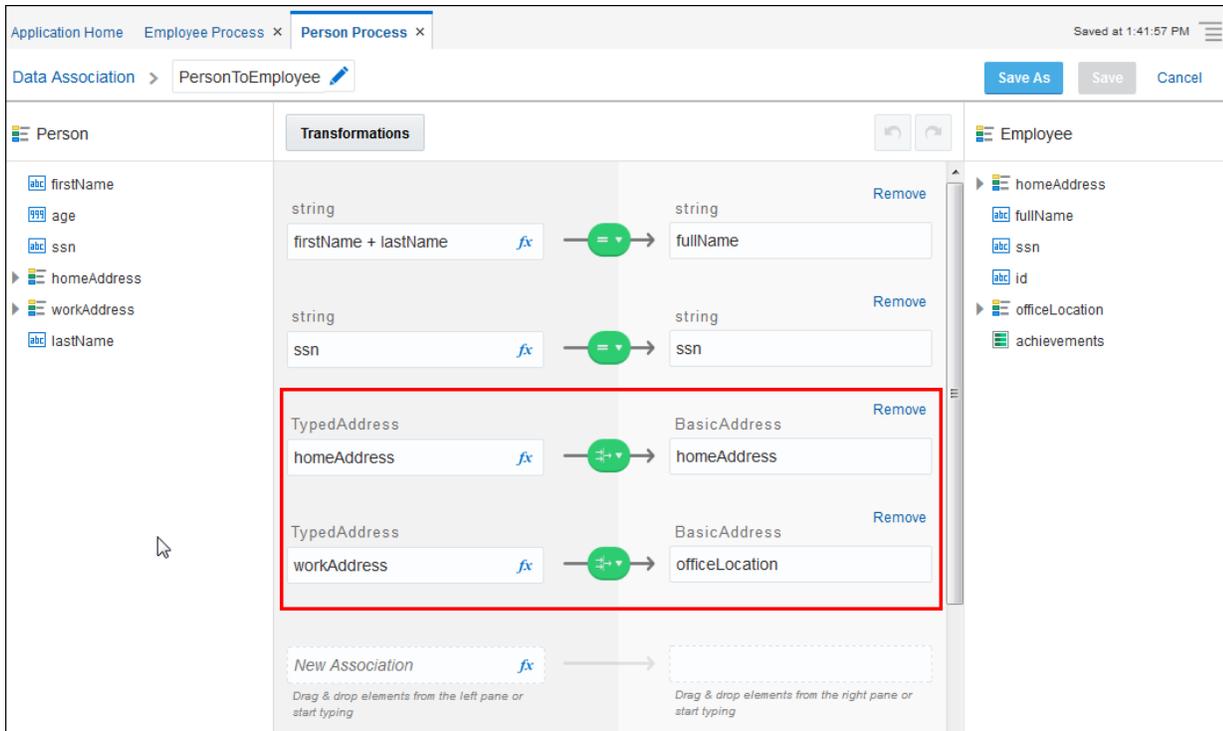


Work with Transformations

A transformation is a special type of data association between input and output data types that don't match. A transformation both associates input and output objects and maps their types. And because transformations are reusable, you can configure them once for use throughout the application, then apply them as needed.

Transformations are particularly useful for matching slightly differing data attributes, such as address or employee data. For example, your process might connect to a service that returns employee data with slightly different attributes that need to be mapped. In the illustration below, the bottom two associations are transformations between mismatched address objects.

You can use transformations with arrays, and they're applied to each item in the array.



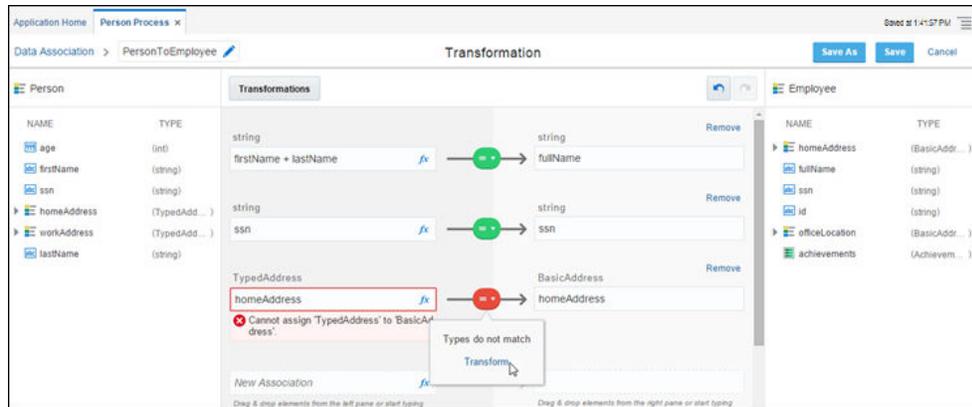
Note that transformations have their own icon, which differs slightly from the data association icon. As with data associations, the icon is green when valid. When invalid, the icon is red and its error displays.

You [create and apply transformations](#) while [configuring data associations](#). You can either transform a selected data association or create a transformation and then apply it. You can even create transformations within transformations.

Create, Apply, and Edit Transformations

You create and apply transformations while creating data associations.

1. In the process editor, follow the steps to [configure a data association](#). Select a flow element that needs data association, click the **Data Association** button, and map objects to the association input and output fields.
2. Click the data association icon, and select **Transform**. (The association can be valid or invalid.)

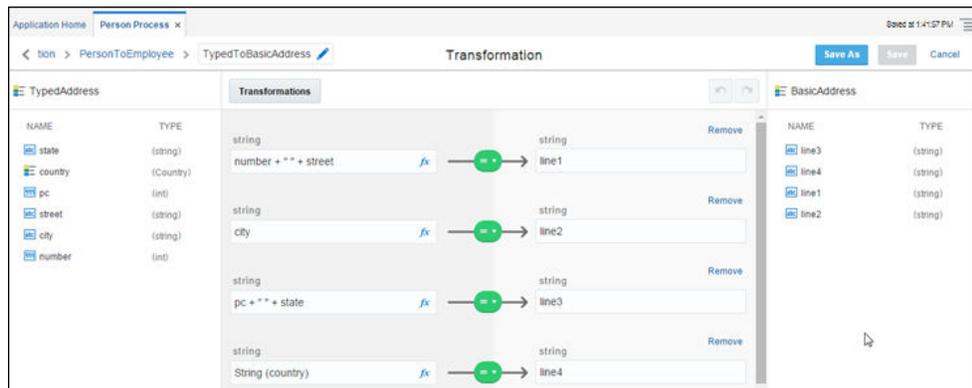


3. In the Transformations window (named based on the objects you selected), click **Create**. In the New Transformation dialog box, enter a name, and click **Create**.

The Transformation Editor displays. It looks and functions much like the Data Association Editor, except that its left and right panes reflect the selected objects to transform.

4. Configure a transformation between the objects by dragging and dropping elements to the input and output fields. You can also begin typing and select from autocomplete entries that display.

If needed, click the **fx** (Expression) icon and use the Expression Editor to create an expression for an input field. See [Work with Expressions](#).



5. Click **Save** to save the transformation. You can use the **Save As** button to create a new transformation based on the selected one.

Process applies and validates the new transformation, displaying a green transformation icon if valid, or a red transformation icon and error if invalid. Note that you can save an invalid transformation and fix it later.

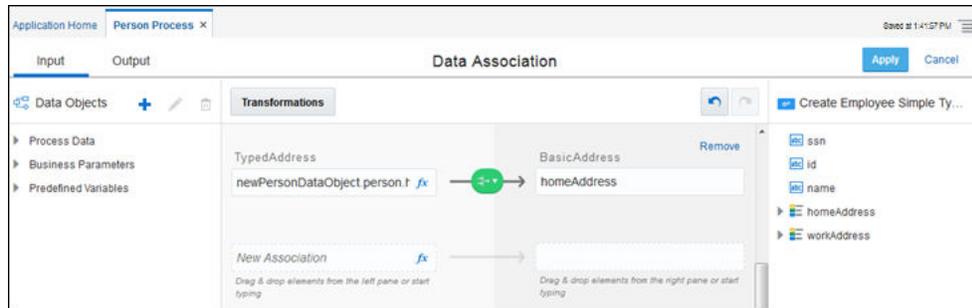


Valid transformation



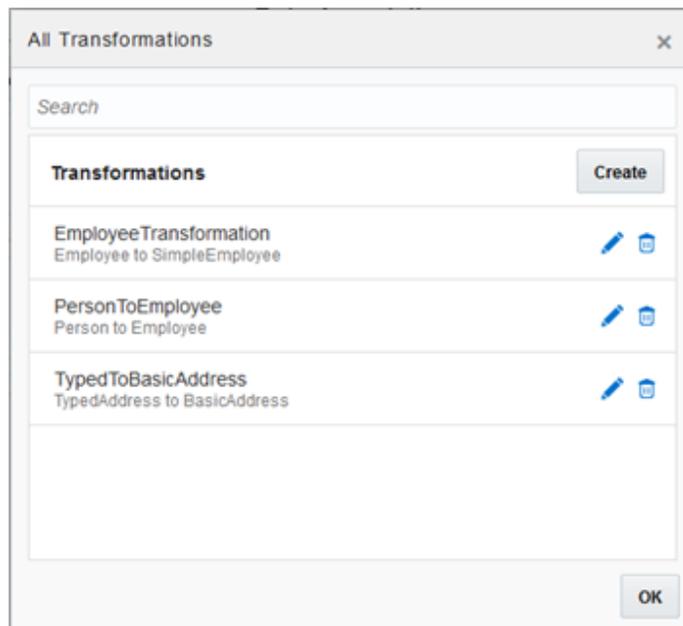
Invalid transformation

6. Click the **Data Association** link to close the transformation editor and return to the data association editor.



7. Create, edit, or change transformations as needed.

At any time, you can display a list of all transformations defined for the application by clicking the **Transformations** button in the Data Association Editor. This displays the All Transformations window, which includes options for searching for, creating, editing, and deleting transformations. If creating a transformation from this window, you must select the input and output objects to transform from the application data objects listed in **From** and **To** fields, then apply the transformation to a data association.



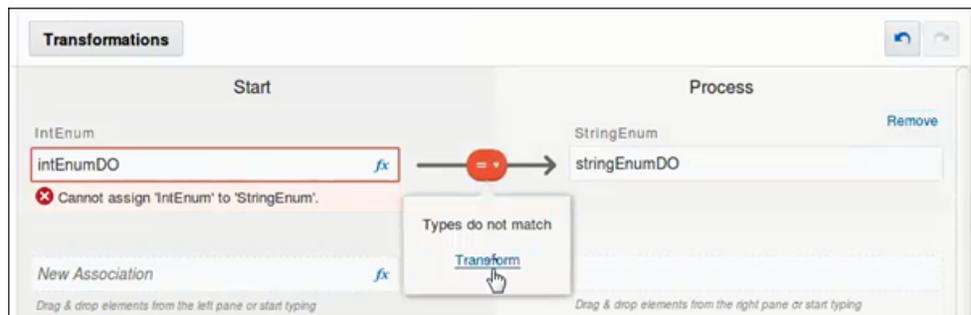
- To apply a transformation to a data association, click its icon in the Data Association Editor and select the **Transform** link. From the list of transformations available for the selected data types, select a transformation to apply and click **OK**.
- To apply a different transformation, click a transformation icon in the Data Association Editor and select the **Change** link. From the list of transformations available for the selected data types, select another transformation and click **OK**.
- To edit a transformation, click its transformation icon in the Data Association Editor, select the **Edit** link, and make edits in the Transformations Editor.
- To delete a transformation, click the **Transformations** button, select it from the Transformations list, and click **Delete**. If the transformation is in use, a warning lists where it's in use and will be removed if you proceed with the deletion.

Create Transformations Between Enum Items

You can create and apply transformations between enum items when creating data associations.

Enum transformations work similarly to [regular transformations](#), with a few differences highlighted below. Note that you can create enum transformations between enums only.

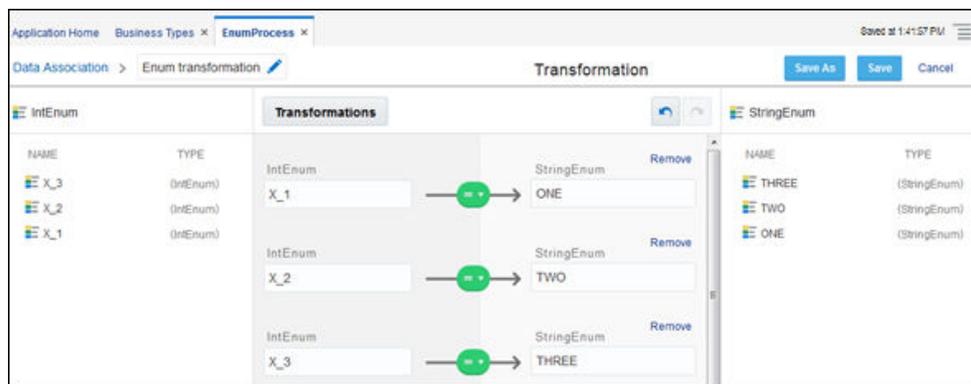
1. In the process editor, follow the steps to [configure a data association](#): select a flow element that needs data association, and click **Data Association**.
2. In the data association editor, drag and drop source and target enums. Click the invalid association icon, and select **Transform**.



3. In the Transformations window (named based on the objects you selected), click **Create**. In the New Transformation dialog box, enter a name, and click **Create**.

The Transformation Editor displays. It looks and functions much like the Transformation Editor does when enum types aren't selected, with several differences:

- Instead of Input and Output tabs on each side of the transformation, you see the enums and their enum items to map individually.
- No expression editor icon is shown, because simple expressions aren't allowed in enum transformations.



4. Configure the transformation by dragging and dropping enum items to map source and target fields. For example, you might convert X_1 from IntEnum to ONE from StringEnum. Note that you can't create transformations between enum items.
5. Click **Save** to save the transformation. You can also click **Save As** to create a new transformation. If needed, click **Apply** to apply the transformation you created to the data association.

Process validates the new transformation, displaying a green transformation icon if valid, or a red transformation icon and error if invalid.

11

Create Decisions

In Process, create decision models to automate the decision logic inherent in your business processes. As part of creating a decision model, add and order decisions, define their input, model their logic using simple or complex types, test them, and expose them as services for use in Process applications.

Topics:

- [How do decisions work in process applications?](#)
- [Ready to create a decision model?](#)
- [Work with Decision Models](#)
- [Create Decision Models](#)
- [Understand Decision Model Views](#)
- [Add and Order Decisions](#)
- [Define Expressions with the Friendly Enough Expression Language \(FEEL\)](#)
- [Define Decision Input and Type](#)
- [Model Decision Logic](#)
- [Test Decisions](#)
- [Expose Decisions as Services](#)
- [Manage Decision Model Snapshots](#)
- [Add Decisions to Applications and Processes](#)
- [Promote Decision Models as Samples to the Gallery](#)
- [Best Practices for Modeling Decision Logic](#)

How do decisions work in process applications?

Use the Decision Model editor as a canvas to define, test, and output decision logic for process applications. The editor supports the Decision Modeling and Notation (DMN) standard, version 1.1, and uses FEEL (Friendly Enough Expression Language) to make decision modeling easier and more intuitive.

Automate and reuse decisions throughout your business processes

In business, decisions are everywhere. Should this loan application or document change be approved? Should emergency vehicles be dispatched to this incident? How many bonus shopping points is this shopping cart worth?

Use the decision model framework to express a full range of automated decisions. Model your decisions as a tree of simple decisions, each automated using decision tables and simple expressions instead of production rules. Enter expressions in a simple standard expression language without worrying about quotation marks or special formatting. Then

activate and use your decision models in one or more applications, and easily modify them as needed.

If you're new to the editor, [start here to learn the basics](#) by creating and configuring a decision model and then test it before activating it.

Want to learn more about creating decision tables or complex logic structures such as functions, relations, or contexts? See [Model Decision Logic](#). Wondering how decision models are deployed and used in applications? See [Work with Decision Models](#).

Ready to create a decision model?

Ever leave the house and wish you'd better prepared for the weather?

Let's create a process application to fix that. When prompted, you input the temperature and rain forecast, and the application decides whether you should bring:

- An umbrella, for warm and rainy weather
- A raincoat, for cold and rainy weather
- An overcoat, for cold and dry weather
- Nothing, for warm and dry weather

Using this example, let's explore the entire decision model development life cycle—from creating a decision model, defining its decision logic, activating it on the DMN server, and using a decision service in a process.

- [Create a Simple Application](#)
- [Create a Decision Model](#)
- [Add Decisions](#)
- [Define Input](#)
- [Model Decision Logic](#)
- [Test Your Decisions](#)
- [Create a Service](#)
- [Activate a Decision Snapshot](#)
- [Use the Decision in Your Application](#)
- [Map Data to and from the Decision](#)
- [Try Your Decision in Runtime](#)

Create a Simple Application

First, let's create an application for the decision. Its process starts with a web form, which has two input fields where you enter temperature and rain forecast numbers. It then uses a decision element to call the decision service. Lastly, a human task element displays the decision result, which tells you what to bring for the day based on the weather.

1. Create an application.

Click **Processes** and then click **Process Applications** to go to the Process Applications page, and then click **Create**.

Click **Create an Application**. Enter a name (What To Bring) and click **Create**.

2. Create a process.

On the **Processes** tab within the application, click **Create** and then **New Process**. In the Create Process pane, select **Form**, enter a process name (Decision Process), and click **Create**.

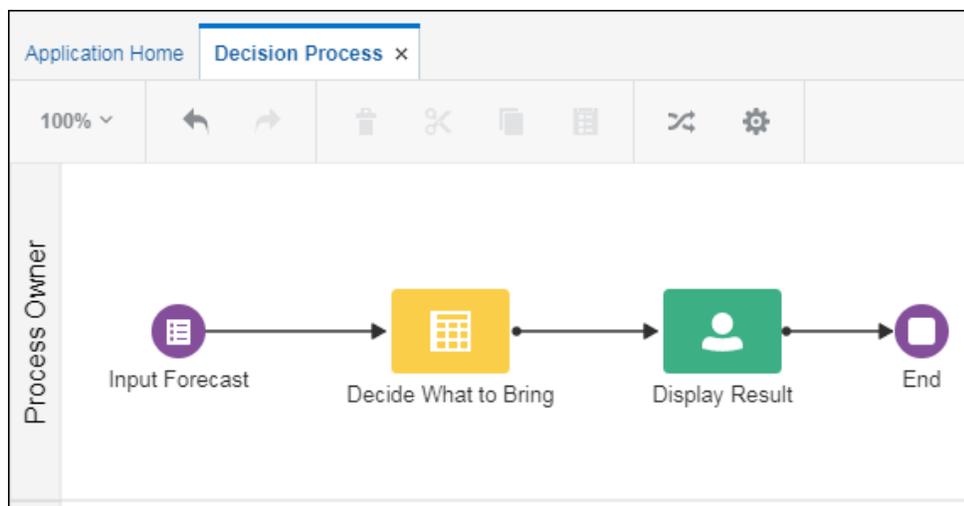
3. Add a decision and human task.

On the Decision Process tab, expand **System** on the Elements palette, drag a **Decision** element to the process, and drop it after the start event.

Expand **Human** on the palette, drag a **Submit** element to the process, and drop it after the decision element.

4. Rename the flow elements.

Double-click the flow elements and rename the start event to `Input Forecast`, the decision to `Decide What to Bring`, and the human task to `Display Result`.



5. Add a web form.

Select the **Input Forecast** element, click **Menu** , and select **Open Properties**. In the **Title** field, enter a name (`Forecast`) to display for the application.

Click **Create New Form**  next to the **Form** field. Enter `InputForecastWebForm` in the **Name** field, select the **Open Immediately** check box, and click **Create**.

6. Add the forecast input fields to the web form.

From the Basic palette, drag and drop two **Number** controls and one **Input Text** control onto the canvas. Select each control and edit its **Label** field in the Properties pane, changing the number controls to `Temperature` and `Rain`, and the text control to `What To Bring`. Further, edit the **Name** field of the controls as follows: `temperature`, `rain`, and `whatToBring`.

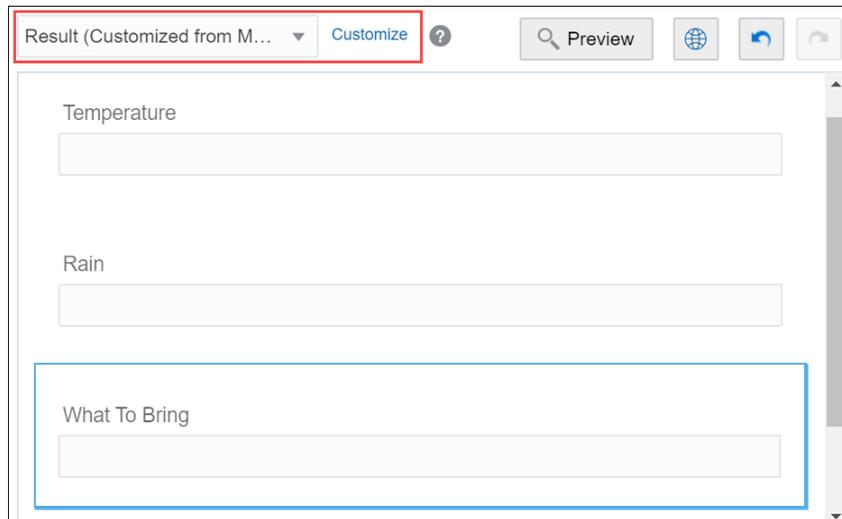
Select the **What To Bring** control, scroll down the General tab, and select the **Hide** check box. Notice that the control becomes dimmed to indicate it's hidden.

7. Add a presentation to display results.

Click the form outside a control. On the Form tab, click **Add Presentation**  next to **Presentations**. In the Select Presentation Type dialog box, select **Customize**. In the

Create Presentation dialog box, select **Main** in the **Select from Previous Presentation** field, enter a name for the new presentation (*Result*), and click **Create**. On the new Result presentation, click **Customize** next to the presentation's name and, in the resulting window, click **hidden** in the **What To Bring** control's row to show the control on this presentation. Click **Close**.

Want to learn more about web forms? See [Create Web Forms](#).



8. Save the application.

Want to learn more about working with applications? See [Create and Manage Applications](#).

9. Return to the process by clicking the **Decision Process** tab. Open the properties for the **Display Result** human task. Click **Browse** next to the **Form** field and select **InputForecastWebForm**. In the **Presentation** field, select **Result**.

Create a Decision Model

A decision model does what its name implies: it lets you define how decisions will work. It contains a collection of decisions, input data, and other definitions.

To create a decision model:

1. Return to the Process Applications page by clicking **Back** on the navigation pane.
2. On the navigation pane, click **Decision Models**.
You can use a decision model in multiple applications.
3. Click **Create**.
4. Click **Create a Decision Model**.
5. In the Create Decision Model dialog box, enter a name for the model (*What To Bring Decision Model*) and click **Create**.

The decision model editor opens in the graph view, with the title you entered displayed at the top. Notice that the editor contains a central canvas area on which you construct decisions, with the diagram palette and expandable Services pane on either side.

See [Work with Decision Models](#).

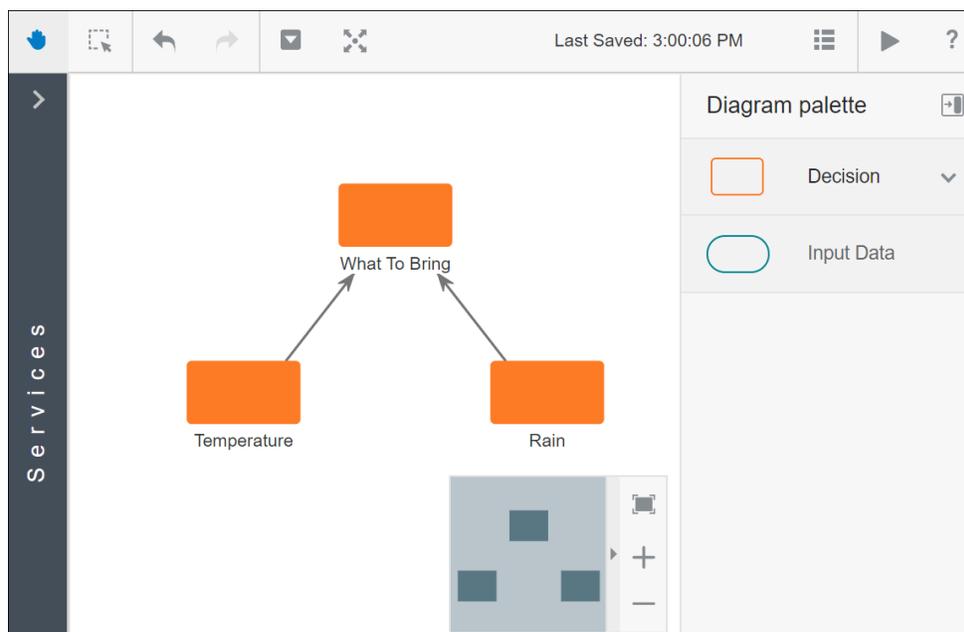
Add Decisions

Let's set up our decision model framework. In Decision Model and Notation (DMN), a decision model answers a central question. This model answers the question: what to bring to be prepared for the weather?

1. Add a result decision.
 - a. On the diagram palette, click **Expand** ▼ next to **Decision**. Select and drag the **Empty Decision** element onto the canvas.
 - b. Edit the decision's name (`What To Bring`). You can change its logic type later.
2. Create two more decisions to act as supporting decisions.

Most decisions models include supporting decisions that feed into the result decision. Deciding what to bring requires supporting decisions to answer these questions: 1) Will it be warm or cold? 2) Will it rain?

- a. From the diagram palette, drag two more empty decisions onto the canvas and add them below the result decision. Edit the names of these decisions (`Temperature` and `Rain`).
- b. Create connections from the supporting decisions to the result decision. Select a supporting decision and click **Create New Connection** ↗ and keep the mouse depressed. Move the cursor to the result decision, and then release the click.



Notice that in the example diagram the topmost decision is the **main result** decision. **Supporting** decisions are added below the main decision. However, you can order decisions in several ways. For example, you can create a top-down sequence, a left to right flow, and so on.

Define Input

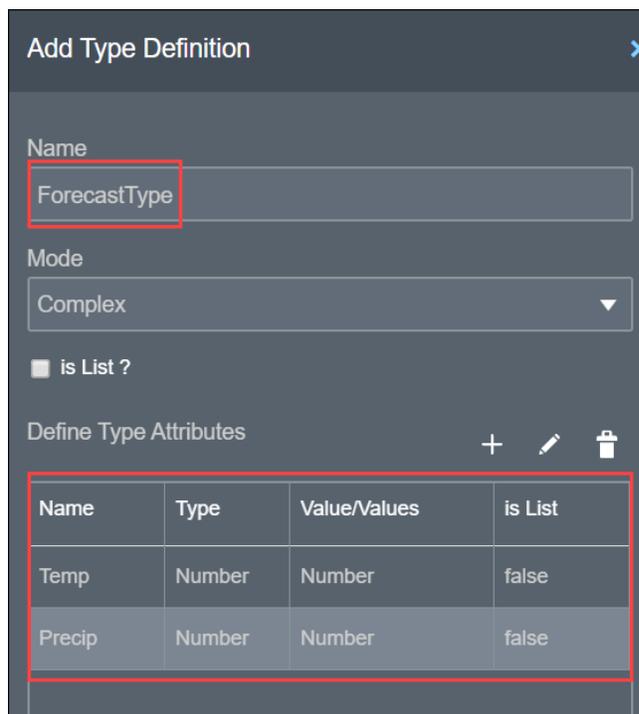
It's hard to make a good decision without input (or data). For our “What to Bring for the Weather” example, the input will be two numbers: temperature and rain percent forecasts. Let's set up their definitions.

1. On the diagram palette, select the **Input Data** element, drag it onto the canvas, and drop it below the supporting decisions. Edit the element's name (`Forecast`).
2. Add a complex type definition for the forecasts.

- a. Select the input element and click **Open Properties** .
- b. In the properties pane, Select the **Mode** as **Other Type**, and then click the **Define New Type** link that appears in the pane.

A new type definition is created with a default name and the Add Type Definition panel opens for editing.

- c. In the type definition panel:
 - i. Edit the definition's name (`ForecastType`) and select the **Mode** as **Complex**.
 - ii. Click **Add Component** next to **Define Type Attributes** to add two attributes.
 - iii. For the first attribute, enter `Temp` for its name and select **Number** as its mode. Click **Close**  to return to the Add Type Definition panel.
 - iv. For the second attribute, enter `Precip` for its name and **Number** as its mode.
 - v. Click **Close** .



| Name | Type | Value/Values | is List |
|--------|--------|--------------|---------|
| Temp | Number | Number | false |
| Precip | Number | Number | false |

3. In the Input Data Properties pane, choose the type definition you just defined in the **Other Types** field to associate it with your `Forecast` data variable. You can edit or delete the type definition using the buttons next to the field.

4. Click **Close**  to collapse the Input Data Properties pane.
5. Now, create connections from the input data element to the supporting decisions.
 - a. Select the element and click **Create New Connection**  and keep the mouse depressed.
 - b. Move the cursor to the **Temperature** element, and then release the click.
 - c. Repeat the steps to create a connection to the **Rain** element.

Want to learn more about input? See [Define Decision Input and Type](#).

Model Decision Logic

Now let's change the empty logic of the decisions. You can configure decision logic in a variety of ways, but here you'll create a Boolean expression, an if-then-else expression, and a decision table.

1. Configure a Boolean expression for the rain forecast.
 - a. Select the **Rain** decision and click **Open Properties**  to open the Decision Properties pane.
 - b. In the **Logic** field, change the decision type to **Expression**.

Notice that the decision's icon changes to an expression icon. Also, a small red circle indicates a validation error because you haven't entered an expression yet.

See [Add and Order Decisions](#) for information on other fields in the properties pane.

- c. Enter the expression using the input data you already defined. Double-click the **Rain** element or click **Edit**  next to the **Name** field in the properties pane and enter this condition:

Forecast.Precip>80



When the precipitation input value is greater than 80 percent, this Boolean expression will be true (rain is likely); otherwise it will be false (rain is unlikely).

Want to learn more about the expression language? See [Define Expressions with the Friendly Enough Expression Language \(FEEL\)](#).

- d. Click **Close**  to return to the canvas. Notice that the validation error no longer displays because the expression is now valid.
2. Configure an if-then-else decision for the temperature forecast.
 - a. Select the **Temperature** decision and click **Open Properties**  to open the Decision Properties pane.
 - b. In the **Logic** field, change the decision type to **If then Else**. The decision element shows a validation error for items you need to complete.
 - c. Double-click the **Temperature** element or click **Edit**  next to the **Name** field in the properties pane and complete the decision's **if**, **then**, and **else** expression fields as follows:
 - **if:** Forecast.Temp>65
 - **then:** "warm"
 - **else:** "cold"

Because "warm" and "cold" values are strings, you must include quotation marks around the values. (Depending on your location, you may want to adjust the temperature condition of **>65** from degrees in Fahrenheit to **>18** in Celsius.)

Temperature
✕

| Operation | Expression |
|-----------|------------------|
| if | Forecast.Temp>65 |
| then | "warm" |
| else | "cold" |

- d. Click **Close** ✕ to return to the canvas and verify that the validation error no longer displays.
3. Configure a decision table for the **What To Bring** decision.

A decision table lets you model input, output, and rules in a compact way. It derives a final output decision using input data and the results of other decisions.

In the example, the temperature and rain input decisions each have two possibilities, resulting in a total of four output possibilities.

| Temperature (input) | Rain (input) | What To Bring (output) |
|---------------------|--------------|------------------------|
| <i>warm</i> | true | umbrella |
| <i>cold</i> | true | raincoat |
| <i>cold</i> | false | overcoat |
| <i>warm</i> | false | nothing |

- a. Select the **What To Bring** decision and click **Open Properties** to open the Decision Properties pane.
- b. In the **Logic** field, change the decision type to **Decision Table**.
- c. Double-click the decision element and explore the table started for you.
 - The yellow column is for **output**, and is always the final column of the table. **What To Bring** is already designated as your output decision.
 - The adjacent gray column is for **input**. You'll add a second input column, using one of the controls located above the output column.
 - Rows are for **rules**. A row labeled **1** is already listed.
 - Above the **1** in rule **1** is a **U**, indicating that the table uses a unique hit policy. The hit policy determines the output of a decision table when more than one rule matches the input data.

See [Configure a Hit Policy](#).

- d. Configure the possible values for the **What To Bring** output column.
 - In the **What To Bring** column, click **Enter Allowed Values**.
 - In the resulting dialog, select **Text** in the **Mode** field, select **list of values** in the **Allowed Values** field, and enter the following values: umbrella, raincoat,overcoat,nothing.
 - Click **Close**.

Decision Table Output

Name
What to Bring

Mode
Text

Allowed Values
list of values

Values
umbrella
raincoat
overcoat
nothing

- e. Configure the **Temperature** input column.

In the **Enter Expression** cell of the gray input column, press **Ctrl+Space** to open a pop-up menu. Select **Temperature** from the menu. For an input column, the Allowed Value cell is auto-populated based on the input expression.

To specify a different set of allowed values, see [Defining Decision Table Input](#).
- f. Add and configure the **Rain** input column.
 - Click the **Temperature** input column and select **Add column after** .
 - In the **Enter Expression** cell, press **Ctrl+Space** to open a pop-up menu. Select **Rain** from the menu. The Allowed Value cell is auto-populated with Boolean values.
- g. Add and configure rules.
 - In row 1, click the cell below **Temperature** and select **warm**.

Notice the  icon in the cell, which stands for text. Clicking this icon, you can toggle between Text , Any , and Advanced  modes for all the cells in this column.

See [Configure Rules](#).

In the text mode, text strings are automatically italicized and quotation marks aren't needed.

- Click the cell below **Rain** and select **true**. Notice the  icon in the cell, which stands for Boolean. Clicking this icon, you can toggle between Boolean , Any , and Advanced  modes for all the cells in this column.
 - Click the cell below **What To Bring** and select **umbrella** from the auto-suggest menu. If needed, set the cell's mode to Text. Quotation marks aren't needed.
- h.** Click **Add rule after**  three times to add three rows to the table.
- i.** Complete rules 2 through 4 of the decision table:
- For rule 2, select **cold**, **true**, and **raincoat**.
 - For rule 3, select **cold**, **false**, and **overcoat**.
 - For rule 4, select **warm**, **false**, and **nothing**.

| What To Bring | | | |
|----------------|---------------------------------|---------------------------|--|
| Decision Table | | | |
| U | Temperature <i>cold,warm</i> | Rain <i>true,false</i> | What To Bring <i>umbrella,raincoat,overcoat,nothing</i> |
| 1 | warm | true | umbrella |
| 2 | cold | true | raincoat |
| 3 | cold | false | overcoat |
| 4 | warm | false | nothing |

- j.** Close the decision.

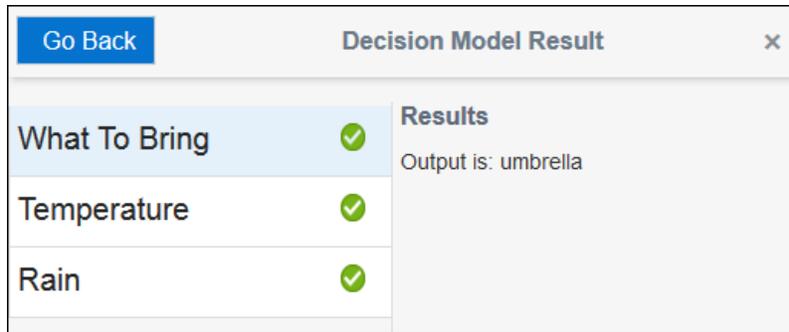
Want to learn more about decision model types? See [Model Decision Logic](#).

Test Your Decisions

Verify that the model produces the results you want by testing it with input values.

1. Click **Test** .
2. In the Test Decision Model pane, enter values in the **Temperature** and **Rain** fields and click **Start Test**. For example, enter 85 into both fields.

The Decision Model Result pane displays decision results. Click each green check mark to see the decision's result.



3. Click **Go Back**, and repeat step 1 and 2 to test each decision rule's outcome.

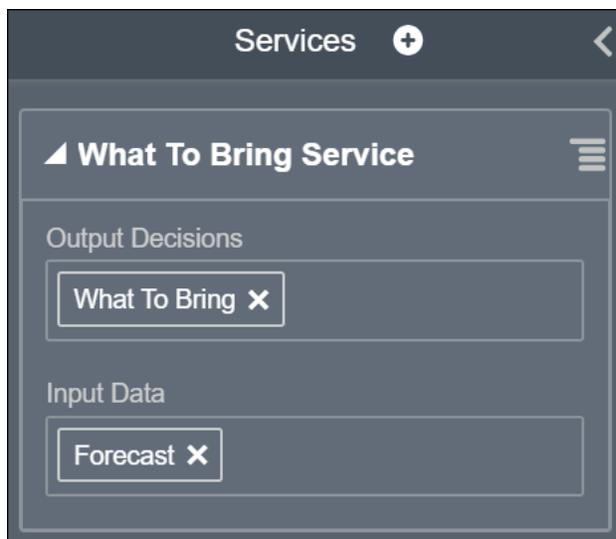
| Temperature (F) | Precipitation % | Outcome |
|-----------------|-----------------|---|
| 85 | 85 | <ul style="list-style-type: none"> • What to Bring: umbrella • Temperature: warm • Rain: true |
| 50 | 85 | <ul style="list-style-type: none"> • What to Bring: raincoat • Temperature: cold • Rain: true |
| 50 | 50 | <ul style="list-style-type: none"> • What to Bring: overcoat • Temperature: cold • Rain: false |
| 85 | 50 | <ul style="list-style-type: none"> • What to Bring: nothing • Temperature: warm • Rain: false |

4. Save the decision model.
Want to learn more? See [Test Decisions](#).

Create a Service

After you complete and test the decision logic, create a RESTful service to communicate with Process. The decision service specifies the decision and input data you want to expose.

1. Expand the Services pane, and click **Add new service** .
2. Enter a name for the decision service (What To Bring Service) and click **OK**.
3. Click the **Output Decisions** field and select **What To Bring** from the list. Validation errors display because you haven't specified input data yet.
4. Similarly, click the **Input Data** field and select **Forecast**.



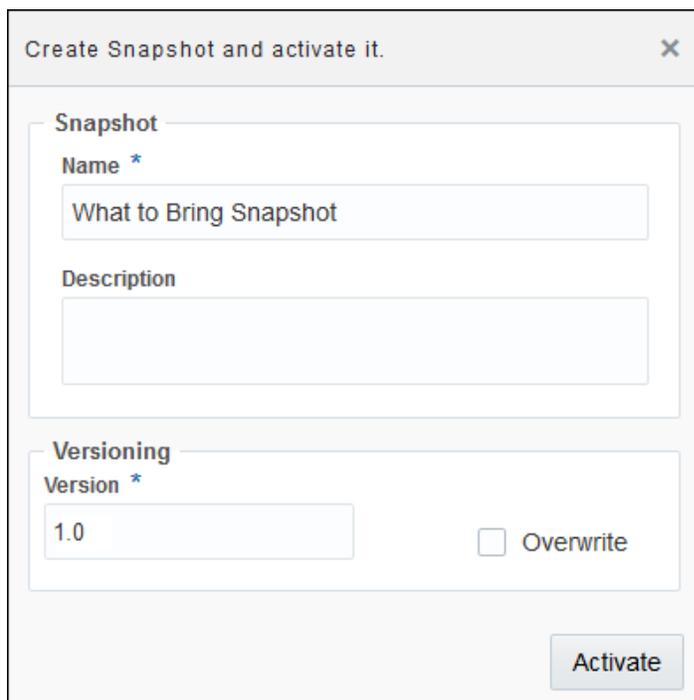
You can output multiple decisions and input multiple data items in a service. See [Expose Decisions as Services](#).

5. Click **Save** on the toolbar.

Activate a Decision Snapshot

To use a decision model, you must create a snapshot of it and activate the snapshot on the DMN server.

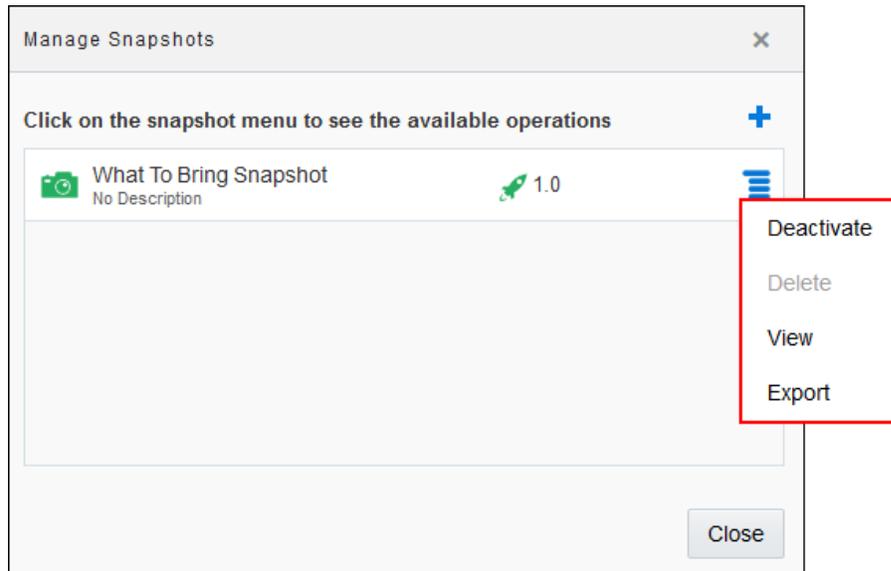
1. In the decision model editor toolbar, click **Activate**.
2. In the Create Snapshot and activate it dialog box, enter a name (What to Bring Snapshot) and click **Activate**.



3. In the decision model editor toolbar, click **Snapshots**.

Notice that the snapshot you just activated is listed, along with any other activated snapshots. Here you can:

- Activate or deactivate a selected snapshot.
- Delete a snapshot. You must deactivate a snapshot before you can delete it.
- View a read-only version of the snapshot.
- Export a snapshot, which you can later import into a new decision model and edit its contents.



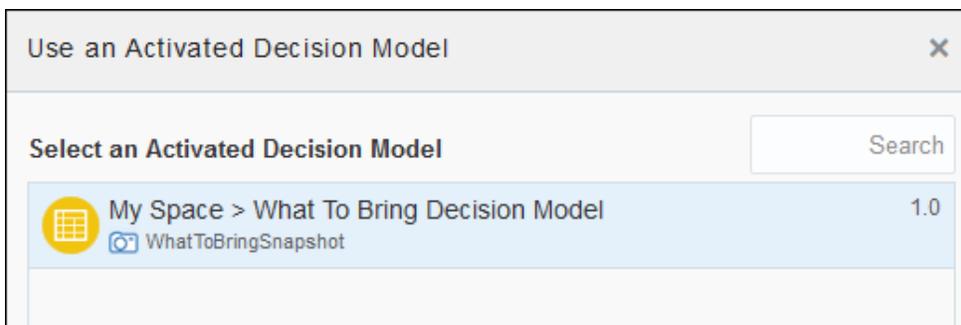
Want to learn more about managing decision model snapshots and their versions? See [Manage Decision Model Snapshots](#).

4. Click **Close**.

Use the Decision in Your Application

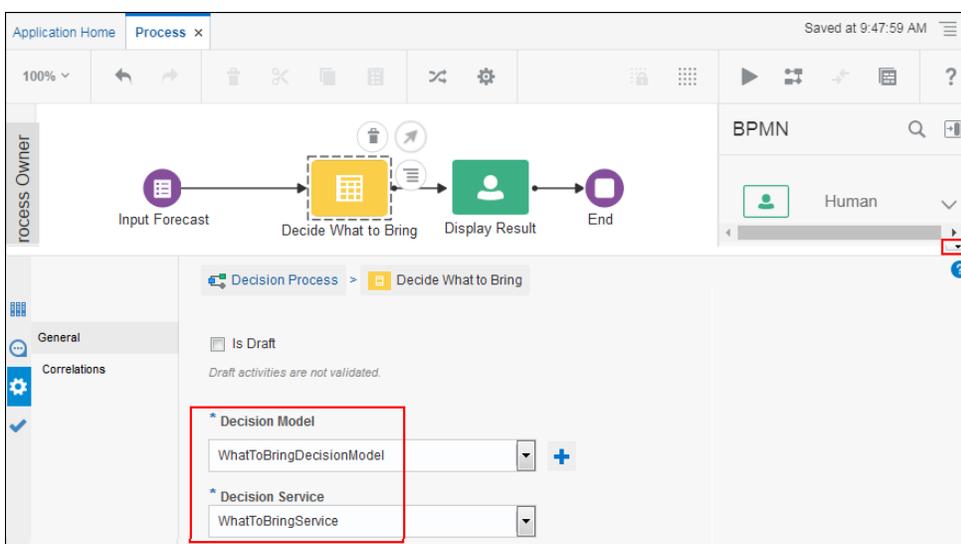
Now let's use your activated decision snapshot in a process within an application.

1. Return to the Decision Models page by clicking **Back** .
2. Click **Process Applications** in the navigation pane and open the **What To Bring** application you created.
See [Create a Simple Application](#).
3. Open the decision process. Select the decision element you added (Decide What to Bring), click **Menu** , and select **Open Properties**.
4. On the Properties pane, click **Use Decision Model**  next to the **Decision Model** field.
5. In the Use an Activated Decision Model dialog box, select the snapshot you created and activated (What To Bring Snapshot), and click **Use**.



Process brings the activated decision model snapshot into the application and performs behind-the-scenes operations such as importing data types and defining the REST connector to the service. See [Add Decisions to Applications and Processes](#).

Notice that the Properties pane now lists the decision model and service you selected to use.

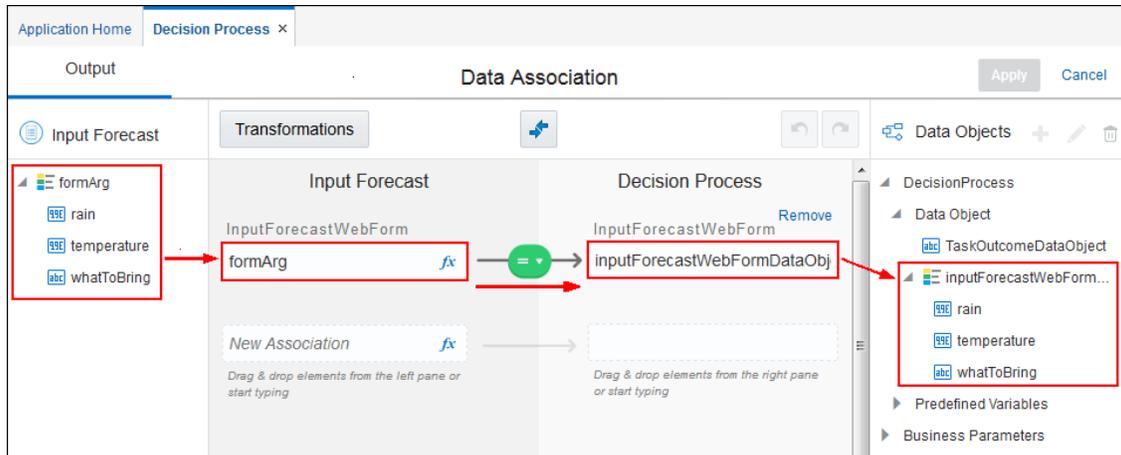


6. Click **Collapse Pane**  to close the properties pane.

Map Data to and from the Decision

Configuring the data mapping is the last key piece in configuring a decision model. The data needs to flow from the web form into the decision and its result must flow out of the decision so the human task can display it.

1. On the Decision Process tab, let's view the data association that was automatically configured for the Input Forecast start activity and the Display Result human task.
 - a. Select the **Input Forecast** element and click **Data Association**  in the process toolbar. Expand the values in both side panes. Notice that values entered into the web form are written to a data object called **inputForecastWebFormDataObject**. Click **Cancel**.



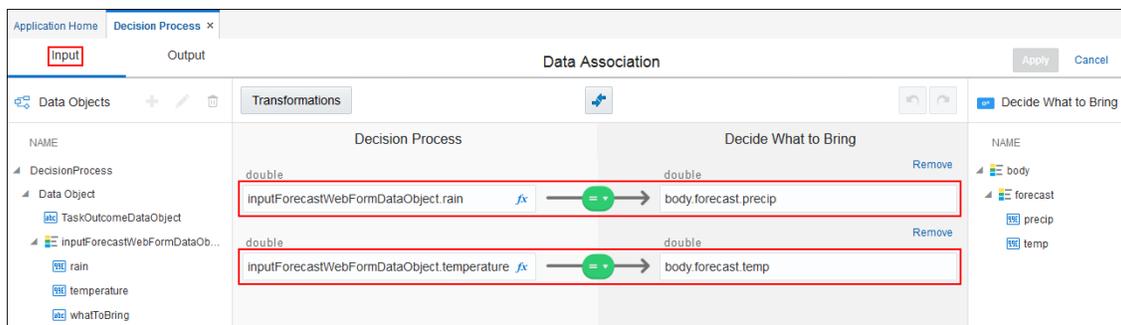
- b. Select the **Display Result** element and click **Data Association**  in the toolbar. Notice that for input, the same data object—**inputForecastWebFormDataObject**—is used to write values back to the web form. Click **Cancel**.
 2. Map input and output data association for the decision element.

The decision needs to use the input values (rain, temperature) contained in **inputForecastWebFormDataObject** and output the decision's result (what to bring) into the data object.

- a. Open data association for the **Decide What to Bring** element. With the **Input** tab selected, expand the values in both side panes.
 - b. Map the rain and temperature values as follows:

`inputForecastWebFormDataObject.rain to body.forecast.precip`

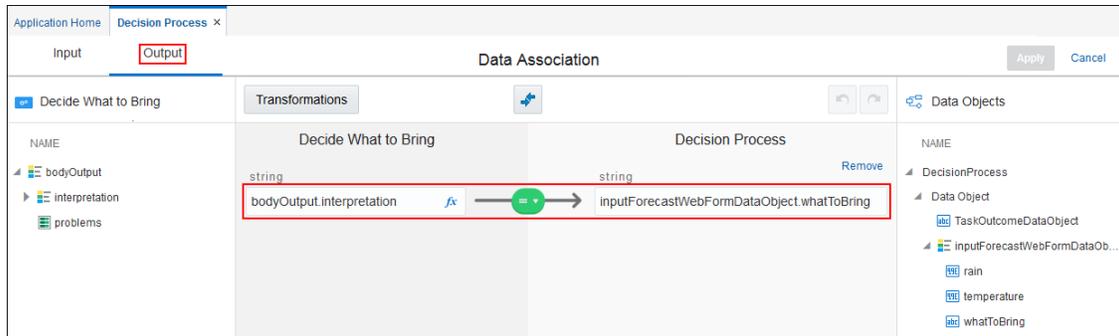
`inputForecastWebFormDataObject.temperature to body.forecast.temp`



- c. With the **Output** tab selected, map the Decide What to Bring value as follows:

`bodyOutput.interpretation to`

`inputForecastWebFormDataObject.whatToBring`

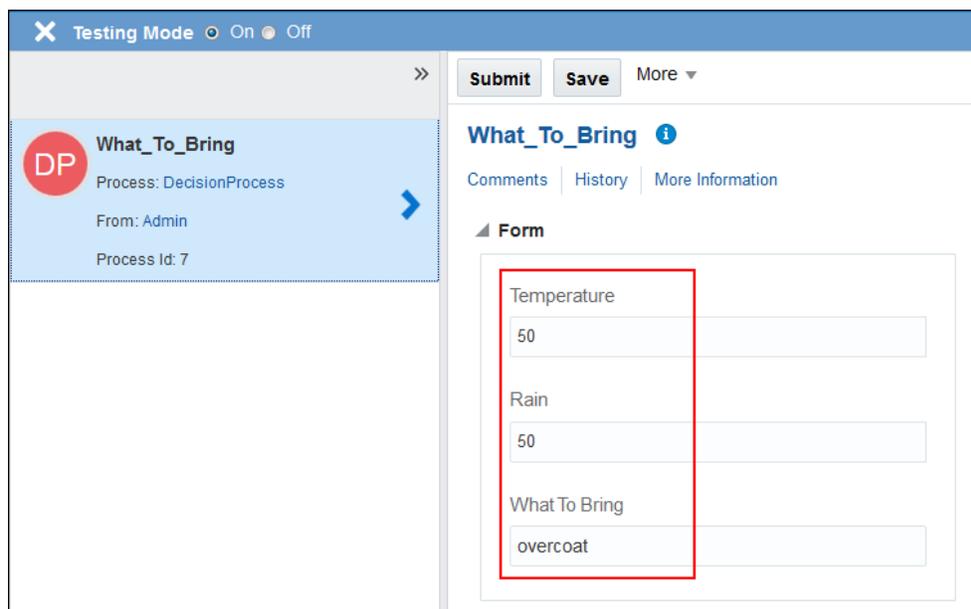


3. Click **Apply**.
See [Configure Data Association](#)

Try Your Decision in Runtime

Let's try running the application in test mode as an end user.

1. From the toolbar, click **Test**.
2. On the Test Application tab, click **Activate**. In the Activate to Test dialog box, leave the **Add Me to All Roles** check box selected and click **Activate**.
3. After the application activates successfully, click **Try in Test Mode**. When the new browser tab opens displaying runtime options, click **Forecast** from the My Apps pane.
4. In the form, enter temperature and rain values, and click **Submit**.
5. Click **My Tasks** and open your new task. The **What To bring** field displays the decision's result based on the values you entered.



Work with Decision Models

When creating a business process for your application, very often, you'll need to create decisions that enable you to automate policies, computations, and reasoning. Instead of

creating multiple decisions, you can create a decision model consisting of decisions and sub-decisions, and associated decision services to allow you to use the decision model in your business processes. Process allows you to create decision models that make business processes less complex, easier to manage, and more robust in the face of change.

Decision models consist of :

- Decisions and sub-decisions along with the implementation logic
- Input data and type
- Associated decision services

Creating a decision model involves the following main tasks, as described in detail in subsequent sections:

- Create a decision model, the container for decisions, sub-decisions, input data, and decision services. See [Create Decision Models](#) and [Understand Decision Model Views](#).
- Add decisions and sub-decisions. See [Add and Order Decisions](#).
- Define the input data and type for your decisions. See [Define Decision Input and Type](#).
- Model the decision logic. Use Friendly Enough Expression Language (FEEL) to define expressions within all notations of decision logic, including decision tables. FEEL is defined by DMN to provide standard executable semantics to all expressions used within a decision model. See [Define Expressions with the Friendly Enough Expression Language \(FEEL\)](#) and [Model Decision Logic](#).
- Test the decisions and sub-decisions within your decision model to ensure that they work as expected. See [Test Decisions](#).
- Create snapshots of your decision model, and activate these on the DMN server. Decision snapshots are read-only copies of a decision model at a particular moment. These snapshots, once activated, can be used independently in multiple applications. You can create, delete or activate snapshots of your decision model or view read-only copies of a snapshot that has already been activated on the DMN server. See [Manage Decision Model Snapshots](#).
- Create decision services to use the activated decision model snapshots in your process applications. See [Add Decisions to Applications and Processes](#).

Decision models facilitate the modeling of complex decisions as a hierarchy of simple decisions. A decision model as a whole, or its decisions and sub-decisions can be used in multiple processes and applications.

You can share decision models with other users directly through the file system. See [Import and Export Decision Models](#).

Create Decision Models

Create decision models to make your business processes less complex, easier to manage, and more robust in the face of change.

You can start with a sample decision model, import a model from your local file system, or create a new one from scratch.

Topics:

- [Create a Decision Model from a Sample](#)
- [Import and Export Decision Models](#)
- [Create a Decision Model from Scratch](#)

Create a Decision Model from a Sample

Use a sample decision model from the gallery and create a copy of it.

You can quickly create a decision model from an available sample to learn more about decision modeling or use the sample model as a starting point for your business use case and modify it to suit your needs.

To create a decision model from a sample:

1. On the Oracle Integration navigation pane, click **Processes**, and then click **Decision Models**.
2. Click **Create**, select **Start with a Sample**, and click **Browse**.

The decision model gallery page appears with a set of Oracle-provided samples.

3. Select a sample from the gallery, click **More** to view a pictorial representation of the model and additional details, and then click **Create**.
4. Enter information into the Create Decision Model dialog. Here are a few things to note:

| Field | Description |
|----------------------|---|
| Name and Description | Make sure the decision model name and description are useful. Give the user a good idea of what the decision model is all about and why they might want to use it. Good names and descriptions help users distinguish decision models with a similar title or purpose. Also, you can't change the name after the decision model is created. |
| Space | Select the space where the new decision model will be stored. To create a new space, select New Space from the drop-down list. |
| Open Immediately | Most of the time you'll want to leave the Open Immediately check box selected so you can start configuring and testing your decision model right away. Deselecting the check box is good if you want to create placeholders for several decision models at once and modify them at a later time. |

5. Click **Create**.

The copy of the sample is created and the decision model editor opens for you to edit or explore the model.

To edit or delete existing decision models, see [Edit or Delete a Decision Model](#).

See [Promote Decision Models as Samples to the Gallery](#) to make more models available as samples in the gallery.

Import and Export Decision Models

You can import and export decision models as DMN files. With this feature, you can share decision models with other users directly through the file system.

Topics:

- [Import a Decision Model](#)
- [Export a Decision Model](#)

Import a Decision Model

You can import a decision model that was previously exported and saved as a DMN file.

To import a decision model:

1. Go to the Process Applications page, click **Create**, select **Import**, and then select **Decision Model**.
2. Click **Browse**, then select the decision model file (.DMN) you want to import.
3. Enter a name for this decision model.
4. Select the space where you want to store your imported decision model.
5. Click **Import**.

Export a Decision Model

Exporting a decision model to your local file system enables you to share decision models.

To export a decision model to a DMN file:

1. On the Process Applications page, find the decision model you want to export.
2. Click **Options**, then select **Download**.
3. Select **Save File**, choose a location on your local file system, and click **Save**.

Create a Decision Model from Scratch

Create a new decision model to use in your process applications.

1. Go to the Home page, click **Processes**, and then click **Decision Models**.
2. Click **Create**, and then click **Create a Decision Model**.
3. Enter information into the Create Decision Model dialog box. Here are a few things to note:

| Field | Description |
|----------------------|---|
| Name and Description | Make sure the decision model name and description are useful. Give the user a good idea of what the decision model is all about and why they might want to use it. Good names and descriptions help users distinguish decision models with a similar title or purpose. Also, you can't change the name after the decision model is created. |
| Space | Select the space where the new decision model will be stored. To create a new space, select New Space from the drop-down list. |
| Open Immediately | Most of the time you'll want to leave the Open Immediately check box selected so you can start configuring and testing your decision model right away. Deselecting the check box is good if you want to create placeholders for several decision models at once and modify them at a later time. |

4. Click **Create**.

The decision model editor opens in the graph view. See [Understand Decision Model Views](#).

In the editor, you can:

- Add decisions and sub-decisions
- Define the input data
- Model the decision logic
- Test your decisions
- Create decision services to implement your decision model

Edit or Delete a Decision Model

You can edit an existing decision model by clicking and opening it in the decision model editor.

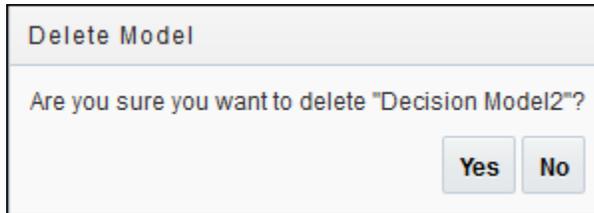
| Name | Space | Created By | Creation Date | Last Updated |
|----------------|----------|------------|-----------------------|-----------------------|
| Decision Model | My Space | William | 12/7/2018 11:25:07 PM | 12/7/2018 11:25:07 PM |

Note:

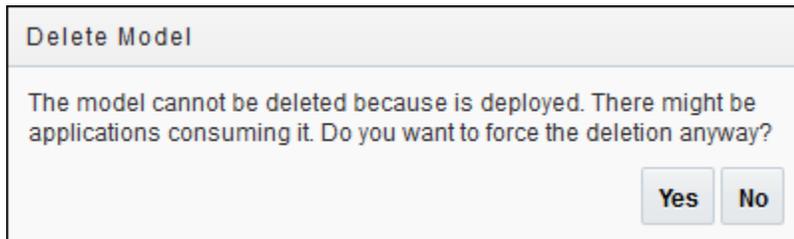
If you add a new service or change the service definition within your decision model, you'll need to create a new reference to the decision model on Process, re-implement the activity, and re-associate the data.

Within the selected decision model, click **Options**  of the decision, service, input data, or input type to edit it.

To delete a decision model, click **Options**  and then select **Delete Model**.



Deleting a decision model that is currently in use in one or more applications deactivates and deletes all the deployed decision model snapshots.



 **WARNING:**

Deleting a decision model or its snapshot that is currently in use in one or more applications does not delete references to it in the Process application. This may result in errors during runtime when the application tries to call the decision service associated with a decision model that has previously been deleted from the DMN server.

Understand Decision Model Views

In Processes, you can work on a decision model using either the graph view or the list view.

When you create a new decision model, you're presented with the graph view by default. You can switch to the list view and vice versa using the **Menu** button  in the toolbar.

 **Note:**

The decision models that you've created previously will continue to use the list view as the default view. If you switch to the graph view, you'll need to redefine the relationships between decisions and input data elements. Similarly, you'll have to redefine the order of the decisions if you switch from the graph view to the list view.

Topics:

- [Graph View](#)
- [List View](#)

Graph View

In the graph view, you can create decision requirement diagrams (DRD), in accordance with the DMN standard, to visually represent your decision models.

In decision modeling, a decision requirement graph (DRG) is a self-contained representation of an entire domain of decision-making that displays the important elements in the domain and the relationships between the elements. Whereas, a decision requirement diagram typically depicts a specific view of a DRG; the view depicted can either be partial or complete.

Processes currently supports one DRD per decision model. The following table lists all the available DRD components, which you can use to create your decision models.

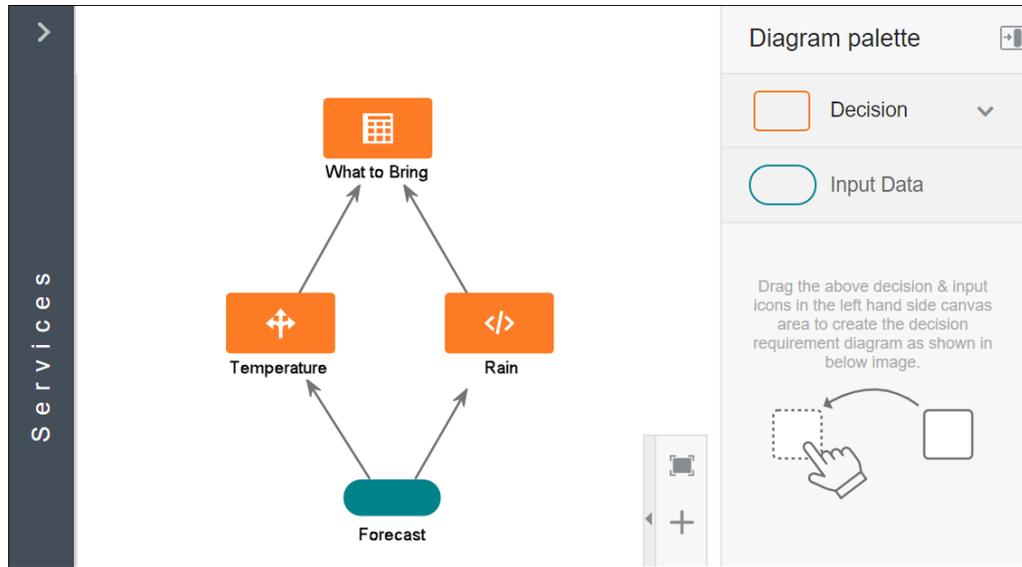
| DRD Component | | Description | Notation |
|---------------|-------------------------|---|---|
| Elements | Decision | Denotes a node that determines an output based on its inputs and the logic it contains. |  |
| | Input Data | Denotes information used as input by one or more decision elements. |  |
| Requirements | Information Requirement | Denotes the flow of information from an input data or a decision element to another decision element. |  |

The graph view is divided into the following three areas:

- [Toolbar](#)
- [Decision Model Canvas](#)
- [Diagram Palette](#)

It also contains the Services pane to the left, which is common to both views. To add a service to your model, see [Expose Decisions as Services](#).

The following figure shows a decision model in the graph view:



Toolbar

The decisions toolbar provides quick access controls to manage the decision canvas.



| Toolbar Icon | Name | Description |
|--------------|-------------|---|
| | Pan Mode | Pan the entire canvas. |
| | Select Mode | Select a particular DRD component. |
| | Undo | Reverse the last change made to your diagram. |
| | Redo | Reverse the last undo action you performed. |
| | Auto Layout | Change the layout of your diagram. Select from the available options. |
| | Maximize | Open the editor in an expanded view. |
| | Menu | Use options in the menu to switch between graph view and list view or promote the model to the gallery. |
| | Test | Test your decision model. |
| | Help | Access usage tips and the tutorial. |

Decision Model Canvas

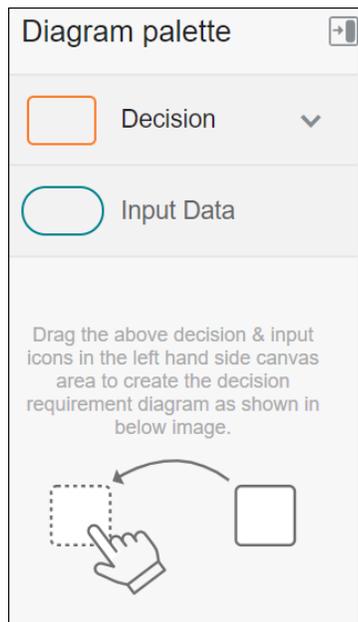
The decision model canvas is the central area of the graph view. Use the canvas to create a diagram that represents your decision model using the elements available in the diagram palette. The zoom controls at the bottom of the canvas allow you to zoom in or fit the canvas in your browser.

Diagram Palette

From the diagram palette, drag and drop DRD components onto the canvas to use them in your diagram.

Click **Expand**  next to **Decision** on the palette to view all the decision types available for use. You can drag a particular decision type or add an empty decision element and then associate a decision notation to it. Further, order the elements and form connections as required by your overall decision logic. See [Add and Order Decisions](#) and [Define Decision Input and Type](#)

Click **Collapse**  to collapse or expand the diagram palette.



List View

The list view provides a simple interface to add and order your decisions within a model.

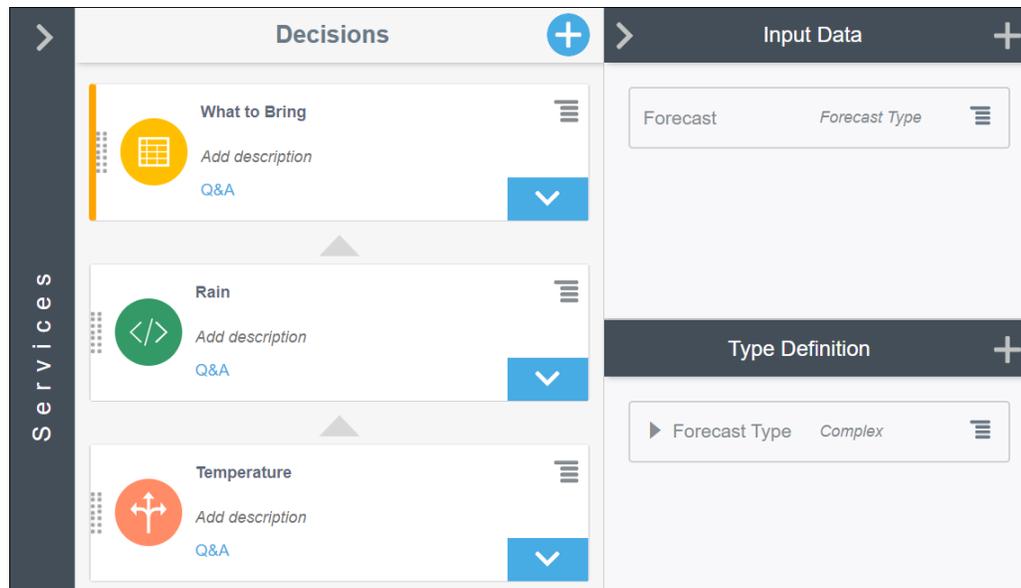
Like the graph view, this view too is divided into three areas, namely toolbar, decisions area, and input data. In addition, it contains the Services pane to the left.

Use the controls on the toolbar to undo or redo changes, switch to the graph view, test the decision model, or access relevant documentation. See [Toolbar](#).

In the decisions area, click **Add**  to add a new decision to the model. While adding a decision, you can also select its type and enter other preliminary details. The decisions you add are displayed in a list. By default, the first decision you add becomes the final or main decision of the model; however, you can reorder the decisions to suit your logic. See [Add and Order Decisions](#).

Further, expand the Input Data pane to add new type definitions and input data, and expand the Services pane to create a decision service. See [Define Decision Input and Type](#) and [Expose Decisions as Services](#).

The following figure shows a decision model created in the list view:



Add and Order Decisions

A decision is a container for logical notations such as decision tables, expressions, if-then rules, and so on. In Processes, you create a decision by selecting the logic type you plan to use to obtain a specific output.

Decision models contain two types of decisions:

- **Main decision:** The output of the main decision provides the result of the decision model.
- **Supporting decision:** One or more supporting decisions provide input to the main decision.

Add Decisions in Graph View

You can add a decision element to your model by dragging it from the diagram palette onto the canvas. After you add a decision, you define its properties, logic, and connections where they apply.

In the graph view, you can order decisions in several ways. For example, you can create a bottom-up sequence flow with the main decision at the top and supporting decisions below it, or a left-right flow, and so on. After adding all the decisions and connections, you can also click **Auto Layout** on the toolbar and select from a variety of layout or ordering options.

To add a decision to your model in the graph view:

1. On the diagram palette, click **Expand** ▼ next to **Decision**. The available decision types are:
 - Empty Decision
 - Decision Table
 - Expression
 - Context

- Loop
 - Function
 - If-Then-Else
 - List
 - Relation
2. Select and drag a decision type onto the canvas.
 3. Position the cursor at the point you want to add the decision and release the mouse click.
 4. To edit decision properties, select the decision and click **Open Properties** . The Decision Properties pane opens. Note that when you add any decision element to the canvas, a new decision logic with a default name (for example, Decision1) is created and associated with the element. You can edit the information related to the logic from the properties pane.
 5. Enter the necessary information about the decision logic in the properties pane.

| Field | Description |
|-----------------|---|
| Name | <p>Enter a unique name for the decision logic you want to create and associate with the element. You can also select a previously defined decision logic for the element. Click Expand  and select an implemented logic from the available options.</p> <div style="border: 1px solid #0070C0; padding: 10px; margin-top: 10px;"> <p> Note:</p> <p>If you add a particular decision type to the canvas and select a different type of logic from the Name drop-down menu, the element's logic type changes.</p> </div> <p>Click Edit  to edit the decision element's logic. You can also double-click element on the canvas to view or edit its logic. See sections within Model Decision Logic to know how to edit the logic for all decision types.</p> <p>Click Delete  to delete the decision logic associated with the element.</p> |
| Logic | <p>Change the decision logic type if required. For example, you can add an empty decision to the canvas and change its logic from the properties pane.</p> <div style="border: 1px solid #0070C0; padding: 10px; margin-top: 10px;"> <p> Note:</p> <p>If you select a previously implemented decision logic in the Name field and change its type in the Logic field, the content of the previous implementation is overwritten.</p> </div> |
| Description | Provide additional information, if any, for the decision logic. |
| Question | Add the questions allowed for the decision logic. |
| Allowed Answers | Add the possible answers for the decision logic. |

| Field | Description |
|--------------------|---|
| Select Output Type | Select the output type for the decision logic. Set this field if you are certain about the logic output type. If the type selected in this field does not match the actual output type, a validation error occurs. If no selection is made, the logic output type is automatically inferred from its inputs. |

6. If required, click **Add New Decision**  in the pane to create a new decision logic to associate with the element or other elements in the model. The new logic you create appears in the **Name** drop-down menu.

Click **Close**  to save the information and close the pane.

7. If the decision you've added is a supporting decision, create outgoing connections from it.
 - a. Select the element and click **Create New Connection**  and keep the mouse depressed.
 - b. Move the cursor to the decision you want to connect to, and then release the click.

You can create multiple outgoing connections from a decision. To delete a connection, hover your cursor over the arrow to select it, and then double-click.

8. To delete the decision element you added to the canvas, select the element and click **Delete** .

Add Decisions in List View

In the list view, the decision order is important. The topmost decision is the main decision. Supporting decisions are listed below it. The main decision can reference the result of supporting decisions by name. Supporting decisions must always be beneath the decision that references them.

To change the order of decisions in the model, drag the decision by its dotted handle and drop it at the desired position so that supporting decisions are always beneath the decisions that they support.

To create a new decision in list view:

1. Click **Add new decision**  in the Decision Model home page to open the Create Decision dialog box.
2. Choose the logic type you want to use, for example, Decision Table or Context
3. Enter the required information into the Create Decision dialog box.

| Field | Description |
|-----------------|---|
| Name | Enter a unique name for the decision. |
| Description | Provide additional information, if any, for the decision. |
| Question | Add the questions allowed for the decision. |
| Allowed Answers | Add the possible answers for the decision. |

| Field | Description |
|--------------------|---|
| Select Output Type | Select the output type for the decision. Set this field if you are certain about the decision output type. If the type selected in this field does not match the actual output type, a validation error occurs. If no selection is made, the decision output type is automatically inferred from its inputs. |

4. Click **Create**.

Use the **Options**  menu of a decision to edit or delete it.

Define Expressions with the Friendly Enough Expression Language (FEEL)

Decision Modeling and Notation (DMN) defines Friendly Enough Expression Language (FEEL) to provide standard executable semantics to all expressions used within a decision model.

In Process, you use FEEL to define expressions within all notations of decision logic, including decision tables.

Topics:

- [Data Types](#)
- [Grammar Rules](#)
- [Built-In Functions](#)
- [List Iteration Expressions](#)
- [Working with Date, Time, and Duration Functions](#)

Data Types

Process supports the following FEEL data types that you can use for input data, expression values, function arguments and return values.

| FEEL Data Type | Notation in Processes | Description |
|----------------|-----------------------|--|
| number | Number | FEEL Numbers are based on IEEE 754-2008 Decimal128 format, with 34 decimal digits of precision and rounding toward the nearest neighbor with ties favoring the even neighbor. Numbers are a restriction of the XML Schema type <code>precisionDecimal</code> , and are equivalent to Java <code>BigDecimal</code> with <code>MathContext DECIMAL128</code> . |
| string | Text | Variable-length sequence of characters italicized or encapsulated in double quotes. |
| boolean | True or False | Logical Boolean (true/false). |
| date-time | Date and Time | Calendar date and time combination. |

**Note:**

You can extend these basic data types by defining custom data types. See [Define Custom Data Types](#).

Grammar Rules

Learn about the syntax for commonly-used FEEL expressions through simple examples. For the complete definition of FEEL syntax, see *Decision Model and Notation (DMN), v1.1*.

Topics:

- [Arithmetic Expressions](#)
- [Interval Expressions](#)
- [Comparison Expressions](#)
- [Other Expressions](#)

Arithmetic Expressions

| Name | FEEL Expression | Return Value |
|---------------------|-----------------|--------------|
| Addition (+) | 0.15+30 | 30.15 |
| Subtraction (-) | 15-30 | -15 |
| Multiplication (*) | .20*40.02 | 8.004 |
| Division (/) | 1/50 | 0.02 |
| Exponentiation (**) | 2**3 | 8 |

Interval Expressions

| Start Value | End Value | FEEL Expression | Return Value |
|-------------|-----------|-----------------|--------------|
| Inclusive | Inclusive | 15 in [15..30] | true |
| Exclusive | Exclusive | 15 in (15..30) | false |
| Exclusive | Inclusive | 30 in (15..30] | true |
| Inclusive | Exclusive | 30 in [15..30) | false |

**Note:**

In decision tablet input entry and input/output allowed value cells, you can use intervals or list of intervals to test against the input data.

Comparison Expressions

| Name | FEEL Expression | Return Value |
|---------------|-----------------|--------------|
| Less than (<) | 8<2**3 | false |

| Name | FEEL Expression | Return Value |
|-------------------------------|-----------------|--------------|
| Less than or equal to (<=) | 15 in (<=15) | true |
| Equal (=) | 8=2**3 | true |
| Greater than (>) | 30 in (>30) | false |
| Greater than or equal to (>=) | 1/5>=0.20 | true |
| Not equal to (!=) | 8!=2**3 | false |

 **Note:**

In decision table input entry and input/output allowed value cells, you can use comparison operators to define unary expressions.

Other Expressions

| Name | FEEL Expression | Return Value |
|-------------|---------------------------|--------------|
| Disjunction | (2*2=2**2) or (3*2=3**2) | true |
| Conjunction | (2*2=2**2) and (3*2=3**2) | false |
| Negation | not(2*2=2**2) | false |

 **Note:**

In decision tablet input entry and input/output allowed value cells, you can use comma-separated list of values to specify disjunction.

Built-In Functions

FEEL includes a library of built-in functions that you can use to define expressions.

Process supports the following types of built-in functions:

- [Conversion Functions](#)
- [Boolean Functions](#)
- [String Functions](#)
- [List Functions](#)
- [Numeric Functions](#)

Conversion Functions

| Name(parameters) | Parameter Domain | Description | Example |
|---------------------|------------------|-------------------------------|---|
| date(<i>from</i>) | date string | convert <i>from</i> to a date | date("2012-12-25") - date("2012-12-24") = duration("P1D") |

| Name(parameters) | Parameter Domain | Description | Example |
|--|-------------------------------------|---|--|
| <code>date(from)</code> | date and time | convert <i>from</i> to a date (set time components to null) | <code>date(date and time("2012-12-25T11:00:00Z")) = date("2012-12-25")</code> |
| <code>date and time(from)</code> | date time string | convert <i>from</i> to a date and time | <code>date and time("2012-12-24T23:59:00") + duration("PT1M") = date and time("2012-12-25T00:00:00")</code> |
| <code>time(from)</code> | time string | convert <i>from</i> to time | <code>time("23:59:00") + duration("PT2M") = time("00:01:00")</code> |
| <code>time(from)</code> | time, date and time | convert <i>from</i> to time (ignoring date components) | <code>time(date and time("2012-12-25T11:00:00Z")) = time("11:00:00")</code> |
| <code>number(from, grouping separator, decimal separator)</code> | number, separator, decimal notation | convert <i>from</i> to a number | <code>number("1 000,0", " ", ";") = number("1,000.0", ",", ".")</code> |
| <code>string(from)</code> | non-null | convert <i>from</i> to a string | <code>string(1.1) = "1.1" string(null) = null</code> |
| <code>duration(from)</code> | duration string | convert <i>from</i> to a days and time or years and months duration | <ul style="list-style-type: none"> <code>date and time("2012-12-24T23:59:00") - date and time("2012-12-22T03:45:00") = duration("P2DT20H14M")</code> <code>duration("P2Y2M") = duration("P26M")</code> |
| <code>years and months duration(from, to)</code> | both are date and time | return years and months duration between <i>from</i> and <i>to</i> | <code>years and months duration(date("2011-12-22"), date("2013-08-24")) = duration("P1Y8M")</code> |

Boolean Functions

| Name(parameters) | Parameter Domain | Description | Example |
|--------------------------|------------------|------------------|---|
| <code>not(negand)</code> | boolean | logical negation | <ul style="list-style-type: none"> <code>not(true) = false</code> <code>not(null) = null</code> |

String Functions

| Name(parameters) | Parameter Domain | Description | Example |
|---|------------------|--|--|
| <code>substring(string, start position, length?)</code> | string, number1 | return <i>length</i> (or all) characters in <i>string</i> , starting at <i>start position</i> . 1 st position is 1, last position is -1 | <ul style="list-style-type: none"> <code>substring("redwood",3) = "dwood"</code> <code>substring("redwood",3,3) = "dwo"</code> <code>substring("redwood", -2, 1) = "o"</code> |
| <code>string length(string)</code> | string | return length of <i>string</i> | <code>string length("red") = 3</code> |
| <code>upper case(string)</code> | string | return uppercased <i>string</i> | <code>upper case("aBc4") = "ABC4"</code> |

| Name(parameters) | Parameter Domain | Description | Example |
|---|------------------|--|--|
| lower case(<i>string</i>) | string | return lowercased <i>string</i> | lower case("aBc4") = "abc4" |
| substring before (<i>string</i> , <i>match</i>) | string, string | return substring of <i>string</i> before the <i>match</i> in <i>string</i> | <ul style="list-style-type: none"> substring before("redwood", "wood") = "red" substring before("redwood", "xyz") = "" |
| substring after (<i>string</i> , <i>match</i>) | string, string | return substring of <i>string</i> after the <i>match</i> in <i>string</i> | <ul style="list-style-type: none"> substring after("redwood", "dw") = "ood" substring after("", "o") = "" |
| replace(<i>input</i> , <i>pattern</i> , <i>replacement</i> , <i>flags?</i>) | string2 | regular expression pattern matching and replacement | replace("abcd", "(ab) (a)", "[1=\$1][2=\$2]") = "[1=ab][2=]cd" |
| contains(<i>string</i> , <i>match</i>) | string | does the <i>string</i> contain the <i>match</i> ? | contains("redwood", "de") = false |
| starts with(<i>string</i> , <i>match</i>) | string | does the <i>string</i> start with the <i>match</i> ? | starts with("redwood", "re") = true |
| ends with(<i>string</i> , <i>match</i>) | string | does the <i>string</i> end with the <i>match</i> ? | ends with("redwood", "d") = true |
| matches(<i>input</i> , <i>pattern</i> , <i>flags?</i>) | string2 | does the <i>input</i> match the regexp <i>pattern</i> ? | matches("redwood", "^re*w") = true |

List Functions

| Name(parameters) | Parameter Domain | Description | Example |
|---|--|--|------------------------------------|
| list contains(<i>list</i> , <i>element</i>) | list, any element of the semantic domain including null | does the <i>list</i> contain the <i>element</i> ? | list contains([1,2,3], 2) = true |
| count(<i>list</i>) | list | return size of <i>list</i> | count([1,2,3]) = 3 |
| minimum(<i>list</i>) | (list of) comparable items | return minimum item | minimum([1,2,3]) = 1 |
| maximum(<i>list</i>) | (list of) comparable items | return maximum item | maximum([1,2,3]) = 3 |
| sum(<i>list</i>) | (list of) numbers | return sum of numbers | sum([1,2,3]) = 6 |
| mean(<i>list</i>) | (list of) numbers | return arithmetic mean (average) of numbers | mean([1,2,3]) = 2 |
| sublist(<i>list</i> , <i>start position</i> , <i>length?</i>) | list, number1, number2 | return list of <i>length</i> (or all) elements of <i>list</i> , starting with <i>list[start position]</i> . 1 st position is 1, last position is -1 | sublist([1,2,3], 1, 2) = [2] |
| append(<i>list</i> , <i>item...</i>) | list, any element including null | return new <i>list</i> with <i>items</i> appended | append([1], 2, 3) = [1,2,3] |
| concatenate(<i>list...</i>) | list | return new <i>list</i> that is a concatenation of the arguments | concatenate([1,2],[3]) = [1,2,3] |
| insert before(<i>list</i> , <i>position</i> , <i>newItem</i>) | list, number1, any element including null | return new <i>list</i> with <i>newItem</i> inserted at <i>position</i> | insert before([1,3],1,2) = [1,2,3] |

| Name(parameters) | Parameter Domain | Description | Example |
|-------------------------------------|---|--|---|
| <code>remove(list, position)</code> | list, number1 | <i>list</i> with item at <i>position</i> removed | <code>remove([1,2,3], 2) = [1,3]</code> |
| <code>reverse(list)</code> | list | reverse the <i>list</i> | <code>reverse([1,2,3]) = [3,2,1]</code> |
| <code>index of(list, match)</code> | list, any element including null | return ascending list of <i>list</i> positions containing <i>match</i> | <code>index of([1,2,3,2],2) = [2,4]</code> |
| <code>union(list...)</code> | list | concatenate with duplicate removal | <code>union([1,2],[2,3]) = [1,2,3]</code> |
| <code>distinct values(list)</code> | list | duplicate removal | <code>distinct values([1,2,3,2,1]) = [1,2,3]</code> |
| <code>flatten(list)</code> | list | flatten nested lists | <code>flatten([[1,2],[[3]], 4]) = [1,2,3,4]</code> |

Numeric Functions

| Name(parameters) | Parameter Domain | Description | Example |
|--------------------------------|------------------|---|--|
| <code>decimal(n, scale)</code> | number, number1 | return <i>n</i> with given <i>scale</i> | <ul style="list-style-type: none"> <code>decimal(1/3, 2) = .33</code> <code>decimal(1.5, 0) = 2</code> <code>decimal(2.5, 0) = 2</code> |
| <code>floor(n)</code> | number | return greatest integer $\leq n$ | <ul style="list-style-type: none"> <code>floor(1.5) = 1</code> <code>floor(-1.5) = -2</code> |
| <code>ceiling(n)</code> | number | return smallest integer $\geq n$ | <ul style="list-style-type: none"> <code>ceiling(1.5) = 2</code> <code>ceiling(-1.5) = -1</code> |

List Iteration Expressions

This sections lists a few frequently-used list iteration expressions.

| Name(parameters) | Description | Example |
|---|---|--|
| <code>for [item] in [list] return [expression]</code> | iterate over a list | <code>for i in [1,2,3,4] return i*i = [1,4,9,16]</code> |
| <code>sum (for [item] in [list] return [expression])</code> | iterate over a list and return the sum of iterations | <code>sum(for i in [1,2,3,4] return i*) = 30</code> |
| <code>every [item] in [list] satisfies [expression]</code> | test if every item in the list satisfies the test condition described by the expression. | <ul style="list-style-type: none"> <code>every n in [12,50,51] satisfies $n > 5$ = true</code> <code>every n in [12,50,51] satisfies $n < 50$ = false</code> |
| <code>some [item] in [list] satisfies [expression]</code> | test if at least one of the items in the list satisfies the test condition described by the expression. | <ul style="list-style-type: none"> <code>some n in [12,50,51] satisfies $n > 50$ = true</code> <code>some n in [12,50,51] satisfies $n > 51$ = false</code> |

Date, Time, and Duration Functions

Because FEEL does not support literal representation of date, time, or duration values, you must use a combination of built-in functions and string/number literals to express these values.

Built-in functions extract date, time, and duration data from strings or numbers. The following sections provide examples of a few frequently-used date and time operations using built-in functions:

- [Conversion Operations](#)
- [Arithmetic Operations](#)
- [Comparison Operations](#)

Conversion Examples

Examples in this section demonstrate conversion of number or string literals into date, time, or duration data types.

Date and Time Data Type Examples

The following examples convert string literals to date or date-time values:

| Example | Description |
|--|---|
| <code>date("2012-12-25")</code> | Represents the date 2012-12-25 in the <i>YYYY-MM-DD</i> format |
| <code>time("23:59:00")</code> | Represents the time 23:59:00 in the <i>hh:mm:ss</i> format |
| <code>time("23:59:00-08:00")</code> | Represents the local time 23:59:00 in the <i>hh:mm:ss</i> format; the local time is 8 hours behind UTC. |
| <code>date and time("2012-12-25T11:00:00")</code> | Represents the date 2012-12-25 and time 11:00:00 in <i>YYYY-MM-DD</i> and <i>hh:mm:ss</i> formats respectively |
| <code>date and time("2012-12-25T11:00:00Z")</code> | Represents the date 2012-12-25 and time 11:00:00 in <i>YYYY-MM-DD</i> and <i>hh:mm:ss</i> formats respectively; "Z" represents UTC time |

Duration Data Type Examples

A few conversion examples using the duration function are listed in the following table:

| Example | Description |
|---|---|
| <code>duration("P1DT12H30M")</code> | Represents a duration of 1 day, 12 hours, and 30 minutes |
| <code>duration("-P120D")</code> | Represents a duration of minus 120 days |
| <code>duration("PT2000H")</code> | Represents a duration of 2000 hours |
| <code>duration("P1Y2M3DT10H30M")</code> | Represents a duration of 1 year, 2 months, 3 days, 10 hours, and 30 minutes |

Arithmetic Operation Examples

Examples in this section demonstrate arithmetic operations that can be performed on date, time, or duration data types.

| Operator | Example | Result | Description |
|----------|--|---------------------------------------|--|
| (+) | date("2016-12-25") + duration("P3Y2M6D") | date("2020-03-02") | Returns the date 3 years 2 months and 6 days after 2016-12-25 |
| (+) | date and time("2016-12-25T12:30:00Z") + duration("P1Y2M3DT10H30M") | date and time("2018-02-28T23:00:00Z") | Returns the date and time 1 year 2 months 3 days and 10 hours 30 minutes after the date 2016-12-25 and time 12:30:00 |
| (+) | date("2016-12-25") + duration("-P3Y2M6D") | date("2013-10-19") | Returns the date 3 years 2 months and 6 days before 2016-12-25 |
| (-) | date("2015-12-25") - date("2012-12-25") | duration("P1095DT0H0M0S") | Returns a duration indicating number of days and time |
| (-) | date and time("2012-12-25T12:00:00") - date and time("2015-12-25T11:12:00") | duration("-P1094DT23H12M0S") | Returns a duration indicating number of days and time |
| (-) | date and time("2012-12-25T12:00:00-08:00") - date and time("2015-12-25T11:12:00Z") | duration("-P1094DT15H12M0S") | Returns a duration indicating the number of days and time between date and time of two different time zones. |
| (-) | date("2016-12-25") - duration("P3Y2M6D") | date("2013-10-19") | Returns the date 3 years 2 months and 6 days before 2016-12-25 |
| (/) | (date("2015-12-25") - date("2012-12-25"))/duration("P1D") | 1095 | Returns the number of days between two dates |
| (/) | (date and time("2015-12-25T17:00:00") - date and time("2015-12-25T09:12:00"))/duration("PT1H") | 7.8 | Returns the number of hours between two date and time values |
| (/) | (date("2015-12-25") - date("2015-12-24"))/duration("P1Y") | 0.0027397260273972603 | Returns the number of years between two dates |
| None | years and months duration(date("2012-12-23"), date("2015-12-25")) | duration("-P3Y0M") | Returns the years and months duration between two dates |

Comparison Operation Examples

Examples in this section demonstrate comparison operations that can be performed on date or time data types.

| Operator | Example | Result | Description |
|----------|---|--------|---|
| > | date("2012-12-25") > date("2015-12-25") | false | Determines if Date A occurs after Date B |
| > | ((date("2015-12-25") - date("2015-11-25")) > duration("P1Y")) | false | Determines if Duration A is greater than Duration B |

Define Decision Input and Type

An input data variable is a placeholder for information that is to be supplied to a decision model when the model is invoked.

Supported data types for input data are text, number, boolean (true or false), and date and time. See [Data Types](#). You can extend the built-in data types to define custom complex data types.

Topics:

- [Create Input Data](#)
- [Define Custom Data Types](#)

Create Input Data

Use built-in or custom data types to create input variables for the decision model.

Create Input Variables in Graph View

1. On the diagram palette, select the **Input Data** element and drag it onto the canvas.
2. Position the cursor at the point you want to add the element and release the mouse click.
3. To edit the element's properties, select the element and click **Open Properties** . The Input Data Properties pane opens. Note that when you add an input data element to the canvas, a new input variable with a default name (for example, inputData1) and type is created and associated with the element. You can edit the information related to the variable from the properties pane.
4. In the pane:
 - a. Enter a suitable name for the variable you want to create and associate with the element. You can also select a previously defined variable for the element. Click **Expand**  and select a variable from the available options.
Click **Delete**  to delete the variable associated with the element.
 - b. Change the **Mode** (that is, the data type) for the variable. You can select one of the built-in data types or choose a custom data type (**Other Type**).
 - If you select a built-in type, you can specify if the variable is a list by selecting the **isList?** check box. Additionally, you can restrict the variable to enumerated values or ranges.
 - If you select **Other Type**, you can either define a new data type or use a previously defined custom data type. See [Define Custom Data Types in Graph View](#).

Note:

If you select a previously defined variable in the **Name** field, the mode and other details are auto-populated.

Click **Close**  to save the details and close the pane.

5. Now, create outgoing connections from the input data element to the decisions in the model.
 - a. Select the element and click **Create New Connection**  and keep the mouse depressed.
 - b. Move the cursor to the decision you want to connect to, and then release the click.

You can create multiple outgoing connections from an input data element. To delete a connection, hover your cursor over the arrow to select it, and then double-click.

6. To delete the input element from the canvas, select it and click **Delete** .

Create Input Variables in List View

1. In the Input Data section, click **Add new input data**  to open the Add Input Data panel.
2. Enter a suitable name for the variable.
3. Select the **Mode** (that is, the data type) for the variable. You can select one of the built-in data types or choose a custom data type (**Other Type**).
 - a. If you select a built-in type, you can specify if the variable is a list by selecting the **isList?** check box. Additionally, you can restrict the variable to enumerated values or ranges.

- b. If you select **Other Type**, you can either define a new type definition or use a previously defined custom data type. See [Define Custom Data Types in List View](#).
4. Click **Close** to save the variable. You can edit or delete an input variable using the **Menu** .

Define Custom Data Types

If built-in data types aren't suitable for an input variable in your decision model, you can create a custom data type.

A custom data type can be a new complex data type or an alias of a built-in data type. Further, to create a complex data type, you can use a combination of built-in data types or other complex data types.

Define Custom Data Types in Graph View

1. Select the input data element on the canvas for which you intend to define and use a variable of custom data type.
2. Click **Open Properties** .
3. In the properties pane:
 - a. Select the **Mode** as **Other Type**.
 - b. Now, click the **Define New Type** link that appears in the pane. A new type definition is created with a default name and the Add Type Definition panel opens for editing.
4. In the type definition panel:
 - a. Edit the definition's name if necessary.
 - b. Select the **Mode** (that is, the data type) for the definition. You can select one of the built-in data types to create aliases or choose to define a complex data type.
 - c. To identify the data type as a list, select the **isList?** check box.
 - d. If you select the mode as **Text**, **Number**, or **Date and Time**, you can optionally define allowed values to restrict the data type definition to enumerated values or ranges.
 - e. If you select the mode as **Complex**, define attributes within the type definition.
 - i. Click **Add Component**  to add a new attribute. An attribute is created with a default name and the Add Component panel opens for editing.
 - ii. In the panel, edit the attribute's name, select the mode, and enter allowed values or a range for the attribute.
 - iii. Select the **isList?** check box to identify the attribute as a list.
 - iv. Click **Close**  to return to the Add Type Definition panel, and repeat steps i-iii to add another attribute to the complex type definition.
 - v. Use **Edit** or **Delete** buttons to modify the attributes.
 - f. Click **Close**  to save and close the type definition panel. The following image shows an example definition:

| Name | Type | Value/Values | is List |
|---------------|--------|--------------|---------|
| Name | Text | Text | false |
| Phone | Number | Number | false |
| Account Nu... | Number | Number | false |

- In the Input Data Properties pane, choose the type definition you just defined in the **Other Types** field to associate it with your input data variable. You can edit or delete the type definition using the buttons next to the field. You can also use this new type definition to create variables in other input data elements within the model.

Define Custom Data Types in List View

- In the Type Definition section of the Input Data pane, click **Add new item definition** . A new type definition is created with a default name and the Add Type Definition panel opens for editing.
- Edit the type definition's name if necessary.
- Select the **Mode** (that is, the data type) for the definition. You can select one of the built-in data types to create aliases or choose to define a complex data type.
- To identify the data type as a list, select the **isList?** check box.
- If you select the mode as **Text**, **Number**, or **Date and Time**, you can optionally define allowed values to restrict the data type definition to enumerated values or ranges.
- If you select the mode as **Complex**, define attributes within the type definition.
 - Click **Add Component** to add a new attribute. An attribute is created with a default name and the Add Component panel opens for editing.
 - In the panel, edit the attribute's name, select the mode, and enter allowed values or a range for the attribute.
 - Select the **isList?** check box to identify the attribute as a list.
 - Click **Close** to return to the Add Type Definition panel, and repeat steps a-c to add another attribute to the Complex type definition.

- e. Use **Edit** or **Delete** buttons to modify the attributes.
7. Click **Close** to save the data type definition. You can edit or delete a type definition using the **Menu** .

Model Decision Logic

Model the decision logic by defining how each decision's output is derived from its inputs.

You can use a bottom-up approach to define the logic within a decision model.

1. To begin with, edit the supporting decision lowest in the order; that is, the decision that'll serve as input to other supporting decisions.
2. Choose a suitable notation to obtain the required result, and configure logic to it using the input data variables. You can choose from several notations available, such as decision table, if-else expression, context, and so on.
3. Proceed to the next level in the decision framework, and define the logic for other supporting decisions. You can use input data and results of other decisions while configuring the logic of a decision
4. Lastly, edit the main output decision, which provides the result of the decision model.
5. Finalize the decision model by iteratively fine-tuning the decision framework and the logic within each decision.

In Processes, you have the following notations to model the logic within a decision:

- [Create Empty Logic Decisions](#)
- [Create Decision Tables](#)
- [Create Expressions](#)
- [Create If-Then-Else Statements](#)
- [Create Functions](#)
- [Create Contexts](#)
- [Create Lists](#)
- [Create Relations](#)
- [Create Loops](#)

Unsure which notation to use, or how to best use it? See [Best Practices for Modeling Decision Logic](#).

Create Empty Logic Decisions

While outlining the framework of a decision model, you can create a decision with the Empty Logic notation as a placeholder for actual logic.

To create a decision with the Empty Logic notation,

- In graph view, click **Expand**  next to **Decision** on the diagram palette. Select **Empty Decision** and drag it onto the canvas.
- In list view, click **Add new decision**  in the Decisions bar and select **Empty Logic** from the Create Decision window.

8. Add Annotation button – Adds a column for documenting or annotating decision rules. Annotations aren't considered as part of the decision logic; they serve as explanatory notes for designers.

 **Note:**

The **Add Annotation** button disappears after you add the first annotation column. However, you can add additional columns for annotations using the column add controls.

9. Row and column controls – Add or remove rows and columns.
10. Expand Table – Opens the table in an expanded view.

The following sections explain each decision table element in detail:

- [Defining Decision Table Input](#)
- [Defining Decision Table Output](#)
- [Configuring Rules](#)
- [Configuring a Hit Policy](#)

Define Decision Table Input

An input (also referred to as input clause) to a decision table consists of an input expression and several input entries. It is represented as a column within a table. A decision table may have multiple inputs.

Specify Input Expressions

In a decision table, you enter an input expression in the header cell of an input column. In combination with input entries, an input expression determines the value of a particular input column. It can be a simple test expression, for example, `Age>50`. You can use input variables, outputs of other decisions, or built-in functions to define input expressions.

In an Input Expression cell, press **Ctrl+Space** to open a drop-down menu. Use the options available in the menu to define your expression. You can also click on the Enter Allowed Values cell to enter or edit the input expression.

In the Decision Model editor, the expression language used for all expressions, including input expressions, is friendly enough expression language (FEEL). Want to learn more about the expression language syntax? See [Understanding FEEL](#).

Allowed Values

Click on the Enter Allowed Values cell within an Input Expression or Output Label cell to specify permitted values for cell entries of that column. However, for an input column, the type of allowed values (or specific allowed values) are automatically populated when you specify the input expression for the column. For example, if an input expression returns Boolean values, allowed values for that column are populated as *true*, *false*. The options you can choose from for changing the type of allowed values are listed in the following table. Note that after you specify the input expression for a column, the type of allowed values you can toggle between are limited.

| Allowed Value Type | Description |
|--------------------|---|
| Auto | The DMN server determines allowed entries for the column based on the input expression. This is the default selection for a new input column. |
| Any | Use this option to specify that there are no restrictions on the data type of entries. Optionally, restrict entries to a given value or list of values. This is the default selection for the output column. |
| Text | Use this option to restrict entries to text strings. Optionally, specify a particular string or list of strings. |
| Number | Use this option to restrict entries to numbers. Optionally, specify a particular number, limit, or range. |
| Date and Time | Use this option to restrict entries to date and time values. Optionally, specify a particular date and time value, limit, or range. |
| True or False | Use this option to permit only Boolean entries. |
| Other Type | Use this option to restrict entries to a given custom data type. |
| Advanced | Use this option to permit FEEL expressions and <i>null</i> values as entries. Optionally, specify constraints using FEEL expressions. |

 **Note:**

The data type of input entry cells is determined by the data type of the input expression. Make sure that the type of allowed values you supply is consistent with the data type of the input clause.

Specify Input Entries

You can enter strings, numbers, Boolean values, date and time values, or FEEL expressions as input entries based on the mode selected. If you've provided specific allowed values or if your input expression returns a finite set of values, an auto-suggest menu appears when you click on an input entry cell.

 **Note:**

If the data type of an input entry does not match the data type of the column, or if the input entry is not among the allowed values, an error is displayed within the decision.

Modes

Based on the input expression and/or allowed values you specify, the mode for entry cells is automatically selected. If required, click on the **Mode** icons to switch to a different mode. The following figure shows the Mode Editor window with Text mode selected:

Decision Rule Input

Mode
Text ▼

Allowed Values
equals ▼ Not

Value
umbrella ?

[Close](#)

The mode options available to you in the editor is dependent on the data type of the input expression or allowed values you've specified. The following table details all the available modes:

| Mode | Description |
|---|--|
| Any  | Use this mode to mark an entry as irrelevant (-). |
| Text  | Use this mode to enter strings. In this mode, you can enter a string as a plain literal without double quotes. |
| Number  | Use this mode to enter numbers. Optionally, use the constraint options available in the Mode editor. |
| True or False  | Use this mode to enter Boolean values. |
| Date and Time  | Use this mode to enter date and time values. Optionally, use the constraint options available in the Mode editor. |
| Advanced  | Use this mode to enter advanced FEEL expressions and <i>null</i> values. You can use the constraint options available in the Mode editor to define an expression. Want to learn more about the syntax and examples? See Grammar Rules . |

Define Decision Table Output

An output (also referred to as output clause) of a decision table consists of an output label and several output entries. When you create a new decision table, a table with a single output column appears by default. To add additional outputs, select the existing output column and use the column add controls.

Specify Output Labels

Generally, for single-output tables, the output label is the name of the decision table. In the Output Label cell, you can specify the allowed values for output entries. See Allowed Values in [Defining Decision Table Input](#).

Specify Output Entries

You can enter text, numbers, boolean values (true or false), date and time values, or FEEL expressions as output entries based on the mode selected. See Modes in [Defining Decision Table Input](#).

Based on the allowed values you specify, an auto-suggest menu appears when you click on an output entry cell.

You can use input variables, outputs of other decisions, or built-in functions to specify FEEL expressions for output entries.

The following figure shows an example decision table with two output columns:

The screenshot shows a decision table titled 'LoanInterest' with the description 'This decision table calculates housing loan interest rates for individuals.' The table has five columns: 'U' (row identifier), 'Salaried' (input, true/false), 'ExistingCustomer' (input, true/false), 'BaseRate' (output, 'Enter Allowed Values'), 'MaxTenure' (output, 'Enter Allowed Values'), and 'Annotations'. There are four rows of data:

| U | Salaried <i>true, false</i> | ExistingCustomer <i>true, false</i> | BaseRate <i>Enter Allowed Values</i> | MaxTenure <i>Enter Allowed Values</i> | Annotations |
|---|--------------------------------|--|---|--|--------------|
| 1 | true | true | 7.5 | 25 | |
| 2 | false | true | 8.0 | 20 | Conditional. |
| 3 | true | false | 7.8 | 20 | |
| 4 | false | false | 9.5 | 15 | |

For a salaried, existing customer, the output returned is as follows:

| | |
|-----------|-----|
| BaseRate | 7.5 |
| MaxTenure | 25 |

To reference a particular output of this multi-output table from another decision, use the following format: *DecisionName.OutputLabel*; for example, *LoanInterest.BaseRate*.

Configure Rules

Rules are expressed as rows within a table. Every rule consists of one or more input entries and a corresponding output entry.

Generally, a decision table consists of multiple rules. When the input data matches the input entries of a rule, the result of the decision table contains the output entry of the rule.

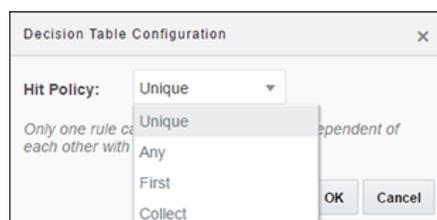
Configure a Hit Policy

The Hit Policy Indicator cell displays the hit policy selected for the table. The hit policy determines the output of a decision table from the output cells of matched rules. A rule is matched when all of its condition cells match the inputs of the decision table.

Based on the hit policy, Decision Model Notation (DMN) broadly groups decision tables into the following categories:

1. **Single Hit:** A single-hit table returns the output of only one rule. Under the single-hit category, Process supports the following hit policies:
 - **Unique (U)** – Only one of the rules can match.
 - **Any (A)** – Multiple rules can match, but all matching rules must have the same output.
 - **First (F)** – Multiple rules can match; the output of the first rule that matches is returned.
 - **Priority (P)** – Multiple rules can match; the output value that has the highest priority is returned.
2. **Multiple Hit:** A multiple-hit table returns the output of multiple rules. Under this category, the following hit policies are supported in Process:
 - **Collect (C)** – Multiple rules can match; outputs are returned as an arbitrarily-ordered list.
 - **Collect Sum (C+)** – Multiple rules can match; the sum of outputs is returned.
 - **Collect Min (C<)** – Multiple rules can match; the smallest output value is returned.
 - **Collect Max (C>)** – Multiple rules can match; the largest output value is returned.
 - **Collect Count (C#)** – Multiple rules can match; the count is returned.

When you create a new table, the Unique (U) hit policy is selected by default. To change the policy, click the Hit Policy Indicator cell and choose from the available options in the Hit Policy drop-down list. If rules within the table violate the selected hit policy, a warning is displayed within the decision.



Hit Policy Examples

Here are examples for all the hit policies.

- **Single Hit Unique**

In a decision table with Unique hit policy, only one rule can match. All rules are independent of each other, and no overlap is permitted. The decision table returns the output of the rule that matches.

Here is a decision table created with the Unique hit policy:

| U | Temperature | What to Wear |
|---|-------------|----------------------|
| | - | Enter Allowed Values |
| 1 | <25 | Wool coat |
| 2 | 25 | Jacket |
| 3 | >25 | Casuals |

In this example, for any input value of temperature, only one rule can match.

- **Single Hit Any**

In a decision table with Any hit policy, multiple rules can match. The overlap is permitted only if the matching rules have the same output. The decision table returns the output of any one of the matching rules. The hit policy is breached if matching rules have different outputs.

Here is a decision table created with the Any hit policy:

| A | Age | Work_Experience | Bonus Percent |
|---|------|-----------------|----------------------|
| | - | - | Enter Allowed Values |
| 1 | - | >=20 | 10 |
| 2 | >=50 | - | 10 |
| 3 | <50 | <20 | 5 |

In this example, for an input age of 50 and work experience of 20, the first and second rules match. This overlap is allowed because these rules have the same output. The decision table returns the output of any one of these rules.

- **Single Hit First**

In a decision table with First hit policy, multiple rules with different output entries can match. The output of the lowest-numbered matching rule is the result of the table.

Here is a decision table created with the First hit policy:

| Decision Table | | Vacation Days |
|----------------|---------------|----------------------|
| F | Service_Years | Vacation Days |
| | - | Enter Allowed Values |
| 1 | <5 | 5 |
| 2 | >=5 | 10 |
| 3 | >10 | 15 |

In this example, if service years are 11, the second and third rules match. The decision table returns only the second rule's output.

- **Single Hit Priority**

In a decision table with Priority hit policy, multiple rules with different output entries can match. The priority of output values (in descending order) is specified as a list in the Allowed Values cell of the output column. The decision table returns the output value that has the highest priority among outputs of all matching rules.

Here is a decision table created with the Priority hit policy:

| P | Age | Discount Percent |
|---|----------|------------------|
| | - | 5, 15, 10 |
| 1 | <18 | 15 |
| 2 | [18..45] | 5 |
| 3 | >45 | 10 |
| 4 | >60 | 15 |

In this example, the last two rules match for an input age of 61. The decision table returns the output value that has the highest priority among of these rules, that is, 15; the priority order is defined in the Allowed Values cell.

- **Multiple Hit Collect**

In a decision table with Collect hit policy, multiple rules with different output entries can match. The decision table returns outputs of all matching rules in an arbitrarily-ordered list.

Here is a decision table created with the Collect hit policy:

| C | Age | Interest Rates |
|---|-----|----------------------|
| | - | Enter Allowed Values |
| 1 | >18 | 10 |
| 2 | >60 | 8 |
| 3 | <18 | 5 |

In this example, two rules match for an input age of 61. The decision table returns output values of these rules in a list:

| Results | |
|---------|----|
| | 10 |
| | 8 |

- **Multiple Hit Collect (Sum)**

In a decision table with Collect (Sum) hit policy, multiple rules with different output entries can match. The decision table returns the sum of outputs of all matching rules.

Here is a decision table created with the Collect (Sum) hit policy:

The screenshot shows a decision table titled "Discount Percent" with a "C+" classification. The table has two columns: "Age" and "Discount Percent". The "Discount Percent" column header includes a sub-header "Enter Allowed Values". There are four rules defined in the table:

| C+ | Age | Discount Percent |
|----|----------|----------------------|
| | - | Enter Allowed Values |
| 1 | <18 | 15 |
| 2 | [18..45] | 5 |
| 3 | >45 | 10 |
| 4 | >60 | 15 |

In this example, the last two rules match for an input age of 61. The decision table returns the sum of output values of these rules, that is, 25.

- **Multiple Hit Collect (Min)**

In a decision table with Collect (Min) hit policy, multiple rules with different output entries can match. The decision table returns the smallest output value among all matching rules.

Here is a decision table created with the Collect (Min) hit policy:

Discount Percent

Add description

Q&A

Decision Table

| C< | Age | Discount Percent |
|----|----------|----------------------|
| | - | Enter Allowed Values |
| 1 | <18 | 15 |
| 2 | [18..45] | 5 |
| 3 | >45 | 10 |
| 4 | >60 | 15 |

In this example, the last two rules match for an input age of 61. The decision table returns the smallest output value among these rules, that is, 10.

- **Multiple Hit Collect (Max)**

In a decision table with Collect (Max) hit policy, multiple rules with different output entries can match. The decision table returns the largest output value among all matching rules.

Here is a decision table created with the Collect (Max) hit policy:

Discount Percent

Add description

Q&A

Decision Table

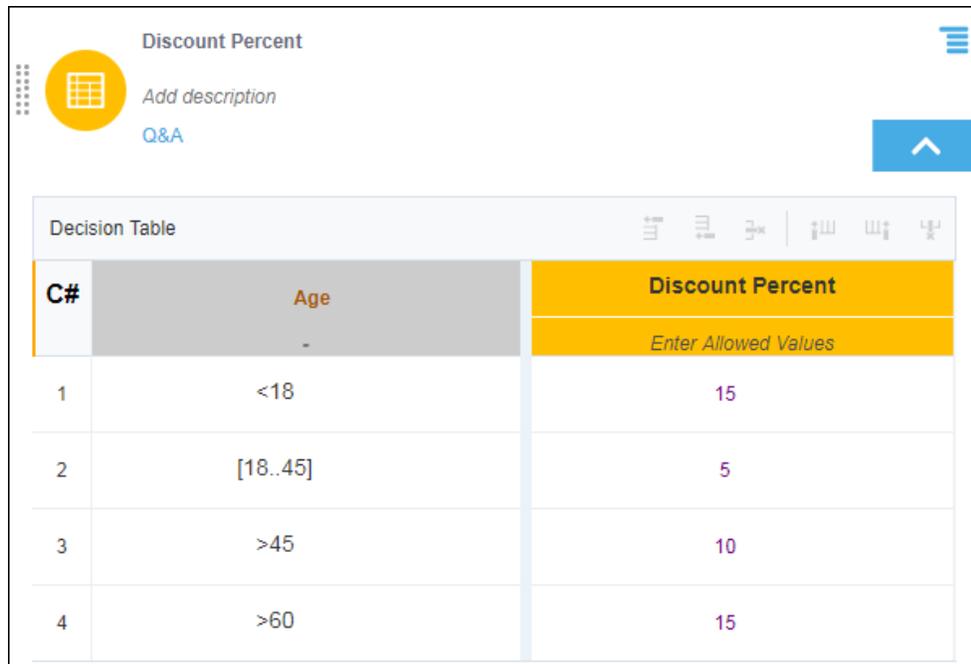
| C> | Age | Discount Percent |
|----|----------|----------------------|
| | - | Enter Allowed Values |
| 1 | <18 | 15 |
| 2 | [18..45] | 5 |
| 3 | >45 | 10 |
| 4 | >60 | 15 |

In this example, the last two rules match for an input age of 61. The decision table returns the largest output value among these rules, that is, 15.

- **Multiple Hit Collect (Count)**

In a decision table with Collect (Count) hit policy, multiple rules with different output entries can match. The decision table returns the count of matching rules.

Here is a decision table created with the Collect (Count) hit policy:



| C# | Age | Discount Percent |
|----|----------|------------------|
| 1 | <18 | 15 |
| 2 | [18..45] | 5 |
| 3 | >45 | 10 |
| 4 | >60 | 15 |

In this example, the last two rules match for an input age of 61. The decision table returns the count of matching rules, that is, 2.

Create Expressions

An expression is a logical notation, defined according to the syntax of FEEL, that evaluates to a single value. It may consist of one or more entities, such as a literal, constant, or variable, interconnected by zero or more operators. In Process, you can also use outputs of other decisions or built-in functions to define an expression.

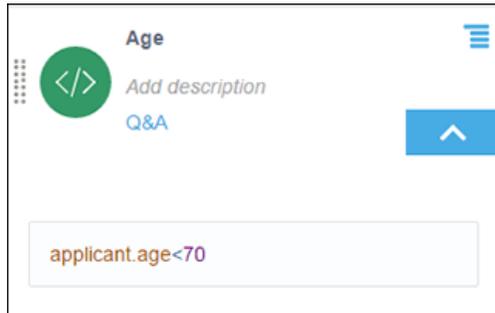
To create a decision with the Expression notation:

1. Add a new decision element to the model.
 - a. In graph view, click **Expand**  next to **Decision** on the diagram palette. Select **Expression** and drag it onto the canvas. Double-click the element to edit its logic. You can also edit the expression's logic from the properties pane; select the element, then select **Open Properties** , and in the properties pane click **Edit**  next to the **Name** field.
Click **Close**  to return to the canvas.
 - b. In list view, click **Add new decision**  in the Decisions bar and select **Expression** from the Create Decision window. See [Adding and Ordering Decisions](#).
2. In the Expression field, press **Ctrl+Space** to view an auto-suggest menu. You can use any decision outputs, variables, functions, and keywords listed in the menu to form your expression.

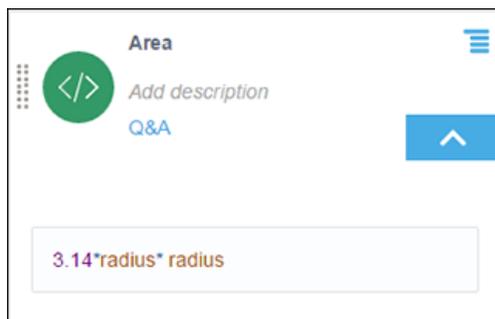
- After the decision logic is complete, click **Save** to save and validate changes manually. Changes you make within the decision model are also automatically saved and validated from time to time. Errors, if any, are displayed within the decision.

Following are some examples of decisions created using simple expressions:

- If the *age* property of the variable *applicant* is less than 70, then the decision returns *true* else it returns *false*.



- The decision calculates the area using a constant and an input variable, *radius*.



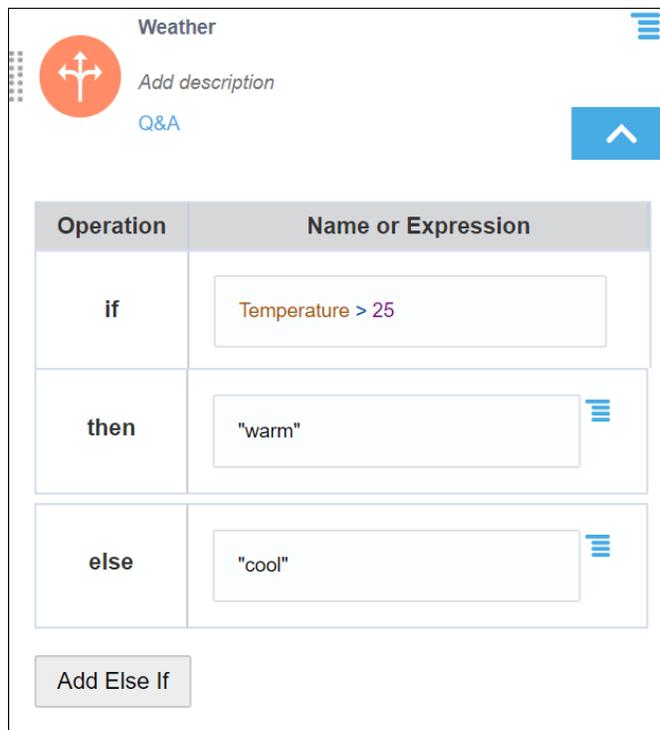
Create If-Then-Else Statements

An If-Then-Else expression is a logical notation that evaluates a test statement. It executes a primary expression if the test is *true* and a secondary expression if the test is not *true*. You can also introduce additional test statements using the **Add Else If** button.

To create a decision with the If-Then-Else logic:

- Add a new decision element to the model.
 - In graph view, click **Expand**  next to **Decision** on the diagram palette. Select **If-Then-Else** and drag it onto the canvas. Double-click the element to edit its logic. You can also edit the element's logic from the properties pane; select the element, then select **Open Properties** , and in the properties pane click **Edit**  next to the **Name** field. Click **Close**  to return to the canvas.
 - In list view, click **Add new decision**  in the Decisions bar and select **If-Then-Else** from the Create Decision window. See [Adding and Ordering Decisions](#).

- In the **if** expression field, press **Ctrl+Space** to view an auto-suggest menu. You can use any decision outputs, variables, functions, and keywords listed in the menu to define expressions in **if**, **then**, and **else** fields. Use the FEEL syntax to define expressions. See [Understanding FEEL](#).
 - If necessary, change the logical notations for **then** and **else** fields to create a nested logic. All fields have the expression notation selected by default. Click **Change body** and select a different notation from the available options for a particular field. Configure logic for the selected notation.
 - Additionally, you can cut or copy a notation from one field and paste it into another field using options available in the **Change body** menu.
 - After the decision logic is complete, click **Save** to save and validate changes manually. Changes you make within the decision model are also automatically saved and validated from time to time. Errors, if any, are displayed within the decision.
- In the following example, the input value of *temperature* determines the output of the If-Then-Else decision:



The screenshot shows a user interface for a decision logic model titled "Weather". At the top left, there is a red circular icon with a white crosshair and the text "Add description" and "Q&A". Below this is a table with two columns: "Operation" and "Name or Expression". The table contains three rows: "if" with the expression "Temperature > 25", "then" with the output "warm", and "else" with the output "cool". Each output field has a "Change body" menu icon (three horizontal lines) to its right. At the bottom of the table, there is a button labeled "Add Else If".

| Operation | Name or Expression |
|-----------|--------------------|
| if | Temperature > 25 |
| then | "warm" |
| else | "cool" |

Add Else If

- The following example uses an additional test statement, *precipitation > 50*, through Else If to determine the final output:

| Operation | Name or Expression | |
|-----------|--------------------|--|
| if | Temperature > 25 | |
| then | "warm" | |
| else if | Precipitation > 50 | |
| then | "rain" | |
| else | "cool, dry" | |

Add Else If

Create Functions

You can create functions to define specific operations that aren't available through built-in functions. In Process, decisions created using the Function notation return a value only when invoked from another decision.

To successfully invoke a function from another decision, the number and type of parameters in the function invocation must match those in the function definition.

The following example demonstrates a function implementation in Process. Here, the output decision invokes a function decision to calculate the seasonal discount percentage. The function decision contains the logic for regular discounts in the form of a decision table.

The screenshot shows the Oracle Decision Logic editor interface. It features two function elements:

- Special Discount:** A green icon with code symbols. Description: "Special seasonal discount." Q&A link. Body field contains the expression: `Discount(count,price)*2`.
- Discount:** A blue icon with "fn". Description: "Regular discount." Q&A link. It has a parameters table and a decision table body.

Parameters Table:

| Parameter | Value | Type | Action |
|-----------|-------|--------|--------|
| c | | number | x |
| p | | number | x |

Body Table (Decision Table):

| A | c | p | Discount |
|---|------|-------|----------------------|
| | - | - | Enter Allowed Values |
| 1 | - | >100 | 10 |
| 2 | >=50 | - | 10 |
| 3 | <50 | <=100 | 5 |

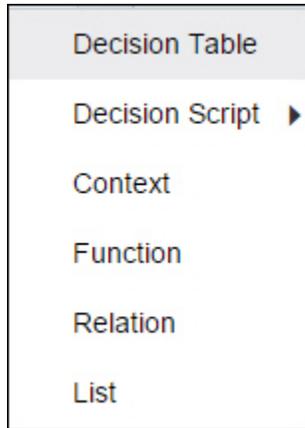
To create a decision with the Function notation:

1. Add a new decision element to the model.
 - a. In graph view, click **Expand** next to **Decision** on the diagram palette. Select **Function** and drag it onto the canvas. Double-click the element to edit its logic. You can also edit the function's logic from the properties pane; select the element, then select **Open Properties** , and in the properties pane click **Edit** next to the **Name** field.

Click **Close** to return to the canvas.
 - b. In list view, click **Add new decision** in the Decisions bar and select **Function** from the Create Decision window.

A function with empty Parameters and Body fields is created, with the Expression notation selected by default for the Body field.

- In the Parameters field, click **Add**  to add a new parameter. Enter a name for the parameter and select a data type for it. See [Data Types](#).
- In the Body field, click **Change body**  to change the logical notation. Select the required notation from available options.



- Now, configure the logic for the selected notation. You can use input variables or built-in functions to define the logic.
- Click **Save** to save and validate changes manually. Changes you make within the decision model are also automatically saved and validated from time to time. Errors, if any, are displayed within the decision.

 **Note:**

Because a function decision by itself doesn't return a result, it's not an output decision. Therefore, you can't add function decisions to a decision service.

Create Contexts

A **context** is a collection of one or more key-value pairs with an optional result field. Each pair is called a context entry. The key attribute within a context entry acts as an identifier to its corresponding value attribute.

You can use a context to collectively document all decision logic related to a particular scenario or entity. Say you need to determine the loan eligibility of an applicant, based on the applicant's net monthly income and expense. For this purpose, you can create a decision named *Loan Eligibility* using the Context notation and add expressions or logic for gross monthly income, monthly expense, and net monthly income. Then, you can either add a result field (within the context) that evaluates the net income and expense for loan eligibility, or you can choose to evaluate these within another decision.

Without a result field, a context decision returns multiple key-value pairs as output. In this case, you can invoke any context entry from another decision.

If you add a result field, the output of this field is displayed as the context's output. Here, you can only invoke the context's result from another decision.

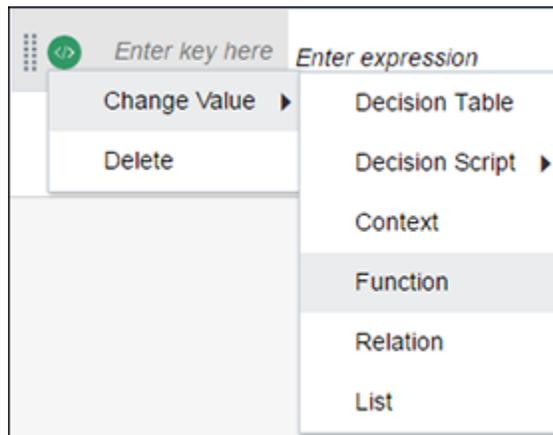
To create a decision with the Context notation:

1. Add a new decision element to the model.
 - a. In graph view, click **Expand**  next to **Decision** on the diagram palette. Select **Context** and drag it onto the canvas. Double-click the element to edit its logic. You can also edit the context's logic from the properties pane; select the element, then select **Open Properties** , and in the properties pane click **Edit**  next to the **Name** field.

Click **Close**  to return to the canvas.
 - b. In list view, click **Add new decision**  in the Decisions bar and select **Context** from the Create Decision window.

An empty Context is created.

2. Click **Add entry** to create a new context entry. A key-value pair is created with the expression notation selected by default in the Value field.
3. To change the logical notation for an entry, click the decision logic icon in the Key field to open the Change Value menu. Select a different notation from the available options.



4. In the Key field of a context entry, enter a unique name.
5. In the corresponding Value field, configure the logic for the selected notation. You can use input variables or built-in functions to define the logic.
6. Repeat steps 2 to 5 to add another entry into the context.
7. Drag and drop context entries to reorder them within the context.
8. Optionally, click **Add result** to include a result field for the context. A key-value pair appears with the expression notation in the Value field and the context name in the Key field.
9. Similar to other entries, you can change the notation for the result field and define a logic using other context entries, input variables, or built-in functions.
10. Click **Save** to save and validate changes manually. Changes you make within the decision model are also automatically saved and validated from time to time. Errors, if any, are displayed within the decision.

To delete a context entry, click the decision logic icon in the Key field and select **Delete**.

The following image shows a context with a result field that determines the loan eligibility of applicants::

Loan Eligibility
Determines loan eligibility
Q&A

| | |
|------------------|-------------------------|
| Gross Income | 10000 |
| Expenses | 5000 |
| Net Income | Gross Income - Expenses |
| Loan Eligibility | Net Income > Expenses |

Add entry

The output of the result field is the context's output. In this case, the context returns a *true* or *false* regarding an applicant's loan eligibility. You can reference the context's result in other decisions within the model using the context name (for example, *Loan Eligibility*).

The following image shows a context without a result field and an expression decision referencing multiple context entries to determine the loan eligibility of applicants:

Loan Eligibility
Determines Loan Eligibility
Q&A

Income.Net > Income.Expenses

Income
Calculates Net Income
Q&A

| | |
|--------------|-------------------------|
| Gross Income | 10000 |
| Expenses | 5000 |
| Net | Gross Income - Expenses |

Add entry Add result

The output of this context is a list containing results of all three context entries. To reference a particular context entry from another decision, use the format

ContextName.EntryKey (for example, *Income.Expenses*). Within a context, an entry can only reference entries that are above it.

 **Note:**

If you add a function as one of the context entries, the context as a whole doesn't return a result. However, you will still be able to invoke results of individual context entries throughout the decision model.

Create Lists

A **list** notation is a vertical list of elements, where each element is an independent logical notation. The output of a list notation contains outputs of all its elements. You can also invoke the output of a particular list element from another decision.

To create a decision with the list notation:

1. Add a new decision element to the model.
 - a. In graph view, click **Expand**  next to **Decision** on the diagram palette. Select **List** and drag it onto the canvas. Double-click the element to edit its logic. You can also edit the list's logic from the properties pane; select the element, then select **Open Properties** , and in the properties pane click **Edit**  next to the **Name** field.

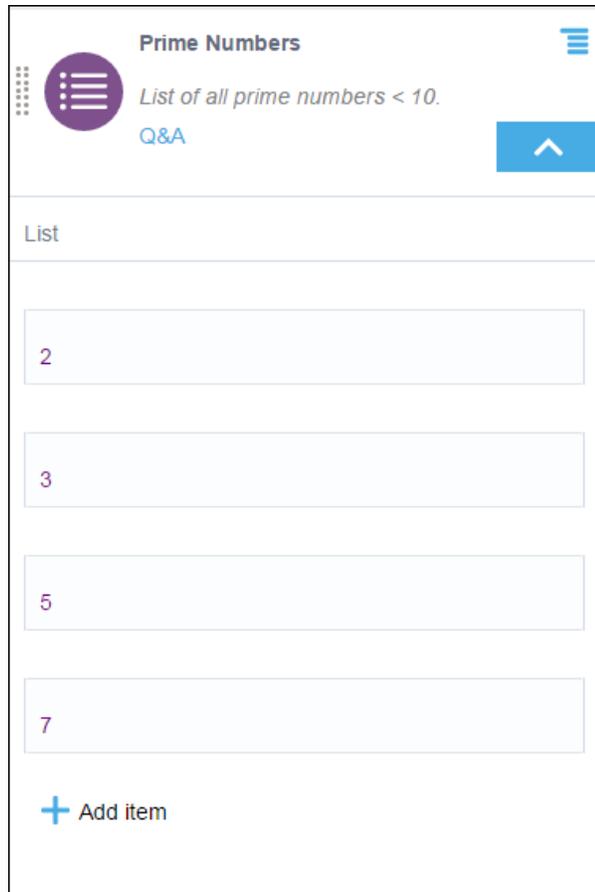
Click **Close**  to return to the canvas.
 - b. In list view, click **Add new decision**  in the Decisions bar and select **List** from the Create Decision window.

An empty list is created.

2. Click **Add**  to create a new list entry. An entry is created with the expression notation selected by default.
3. To change the logical notation for an entry, click **Edit**  for the particular entry. Select a different notation from the available options.
4. In the entry field, configure the logic for the selected notation. You can use input variables or built-in functions to define the logic.
5. Repeat steps 2 to 4 to add another entry into the list.
6. Click **Save** to save and validate changes manually. Changes you make within the decision model are also automatically saved and validated from time to time. Errors, if any, are displayed within the decision.

To delete a list entry, click **Edit** for the particular entry and select **Delete**.

The following example is a list of simple expressions containing prime numbers that are less than 10:



In a list of n elements, use $List_name[n]$ to invoke the n^{th} element from the beginning of the list, and use $List_name[-n]$ to invoke the n^{th} element from the end of the list. In this example, to invoke the list entry of 2, you can either use $Prime\ Numbers[1]$ or $Prime\ Numbers[-4]$.

You can also use suitable built-in list functions on a decision containing a List notation. For example, the following Expression decision returns the sum of all items in the *Prime Numbers* decision.



 **Note:**

- If you add a function as one of the list entries, then the list notation as a whole doesn't return a result. However, you will still be able to invoke results of other list entries throughout the decision model.
- According to the FEEL syntax, you can also define horizontal lists in expression fields across all notations. For example, a list of all prime numbers less than 10 can be defined as [2,3,5,7].

Create Relations

You can use a relation notation as convenient shorthand to represent multiple contexts.

A **relation** is a vertical list of similar contexts arranged horizontally. In other words, each row of a relation table is a context and each column consists of context entries, where the column name is the common key attribute for all cell entries under it that act as value attributes of respective contexts (rows). For details about contexts and key-value pairs, see [Using Contexts](#). In a relation, each cell entry is an independent logical notation.

In the output of a relation notation, outputs of all contexts within it are clearly distinguished. You can also invoke the output of a particular context or context entry from another decision.

To create a decision with the Relation notation:

1. Add a new decision element to the model.
 - a. In graph view, click **Expand**  next to **Decision** on the diagram palette. Select **Relation** and drag it onto the canvas. Double-click the element to edit its logic. You can also edit the relation's logic from the properties pane; select the element, then select **Open Properties** , and in the properties pane click **Edit**  next to the **Name** field.

Click **Close**  to return to the canvas.

- b. In list view, click **Add new decision**  in the Decisions bar and select **Relation** from the Create Decision window.

An empty Relation table is created.

2. Use row/column controls available within the decision to add additional rows or columns and use **Expand Table**  to view and edit the relation on an expanded dialog. All cells have expression notation selected by default.
3. Enter a name for each column.
4. Select a cell and click **Change logic**  to change the logical notation for the cell. Select a different notation from the available options. Note that you cannot insert a decision table within a relation.
5. Within cells, configure the logic for selected notations. You can use input variables or built-in functions to define the logic.
6. Click **Save** to save and validate changes manually. Changes you make within the decision model are also automatically saved and validated from time to time. Errors, if any, are displayed within the decision.

The following relation contains the stock information for a particular brand of phone in the form of multiple contexts:

| Color | Count | Price |
|-------------|-------|-------|
| "Black" | 2 | 500 |
| "White" | 3 | 700 |
| "Rose Gold" | 5 | 600 |

Similar to list notations, use either *Phone[1]* or *Phone[-3]* to access the entire context related to black-colored phones. To access all cell entries of a particular column, use the relation name in combination with the column name, for example, *Phone.Price* returns all entries of the Price column in a list. To access a particular context entry (for example, "Rose Gold"), use *Phone.Color[3]* or *Phone.Color[-1]*.

The following image is the result of the entire relation, which has results of each context listed separately:

| Results | |
|---------|-----------|
| Color | Black |
| Price | 500 |
| Count | 2 |
| Color | White |
| Price | 700 |
| Count | 3 |
| Color | Rose Gold |
| Price | 600 |
| Count | 5 |

Note:

If you add a function as one of the cell entries, then the relation as a whole doesn't return a result. However, you will still be able to invoke results of other cell entries or contexts throughout the decision model.

Create Loops

Create loops to iterate over lists or arrays. Using the **loop** logical notation, you can create three different types of loops, namely For, Some, and Every.

- **For**: Iterates over a list and returns an expression.
- **Some**: Checks if at least one list item satisfies the test condition defined by an expression and returns a Boolean value.
- **Every**: Checks if every list item satisfies the test condition defined by an expression and returns a Boolean value.

To create a decision with the Loop logic:

1. Add a new decision element to the model.
 - a. In graph view, click **Expand**  next to **Decision** on the diagram palette. Select **Loop** and drag it onto the canvas. Double-click the element to edit its logic. You can also edit the loop's logic from the properties pane; select the element, then select **Open Properties** , and in the properties pane click **Edit**  next to the **Name** field. Click **Close**  to return to the canvas.
 - b. In list view, click **Add new decision**  in the Decisions bar and select **Loop** from the Create Decision window.

An empty loop decision is created.

2. In the **Operation** column, choose the type of loop to create from the drop-down menu. In the corresponding expression field, enter the loop variable.
 3. In the **in** field, enter the list or array over which to iterate. In the **return** or **satisfies** fields, enter the expression to return (For loop) or the test expression (Some and Every loops), respectively. Press **Ctrl+Space** to view an auto-suggest menu. You can use any decision outputs, variables, functions, and keywords listed in the menu to define expressions in the **in**, **return**, and **satisfies** fields. Use the FEEL syntax to define expressions.
 4. If necessary, change the logical notations for **return** and **satisfies** fields to create a nested logic. These fields have the expression notation selected by default. Click **Change body**  and select a different notation from the available options. Configure logic for the selected notation. From the **Change body** menu, you can also cut or copy a notation from these fields and paste it into another identical field within the model.
 5. Furthermore, for a For loop, you can add an additional condition, **where**, using the **Add Condition** button.
 6. After the decision logic is complete, click **Save** to save and validate changes manually. Changes you make within the decision model are also automatically saved and validated from time to time. Errors, if any, are displayed within the decision.
- The following example shows a simple For loop that returns the squares of list items:

Square
Add description
Q&A

| Operation | Name or Expression |
|-----------|--------------------|
| For | n |
| in | [1,2,3,4] |
| return | n*n |

Add Condition

- The following decision contains a Some loop that checks if at least one element in the list is greater than 50:

Greater Than
Add description
Q&A

| Operation | Name or Expression |
|-----------|--------------------|
| Some | n |
| in | [12,50,51] |
| satisfies | n > 50 |

- The following decision contains an Every loop that checks if every element in the list is greater than 50:

| Operation | Name or Expression |
|-----------|--------------------|
| Every | n |
| in | [12,50,51] |
| satisfies | n > 50 |

Test Decisions

After creating the decisions and supporting decisions within your decision model, you can verify that your decision model works the way you want by testing your model.

To test the decision model:

1. Click **Test** ►.
2. In the Test Decision Model pane, enter the input data to test your decision and click **Start Test**. The Decision Model Result pane displays decision results.
3. Click each green check mark to see the decision's result.
4. Click **Go Back**, and repeat steps 1 to 3 to test each decision rule's outcome.

Expose Decisions as Services

To use your decision models in one or more Process applications, you must add at least one decision service in the decision model before you deploy the decision model. The decision service will expose one or more output decisions of your decision model as public REST APIs. A decision service consists of a set of input data and a set of output decisions from the containing decision model. You can use the decision services in your process to implement a decision model.

To configure a decision service:

1. In the decision model editor, expand the Services pane.
2. Click **Add**  to open the Add New Decision Service dialog box.
3. Enter a name for the decision service and click **OK**.

The new service appears in the pane with two fields, namely **Output Decisions** and **Input Data**.

4. Click the **Output Decisions** field to select the output decision you want to expose through the service.
5. Similarly, click the **Input Data** field to select the input data to expose through the service.

You can add multiple values for the input data and the output decision.

To edit or delete an existing decision service, click **Options**  next to the service name. You can also view or copy the URL, request payload, and response payload of the decision service.

Manage Decision Model Snapshots

Decision model snapshots are read-only copies of a decision model at a particular moment.

You can:

- Create a snapshot at any point while creating a decision model
- View the contents of a snapshot
- Delete a snapshot
- Export a snapshot to your local file system

Participants with space owner or space editor permissions can create a snapshot.

You can create and deploy a snapshot in any one of the following ways:

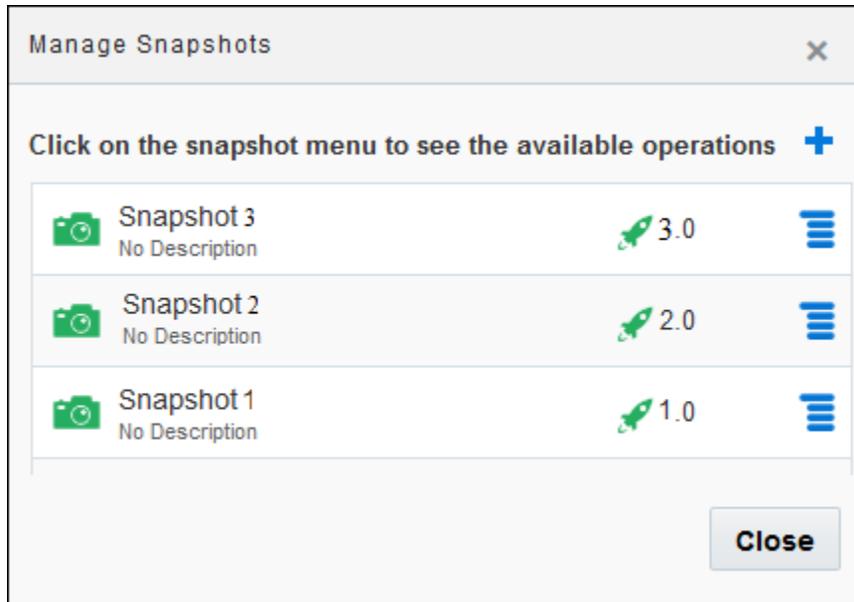
- From the Decision Model editor
- From the Manage Snapshots dialog

To create and activate a snapshot from the Decision Model editor:

1. While creating a decision model, click **Activate** to open the Activate Application and Create Snapshot dialog.
2. Enter a name and description for the snapshot.
3. Assign a runtime version ID with which the snapshot will be activated. A runtime version ID uniquely identifies an activated snapshot among other deployed snapshots of the same model.
4. Optionally, select **Overwrite** to reuse an existing runtime version ID.
5. Click **Activate** to create and deploy the snapshot.

To create a snapshot from the manage Snapshots dialog:

1. Click **Snapshots** to open the Manage Snapshots dialog.



2. Click **Add**  to open the Create Snapshot dialog.
3. Enter a name and description for the snapshot and click **Create**.

To manage the existing snapshots:

1. Click **Snapshots** to open the Manage Snapshots dialog box.
2. Right-click an existing snapshot. The available options are:

| Option | Description |
|-------------------------|---|
| Activate/ Deactivate | Activate or deactivate the snapshot. |
| View | View a read-only version of the decision model snapshot. |
| Delete | <p>Delete the selected snapshot. You need to deactivate a snapshot that is already activated before deleting it.</p> <p>Warning: If you delete a snapshot that is currently in use in one or more applications, the reference to the associated decision model does not get deleted and remains in the process application. This may result in errors during runtime when the application tries to call the decision service associated with a decision model snapshot that has previously been deleted from the DMN server.</p> |
| Export | <p>Export any snapshot of the decision model, activated or not, to your local file system.</p> <p>Note: You can import a previously exported snapshot as a new decision model and edit its content.</p> |

Add Decisions to Applications and Processes

After creating and activating a decision snapshot, you can add it to your application for use within processes.

To add a decision connector to an application:

1. In the **Application Home** pane, click **Decisions**.

2. Click **Link to a Decision model**.

Optionally, click **Create** and select **Use Decision Model**.

3. In the resulting dialog box, select an activated decision snapshot, enter a name for the connector, and click **Use**.

A decision connector is now created. You can use this connector in all processes within the application after performing the required data associations.

4. Optionally, click the connector to view a read-only version of the decision snapshot from within the application.

5. Click **Edit** to open the latest version of the decision model for editing.

Note that the latest version of the model may differ from the version you've used in your connector. To use the latest version after your edits, activate the model again and update your connector. See [Update Decision Connectors](#).

 **Note:**

- Within any space, you can create only one connector per decision model. If you want to use a different version of a particular decision in your application, click the **Update definition** link of the corresponding connector and select the required snapshot.
- If the decision snapshot used by a connector is deactivated, the connector displays an error indicating that the decision model is inactive. Click the **Reconnect to model** link on the connector to re-associate it with an active decision snapshot. The same error is also displayed when the decision model associated with the connector is deleted.

To use a decision connector in a process within an application:

1. On the Process Editor's elements palette, expand the System elements.
2. Drag and drop a Decision flow element into the process.
3. Implement the Decision element:
 - a. Click the element and select **Open Properties, General**.
 - b. Select a decision connector from the **Decision Model** list. All the connectors that you have created within the application are listed here.

 **Note:**

You can also use a decision snapshot for which you have not created a connector previously. Click **Add +** next to the **Decision Model** field to display a list of activated decision models. Select a decision from the list, enter a name for it, and click **Use**. When you add a decision model to a process in this way, a corresponding decision connector is automatically created in the **Decisions** tab of the Application Home page.

- c. Select a decision service to use from the list of services defined for the selected decision model. Process creates all the metadata definitions such as importing the required types and defining the connector to the service.
4. Click **Save**  to save your changes.

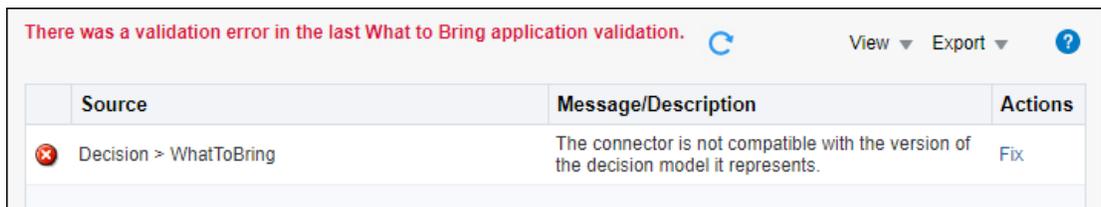
Update Decision Connectors

Update decision connectors to use a different snapshot or to resolve incompatibility errors.

A connector becomes incompatible when the data interface is updated for the version of the decision model used by it.

Say a connector uses version 2.0 of the What to Bring decision model. If you make any of the following interface updates for this version of the model, the connector becomes incompatible and the application reports a validation error:

- Add, edit, or delete input data and type definitions
- Add, edit, or delete decision services
- Add a new or delete the existing output decision
- Edit the output decision's type

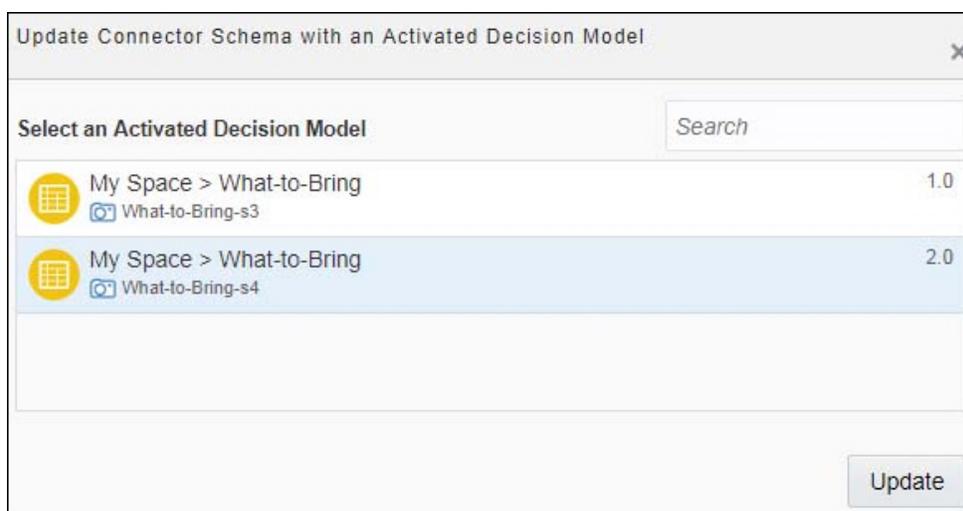


The image shows a validation error message: 'There was a validation error in the last What to Bring application validation.' with a refresh icon. Below the message is a table with the following data:

| Source | Message/Description | Actions |
|--|---|---------------------|
|  Decision > WhatToBring | The connector is not compatible with the version of the decision model it represents. | Fix |

To update a decision connector:

1. Click the **Update definition** link on the connector.
2. In the resulting dialog, select a different snapshot or the updated snapshot (for the same version), and click **Update**.



3. If the snapshot contains interface changes, you're prompted to fix the data association and other validation errors that may occur due to this update. Click **Update and Fix Later**. Notice that the connector is no longer displayed as incompatible.
4. Select **Validate Application** from the application's navigation menu () . The resulting pane displays all the validation errors that may have occurred due to the connector update. Click **Fix** against an error to navigate to the corresponding process and resolve the error.

 **Note:**

You can also update a decision connector from the validation pane.

5. Publish your changes and reactivate the application.

Promote Decision Models as Samples to the Gallery

If you want to make your decision models available to others, and also allow developers to adapt them for their own use, promote them as *Samples* to the gallery.

To promote a decision model as a sample to the gallery:

1. Open the model you want to add as a sample to the gallery.
2. In the decision model editor, click **Menu**  on the toolbar, and then select **Promote to Gallery**.
3. On the **Properties** train stop of the Promote to Gallery as a Decision Sample App wizard, define how the sample decision model displays in the gallery.
 - a. Enter a suitable name in the **Name** field.
 - b. Enter a meaningful description in the **Description** field.
 - c. Add a preview image that best illustrates the model.

4. Click **Next**.
5. On the **Promote** train stop, enter a name for the decision snapshot that'll be saved and promoted to the gallery. Optionally, in the **Snapshot Comment** field, enter notes relevant to the snapshot being saved.

6. View and check for warnings under **Validation Results**. Fix all major validation errors before promoting the model to the gallery.
7. Click **Promote**. You get a confirmation that a snapshot of your decision model has been saved and promoted to the gallery.

8. Click **Close**.

Your decision model will now be available in the gallery for users to select and create copies of it.

To delete the decision samples you promoted to the gallery, contact your design-time administrator. See [Delete Apps from the Gallery](#).

Best Practices for Modeling Decision Logic

This section provides some best practices and recommendations for creating decision models in Oracle Integration. It also illustrates these recommendations using a simple example.

Here's the list of best practices to make decision models easy to develop, maintain, and interpret.

- Use decision tables where possible; they're the preferred form of logic.
- In decision tables, avoid using the First (F) hit policy. This hit policy makes the decision logic overly reliant on the order of the rules.
- Use functions to apply a decision logic multiple times, for example, to apply a logic to each element of a list.
- Avoid using functions unnecessarily; they make decision models difficult to test and debug.
- Always break down a complex, nested expression into simpler expressions.
- Use boxed expressions instead of FEEL expressions, wherever possible, to improve readability.
- Use nouns or short noun phrases to name each decision and input data (for example, Student Days, Total Days, Person). Do not use verbs in names.
- Provide a description for each decision to indicate what it accomplishes, for example, *Calculates total days as sum of other days*. Make all descriptions consistent with each other.

Paid Time Off (PTO) Calculator Example

In accordance with the best practices listed previously, let's develop a decision model to calculate paid time off days for employees in an organization. (Credit for this PTO calculator example goes to Professor Jan Vanthienen, who posed the challenge on the decision modeling community page, [dmcommunity.org](#).)

This example uses the following conditions to arrive at the total paid time off (in days) for an employee:

1. All employees receive a certain number of base vacation days according to their region. Employees in a US office receive 18 base vacation days, employees in EU receive 21 days, and those in APAC/LATIN-AMERICA receive 17 days.
2. Employees younger than 18 or at least 60 years of age or employees with at least 30 years of service receive 5 additional vacation days.
3. Employees with at least 30 years of service and also of age 60 or more receive 3 extra vacation days over the possible additional days already granted.

4. If an employee has at least 15 but less than 30 years of service, 2 extra days are granted. Employees of age 45 or more also receive these 2 additional days.
5. Employees working the second shift (from 4pm - 12am) get 2 extra days, while employees working the third shift (from 12am-8am) get 4 extra days.
6. Employees working the weekend schedule get 1 extra day off if their schedule covers one weekend day. Employees with weekend schedule that covers both days of the weekend get 2 extra days off. These vacation days are in addition to the days already granted for their shifts.
7. A college student is eligible for 1 extra vacation day, and a veteran is eligible for 2 extra days.

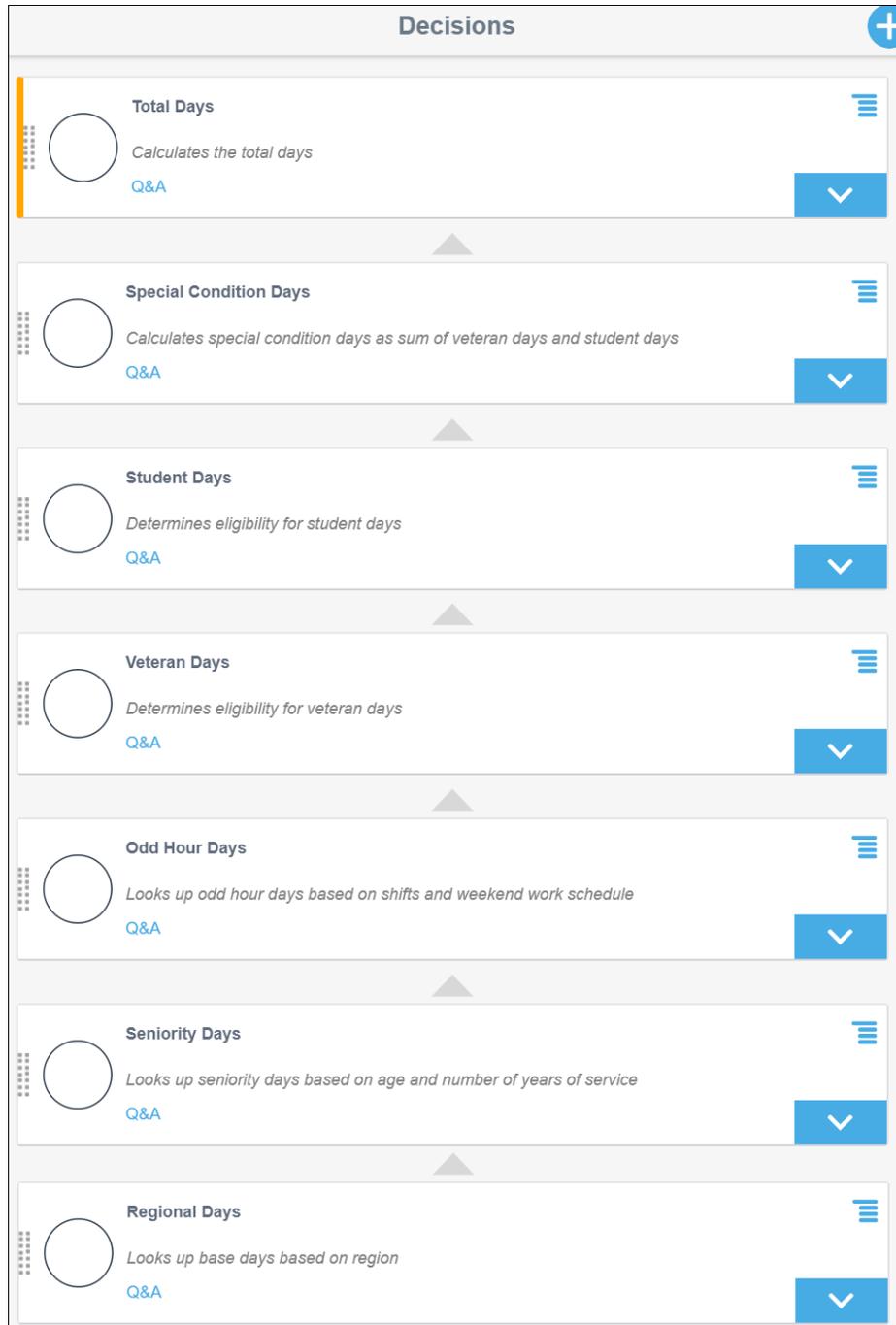
Based on the conditions defined, let's design the decision model in the following order:

- [Add Decisions](#)
- [Define Input](#)
- [Model Logic](#)
- [Test Decisions](#)
- [Create Decision Service](#)

Add Decisions

As a first step, determine the goal of your decision model. In this case, the goal is to calculate the total PTO days for an employee. Generally, the goal translates to the top or final output decision within the model. All other conditions become sub-decisions contributing, either directly or indirectly, towards the output of the top decision.

The following figure shows placeholder decisions added for this example. The decision names use nouns or noun phrases, and all decisions have a consistent description.

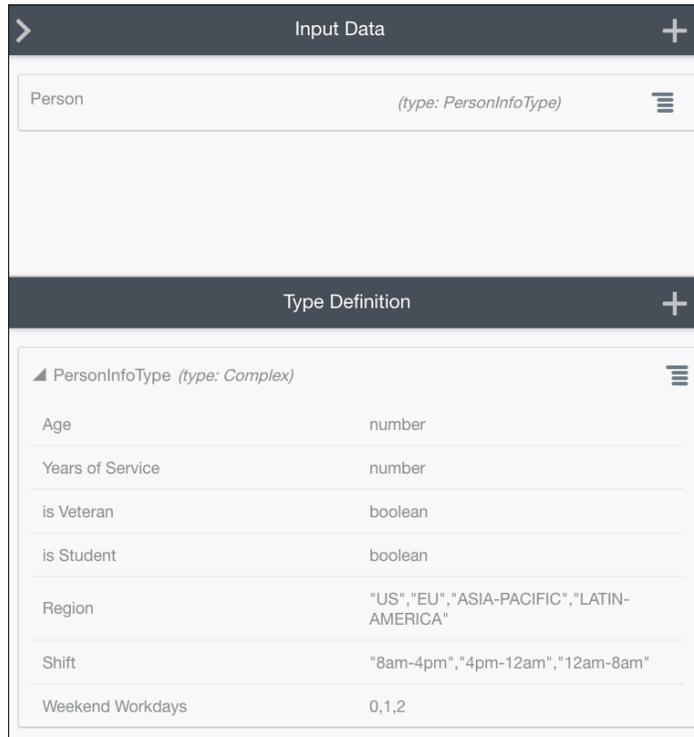


Define Input

Define input data variables to allow users to supply information for the decision model. You can either use built-in data types (such as string, number, Boolean, or datetime) to create input variables or define custom complex data types according to your requirements. The following figure shows a complex data type, *PersonInfoType*, defined that contains all the attributes necessary for this example. This custom data type is then used to create an input variable, *Person*.

As a best practice, define attributes as constraints whenever possible. Constraints are conditions imposed on the attributes of the input data. When defined correctly,

constraints help to check the integrity of input data. A constraint can be a range or list of valid values. In this example, the attributes Region, Shift, and Weekend Workdays are defined as a list of valid string and number values.



Model Logic

To model the decision logic for this example, we'll primarily use decision tables to help us efficiently capture all possible combinations of the conditions specified.

- [Condition 1](#)
- [Conditions 2, 3, and 4](#)
- [Conditions 5 and 6](#)
- [Condition 7](#)
- [Final Output Decision](#)

Condition 1

The following decision table, **Regional Days**, implements Condition 1 listed previously. Because the attribute Region is already defined as a constraint with a list of valid values, choose **Auto** in the Allowed Values cell of the column, *Person.Region*. The allowed values for this column are automatically derived from the constraint of the Region attribute of the input data type and populated into this cell.

| U | Person.Region | Regional Days |
|---|----------------------------------|----------------------|
| | US,EU,ASIA-PACIFIC,LATIN-AMERICA | Enter Allowed Values |
| 1 | US | 18 |
| 2 | EU | 21 |
| 3 | ASIA-PACIFIC,LATIN-AMERICA | 17 |

Conditions 2, 3, and 4

As a best practice, group related conditions into a decision where possible. Here, the decision **Seniority Days** implements Conditions 2, 3, and 4 of this example as all of these conditions serve to determine the seniority of an employee.

Choose the Unique (U) hit policy to ensure only one rule matches for an employee. This hit policy also helps you validate if the rules are created in a logically consistent manner, without conflicts, gaps, or redundancies. Additionally, you may add an annotation column to further clarify the implementation of each rule.

| U | Person.Age | Person.Years of Service | Seniority Days | Explanation |
|---|------------|-------------------------|----------------------|---|
| | - | - | Enter Allowed Values | |
| 1 | <18 | - | 5 | Under 18 gets 5 days |
| 2 | [18..45) | <15 | 0 | |
| 3 | [18..45) | [15..30) | 2 | 15 but less than 30 years of service gets 2 da |
| 4 | [18..45) | >=30 | 5 | 30 yrs of service or more gets 5 days |
| 5 | [45..60) | <30 | 2 | 40 but less than 60 gets 2 days |
| 6 | [45..60) | >=30 | 5 | 30 yrs of service or more gets 5 days |
| 7 | >=60 | <15 | 5 | 60 and over gets 5 days |
| 8 | >=60 | [15..30) | 5+2 | 60 and over + at least 15 yrs of services get 7 |
| 9 | >=60 | >=30 | 5+3 | 60 and over + at least 30 years of service get |

Conditions 5 and 6

The decision **Odd Hour Days** implements Conditions 5 and 6, which are related to work schedule of the employees.

Odd Hour Days
Looks up odd hour days based on shifts and weekend work schedule
Q&A

| U | Person.Shift | Person.Weekend Workdays | Odd Hour Days | Explanation |
|---|---------------------------|-------------------------|----------------------|--|
| | 8am-4pm,4pm-12am,12am-8am | 0,1,2 | Enter Allowed Values | |
| 1 | 8am-4pm | 0 | 0 | Day shift |
| 2 | 4pm-12am | 0 | 2 | Evening shift |
| 3 | 12am-8am | 0 | 4 | Overnight shift |
| 4 | 8am-4pm | 1 | 1 | Day shift + 1 work day on weekends |
| 5 | 4pm-12am | 1 | 2+1 | Evening shift + 1 work day on weekends |
| 6 | 12am-8am | 1 | 4+1 | Overnight shift + 1 work day on weekends |
| 7 | 8am-4pm | 2 | 2 | Day shift + 2 work day on weekends |
| 8 | 4pm-12am | 2 | 2+2 | Evening shift + 2 work day on weekends |
| 9 | 12am-8am | 2 | 4+2 | Overnight shift + 2 work day on weekends |

Condition 7

Two decisions, **Student Days** and **Veteran Days**, implement Condition 7 after breaking it down to simpler expressions. In addition, these decisions make use of boxed expressions (If-Then-Else) instead of a FEEL expression such as the following:

```
if Person.is Student then 1 else 0
```

Student Days:

Student Days
Determines eligibility for student days
Q&A

if

Person.is Student

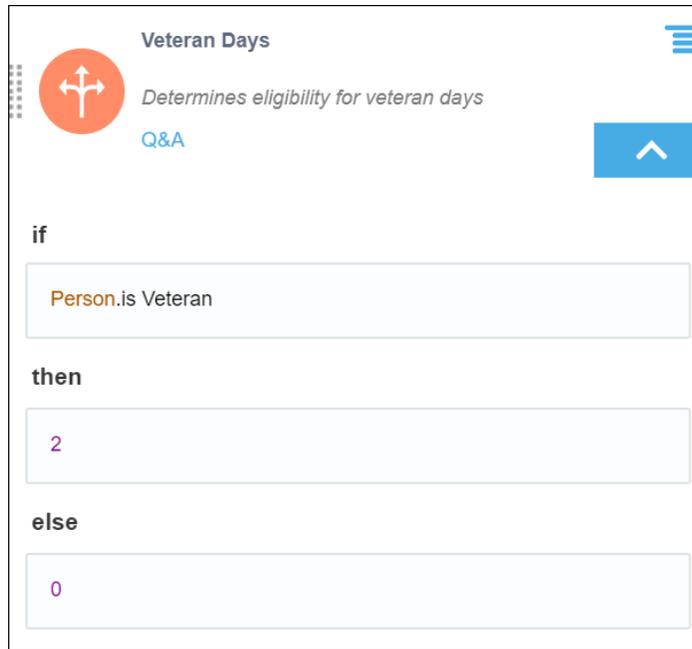
then

1

else

0

Veteran Days:

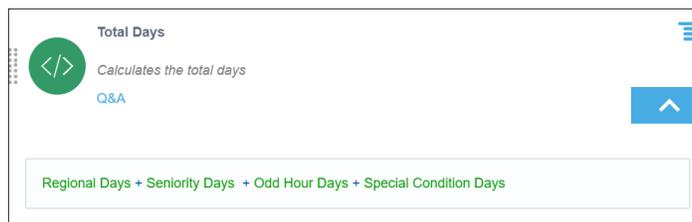


An additional decision totals these special condition days:



Final Output Decision

The topmost decision aggregates outputs of all sub-decisions and returns the final PTO days for an employee:



Test Decisions

After modeling the logic, test your decision model with different combinations of input data to ensure the model works as expected.

The following figures show a sample data set and its result for this example:

Decision Model Result
Start Test ×

Fill in the test data below

Person (Type: PersonInfoType)

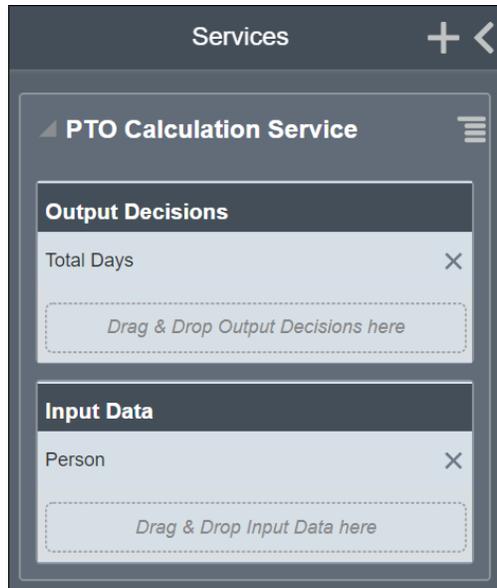
| | |
|------------------|----------------|
| Age | 62 |
| Years of Service | 30 |
| is Veteran | true |
| is Student | true |
| Region | "ASIA-PACIFIC" |
| Shift | "4pm-12am" |
| Weekend Workdays | 1 |

Go Back

| | | |
|------------------------|---|---------------------------------|
| Total Days | ✓ | Results Output is: 31 |
| Special Condition Days | ✓ | |
| Student Days | ✓ | |
| Veteran Days | ✓ | |
| Odd Hour Days | ✓ | |
| Seniority Days | ✓ | |
| Regional Days | ✓ | |

Create Decision Service

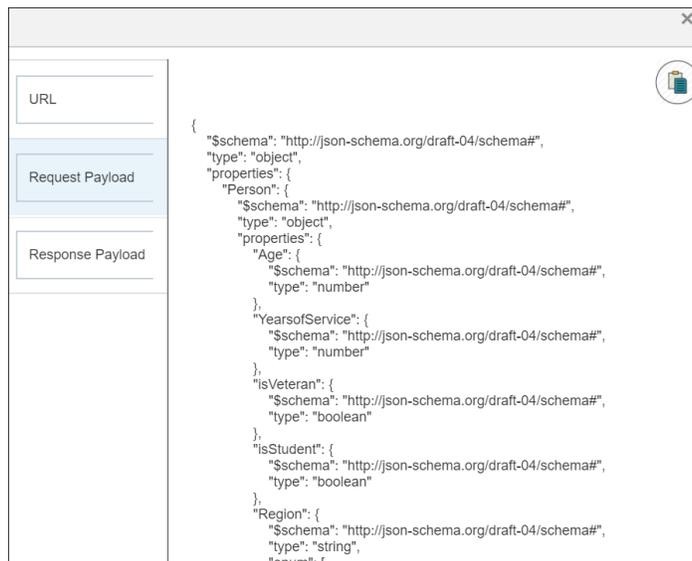
Finally, create a decision service in order to use the decision model in Oracle Integration components or external applications. Choose the required output decision and input data for the service. A decision service exposes the output decision as a public REST API.



After you create the decision service, you can examine the JSON schema of the request and response payloads of the service API.

The following images show the request and response payloads for this example.

Request payload:



Response payload:

| | |
|------------------|--|
| URL | |
| Request Payload | |
| Response Payload | |

```
{
  "$schema": "http://json-schema.org/draft-04/schema#",
  "type": "object",
  "properties": {
    "interpretation": {
      "$schema": "http://json-schema.org/draft-04/schema#",
      "type": "number"
    },
    "problems": {
      "$schema": "http://json-schema.org/draft-04/schema#",
      "type": "array",
      "items": {
        "type": "object",
        "properties": {
          "severity": {
            "$schema": "http://json-schema.org/draft-04/schema#",
            "type": "string",
            "enum": [
              "Error",
              "Warning"
            ]
          },
          "path": {
            "$schema": "http://json-schema.org/draft-04/schema#",
            "type": "array",
            "items": {
              "type": "string"
            }
          },
          "message": {
            "$schema": "http://json-schema.org/draft-04/schema#",
            "type": "string"
          }
        }
      }
    }
  }
}
```

To use this decision model in a process application, create a snapshot of the model, activate the same, and, finally, add this snapshot to the application. See [Activate a Decision Snapshot](#) and [Add Decisions to Applications and Processes](#).

12

Develop Smart Processes

Use smart processes to separate your process lifecycle from external intelligent sources. Externalizing intelligent sources offers several benefits: it simplifies the process model and makes it more stable, and lets other intelligence sources evolve independently and leverage new technologies and innovation.

Adaptive, smart processes move away from simple expressions and rule-based logic to a wider range of complex, frequently evolving intelligent sources. For example, smart processes can rely on intelligence consumed from sources like decision (DMN) models or machine learning models consumed using a REST interface.

Smart sentries are smart conditions that enable processes to react to different results of external intelligence sources. For example, a decision model might use credit score and debt-to-income ratio input to decide whether to present a low interest rate offer to a customer. Or, input values consumed via a REST API might determine whether a manual approval is needed.

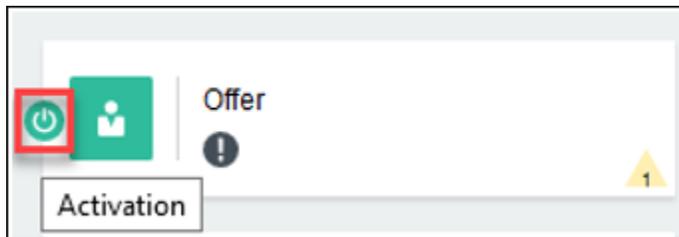
Topics:

- [Smart Sentries in Dynamic Processes](#)
- [Configure a Decision Sentry in a Dynamic Process](#)
- [Configure a REST Sentry in a Dynamic Process](#)

Smart Sentries in Dynamic Processes

In dynamic processes, define smart sentries as conditions that decide if an activity, stage, or process gets activated or terminated.

Conditions are integral to most dynamic processes. Their outcome decides if a specific plan item (process, stage, activity, or milestone) becomes activated or terminated. You get an indication on the canvas that a condition (Activation or Termination) has been added using a smart sentry.



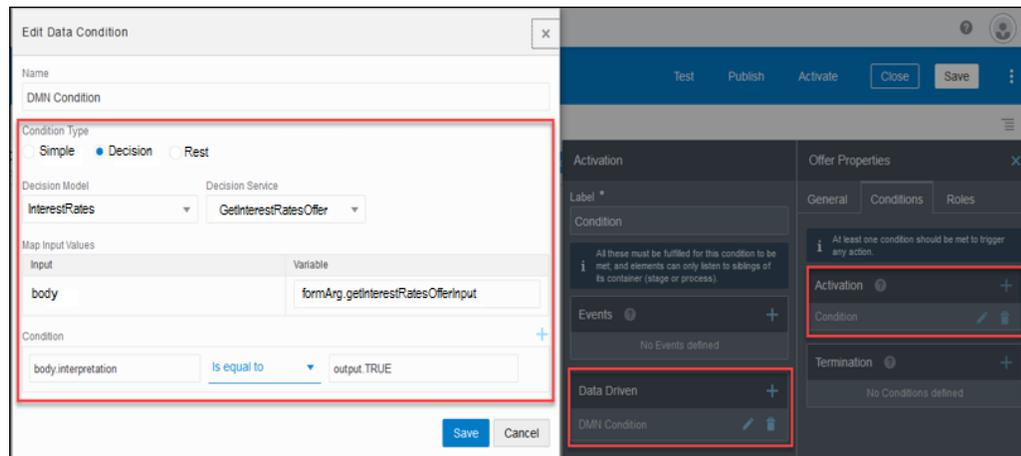
The different types of sentries that you can use in dynamic processes are:

- Simple
- Decision
See [Configure a Decision Sentry in a Dynamic Process](#)
- REST

Use smart sentries for complex and changing logic

When a condition requires complex and frequently changing logic, you can use a decision or REST connector to act as a smart sentry and control access to the plan item. If the response of the decision service is TRUE, activate the stage, activity, or process.

For example, your dynamic process involves complex logic centered on offers based on a credit score. Depending on form input, the decision service decides the offer and whether to present the offer activity to the user. Instead of creating rules within the process, the decision handles the complex logic outside the process framework, where it can be easily updated as needed.



Key points about smart sentries in dynamic processes

- In a dynamic process, you can apply a smart sentry wherever conditions are available. They are most commonly used to drive process flow when applied to a process, stage, or activity. But you can also apply them to milestones and markers.
- You can apply a smart sentry to an activation or termination condition.
- If applying a decision as a sentry, the decision must be activated before you can select it when configuring a condition in the dynamic process.
- When configuring the condition, you can apply multiple conditions, and specify AND or OR condition between them. You can also combine the condition with an event (for example, activate a stage if an event occurs and a condition is true).

Configure a Decision Sentry in a Dynamic Process

Use a decision model to activate or terminate a stage, activity, or process in a dynamic process.

Here are the main steps for using a decision model to activate or terminate a stage, activity, or process in a dynamic process.

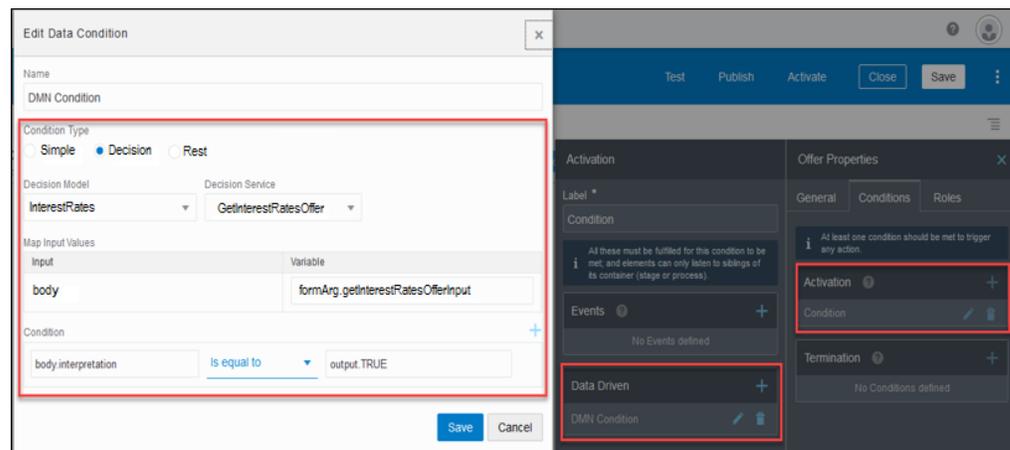
1. Create the process applications and its components.
 - a. Create a *dynamic process* with activities to support the decision. For example, the first activity may include a form that provides input to the decision. The second activity may activate based on the decision's result.

- b. Create or import a *decision model*, and activate it. Ensure that the decision is visible and active on the Decisions page of the process application.
 - c. Create a form to capture user input. Set the dynamic process to start with the form. Most likely, you'll use the values entered in the form as input values for the decision. (Alternately, you can start the dynamic process with data instead, such as an API.)
2. Configure data and its flow in the process.

Data values drive the decision as well as the dynamic process. Typically, business objects are shared among the form, decision, and dynamic process elements. You configure data association so that data flows in and out of the decision and other activities of the dynamic process.

For example, you may create a business type that is used in the form and as input data type in the decision.

3. Define the decision condition.
 - a. Select the activity, stage, or process and edit properties. Choose an activation or termination condition and set a data driven condition.
 - b. In the resulting dialog box, select **Decisions** as the condition type.
 - c. Select your activated decision model and service.
 - d. Map the input values of the decision and the form.
 - e. Define the decision's output as a condition. For example, if the decision's output evaluates to TRUE, activate or terminate the activity.



4. Try out the application in runtime with different form entries to test the decision condition.

Example of a Decision in a Dynamic Process

Explore with an example how you can use a decision in a dynamic process.

Suppose you're a customer service manager of a credit card company. You decide the credit card interest rates for potential customers. However, interest rates on credit cards aren't arbitrarily set. Typically, banks set interest rates based on the risk a customer poses. A lower credit risk qualifies a customer for lower interest rate. A customer's credit score and debt-to-income ratio are deciding factors. A high credit score and low debt-to-income ratio indicate that the customer has handled credit well in the past and is likely to pay new credit on time.

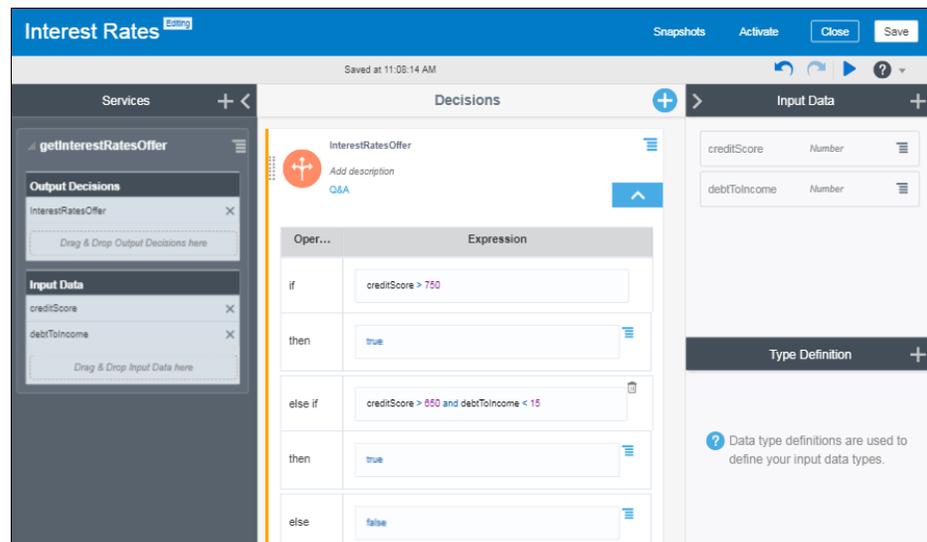
You need a decision model that *decides* based on the two deciding factors – credit score and debt-to-income ratio - if a potential customer can get a low interest rate offer.

The main artifacts that you create for implementing the example are:

- A dynamic process *CreditCard* that houses the various activities that the user has to complete while applying for a credit card.
- A decision model *InterestRates* that defines the rules that determine if the customer is eligible for a low interest rate offer.
- An activity *Offer* within the dynamic process that gets activated in runtime if the decision's outcome or result is TRUE.
- A form *CreditCardApplicationForm* where among other details such as customer name and email, you enter values that are used as input values for the *InterestRates* decision model.

Before you begin, review the main steps for using a decision in a dynamic process. See [Configure a Decision Sentry in a Dynamic Process](#).

1. Create a process application and its components.
 - a. Create a dynamic process *CreditCard* that houses activities required while applying for a credit card. One of the activity can be the *Offer* activity that offers the low interest rate.
 - b. Create or import a decision model *InterestRates*, and activate it. Use the activated decision in your application so that it is present under **Decisions** in the Process Application page.



The *InterestRates* decision model's logic decides if a customer is eligible for a low interest rate offer (when the *Offer* activity gets activated), depending on the credit score and debt-to-income ratio.

- Low interest rate is available if the credit score is greater than 750
 - Low interest rate is also available if the credit score is greater than 650 and the debt-to-income ratio is less than 15%
- c. Create a form with relevant fields such as **Credit Score** and **Debt To Income Ratio** whose values are used as input for the decision. Set the dynamic process to start with the form.
2. Configure data and its flow in the dynamic process.

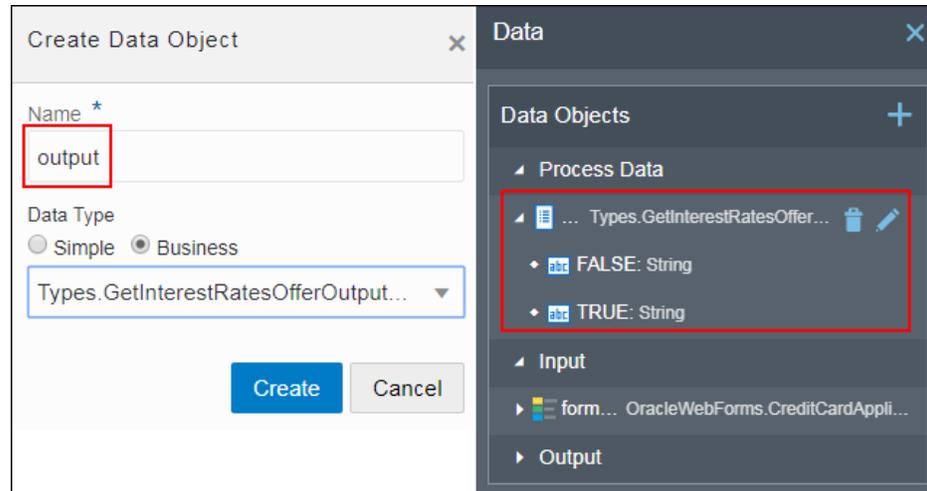
Use the `GetInterestRatesOfferInput` business object from the decision's services in the Start form of your dynamic process. This ensures that the values entered in the **Credit Score** and **Debt To Income** fields flow into the `InterestRates` decision model from the dynamic process.

3. Configure the decision's condition from the properties pane of the activity that you want to activate.
 - a. Open the properties pane of the Offer activity and create an activation condition. In the resulting Activation pane, click the **Create Data Condition** icon to create a data driven condition.
 - b. In the Create Data Condition dialog box, name the data driven condition as **DMN Condition**.
 - c. Select **Decision** as the condition type.
 - d. Select **InterestRates** as the decision model, and **GetInterestRatesOffer** as the decision service.
 - e. In the **Map Input Values** fields, the input value (body) from the decision model is auto-populated. Map the input value to the form's data object `formArg.getInterestRatesOfferInput`.

When the user enters values into the fields **Credit Score** and **Debt To Income**, the values flow into the decision model as input values, and the decision logic gets executed.

- f. Finally, define the condition in the **Condition** fields. Set `body.interpretation is equal to output.true`. This means if the response of the `InterestRates` decision model is true, then the Offer activity gets activated.

Note that `output` is the data type you create in the dynamic process. It contains the possible interpretation types (TRUE, FALSE) from the `InterestRates` decision model.



4. Try out the application in runtime as an end user.
Begin by test activating the application to validate and make it available in runtime.
 - a. Enter some values in the start form. Submit the form to create a dynamic process instance.

The screenshot shows a web application interface titled "My Apps". At the top, there is a blue header with the text "My Apps". Below the header, there is a sub-header with "Testing Mode" and two radio buttons, "On" (which is selected) and "Off". In the top right corner of the form area, there are "Submit" and "Cancel" buttons. The form contains several input fields with the following labels and values:

- Name: John Doe
- Email: john.doe@example.com
- Annual Income: 65,000
- Gross Monthly Income: 5,000
- Recurring Monthly Debt: 500
- GetInterestRatesOfferInput (checkbox): checked
- Credit Score: 800
- Debt To Income Ratio: 10

- b. Select and open the dynamic process to see the list of activities.

The InterestRates decision uses the values submitted in the start form (Credit Score:800 and Debt To Income=10) as inputs, and the resulting output is true as the defined rule *creditScore >650* and *debtToIncome <15* is fulfilled.

As the condition defined for activating the Offer activity fulfills, therefore we can see the Offer activity in the activity list.

Try out the instance again by entering different values in the start form. You'll observe that the Offer activity is not visible (that is, it doesn't get activated) if the defined decision condition doesn't fulfill.

You've successfully defined a decision sentry in a dynamic process.

Configure a REST Sentry in a Dynamic Process

Use a REST connector to activate or terminate a stage, activity, or process in a dynamic process.

Suppose, as a customer service manager of a credit card company, you need to check the credit scores of potential customers. Credit scores depend on several factors such as credit inquiries, account mix, credit age, credit utilization, and payment history. Based on these factors, complex mathematical algorithms are applied to get a credit score, and this is handled by the REST service. Based on the credit score provided by the REST service, you activate/terminate an activity with the interest rate offer for a potential customer.

Here are the main steps for using a REST connector to activate or terminate a stage, activity, or process in a dynamic process.

1. Create the process application and its components.
 - a. Create a dynamic process with activities to support the REST connector. For example, the dynamic process can contain an activity that is activated depending on the REST connector's response.
 - b. Create a REST connector and use it in the dynamic process. Optionally, import a REST connector into your application and use it in the dynamic process.

Ensure that the connector is available in the Integrations page of your application.

See [Create a REST Connector](#).

- c. Create a form to capture user inputs for the various human task activities in the dynamic process.
2. Configure data and its flow in the process.

Business objects are generally shared among the form, REST service, and dynamic process elements. For example, you can create a business type that is used in the form and as input data types in the REST service.

3. Configure the REST condition.
 - a. Select the activity, stage, or process and edit properties. Choose an activation or termination condition, set it up as a data driven condition, and create the condition.
 - b. In the Create Data Condition dialog box, enter a suitable name for the condition in the **Name** field.
 - c. Select **REST** as its condition type.
 - d. Select your (REST) integration in the **Integration** field, and the related resource and operation in the **Resource** and **Operation** fields.
 - e. In the **Map Input Values** fields, map a process variable as input to the connector's response body or parameter.
 - f. Define the REST connector's response as a condition. For example, if the REST service returns a customer's credit score as greater than 750 then activate the activity.
4. Try out the application in runtime to test the REST condition.

13

Integrate Documents

Some tasks require more collaboration than others. To help move these tasks along you might need to share documents with other task participants. You can do this by integrating Oracle Content Management with Oracle Integration.

Topics:

- [Why should I integrate documents?](#)
- [How do I integrate with Oracle Content Management?](#)
- [Roles and Responsibilities](#)
- [Quick Tour of the Documents Page](#)
- [Edit the Properties of the Documents Root Folders](#)
- [Define Folders and Documents](#)
- [Create a Document- or Folder-Initiated Process](#)
- [Get Properties for Documents and Folders](#)
- [Sample Use Case for Documents and Process](#)

Why should I integrate documents?

Although you can use email to discuss a task and send associated documents, it's much easier to have all that content available while viewing the task and tracking a process instance. Oracle Content Management integrates documents with your process applications.

What are the benefits of integrating Oracle Content Management:

- **Documents:** Process provides simple file attachment functionality, but if you need something more robust to handle document-intensive processes, you can integrate Oracle Content Management. This service enables you to organize files into folders, manage access to each folder, and even start a process when you upload a document. For example, if you're processing a home loan, you need to manage documents such as loan applications, employment histories, and house appraisals, making sure that the right users see the documents they need to submit, review, or approve, but they don't get access to restricted information.
- **Document- and Folder-Initiated Processes:** You can automatically start a process when someone uploads a document (or folder of documents) to a chosen document folder.

After you integrate Oracle Content Management, document folders are automatically created for each new application. You can also enable document functionality in existing applications. You can create additional folders as necessary to further organize documents, setting access to restrict content to the appropriate users. Then your users can upload and view documents while viewing the task (through icons on the Task Details page).

For a summary of the roles and responsibilities involved in configuring and using Oracle Content Management with Process, see Roles and Responsibilities. For general information about Oracle Content Management, see its [online library](#).

How do I integrate with Oracle Content Management?

Before you or any users can access the documents feature, an administrator must configure settings in both Oracle Content Management and Oracle Integration.

Only a user with administrator privileges can establish a connection between the two services.

For Process applications that were created *before* the connection with Oracle Content Management was configured, the documents feature is disabled by default. Developers can manually enable these features in their existing applications.

Topics:

- [Access Requirements for a Successful Integration](#)
- [Use Documents or Attachments in a Process Application](#)
- [Configure Settings in Oracle Content Management](#)
- [Configure Documents Settings in Oracle Integration](#)
- [Enable or Disable Documents](#)
- [Access Issues and Configuration Changes](#)

Access Requirements for a Successful Integration

Note these access requirements for a successful integration:

- The Oracle Content Management user who is configured in Oracle Integration must have full access to the folder (that is, the folder of the incoming document) that's configured in Oracle Content Management to be able to kick off a process.
- For a folder-initiated process (that is, a process with a **Folder Start** event), the Oracle Content Management user who is configured in Oracle Integration must have manager access to the folder in order to access its folder in Oracle Integration. For example, manager access is required to see the folder on the Task Details page and the Process Tracking page, to access the folder when embedding Process UI components in an external application, or see the folder in the Process Mobile application.
- To see a process in the process list for a folder, and to be able to initiate a process instance when a new document arrives, the Oracle Integration user who is configured in Oracle Content Management must be granted the process initiator role.

Use Documents or Attachments in a Process Application

Oracle Integration automatically includes standard **file attachment** functionality in your process applications. You can upload files and attach them to a process. When you use Oracle Content Management with Oracle Integration, you also get **documents** functionality, which lets you upload files, organize files into folders, manage access to

each folder, and even start a process by uploading a document. In addition, each process application has the option of using either documents or attachments.

You control whether a process uses documents or attachments at the application level. Documents are enabled by default. The setting is in the Information pane for the application.



An application that has **Documents Integration** enabled can use documents (and only documents). An application that has **Documents Integration** disabled can use only attachments. Basically, by disabling documents for an application, you enable attachments for that application.

Remember that the setting applies to the application. You can use documents or attachments in an application, but not both.

For attachments, ensure that the **Hide Attachments** option is deselected in the UI Customization screen. If the **Hide Attachments** option is selected, you will not be able to see attachments in task details and start forms.

If you use REST APIs to interact with Oracle Integration, be sure to use the appropriate API depending on whether your process application uses documents or attachments:

- For documents, be sure to use the /folders REST APIs.
- For attachments, be sure to use the /attachments REST APIs.

If you use the wrong API, the application either returns an error message or ends in a no-operation.

Note that if you're not able to see attachments in task details then check the options in the UI Customization screen. Ensure that the **Hide Attachments** check box is deselected to see attachments.

Configure Settings in Oracle Content Management

In Oracle Content Management, an administrator must enable Oracle Integration, enter connection information, and then enable Oracle Integration use for one or more folders.

Want to learn more about each step? See *Integrate with Oracle Integration* in *Integrating and Extending Oracle Content Management*.

Configure Documents Settings in Oracle Integration

As the administrator, you must also configure the connection between Oracle Integration and Oracle Content Management. You need to enter information such as the URL and sign-in credentials for your Oracle Content Management.

To configure the settings in Oracle Integration:

1. In the Oracle Integration navigation pane, click **My Tasks**, click **Workspace**, and then click **Administration**.
2. Click **Services**.

3. In the Oracle Content Management section, enter the following information:
 - **URL:** the web address of your Oracle Content Management. Your service administrator receives a Welcome to Oracle Cloud email when the service is ready to use. The email has the URL for your Oracle Content Management. For example, `https://your_service_name.com/documents`.
 - **Identity Domain:** the name of the identity domain that your Oracle Content Management belongs to. You can successfully configure a connection to your Oracle Content Management (for the documents feature) without providing an identity domain.
 - **User and Password:** the account credentials for a user who has access to your Oracle Content Management. This user account is used to test the connection between the services. It's also used during runtime to connect to the services and perform all the runtime operations, such as creating folders.
4. Click **Test**.

Whenever you make any changes to the configuration settings, it's a good idea to verify that the values you entered are correct. You want to confirm that a successful connection has been established with Oracle Content Management.

Review the test results, which may include messages, errors, and warnings.
5. Select one of the following options to continue:
 - If there are any errors or warnings, make the necessary changes and then click **Test** again to verify the new values. Repeat the test each time you change the settings.
 - If the connection test is successful, click **Save** to save the configuration settings.
 - If you want to cancel and return to the last-saved values, click **Revert**.

Enable or Disable Documents

After an administrator establishes a connection between Oracle Content Management and Oracle Integration, you can use documents in your process applications. These features are enabled by default for all new applications you create.

For your process applications that were created *before* the connection with Oracle Content Management was configured, the documents feature is disabled by default. If you want this feature in existing process applications, then you need to manually enable the features.

There are many benefits to having documents available when you're modeling a process, viewing tasks, and tracking a process instance. However, for some particular process applications, you might not want the documents feature to be available. In such cases, you can manually disable the feature.

To enable or disable the documents feature for a particular process application:

1. Open the application so you can see it on the Application Home tab.

If you're viewing a QuickStart App or a QuickStart Master, then you'll need to click **Switch to Advanced View**.
2. Click **More details**  to expand the Information pane.

The check box next to **Document Integration** show you the current status of the documents feature for this application.



3. Use the check box to enable or disable the documents feature for this application.
4. Activate a new revision of the process application. On the **Activate Options** tab, select the **Force Default** option.
 - Once you can activate Oracle Content Management for a process application already in use, new instances will use documents and existing instances can continue to use native attachments.
 - For any given process application, documents and attachments are mutually exclusive. For applications where **Documents Integration** is enabled, then those applications can use documents and only documents. When you disable **Documents Integration** for an application, then that application can use attachments and only attachments. By disabling documents, you enable attachments.

Access Issues and Configuration Changes

After you configure Oracle Content Management in Oracle Integration, keep in mind that access issues or configuration changes can result in errors.

For example:

- If you modify the Oracle Content Management configuration that is currently in use, its associated documents can no longer be accessed.
 - For documents, you'll get an error when you click **Documents**  or when you access a document from the list on either the Task Details page or the Tracking page.
- If an application folder created by a process gets removed, an administrator must restore it from the Oracle Content Management trash folder.

Roles and Responsibilities

The following table provides a summary of the roles and responsibilities involved in configuring and using Oracle Content Management with Oracle Integration.

| Role | Tasks |
|-------------------------|---|
| Administrator (Runtime) | Configure a connection to Oracle Content Management. You must configure a connection before anyone can use Oracle Content Management features. |

| Role | Tasks |
|-------------------------|--|
| Developer (Design time) | <ul style="list-style-type: none"> • Manually enable the documents feature. To include documents features in applications that were created before you configured the connection to Oracle Content Management, you need to manually enable them. • Create additional document folders. To further organize documents or restrict access to folders that will store important documents, you can create additional folders for a process instance. • Edit the basic properties for root document folders. If you created additional document folders, select which folder is displayed when the process is started. You can also change the default application folder name or the default process instance folder name (for example, if you have a folder naming standard). • Override the default document folder access for a particular task. To change the folder access for a user completing a particular task, you can override the default folder access. For example, you might hide an underwriting documents folder by default (by setting the access to None), but change the access to Contributor during the underwriter's approval task. • Create document- or folder-initiated processes. To start a process when someone uploads a document (or folder of documents) to a particular document folder, you can use the document start or folder start event to model a process. |
| End User (Runtime) | <ul style="list-style-type: none"> • Add a document to a task. Adding a document might complete a task within the process (for example, submitting a loan application or travel request), or it might just provide information to other users that interact with the task (for example, uploading justification for a travel expense). • Manage documents in folders. You can perform actions on files, such as moving or uploading a new version. You can also browse and create folders, including multiple levels. |

Quick Tour of the Documents Page

The Documents page provides the tools required to quickly update the settings and permissions for folders and documents that are stored in Oracle Content Management and used in Oracle Integration at runtime.



Note:

The Documents page is available only if an administrator for Oracle Integration created a connection to Oracle Content Management. In addition, for process applications that were created *before* a connection to Oracle Content Management was configured, the Documents feature is disabled by default. For these cases, you must manually enable the Documents feature.

Configured Folders and Documents

The Documents page displays information about the folders and documents that have been configured for your process applications.

For each process application that has the Documents feature enabled, Oracle Integration automatically creates the following default folders:

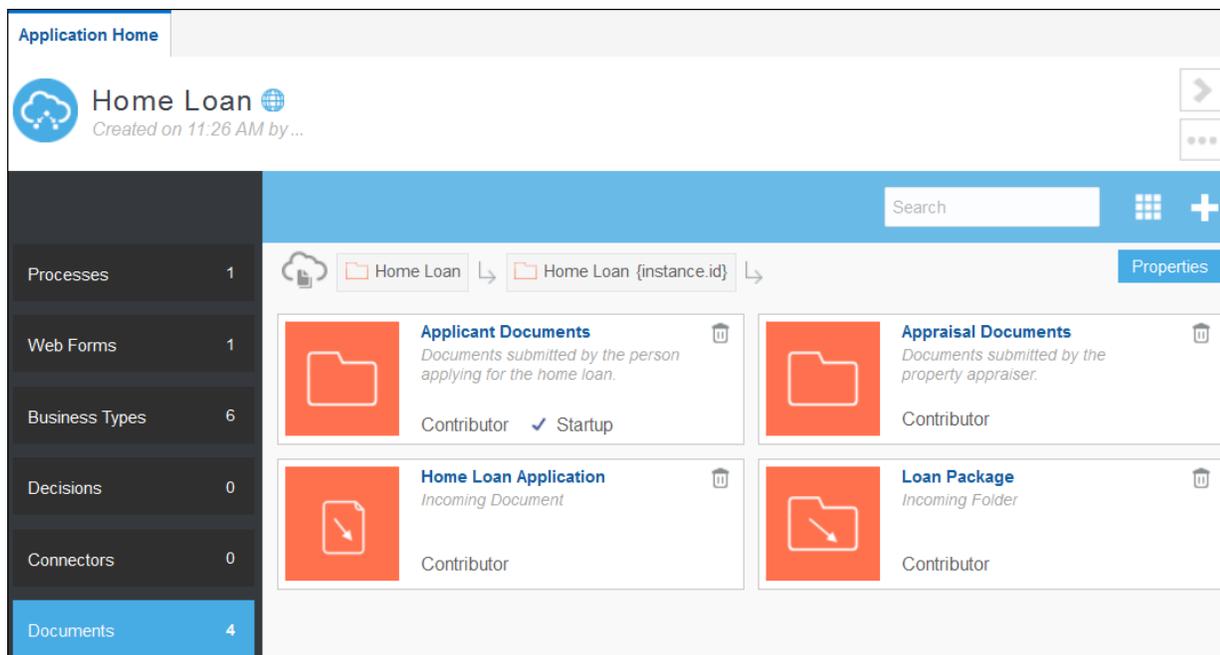
- **Application Folder:** This main root folder contains all the process instance folders.
- **Instance Folder:** One root folder is created for each process instance. It contains the set of managed folders that have been defined in the design-time environment.

 **Note:**

The incoming documents and incoming folders that have been defined in design-time environment aren't stored in the Instance Folder.

- **Managed Folders:** These folders are defined to organize the uploaded files and to set the access level. One managed folder is automatically defined and marked as the Startup folder. The **Startup folder** is the only folder that's shown for users to upload documents when starting an application. You can define additional managed folders. All managed folders are created for each process instance.

Select **Properties** to change the name of the root folders or to specify a different folder as the startup folder.



Information Displayed for Each Folder or Document

Each property card includes an icon that indicates whether the definition is for a managed folder (), an incoming folder (), or an incoming document ().

Each property card also displays the following information:

- The name of the folder or document. You can click the name to edit the basic properties.
- A description of the folder or document. If no description was provided, then the card displays the definition type: **Managed Folder**, **Incoming Folder**, or **Incoming Document**.
- The access type displays the default permission level required to access the folder or document. You can override the default access type at the task level. The available access types are Contributor (default), Downloader, Viewer, or None.

Note that **Startup**  indicates which folder has been selected as the Startup folder. Select **Properties** to select a different folder as the startup folder. The Startup folder is the only folder that is shown when a process is started.

Summary of Tools for Managing Folders and Documents

Use the following tools to view, configure, and manage your folders and documents:

- Click **Views**  to alternate between viewing the page in either a grid format or a list format.
- Click **New Folder or Document**  to configure folders and documents.
- Click **Properties** to edit the name of the root folders or to change the startup folder.
- Click the folder name to edit the basic properties of the folder or document.
- Click **Delete**  to delete a folder or document.

Edit the Properties of the Documents Root Folders

For each application that has the Documents feature enabled, Process automatically defines an application folder, a process instance folder, and a managed folder. You can change the name of the root folders or specify a different folder as the startup folder.

To edit the basic properties of these folders:

1. Select your process application.
2. Click **Documents**.
3. Click **Properties**. The Edit Documents Root Folders dialog is displayed.

*** Application Folder Name**
Travel Approval

*** Instance Folder Name**
Travel Approval Request {instance.id}

The instance ID is automatically added to the instance folder name.

*** Startup Folder**
Application Documents

The startup folder becomes available when you start a process.

Save

4. Make any necessary changes.
 - **Application Folder Name:** This folder is the main root folder for the application. It contains all the process instance folders.
 - **Instance Folder Name:** One root folder is created for each process instance. It contains the set of managed folders that have been defined in the design-time environment.

 **Note:**

The incoming documents and incoming folders that have been defined in design time aren't stored in the Instance Folder.

- **Startup Folder:** The folder that you select is the only folder that's shown for users to upload documents when starting an application.
5. Click **Save**.

Define Folders and Documents

When you're designing a process application, you can create definitions for managed folders, incoming folders, and incoming documents for each process instance. You use managed folders to organize files. You create definitions for incoming folders or documents if you plan to model a process that uses a folder start event or a document start event.

For managed folders, incoming folders, and incoming documents, you can define the access level (also called permission), and override the default permission at the task level. For example, you can restrict access to important documents that are required during the business process.

To configure folders and documents:

1. Select your process application.
2. Click **Documents**.

3. Click **New Folder or Document** .
 4. Select whether you want to create a definition for a managed folder, an incoming folder, or an incoming document.
 5. In the **Name** field, enter a unique name.
 - If you're defining a managed folder, the folder gets created with this name when the process instance gets initiated.
 - If you're defining an incoming folder, you use this name to associate the folder with a folder start event.
 - If you're defining an incoming document, you use this name to associate the document with a document start event.
 6. In the **Description** field, enter a brief explanation that can help other users understand the purpose of the folder or document.
 7. In the **Default Access Type** field, select one of the following roles from the drop-down list:
 - **Contributor**: Contributors have complete access. They can modify, update, upload, delete, download, save, and view files.
 - **Downloader**: Downloaders can download files and save them to their own computer. They can also view folders and files, but can't change things.
 - **Viewer**: Viewers can only look at folders and files. They can't change things.
 - **None**: No access allowed.
- The role you select defines the default access. You can then [override the default access](#) at the task-level, if required.
8. Click **Create**.

The new folder or document appears on the Documents page.

To edit the basic properties of a folder or document, go to the Documents page and click the name of the folder or document.

Create a Document- or Folder-Initiated Process

Use the **Document Start** event to model a process that can be initiated by a document. Use the **Folder Start** event to model a process that can be initiated by a folder.

You must have an Oracle Content Management, and you must configure a connection between that service and your Oracle Integration before you can create a document-initiated or a folder-initiated process.

By enabling the Oracle Content Management integration, you can define folders that will be created automatically on Oracle Content Management for every process instance, providing a predefined organization of the documents involved. You can also override the access type at the task-level to define the right permissions for that folder or document for a particular task based on your business needs. For example, you might want to prevent users from viewing a classified document or folder associated with a task.

Note these access requirements for a successful integration:

- The Oracle Content Management user configured in Oracle Integration must have full access to the folder (that is, the folder of the incoming document) configured in Oracle Content Management to be able to kick off a process.
- For a folder-initiated process (that is, a process with a **Folder Start** event), the Oracle Content Management user configured in Oracle Integration must have manager access to that folder in order to access the folder in Oracle Integration. For example, manager access is required to see the folder on the Task Details page and the Process Tracking page, access the folder when embedding Process UI components in an external application, or see the folder in the Process Mobile application.
- The Oracle Integration user configured in Oracle Content Management must be granted the process initiator role in order to see the process in the process list for a folder and to be able to initiate the process instance upon arrival of a new document.

To design a process that can be started by a document or folder:

- Define the **Incoming Document** or the **Incoming Folder** in Oracle Integration.
- Model a process that has a document start event or a folder start event.
- Customize its implementation to map the start event with the corresponding incoming document or folder you created. Implementation options allow you to define the way in which the document or folder is exposed to users.
- Modify what role can access the document or folder at the task level (optional).
- Configure the folder on Oracle Content Management to initiate a process on document arrival.

Alternatively, you can use REST APIs to instantiate a process instance and provide all the input values.

Define the Incoming Document or Folder

To define the incoming document or folder, open the process application you are modeling and click **Documents**. The Documents page lists the incoming documents and incoming folders that have already been defined.

To create a new incoming document or folder, click **New** , select the appropriate type, enter a name, and select the default access (permission).

Model Processes with Document or Folder Start Events

Next, build your process that has a document start or folder start event. In the current release, there are some limitations for how you can create the process. Failure to follow the required procedure will invalidate the process.

Caution:

You must add the document start or folder start event from the Elements palette. Only a single, and only the first, start event that you add to a process will be supported. Multiple start events aren't supported in this release. Also, don't change a start event using the Change Type option, and don't delete the first start event. Both actions will invalidate the process for use.

To model a process that has a document start or folder start event:

1. On the Application Home tab, click **Processes**.
2. Click **New process**  to open the Create Process dialog.
3. Select **None**.

 **Note:**

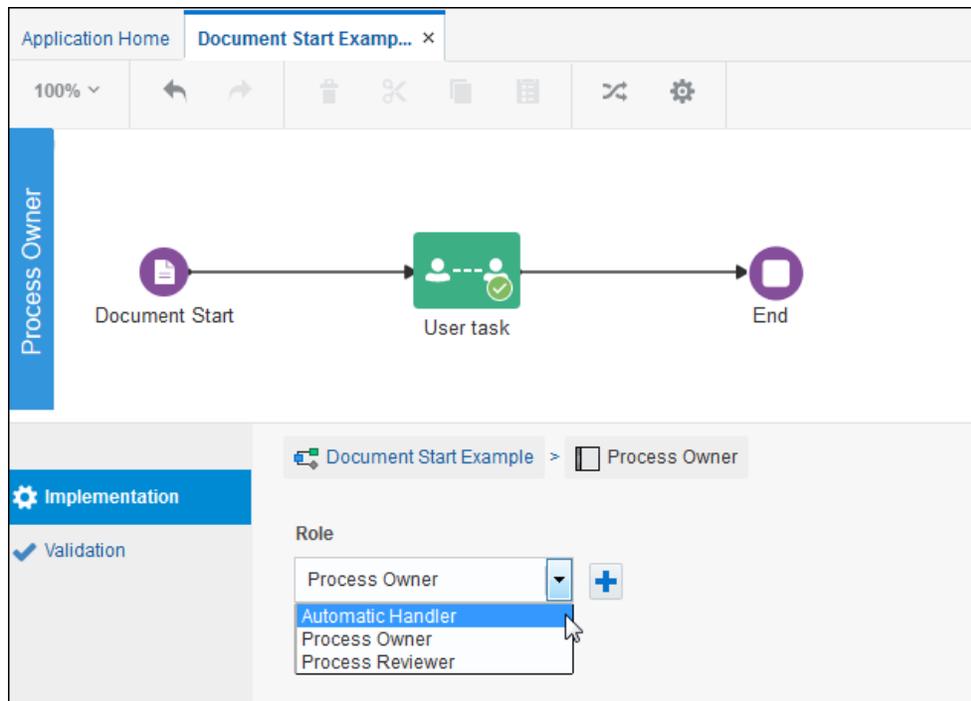
Be sure to select **None**. You will change this empty start event later.

4. Enter a name for the process, confirm the **Open Immediately** check box is selected, and then click **Create**.
The canvas displays the start and end events in the process.
5. Click **Events** in the Elements Palette.
6. Drag either the **Document Start** event or the **Folder Start** event to the canvas.
7. Add a sequence flow from your new document or folder start event to the end event.
8. Delete the empty start event.
9. Continue to modify and define the process. Be sure to test and deploy the process.

Allow All Users to Start a Document- or Folder-Initiated Process

To allow any valid user to upload a document to start a process:

1. Open the process.
2. In the swimlane with the document or folder start event, click the role name and then click **Edit** .
3. In the **Role** field, select **Automatic Handler**.



Customize Your Document- or Folder-Initiated Process

After you model the process, you can add a start document or folder in a process and customize it.

In your process, click **Document Start** or **Folder Start** event in the process diagram. From the icons that display, click **Menu**  and then select **Open Properties**. The Properties pane expands into view below your process diagram.

Define how you want to handle the incoming document or folder:

- **In Place:** Selecting this value keeps the location of the document as it is wherever it is. Optionally, you can map the incoming document or folder with one of the predefined documents or folders (open the drop-down list) for its management.

If there isn't a predefined incoming document or folder suitable for this particular process, then click **New**  to create one.

- **Unmanaged:** The document or folder is ignored by the current process. Oracle Integration won't show an unmanaged document or folder in runtime. It's up to the process modeler to handle the incoming document or folder. For example, if you want to move the incoming folder or document to another location, then you can use an XPath expression to get properties, such as `Id` or `Type`, and pass the property information to a REST service.

Customize Access at the Task Level

In your process, click a user task in the flow diagram. From the icons that display, click **Menu**  and then select **Open Properties**.

The Properties pane expands into view below your process diagram.

Click **Override** to customize the current permission of any element defined.

Configure the Folder

To have a process automatically start when a document is uploaded to a folder, you need to configure the folder on Oracle Content Management to initiate a process on document arrival:

1. Sign in to Oracle Content Management.
2. Select the folder.
3. Select **Properties** from the menu bar.
4. Enable the **Initiate process on document arrival** setting.
5. Select the process from the list.
6. Click **Save**.

The folder and its subfolders are now available for use within the Oracle Integration interface. Any change to a file in the folder or any new file uploaded to the folder triggers the process associated with the folder. You can override the inherited process for a subfolder, but you can't disable the association with a process.

When a file is uploaded from Oracle Content Management into a folder configured for use with Oracle Integration, the file is used for the task associated with that folder. Users in Oracle Integration can take any actions on the files there, such as approvals. When a task step is completed, the file can be moved or managed according to the defined process.

When Oracle Content Management starts a process, the payload sent to launch the process includes this information:

- Document ID
- Document name
- ID of the user who started the process
- Type
- Role (indicates the role that should be used to generate subsequent application links)
- Version

The following example uses only the document ID and document name for display in the form in Oracle Integration. In addition, the document ID is used for making REST API calls to move or copy the file in Oracle Content Management into the task folder.

```
{
  "processDefId": "testing~UserFileApproval!1.0~FormApprovalProcess",
  "operation": "startEvent",
  "params": {
    "id": "D2806600E495B744E66BF3981212FF6185DE89BE6812",
    "type": "d",
    "name": "document-name",
    "startedBy": "user-id",
    "role": "role that should be used to generate subsequent
aplinks",
    "version": "version"
  }
}
```

```
    }
}
```

**Note:**

The first operation in the process must be a `start` event. Otherwise, the process application will return a 500 error to Oracle Content Management.

As a developer, you must be aware of the following requirements for the process you develop:

- It needs to be a process that uses an Oracle Content Management Start event.
- When deploying the process, you need to share it with the user specified for enabling the integration so that user has the rights to trigger the process.
- For the user who uploaded the file to show up as the user who started the task, the process must use the value passed in the **startedby** field as the display name for the initiator.
- If you enable the process integration for a folder, you need to share this folder with the Oracle Content Management user that was used to enable the integration in Oracle Integration.

Use REST API Calls to Instantiate a Process

Most times, you will want to use the document start event or the folder start event to model a process that can be initiated by a document or folder. Alternatively, you can use REST APIs to instantiate a process instance and provide all the input values.

Fetching Processes Started by Documents or Folders

To fetch only those processes that can be started by a document, use the **doc** value in the **interfaceFilter** filter for the process-definitions API.

```
GET /ic/api/process/v1/process-definitions?interfaceFilter=doc
```

To fetch only those processes that can be started by a folder, use the **folder** value in the **interfaceFilter** filter for the process-definitions API.

```
GET /ic/api/process/v1/process-definitions?interfaceFilter=folder
```

As with any process, Oracle Integration doesn't validate if a process instance has already been initiated for the same set of input arguments. An end user can instantiate any number of process instances for the given document or folder details.

Any user who has the permission/role to instantiate this process can invoke the above API to query and fetch the list of processes that can be started by a document or folder.

Instantiating a Process Instance

To instantiate a process instance, including a document-initiated or a folder-initiated process, use the standard API:

```
POST /ic/api/process/v1/processes
```

You pass several parameters that define the ID, type, and name of the document or folder.

For example:

```
{
  "processDefId": "process-definition-id",
  "operation": "startEvent",
  "params": {
    "id": "document-id|folder-id",
    "type": "d|f",
    "name": "document-name|folder-name",
    "startedBy": "user",
    "role": "contributor"
  }
}
```

For the `type` input parameter, you can enter "d" for document or "f" for folder.

Currently, the `role` input parameter is ignored. All process participants automatically get contributor role access to the document or folder that initiated the process instance.

Like all processes, only users who are members of the start swimlane role can instantiate a process.

Sample Code

You can use the `cURL` command to invoke the Process REST API. For example:

```
curl.sh
host=localhost
port=7001
user="user_name:password"
curl -H "Content-Type:application/json" -u $user -d @input.json
http://$host:$port/ic/api/process/v1/processes
```

Here is sample `input.json` that adheres to the required interface. Note that for the `type` parameter, a "d" indicates document and an "f" indicates folder.

```
input.json
{
  "processDefId": "default~DocumentStart!2.0~Process1",
  "operation": "startEvent",
  "params": {
    "id": "document-id|folder-id",
    "type": "d|f",
    "name": "document-name|folder-name",
    "startedBy": "user",
    "role": "contributor"
  }
}
```

And here is the sample response returned by this API:

```
JSON response
{
```

```
"levels": 0,
"links": [
  {
    "href": "http://abc01abc.example.com:7001/ic/api/process/v1/processes/19",
    "length": 0,
    "rel": "self"
  },
  {
    "href": "http://abc01abc.example.com:7001/ic/api/process/v1/processes/19/audit",
    "length": 0,
    "rel": "audit"
  }
],
"title": "Instance #19 of Process1",
"processId": "19",
"processDefId": "default/DocumentStart!2.0/Process1g",
"processName": "Process1",
"priority": 0,
"owner": "DocumentStart.ProcessOwner",
"creator": "myname",
"state": "OPEN"
}
```

Get Properties for Documents and Folders

You can use an XPath expression to fetch document properties or folder properties given the name of the document or folder. You can then pass the property information to a REST service.

For example, as part of your document workflow patterns, you may have a need to copy documents *to a process folder* or get documents *from a process folder*. Suppose you have a process that manages transactional content and documents from a central IN box initiate the process. Once the process is initiated, the documents need to be moved from the central IN box to a process folder.

Or, after documents have been successfully reviewed and approved, you might want to move those documents to a final approved location. For example, think about a process that manages market assets. In this case, you might want to get the documents from a process folder and move them to an approved marketing collateral folder.

Get Document Asset Property

In the Expression Editor, you can select **Get Document Asset Property** from the Document Service option. You can get a property for an asset, which in this case can be a document or a folder.

Here is the usage format:

```
DocumentService.getDocumentAssetProperty(<propertyName>,<documentAssetName>)
```

You need to enter two parameters:

- Property Name, which is of String type, defines the property that you want to get. Supported properties are `Id` and `Type`. The value returned for `Id` is the actual ID of the

document or folder used in Oracle Content Management. The value returned for Type is INCOMING_DOCUMENT, MANAGED_FOLDER, or INCOMING_FOLDER.

- Document Asset Name, which is the name of the document or folder.

The command returns a property value of string type.

If you don't provide a parameter or if you provide more than two parameters, an error occurs.

Defining the Simple Expression in the Editor

To get properties and access the ID for folders or documents:

1. In the Oracle Integration navigation pane, click **Processes**.
2. Open an application.
3. Click **Processes**.
4. Open a process, and click an activity in the process.
5. Click **Menu** , and select **Open Data Association**.
6. Click **Expression Editor** .
7. Select the **Operators** tab.

In the Expression Editor dialog box, locate the following options on the Operators tab:

- Document Service
8. Select the expression to insert.

As with all simple expression functions, you click **Insert into Expression** to enter the parameter values and then click **Validate** to verify your expression.

An empty string is returned if there are no values for the document or folder properties.

| Example XPath Expression | Sample Result |
|---|--|
| DocumentService.getDocumentAssetProperty("Id" , "TestFolder") | F14573D5EECE5BAB5724CE41T0000DEFAULT00000000 |
| DocumentService.getDocumentAssetProperty("Type", "TestFolder") | MANAGED_FOLDER |
| DocumentService.getDocumentAssetProperty("Id" , "Incoming_Document") | D214D537E68C726FB70FF6E1T0000DEFAULT00000000 |
| DocumentService.getDocumentAssetProperty("Type", "Incoming_Document") | INCOMING_DOCUMENT |

Sample Use Case for Documents and Process

Documents are an integral part of a process application because they're commonly used in approval flows. Let's look at a sample process for getting a home loan.

During the loan process, different types of documents must be filed and reviewed, including:

- Applicant related documents, such as bank statements, pay stubs, W-2 form, and credit reports
- Property related documents, such as title and appraisal

Different roles, such as loan officer, loan processor, and loan underwriter, review the documents and approve the loan. Once the loan is approved, additional documents, such as loan agreement documents, must be created, signed, and approved.

These review and approval activities require segregating documents into different folders, moving them to approved folders, and so on, during the loan processing flow. Integrating Oracle Content Management with Oracle Integration provides the capability for you to create and manage these folders in design time.

Document Folders

Using the home loan as an example, the following table lists the folders that would be required for the loan process and the types of documents you would expect to find in each folder.

| Folder | Description |
|---------------------|--|
| Applicant Documents | Documents related to the applicant, such as W-2 form and bank statements |
| Appraisal Documents | Documents related to the appraisal, such as property photos and recent comparable sales |
| Approval Documents | Documents related to the loan approval, such as credit score, risk model analysis, and loan term computation |
| Agreement Documents | Documents containing the loan terms and conditions |
| Property Documents | Documents related to the property, such as title and insurance |
| Funding Documents | Documents created as part of funding, such as updated title and wiring instructions |

Actions Performed by Tasks on Folders

Continuing with our home loan example, the following table shows the sequence of tasks in the loan process, lists who performs the task (swimlane), and describes the task performed on the folder.

| Sequence | Swimlane | Task |
|----------|--------------------|--|
| 1 | Loan Applicant | Uploads W-2 forms and bank statements to the Applicant Documents folder. |
| 2 | Property Appraiser | Uploads appraisal related documents to the Appraisal Documents folder. |
| 3 | Loan Agent | Reviews both the applicant documents and the appraisal documents to ensure completeness. If additional documents are required, then request the applicant or appraiser to upload them. |

| Sequence | Swimlane | Task |
|----------|-----------------|--|
| 4 | Loan Officer | Reviews both the applicant documents and the appraisal documents, approves the loan, and creates loan documents in the Agreement Documents folder. The applicant documents, appraisal documents, and approval justification documents are stored in the Approval Documents folder. |
| 5 | Loan Applicant | Signs agreement documents and uploads them to the Agreement Documents folder. |
| 6 | Title Company | Uploads all documents related to the property to the Property Documents folder. |
| 7 | Finance Officer | Reviews approval documents, agreement documents, and property documents in their respective folders, and funds the loan. Also uploads related documents to the Funding Documents folder. |

Folder Permission for Various Tasks

Finally, for the home loan example, the following table lists the access permission required for each folder based on the user and the task.

| Swimlane | Tasks | Folder and Access Permission |
|--------------------|---|---|
| Loan Applicant | Submits applicant documents, and if approved, submits agreement documents | Applicant Documents and Agreement Documents folders; Contributor access |
| Property Appraiser | Submits appraisal documents | Appraisal Documents folder; Contributor access |
| Loan Agent | Reviews applicant documents | Applicant Documents and Appraisal Documents folders; Viewer access |
| Loan Officer | Approves the loan | Applicant Documents and Appraisal Documents folders; Downloader access Approval Documents and Agreement Documents folders; Contributor access |
| Title Company | Submits property documents | Property Documents folder; Contributor access |
| Finance Officer | Funds the loan | Approval Documents folder; Viewer access Agreement Documents and Property Documents folders; Downloader access Funding Documents folder; Contributor access |

14

Integrate with Applications and Services

You can integrate Process with other applications and services in a number of ways.

Topics:

- [What You Can Do on the Integrations Page](#)
- [Work with Integrations](#)
- [Create REST and Web Service Connectors](#)
- [Apply Message Security to Integrations](#)
- [Invoke Integrations, REST, and Web Services](#)
- [Embed Process UI Components in Other Applications](#)
- [Use Process UI Composite Components \(CCA\)](#)
- [Work with Insight Models](#)
- [Integrate with Robotic Process Automation Tools](#)
- [Integrate an External UI with the Process Task List](#)

Process contains an optional out-of-the-box integration with Oracle Content Management for document sharing. See [Integrate Documents](#).

What You Can Do on the Integrations Page

Use the Integrations page to manage integrations, web service and REST connectors, and to manage the life cycle of web service definition files.

The Integrations page is divided into two views:

- Integrations
- Definitions

Use the view option to alternate between views:



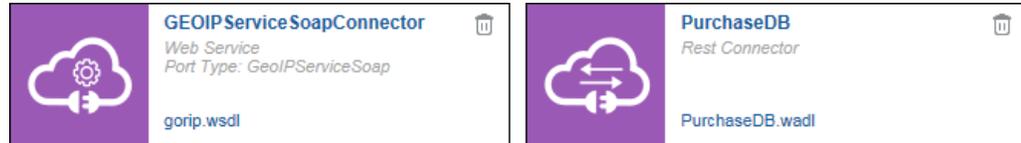
Integrations View

Use the Integrations view to perform the following tasks:

- [Configure integrations](#)
- [Create web service connectors](#)
- [Create REST connectors](#)
- View connector information in either Grid or List format. Click **Views**  to alternate between views

- View and edit integrations, and web service and REST connector information
- Delete integrations, and web service and REST connectors

You can see the integrations, and the web service and REST connectors that are available for use in any process created for the selected application, as shown in the Grid view.



To edit a REST or web service connector, click its name. To view the details of the WSDL file, click its name.

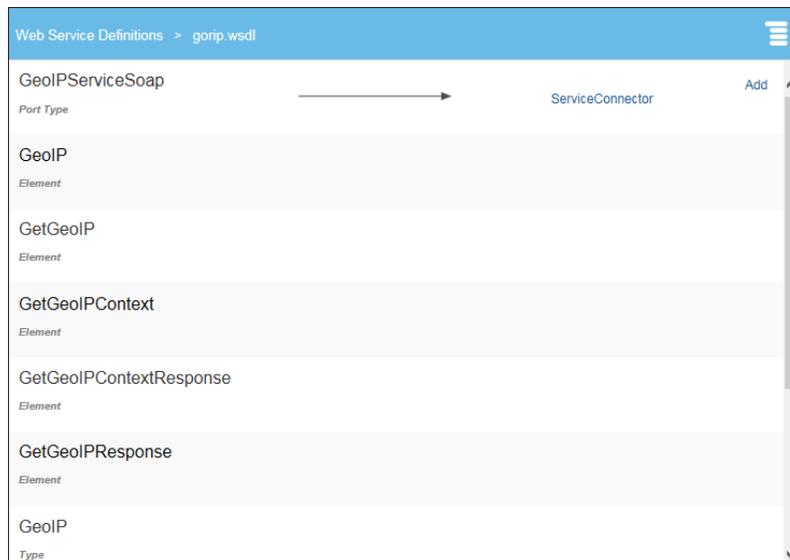
You can see if a definition update is available for an integration or if an integration is unavailable (deactivated or deleted) by two different icons in the Grid view.

| Integration Update Icon | Description |
|---|--|
|  | The integration is active but a definition update is available for the same. |
|  | The integration has been deactivated or deleted. Check for updates. |

Definitions View

Use the Definitions view to perform the following tasks:

- Import web service definitions.
- View WSDL file information either in Grid or List format. Click **Views**  to alternate between views.
- View details of the WSDL file, as shown here.



- Upload a new version of the WSDL file.
- Add a new web service connector for a specific port type.
- Delete WSDL files.

In this view, you see the WSDL files that are included in the application.



Click **Edit**  to display a detail view for the web service definition. In this view, a list with the elements of the WSDL is displayed. You can perform the following tasks:

- Edit the web service connector defined for the specific port type.
- Add a web service connector based on the WSDL file and specific port type.
- Upload a new version of the WSDL file.
- Delete a WSDL file.

Note that you can't delete WSDL files that are being used by connectors.

See [Work with Web Service Definition Files](#).

Work with Integrations

Integrations enable users to communicate with local and remote applications that are exposed as either SOAP or REST.

Note that Oracle Integration Processes supports SOAP and REST technology adapter based integrations only.

Topics:

- [About Integrations](#)
- [Configure Integrations](#)
- [Edit Integrations](#)

- [Update Integrations](#)

About Integrations

Integrations enable you to communicate with applications in the cloud using adapters.

Use integrations to integrate between diverse systems or endpoints. An integration is composed of two different connections, the source and the target, and a mapping set between the types used by the distinct systems. The target connection is the endpoint system, which is called and is implemented by an adapter. The source connection is the entry point to the integration from Process. It's represented by a source system or endpoint, which triggers the integration execution.

Process applications can communicate and exchange data with local and remote applications that are exposed as either SOAP or REST.

In Oracle Integration, you first create and activate integrations and then use these integrations in your process applications to communicate with other applications in the cloud. For more information on developing integrations, see *Developing Integrations with Oracle Integration* in *Using Integrations in Oracle Integration Generation 2*.

Once you have set up an integration, you can select it and use it in a process. After setting up an integration, you must map a data object to hold the information to be sent or received as a payload. See [Configure Integrations](#).

Configure Integrations

Use integrations to communicate with applications using adapters.

Configuring an integration involves the following main tasks:

- Add a connection to an active integration
- Use the integration in a process

To configure an integration:

1. Add a connection to an active integration.
 - a. On the Application home page, click **Integrations** and select the **Integrations** view.
 - b. Click **Create** and select **Use an Integration**.
 - c. In the Use an Integration dialog box, select an integration from the list of active integrations and click **Create**.
2. Use the integration in a process.
 - a. In the Elements palette, click **Expand**  next to the Integrations flow element. Drag and drop the integration that you want to add onto the canvas.
 - b. Optionally, if you choose not to display the integration in the elements palette, you can implement the integration in one or more service tasks in the process. Add or open a service task in the process. In its properties, implement a service call to an integration. See [Invoke Integrations, REST, and Web Services](#).
 - c. Configure data association to pass new values as input and output to the integration. For example, if you're querying an order from Oracle Sales, pass an Order ID as input and you'll receive an object representing an order as

output. You need to map this order object to your business data objects as needed.

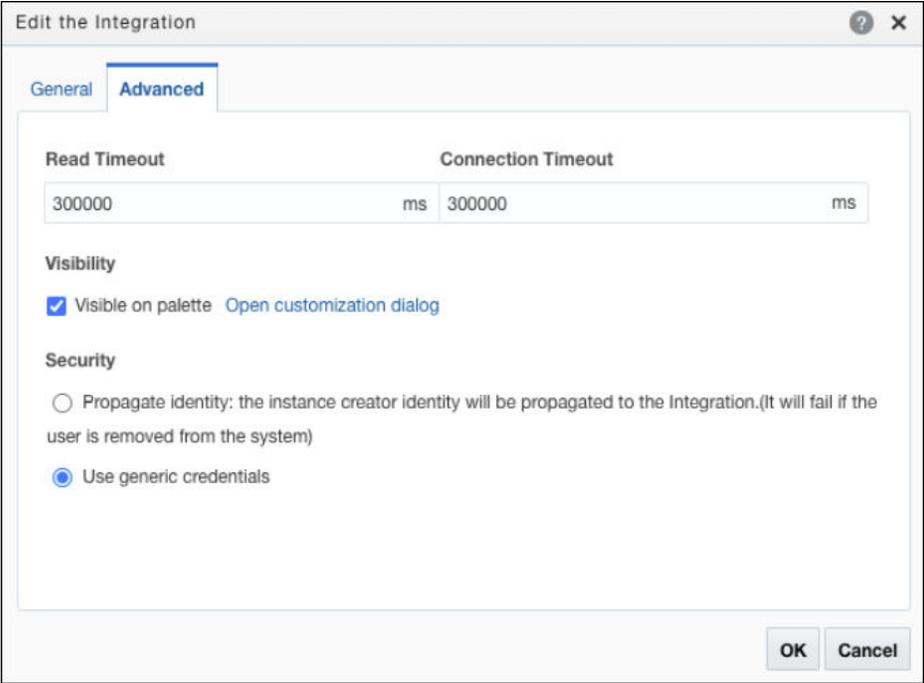
- d. Repeat these steps if you need to configure other integrations, perform an operation, and store data changes in the business object.

To delete an integration, click its delete icon on the Integrations page. You can't delete an integration that is being called in a process.

Edit Integrations

You can edit an existing integration in the Edit Integration dialog box in Integrations view.

1. Click an integration to open the Edit Integration dialog box.
2. In the Advanced tab, specify timeout settings and visibility settings for the integration.
 - a. In the **Advanced** tab, specify the read timeout and the timeout for connecting to the integration in milliseconds.



The screenshot shows the 'Edit the Integration' dialog box with the 'Advanced' tab selected. The dialog has two tabs: 'General' and 'Advanced'. Under the 'Advanced' tab, there are two input fields for timeouts: 'Read Timeout' and 'Connection Timeout'. Both are set to '300000 ms'. Below the timeouts, there is a 'Visibility' section with a checked checkbox 'Visible on palette' and a link 'Open customization dialog'. The 'Security' section has two radio buttons: 'Propagate identity: the instance creator identity will be propagated to the Integration.(It will fail if the user is removed from the system)' (unselected) and 'Use generic credentials' (selected). At the bottom right, there are 'OK' and 'Cancel' buttons.

- b. Deselect or select the **Visible on palette** check box to hide or display the integration in the elements palette. This option is selected by default.
- c. If you choose to display the integration in the palette, click **Open customization dialog**, and specify a name to be used for the integration in the elements palette.



- d. Under **Security**, choose one of the options to specify the credentials for authenticating the integration.
 - Select **Propagate identity**, if you want to use the credentials of the user that created the instance. Note that, if for any reason this user is removed from the system, the integration call will fail.
 - Select **Use generic credentials**, if you don't want identity propagation to happen. In this case, Process uses the same internal system provided credentials for every integration call. Note that these credentials cannot be changed or removed.
- e. Click **Apply** to save your changes and close the dialog box.
- f. In the Use an Integration dialog box, click **OK** to save your changes.

Update Integrations

You can keep integrations that you use for your process application up-to-date. Update to the latest definition or to the latest version of the integration from the Edit the integration dialog box in Integrations view.



If a schema definition update is available for an integration, you can update to the latest definition. Click the integration to open the Edit the integration dialog box. In the Edit the integration dialog box, click the update to latest definition link.



If you see the deactivated or deleted icon for an integration, click the integration to open the Edit the integration dialog box. In the Edit the integration dialog box, you get one of the three options:

- Update to the latest version.
- Update to the latest incompatible version. In this case, you get a message to validate the application after you update the integration, as you may need to fix some references to the updated schema.
- Update is not supported as the SOAP port type changed. This is applicable only to SOAP (web service) integrations.

Create REST and Web Service Connectors

You can create connections to exposed REST and web services using Process. Your process applications can communicate and exchange data with these services.

Topics:

- [About REST and Web Services](#)
- [Create a REST Connector](#)
- [Create a Web Service Connector](#)
- [Work with Web Service Definition Files](#)

About REST and Web Services

Process applications can communicate and exchange data with local and remote applications that are exposed as either REST or web services.

When considering whether to use REST or web services, keep in mind that some applications support one and some the other, so the protocol decision may already be made for you. As a general rule, use REST services for integration over the web, and use web services for enterprise application integration scenarios. Cloud applications provide open REST APIs for consumers to interact with them and applications running in the cloud typically communicate through REST calls.

REST and web services are client and server applications that communicate over the World Wide Web (WWW) using HyperText Transfer Protocol (HTTP).

For general information about REST and web services, see:

- The W3C home page for the Web Application Description Language (WADL) file:
www.w3.org/Submission/wadl
- The W3C home page for the Web Services Description Language (WSDL) file:
www.w3.org/TR/wsd1
- The Web Services chapter of the Java EE 6 Tutorial:
docs.oracle.com/javaee/6/tutorial/doc/gijti.html

You will need some basic information about the REST or web service you want to connect to. However, you don't need to know the details of how services are structured.

Connection to a REST Service

To create a connection to a REST service, you need the following service information:

- Definition of the REST service to connect to WADL, RAML, YAML or other

- URLs to the location of the different resources
- Access to these URLs to get the JSON sample used to create the types needed to send and receive data to/from the service
- List of operations to use on each resource
- List of parameters to pass to operations
- For a secure REST service, the user name and password required to access the service

Connection to a Web Service

To create a connection to a web service, you need the following service information:

- Location of the WSDL file, as a local file path or a URL
- Port type and callback port type in the WSDL file to select
- For a secure web service, the user name and password required to access the service

Create a REST Connector

Use the outbound REST connector to call a REST service to retrieve, create, update, or delete data on a web server that supports the REST architecture. The connector enables Oracle Integration to interact with other Oracle Cloud applications via REST, including SaaS and PaaS applications running inside or outside Oracle Cloud. Using the REST Connector editor, you define the resources and operations needed to connect to a REST service, regardless of the description language used to define the service.

Creating a REST connector involves the following main tasks:

- Create the connector, which opens the editor.
- Set connector timeout settings.
- Apply authentication to the connector (optional).
- Add resources to the connector.
- Add operations to resources.
- Specify request and response parameters for operations.
- Specify visibility settings for the connector on the elements palette.
- Implement the REST connector within a process.

For example, you might create an Oracle Integration application that interacts with Oracle Eloqua via REST service calls, in which the application retrieves an email from Oracle Eloqua and updates the email with information from a process.

- The REST connector uses a GET operation to retrieve email content from Eloqua, and a PUT operation to update the email with the response. Business objects contain the sample payload values.
- The process contains two service tasks that act as Eloqua integration points. The process receives the email ID to approve. One service task fetches the email content from Eloqua, using the supplied email ID. After an approval or reject action, the other service task modifies the email content to contain the outcome of APPROVED or REJECTED.

To create, configure, and implement a REST connector:

1. Create a REST connector.
 - a. On the **Application Home** tab, click **Integrations** and select the **Integrations** view.
 - b. Click **Link to an Integration** and then select **Create a REST connector**.
Optionally, click **Create**, then click **External**, and then select **REST**.
 - c. In the Create REST Connector dialog box, enter a name and base URL for the REST service, and click **Create**.

The Rest Connector editor opens, with expandable Configuration and Resources sections. The Configuration section displays the base URL you specified.
 - d. Click **Edit** to expand the connector's configuration settings.
2. In the General tab, set timeout settings for the REST connector (optional).

The screenshot shows the 'Configuration' dialog box with the 'General' tab selected. The 'Base URL' field contains the placeholder text `\${localServerURL}`. Below it, there are two spinner controls: 'Read Timeout' and 'Connection Timeout', both set to the value '300,000'. The dialog has 'Apply' and 'Cancel' buttons in the top right corner.

- a. In the **Read Timeout** field, specify the timeout in milliseconds to wait to read data from the host.
 - b. In the **Connection Timeout** field, specify the timeout in milliseconds to make the initial connection.
3. In the **Security** tab, apply authentication to the REST connector (optional).

By default, no security is configured. You can apply HTTP basic authentication, which provides access control to web resources by requiring a user name and password when making requests.

The screenshot shows the 'Configuration' dialog box with the 'Security' tab selected. The 'Security Type' dropdown is set to 'APP Id - Basic Authentication'. The 'Keystore Credential' dropdown is set to 'New Key'. The 'Key Name' field is empty with the placeholder text 'Enter the name of the key'. The 'Username' and 'Password' fields are also empty. The dialog has 'Apply' and 'Cancel' buttons in the top right corner.

- a. In the **Security Type** field, select **APP Id - Basic Authentication** to apply basic authentication to the connector.
- b. Complete the keystore credential fields that display. To create a new key, select **New Key** and enter a key name, user name, and password.

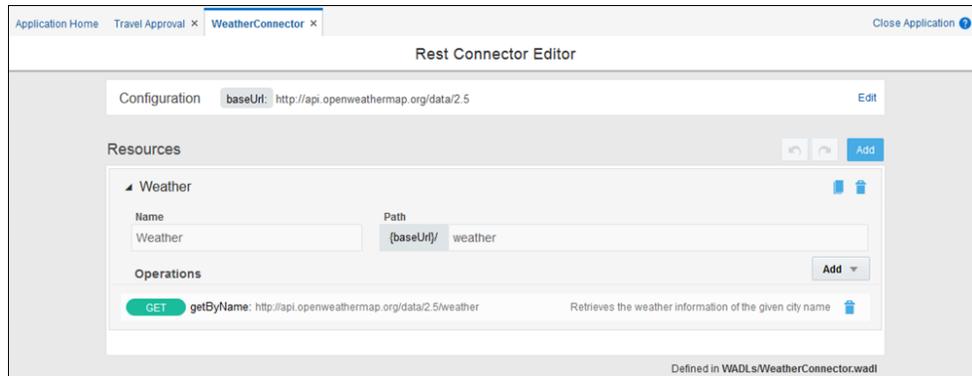
- c. Click **Apply** to save the configuration settings.

Note that Apply is active only when there are changes to save.

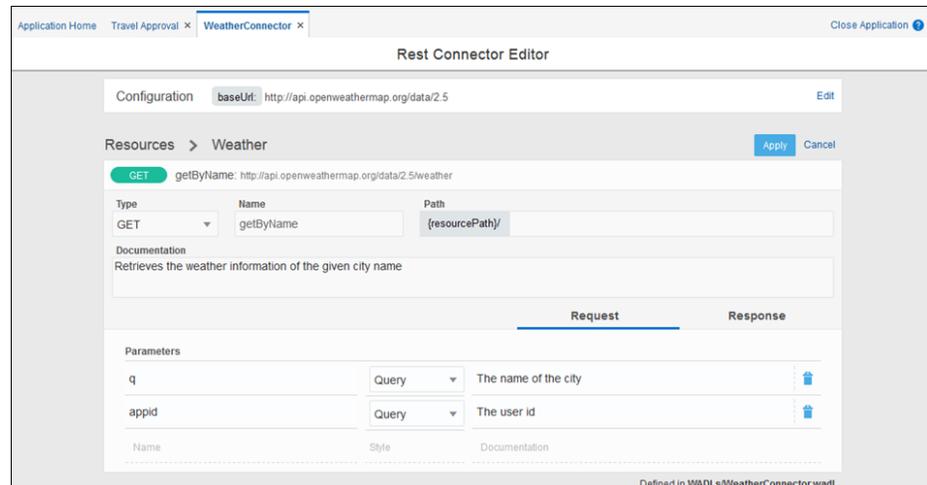
Want to learn more about managing credentials? See [Configure Credentials for Web Services](#).

- 4. Add resources to the connector.

A resource contains one or more operations that control data by performing basic create, read, update, and delete operations (CRUD) on resources using standard HTTP method requests.



- a. At the top of the Resources section, click **Add**. Click the new resource that was added to the Resources list to expand it.
 - b. Enter a name for the resource. You'll select this resource name when implementing a service task.
 - c. In the **Path** field, identify the resource path within the base URL.
 - d. If needed, duplicate (clone) and delete resources, and undo and redo changes to them, using options in the Resources section. Duplicating a resource can be useful when configuring similar resources.
- 5. Add operations to resources.
 - a. In the Operations section, click **Add**.
 - b. Select an HTTP method to add (GET, PUT, POST, PATCH, DELETE, or HEAD). The new operation is added to the Operations list, along with its name and path.



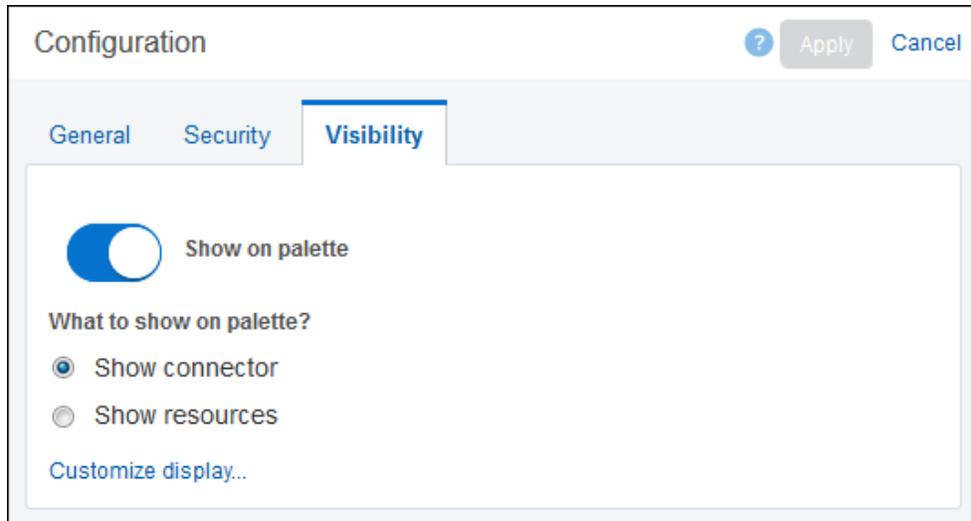
6. Specify request and response parameters for operations.
 - a. In the Operations list, select an operation to expand it. If needed, make changes to the operation's **Type**, **Name**, and hierarchical **Path** fields. Optionally, enter a description in the **Documentation** field.
 - b. Click the **Request** and **Response** tabs to view the operation's parameter and body fields. The HTTP method you selected determines the request and response message combination to complete for the operation. An asterisk displays if one or more required fields on the tab need to be completed.

Refer to the table below for required and optional parameters for HTTP methods. Information in this table is based on the Methods definition section of the [HTTP/1.1 protocol definition](#).

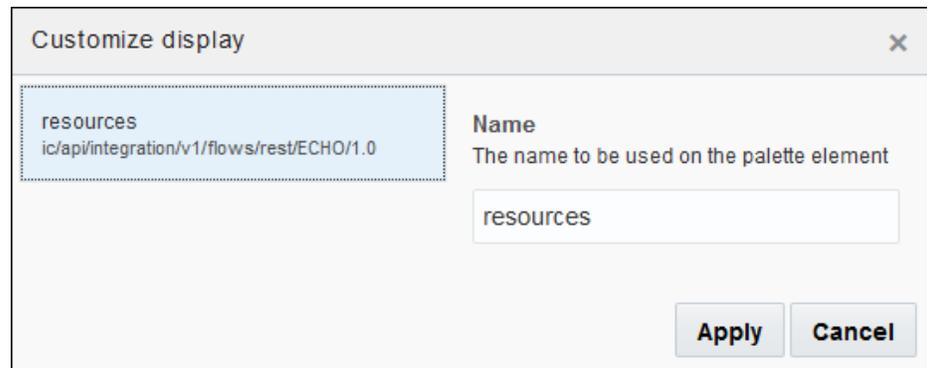
| HTTP Method | Description | Request Message-Body | Response Message-Body |
|-------------|--|----------------------|-----------------------|
| GET | Retrieve information from a resource | Optional | Yes |
| POST | Create a resource | Yes | Optional |
| PUT | Completely update an existing resource | Yes | Optional |
| PATCH | Partially update an existing resource | Yes | Optional |
| DELETE | Delete a resource | Optional | Optional |
| HEAD | Identical to a GET except that no message body is returned in the response | Optional | No |

- c. In the **Body** field, add or select a business object to store the message data. You can create business objects based on JSON files for REST operation payloads. To create a business object by importing or pasting a JSON sample, click **Create Business Object +**, and complete the fields in the Import Business Object from JSON dialog box, uploading or pasting JSON sample text, and clicking **Import**. The JSON text is validated, and if invalid, an error displays.

Ensure that the JSON output of the REST connector does not contain elements that start with a number or contain spaces. These elements may throw up a failed mapping error during runtime when converted to XML. Element names are case sensitive, and must start with a letter or underscore.
 - d. When you select a business object in the **Body** field, you must also specify the message's media type in the **Media type** field. Oracle Integration supports *application/json*.
 - e. In the **Parameters** table, configure one or more parameters for the operation. Click an empty row to add a line, and complete name, style, and description fields. Depending on the selected method, supported style options include template, header, body, and query. When you add, edit, or remove a parameter token in the operation's path (for example, {Id}), its corresponding template parameter is automatically created, edited, or deleted.
 - f. Click **Apply** to save the operation. Apply is active only when all required information has been entered.
7. In the **Visibility** tab, specify visibility settings for the connector on the elements palette.



- a. Click the **Show on Palette** icon to hide or display the connector on the elements palette. The connector is set to show on the palette by default.
- b. Select:
 - **Show connector** to display the connector on the palette.
 - **Show resources** to display the connector resources on the palette.
- c. Click **Customize display...** to customize the name of the connector or resource to be displayed in the palette. If you choose to display a connector resource, you can choose the display name for each resource to be displayed in the palette.



8. Implement the REST connector within a process.
 - a. In a process, expand the **Integrations** flow element, and drag and drop the connector you want to use onto the canvas.
 - b. Using data association, configure input and output for the connector. For example, if the REST connector uses a GET operation to retrieve email content from Eloqua, and a PUT operation to update the email with the response, configure the business object to store the values as they're changed by the service calls.
 - c. If you have chosen not to display the REST connector in the palette, you can implement it in one or more service tasks within a process.

In this step, configure one or more service tasks to call the REST service, perform an operation and store data changes in the business object. See

[Invoke Integrations, REST, and Web Services](#). Configure [data association](#) to pass new values (such as the body message and template) as input and output to the service task.

- d. Add or open a service task in the process. In its properties, implement a service call to a REST service. Select the REST connector, resource, and operation to call.
- e. Using data association, configure input and output for the service task. For example, if the REST connector uses a GET operation to retrieve email content from Eloqua, and a PUT operation to update the email with the response, configure the business object to store the values as they're changed by the service calls.

To delete a REST service connector, click its delete icon on the Integrations page. You can't delete a REST service connector that is being called by a service task.

Work with Web Service Definition Files

From the Definitions view of the Integrations page, you can import a web service definition (WSDL) file, upload a new version of the file, and add a new web service connector file based on a specific port type.

On the Application Home tab, click **Integrations** and then click the **Definitions** view option.

Import a WSDL

You can use this import feature to update current versions of WSDL files. However, updating your WSDL files using the import feature is different from the update feature you can find in the Definitions — Details view where you can only update the current WSDL file. When updating WSDL files using the import feature you update the WSDL file and XSD dependencies contained in the ZIP file or imported through a remote URL.

To import a WSDL:

1. Click **Import**  to open the Upload Web Service Definition File dialog box.
2. Select one of the following options:
 - **Upload from file:** Click **Browse** to browse for a WSDL or ZIP file.
 - **Use URL:** Provide the URL for your WSDL or Zip file.
3. Click **Validate**, and then click **Upload**.

After clicking **Validate** a summary of the files to be added are shown if the validation is successful.

Note:

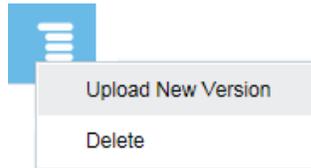
If validation is unsuccessful, instead of showing a summary of the files, the validation errors or warnings are displayed. Validation warnings occur if some of the files that you want to upload override existing ones. In this case you can upload and override existing files. Whenever an unexpected error occurs, the error is logged in the server log and the upload is cancelled. You can't import files that fail validation.

When you accept the import by clicking **Upload**, the operation is committed and the files are uploaded to the application and shown in the Definitions view of the Integrations page.

Upload a New Version of a WSDL

To upload a new version of a WSDL:

1. Click the name of the WSDL that you want to override.
2. Click the **Options** menu and then select *Upload New Version* to open the Upload New Version dialog.



3. Select one of the following options:
 - **Upload from file:** Click **Browse** to browse for a WSDL or ZIP file.
 - **Use URL:** Provide the URL for your WSDL or Zip file.
4. Click **Validate** and then click **Upload**.

After clicking **Validate** a summary of the files to be added are shown if the validation is successful.

In certain business cases, you may need to update a WSDL (SOAP integration) with changes that are *incompatible* with the current WSDL.

An incompatible WSDL (SOAP integration) can include changes like:

- Port type (that is being used in the current WSDL) has been removed
- Some operations have been removed from port type
- Some elements have been updated to new types or removed from (XSD) schema

While updating a WSDL with incompatible changes, you get warning messages to indicate the impact of the changes. After you have updated a WSDL with incompatible changes, validate your application and fix any errors that may have occurred due to the update.

Add a Web Service Connector

To add a new web service connector based on a specific port type:

1. Click **Add** to open the Add Service Connector dialog. Enter a name and select the port type. You can also select a callback port type if required.
2. Expand **Advanced** and provide the following information, and then click **OK** to add the new web service connection.
 - **Read Time Out:** Specify the time out for reading the WSDL file in milliseconds.
 - **Connection Time Out:** Specify the time out for connecting to the web service in milliseconds.
 - **Security:** Select *None*, *APP Id – Basic Auth*, *APP Id – Username Token*, or *APP Id – Username Token With Message Protection* from the drop-down list.
See [Apply Message Security to Integrations](#).

- **Certificate:** If the Security is set to *APP Id — Username Token With Message Protection*, select **New Certificate Alias** from the drop-down list and click **Add Certificate** to upload a certificate.

See [Manage Security Certificates During Design Time](#).

- **Keystore Credential:** If Security isn't set to *None*, select a key from the drop-down list.

You can also select **[New Key]** and type a *Name*, *Username*, and *Password*. For an existing key, the **Username** and **Password** values are automatically populated.

Want to learn more about how to add credentials? See [Configure Credentials for Web Services](#).

Prepare Local WSDL Files Containing Remote References

If a WSDL file has dependencies on remote schema files, you can download the files and set up local references. Note that this step is needed if creating from a local file only. If creating from a remote URL, Process handles remote references for you.

You must follow the same steps for a process exposed as a web service or for a web service created outside of Process.

To prepare a WSDL file with remote references for use in a web service connection in Process:

1. Create a root directory with two subdirectories named `WSDLs` and `Schemas`.
You can create additional subdirectories of `WSDLs` and `Schemas`, as long as all files under `WSDLs` are WSDL files and all files under `Schemas` are schema files.
2. If the WSDL file is only accessible using a URL, download it and copy it to the `WSDLs` directory.
3. If the WSDL file has an extension other than `.wsdl`, such as `.xml`, rename it and give it a `.wsdl` extension.
4. Open the WSDL file in a text editor.
5. Search the WSDL file for URLs that reference schema (`.xsd`) files.

For example, a schema file reference might look like this: `schemaLocation="http://www.example.com/ExampleService/ExampleSchema.xsd"`

6. Download each schema file and copy it into the `Schemas` directory or a subdirectory of `Schemas`.
7. In the WSDL file, change each schema file reference to refer to the `Schemas` directory.

For example, an edited schema file reference might look like this: `schemaLocation="/Schemas/ExampleService/ExampleSchema.xsd"`

8. Save and close the WSDL file.
9. Zip the root directory that contains the `WSDLs` and `Schemas` directories.

Make sure the zipping process doesn't create any extra files in the `.zip` file. The `.zip` file must contain only `WSDLs` and `Schemas` directories and subdirectories that in turn contain only WSDL and schema files.

Use the resulting `.zip` file to create a connection to the web service. See [Create a Web Service Connector](#).

Use SOAP Headers in Data Associations

Create data inputs and outputs based on headers specified in the WSDL of a SOAP connector. The headers get exposed through the SOAP connector and the header values can be used for mapping input and output data while configuring data associations in the process.

1. Create a SOAP integration that has headers specified in its WSDL.

See Developing Integrations with Oracle Integration in *Using Integrations in Oracle Integration Generation 2*.

For example, we have created an integration that uses a WSDL (as shown below) with header.

```
> CUSTCORPORATEREA_GETCVRNUMBERBYP_01_00_0000.wsdl <xs:sequence>
</wsdl:portType>
<wsdl:binding name="CustomerCorporateRead_ptt_GetInput_REQUEST_binding" type="tns:CustomerCorporateRead_ptt_GetInput_REQUEST">
<soap:binding style="document" transport="http://schemas.xmlsoap.org/soap/http"/>
<wsp:PolicyReference xmlns:wsp="http://schemas.xmlsoap.org/ws/2004/09/policy" URI="#ws_http_token_or_jwt_token_service_policy" wsdl:required="false"/>
<wsdl:operation name="retrieveCvrNumber_byPolicyNumber">
<soap:operation soapAction="retrieveCvrNumber_byPolicyNumber"/>
<wsdl:input>
<soap:header message="tns:serviceContextHeader" part="header" use="literal" encodingStyle=""/>
<soap:body use="literal" parts="parameter"/>
</wsdl:input>
<wsdl:output>
<soap:body use="literal" parts="parameter"/>
</wsdl:output>
</wsdl:operation>
</wsdl:binding>
```

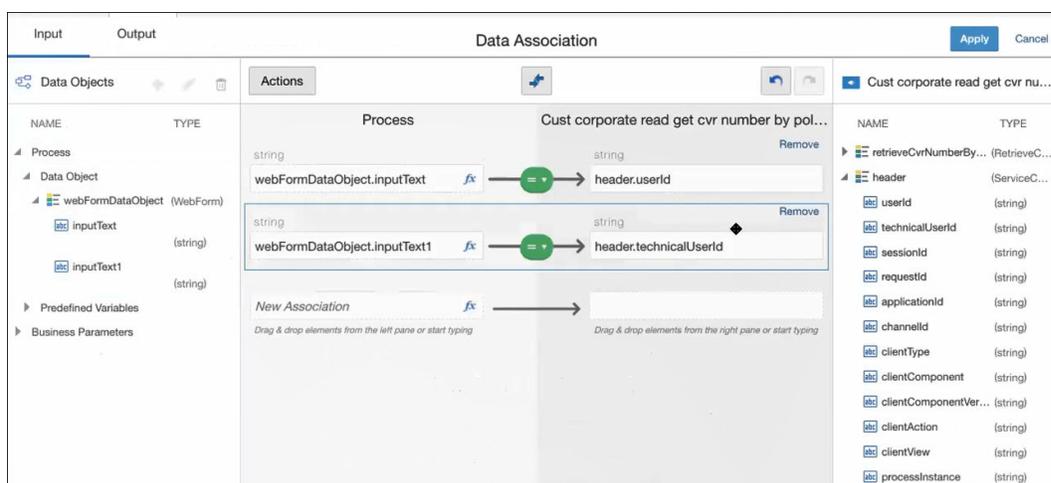
In the message of the header, there are header values such as `userId`, `technicalUserId`, and `sessionId`. These values can be used during data association.

2. Use an active integration in a process by creating a connector.
 - a. On the **Application** home tab, click **Integrations**.
 - b. In the Use an Integration dialog box, browse and select the integration you created with the WSDL containing the header.
 - c. Click **Create**.
3. Call the SOAP connector in a process.

You can do this either directly from the Integrations flow element (as shown in the steps below) or through a service task that calls the SOAP endpoints in the process .

- a. In a process, expand the **Integrations** flow element in the Elements palette.
 - b. Drag and drop the connector into the process flow in the canvas.
4. Configure data associations using the header values as input and output data.

For example, map `InputText` to `userId` (header value) and map `InputText1` to `technicalUserId` (header value).



See [Configure Integrations](#).

Create a Web Service Connector

When creating a connection to a web service using Process, you specify a WSDL file that is stored locally on your machine or available online. For a secure web service, you can ensure that only users with the proper credentials can access it.

If you're connecting to a secure web service outside of Process, you must obtain the credentials according to the instructions for that web service.

When you deploy a process beginning with a message start event, the process is exposed as a web service. The WSDL file URL is displayed when you select **Web Services** from the Actions menu on the Manage Deployed Applications page. You can then connect to the process just as you would to any other web service. See [Communicate Between Processes](#) and [Administrator Basics](#).

To configure credentials for a process exposed as a web service, you must have administrator privileges in Process. You can create a security key in two ways:

- When you create a web service connector in Process
- When you select **My Tasks** in the Oracle Integration navigation pane, click **Workspace**, and then click **Administration**.

Want to learn more about configuring credentials in Workspace? See [Configure Credentials for Web Services](#)

1. Create a web service connector in any one of the following ways:

- From the Application Home tab
- From the properties of a service task

To add a connector from the Application Home tab:

- a. On the Application Home tab, click **Integrations** and select the **Integrations** view.



- b. Click **Link to an Integration** and then select **Create a SOAP connector**.
Optionally, click **Create**, click **External**, and then select **SOAP**.

To add a connection from the Properties dialog box of a service task:

- a. Add or open a service task in the process. In its properties, implement a service call to a web service.
 - b. Click **Add** .
2. Select the source for the WSDL or Zip file, then click **Next**:
 - **Existing File**: Select an existing definition from the drop-down list.
 - **Upload from file**: Import a WSDL or ZIP file located on your local machine. A ZIP file can only contain WSDL and XSD files. The WSDL file must have valid references.
 - **Use URL**: Enter a URL address.
The WSDL is retrieved along with other depending files. These files are stored locally inside the application.

 **Note:**

When you click **Next**, the system automatically validates your selected files before importing.

3. In the Basic Properties dialog box, provide the following information:
 - **Name**: Enter a name for your service connector or use the default. Don't use spaces or special characters such as &%\$#/?.
 - **Port Type**: Select a port type from the drop-down list. A WSDL file exposes one or more port types for the service it defines.
 - **Callback Port Type**: Select a callback port type from the drop-down list.
4. Click **Create** to create your web service connector now, or **Next** to open the Advanced Properties page.
5. If you clicked **Next**, complete the Advanced Properties settings, and click **Create**:

- **Read Time Out:** Specify the time out for reading the WSDL file in milliseconds.
- **Connection Time Out:** Specify the time out for connecting to the web service in milliseconds.
- **Security:** Select *None*, *APP Id – Basic Auth*, *APP Id – Username Token*, or *APP Id – Username Token With Message Protection* from the drop-down list.
- **Certificate:** If the Security is set to *APP Id – Username Token With Message Protection*, select **New Certificate Alias** from the drop-down list and click **Add Certificate** to upload a certificate.
Want to learn more about how to add certificates? See [Manage Security Certificates During Design Time](#) and [Apply Message Security to Integrations](#).
- **Keystore Credential:** If Security isn't set to *None*, select a key from the drop-down list.
You can also select **[New Key]** and type a *name*, *username*, and *password*.
For an existing key, the **Username** and **Password** values are automatically populated.
- **Visibility on palette:** In the Visibility section, select **Visible on palette** to display the connector in the elements palette. If you choose to display the connector in the palette, click **Open customization dialog** to customize the name of the connector to be displayed in the palette. Enter a name and click **Apply**.

After you create a web service connector, you can only change the port types and the advanced settings. To change other settings, you must delete and recreate the web service connection.

You can't create business objects based on in-line types defined in a WSDL file. However, read-only business objects are automatically created from the WSDL file when you create a web service connector. You can create data objects based on these read-only business objects. Want to learn more about business objects and data objects? See [Manage Application Data](#).

Apply Message Security to Integrations

To protect message exchange for your integrations, especially when interacting with external cloud services leveraging SOAP, apply message protection. This protection applies request/response message encryption using OWSM message policies to the connection. As part of message security, you apply keystore credentials and security certificates. After you apply message protection, you can manage certificates and credentials in multiple ways in Process.

To configure message security:

1. Add or edit an integration. See [Creating a Web Service Connector](#).
2. In the Advanced Properties options, select **APP Id - Username Token With Message Protection**.

The window expands to include certificate alias and keystore credential fields.

The screenshot shows the 'Add Service Connector' dialog box with the 'Advanced Properties' tab selected. The dialog has three tabs: 'Choose Source', 'Basic Properties', and 'Advanced Properties'. The 'Advanced Properties' tab is active, showing fields for 'Read Time Out' and 'Connection Time Out' (both in ms). Below these is a 'Security' dropdown menu set to 'APP Id - Username Token With Message Protection'. Underneath is an 'Alias' dropdown menu. The 'Keystore Credential' section includes a dropdown menu set to '[New Key]' and a 'Key Name' text field. The 'Username' field contains 'User Name' and the 'Password' field contains a masked password '*****'. At the bottom, there are 'Back' and 'Create' buttons.

3. Complete the credentials fields.
Select a design-time key or select [New Key] and add one, entering a username and password. If needed, administrators can update credentials in runtime, as described in [Configure Credentials for Web Services](#).
4. In the **Certificate Alias** field, select a design-time certificate or select [New Certificate Alias] and create a new one. To create a new design-time certificate, click **Add Certificate**, enter an alias name, and either upload a file or paste certificate content.
5. Manage certificates and their expiration dates as needed.

You can manage and maintain separate certificates for design time and runtime, applying them as needed during deployment. See [Manage Security Certificates During Design Time](#) and [Manage Security Certificates during Runtime](#).

Invoke Integrations, REST, and Web Services

Using the connectors that you created, your applications can communicate with local and remote applications that are exposed as SOAP, REST, or web services.

If you choose not to display the integration, REST, or web service connector in the elements palette, you can implement the integration, REST connector, or web service in a service task in a process.

To invoke an integration, or a REST or web service:

1. On the Application Home tab, click **Processes**, and then click the name of the process to which to invoke the integration, or the REST or web service.
2. Drag and drop a Service flow element into the process.
3. Select the Service flow element, click **Menu** , and then select **Open Properties**.
4. Select *Service Call* from the **Type** drop-down list.
5. Click **Edit**  to display the Configure dialog.
6. Select *Integrations*, *Web Service* or *Rest*, then click **Browse** to display a list of integrations, or REST or web service connectors.
7. Select an active integration, or service connector, then click **OK**.
 - *Integration*: Select an active integration, then click **OK**.
 - *REST Connector*: Select a resource and operation, then click **OK**.
 - *Web Service Connector*: Select an operation, then click **OK**.

Embed Process UI Components in Other Applications

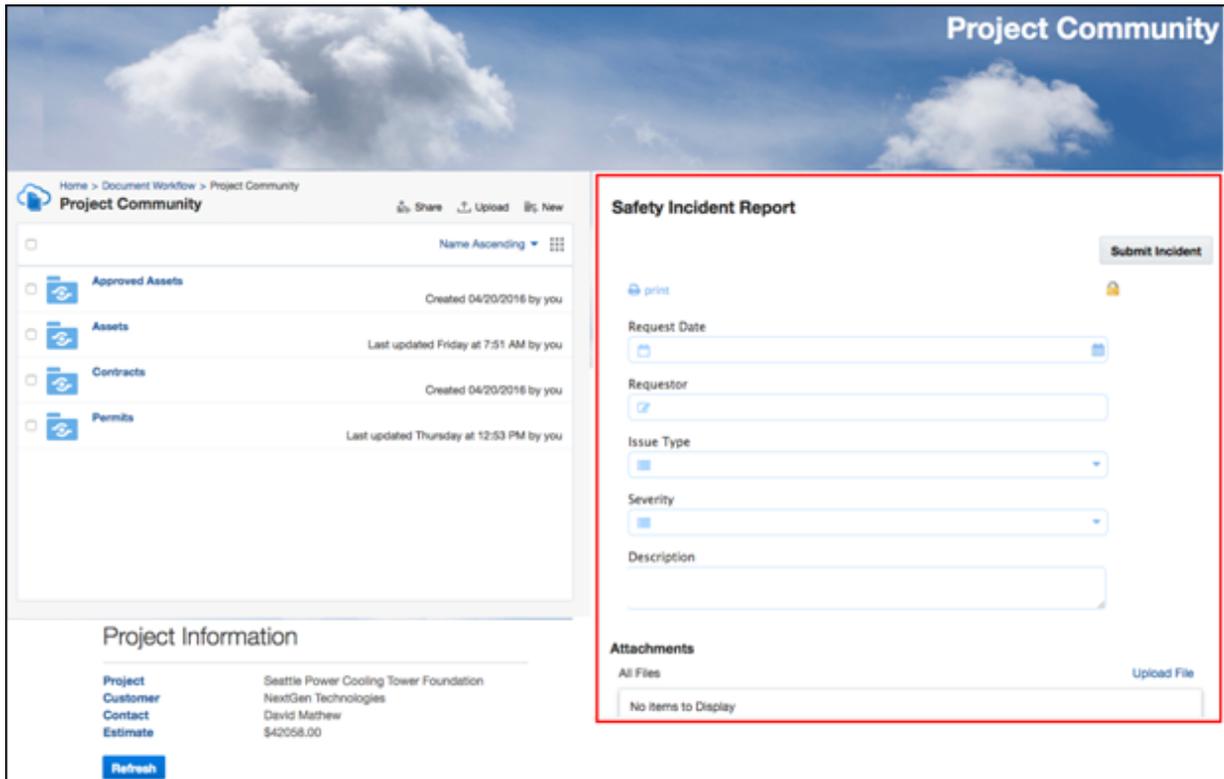
Integrate Process functionality in external applications, such as self-service applications, services, or portals, by embedding Process UI components provided as jQuery widgets. The external application or service consumes the embeddable Process component and renders it on the appropriate mobile, tablet, or web platform.

Topics:

- [Process UI Components Available for Embedding](#)
- [Before You Embed Process UI Components](#)
- [Download Process UI Component Files](#)
- [Experiment with Process UI Components](#)
- [Consume the Process UI Component](#)
- [Use Process UI Composite Components \(CCA\)](#)
- [Best Practices for Embedding Process UI Components](#)

Process UI Components Available for Embedding

After you embed Process UI components in an external application, your users can then start their tasks within the context of their primary focus rather than in Process runtime. For example, you might configure a Process form that initiates a process by enabling users to submit a safety incident report.



Use jQuery to embed a selected reusable Process UI component in any HTML page. Components related to the following categories are provided.

- **Task**
- **Application:** Display the list of applications and their start form or start form and attachments or documents. Documents display only if Oracle Content Management is configured, and only if documents are enabled for the application..
- **Dynamic Process:** Displays various components for dynamic process.
- **Charts**
- **Miscellaneous**

Embeddable Process UI components expose several options and events that the consuming application can use. You can customize the components using these options and get access to the internal data using public methods.

Whenever a significant user action occurs, the components trigger events. For example, when a user submits a form, an event is fired along with the instance data. You can listen to the events, and perform actions on the consuming application. See <http://learn.jquery.com/events/introduction-to-custom-events/>.

Before You Embed Process UI Components

Review these requirements and tips before you embed Process UI components in external applications.

- Embedding components requires knowledge of JavaScript, HTML, jQuery, and Require JS.
- An open endpoint to Process UI components is provided, and the cookbook is hosted directly on it. The open endpoint gives you several options:
 - Download and use the code in the zip file
 - Point directly to the Oracle Integration server to load JS/CSS resources remotely without the need to include the resource inside your application
 - Use the HTML Inline Frame Element `<iframe>` to embed the component pages
- Downloading files, including the entire library of embeddable Process UI component files and dependencies, are provided. See [Download Process UI Component Files](#).
- Newer versions of the component files may be available with each release, although you can download previous versions. Note that any customizations you make to previous versions must be made again to a newer version.
- To view and experiment with the UI component capabilities, use the cookbook. See [Experiment with Process UI Components](#).
- To change the look and feel of either a single component or the entire UI, override the cascading style sheet (CSS). See [Consume the Process UI Component](#).
- Embeddable components and REST APIs support cross-origin resource sharing (CORS).

Download Process UI Component Files

Before you can experiment with the Process UI components, you need access to the entire code base of the embeddable components, including the Process library, cookbook sample code, and dependency library. The library contains both the debug (complete) version and the minified version for JS and CSS files. You can download and host the component files. Alternatively, you can use an HTML Inline Frame Element (`<iframe>`) to embed the components.

To download and host the component files:

1. Access the open components endpoint on your Oracle Integration server using the following format:

```
host:port/ic/pub/components
```
2. Click **Resources**.
3. Download the files listed in the **Download** section.

Be sure to download and update component files after releases as needed. The Process library file identifies the component feature's version number. To ensure that you're using the latest embedded Process UI component files available in future releases, compare your current version number with the version number listed on the Resources page, and download the newer version if needed.

As an alternative to hosting the components, you can use them in an <iframe> element instead. For information, go to the **Resources** page, scroll to the **Server Access** section, and expand the **HTML** section.

Experiment with Process UI Components

Use the cookbook to try out Process UI component functionality, and view supported methods, events, and options. Note that the cookbook is provided for demonstration purposes only. You can't use it to consume the UI components.

To experiment with Process UI components:

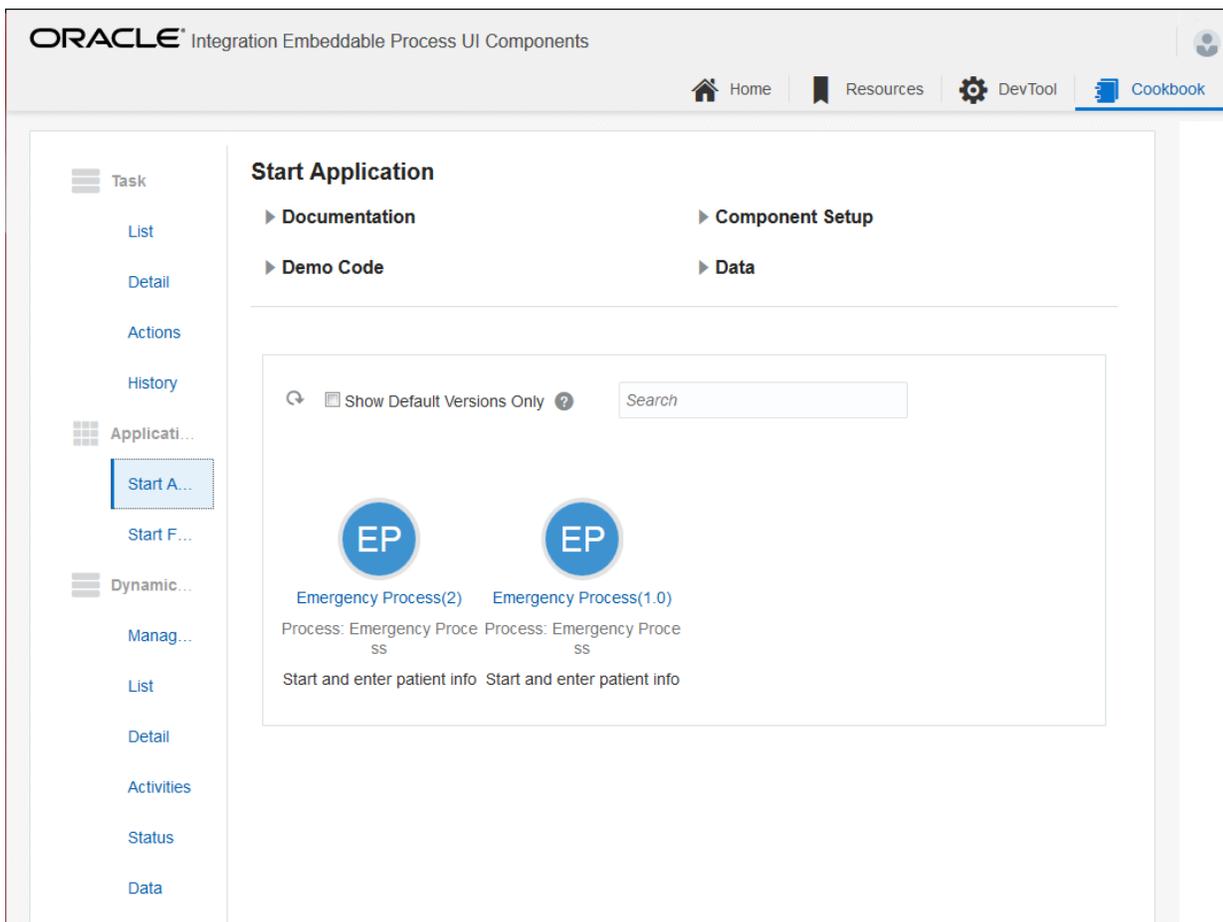
1. Access the cookbook at the following endpoint:

```
host:port/ic/pub/components
```

The Home page opens. It displays settings and information that apply to all the embeddable Process UI components.

2. On the Home page, complete the **Server Setup** fields to identify your Oracle Integration server.
 - You must configure these fields if you want to experiment with using the Process UI components.
 - Select the **testMode** option if you want to get and test production data. This option is equivalent to the runtime test mode.
3. Click **Apply**.

- **Methods:** Lists methods to access various data in the component. For example, use the applist component's `getStartformList` method to get a list of available forms. This information is useful for consuming the Start Form component independently. Expand the **Data** section to view event and method call results.
 - **Events:** Lists the user actions on the component that fire events that the consuming application can listen to and use in the external application's logic. Expand the **Data** section to view event and method call results.
8. Expand the **Demo Code** section to review sample JS and HTML code for the selected component, along with the component options currently being used to load the component. These options change when you change Component Setup values.
 9. Expand the **Component Setup** section, select or enter options, and click **Apply** to see how your changes affect the component. For example, try renaming or hiding buttons.



Consume the Process UI Component

Use the configuration, documentation, and demo code provided in the cookbook to guide you in integrating the component:

1. Consume the downloaded library's js/libs and css/libs inside your external application.

Alternatively, you can directly point the 'pcs' path in your requirejs file to the remote Oracle Integration server. For example:

```
'pcs' : 'PCS_SERVER_URL/ic/pub/components/js/libs/pcs/v1.2/min'
```

This command means you don't have to include the Process code in your application, and can get it remotely instead.

2. Create the Process connection (\$.pcsConnection) in the app main.js file and configure it to identify the server URL and authentication. Go to the Home page and expand the **Connection** section for details.

The authentication information is provided through the authInfo property, which can contain the Basic, JWT, or OAuth access token. The app main.js file is responsible for maintaining a valid token.

Basic authentication token can be provided by:

```
var tok = username + ':' + password;
$.pcsConnection.authInfo = 'Basic ' + btoa(tok);
```

Other access tokens (JWT or OAuth) can be provided by:

```
$.pcsConnection.authInfo = 'Bearer {token string}'
```

3. Configure component use in the app main.js file. For details, go to the Home page and expand the **Configuration** section.
4. Configure the selected component, as described on the Cookbook UI's component pages. For example, see Require, Usage, Options, Methods, and Events fields.
5. Optionally change a component's look and feel, or that of the entire UI, by overriding the cascading style sheet (CSS) files.

Each UI component has an ID and a class. For example, to change the application list links from circles to squares, override `.pcs-applist-process-holder-smaller` class.

- a. To change a component's css, expand the **Demo Code** and **HTML** sections on the component page and determine the CSS file. You can copy, modify, and override the CSS file listed.
- b. Use CSS selectors to access HTML elements and modify the look and feel after the UI component has rendered.

Best Practices for Embedding Process UI Components

To integrate the Process UI with external applications, you can either embed Process components directly into an external application's HTML or JS code or use URL iFrames.

The tips and best practices listed in this section correspond to the first approach; they're aimed towards facilitating a hassle-free integration of Process UI by adding the required components into your consuming application's code, without employing iFrames.

- Download all the required dependent libraries and CSS files to the client location and reference them using a local path to prevent unexpected interruptions while loading the client interfaces. For example, download and reference `oj-alta-min.css` locally in the consuming HTMLs, instead of including it as follows:

```
<link rel="stylesheet" href="https://<pcsserverhost:port>/ic/pub/
components/css/libs/oj/v2.0.0/alta/oj-alta-min.css" type="text/css"/>
```

- In addition to the standard dependent libraries, load the JS configuration files of the corresponding embeddable component in the consuming HTML. For example, you'll require the following code to load the Start Form widget:

```
<script data-main="<local_host>/main/startform-main "
src="<local_host>/js/libs/require/require.js"></script>
```

- Configure the client applications to consume the component-load events that are fired by an embeddable component. For instance, an embeddable component can open up further pop-ups to resize or repaint itself.
- Avoid using iFrames to consume UI embeddable components (snippets). Use them only as a last resort.
- Use the minified JS and CSS files for better performance. Minified versions of components are available in the `min` folder of the downloaded code.
- If you need to use Process components as part of Sites in Oracle Content Management, use the out-of-the-box (OOTB) components to avoid integration issues. The OOTB Process UI components are available in Oracle Content Management.
- Before you use a newer version of UI embeddable components, always port the customizations performed on the previous version to avoid mismatches in the UI rendering.
- You can use standalone widgets, such as Comments, Attachments, and so on, in similar ways to other jQuery widgets for your business requirements, without having to connect to the Process server. The widgets fire events—with all the required data—whenever a comment or attachment is added or updated, so that consuming applications can act on these events.
- Optionally, retrieve the Process test data corresponding to applications, start forms, and task details by adding the `testMode` parameter to the default URL. The following example fetches the applications that are deployed to the test platform:

```
IS_SERVER_UR/ic/pub/components/pages/applist.html?testMode=true
```

- Here are working samples for embedding a task list and start application.

– **Task List:**

```
<html>
<head lang="en">
  <meta charset="UTF-8">
  <title>TaskList</title>
  <!-- This is the main css file for the default Alta theme -->
  <link rel="stylesheet" href="https://<local_host>/ic/pub/
components/css/libs/oj/v2.0.0/alta/oj-alta-min.css" type="text/
css"/>
  <!-- This is where you add application-specific styling -->
  <link rel="stylesheet" href="https://<local_host>/ic/pub/
components/css/libs/pcs/v1.3/alta/pcs-tasklist-min.css"
type="text/css"/>
  <!-- RequireJS configuration file -->
  <script data-main="https://<local_host>/main/tasklist-main"
src="https://<local_host>/ic/pub/components/js/libs/require/
require.js"></script>
<script>
requirejs.config({
```

```

baseUrl: "https://<local_host>/ic/pub/components/js",
paths:
{
'knockout': 'libs/knockout/knockout-3.4.0',
'jquery': 'libs/jquery/jquery-2.1.3.min',
'jqueryui-amd': 'libs/jquery/jqueryui-amd-1.11.4.min',
'promise': 'libs/es6-promise/promise-1.0.0.min',
'hammerjs': 'libs/hammer/hammer-2.0.4.min',
'underscore' : 'libs/underscore/underscore-1.8.3.min',
'ojdnd': 'libs/dnd-polyfill/dnd-polyfill-1.0.0.min',
'ojs': 'libs/oj/v2.0.0/min',
'ojL10n': 'libs/oj/v2.0.0/ojL10n',
'ojtranslations': 'libs/oj/v2.0.0/resources',
'signals': 'libs/js-signals/signals.min',
'text': 'libs/require/text',
'pcsMsg' : 'libs/pcs/v1.3/resources',
'pcs' : 'libs/pcs/v1.3/min',
'rendererMsg': 'libs/pcs/v1.3/rendererMsg'
},
shim: {
'jquery': {
exports: ['jQuery', '$']
}
}
});
require(['jquery', 'pcs/pcs.tasklist' ],
function ($) {
//var tok = 'cloud.admin' + ':' + 'StacKed@7SUIte';
$.pcsConnection = {
serverURL: 'https://<local_host>',
authInfo: 'Basic c2VyZW51LnRhbkbVcmFjbGUuY29tOk9uQ2U3c2F0MSE=',
};
//$.pcsConnection.authInfo = 'Basic ' + btoa(tok);
var tasklist = $('#tasklist');
$("#tasklist").tasklist({
"hideToolbar": false,
"hideFilter": false,
"hideSystemActions": false,
"hideCustomActions": false,
"hideSearch": false,
"hideSort": false,
"hideRefresh": false,
"hideFromUserName": false,
"hideAssignedDate": false,
"hideDueDate": false,
"hideCreatedDate": false,
"hideSummary": false,
"hideTaskDetails": false,
"pageSize": 10,
"hideSelectAll": true,
"selectedSortType": "assignedDate",
"selectedSortOrder": "ascending",
"taskDetailOptions": "{ \"attachmentsOptions\":
{ \"showDocsInline\": true } }",
"systemActions":

```

```

"SUSPEND,ESCALATE,RENEW,REASSIGN,INFO_REQUEST,WITHDRAW,ACQUIRE,PU
RGE,DELETE,RESUME,RELEASE",
"tasklistFilter": "{}"
});
// Defining the event listeners --
tasklist.on('tasklist:taskSelect', function(event, data){
});
tasklist.on('tasklist:taskCheck', function(event, data,instance){
});
}
);
</script>
</head>
<body style="overflow-y:scroll">
<div id="tasklist"></div>
</body>
</html>

```

– Start Application:

```

<html>
<head lang="en">
  <meta charset="UTF-8">
  <title>Start Application</title>
  <!-- This is the main css file for the default Alta theme -->
  <link rel="stylesheet" href="https://<local_host>/ic/pub/
components/css/libs/oj/v2.0.0/alta/oj-alta-min.css" type="text/
css"/>
  <!-- This is where you add application-specific styling -->
  <link rel="stylesheet" href="https://<local_host>/ic/pub/
components/css/libs/pcs/v1.3/alta/pcs-applist-min.css"
type="text/css"/>
  <!-- RequireJS configuration file -->
  <script data-main="https://<local_host>/main/applist-main"
src="https://<local_host>/ic/pub/components/js/libs/require/
require.js"></script>
<script>
requirejs.config({
baseUrl: "https://<local_host>/ic/pub/components/js",
paths:
{
'knockout': 'libs/knockout/knockout-3.4.0',
'jquery': 'libs/jquery/jquery-2.1.3.min',
'jqueryui-amd': 'libs/jquery/jqueryui-amd-1.11.4.min',
'promise': 'libs/es6-promise/promise-1.0.0.min',
'hammerjs': 'libs/hammer/hammer-2.0.4.min',
'underscore' : 'libs/underscore/underscore-1.8.3.min',
'ojdnd': 'libs/dnd-polyfill/dnd-polyfill-1.0.0.min',
'ojs': 'libs/oj/v2.0.0/min',
'ojL10n': 'libs/oj/v2.0.0/ojL10n',
'ojtranslations': 'libs/oj/v2.0.0/resources',
'signals': 'libs/js-signals/signals.min',
'text': 'libs/require/text',
'pcsMsg' : 'libs/pcs/v1.3/resources',
'pcs' : 'libs/pcs/v1.3/min',

```

```
'rendererMsg': 'libs/pcs/v1.3/rendererMsg'
},
shim: {
  'jquery': {
    exports: ['jQuery', '$']
  }
});
require(['ojs/ojcore', 'ojs/ojmodule', 'pcs/pcs.applist'],
function (oj,$) {
  oj.ModuleBinding.defaults.modelPath = './';
  oj.ModuleBinding.defaults.viewPath = 'text!./';
});
require(['jquery', 'pcs/pcs.applist' ],
function ($) {
  //var tok = 'cloud.admin' + ':' + 'StacKed@7Suite';
  $.pcsConnection = {
    serverURL: 'https://<local_host>',
    authInfo: 'Basic c2VyZW51LnRhbkbVcmFjbGUuY29tOk9uQ2U3c2F0MSE=',
  };
  //$.pcsConnection.authInfo = 'Basic ' + btoa(tok);
  var appList = $('#applist');
  $("#applist").applist({
    "hideStartform": false,
    "startformDialog": false,
    "hideSubmit": false,
    "hideSave": false,
    "hideDiscard": false,
    "submitLabel": "Submit",
    "hideAttachment": false,
    "hideSearchBox": true,
    "hideDefaultCheck": true,
    "iconSize": "small",
    "hideEmptyText": false,
    "hideToolbar": false,
    "hideRefresh": false,
    "defaultVersion": false
  });
  // Defining the event listeners --
  appList.on('applist:formSelected', function(event, data){
  });
  appList.on('applist:formSubmitted', function(event, data,instance){
  });
  appList.on('applist:formSaved', function(event, data ,instance){
  });
}
);
</script>
</head>
<body>
<div id="applist"></div>
</body>
</html>
```

Use Process UI Composite Components (CCA)

Process UI components are also available as Oracle JET (JavaScript Extension Toolkit) composite components, which you can easily consume in multiple JET-based applications. A JET composite component, similar to a standard web component, allows you to define and use custom DOM (Document Object Model) elements.

Oracle JET composite components for Process UI snippets are available in Version 3.0 or later of the Process UI Library (`host:port/ic/pub/components`). For information on extending Process UI Components to JET-Based applications, including examples using composite components to external applications or Visual Builder, see our blog:

[Oracle Integration blog](#)

Work with Insight Models

Bring real-time visibility and analytics into your process applications by linking to an activated Insight model in a structured process.

How Insight models work in process applications

As an integral feature of Oracle Integration, Insight simplifies modeling and extracting meaningful business metrics. For background about Insight and its capabilities, see Introduction to Integration Insight in *Using Integration Insight in Oracle Integration Generation 2*.

After you link an Insight model to a process application, you can drag the corresponding Insight element in the structured process editor to key points in the process application flow. For each element inserted into the process application, you select the pertinent milestone at that point, define an identifier to correlate the activities for each instance of the process, and define the data to extract. At runtime, you can easily monitor and analyze your business processes in real time using Insight dashboards that reflect the data you choose to extract, and react quickly to business demands and problems. For example, dashboards can generate graphical visualizations of how many orders have been received, how many had discounts approved or rejected, the details about a single order, or where failures occur in a business transaction.

Key steps

1. In Insight, configure and activate a model.
2. In Processes, link the Insight model to a process application.
3. In the structured process editor, drag the corresponding Insight element to points in the process application where you want to extract data for analysis in Insight dashboards.
4. Define the properties for each Insight element, selecting the milestone that maps to the activity in the process flow at that point.
5. Define the data association for each Insight element, specifying a correlation identifier and a unique instance identifier that is guaranteed to be non-null to track the activities for each instance of the business process (which may span multiple processes and integrations), and the data to extract from the process application

for the unique instance identifier and the indicators that are associated with the milestone selected in the Insight element. See details in Step 4 below.

6. Activate the process application, run process instances, and analyze the results in Insight dashboards.

Important points

- An Insight model must be *activated* to be available for linking it to a process application.
- Insight models can be used in *structured processes*. They are not supported in dynamic processes.
- In the structured process editor, use data association to map *input* from the process to define extraction criteria for a model's unique instance identifier and indicators. Do not map output from the process.
- This topic addresses the use case of mapping values from a process application to an Insight model. You can also implement a use case for a business process implementation that spans one or more integration flows (in Integrations) and process applications, and then maps to an Insight model. For information, see Associate a Model to a Business Process Implementation in *Using Integration Insight in Oracle Integration Generation 2*.

When you map a model's milestones to an *integration*, the mapping details are reflected in the model definition in Insight. However, when you link a model to a *process application* and select model milestones at points in the process application flow, the model definition does not reflect the milestone, identifier, and indicator associations in Insight. They are shown only in the process application flow in the structured process editor.

Detailed steps

To configure an Insight model in a process application:

1. In Insight, configure and activate a model.
In the Oracle Integration navigation pane, click **Insight**, then **Models**. See Create a Model in *Using Integration Insight in Oracle Integration Generation 2*.
2. In Processes, link the model to the process application.
 - a. In the Oracle Integration navigation pane, click **Processes**, then **Process Applications**.
 - b. Create a new process application or click the name of an existing process application to which you want to link an Insight model.
 - c. In the Processes navigation pane, click **Insight**.
If you do not already have a linked model, the Link to Insight page is displayed.



Link to Integration Insight

Use Insight models in process applications for monitoring, analysis, and reporting.

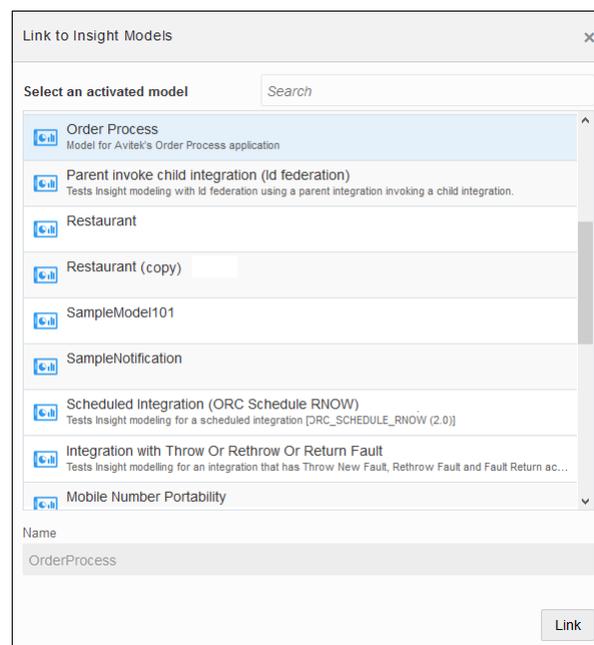
[Link](#)

[Learn more](#)

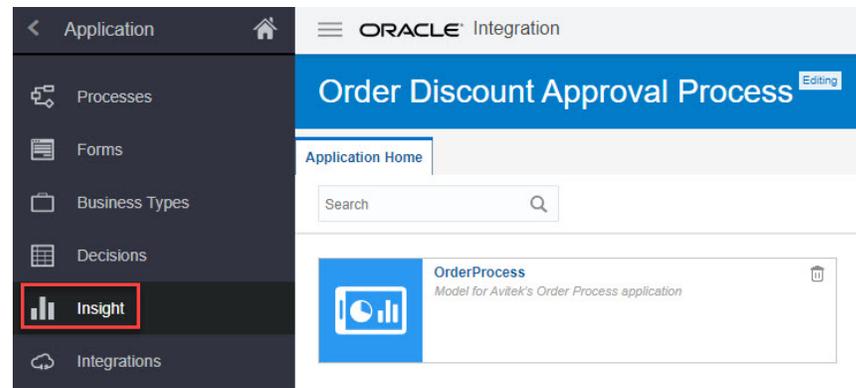
- d. Click **Link**. If you already have a linked model and want to link another, click **Create**, then **Link an Insight Model**.

The Link to Insight Models dialog is displayed, listing activated Insight models available for linking.

- e. Select an activated model to link to your process application, and click **Link**. You can link multiple models to a process application.



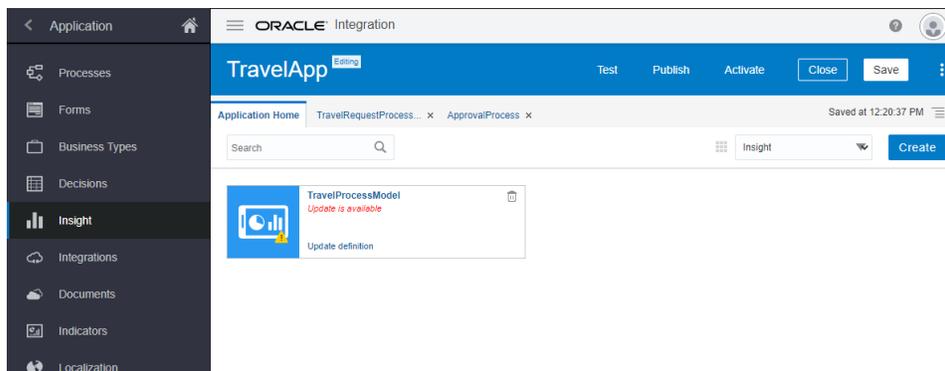
After you link one or more models to the process application, they're listed when you select **Insight** in the Processes navigation pane.



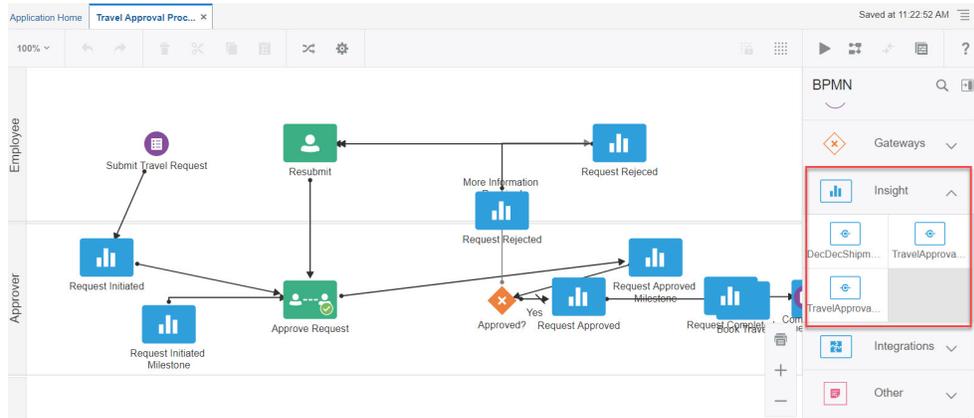
If you make changes to a model and reactivate it, the linked Insight model shows that an update is available.

 **Note:**

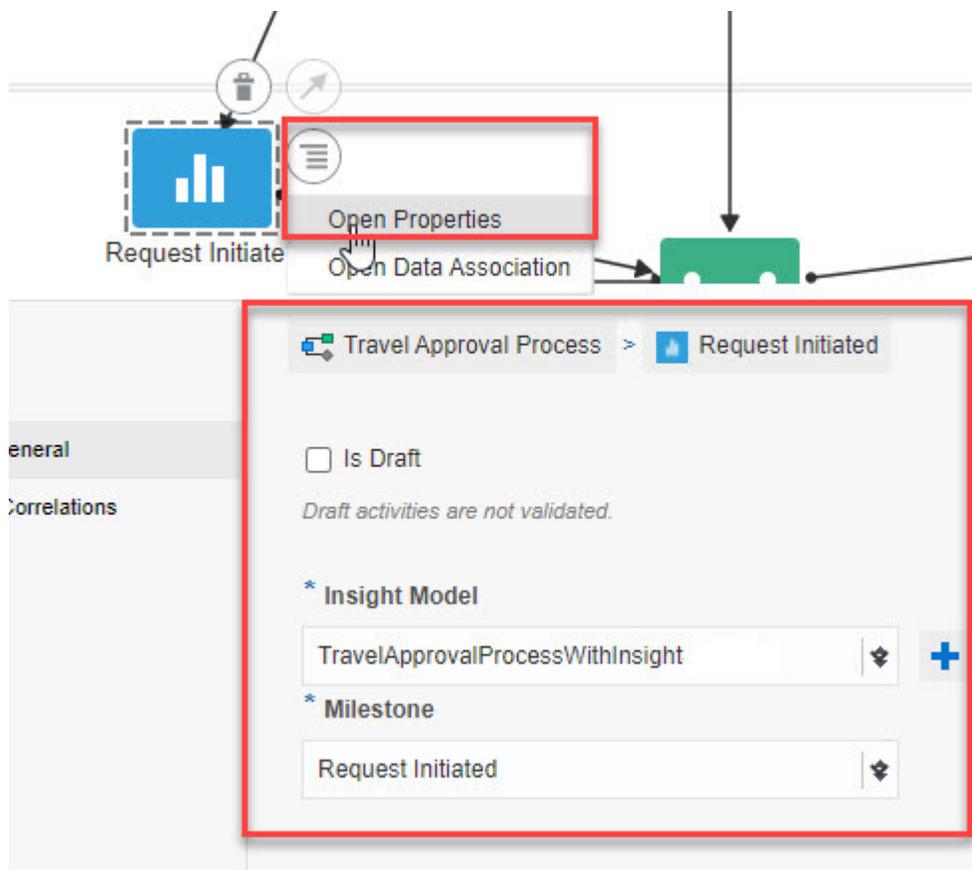
If an Insight model is updated, you may need to adjust the properties or data association based on the model's changes. It is important to confirm that the configuration of the model in the process application is valid based on the latest version of the model, and make necessary adjustments to avoid potential errors and loss of data.



3. In the structured process editor, place the corresponding Insight element on the flow:
 - a. In the Processes navigation pane, click **Processes**. Open or create the process application to which the required Insight model is linked.
 - b. In the structured editor's BPMN palette, notice an **Insight** category is listed after linking an Insight model. Click to expand it, and you'll see an Insight element corresponding to each model linked to the process application.
 - c. Drag the Insight element to a point on the process flow where you want to extract data for analysis in Insight dashboards.
 - d. Double-click the default name of the Insight element to rename it. This is highly recommended to reflect the milestone to which each Insight element is mapped.
 - e. Repeat dragging the Insight element to all points on the process flow where you want to extract data.



4. For each Insight element on the process flow, set the properties and data association (the data you want to extract for the metrics you want to analyze):
 - a. Click the Insight element, then click  and select **Open Properties** to open the properties pane.



In the properties pane:

- Ignore the **Is Draft** checkbox. It does not apply to Insight models.

- From the **Insight Model** list, leave the selection as the current model, select a different linked model, or click **Link an Insight Model**



to link to a new model if required.

- From the **Milestone** list, select the milestone applicable to the position of the Insight element in the flow.

 **Note:**

For any business process implementation, the Initial milestone must be assigned to the first activity, and the Terminal milestone must be assigned to the last activity. If a business process is implemented across multiple sources (for example, the initial milestone is in one process application and the terminal milestone is in another, or milestones span between an integration and a process), the model's Initial and Terminal milestones must be assigned accordingly to map to the first and last activity in the business process. For more information, see Milestones in *Using Integration Insight in Oracle Integration Generation 2*.

If you change the milestone selection from a previous selection, be sure to also redefine the data association pertinent to the new milestone.

- Click  below the BPMN pane to close the properties pane.
- b. Click the Insight element, then click  and select **Open Data Association** to open the Data Association editor.
- In the Data Objects pane, expand the process application's data objects and predefined variables. If your process uses a form start, for example, the form's data objects are listed. You'll pass values *from* these objects.

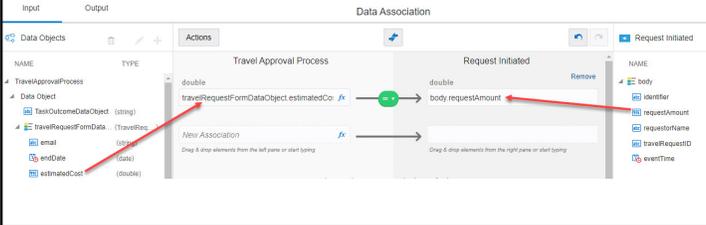
| Input | Output |
|--|--------|
| <div style="border: 1px solid #ccc; padding: 5px;"> Data Objects </div> | |
| NAME | TYPE |
| <ul style="list-style-type: none"> ▲ TravelApprovalProcess <ul style="list-style-type: none"> ▲ Data Object <ul style="list-style-type: none"> TaskOutcomeDataObject (string) ▲ travelRequestFormData... (TravelReq...) <ul style="list-style-type: none"> email (string) endDate (date) estimatedCost (double) firstName (string) justification (string) lastName (string) startDate (date) ▲ Predefined Variables <ul style="list-style-type: none"> creationDate (dateTime) instanceId (string) instanceNumber (long) | |

- In the element pane on the right, expand the Insight element's data objects, which are those data objects that pertain to the selected milestone. You'll pass values to these objects.

Request Initiated

| NAME | TYPE |
|-----------------|-------------------|
| body | (RequestInit_...) |
| identifier | (string) |
| requestAmount | (double) |
| requestorName | (string) |
| travelRequestID | (string) |
| eventTime | (dateTime) |

| | |
|------------|--|
| identifier | <p>A value that correlates the various milestones of the same instance (business transaction) of a business process together to allow Insight to track each instance throughout the business process implementation.</p> <p>If the business process is implemented in <i>only Processes</i> (one or more process applications), associate this value with a process variable that is unique and guaranteed to be non-null for every instance of the business process (for example, <code>instanceID</code> when implemented in a single process application, or a value that identifies the same instance when implemented in multiple process applications).</p> <p>If the business process is implemented in <i>both Integrations and Processes</i> (for example, an integration calls a process application), you must set the <code>identifier</code> value using a special syntax:</p> <pre>"model-id:identifier-id:" + identifier-value</pre> <p>where:</p> <ul style="list-style-type: none"> <code>model-id</code> and <code>identifier-id</code> are the model ID and unique instance identifier ID, static values specified in the Integration Insight console manifest <code>identifier-value</code> is the unique instance identifier for the model, a dynamic value that resolves to a unique value for each business transaction <p>For example:</p> <pre>"Orders_PKdkqDWS:OrderNumber_dQRnmib1:" + orderNumber</pre> <p>At runtime, this expression resolves to a unique value for each instance. For example:</p> <pre>Orders_PKdkqDWS:OrderNumber_dQRnmib1:100201</pre> |
|------------|--|

| | |
|-----------------------------------|--|
| <p>unique instance identifier</p> | <p>The unique instance identifier defined by the model (for example, <code>travelRequestID</code> in the example screenshot above). Associate this value with a process variable that is unique and guaranteed to be non-null for every instance of the business process (for example, <code>instanceID</code> when implemented in a single process application, or a value that identifies the same instance when implemented in multiple process applications). This value is extracted at runtime every time the associated milestone is passed and correlates the actions and data that belong to the same instance (business transaction) of the business process. For more information, see Unique Instance Identifier in <i>Using Integration Insight in Oracle Integration Generation 2</i>.</p> |
| <p>indicators</p> | <p>The indicators (dimensions and measures) that are associated with the milestone selected in the properties pane. Associate these values with the business process data objects that extract the expected data. For example, drag and drop to associate the data object <code>estimatedCost</code> with indicator <code>requestAmount</code>. For more information, see Indicators in <i>Using Integration Insight in Oracle Integration Generation 2</i>.</p>  |
| <p>eventTime</p> | <p>Ignore this object. It is automatically set by Processes.</p> |

Note that the **Output** tab in the Data Association editor does not apply to Insight models.

5. Validate, save, publish, and activate the process application. Add users to interact with the activated process and run process instances.

In the Processes navigation pane, click **My Apps**, select the process application, provide any required information (for example, fill in a form), and click **Submit**.

As the process application progresses, you can immediately begin to monitor your business process using Insight dashboards, either in Oracle Integration or in an external application where the dashboards are embedded. See Work with Consoles and Dashboards in Integration Insight and Embed Insight Dashboards in Other Applications in *Using Integration Insight in Oracle Integration Generation 2*.

Integrate with Robotic Process Automation Tools

You can use robotic process automation (RPA) tools with Process to extend workflow automation to applications that don't have an adapter in Oracle Integration and don't provide APIs for integration. RPA tools automate tasks on such an application using the application's graphical user interface (GUI).

An RPA tool can record a sequence of user actions, usually repetitive in nature, performed on an external application's GUI and mimic these actions using software robots.

Oracle Integration provides adapters for leading RPA providers. Use these adapters to create an integration to the RPA provider and call it within your process through a bot activity. See [Work with Bot Activities](#).

Choose from these providers to extend and accelerate your business workflow automation:

- **UiPath:** See Accelerate Process Automation in *Using the UiPath Robotic Process Automation Adapter with Oracle Integration Generation 2*.
- **Automation Anywhere:** See Accelerate Process Automation in *Using the Automation Anywhere Adapter with Oracle Integration Generation 2*.

Integrate an External UI with the Process Task List

If you've configured a human task to use an external UI instead of a web form, the external form is displayed when you select the task in the runtime task list.

If an external form is used for a human task, then the associated external application must provide the action buttons for the task. In addition, when an action is performed within the task, the external UI must make the corresponding REST API call to the Process server and also fire an appropriate JavaScript event to update the runtime task list. The process task list listens to these events and refreshes accordingly.

The following table lists all the JavaScript events an external UI must implement for successful integration with the process task list:

| Event | Description | Example Syntax |
|--------|--|--|
| loaded | The external form sends a loaded event, so that the parent can calculate formHeight. | <code>{source:externalForm, event:loaded, data: {formHeight : 600,showCloseIcon : true,showResizeIcon : true} }</code> |
| reload | The external form requests to reload itself, without refreshing the task list. | <code>{source:externalForm, event:reload, data: FUTURE}</code> |
| close | The external form requests to close itself, without refreshing task the list. | <code>{source:externalForm, event:close, data: FUTURE}</code> |
| done | The external form requests to close itself after an action, and it also requests to refresh the task list. | <code>{source:externalForm, event:done, data: {actionId : ACTION_ID}}</code> |

To configure a human task with an external UI, see:

- [Use an External UI to Display Task Information](#) for structured processes.
- [Associate an External UI](#) for dynamic processes.

15

Troubleshoot Application Development

Here are common issues you might encounter using Process and possible solutions.

Topics:

- [Troubleshoot Application Import](#)
- [Troubleshoot Business Processes](#)
- [Troubleshoot Data Association](#)
- [Troubleshoot Application Playing](#)

Troubleshoot Application Import

The following troubleshooting tips pertain to importing applications.

When I import an application, I receive an error stating that an application with the same name already exists.

Application names must be unique, even if they have different owners or are in different spaces. For example, if two users try to upload one of the QuickStart Apps, the second user will be unsuccessful.

To solve the problem, rename the application:

1. Unzip the `.exp` file.
2. Rename the resulting root directory.
3. Zip the renamed root directory to create a new `.zip` file.
4. Change the `.zip` extension to `.exp`.

When I import a renamed application, I receive an error stating that the project Info file is invalid.

Some unzipping commands put the root directory under a new directory with the same name. Then when you zip the new directory thinking it's the root directory, the file structure is not recognized. This structure is incorrect:

```
Application Name (added)
  Application Name (root)
    SOA
      (other subdirectories)
```

To solve the problem, make sure the root directory that gets zipped contains only one subdirectory, named SOA. This structure is correct:

```
Application Name (root)
  SOA
  (other subdirectories)
```

Troubleshoot Business Processes

The following troubleshooting tips pertain to creating and editing business processes.

Are there restrictions on loops or activities in processes?

To prevent a process executing a very large number of loops or activities as part of a single process instance, the number of direct/indirect loops is restricted to 1000, at which point execution is suspended. This restriction prevents audit data growth and releases system resources.

How are repeated exceptions handled?

To ensure that a process instance does not result in repeated exceptions, especially when invoking an external service, auto-recovery on faulted process instances is limited to two times.

I'm receiving a validation error stating that the condition in a sequence flow is empty.

You created a gateway element, created a data object for it, and connected its outbound paths, but you didn't implement the non-default paths.

To solve the problem, perform these steps for each non-default path:

1. Click the path, click the edit icon on the path, and click **Implementation**.
2. Type a name for the path.
3. Click **Edit** next to the Condition field.

The Expression Editor appears.

4. Type an expression that tests whether the gateway data object has one of the outcome values.

For example, if the gateway handles the outcome of an approval task, the name of the gateway data object is `approvalOutcome`, and the APPROVE outcome is the default path, type `approvalOutcome == "REJECT"`. Note the double equal sign.

5. Click **OK** to close the Expression Editor and then close the implementation pane.

You don't need to implement the default gateway path.

I created two business objects based on two schema definitions that have unique element names but the same namespace. I'm receiving exception errors when I use these business objects in my process.

Schema type definitions on the same `targetNamespace` from two different locations can't be resolved properly. Any attempt to use these schemas such as, when creating a business object or as part of a WebService definition, causes errors throughout the

application. Therefore, you should either provide another namespace or define those schema types in the same file.

Troubleshoot Data Association

The following troubleshooting tips pertain to performing data association.

When I click Apply in the data association editor, I receive an error that validation of associations failed.

The output data type from the previous process flow element doesn't match the input data type that the current process flow element expects.

To solve the problem, first make sure you dragged and dropped the correct data object. If the data object is correct, add a function that converts the data type:

1. In the data association editor, click the Launch Expression Builder icon next to the input field.

The Expression Editor opens.

2. Surround the data object name with a function named for the type that the current process flow element expects.

For example, if the field name is `inputDataObject`, and the type must be string, type `string(inputDataObject)` in the Expression Editor.

3. Click **OK** to close the Expression Editor and then click **Apply**.

When I click the Validate Application icon, I receive an error stating that data association isn't valid because double or float can't be assigned to int.

A field with a data type of int can only contain integers, or whole numbers. A field of type double or float can contain decimal numbers.

In data association, if the input type is int, the output type can be any numeric data type, such as int, double, or float. However, if the output type is int, the input type must also be int, or an error occurs.

You can solve the problem in one of these ways:

- Change the data type of the input field to int.
- Change the data type of the output field to double or float.
- Use an expression to convert the input field type to int.

To convert the input field type:

1. In the data association editor, click the Launch Expression Builder icon next to the input field.

The Expression Editor opens.

2. Surround the field name with the `round()` and `int()` functions.

For example, if the field name is `loanAppDataObject.form.income`, change it to `int(round(loanAppDataObject.form.income))`.

3. Click **Validate** to verify the expression and then click **OK**.

The Expression Editor closes.

For example, if you're performing data association for a Decision and the inputs are from a form, you can solve the problem in one of these ways:

- In the form editor, use a Quantity field, which has a data type of int.
- Recreate the Decision and make sure the Fact Type is double or float.
- Use an expression to convert the input field type to int as described previously.

Troubleshoot Application Playing

The following troubleshooting tips pertain to using the application player.

I'm receiving an error stating that there was a problem activating the application and the application player can't be initialized.

The instructions to enable the player were followed, but the credentials entered were incorrect. Click **Clear** and try again. See [Enable the Application Player](#).

I'm still receiving the message that *Credentials aren't configured* after the administrator entered them.

If your application was opened before the administrator entered the credentials, you must close and reopen your application to enable the application player for that application.

Part III

Administrator Tasks

Topics:

- [Administrator Basics](#)
- [Manage the Runtime Environment](#)
- [Manage the Design-Time Environment](#)

16

Administrator Basics

Users assigned an administrator role can perform a variety of configuration, management, and monitoring tasks in Design Time and Runtime.

Topics:

- [About Administrator Privileges](#)

About Administrator Privileges

If you're assigned to the Administrator role in Process, you have access to the Administration pages in design time and runtime.

- **Runtime administration:** On the Home page, click **My Tasks**, then **Workspace** from the top bar. Select **Administration** from the navigation pane to assign roles to users, configure notification logs, configure Oracle Content Management, and much more.
- **Design-time administration:** On the Home page, click **Processes**, then **Administration** from the navigation pane to manage spaces, applications, QuickStart applications, and the application player.

To activate applications to your test and production environments, click **Processes**, then **Process Applications**. Select **Activate** from the top bar.

17

Manage the Runtime Environment

As an administrator, you manage, monitor, and configure the runtime environment.



Note:

The timeout limit is 60 minutes on inactivity from any page in the runtime environment.

Topics:

- [Task Overview for Administering the Runtime Environment](#)
- [Assign and Manage Roles](#)
- [Monitor and Adjust Processes](#)
- [Configure Application Settings](#)
- [Troubleshoot the Runtime Administration](#)

Task Overview for Administering the Runtime Environment

As an administrator, you're responsible for tracking process instances, monitoring dashboards, and configuring application settings.

| Your Responsibilities | Description |
|--------------------------------|--|
| Alerts | View alerts for suspended, recoverable, and alerted instances on the Home page, and navigate instance details. |
| Instances | Track running, inactive, and problematic process instances. |
| Dashboards | Monitor key process metrics such as workload, cycle time, running process instances, and closed instances. |
| Administration | Assign users to roles, configure Oracle Content Management, configure and manage notification logs, and much more. |

Assign and Manage Roles

To activate an application to end users, locate the application's roles and assign users, groups, or roles to them. Roles assigned to end users define their permissions, such as task assignment and whether they can start an application.

By default, roles get created for each swimlane defined in an application's processes. In addition, several key application roles are defined for you, as described in the following table. Want to learn more about defining roles? See [Define Application Roles](#).

| Role | Description |
|------------------|--|
| Process Owner | Users assigned this role can view process activity history, take actions (such as approve or reject), alter process flow, and view form data for applications they own. Process owners typically manage activated business processes and use metric analysis tools such as dashboards to monitor business processes and alter task flow as needed. |
| Process Reviewer | Users assigned this role can view tracking and process activity history and view or add comments, attachments, or documents for the specified application. Process reviewers cannot take actions on tasks or alter task flow. They aren't process participants, but typically responsible for reporting on current process instance status, such as in a help desk or front office role. |
| Analytics Viewer | Users assigned this role can create and view business analytics dashboards associated with the specified application. (Analytics viewers can't update tasks.) See Create and View Business Analytics Dashboards . |

To assign or manage roles:

1. In the Oracle Integration navigation pane, click **My Tasks**, click **Workspace**, and then click **Administration**.
2. Click **Manage Roles**.

The Manage Roles page displays. Below **Manage Roles**, all roles defined in swimlanes for all applications are listed. In addition, several special roles (Process Owner, Process Reviewer, and Analytics Viewer) are automatically created upon application activation and available for assignment. Each role includes the application name followed by its role, and lists its members below. For example, Application1.Process Owner refers to the Process Owner role in Application1. If needed, enter an application's name in the **Search** field to limit the display to that application's roles.

If needed, add or delete roles. You can add and delete roles to a selected application, as described in [Define Application Roles](#).

3. Select the role to which you want to assign users.
4. In the **Assign Role** section, click **Add Member**. In the Search users, groups, roles window, search for and select a user, group, or role, using one of the search methods listed. You can use an * asterisk as a wildcard character. Select users from the results, and click **OK**. The selected user, group, or role is now listed in the **Assign Role** table.
5. Click **Save** to save the role assignment. If needed, click **Revert** to discard unsaved changes.
6. Optionally, search for and add a member (user, group, or role) to the **Escalation Path** field to whom to escalate the task.
7. Assign the Process Owner or Process Reviewer roles to users who need enhanced visibility into the task history of process instances.

Process Owners can see the tasks that they own in the process by selecting **Administered by Me** under **My Tasks**. Process Reviewers can see the tasks that they review in the process by selecting **Reviewed by Me** under **My Tasks**.

When users assigned either of these key roles open a task and view the task's history, they see an extra drop-down field with two options:

- **Task View** (default) shows the current task's history.

- **Full View** shows the history of all tasks in the process. Process Owners can see details of the current task and details of all the completed tasks in the process. Process Reviewers can see details of the current task only. They can't see details of all the completed tasks in the process.

Let's take the example of a simple hiring process with these key swimlane roles, among others:

- *Candidate*, who starts the process instance by submitting a job application form
- *Department Manager*, who makes the final decision based on others' feedback
- *Human Resources Manager*, who prepares the candidate's job offer if approved

After the candidate submits the form, multiple people such as interviewers and hiring managers complete tasks in the process. They can each view history for their assigned task only.

Because the Department Manager is assigned the Process Owner role, and the Human Resources Manager is assigned the Process Reviewer role, they can use the **Full View** option to see the history details. Both the Department Manager and Human Resource Manager can see details of the current task. The Department Manager, who is assigned the Process Owner role, can also see the details of all the tasks that have been completed. This information gives them the oversight to make appropriate hiring decisions.

Monitor and Adjust Processes

Use these options to monitor overall process flow and make adjustments as needed.

Topics:

- [View Alerts](#)
- [Track Process Instances](#)
- [Alter the Flow of a Process Instance](#)
- [Monitor Key Metrics in Dashboards](#)
- [Create and View Business Analytics Dashboards](#)
- [View and Resend Email Notifications](#)

View Alerts

If you're an administrator, your Home page displays alerts with the number of suspended and recoverable instances, and alerted tasks. Click an alert to view details about the problematic instances or tasks. Select an instance or task and correct the problem.

Track Process Instances

Tracking a process instance enables you to identify any problems with the flow or the assignment that might be delaying the process. Depending on your role, you can locate and track the flow of a specific instance, and view the instance history, comments, attachments, details, and documents.

Users assigned the Process Owner and Process Reviewer roles can track the entire process. Want to learn more about these roles? See [Assign and Manage Roles](#).

To track process instances:

1. In the Oracle Integration navigation pane, click **My Tasks**, click **Workspace**, and then click **Processes**.
2. Click the instance you want to track.
3. In the Instance Details pane, view the following:
 - **Open Activities**: displays the currently running activities in the process. At a certain point in the flow, the process might have multiple activities running at the same time.
 - **Comments**: displays the history of the comments for this process instance.
 - **Attachments**: displays the attachments for this process instance.
 - **History**: enables you to view the process flow this instance followed before getting to the current activity. You can view the history as a list, a tree, or a graphic.
 - **Details**: displays detailed information about the instance such as the priority, the status and the creation date.
 - **Documents**: displays the documents that are integral to the instance.

You can change the process flow of an instance that is currently suspended because of a problem, or move an instance that is running to another activity. You can also modify values that are causing an activity to fail and then retry running the current activity again. See [Altering the Flow of a Process Instance](#).

Alter the Flow of a Process Instance

You can change the process flow of an instance that is currently suspended because of a problem, or move an instance that is running to another activity because of a specific reason. If an activity is failing because of the value of the data objects and instance attributes, then you can modify them and retry running the current activity again.

To alter the flow of a process instance:

1. In the Oracle Integration navigation pane, click **My Tasks**, click **Workspace**, and then click **Processes**.

Alternatively, you can access suspended instances, recoverable instances, and alerted tasks from the alerts box on the home page. See [Viewing Alerts](#).
2. Click the instance you want to alter.
3. In the Instance Details pane, click **Alter Flow & action**.
 - If the status of the selected instance is *In Progress*, then the button is labeled **Alter Flow & Suspend**.
 - If the status of the selected instance is *Suspended*, then the button is labeled **Alter Flow & Resume**.
4. In the Open Activities table, go to the New Activity column and select an *activity* that redirects the process instance.
5. In the **Comments** field, enter a comment justifying your action or adding information.
6. Optionally, you can change the value of the data objects or instance attributes:
 - a. To view the process instance attributes, click **Show Instance Attributes**.

- b. To display the data object in a tree, click **Show as Tree** located below the Data Objects list.
 - c. Click the data object whose value you want to modify.
The XML that represents that data object appears to the right of the list of data objects.
 - d. Locate the value of the data object in the XML and edit it.
7. Click either **Resume** or **Suspend**. Depending on the current status of the process instance only one of these buttons appears.

The process instance moves to the selected activity. Suspended instances are resumed, and running instances are suspended.

Monitor Key Metrics in Dashboards

Use the dashboards to monitor the overall state of your processes and view specific process metrics such as bottleneck processes. You can also create custom graphs to view process data based on the business indicators defined in your business processes.

To monitor dashboards:

1. In the Oracle Integration navigation pane, click **My Tasks**, click **Workspace**, and then click **Dashboards**.



2. On the Process Monitoring tab, select each dashboard to view information about your processes.

The available dashboards are:

| Dashboard | Description |
|-----------|---|
| Health | Monitor the overall state of the processes available to you, and spot any anomalies. Shows the total number of instances that are in progress, suspended, and recoverable for each process. Click Total Open , Active , Recoverable , or Suspended to view the instances for the selected item or double-click a line in the chart to view the instances for that process. |

| Dashboard | Description |
|-----------|--|
| Open | Shows a top level view of the current running instances, and a detailed analysis of the running instances for each process. This analysis includes the number of instances on track, due this week, overdue, suspended, recoverable, created today and closed today. Click any of the rows to view the instances for that process. |
| Workload | Shows the workload for the top ten processes or bottleneck processes. View the workload for each of the processes by task or by assignee. |
| Trend | View the workload and cycle trend for each process, or the workload and cycle trend for each of the tasks in those processes. |
| Closed | View the closed instances for the current day, week, or month, that are successfully completed, aborted, and errored. Also see the average cycle time for the current period, the previous period, and the maximum cycle time for the current period. |

You can click **Filter** at any time to select specific processes to view for that dashboard.

These filters enable you to select which processes to display. For some types of dashboards, you can also select the assignees and time period to display.

Create and View Business Analytics Dashboards

Use the Business Data Query to plot and view charts and graphs for application metrics and create Business Analytics Dashboards. You can create charts that display business indicator values (metrics specific to a process) and system indicator values (metrics automatically captured).

When designing a process, developers create [business indicators](#) for data objects whose metrics they want to capture and display as X axis, Y axis, and filter values. You select the business or system indicators to plot them in charts and graphs. Users can create business analytics reports if they have the Process Owner role, the Administrator role, or the Analytics Viewer role on the corresponding application on which the report is being generated. Business analytics reports created by one user cannot be shared with other users even if they belong to the same role. See [Assign and Manage Roles](#).

To create and view business analytics charts and graphs:

1. In the Oracle Integration navigation pane, click **My Tasks**, click **Workspace**, and then click **Dashboards**.
2. Click the **Business Analytics** tab.

If you're creating a query for the first time, click **Get Started**.

3. In the Reports pane, click **New Query** .
4. Enter a name in the **Untitled** field.

You can also add a description for the new query in the **Add description** field.

5. In the **Data Source Type** field, select a source for the chart or graph. Typically, this value is left as Process, the default value.
6. In the **Application** field, select the application to report on.

You can select **All Applications** to report across multiple active applications, using system indicators only.

- In the **X Axis** options, scroll the **Series** and **Group** fields to see available indicators, and select dimension indicators to plot for the chart's X axis.

Business indicators are listed first, followed by standard system indicators. Want to learn more about dimension indicators? See [Add Dimension Business Indicators](#). You must specify both a series and a group indicator for the X axis.

 **Note:**

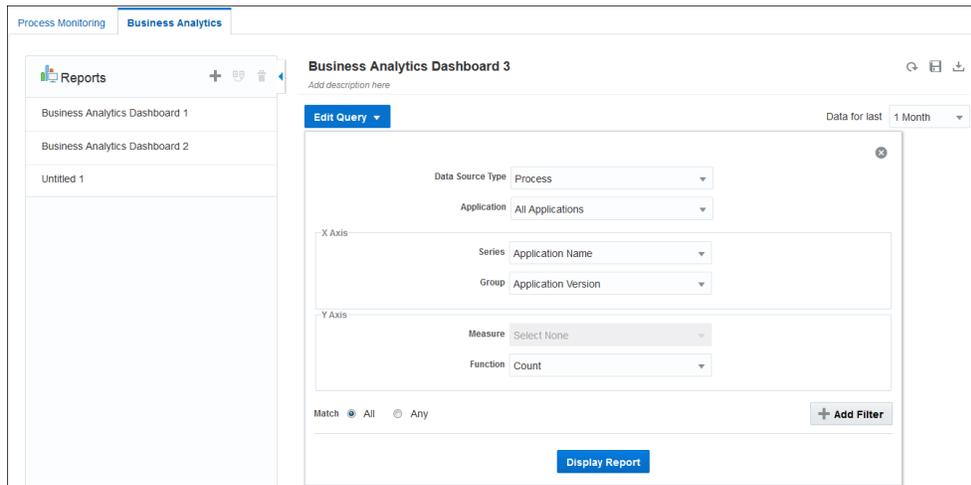
If you select the **Date** data type indicator for the X axis, then you can use the **Time Grouping** option to specify the time frame for the business analytics dashboards of the application. For example, in case of a Travel Request Application, if you select **Year**, you can view the number of Travel Requests by the year.

- In the **Y Axis** fields, scroll the **Measure** field to see available indicators, and select a measure indicator and standard function to plot for the Y axis.
See [Add Measure Business Indicators](#).
- Optionally, add one or more filters. Click **Add Filter**. From the fields that display, click the left one to see available indicators, and select an indicator by which to filter. For available system indicators, see [Select System Indicators](#).

Want to learn more about attributes? See [Add Attribute Business Indicators](#).

- Click **Display Report**.

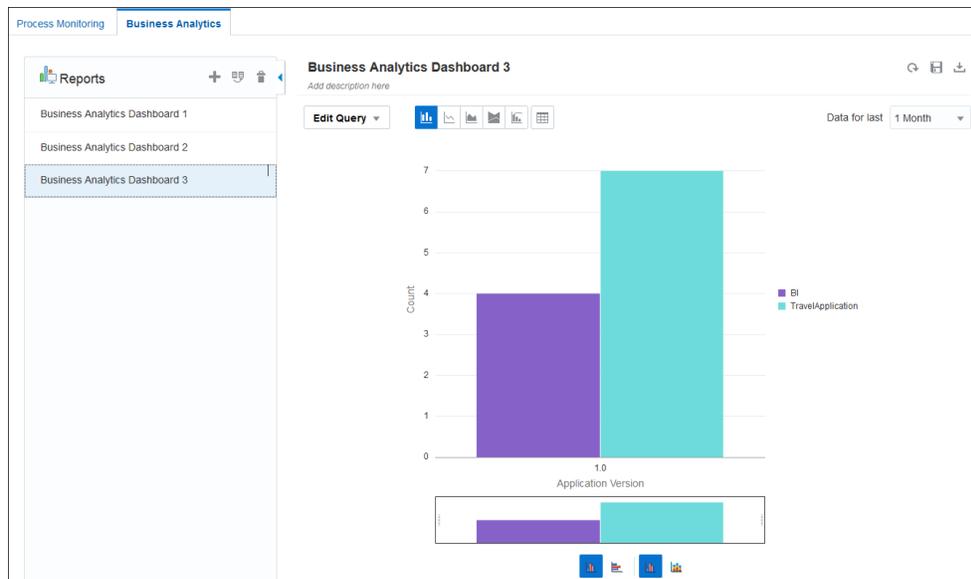
The chart is plotted and displayed on the right side of the page.



- Click **Save** to save the business analytics dashboard that you just created.
After you save the query, you can select and view it on the Reports pane.

 **Note:**

You can use the **Collapse** button to hide the Reports pane when not in use.



12. Optionally, change the chart or graph's display.
 - a. Use the **Data for last** drop down list to select the duration of the activity. By default, the data for last 1 month is selected.
 - b. Use the icons in the top and bottom to change the chart's type and orientation and stack.
13. You can **Delete** or **Copy** the Business Analytics dashboard or **Reset** the fields for **X axis** and **Y Axis** and create a new graph.
14. Optionally, export the data to a file, click **Download CSV**, and open or save the export.csv file.



Note:

Editing a dashboard's query and saving it results in overwriting of the selected dashboard or report. If you want to retain a previously generated dashboard and also create a new one with modifications to the earlier query, use **Copy Query** .

Select System Indicators

You can select the system indicators when you add filters to create business analytics charts and graphs.

The following are the valid filter values for some of the system indicator columns:

1. Is Recoverable
 - **Y**: Represents Yes
 - **N**: Represents No
2. All other system indicator columns starting with "Is ..."
 - **0**: Represents false

- **1**: Represents true
3. Due Status
 - DUE SOON
 - MISSED DUE
 - ON TRACK
 - OVERDUE
 4. Process Activation Status
 - **-1**: refers to un-activated status
 - **0**: refers to retired status
 - **1**: refers to activated status
 5. Process Instance Status
 - ABORTED
 - ACTIVE
 - COMPLETED
 - FAULTED
 - SUSPENDED
 6. Activity Instance Status
 - ABORTED
 - ACTIVE
 - COMPLETED
 - FAULTED
 - MOVED
 - SUSPENDED
 7. Assignment state
 - ACQUIRED
 - ASSIGNED
 - COMPLETED
 - ERRORED
 - EXPIRED
 - DELETED
 - WITHDRAWN
 - SUSPENDED
 8. Task State
 - ALERTED
 - ASSIGNED
 - COMPLETED
 - ERRORED

- EXPIRED
- DELETED
- INFO_REQUESTED
- WITHDRAWN
- SUSPENDED

View and Resend Email Notifications

You can view the notification logs to monitor the status of notifications sent for tasks and Business Activity Monitoring. You can try to resend the email notifications to all the original recipients or to some of them by using the notification logs if the email notifications fail due to lack of network availability, wrong email addresses, or temporarily unavailable email servers. By default, a resend is automatically attempted for all failed notifications after an interval of fifteen minutes.

To view the notification logs:

1. In the Integration Cloud navigation pane, click **My Tasks**, click **Workspace**, and then click **Administration**.
2. Click **Notification Logs**.

The Notifications table displays the following information:

- Process
- Source
- Recipients
- Sent On
- Status
- Actions

3. On the **Notification Logs** page, you can:

- Search for a specific notification.
- Sort the notifications by date or by status.
- View bad addresses.

Click the **View Bad Addresses** button. A dialog box with the list of emails with bad email addresses opens.

- Test notifications.

Click the **Test Notifications** button. A dialog box opens for you to send a notification email as a test.

- Fix notification problems.

Click the drop-down list on the actions column and select one of the following actions:

- Resend to all members
- Resend to specific members
- Delete

Configure Application Settings

You can configure the settings for all your applications.

In the Oracle Integration navigation pane, click **My Tasks**, click **Workspace**, and then click **Administration**.

| Option | Description |
|---|--|
| Oracle Content Management | Under Services, create a connection to Oracle Content Management so that end users can collaborate using documents. |
| Oracle Storage Service | Under Services, create a connection to Oracle Storage Service to store your archived data as objects that are stored in a container. |
| Notification Service | Under Services, configure e-mail (human task) notifications. |
| Runtime Settings | Under Runtime Settings, configure the process audit level and schedule a daily recovery for process instances that encountered a remote fault while invoking a web service or expired timer messages and edit the logger's settings. |
| Archive and Purge | Schedule instances archive or analytics archive to back up your data from one or more applications. |
| UI Customization | Update time zone settings. |
| Manage Roles | Assign roles to users and groups. If needed, create, modify, and remove roles. |
| Manage Credentials | Manage runtime keystore credentials for web and REST services. Upload, update, or delete credentials as needed. |
| Notification Log | View details and resend email notifications sent for human tasks. |

Configure Oracle Content Management

Before users can access the documents feature, an administrator must configure settings in both Oracle Content Management and Process.

- To read about the benefits of integrating with Oracle Content Management, see [Why should I integrate documents?](#)
- To learn more about how to configure and set up your services, see [How do I integrate with Oracle Content Management?](#)

Configure Audit and Log Levels

You can select the type of messages you want to store in the audit trail and schedule a daily auto recovery for process instances that encounter a remote fault while invoking a web service or expired timer messages. You can also recover the invoke or callback messages that were not delivered and resubmit them. You can use the logger setting to change the log levels of different loggers.

Configure the Process Audit Level

Auto recovery is performed once each 24 hours to recover any faulted (non-service task related faults) or stranded process instances. Note that auto recovery doesn't pick up process instances that fault on a service task and then wait for manual recovery on the same service task. By default, auto recovery starts at 00:00 hours and stops at 04:00 hours (server time

zone), or earlier if there are no instances pending recovery. By default, it recovers in batches of 50 instances.

To configure the process audit level:

1. In the Oracle Integration navigation pane, click **My Tasks**, click **Workspace**, and then click **Administration**.
2. Click **Runtime Settings**.
3. In the **Process Runtime** section, select an audit level from the **Audit Level** drop-down list.

Available audit levels are:

- **Production**: logs all events but doesn't log the values used when assigning input or output values to data objects.
- **Development**: log all events and also logs the values used when assigning input or output values to data objects.
- **Off**: doesn't log any events.

 **Note:**

If the audit level is **Production** or **Development**, the following activities will record the payload details:

- USER_TASK
- SERVICE_TASK
- RECEIVE_TASK
- SEND_TASK
- THROW_INTERMEDIATE_EVENT
- CATCH_INTERMEDIATE_EVENT
- BUSINESS_RULE_TASK
- START_EVENT END_EVENT

If the audit level is **Off**, then no audit information will be recorded.

4. Select a time to start and stop the scheduled recovery using the time editor.

The runtime environment uses the selected time and configuration to perform a daily scheduled recovery. It is recommended that you set the recovery window to off-peak hours.

5. Enter the maximum number of instances to recover.

It is recommended that you specify a small batch size (maximum number of instances to recover) such as 50.

6. Click **Save**.

Configure Logger Settings

As an administrator, you can use the **Logger Settings** section to change the log levels of different loggers and send an error report to Oracle if an error occurs. Logger levels

include **Incident_Error**, **Error (Severe)**, **Warning**, **Notification (Info)**, **Notification (Config)**, and **Trace**.

Configure Oracle Storage Service

Oracle Storage Service is an object store that is used to save objects that are identifiable by names within containers. You must request access to an Oracle Storage Service account before provisioning and then create a container to be able to export your data to Oracle Storage Service. After the account and container are created, use the Oracle Storage Service URL, container name, and login credentials to create a connection between the two services. Only when the connection has been made and tested successfully can you enable archiving.

To configure the Oracle Storage Service settings:

1. In the Oracle Integration navigation pane, click **My Tasks**, click **Workspace**, and then click **Administration**.
2. Click **Services**.
3. Click the **Infrastructure** tab.
4. In the **Oracle Storage Service** section, enter the following information:
 - **URL:** enter the Cloud Storage REST URL that was provided during the provisioning of the Oracle Storage Service. For example, `https://your_storage_service_name.com/myservice`.
 - **Container Name:** enter the container name that you created in Oracle Storage Service during provisioning.
5. Click **Test** to verify that the connection is successful.
6. Be sure to save your changes.

Note:

Do not include any slashes (/) when entering the container name.

- **User and Password:** the account credentials for a user who has access to your Oracle Storage Service.

Enable Email Notifications

You can configure Process to use emails for human workflow notifications that are sent to the task assignee when events such as assignment and reassignment occur.

Note:

Before enabling email notifications in runtime, make sure you customize the email notifications, for example, their content, template, attachments, and subject lines in design time. See [Customize Notification Emails for Human Tasks](#). After enabling email notifications, you can view the notification logs and resend emails to all or some of the original recipients. See [View and Resend Email Notifications](#).

For information about SPF and DKIM settings, see Configure Email Authentication Settings for SPF and DKIM in *Provisioning and Administering Oracle Integration Generation 2*.

To enable email notifications for the human tasks defined in your process applications:

1. In the Oracle Integration navigation pane, click **My Tasks**, click **Workspace**, and then click **Administration**.
2. Click **Services**.
3. Click **Infrastructure**.
4. In the **Oracle Notification Service** section, enter the following details:
 - In the **Notification Mode** field, select **Email**.
 - In the **From** field, enter the email address that will serve as the sender of all email notifications.
5. Click **Save**.
6. Click **Register**. Oracle will send an email to the address you specified in the **From** field. Follow the instructions in that email to complete the registration process.

Email Notifications from Comments in Tasks

You can send email notifications to other task assignees and also to a specific user while entering a comment in the **Comments** field of a task.

Email notifications can be sent to task assignees or to a specific user by tagging them in the comment entered for a task. End users can tag comments with @assignees to notify all task assignees and @userid to notify a specific user.

An email with the specified comment is sent accordingly. The subject of the email notifies that a comment has been entered by the sender for a particular task. The recipient can view the comment entered for the task in the email without actually opening the task. A link to the task for which the comment was entered is provided in the email. The recipient can click the link to access the task and view details.

Note that before you can send email notifications from a comment, you have to enable email notifications in runtime. You can also view the notification logs and resend emails to recipients. See [View and Resend Email Notifications](#).

See [How do I send email notifications from a comment?](#)

Archive and Purge Data

You can submit requests for scheduled instances archive and purge or scheduled analytics archive to back up your process instances from one or more applications. The data you archive gets saved in Oracle Storage Service.

You can also schedule the Auto Purge from the Schedule Instances Archive page and be able to remove the BPM runtime information. The purge runs as a separate job on the database and ensures optimal performance.

 **Note:**

Make sure you have configured the Oracle Storage Service settings before you archive. See [Configure Oracle Storage Service](#).

Schedule Instances Archive and Purge

Specify whether and when to archive your data automatically by creating a schedule based on either a time you select or a CRON expression. Also specify the number of days to retain data before it is purged. (You do not need to enable archiving to set purging retention.)

 **Note:**

Data retention settings configured in the Oracle Integration Settings area can affect Process instance retention. For example, the **Purge When Low Space Reached** field determines whether Integration and Process runtime instances are automatically purged without reclaiming database space. In addition, clicking the **Perform Manual Purge** button performs a manual purge of all integration and process instances. See Set Data Retention in *Provisioning and Administering Oracle Integration Generation 2*.

To schedule instance archive and purge:

1. In the Oracle Integration navigation pane, click **My Tasks**, click **Workspace**, and then click **Administration**.
2. Click **Archive and Purge**.
3. Click the **Schedule Instances Archive** tab.
4. To schedule archiving, select the **Enable Archive** check box and select archive settings.

 **Note:**

Disabling **Enable Archive** and clicking **Save** terminates all archive schedules.

- a. Configure a schedule for the archive. You can use either the time fields or the advanced scheduling interval.
 - Use the time fields to specify when (how often, what day, and what time) the archive occurs. For example, to schedule an archive for *every week on Friday at 03:20*, for **Every**, select **Week**, for **on** select **Friday**, and for **at**, select **03** for hour and **20** for minutes.
 - Use the advanced scheduling option to specify a CRON expression. Select the **Use Advanced Scheduling Interval** check box and enter a valid CRON expression. Click the adjacent help icon for CRON details and examples.

 **Note:**

Enabling the **Use Advanced Scheduling Interval** check box automatically disables the settings in the time fields.

- b. In the **Configure Archive Content** fields, select items to include in the archive.
 - c. In the **Archive Job Timeout** field, enter the maximum number of minutes that the archive is allowed to run.
 - d. In the **Failure Notification Address** field, enter an email address to send archive error notifications to.
5. Schedule purging in the **Purge Retention** field.
- Enter the number of days that data should be retained. Once the number of days is complete, the data gets purged. (The default number of days is 7.) You can set the purge retention without enabling archiving.
6. Click **Save**.

Review the confirmation message for details about your request.

You can expand the **Archive Requests** section to view the status of your scheduled archives. See [View Archive Requests](#).

Schedule Analytics Archive and Purge

Specify whether and when to automatically archive your analytics data by creating a schedule based on either a time you select or a CRON expression. Also specify the number of days to retain analytics data before it is purged. (You do not need to enable archiving to set purging retention.)

To schedule analytics archive and purge:

1. In the Oracle Integration navigation pane, click **My Tasks**, click **Workspace**, and then click **Administration**.
2. Click **Archive and Purge**.
3. To schedule analytics archiving, select the **Schedule Analytics Archive** tab, select the **Enable Archive** check box and select archive settings.

 **Note:**

Disabling **Enable Archive** and clicking **Save** terminates all analytics archive schedules.

- a. Configure a schedule for this archive. You can use either the time fields or the advanced scheduling interval.
 - Use the time fields to specify when (how often, what day, and what time) the archive occurs. For example, to schedule an archive for *every week on Friday at 03:20*, for **Every**, select **Week**, for **on** select **Friday**, and for **at**, select **03** for hour and **20** for minutes.

- Use the advanced scheduling option to specify a CRON expression. Select the **Use Advanced Scheduling Interval** check box and enter a valid CRON expression. Click the adjacent help icon for CRON details and examples. Enabling the **Use Advanced Scheduling Interval** check box automatically disables the settings in the time fields.
- b. In the **Failure Notification Address** field, enter the email address that all error notifications will be sent to.
4. Schedule analytics purging.
Click the **Schedule Analytics Purge** tab and enter the number of days that data should be retained in the **Purge Retention** field. Once the number of days is complete, the data gets purged. (The default number of days is 7.) You can set the purge retention without enabling archiving.
 5. Click **Save**.
Review the confirmation message for details about your request.

You can expand the **Archive Requests** section to view the status of your scheduled archives. See [View Archive Requests](#).

After you schedule an analytics archive, some records may not be archived to the Oracle Storage Service. In such cases, the missing records are picked up during the next archive cycle.

View Archive Requests

When you submit a request for a scheduled instance archive or a scheduled analytics archive, the Archive Requests table displays the current status of the job.

To view the Archive Requests table:

1. In the Oracle Integration navigation pane, click **My Tasks**, click **Workspace**, and then click **Administration**.
2. In the Administration pane, click **Archive and Purge**.
3. Expand the **Archive Requests** section on the appropriate archive tab.

| State | Description |
|------------|--|
| Pending | The archive request has been submitted for processing, but it hasn't been picked up by the scheduler. |
| Running | The archive is in progress. |
| Completed | The archive is complete. The archived data is stored in Oracle Storage Service. |
| Terminated | The request was canceled or stopped, most likely due to a server restart. May also occur when the schedule gets changed. |
| Failed | An exception occurred while executing the job. To get details, view the server log file or use the REST APIs. |

 **Note:**

A scheduled request maintains its job ID and one row, and the state alternates between **Pending** and **Running**. The scheduled request gets **Terminated** only when the schedule gets changed. In that case, the scheduled request gets a new job ID. The request will again go into the **Pending** and **Running** states.

Work with Archive Data

The data that you archive from one or more applications can be retrieved and used for audit and other purposes.

The key steps to archive and then extract the archived data for use are:

1. Configure Oracle Storage Service settings. Use the Oracle Cloud Infrastructure (OCI) Object Storage service accessed with Swift style API. Ensure that you have adequate space for the archive data.
2. Enable archiving. In the Administration pane, click **Archive and Purge**. Ensure that the **Enable Archive** check box is selected in the Schedule Instances Archive of the Configure Archive and Purge page.
3. Configure the content to be archived.
See [Schedule Instances Archive and Purge](#).
4. Use a tool to extract the data from Oracle Storage Service.
5. Analyze and format the archive data to fit your organization's needs.

Want to learn more? Start by exploring the archive structure and then learn about the various components of the archive.

- [Explore Archive Structure](#)
- [Components of Archive](#)

Note the following about archive jobs:

- Users can set the limit for archive job timeout.
- The maximum allowed export size is 500 MB. If the file system usage exceeds the maximum threshold of 500 MB, whatever has been exported till that time will be packaged and uploaded to the object storage.
- In an archive job, when the file system usage exceeds the maximum threshold (500 MB), but the timeout limit set by the user for the archive job has not yet been exceeded, the archive job will attempt another batch. In such a case the last flow id to be exported may become the first flow id exported in the next batch, resulting in duplicate exports.

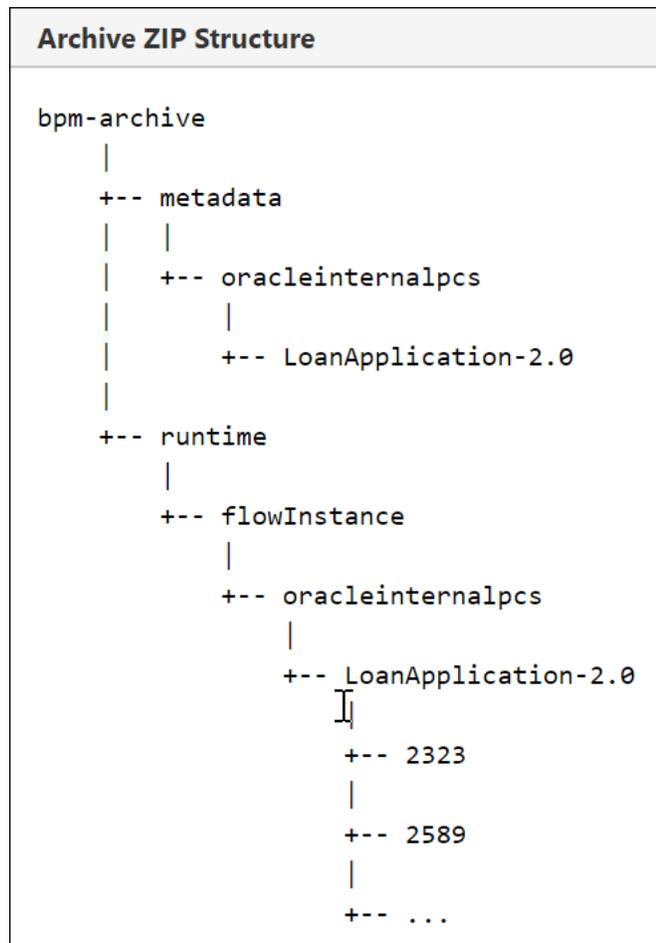
Explore Archive Structure

The data you archive gets saved as a ZIP file. The root directory is bpm-archive. There are a number of sub-directories under this root directory. The two main sub-directories are – metadata and runtime.

The metadata directory provides details about the deployed applications.

The runtime directory contains the exported runtime data. This runtime data is distributed across sub-directories that identify the application, its version, and the Flow_ID for the application.

The Flow_ID is an internal identifier that unites related process instances into a common flow. Each time a process gets executed a process instance ID is created. For example, two processes will have unique process instance IDs. And if one of the process calls the other process, another unique process instance ID is created. But all the three will share a common Flow_ID. The runtime data is exported based on this common Flow_ID so that related processes get exported together and contained in the same export sub-directory.



Components of Archive

You can use the components of the archive to display and examine the extracted information.

The archive contains the following components:

- MetaData
- Runtime
- Audit Diagram
- Audit Trace
- Task

Metadata

The metadata directory contains information about the process applications that corresponds to the process instances contained in the archive. The application files are stored in their own directory structure that identifies the partition they are deployed to – oracleinternalpcs, application name, and its version. The files are stored in this way to ensure separation if there are multiple deployments of the same application.

Runtime

The runtime directory contains the exported runtime information. The runtime information is distributed in several sub-directories to keep the overview of the applications, their versions, and the related process instances.

It contains an important directory called the FlowInstance directory. See [FlowInstance](#).

Audit Diagram

The Audit Diagram is a PNG file that contains a snapshot of the process instance and the process flow, that is, the path taken by the process during its execution.

The file is named as *audit-diagram- $\{processInstanceid\}.png$* , where, $\{processInstanceid\}$ is the process instance identifier that the diagram represents.

Audit Trace

The audit trace file is an export of the information collected by the audit system. This contains a root level audit log element and then any number of audit instance elements. Each audit instance element represents some operation that occurred in the execution of the process. The name of the file follows a similar structure to the audit diagram in that it is appended with the Process Instance Id, for example, *audit-trace- $\{processInstanceid\}.xml$* .

Note that if more than one process instance have been invoked, then there will be more than one audit trace file.

See [Audit Trace Elements](#).

Audit Log

The audit log is a structured object that contains the detailed information about the audit instance.

The components of the audit log are:

- Data State
- Gateway Execution
- Service Input
- Service Output

See [Audit Trace Elements](#).

Task

Task provides information about the human tasks executed during the process execution. It also includes the comments and attachment details. When external Social

Network or Oracle Docs is not configured, then the comments are included in-line of the task export and the attachment are included in the archive content.

See [Tasks Elements](#).

Audit Trace Elements

Audit trace contains the following:

ActivityName

This is the internal generic name for an activity. It is of type string, and the allowed values are:

- CATCH_INTERMEDIATE_EVENT
- PARALLEL_GATEWAY
- SERVICE_TASK
- START_EVENT
- USER_TASK

AuditInstance

| Element | Type | Description |
|-------------------|---------------------|--|
| DNApplicationName | string | The name of the partition where the application is deployed into. |
| DNCompositeName | string | The name of the Composite, that is, the application name. |
| DNLabel | string | The unique label for an application deployment. |
| DNRevision | string | The revision number as given by the user during the deployment of the application. |
| ECID | string | An internal identifier for a group of related process instances. This identifier has given way to the Flow ID. |
| activityId | string | The identifier for an application with a process definition. |
| activityName | string - enumerated | The BPMN name for the activity. |
| auditInstanceType | string - enumerated | The audit instance type determines at what stage of an activities life the audit message was recorded. Traditionally, this was START and END to represent pre-audit and post-audit messages. |
| auditLevel | integer | The audit level when the event was recorded. |
| auditLog | object | The audit log records more detailed information about the audit instance such as the state of the output data associations. |
| componentName | string | The name of the process. |
| componentType | string | The type of the process engine that executed the component. This will always be BPMN. |

| Element | Type | Description |
|---------------------|---------------------|--|
| compositeDn | string | The Domain Name identifier for the deployed application. This is based on four parts: <ul style="list-style-type: none"> • Deployment partition • Application name • version • label |
| compositeInstanceId | integer | The composite instance ID is an identifier for the deployed application. This is largely replaced by the Flow ID and is provided for backward compatibility. |
| compositeName | string | The name of the application process to which this audit record belongs. |
| createTime | dateTime | The timestamp at which the audit instance record was created. |
| dn | string | Domain name identifier for the deployed application. This is based on four parts: <ul style="list-style-type: none"> • Deployment partition • Application name • version • label |
| faultIsRecoverable | boolean | If a fault was recorded on this audit event, then this flag will indicate if this was a recoverable fault. |
| flowId | integer | The unique identifier that binds related process instances. |
| instanceCount | integer | The instance number for a process activity that has been replicated multiple times. |
| label | string | The user provided name for an activity in the process model. |
| loopCount | integer | During a looping construct, an activity can be visited more than once. This counter tracks the number of times an activity is executed. |
| operation | string - enumerated | This contains the audit instance operation. |
| parentThread | integer | Each branch of a BPMN process is called a thread. Each thread is assigned a unique number. By default, this value will be -1 meaning that this is the primary or initial thread. As the process branches, the thread ID will increase and this will contain the value of the threads parent thread ID. |
| partitionDate | dateTime | Internal use for the purge function where partition tables are used. |
| processName | string | The name of the actual process being executed within the application. |
| processTitle | string | The display name for the process being executed within the application. |
| queryId | integer | The identifier for the audit record. |
| scaPartitionId | integer | For internal use only. Used by the purge system. |
| scopeId | string | The scope contains some run time information such as variables and data. These are tracked using a scope ID. |
| sourceActivity | string | The previously executed activity in relation to the current activity. |

| Element | Type | Description |
|----------------|---------|--|
| targetActivity | string | The next activity to be executed in relation to this activity. |
| tenantId | integer | The identifier used for a multi-tenant install. This is for internal use only. |
| threadId | integer | Each branch of a BPMN process is called a thread. Each thread is assigned a unique number. By default, this value will be -1 meaning that this is the primary or initial thread. As the process branches, the thread ID will increase. |
| variables | array | This tracks the Analytics Variables. |

AuditInstanceType

The audit instance record can be classified into several types based on where they appear in the project model. It is of type string. The allowed values are:

- AFTER_INPUT_DATA_ASSOCIATION
- AFTER_INSTANCE_EXECUTION
- AFTER_OUTPUT_DATA_ASSOCIATION
- BEFORE_INPUT_DATA_ASSOCIATION
- BEFORE_INSTANCE_EXECUTION
- BEFORE_ITERATION
- BEFORE_OUTPUT_DATA_ASSOCIATION
- COMPONENT_STATUS
- END
- EXECUTION_LOGGING
- INTERMEDIATE
- START

AuditOperation

It is of type string. The allowed values are:

- AFTER_INPUT_DATA_ASSOCIATION
- AFTER_INSTANCE_EXECUTION
- AFTER_OUTPUT_DATA_ASSOCIATION
- BEFORE_INPUT_DATA_ASSOCIATION
- BEFORE_INSTANCE_EXECUTION
- BEFORE_ITERATION
- BEFORE_OUTPUT_DATA_ASSOCIATION
- COMPONENT_DEPLOYED
- COMPONENT_RETIRED
- COMPONENT_SUSPENDED
- COMPONENT_UNDEPLOYED

- EXECUTION_LOGGING
- FLOW_NODE_CANCELLED
- FLOW_NODE_DATA_CHANGED
- FLOW_NODE_IN
- FLOW_NODE_MOVED
- FLOW_NODE_OUT
- INSTANCE_ABORTED
- INSTANCE_CREATED
- INSTANCE_FAULT
- INSTANCE_RECOVERY_REQUESTED
- INSTANCE_RESUMED
- INSTANCE_SUSPENDED
- INSTANCE_SYSTEM_FAULT
- INSTANCE_TERMINATED
- INSTANCE_UPDATED
- MEASUREMENT_COUNTER
- MEASUREMENT_START
- MEASUREMENT_START_STOP
- MEASUREMENT_STOP
- STALE_ABORTED
- STALE_COMPLETED

AuditQueryPayload

| Element | Type | Description |
|---------|---------|--|
| auditId | integer | The instance ID for the current audit event. |
| ciKey | integer | The identifier for the process instance, that is, the Process Instance ID. |

BPMAudit

BPMAudit is the root element for the Audit Trace document. It contains an element `auditInstance` of type array. The `auditInstance` is the instance of the audit message. This will either be a pre-audit, post-audit or an ad-hoc message generated but the system to record such things as updates to the process flow (alter flow).

BPMNActivityNType

The type of activity that is being described. This can be a MESSAGE. It is of type string.

DataObject

The details of a data object. This can be a primitive, simple type or complex type. The structure of the value depends on the definition of the object within the process model design.

| Element | Type | Description |
|---------------------|---------|--|
| detailId | integer | The identifier for the data object value where it has had to be moved the overflow storage. For internal use only. |
| isBusinessIndicator | boolean | The flag to indicate if the data object is associated with an Analytics Business Indicator. |
| name | string | The name of the data object. |
| value | anyType | The raw value of the data object. |

DataState

It has the element dataObject. dataObject contains details of a data object. This can be a primitive, simple type or complex type. The structure of the value depends on the definition of the object within the process model design.

Element

It contains the details of a data object. This can be a primitive, simple type or complex type. The structure of the value depends on the definition of the object within the process model design.

| Element | Type | Description |
|---------------------|---------|---|
| isBusinessIndicator | boolean | The flag to indicate if the data object is associated with an Analytics Business Indicator. |
| name | string | The name of the data object. |
| value | anyType | The raw value of the data object. |

FlowElementType

The flow element type refers to the basic BPMN type of the element. It of type string. The allowed values are:

- ACTIVITY
- EVENT
- GATEWAY
- PROCESS

ServiceInput

The details of a data object. This can be a primitive, simple type or complex type. The structure of the value depends on the definition of the object within the process model design.

ServiceOutput

The details of a data object. This can be a primitive, simple type or complex type. The structure of the value depends on the definition of the object within the process model design.

Variable

It contains the elements - name and variableInstance.

name is the name of the variable.

variableInstance contains the details of an analytics variable that are being tracked by audit.

VariableDirection

It is of type string. Allowed value: DIRECTION_STATE.

VariableInstance

| Element | Type | Description |
|-----------|---------|--|
| attribute | integer | The slot number that the variable occupies. |
| name | string | The name of the variable. |
| table | string | The name of the database table where the data value is stored. |
| type | string | The type of variable. |
| value | anyType | The value of the variable for this instance of the audit. The value is of type Object and can represent xsd:anyType. |

VariableModelType

The data type of the variable as it is declared in the application process model. It is of type string. The allowed values are Integer and String.

VariablePrimitives

The data type of the variable. It is of type string. The allowed values are:

- DATE
- NUMBER
- STRING

FlowInstance

The flowInstance directory is present immediately under the runtime directory.

Under the flowInstance directory is the partition directory – *oracleinternalpcs*. This is present to distinguish applications that may be deployed to another partition. The sub-directories for individual applications can be found under the partition directory. The name of the sub-directories are based on the application name and its version.

Under the sub-directories (named based on the application name and version) are the individual process instance directories. The name of these directories are derived from the Flow_ID of the process instance.

Key points to note about the Flow_ID:

- The Flow_ID maintains reference to all related process instances.

- The Flow_ID is the internal reference that ties all related processes together. For example, when a process calls another process, a Process Instance ID is created for the new process. The two processes will each have their own process instance IDs. In this case all of them will share the same Flow_ID. So the Flow_ID can contain runtime information of more than one related process.
- Flow_ID provides the sub-directory name for the exported runtime information. The sub-directories contain the individual export files that are all related to the same Flow_ID.

ProcessInstance

The process instance file contains the summary of the process instance. Note that the information contained in the file is at the time of export.

| Element | Type | Description |
|--------------------|---------|--|
| applicationContext | string | The application name. |
| creator | string | The creator of the application instance. |
| cubeInstancelId | string | The cube instance identifier. |
| identityContext | string | The scheme used for identity |
| priority | integer | The priority designator of the deployment. |
| processDN | string | The Domain Name for the process. |
| title | string | The display name for the process instance |

| Element | Type | Description |
|----------------------|---------|--|
| componentInstancelId | integer | The instance ID of the component. This should also represent the process instance ID. |
| componentName | string | The name of the process within the component/application. |
| compositeDN | string | The Domain Name for the composite/application. |
| compositeInstancelId | integer | The identifier of the composite or application. This Id is not exposed to users. |
| compositeVersion | string | The revision number of the deployed application. |
| ecId | string | The ECID is an identifier to bind several related process instances together. This has been replaced with the FLOW_ID. |
| flowId | integer | The unique identifier that binds related process instances. |
| scaPartitionId | string | The deployment partition for the application. |

Tasks Elements

Tasks contains the following:

Action

It is of type string. The allowed values are:

- VIEW_PROCESS_HISTORY
- VIEW_SUB_TASKS

- VIEW_TASK
- VIEW_TASK_HISTORY

Attachment

| Element | Type |
|----------------------|----------|
| attachmentScope | string |
| contentType | string |
| name | string |
| size | integer |
| taskId | string |
| updatedBy | string |
| updatedByDisplayName | string |
| updatedAt | dateTime |
| version | integer |

Callback

| Element | Type |
|----------------|--------|
| conversationId | string |
| id | string |

Conversations

It contains the element conversationsEnabled which is of type boolean.

CustomAttributes

| Element | Type |
|------------------------|--------|
| customAttributeNumber1 | number |
| customAttributeNumber2 | number |

DocumentationDetails

It contains the element DocsEnabled which is of type boolean.

Payload

It contains the element IntakeWebForm which is of anyType.

ProcessInfo

| Element | Type |
|-------------|---------|
| instanceId | integer |
| processId | string |
| processName | string |

Sca

| Element | Type |
|------------------------------|-------------|
| applicationName | string |
| componentInstancelId | integer |
| componentName | string |
| compositeCreatedTime | dateTime |
| compositeDN | string |
| compositeInstancelId | integer |
| compositeName | string |
| compositeVersion | string |
| eclId | string |
| flowId | integer |
| parentComponentInstancelId | string |
| parentComponentInstanceRefId | string |
| scaPartitionId | integer |

ShortHistoryTask

| Element | Type |
|----------------|-------------|
| state | string |
| updatedAt | dateTime |
| version | integer |
| versionReason | string |

SystemActions

| Element | Type |
|----------------|-------------|
| displayName | string |

SystemAttributes

| Elements | Type |
|-----------------------|-------------|
| actionDisplayName | string |
| activityId | string |
| activityName | string |
| approvalDuration | integer |
| approvers | string |
| approversDisplayNames | string |
| assignedDate | dateTime |
| componentType | string |

| Elements | Type |
|--------------------------|-------------|
| createdDate | dateTime |
| digitalSignatureRequired | boolean |
| endDate | dateTime |
| formName | string |
| hasSubTask | boolean |
| imageUrl | string |
| inShortHistory | boolean |
| isDecomposedTask | boolean |
| isGroup | boolean |
| isTemplateTask | boolean |
| isTestTask | boolean |
| numberOfTimesModified | integer |
| outcome | string |
| parentThread | integer |
| participantName | string |
| passwordRequiredOnUpdate | boolean |
| pushbackSequence | string |
| rootTaskId | string |
| secureNotifications | boolean |
| state | string |
| swimlaneRole | string |
| systemActions | array |
| systemStringActions | string |
| task | array |
| taskDefinitionId | string |
| taskDefinitionName | string |
| taskId | string |
| taskNamespace | string |
| taskNumber | integer |
| thread | integer |
| timersSuspended | boolean |
| updatedAt | dateTime |
| version | integer |
| versionReason | string |
| workflowPattern | string |

SystemMessageAttributes

| Element | Type |
|----------------------------|-------------|
| numberAttribute1 | number |
| numberAttribute10 | number |
| numberAttribute2 | number |
| numberAttribute3 | number |
| numberAttribute4 | number |
| numberAttribute5 | number |
| numberAttribute6 | number |
| numberAttribute7 | number |
| numberAttribute8 | number |
| numberAttribute9 | number |
| protectedNumberAttribute1 | number |
| protectedNumberAttribute10 | number |
| protectedNumberAttribute2 | number |
| protectedNumberAttribute3 | number |
| protectedNumberAttribute4 | number |
| protectedNumberAttribute5 | number |
| protectedNumberAttribute6 | number |
| protectedNumberAttribute7 | number |
| protectedNumberAttribute8 | number |
| protectedNumberAttribute9 | number |

Task

| Element | Type |
|----------------------|-------------|
| applicationContext | string |
| correlationId | string |
| creator | string |
| creatorDisplayName | string |
| identityContext | integer |
| isPublic | boolean |
| mdsLabel | string |
| ownerRole | string |
| ownerRoleDisplayName | string |
| percentageComplete | number |
| priority | integer |
| taskDefinitionId | string |
| taskDefinitionURI | string |

| Element | Type |
|-------------|--------|
| title | string |
| userComment | array |

UpdatedBy

| Element | Type |
|-------------|--------|
| displayName | string |
| id | string |
| type | string |

Configure Credentials for Web Services

You can use a credential to securely call a web service. You can add new credentials, make changes to existing credentials, or delete them.

You can manage credentials for a web service in both the environments. However, it's more common to create a credential when you're creating the connector to the web service. See [Create a Web Service Connector](#).

Add a Runtime Credential

1. In the Oracle Integration navigation pane, click **My Tasks**, click **Workspace**, and then click **Administration**.
2. Click **Manage Credentials**.
3. Click **Add new credential**.
4. Click the **Map** field and select **BPM WebServices Credential** for a web service credential.
5. Complete the following fields:
 - **Key**: an identifier for the web services credential.
 - **User Name**: the name of the user to access the service.
 - **Password**: the password to access the service. You must also enter this same password in the **Confirm Password** field.
6. Click **Add**.
7. Click **Save**.

After you add a credential, you can modify the user name, password, and description. To modify other settings, you must delete and recreate the credential.

- To undo your current changes, click **Revert**.
- To remove the credential, click **Delete**.

Manage Security Certificates during Runtime

Certificates are used to validate an application's external web service connections when message security is applied. You can replace expired certificates and maintain a separate set of certificates for production use only.

See [Apply Message Security to Integrations](#) and [Manage Security Certificates During Design Time](#).

To view, add, update, or delete certificates for production:

1. In the Oracle Integration navigation pane, click **Settings**, then **Certificates**.
The Certificates screen appears. Use this page to manage certificates for runtime certificates for process applications and integrations.
2. View, upload, update, or delete certificates. See *Upload an SSL Certificate in Provisioning and Administering Oracle Integration Generation 2*.

Show Dates in User Time Zone

You can configure Process to show dates based on the user's time zone.

1. In the Oracle Integration navigation pane, click **My Tasks**, click **Workspace**, and then click **Administration**.
2. Click **UI Customization**.
3. Select the **Show Dates in User's Time Zone** check box to display dates to users based on their system's time zone. By default, this setting is unchecked and all times are displayed based on server time zone.
4. Click **Save**.

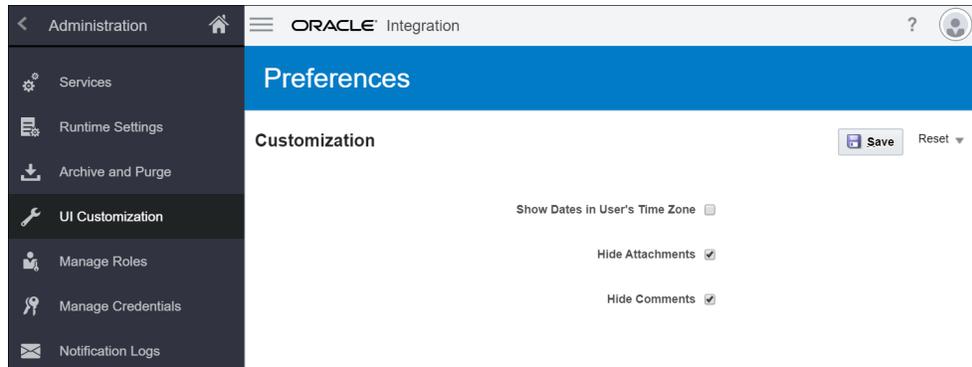
To restore the default settings, click **Reset**, and then click **Reset to Default**. To revert your changes to the last saved version, click **Reset**, and then click **Reset to Last Saved**.

Hide Comments and Attachments

You can configure Process to hide comments and attachments in tasks and start forms.

To hide comments and attachments:

1. In the Oracle Integration navigation pane, click **My Tasks**, and then click **Administration**.
2. Click **UI Customization**.
3. On the Preferences screen, choose the customization options:
 - Select **Hide Attachments** to hide attachments from tasks and start forms.
 - Select **Hide Comments** to hide comments from tasks and start forms.



4. Click **Save**.

To restore the default settings, click **Reset**, and then click **Reset to Default**. To revert your changes to the last saved version, click **Reset**, and then click **Reset to Last Saved**.

Set the Default View for Process History

You can set the default view for process history from the UI Customization page.

1. In the Oracle Integration navigation pane, click **My Tasks**, click **Workspace**, and then click **Administration**.
2. Click **UI Customization**.
3. Select one of the following options available from the **Process History View Type Setting** drop-down list.
 - List View
 - Tree View
 - Graphical View
4. Click **Save**.

The option you select will be set as the default view for Process history in all available processes in Workspace.

For example, if you selected *Graphical View* in the **Process History View Type Setting**, the process history of all available processes in Workspace will appear in the graphical view. If required, you can change the process history view of a particular process by choosing another option such as *Tree View* or *List View* from the drop-down list available under the History section of that process.

Troubleshoot the Runtime Administration

The following troubleshooting tips apply to the runtime environment.

Why are several options, such as Dashboards and Administration, not available in the navigation pane?

Verify that you're assigned the administrator role for Process. That role is required to access many administration features. See [Learn About Roles for Processes](#).

How do I find out who is currently assigned to a specific process instance?

In the Instances page, locate the instance using the filters or by searching. Select it to view its details. In the Details pane, expand the **More Information** section. The Assigned To field indicates the current assignee for that process instance. See [Track Process Instances](#).

How do I recover a failed process instance?

Locate the instance using the filters in the Instances page. If you're an administrator, you can also click the alerts on the home page. Select the instance to view its details. Click **Alter Flow & Suspend** to edit the value of the process instance data objects and resume it. See [Alter the Flow of a Process Instance](#).

How do I reassign a task that a user assigned to the wrong person?

If you or a user assigned a task to the wrong person by mistake, you can locate the process instance and reassign the task. Use the filters to locate the instance, select it, and reassign it to a different user. See [Can I reassign my tasks to someone else?](#)

18

Manage the Design-Time Environment

As an administrator, you can administer and manage the design-time environment.

Topics:

- [Administer Spaces, Applications, and the Player](#)
- [Manage Environments and Activate Applications](#)

Administer Spaces, Applications, and the Player

In the design-time environment, manage spaces, applications, and the player.

Topics:

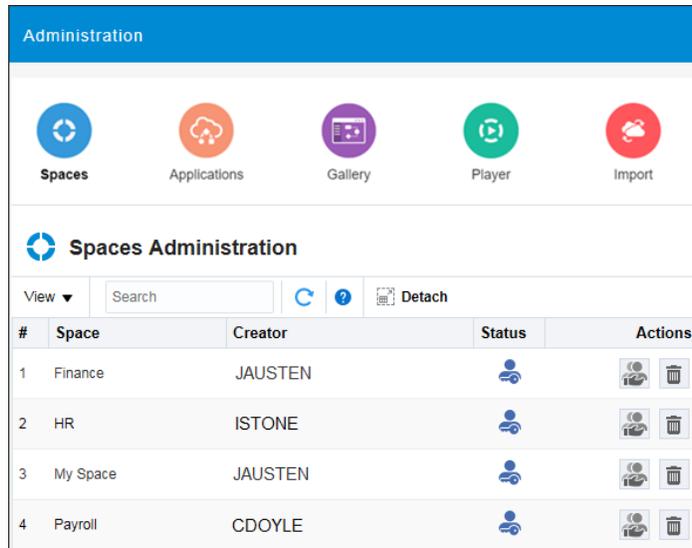
- [Administer Spaces](#)
- [Unlock and Delete Applications](#)
- [Delete Apps from the Gallery](#)
- [Enable the Application Player](#)

Use the **Import** tool to move process applications and active decision models from one location to another. See [Import Process Assets to Oracle Integration](#) in *Administering Oracle Integration Classic*.

Administer Spaces

Check the status of a space (private or shared), control permissions to a space, or delete a space. As an administrator, you may need to perform these actions if, for example, the owner of the space is on vacation.

In the Oracle Integration navigation pane, click **Processes**, and then click **Administration** to display the Administration pages. The Spaces Administration page displays by default



To control permissions to a space, click **Share** in the **Actions** column. In the Share -*space-name* dialog box, perform any of the following actions:

- To add a user to the space, enter or search for the user, select a role, and click **Share**.
- To change the user's role in the space, select the user from the table, click **Edit Role**, select a new role, and click **OK**.
- To delete a user from the space, select the user from the table, click **Delete**, and click **Yes**.

To delete a space, click **Delete space and content** in the **Actions** column. Note that deleting a space deletes the space and all the applications it contains.

! Important:

Contents can't be recovered once the space is deleted.

Unlock and Delete Applications

Use the **Applications** option on the Administration page to unlock a shared application locked by another user, or to delete a shared application.

Shared applications are automatically locked when a user opens the application in **Edit** mode to prevent other users from editing an application at the same time. As an administrator, you may have to release a lock if, for example, a user has forgotten to close an application before going on vacation.

To unlock a shared application, click **Unlock Application** in the **Actions** column.

 **Caution:**

If you unlock an application that has unpublished changes, the changes are lost and can't be recovered.

As an administrator, you can delete shared applications. A private application, on the other hand, can be deleted only by one of its owners.

To delete a shared application, click **Delete Application**  in the **Actions** column. The application is removed from the Process repository.

You can't delete an application that is locked.

Delete Apps from the Gallery

Users can rapidly build a new application based on a Sample or QuickStart App listed in the gallery. They can personalize the application to fit their business needs and then activate it. Similarly, users can also create a decision model using Decision Samples available in the gallery. As an administrator, you need to monitor and manage the list of Apps available in the gallery. For example, you may want to remove a QuickStart App that is no longer used or a duplicate Decision Sample.

Deleting an App removes the App from the gallery only. Users will no longer be able to select the App to create an application or decision model. However, the source application or model will still be available on the design-time Home page. You can promote it back to the gallery at any time.

To delete an App from the gallery:

1. On the Oracle Integration navigation pane, click **Processes**, and then click **Administration**.
2. Click **Gallery**.

This page lists all your Apps that have been promoted to the gallery. It doesn't include the standard Apps, such as Loan Application or Travel Approval, provided by Oracle. You can't delete the standard Apps from the gallery.

3. Locate the App you want to remove from the gallery, and click **Delete**  in its **Actions** column.

Use the **Show** filter to look for a specific type of Apps, such as Samples, QuickStart Apps, or DMN Samples. Alternatively, type the name of an App in the **Search** field to find it.

Enable the Application Player

You can use the application player to test the behavior of a business process at design time. You can test a process using different user IDs without having to explicitly activate the application and test the business process using the runtime environment

When testing a business process, the application player activates a version of your application to a special partition in Process. As a result, the player runs your business process using the same environment as your normally activated applications.

To enable the application player, you must provide the credentials for your Process administrator. These credentials are the user name and password of any user with administrator privileges.

After you provide the administrator credentials, any user who has editing privileges for a process application can run the application player.

When starting the player, the design-time environment **test activates** the application to the cloud using the specified credentials. When the current user wants to perform an activity or a task as another user, the design-time environment also uses these credentials.

To enable the application player:

1. In the Oracle Integration navigation pane, click **Processes**, and then click **Administration**.
2. Click **Player**.
3. Enter the credentials for a Process administrator.
4. Click **Save**.

After enabling the application player, users that have editing privileges can use the player to test the behavior of their business processes.



Note:

Clicking **Clear** removes the credentials and as a consequence, the application player is disabled and no longer available to users. The administrator can also use this procedure to simply change the credentials (change the password).

Want to learn more about using the application player? See [Play Processes and Test Applications](#).

Manage Environments and Activate Applications

On the Manage Active Applications page in the design-time environment, administrators can activate applications, configure environments and activation permissions, and manage activations.

In the Oracle Integration navigation pane, click **Processes**, and click **Process Applications**. Then click **Activate** and enter your credentials.

The **Activate** page displays information about your servers, such as current status and activate permissions (private or open), and information about each activated application, such as name, revision ID, and current status. If an application was activated from a particular snapshot, then the Application Name column includes both the application name and the snapshot name.

| Manage Active Applications | | | | | | |
|----------------------------|------------------|-------------|--------|--------|------------------|------------------|
| Search | | | | | | All Applications |
| My Server | Application Name | Revision id | Mode | Status | Activation Date | Actions |
| Private | Application | 1.1 | Active | On | 7/5/2017 8:50 AM | |

From the **Activate** page, you can:

- [Configure activation permissions](#)
- [Activate applications](#)
- Perform specific actions, such as [deactivate](#), [retire](#), or [shutdown](#) an activated application
- [Enable message security and manage certificates](#)

Configure the Activation Permissions

Let's talk about the activation strategy. Do you want just your administrators to be able to activate an application? Developers can create applications and test activate them to a sandbox environment, but they have to ask administrators to activate the final application to production. On the other hand, do you want developers to be responsible for activating their applications?

Think about how you want to handle this and what approach is best for your organization. Also, if you have more than one instance of an Oracle Integration, then different environments can have different strategies. For example, one instance might be for building, modeling, and testing applications. This instance is **open** so developers and administrators can activate applications. The second instance is only for production applications. This instance is **private**, and only administrators can activate applications.

To configure the activation permissions:

1. In the Oracle Integration navigation pane, click **Processes**, click **Process Applications**, and then click **Activate**.
2. For each server, use the drop-down list to select the permission for activating applications to the server.
 - **Private**: Only administrators can activate applications to the server.
 - **Open**: Administrators and developers can activate applications to the server.

Activate Applications

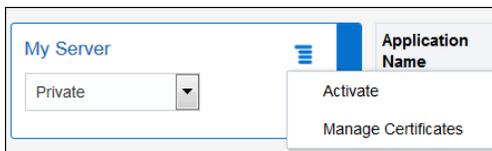
If you have administrator privileges, then you can activate applications to a production or testing environment from the Manage Active Applications page in design time.

Note:

To quickly back up your application, export it as an EXP file, and save it locally. You can then import it at anytime from the Process Applications page. See [Import and Export Applications and Snapshots](#).

To activate an application:

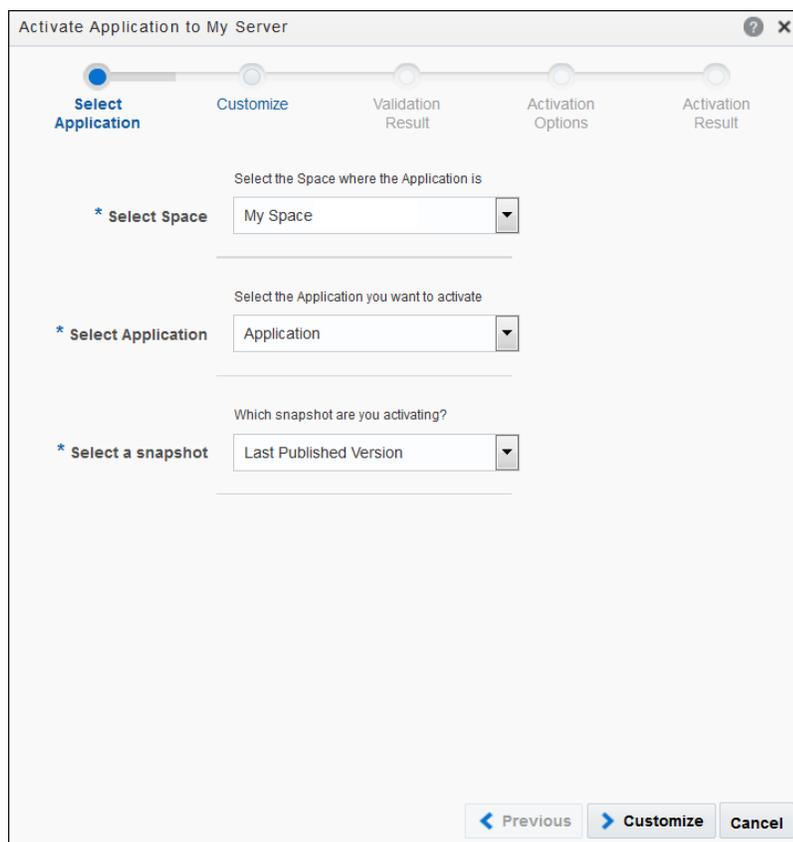
1. In the Oracle Integration navigation pane, click **Processes**, click **Process Applications**, and then click **Activate**.
2. Click **Options**  and select **Activate**.



The activate wizard opens. It will walk you through the activation process from selecting your application to setting options and viewing results. Continue reading if you want a preview of the activation process.

Select the Application to Activate

In the activate wizard, you select the space where the application is located, and the name and version of the application you want to activate. The available versions of the application are the **Last Published Version** and all snapshots that have been created.



Note:

As an administrator, you can select from all applications inside any space.

Customize REST and SOAP Services and Properties

If the process application you are activating uses REST or SOAP integrations, you can select different production settings for these integrations before activation. Your changes modify the original application and will be activated to the server. Your changes aren't published to the repository so you won't see them when editing the application.

Note:

OIC Integrations (integrations created in the Integrations feature) cannot be customized here, and are not shown. If your application contains no REST or SOAP integrations, this page is skipped and you move directly to the Validation Result page.

Activate Application to My Server

Select Version **Customize** Validation Result Activation Options Activation Result

Use design-time credentials and certificates
This option will create/update activation keystore credentials and certificates with the values entered during the design phase.

Customize Services used in the Application

Integrations: Weather

Operation Name

* Base URL: http://api.worldweatheronline.com/premium/v1/weath

Read Timeout: 300000 Milliseconds

Connection Timeout: 300000 Milliseconds

Security: APP Id - Basic Auth

* Keystore Credential: Test

* User: Testname

* Password:

< Previous Validate > Cancel

- From the **Integrations** field, select a REST or SOAP integration used in the process application to customize, and its properties display. (Integrations must be used in the process application to display in this list.)
- From the selected integration's properties, make changes as needed. For example, you might change a REST integration's address in the **Base URL** field.
- Specify credentials and certificates to use for the integration. As described below, you can create new runtime credentials or check the **Use design-time credentials and certificates** option, which populates the fields with design time settings.

When designing the application, you can define authentication keys to use when activating the application to the test environment. There are two different sets of credentials:

- One set for design time
- One set for runtime

This helps to keep the two environments completely isolated from one another. Nobody can change or use a runtime credential when designing or testing the application.

When the Customize page opens, it displays a list of connectors that are used in the processes. After you select a connector, you can view and edit its advanced options and security policy. Depending on the security policy, you must either select or create a security credential, and select or add a certificate alias.

On the Customize page, you're defining the runtime credentials. Remember that. You can select the **Use design-time credentials and certificates** check box to determine which set of certificates and credentials display in the Certificate Alias and Keystore Credential fields. Selecting the check box is a shortcut. It's a quick way to add the credentials and certificates that were defined at design time, and use them to create the runtime credentials.

If you select one of the design-time credentials to use (for example, `myKey`), then that credential is copied and a new runtime credential is created in the credential store. There are now two versions of this credential key: one for design time; the other for runtime. If you change the *username* and *password* during the activation wizard, then your change only affects runtime credentials. If you change the *username* and *password* in the design-time web service connector wizard, then your change only affects design-time credentials.

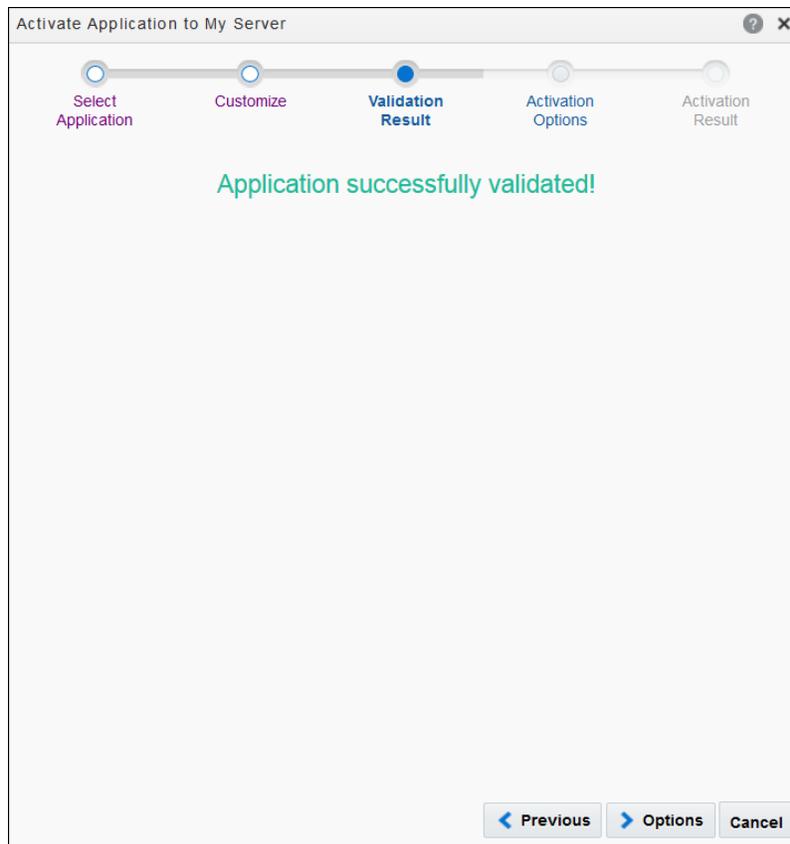
The credential and certificate are created or updated when you click **Activate**.

Want to learn more about integrations, credentials, and certificates? See [Integrate with Applications and Services](#).

Review Validation Results

The Validation page displays the validation results.

- *Validation Success*: A message indicating validation is successful displays. However, although the validation may be successful, a table with a list of warning messages may display.



If no warnings exist, a placeholder image is displayed. The **Options** button is now enabled.

- *Validation Failed:* A message indicating validation failed displays as well as a table with the validation errors. The **Options** button is inactive.

When validation fails, you can't continue until all errors have been fixed and validation is successful.

Specify Activation Options

On the Activation Options page:

- Enter a number in the **Version** field, such as 1 or 1.0, for this application revision. When activating the same process multiple times, use incremental version numbers each time.
- Optionally, make the new revision the default version. If you select the **Make Default** check box, only the default (primary) version of each application is used.
- In certain cases, it may be useful to disable automatic importing of system or runtime exceptions so that they can be handled at the process level. By default, however, the **Use Fault Policies** check box is selected and system faults are NOT caught by the process.

You have two options under **Use Fault Policies**. If you want any failed activity to be retried twice in runtime before it is considered as FAILED, select **Default**. If you want to skip failed activities being retried, select **No Retries**.

Activate Application to My Server

Select Application Customize Validation Result Activation Options Activation Result

* Version
Enter a unique id for the application version.

Make Default Displays this version prominently in My Apps and uses it in REST calls when no version is specified

Use Fault Policies Includes fault policies by default to handle system exceptions

Default No Retries

Process instances and tasks complete for more than 7 days are automatically deleted. Your administrator can change this purge setting.

View Activation Results

When you click **Activate**, the application is activated to the selected environment.

- *Activation Success*: A message indicating the application activated successfully displays. The active application is added to the Active Applications table for the environment.
- *Activation Failed*: A message indicating an error displays with more specific information about the error.

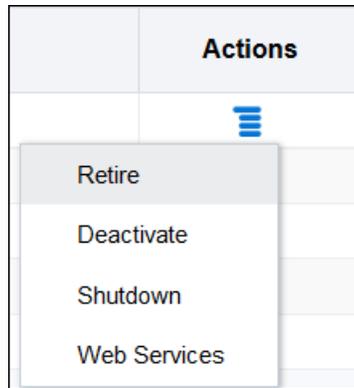
Manage Active Applications

After an application is activated, you might need to change its status or mode. For example, you might need to shut down an application temporarily or retire an application indefinitely.

In the Oracle Integration navigation pane, click **Processes**, then click **Process Applications**, and then click **Activate**.

In the Actions column, click **Options**  for the application you want to manage.

You can select to retire (or activate), deactivate, and shut down (or start up) an application. You can also view the web services exposed and consumed by the application.



Retire and Activate Applications

Retire and **Activate** are opposite actions. Retiring an application retires an active application and you can no longer create a new instance from that application version. Instances that are already created continue to run and new requests for existing instances are accepted. Activating an application brings it back from the retirement stage. The Manage Active Applications page reflects the new status in the **Mode** column (**Active** or **Retired**).

Activate activates a retired application revision. Note the following behavior with this option:

- All applications are automatically active when activated.
- Other revisions of a newly activated application remain active (that is, other revisions aren't automatically retired). If you want, you must explicitly retire them.

Deactivate an Application

When you deactivate an application version:

- All running and completed dynamic process instances and tasks of the deactivated application version are deleted.
- All running structured process instances and tasks of the deactivated application version are marked as **Stale** indicating that they are no longer active. All completed structured process instances remain as it is and continue to be marked as **Completed**.

Note that **Stale** and **Completed** structured process instances can be removed permanently through archive and purge. See [Archive and Purge Data](#).

Shut Down and Restart Applications

Shutting down an application is possible only after starting an application. Once an application is shut down, any request related to the application is rejected and no new request can be processed. All instances are stopped. You can restart the active application revision that was shut down. This action enables new requests to be processed (and not be rejected). No recovery of messages occurs. The **Activate** page reflects the new status in the **Status** column (**On** or **Off**).

 **Note:**

Shutdown shuts down a running application revision. Any request (initiating or a callback) to the application is rejected if the application is shut down. However, the behavior differs based on which component is used. For example, a web service request is rejected back to the caller.

View Information About Web Services

To view information about the web services used by a active application, click **Web Services**.

- The Exposed tab displays the web services URLs that the active application is exposing.
- The Consumed tab displays the web services that the currently active application connects to.

Manage Security Certificates During Design Time

Certificates are used to validate external web service connections for an application when message security is applied. If an external endpoint requires a specific certificate, request the certificate and upload it into Oracle Integration. An expired certificate results in a process instance error.

You can maintain separate certificates in design time and runtime, and you can override credentials during activation. This flexibility enables your organization to maintain separate certificates for test activation and production activation environments.

To configure message security and credentials:

1. Enable message security for a web service connector, as described in [Apply Message Security to Integrations](#).
Certificate and credential fields display when security is enabled by selecting the **APP Id - Username Token With Message Protection** option in the connector's Advanced **Security** field.
2. Specify a keystore credential and a certificate alias for the connector.
 - If needed, create a new certificate alias and upload a certificate by selecting a certificate file or pasting certificate contents.
 - If needed, create a new keystore credential and enter a name, user name, and password.
3. Manage credentials as needed. See [Configure Credentials for Web Services](#).
4. Manage certificates as needed. You can use any of the following methods to manage certificates.

| To... | Follow these steps... |
|--|--|
| Manage certificates during design time | <ol style="list-style-type: none"> 1. In the Oracle Integration navigation pane, click Processes, and then click Process Applications. 2. Click Activate. 3. Click Options  for your server and select Manage Certificates. <ul style="list-style-type: none"> • The Test tab displays information, such as alias and expiration date, for the design-time certificates uploaded in the design-time environment. • The Server tab includes a Manage Certificates link. Runtime certificates are managed centrally for processes and integrations. See Manage Security Certificates during Runtime. 4. Add or delete certificates. |
| Override certificates when you activate an application | <ol style="list-style-type: none"> 1. In the Oracle Integration navigation pane, click Processes, and then click Process Applications. 2. Click Activate. 3. Click Options  for your server and select Activate. 4. Enter information about the application you want to activate, and then go to the Customize page. 5. Use the options on the Customize page to specify services and security information. <ul style="list-style-type: none"> • Select the Use design-time credentials and certificates check box to select from design-time certificates created during design time rather than runtime. • Deselect the field to display and select from run-time credentials and certificates. |
| Update and manage runtime certificates | <ol style="list-style-type: none"> 1. In the Oracle Integration navigation pane, select Settings, then Certificates. 2. Create, upload, update, and delete certificates. See Manage Security Certificates during Runtime. |