

Oracle® Cloud

Developing Integrated Spreadsheets Using Oracle Visual Builder Add-in for Excel



Version 2.7

F43673-01

August 2021

The Oracle logo, consisting of a solid red square with the word "ORACLE" in white, uppercase, sans-serif font centered within it.

ORACLE®

Oracle Cloud Developing Integrated Spreadsheets Using Oracle Visual Builder Add-in for Excel, Version 2.7

F43673-01

Copyright © 2021, 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 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" or "commercial computer software documentation" 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

2 What's New in Release 2.7.0

New and Changed Features	2-1
Known Limitations	2-2

3 Introduction to Oracle Visual Builder Add-in for Excel

Key Concepts, Components, and Terms	3-1
Installation	3-1
Next Steps	3-1

4 Install Oracle Visual Builder Add-in for Excel

Install Using the All Users Installer	4-1
Install Using the Current User Installer	4-2
Run the Installer from the Command Line	4-3
Upgrade to the Latest Version	4-4
Upgrade Policy	4-5
Uninstall the Oracle Visual Builder Add-in for Excel	4-5
Software Dependencies	4-6
Supported Platforms	4-7

5 Create Layouts in an Excel Workbook

Create a Table Layout in an Excel Workbook	5-2
Work with Service Path Parameters in a Table Layout	5-4
Create a Form-over-Table Layout in an Excel Workbook	5-6
Use Descriptive Flexfields in a Layout	5-9
Manage Layout Capabilities	5-13

6 Edit Service Descriptions and Business Objects

Create a Service Description in Oracle Visual Builder	6-4
Add a Business Object to an Existing Catalog	6-4

Override a Business Object's Base Path	6-4
Configure Pagination for a Business Object	6-5
Configure Row Finders for a Business Object	6-7
Configure GZIP Compression for Request Payloads	6-9
Refresh a Business Object Catalog	6-9
Configure the REST-Framework-Version	6-10

7 Configure Search Options for Download

Use Search to Limit Downloaded Data	7-1
Use Row Finders to Limit Downloaded Data	7-2
Use Search Parameters to Limit Downloaded Data	7-3
Download Behavior in Layouts that Use Search and Row Finders	7-5

8 Custom Actions

Use Custom Actions	8-1
Edit Custom Actions	8-3
Service Description for Custom Actions	8-4
Batch Mode for Custom Actions	8-5
Custom Action Limitations	8-6

9 Use Lists of Values in an Excel Workbook

Configure a List of Values for a Business Object Field	9-2
Configure a List of Values for a Custom Action Field	9-3
Configure a List of Values for a Row Finder Variable	9-5
Configure Filters for a List of Values	9-6
Configure Search for a List of Values	9-7
Configure a List of Values with Multiple Display Fields	9-9
Configure a Cascading List of Values in a Layout	9-11
Configure the Bind Parameters for a Descriptive Flexfield's List of Values	9-13
Clear Cache for a List of Values	9-14
Notes and Limitations for Lists of Values	9-15

10 Appearance of an Integrated Excel Workbook

Reset Workbook Styles	10-1
Choose Field Formats	10-2
Add Tooltips to Your Workbook	10-2

11 Upload Changes

Upload Changes from a Table Layout	11-1
Upload Changes from a Form-Over-Table Layout	11-2
Upload Changes for Custom Actions	11-3
Upload Changes Using Batch Requests	11-5
Upload Changes Using Upsert Mode	11-5
Omit Empty Values During Upload	11-6
Data Consistency	11-7
Configure Parallel Requests During Upload	11-8

12 Use Multiple Layouts for Multi-level Business Objects

Create a Set of Dependent Layouts	12-4
Download, Upload, and Clear Operations on Dependent Layouts	12-8
Support for Creating Items	12-10
Delete a Dependent Layout	12-11

13 Use Macros in an Integrated Excel Workbook

14 Publish an Integrated Excel Workbook

15 REST Service Support

Service Types	15-1
Supported Data Types	15-1
Required Fields	15-2
REST Operations	15-2
Language Support	15-2
Oracle Business Object REST API Services	15-3
Oracle REST Data Services	15-3
Known Issues with ORDS	15-4
Other Services	15-4
Object-typed Fields and Sub-fields	15-4
REST Service Support Limitations	15-5

16 Internationalization

Change the Add-in's Language	16-1
Refresh All Field Titles	16-2

17 Security Best Practices

18 Troubleshoot Excel Workbooks

Check Your Environment	18-1
Apply Microsoft Updates	18-2
Re-Enable Oracle Visual Builder Add-in for Excel	18-3
Network Monitor	18-4
Logging	18-5
Logging Console	18-5
Diagnostic Report	18-6

19 Migration

20 Embedded Browsers

21 Third Party Licenses

Preface

Developing Integrated Spreadsheets Using Oracle Visual Builder Add-in for Excel describes how to develop Excel workbooks that can retrieve and modify business data exposed by a REST service and send modified data back to the service.

Topics:

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

Audience

Developing Integrated Spreadsheets Using Oracle Visual Builder Add-in for Excel is intended for developers who want to create and publish Excel workbooks that integrate with enterprise applications that they use.

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

For more information, see these Oracle resources:

- [Oracle Public Cloud](#)

<http://cloud.oracle.com>

- About Oracle Visual Builder in *Developing Applications with Oracle Visual Builder*
- Introduction to Accessing Business Objects in *Accessing Business Objects Using REST APIs*
- View and Edit Data Using an Excel Workbook in *Managing Data Using Oracle Visual Builder Add-in for Excel*

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.
monospace	Monospace type indicates commands within a paragraph, URLs, code in examples, text that appears on the screen, or text that you enter.

2

What's New in Release 2.7.0

Here's an overview of new features and enhancements added to Oracle Visual Builder Add-in for Excel in Release 2.7.0. Take note also of some limitations when using the add-in.

New and Changed Features

- Support for including table layouts for second-level "siblings" in a set of dependent layouts with a Form-over-Table primary layout. When creating a set of dependent layouts for a multi-level hierarchy of business objects, you can now link a separate table layout based on a second-level business object to a top-level Form-over-Table layout. See [Use Multiple Layouts for Multi-level Business Objects](#).
- A new MSI-based installer (`vbafe-installer-current-user.msi`) for Current-User installs that replaces the existing EXE-based installer. To upgrade using this new installer, simply download and install the new one. See [Install Using the Current User Installer](#).
- Support for "upsert" mode on pending create rows. Oracle business object REST API services may support an optional updating or creating ("upsert") mode that you can use when uploading changes to your REST service. See [Upload Changes Using Upsert Mode](#).
- Support for showing REST warning messages in addition to error messages in the Status Viewer. Oracle business object REST API services support the inclusion of warning messages in the response payload to a REST request. The add-in captures warnings for updates, creates, and custom actions and displays them in the Status Viewer. See [Upload Changes](#).
- Support for displaying the response to a custom action. After Upload (for table rows) or Perform Action (for the form row), the add-in captures the result, if any, from the response payload and displays it in the Status Viewer. See [Upload Changes for Custom Actions](#).
- Support for configuring the bind parameters for a descriptive flexfield (DFF) list of values. The add-in currently only supports configuring the polymorphic bind parameters. Other details of the list of values for DFFs must come from the metadata and are not configurable. Configure bind parameters using the Business Object Field Editor. See [Configure the Bind Parameters for a Descriptive Flexfield's List of Values](#).
- Support for omitting fields with empty values during upload. Request payloads for Create and Update operations normally includes a value for every column in the table (except those for read-only fields and custom action payload fields) even if the value is empty or null. You can now configure the add-in to omit null values from the payload. See [Omit Empty Values During Upload](#).
- Support for creating a list of values that displays multiple columns in the search-and-select window. For example, you might want to create a list of values for a "manager" column that displays both first and last names. When a user selects a cell in this column, the search-and-select window shows first and last names in separate columns. See [Configure a List of Values with Multiple Display Fields](#).
- Support for adding tooltips to custom action fields. See [Edit Custom Actions](#).

Known Limitations

- Review [REST Service Support Limitations](#) to understand the limitations around data type support.
- Review [Notes and Limitations for Lists of Values](#) to understand limitations around using lists of values.
- Review [Custom Action Limitations](#) to understand the limitations of custom action support.
- Workbooks created or modified with version 2.7 are generally considered to be incompatible with prior versions of the add-in.

3

Introduction to Oracle Visual Builder Add-in for Excel

Oracle Visual Builder Add-in for Excel integrates Excel spreadsheets with REST services to retrieve, analyze, and edit business data from the service. You download your data to an Excel spreadsheet, work with it, and upload your changes back to the service.

Key Concepts, Components, and Terms

Before you use Oracle Visual Builder Add-in for Excel, it helps to become familiar with these key concepts, components, and terms.

Term	Description
Integrated workbook	An Excel workbook configured to work with one or more business objects.
Business object	A resource - like a purchase order or invoice - that has fields to hold your application's data. A business object includes a collection path, an item path, a set of fields, and other properties.
Business object catalog	A set of one or more business objects with a common host and base path.
Collection path	A service path (endpoint) that can be used to fetch multiple rows of data from the business object and/or to perform operations on the collection.
Item path	A service path (endpoint) that can be used to fetch, or operate on, a single row from the business object.
Layout	A way to display a business object in an Excel worksheet. Each worksheet supports one of two layouts: Table or Form-over-Table. Layouts are created by workbook developers and are visible to business users in published workbooks.

Installation

Install the latest version of Oracle Visual Builder Add-in for Excel available from the [Downloads page](#) on Oracle.com.

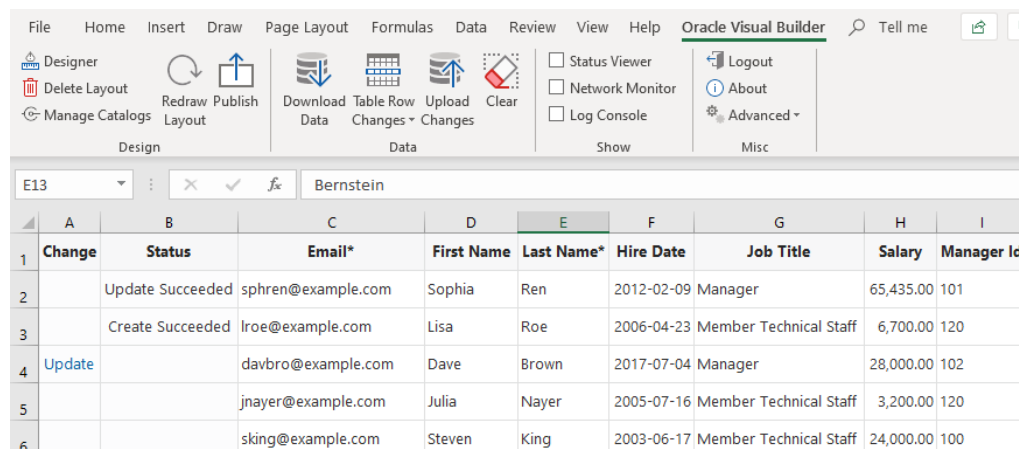
Before you begin, ensure that your environment meets the requirements for installation. See [Software Dependencies](#) and [Supported Platforms](#) for more details.

When you are ready, install the add-in, as described in [Install Oracle Visual Builder Add-in for Excel](#).

Next Steps

After you install the add-in, a new **Oracle Visual Builder** ribbon tab appears in Microsoft Excel. As a workbook developer, you use the options in this ribbon tab to configure a worksheet to integrate with a service and download data to a data table that you create in the worksheet. Once the data table is created and populated with data, you can review, modify, and create data, then upload your changes to the service.

This image shows a worksheet that is integrated with an REST service that manages employees:



	A	B	C	D	E	F	G	H	I
1	Change	Status	Email*	First Name	Last Name*	Hire Date	Job Title	Salary	Manager Id
2		Update Succeeded	sphren@example.com	Sophia	Ren	2012-02-09	Manager	65,435.00	101
3		Create Succeeded	lroe@example.com	Lisa	Roe	2006-04-23	Member Technical Staff	6,700.00	120
4	Update		davbro@example.com	Dave	Brown	2017-07-04	Manager	28,000.00	102
5			jnayer@example.com	Julia	Nayer	2005-07-16	Member Technical Staff	3,200.00	120
6			sking@example.com	Steven	King	2003-06-17	Member Technical Staff	24,000.00	100

Here are the high-level steps you'll need to follow to create a similar data table in a worksheet:

1. In the **Oracle Visual Builder** tab, click **Designer**.
2. Provide the URL of a service description that complies with the OpenAPI specification.
3. Pick a business object.
4. Download data.

Review subsequent sections in this guide to understand available layout types and other add-in capabilities. For Excel specifications and limits, see [Microsoft documentation](#).

4

Install Oracle Visual Builder Add-in for Excel

To install Oracle Visual Builder Add-in for Excel, download and run one of the two available installers. You can download these installers from the Oracle [Downloads](#) page.

There are two available installers for the add-in: a "Current User" installer and an "All Users" installer. Use the Current User installer to install the add-in on your local desktop for your own use. The All Users installer is intended for IT administrators.

The add-in runs in Excel on a Windows environment and requires some additional Microsoft components. Check out the [Software Dependencies](#) topic for details.

Installer Types

The Current User installer installs the add-in for the current user's Windows profile only. If there are other people using this computer, they will need to install the add-in for their own Windows profile.

When you use the All Users installer, the add-in is installed in the Programs Files folder and is available for all users on the target Windows machine.

Refer to this table for a full comparison of the two installers:

Comparison	Current User	All Users
File name	vbafe-installer-current-user.msi	vbafe-installer-all-users.msi
Installation Location	Current Windows user profile	Program Files
Target Audience	Excel users	IT administrators
Windows Registry Entries	HKEY_CURRENT_USER	HKEY_LOCAL_MACHINE
Elevated Privileges	Not required	Required

Install Using the All Users Installer

To install Oracle Visual Builder Add-in for Excel for all users, download the All Users installer from the Oracle [Downloads](#) page and run the installer.

If any required software is missing, the installation terminates without installing the add-in. Refer to [Software Dependencies](#) for details including information on how to check for and install required components.

You must have elevated privileges for this installation.

The add-in includes designer tools for developing workbooks by default. If the user doesn't need these tools, you can disable them when you install the add-in. If these tools are needed later on, simply rerun the installer and enable them.

This installation is available to all users on the Windows machine. You do not need to install the add-in separately for each user profile.

1. Double-click the `vbafe-installer-all-users.msi` file that you downloaded previously to launch the installation wizard.

If you are not logged in with elevated privileges, you'll be prompted to provide credentials with elevated privileges.

2. To install the add-in without the available developer tools, click **Developer Options** and select **Disabled**.

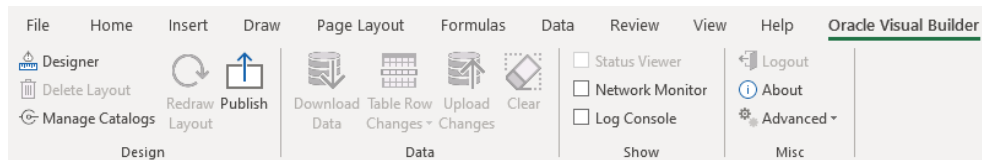
If you need to enable these tools after initial installation, re-run the installer, choose the option to repair your installation, and select **Enabled**.

3. Click **Install** to install the add-in.

When the installation is complete, click **Close**.

4. Start Excel and open a new workbook.

A new Oracle Visual Builder ribbon tab appears, with commands to integrate the Excel spreadsheet with a REST service. If you disabled the design tools during installation, you won't see the Design options in the first pane on the ribbon.



Install Using the Current User Installer

To install Oracle Visual Builder Add-in for Excel for the current user's profile, download the Current User installer (`vbafe-installer-current-user.msi`) from the Oracle [Downloads](#) page and run the installer.

If any required software is missing, the installation terminates without installing the add-in. Refer to [Software Dependencies](#) for details including information on how to check for and install required components.

The add-in includes designer tools for developing workbooks. These tools are included by default. If you are an application developer, make sure to install the designer tools with the add-in. If you need these tools but don't have them, simply rerun the installer and enable the designer tools.

This installation is specific to the current Windows user profile. If multiple users on a Windows machine need the add-in, consider using the All Users installer instead. See [Install Using the All Users Installer](#).

Note:

Running both installers on the same machine is not recommended. If you do install the add-in using both installers, Excel loads the add-in installed in the Program Files folder ("All Users") and not the version in the current user's Windows profile ("Current User"). If you decide to switch the installation type, the best practice is to uninstall the previous add-in installation type first to avoid confusion.

1. Sign in to the Windows user profile that will be using the add-in with Excel.

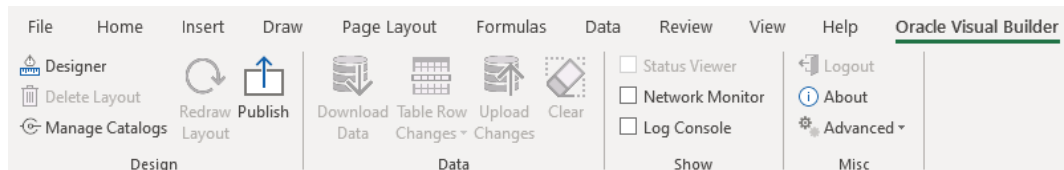
- Quit Excel before you begin installation.
- Double-click the `vbafe-installer-current-user.msi` file that you downloaded previously to launch the installation wizard.
- To install the add-in without the available developer tools, click **Developer Options** and select **Disabled**.

 **Tip:**

If you need these tools after initial installation, re-run the installer, choose the option to repair your installation, and select **Enabled**.

- Click **Install** to install the add-in.
When the installation is complete, click **Close**.
- Start Excel and open a new workbook.

A new Oracle Visual Builder ribbon tab appears, with commands to integrate the Excel spreadsheet with a REST service. If you disabled the design tools during installation, you won't see the Design options in the first pane on the ribbon.



When you first run Excel after the current-user installation, you may be prompted to confirm the installation of the "Office customization". This prompt generally appears the first time for each profile and whenever the digital certificate (used to sign the add-in) is changed.

Run the Installer from the Command Line

Since both installers are standard Windows Installer Packages (MSI), you can run `msiexec` on the command line to install Oracle Visual Builder Add-in for Excel.

`msiexec` includes a number of options that let you control or customize your installation. See [Standard Installer Command-Line Options](#) on the Microsoft web site.

Examples

The following example shows a silent installation with logging enabled:

```
msiexec /package vbafe-installer-all-users.msi /quiet /log vbafe-install-log.txt
```

The following example shows a silent uninstall:

```
msiexec /uninstall vbafe-installer-all-users.msi /quiet
```

Unsupported Options

Not all `msiexec` options are relevant or supported for Add-in installation. Unsupported options include `/j` and `/a`.

Enable Designer Tools

The installers define a public property, "DESIGNER", that can be used to enable or disable designer tools during command-line installation. To disable designer tools from the command-line, set `DESIGNER=0`. Refer to the `msiexec` documentation for details on how to set public properties during installation.

When performing an upgrade or repair, the installer uses the previously set value if this property is omitted.

Upgrade to the Latest Version

To take advantage of all the latest Oracle Visual Builder Add-in for Excel features, make sure you are running an up-to-date version of the add-in. To upgrade to a new version, simply download and run the installer.

For recommendations on when you should upgrade, check the [Upgrade Policy](#).

To check for new versions, select **Check for Updates** from the **Advanced** drop-down on the Oracle Visual Builder ribbon. If a newer version is available, you'll be prompted to open the Oracle [Downloads](#) page in your browser.

You do not need to uninstall the previous version unless the installer instructs you to do so.

If you want to upgrade from one installation type to the other, uninstall the existing instance of the add-in first.

To determine which type of installation you have, run a diagnostic report and check the "Code Base" property under Properties. If the path is under the current user profile, the add-in was installed using the Current User installer. If the path is under Program Files, the add-in was installed using the All Users installer.

To run the diagnostic report, select **Diagnostic Report** from the **Advanced** drop-down on the Oracle Visual Builder ribbon.

To ensure a clean upgrade, follow these instructions when upgrading your installation.

1. Before you upgrade the add-in:
 - a. Upload any pending changes using the current add-in version.
 - b. Save changes in open workbooks, then close Excel.
2. Run the installer for the new version and follow the instructions in the wizard. The installer automatically replaces the previous version with the new version.
3. After you upgrade:
 - a. Launch Excel to complete any final installation steps.
 - b. Open your integrated workbook.
 - c. Clear any layouts of old data and download data again as required.

Upgrade Policy

When a new version of Oracle Visual Builder Add-in for Excel is released publicly, customers should upgrade at their earliest opportunity to take advantage of the latest fixes and improvements.

However, IT departments may need some time to evaluate and deploy a new version to business user desktops. The Oracle upgrade policy allows some flexibility to accommodate this process.

Customers may defer the add-in upgrade by up to six months from the time a new version becomes publicly available. After six months, the old version of the add-in is no longer supported and must be upgraded. So, if the latest version is released on May 15, customers may defer upgrading from the previous version until November 15 at the very latest. After this date, upgrade is mandatory.

Limitations

This upgrade deferral policy is limited in several ways:

- **Fixes for defects:** To obtain a fix for an issue, it is necessary to install a new version of the add-in.
- **Troubleshooting:** When investigating a potential problem, one step in the troubleshooting process is to upgrade to the latest version. Deferral is not an option during the support process.
- **New Workbooks:** Frequently, new versions of the add-in offer new functionality. If your business users have integrated workbooks that leverage new functionality available only in the latest version of the add-in, then add-in upgrade becomes mandatory since the new functionality is not available in the previous version of the add-in.

Uninstall the Oracle Visual Builder Add-in for Excel

Uninstall the add-in from the Windows Settings app.

If you're uninstalling the Current User version, sign in to the Windows user profile where the add-in is installed.

1. From the Start Menu, select **Settings**, then **Apps**.
2. From the Apps & Features page, select Oracle Visual Builder Add-in for Excel from the list of programs.

Tip:

Type "Oracle" in the search box to filter on Oracle applications.

3. Click **Uninstall** and follow the instructions.
4. If you have performed a current user installation for multiple Windows user profiles and you want to uninstall them all, log in to each profile and repeat these steps.

Software Dependencies

Oracle Visual Builder Add-in for Excel runs in Excel on a Windows environment and requires some Microsoft components to operate. Besides the required components, there is also optional software you may want to install depending on your needs.

This topic lists the required and optional software. For information about supported Excel and Windows versions, see [Supported Platforms](#).

Required Software

The add-in requires the Microsoft components listed here.

Components	Notes
Microsoft .NET Framework 4.7 or later	The .NET Framework 4.8 is included with the May 2019 update of Windows 10
Microsoft Visual Studio 2010 Tools for Office (VSTO) Runtime	The VSTO runtime is included with most recent versions of Excel



Note:

If you are running recent versions of Windows and Excel, you may already have these components.

By using the various Microsoft components, you are subject to Microsoft data collection and privacy policies. Be sure to review them carefully before proceeding. Oracle is not responsible for the behavior of Microsoft components. See the [Microsoft Privacy Statement](#).

The installer checks your system for required software and quits without installing the add-in if any required software is missing. Run the [Health Check tool](#) to ensure you have all required software and versions. If you are missing these required components, follow the instructions in the Health Check tool to install the required software.

Optional Software

The add-in supports the use of Microsoft Edge WebView2 as an embedded browser for displaying log-in pages. If this component is not installed, the add-in uses Internet Explorer components instead.

To take advantage of the WebView2 embedded browser, download one of the "evergreen" installers and run it. You can download the software from the Microsoft Edge WebView2 page here: <https://developer.microsoft.com/en-us/microsoft-edge/webview2/#download-section>.

See [Embedded Browsers](#).

Supported Platforms

Oracle Visual Builder Add-in for Excel runs in Excel on Windows. This section provides details on which versions are supported. Review this topic carefully to make sure your configuration is supported.

Microsoft Excel

Version	Support Expires
Excel 2016	2025
Excel 2019	2025
Excel for Microsoft 365 (desktop installation only)	Refer to Microsoft's Modern Lifecycle Policy .

 **Note:**

Oracle recommends the 32-bit version of Excel whenever possible, as the 64-bit editions have been found to be less stable. Refer to the Microsoft article, [64-bit editions of Office 2013](#).

The following editions of Excel don't support VSTO/COM add-ins and are, therefore, *not supported*:

- Excel Online
- Excel for Microsoft 365 installed from the Microsoft Store

Microsoft 365 "Beta" and "Preview" update channels provide experimental versions of Excel. Oracle cannot provide support for software from these channels.

Microsoft Windows

The add-in is supported on Windows 10.

 **Note:**

Please ensure that your version (or "feature update") of Windows 10 is still in service according to the Microsoft policy. Oracle cannot provide support for versions beyond Microsoft's end of servicing date. See [Windows 10 Lifecycle Policy](#).

The add-in isn't supported on:

- Windows server editions
- MacOS, iOS, Linux, or any other operating system

Virtual Desktop Infrastructure

The Visual Builder Add-in for Excel was designed and tested to work on regular Windows desktop and laptop computers. A number of vendors offer virtual desktop infrastructure (VDI) solutions to provide virtual machines that mimic a standard Windows desktop environment.

Some VDI implementations reproduce the standard desktop very closely whereas others are significantly different. Refer to [Desktop Virtualization](#) for an overview of this technology.

Oracle does not test the add-in with VDI products formally. VDI products are not supported on an official basis.

Even though it may be possible to run the add-in on some virtual machines successfully, doing so is strictly at your own risk. If you run into problems, you'll have to reproduce the problem on a standard desktop computer before Oracle Support can assist you. If the problem appears to be a VDI issue, contact your VDI vendor for help.

Microsoft Application Virtualization

Microsoft Application Virtualization (App-V) is not supported in any version at this time.

Notes

- If a software application or version isn't listed here, it is not supported.
- Later versions of Excel and Windows are not automatically supported when they become available. Support can only be added through an enhancement request.
- Oracle doesn't support the add-in on unsupported software. If a vendor drops support for a given software version, Oracle support ends at the same time. This is true even if the software is listed here.
- Microsoft may not support all possible combinations of their software and operating systems listed here. If Microsoft doesn't support a given combination, Oracle doesn't either. If you're unsure if your versions of Excel and Windows are compatible, consult your software or operating system documentation.
- Support expiration dates are determined by Microsoft support policies. See [Microsoft Lifecycle Policy](#).

5

Create Layouts in an Excel Workbook

To integrate a workbook with a REST service, create a layout for a business object on a new worksheet and link the layout to the REST service's service description. You can then download data for the business object to the layout and start working with it.

Layouts

Oracle Visual Builder Add-in for Excel lets you create different layouts to work with data in an Excel worksheet. Layouts are a way to display a business object in an Excel worksheet. Each worksheet supports one of two layouts: **Table** or **Form-over-Table**.

- Use a Table layout to view and edit data from a REST service in a tabular format. Here's a worksheet showing employee data in a Table layout:

	A	B	C	D	E	F	G	H	I	J	K	L
1	Change	Status	First Name	Last Name*	Email*	Hire Date	Job Title	Salary	Manager Id	Department*	Id*	Key
2			Sophie	Ren	sphren@example.com	2012-02-09	Manager	65,435.00	101	Research	2
3			Dave	Brown	davbro@example.com	2017-07-04	Manager	28,000.00	102	Accounting	3
4			Julia	Nayer	jnayer@example.com	2005-07-16	Member Technical Staff	3,200.00	120	Accounting	6
5			Steven	King	sking@example.com	2003-06-17	Member Technical Staff	24,000.00	100	Accounting	7

- Use a Form-over-Table layout when a parent-child relationship exists in the business objects used by your web application. Here's a worksheet showing purchase order and line data in a Form-over-Table Layout, where the parent object's data (Purchase Orders) is shown in the form and the child object's data (Lines) is shown in the table:

	A	B	C	D	E
1	Purchase Orders				
2	Order	1004976	Status	Open	
3	Description		Document Style	Purchase Order	
4	Procurement BU	Vision Operations	Requisitioning BU	Vision Operations	
5	Buyer	Furey,Clare	Purchase Order Email Sender		
6	Currency	US Dollar	Supplier	CV_SuppA00	
7	Supplier Site	CVSuppA00Site01	Shipping Method	Airborne	
8	Pay on Receipt	TRUE			
9					
10					
11	Change	Status	Line	Line Type	Item
12			1.00	Goods	zPOR-SCH005_Dell_Inspiron_1525

You can create one layout per worksheet in your Excel workbook. Layouts are created by workbook developers and are visible to data entry users in their workbooks.

Service Descriptions

To create a layout in the workbook, the REST services that you use must provide a service description that complies with the OpenAPI specification. The service description can be a URL or a local file. For Oracle business object REST API services, the URL typically includes a describe path segment, as in `https://my-service-host/fscmRestApi/resources/latest/invoices/describe`. For Oracle REST Data Services (ORDS), the URL may be similar to `https://host/ords/great_app/open-api-catalog/employees/`.

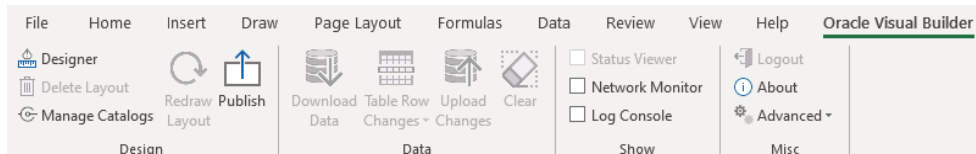
You can provide the service description document when you create a layout by clicking **Designer** in the Oracle Visual Builder tab, as described in subsequent sections. You can also provide the service description document by clicking **Manage Catalogs** in the Oracle Visual Builder tab (see [Edit Service Descriptions and Business Objects](#)).

The service description that you provide helps generate a **business object catalog** for the workbook. A business object catalog is essentially a list of business objects. As a workbook developer, you can edit portions of the business catalog as desired, or use it as is to create layouts.

Create a Table Layout in an Excel Workbook

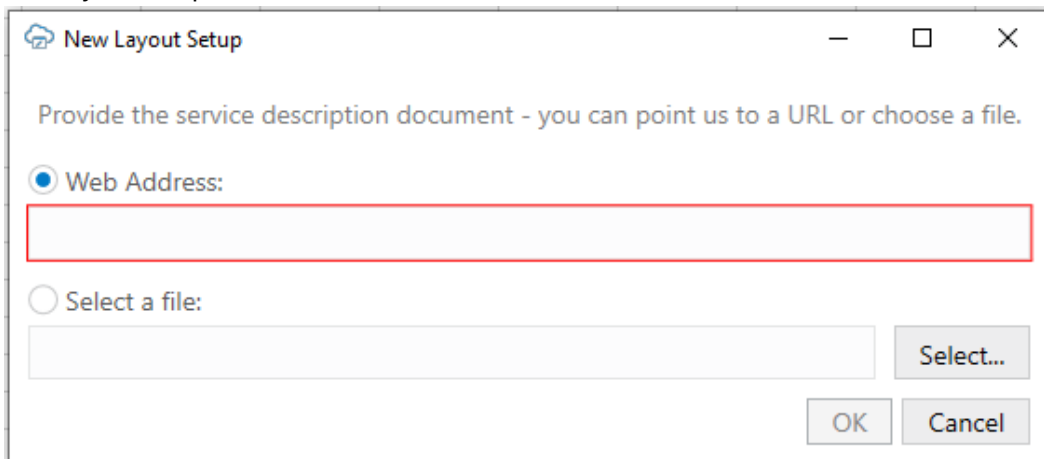
Create a Table layout in the Excel worksheet when you want to view and edit data from a REST service in a tabular format.

1. Create a blank Excel workbook using the standard `.xlsx` file format or the macro-enabled `.xlsm` format. Other Excel formats (`.xls` and so on) are not supported.
2. In the Excel ribbon, select the **Oracle Visual Builder** tab.



3. Click the cell where you want to locate the data table.
4. In the Oracle Visual Builder tab, click **Designer**.
5. When prompted, provide the service description document. Use the **Web Address** option (the default) if you access the service description from a URL. Note that you can't provide a data URL (a URL that returns data) as the starting point to creating a layout. Use the **Select a file** option if the service description document is a local

file on your computer.

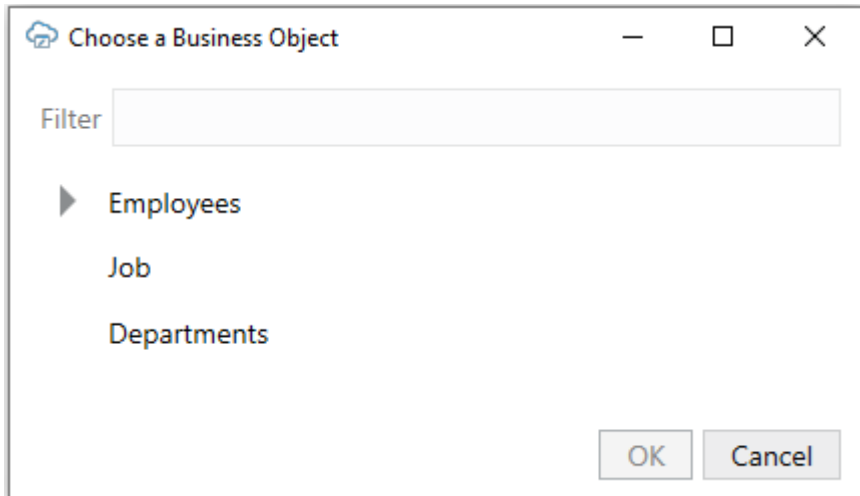


The 'New Layout Setup' dialog box has a title bar with a home icon, the text 'New Layout Setup', and standard window controls. Below the title bar is the instruction: 'Provide the service description document - you can point us to a URL or choose a file.' There are two radio buttons: 'Web Address:' (selected) and 'Select a file:'. The 'Web Address:' option has a red rectangular border around its text input field. The 'Select a file:' option has a text input field and a 'Select...' button to its right. At the bottom right are 'OK' and 'Cancel' buttons.

 **Tip:**

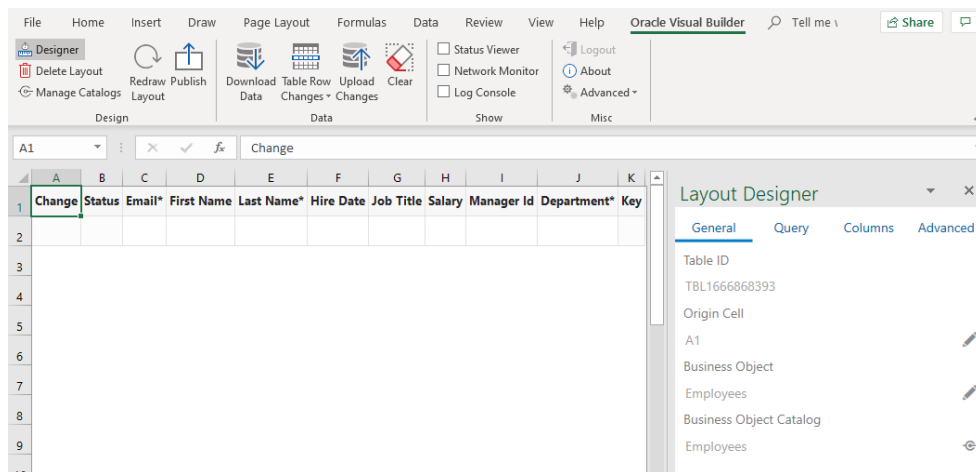
If you are working with Oracle business object REST API services, the URL for the service description usually ends with `/describe`, for example, `https://my-service-host/fscmRestApi/resources/latest/invoices/describe`.

If multiple business objects are found, the Choose a Business Object window prompts you to choose from the available business objects.



The 'Choose a Business Object' dialog box has a title bar with a home icon, the text 'Choose a Business Object', and standard window controls. Below the title bar is a 'Filter' text input field. Underneath is a list of business objects: 'Employees', 'Job', and 'Departments'. Each item has a right-pointing triangle icon to its left. At the bottom right are 'OK' and 'Cancel' buttons.

The add-in creates a Table layout in the Excel workbook that includes column headers and a placeholder data row. The Layout Designer opens in the Excel Task Pane.



6. Click the **Columns** tab in the Layout Designer to add or remove columns, or change the order in which columns appear. When a data table is created, most (but not all) fields are added as columns. Click the Manage Columns button (+) to add or remove columns as needed.
You can also modify a field associated with a column to, for example, show a tooltip or display a list of input values for your business users. To update a field, select it from the Table Columns table and click the Edit button (✎) to open the Business Object Field Editor. To add a tooltip, click the **General** tab and type help text in the **Tooltip** property. The tooltip is displayed when the user selects the table column header. To create a list of values, see [Configure a List of Values for a Business Object Field](#).
7. After making changes, click **Redraw Layout** in the ribbon tab to see your changes reflected in the worksheet.
8. Optionally, configure the workbook further to limit the data that the add-in downloads, as described in [Configure Search Options for Download](#).

The workbook is now complete and ready to be published. At this point, it's a good idea to perform various data operations, such as download, update, and upload, to test the workbook before you publish and distribute it to users. For information on managing data, see View and Edit Data Using an Excel Workbook in *Managing Data Using Oracle Visual Builder Add-in for Excel*.

Work with Service Path Parameters in a Table Layout

Some service paths include path parameters. The add-in provides support for configuring a Table layout using a parameterized service path. It automatically extracts the path parameters and prompts the user to provide the corresponding values at download time.

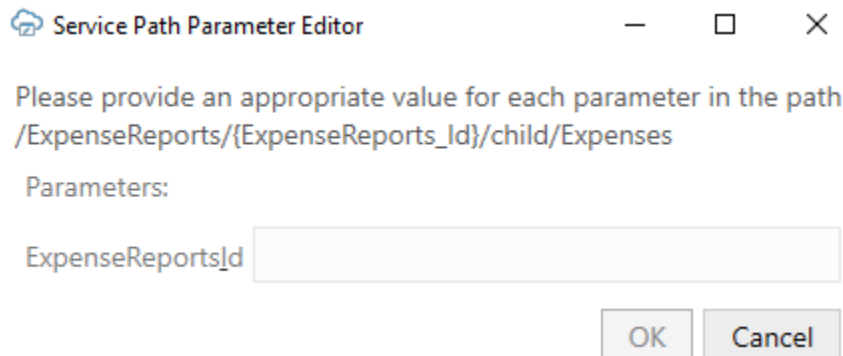
To configure a Table layout with a parameterized service path, first provide an OpenAPI-compliant service description. When prompted, choose a child business object or any parameterized path from the business object picker.



Tip:

When working with Oracle business object REST API services, you should start with the web address to the parent business object description (and not the child address). For example, for a parameterized service path such as `/ExpenseReports/{ExpenseReports_Id}/child/Expenses/`, provide the address to the `ExpenseReports` description (not `Expenses`). Oracle business object REST API services cannot provide OpenAPI service descriptions for parameterized service paths.

Complete the layout configuration. When users click **Download Data** in the Oracle Visual Builder tab, the add-in displays the Service Path Parameter Editor where users provide the required path parameter values that enables the download of data to complete.



Path parameters of type string or integer are supported; other data types are not supported. For string-typed path parameters, values that users enter in the Service Path Parameter Editor are used verbatim when the add-in constructs the request to the service. For integer-typed values, certain culture-specific formatting is removed (for example, commas for thousands separators, parentheses for negative). In all cases, the values used on the URL path are not URL-encoded, so the values entered must be acceptable by the REST service.

Here is an example of a service path with an embedded parameter:

```
/ExpenseReports/{ExpenseReports_Id}/child/Expenses/
```

Note `{ExpenseReports_Id}` is in the middle of the service path.

Using the Service Path Parameter Editor, you provide the proper value for `{ExpenseReports_Id}`, for example, 123456, which results in the add-in using the following path:

```
/ExpenseReports/123456/child/Expenses/
```

Accessing this service path will provide all the expenses for expense report 123456.

The Service Path Parameter Editor does not validate the value(s) that users enter. The value(s) that users provide must be valid. If the path includes multiple embedded parameters, the Service Path Parameter Editor prompts the user to provide a value for each embedded parameter.

The add-in remembers the values provided at download time. These values are used again at upload time to construct the upload requests. If you upload without having done a previous

download (for example, when exclusively creating new rows), you'll be prompted for the path parameter values at the beginning of the upload.

Create a Form-over-Table Layout in an Excel Workbook

You can create a Form-over-Table layout in an Excel worksheet when a parent-child relationship exists in the chosen service.

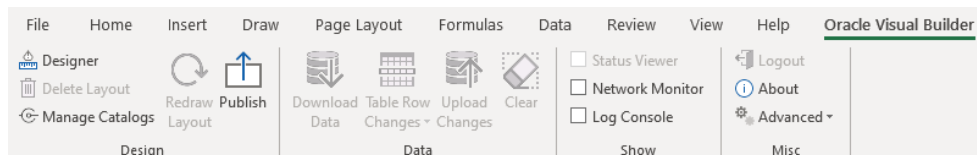
Note:

A parent-child relationship at the service level requires:

- A parent service path, for example, `fscmRestApi/resources/latest/invoices`
- A child service path with a parameter, for example, `fscmRestApi/resources/latest/invoices/{invoice-id}/child/invoicelines`

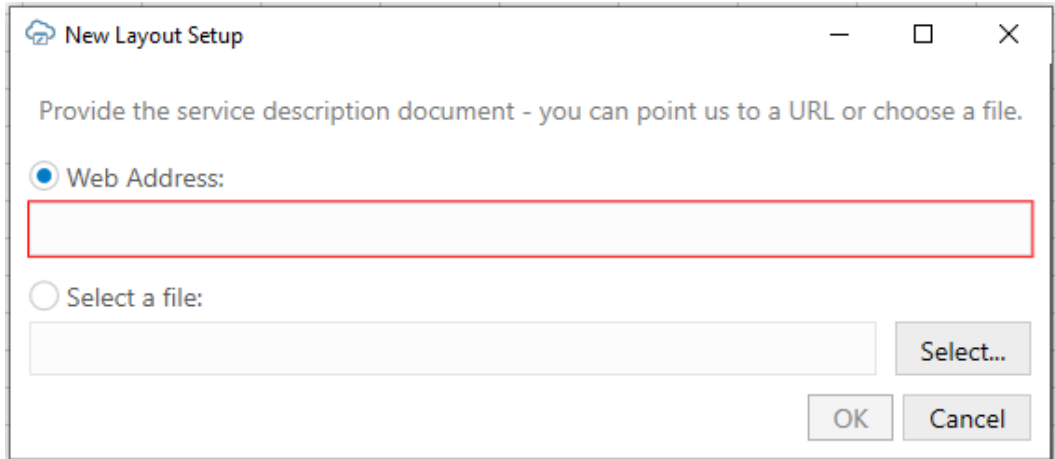
In your workbook, both business objects must be declared in the same catalog. Continuing our example, `invoicelines` must appear as a child of `invoices`. The add-in uses the primary ID of the parent row and inserts it into the child's path. The workbook service must be able to support operations on these paths.

1. Create a blank Excel workbook using the standard `.XLSX` file format or the macro-enabled `.XLSM` format. Other Excel formats (`.XLS` and so on) are not supported.
2. In the Excel ribbon, select the **Oracle Visual Builder** tab.



3. Click the cell where you want to locate the form and table.
4. In the Oracle Visual Builder tab, click **Designer**.
5. When prompted, provide the service description document. Use the **Web Address** option (the default) if you access the service description from a URL. Note that you can't provide a data URL (a URL that returns data) as the starting point to creating a layout. Use the **Select a file** option if the service description document is a local

file on your computer.

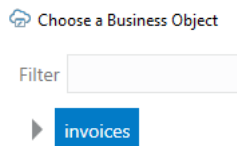


The 'New Layout Setup' dialog box prompts the user to provide a service description document. It has two radio button options: 'Web Address:' (selected) and 'Select a file:'. The 'Web Address:' option has a red rectangular box around its text input field. The 'Select a file:' option has a text input field and a 'Select...' button. At the bottom right, there are 'OK' and 'Cancel' buttons.

 **Tip:**

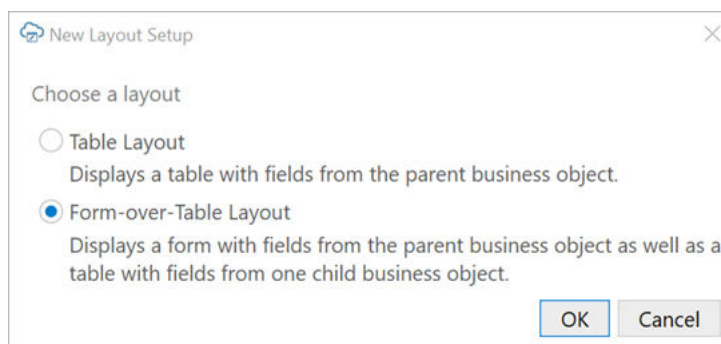
If you are working with Oracle business object REST API services, the URL for the service description usually ends with `/describe`, for example, `https://my-service-host/fscmRestApi/resources/latest/invoices/describe`.

When your OpenAPI service description includes multiple business objects, a window similar to the following prompts you to pick the business object you want to use in the Form-over-Table layout:



The 'Choose a Business Object' dialog box features a 'Filter' text input field. Below it, a blue button labeled 'invoices' is highlighted with a white arrow pointing to it from the left.

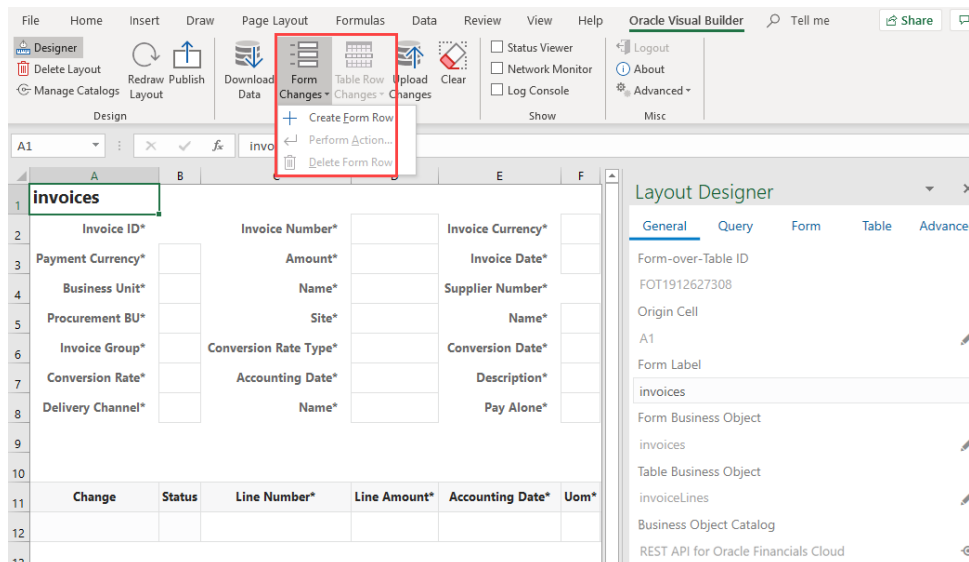
6. Choose a business object where a parent-child relationship exists (such as `invoices`), choose **Form-over-Table Layout** in the New Layout Setup window that appears, and click **OK**.



The 'New Layout Setup' dialog box shows two radio button options for layout selection: 'Table Layout' (unselected) and 'Form-over-Table Layout' (selected). Each option includes a brief description of what it displays. At the bottom right, there are 'OK' and 'Cancel' buttons.

If there's more than one child business object, select the child to use in the Form-over-Table layout and click **OK**. This example uses the `invoiceLines` child business object.

The add-in creates a Form-over-Table in the Excel worksheet and opens the Layout Designer that you use to modify the newly-inserted form and table, as shown here:




If the parent business object (in this case, `invoices`) supports a create action, a **Create Form Row** option appears in the Form Changes menu, as shown in the image. Use this option to create a new form row during your next upload (see *Create a Parent Row in a Form-over-Table Layout in [Managing Data Using Oracle Visual Builder Add-in for Excel](#)*).

7. Customize the form and table by modifying the automatically populated properties in the Layout Designer. Use the Form page to add, remove, or rearrange fields on the form. The Table page includes the same functionality for the table under the form.

Tip:

From the Form or Table page of the Layout Designer, right-click a field or column to see choices for changing the order.

To modify a form field or table column, select it from the table and click the Edit button () to open the Business Object Field Editor. From here, you can add a tooltip or a list of values to a field. To add a tooltip, click the **General** tab and type help text in the **Tooltip** field. The tooltip is displayed when the user selects a form field label or table column header. To create a list of values, see [Configure a List of Values for a Business Object Field](#).

8. Click **Redraw Layout** to see changes you make in the Layout Designer reflected in the form and table.
9. Optionally, configure the workbook to limit the data that the add-in downloads, as described in [Configure Search Options for Download](#). If, for example, you do not specify a value for the Search field or Row Finder property, the add-in downloads the first parent item it encounters in the REST service to the form, and the child items, if any, to the table.

Configuration is now complete and you can publish the workbook. At this point, it's a good idea to test the workbook before you publish and distribute it to users. For information on managing data in a Form-over-Table layout, see *Manage Data in Form-over-Table Layouts in Managing Data Using Oracle Visual Builder Add-in for Excel*.

Use Descriptive Flexfields in a Layout

Oracle Visual Builder Add-in for Excel supports the use of descriptive flexfields (DFF) in your layout's forms and tables. DFFs expand to display additional fields for tracking supplemental information. These fields are supported in REST services using Oracle ADF Business Components.

A DFF consists of a top-level field, or "discriminator", and a number of sub-fields called "segments". Segments may be global (available for all values in the discriminator) or context-sensitive (dynamic based on the value in the discriminator).

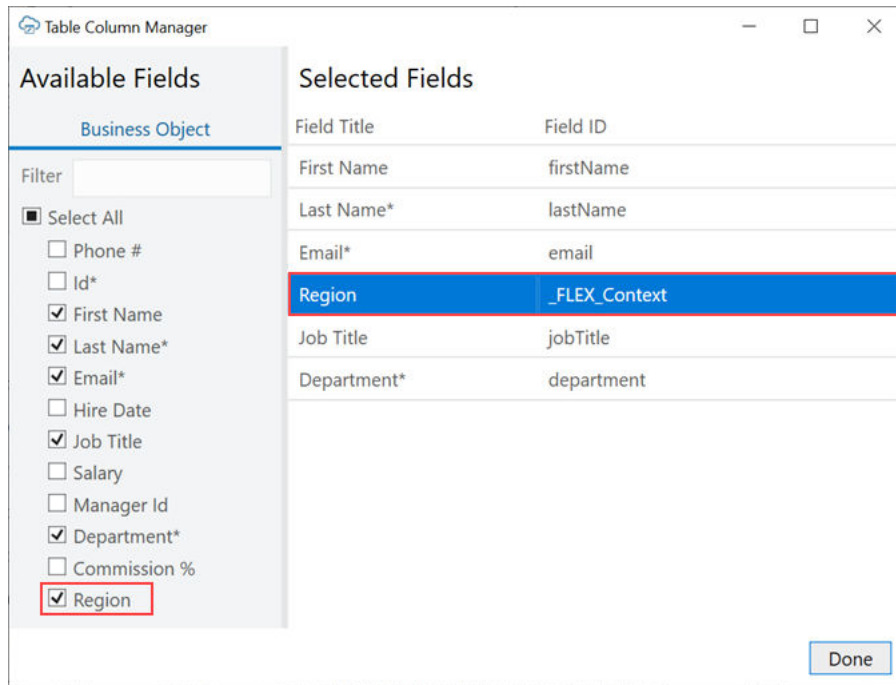
For example, an "Employees" REST resource may include a child DFF that defines regional information for employees. When added to a table, the DFF displays a Region column (for the discriminator) as well as additional columns for the DFF's segments. These columns could, for example, be used to capture extra regional information such as site and time zone. The DFF may also include context-sensitive columns such as a "Postal Code/Zip Code" column for employees in the North American region.

See [Use Descriptive Flexfields](#) for more information.

Add DFFs to Forms and Tables

Add DFFs to forms and tables using the Form Field Manager or Table Column Manager. You can add multiple DFFs and place them anywhere in the form or table.

1. Open the worksheet with the layout that you want to modify.
2. Click either the **Form** or **Columns** tab in the Layout Designer as needed.
3. Click the Manage Columns button (+) to add your DFF. Available DFFs are identified by the title of the discriminator field (in this case, "Region") and appear at the bottom of the list of available fields. The field ID is the discriminator field ID ("__FLEX_Context").
4. Select the DFF from the Available Fields list.



You can also change the order of fields in the form or table by dragging and dropping fields in the Selected Fields list.

- After you add a DFF to a layout, click **Redraw Layout** to expand the DFF. The associated segments appear in the following order: global segments, the discriminator, and context-sensitive segments.

Context-sensitive columns for all possible discriminator values are included in the table. However, only those cells in a row that are relevant to the value in the discriminator field are editable. All other context-sensitive cells are read-only (grayed out).

F	G	H	I	J	K	L	M	N	O	P	Q	R	S
Job Title*	Phone #	Email*	Site	Time Zone	Region	Zip Code	State	Postal Code	Province	Hire Date*	Salary	Manager Id	Departme
resident	515.123.4567	SKING	HQ	GMT-7	United States	12345	CA			6/17/2003 0:00	240,000.00		Executive
Administration VP	515.123.4568	NKOCHHAR	Burlington	GMT-4	United States	54321	MA			9/21/2005 0:00	170,000.00	100	Executive
Administration VP	515.123.4569	LDEHAAN	Toronto	GMT-4	Canada		A1A 1A1	Ontario		1/13/2001 0:00	172,000.00	100	Executive
Programmer	590.423.4567	AHUNOLD	Toronto	GMT-4	Canada		A1A 1A1	Ontario		1/3/2006 0:00	98,000.00	102	IT

In a form, the context-sensitive form fields relevant to the current discriminator value appear after the discriminator field. Context-sensitive fields that are not relevant are not included in the form. If you change the value in the discriminator field, the form automatically updates to include the relevant context-sensitive field for that value.

	A	B	C	D
1	Employee			
2	Phone #	515.123.4567	Id*	100
3	First Name	Steven	Last Name*	King
4	Email*	SKING	Hire Date*	6/17/2003 0:00
5	Job Title*	President	Salary	240,000.00
6	Site	HQ	Time Zone	GMT-7
7	Region	United States	Zip Code	12345
8	State	CA		
9				
10				
11	Change	Status	Phone #	First Name Last
12			515.123.4568	Neena Korch

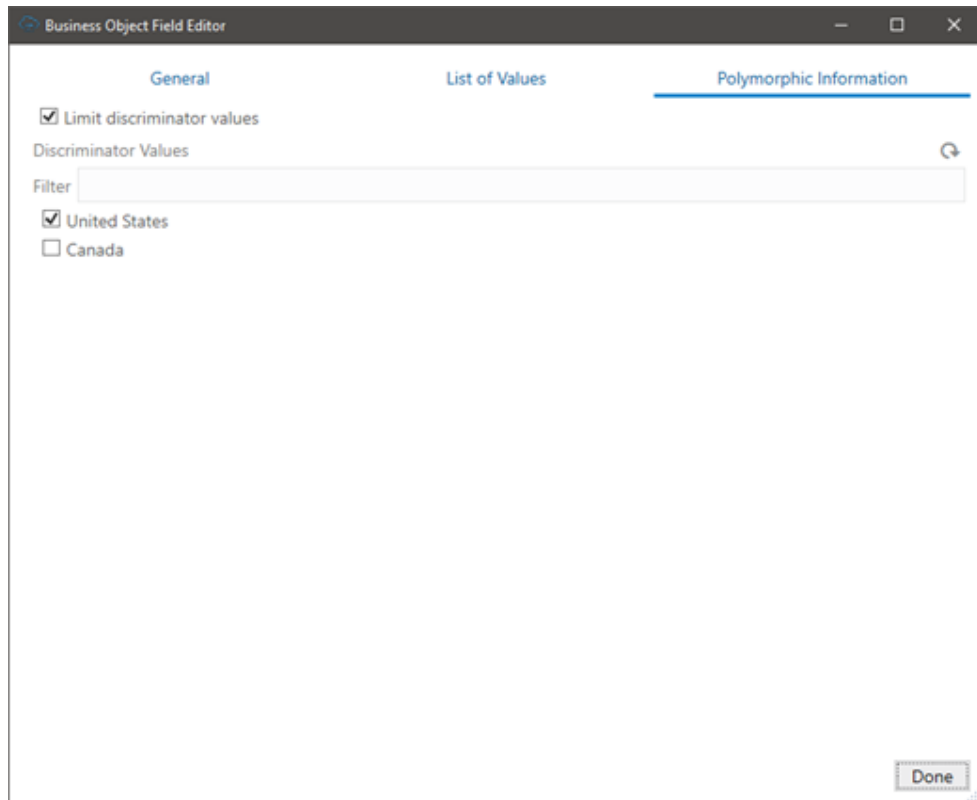
Showing and Hiding Context-Sensitive Columns in a Table Layout

Once you add a DFF to a Table layout, you can select which context-sensitive columns you want to display using the **Polymorphic Information** tab of the DFF's Business Object Field Editor.

In the case of the region DFF, you may choose to show regional information for US employees only.

To show or hide context-sensitive columns:

1. Open the worksheet with the layout that you want to modify.
2. From the Layout Designer, click the Edit icon (✎) next to the Business Object field.
3. From the Business Object Editor, click the **Fields** tab and then select the business object's field.
4. Click the Edit icon (✎) in the Business Object Editor to open the Business Object Field Editor.
5. From the **Polymorphic Information** tab, select **Limit discriminator values**.
6. To ensure you see all available discriminator values, click the Refresh icon (🔄) to fetch the latest DFF metadata from the service.
7. From the Discriminator Values list, select the discriminator values for the context-sensitive segment columns that you want to display. For example, to show zip code and state columns in your Table layout, select the **United States** check box.



- After you select or clear discriminator values, click **Redraw Layout** to show or hide the context-sensitive segment columns.

F	G	H	I	J	K	L	M	N	O	P	Q
Job Title*	Phone #	Email*	Site	Time Zone	Region	Zip Code	State	Hire Date*	Salary	Manager Id	Department
President	515.123.4567	SKING	HQ	GMT-7	United States	12345	CA	6/17/2003 0:00	240,000.00		Executive
Administration VP	515.123.4568	NKOCHHAR	Burlington	GMT-4	United States	54321	MA	9/21/2005 0:00	170,000.00	100	Executive
Administration VP	515.123.4569	LDEHAAN	Toronto	GMT-4	Canada			1/13/2001 0:00	172,000.00	100	Executive
Programmer	590.423.4567	AHUNOLD	Toronto	GMT-4	Canada			1/3/2006 0:00	98,000.00	102	IT

Refreshing DFF Metadata

Because DFFs are configurable by customers and subject to change, metadata stored in the workbook for the DFF may become stale.

To clear metadata when publishing a workbook, select **Clear all layouts** from the Publish Workbook window. The add-in refreshes the DFF metadata during the first download operation performed after you open the workbook.

Limitations

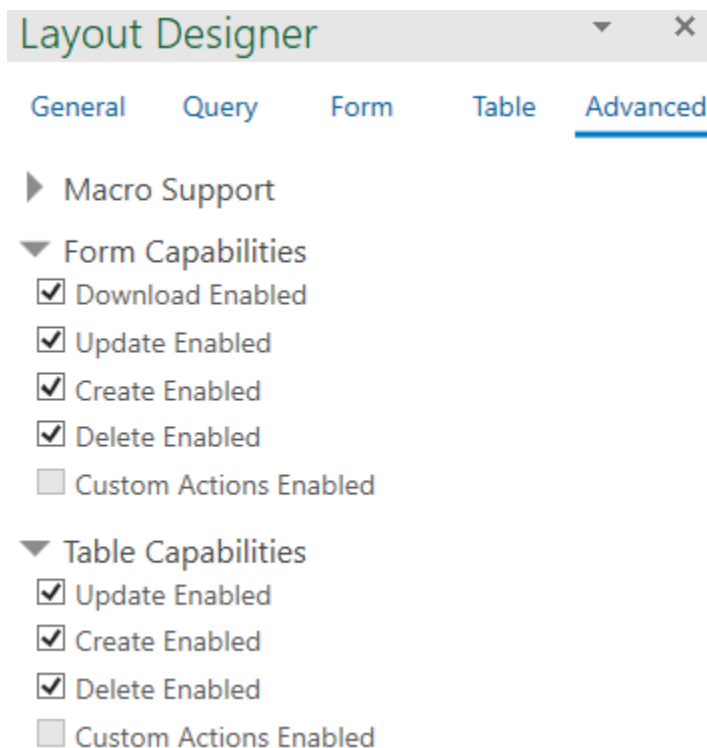
- A DFF child resource must have a one to one relationship with its parent resource.
- A DFF child resource can only contain a single discriminator field. Multiple discriminators are not supported.
- Changing the order of the global or context-sensitive segments is not supported.
- Binding layouts directly to a DFF resource is not supported. Layouts must be bound to the parent resource of the DFF resource.

- Before you can add a DFF to a table, it must expose polymorphic business object metadata. This means that the OpenApi3 describe document must contain a "discriminator" and "oneOf" syntax in the schema for the business object. "anyOf" polymorphic business object syntax is not supported.
- When a layout contains a DFF, the "fields" and "expand" query parameters should not be manually configured in the "Search Parameters" in the "Query" tab of the designer.
- Limiting discriminator values for a DFF displayed in a Table layout does not limit:
 - The set of values that can be chosen in a list of values (LOV) for a discriminator field
 - The values users can provide for a discriminator field

Manage Layout Capabilities

Each layout in Oracle Visual Builder Add-in for Excel enables you to perform various standard and custom actions in an Excel workbook, so long as the operation is supported by your service. You can enable or disable these supported capabilities to control their availability in a layout.

1. In the Excel ribbon, click **Designer** to open a workbook's Layout Designer.
2. Click **Advanced** to see the layout's capabilities. Here's an example of a Form-over-Table layout, indicating form capabilities and table capabilities:



3. Select or deselect options as required. If the business object doesn't support an action, it won't be available for selection (like **Custom Actions Enabled** in the example).
4. Click **Redraw Layout**.

6

Edit Service Descriptions and Business Objects

Oracle Visual Builder Add-in for Excel provides editors to modify a workbook's business object definitions.

Using the business object editors, you can change the URL of the host that provides access to the REST service your workbook uses.

Other examples of tasks that you can accomplish using these editors include:

- Edit field titles and tooltips
- Configure the [list of values](#) for a field
- Change when a field is editable
- Configure [pagination](#)
- Adjust the field data types (Advanced)

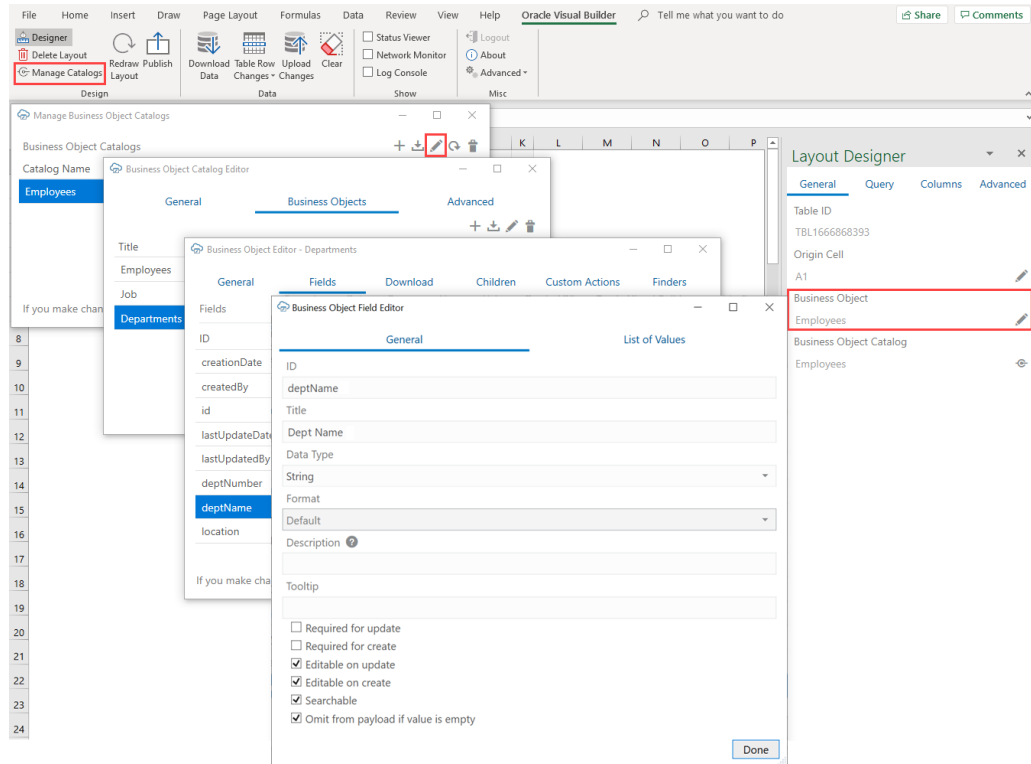
While you can provide the service description document when you [create a layout](#), using the Catalog editors lets you improve the business objects in a variety of ways to enhance overall user experience.



Note:

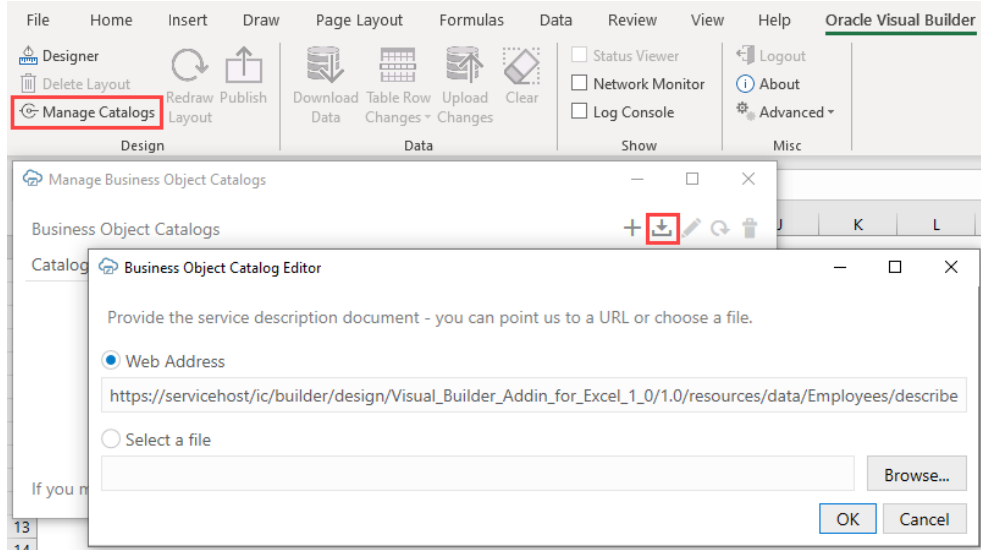
When you edit the details of business objects in your workbook, you're only telling the add-in how the service operates, you are not telling the service what it should do. Service behavior cannot be changed from the workbook. So any changes you make to the service description in the workbook must be compatible with the service.

Click **Manage Catalogs** from the Oracle Visual Builder tab to view the Catalog editors progressively. That is, you first access the editor that lets you edit the catalog, then access subsequent editors for business objects and business object fields that the catalog supports. You can also open the business object and business object field editors from the Layout Designer. Both options (using **Manage Catalogs** and the Layout Designer) are shown here:

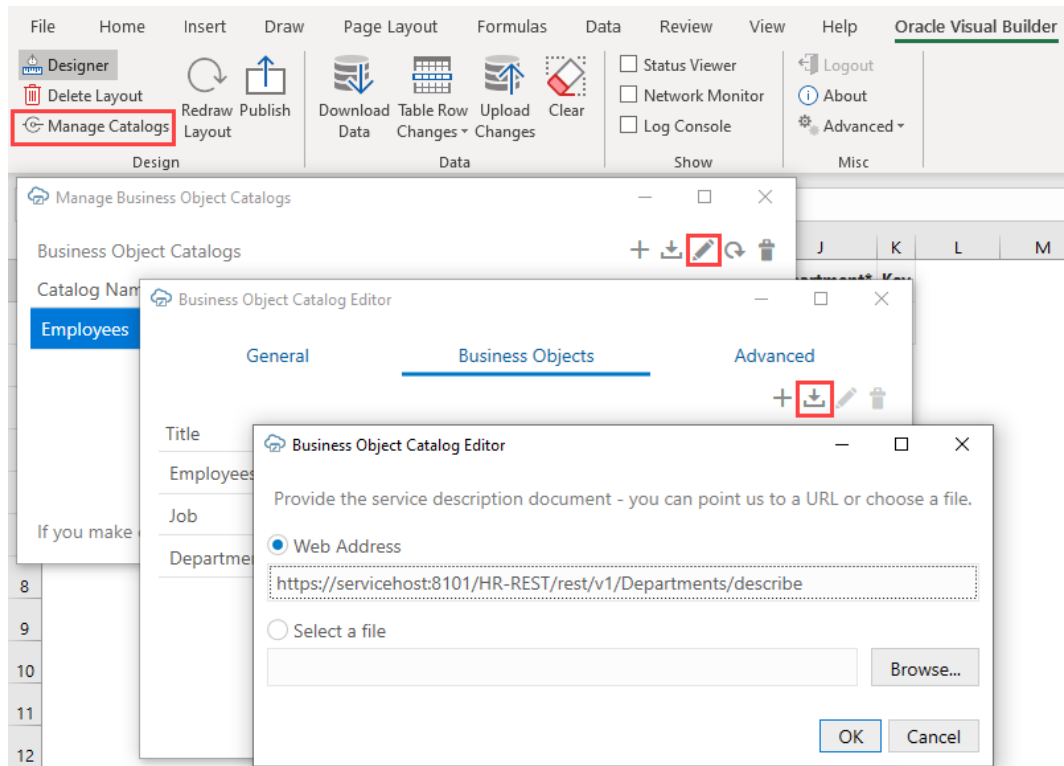


To import a service description document, you have two options:

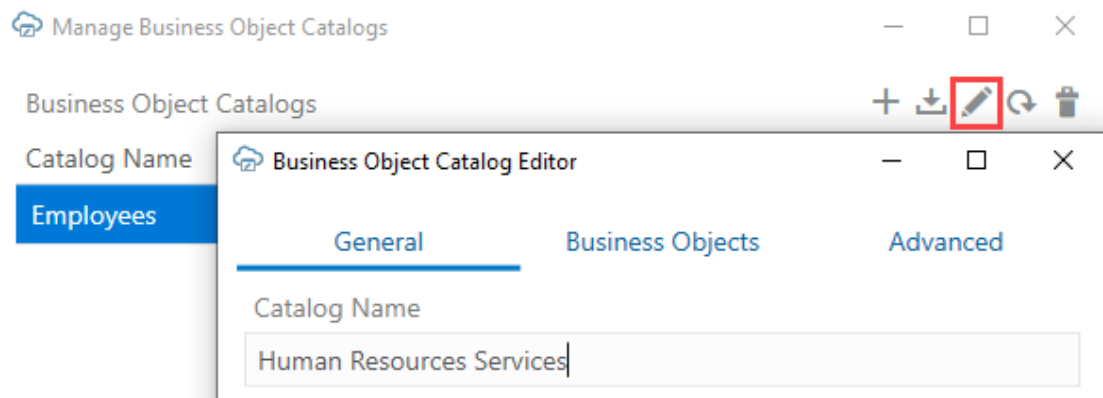
- To create a new catalog, particularly if the OpenAPI document has multiple business objects, use Manage Catalogs.



- To add a business object to an existing catalog (with the understanding that the new business object will be located at the same host/base path as the other business objects in that catalog), use the Business Object Catalog editors (see [Add a Business Object to an Existing Catalog](#)). This option is particularly useful when configuring a list of values.



After you import a service description document into the Excel workbook, consider editing it in the Business Object Catalog Editor so that the catalog has a descriptive name. Very often, the title that the add-in displays for the catalog in the Excel workbook is the name of one of the business objects that the service exposes. For example, this image shows a catalog in which the default value for the Title property is Employees. This catalog exposes an Employees business object, so to avoid confusion, you can change the title that the catalog uses in the Excel workbook to something like Human Resources Services.



After making changes in the Business Object Catalog Editor, such as changing the service host or details of a business object, click **Redraw Layout** in the Oracle Visual Builder tab to see your changes reflected in the worksheet.



Note:

Certain OpenAPI document properties, such as Description, can contain formatting hints. The add-in displays the description text as is, with no interpretation of such hints.

Create a Service Description in Oracle Visual Builder

If your target service does not provide an OpenAPI-compliant service description, you can create one in Visual Builder.

1. Sign in to your Visual Builder account in the Oracle Cloud.
2. Create a new visual application or open an existing visual application.
3. Create a service connection from an endpoint. See [Create a Service Connection from an Endpoint](#).
4. Configure the service with the details that you will need in the Excel workbook integration.
5. In a Visual Builder web application, test the service connection by creating a sample page and table that retrieves data from the service.
6. In the source view of the visual application, locate and download the `service.json` file to your computer.

The `service.json` file contains the OpenAPI service description for the REST service that you connected to from your visual application.

Add a Business Object to an Existing Catalog

If your catalog is missing a business object, you can add a new business object description to the catalog using the Business Object Catalog Editor:

1. Click **Manage Catalogs**.
2. In the **Manage Business Object Catalogs** window, select your existing catalog and click the Edit Business Object Catalog icon.
3. In the **Business Object Catalog Editor**, click **Business Objects**, then click the Import a Business Object from a Service Description icon.
4. Provide the URL or a file that contains the OpenAPI description of the new business object and click **OK**.
5. Click **Done** first in the **Business Object Catalog Editor**, then in the **Manage Business Object Catalogs** window.

Override a Business Object's Base Path

You can configure a business object to use a specified base path. The Oracle Visual Builder Add-in for Excel then uses this base path when making REST requests for the business object.

Typically, a Business Object Catalog holds business objects that share a base path (say, `/abcRestApi-context-root/resources/v1`). But if a business object

needs to use a different base path (for example, your list of values come from `/123RestApi-context-root/resources/v1`), you can provide a new base path for the business object through the Business Object Editor.

To configure a business object to use a base path different from the one shared by business objects in the Catalog:

1. Add a business object to your Business Object Catalog. See [Add a Business Object to an Existing Catalog](#).
2. Click the **General** tab in the Business Object Editor, enter the different base path in **Base Path Override** to override the base path used by all business objects in the Catalog, and click **Done**.

Business Object Editor - Countries

General Fields Download Children Custom Actions

Title
Countries

Description

Parent Business Object
Top-level business object (no parent)

Base Path Override ?
/123RestApi-context-root/resources/v1

Collection Path
/Countries Edit

Item Path
/Countries/{Countries_Id} Edit

If you make changes, redraw the associated layouts Done

The add-in assumes that no extra authentication is required by business objects with a base path override.

You can now use the newly created business object as the **Referenced Business Object** for a list of values if required. See [Configure a List of Values for a Business Object Field](#).

Configure Pagination for a Business Object

If the REST service supports pagination, you can download pages of rows.

Imagine you need to download 10,000 rows of data. Downloading one row at a time is too time-consuming, and attempting to download all 10,000 rows in one request might result in a timeout error. Instead, download one page at a time where the page contains, for example, 500 rows.

You can configure the pagination behavior using the Download tab in the Business Object Editor.

H	I	J	K	L
Salary	Manager Id	Department*	Key	
65,435.00	101	Research	...	
28,000.00	102	Accounting	...	
3,200.00	120	Accounting	...	
24,000.00	100	Accounting	...	
17,000.00	100	Accounting	...	
17,000.00	102	Accounting	...	
10,000.00	145	Accounting	...	

Layout Designer

General Query Columns Advanced

Table ID
TBL1666868393

Origin Cell
A1

Business Object
Employees

Business Object Editor - Employees

General Fields **Download** Children Custom Actions Finders

Offset Parameter Name ?

offset

One-Based Offset

Limit Parameter Name

limit

Limit Parameter Value

499

Response Payload Items Member Name

items

If you make changes, redraw the associated layouts

Done

Configure the following as required:

- **Offset Parameter Name:** The name of the URL parameter that controls where to start the next page. When fetching the first page, the add-in uses a value of zero. When fetching the second page, the add-in uses a value of 499, assuming the limit value is 499.
- **One-Based Offset:** Controls whether the service starts counting from one. If this check box is cleared, the service assumes the service starts counting from zero.
- **Limit Parameter Name:** The name of the URL parameter that controls how many rows to fetch for each page.
- **Limit Parameter Value:** Controls the page size (number of rows that the add-in downloads).

For example, using the defaults for a Oracle business object REST API service, the add-in appends `?offset=0&limit=499` for the first download request.

For other service types, pagination may or may not be supported. If supported, the service may use parameter names like `offset` and `limit` or it may use other parameter names for the same purpose.

Consult the service API documentation to determine which parameters to use.

Configure Row Finders for a Business Object

Row finders are predefined filters available through Oracle business object REST API services that allow you to download a specific subset of records. If your resource developer has included row finders with the service, you can view and configure them through the Finders tab of the Business Object Editor.

For example, if your service includes a row finder that filters expense reports by "unapproved" status and assigned to the "current user", you can use the row finder to download only those expense reports that require your approval.

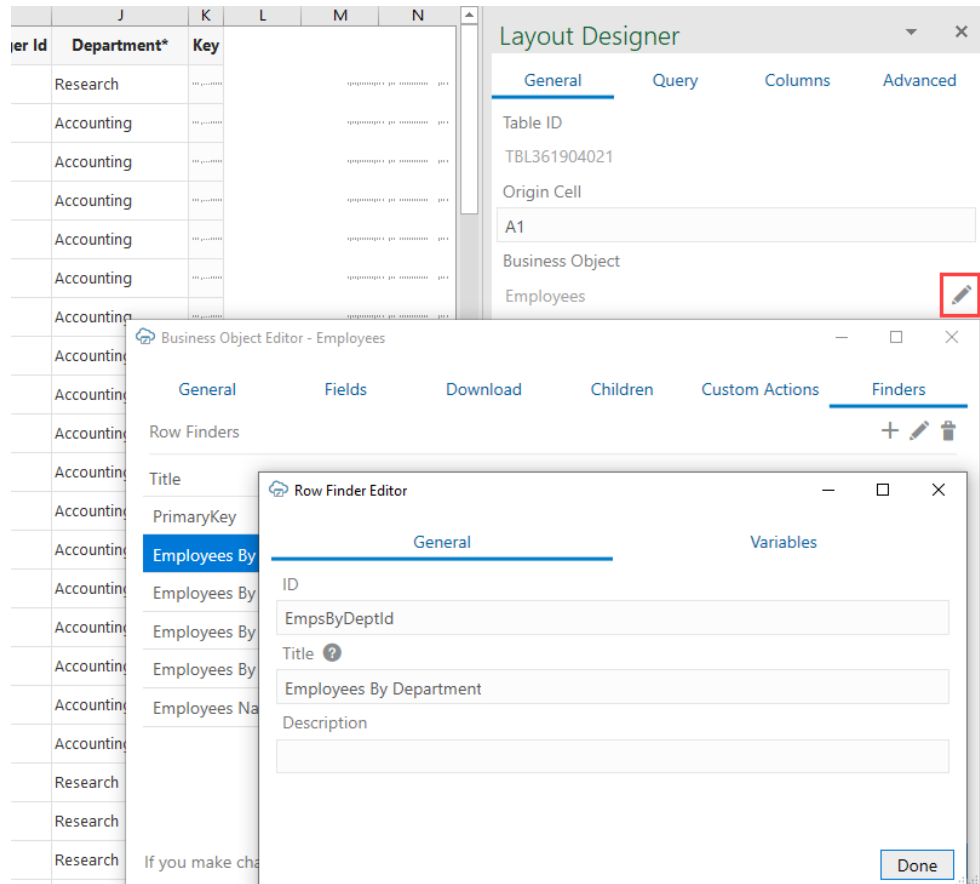
Row finders are defined through the service. You can't add or configure them through Oracle Visual Builder Add-in for Excel. However, you can configure how they appear in the add-in. For example, you can remove unneeded row finders so that they aren't available for use. You may also want to give row finders or row finder variables a more readable title.



Note:

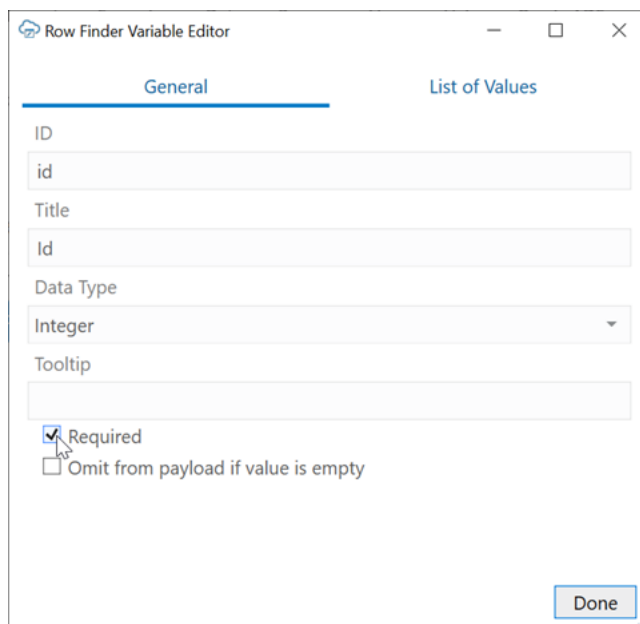
Any changes you make must be compatible with the service. If your changes are incompatible, you will likely see errors during a download.

1. In the Excel ribbon, click **Designer**.
2. In the Layout Designer, click the Edit icon next to the Business Object field.
3. Click the **Finders** tab in the Business Object Editor to view a list of available row finders. Note that the Finders tab shows only for Oracle business object REST API services (not ORDS or other REST services).
4. Remove row finders if required. You can also edit a row finder's details, like changing the title to be more readable:



When editing a row finder, take care to not change the row finder's ID.

5. If a row finder supports variables that act as arguments or parameters to the finder, click the **Variables** tab to update or delete variables. For example, you can change the row finder variable's title (but not the ID) or set the variable as required to force the user to provide a value at download time, as shown here:



6. If required, configure a list of values for a row finder variable from the List of Values page in the Row Finder Variable Editor. See [Configure a List of Values for a Row Finder Variable](#).
7. Click **Done** until you return to the Layout Designer. Once you are done making changes, the row finders become available to you as an option to limit downloaded data, as described in [Use Row Finders to Limit Downloaded Data](#). For details on how your configuration takes effect, see [Download Behavior in Layouts that Use Search and Row Finders](#).

Configure GZIP Compression for Request Payloads

For a POST, PUT, or PATCH request to the service that takes payloads, you can choose to compress the payload body using GZIP if the service accepts compressed request payloads.

When the **Supports gzip compression for request payloads** option is selected, the `Content-Encoding: gzip` header is added to the request. This option (found in the Business Object Catalog Editor's **Advanced** tab) is selected by default for Oracle business object REST API services.

Refresh a Business Object Catalog

A business object catalog is generated when a workbook is first configured to integrate with a service, based on the service description that you provide when creating a layout. If the service owner makes significant changes to the service description—especially its business object definition—after the workbook is integrated with the service, you can refresh the workbook's business object catalog to take advantage of the latest changes.


Refreshing a catalog is useful when changes to a service description are extensive—for example, when new fields, custom actions, row finders, or business objects are added to the service—and updating the business objects to incorporate these changes would take considerable time and effort. You can also refresh the catalog when an important property, such as a required value for a field, has a change that you'd like to readily use in your workbook.

Caution:

A refresh overwrites any changes that you've made locally to the business object catalog. If you've spent a lot of time customizing the business object (especially field titles), consider whether it's worth redoing those changes; it might be simpler to manually add that new field or custom action.

Before refreshing a catalog, back up the workbook in case you want to revert your changes.

To refresh a business object catalog:

1. Click **Manage Catalogs** in the Oracle Visual Builder tab.
2. Select the catalog to refresh in the Manage Business Object Catalogs window.
3. Click the Refresh Business Object Catalog icon (.

4. Provide the URL or a file that contains the new service description.
5. Click **OK**, then **Yes** when prompted to confirm the overwrite.
6. Click **OK**, then **Done**.
7. Refresh all layouts in the workbook before you proceed to other tasks.

If the new catalog has new business objects, those objects are added to the catalog. Existing business objects are refreshed with the corresponding ones in the new catalog. No business objects are removed.

For existing business objects, new fields, finders, custom actions, and so on are added and existing items are refreshed. No existing items are removed.

When an existing item is refreshed, the item's new properties replace the older properties. For example, a field's title, required, and editable properties are copied from the newer version to the older version, except when a new property is empty. If a field's newer version has a list of values configured, the newer list of values replaces the older list of values. If the newer version doesn't have a list of values, the previous list of values is left unchanged.

If you want a new field in your layout, use the Table Column Manager to add the field after refreshing the catalog:

1. In the Layout Designer, click **Columns**.
2. Click Manage Columns (+) and select the new field. Click **Done**.
3. Click **Redraw Changes**.

A refresh does not change any field formats, as those are always set manually.

Configure the REST-Framework-Version

You can change the value of the REST-Framework-Version request header, set to version 6 by default for Oracle business object REST API services. The REST framework version determines functionality for accessing business objects and is set to the default when a layout is first created in a workbook.

To configure a specific REST-Framework-Version that overrides the default framework version:

1. In the Oracle Visual Builder tab, click **Manage Catalogs**.
2. Select the business object catalog, then click the Edit Business Object Catalog icon.
3. In the Business Object Catalog Editor, click the **Advanced** tab.
4. Specify the framework version you want to use in the **REST API Framework Version** field. This field is available only for Oracle business object REST API services (not ORDS or other REST services).

The default value is version 6 (the only supported value). If the value is empty or lower than 6, it is set to the default. Framework versions higher than 6 are not certified.

5. Click **Done**.

Once you specify a framework version, it stays in effect until you manually change it again. Every request sent to any endpoint that belongs to this catalog includes the

REST-Framework-Version header with the specified value. For details on framework versions, see About REST API Framework Versions.

7

Configure Search Options for Download

You can configure the search options for a layout, which are used when the user clicks **Download** in the Oracle Visual Builder tab.

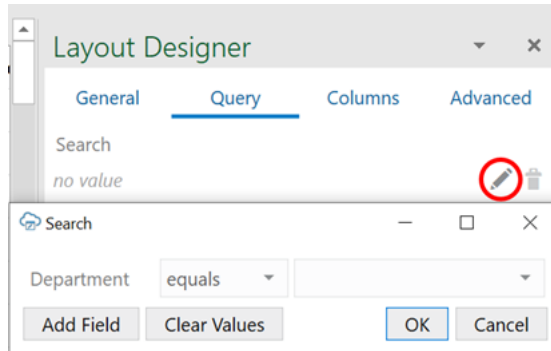
Search Option	Available for	Benefits	Limitations
Graphical Search Editor	Oracle business object REST API and Oracle REST Data Services	<ul style="list-style-type: none">Provides a graphical user interface for building a searchAllows business user to provide values and modify the search	Some limits on search syntax
Row Finders	Oracle business object REST API	<ul style="list-style-type: none">Provides access to powerful, complicated searches defined by the serviceAllows the business user to provide values	Only available for Oracle business object REST API services
Search Parameters	Any service type	Developers can use any syntax the service supports	Business users cannot provide values or change the search at download time

Use Search to Limit Downloaded Data

For workbooks that integrate with Oracle business object REST API services and Oracle REST Data Services, you can configure the workbook to enable a user to specify search values to limit the data that the add-in downloads to the workbook.

All searchable fields are available for you to choose from. You can adjust which fields show by selecting or deselecting **Searchable** in the Business Object Field Editor. Make sure that the service supports searching on that field.

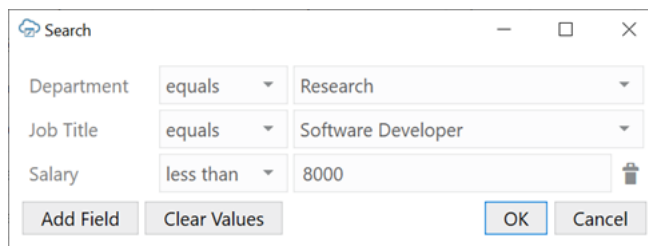
1. In the Excel ribbon, click **Designer**.
2. In the Layout Designer's Query tab, click the Edit icon next to the Search property to open the Available Business Object Fields window.
3. Select the business object field that you want to enable users to enter search terms for. For example, select **Department Name** if you want to enable users to search on employees by department, as shown here:



 **Note:**

You can only specify a single value for each field.

4. To add more fields for complex searches, click **Add Field**. For example, you might want to search for employees with a job title of Software Developer, but only those who earn more than 8000. In this case, first select **Job Title**, then click **Add Field** and select **Salary**. Finally, define the search parameters as follows:



 **Note:**

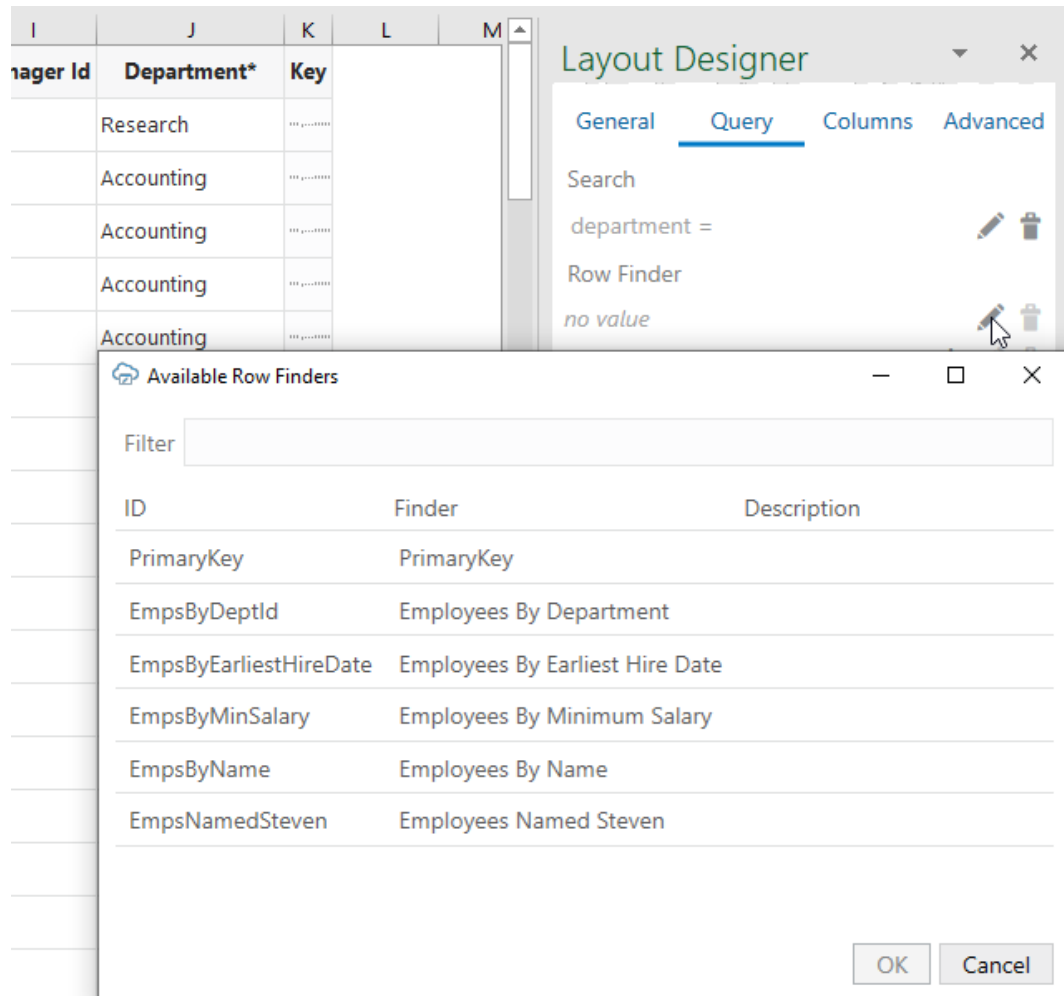
When searching using multiple fields, the add-in always inserts the logical AND operator between each clause.

5. Click **OK**.

Use Row Finders to Limit Downloaded Data

For workbooks that integrate with Oracle business object REST API services, you may select one of the predefined row finders if any are associated with the layout's business object.

1. In the Excel ribbon, click **Designer**.
2. In the Layout Designer's Query tab, click the Edit icon next to the Row Finder property to see the row finders configured for the service in the Available Row Finders window:



For details on modifying how row finders appear in the add-in, see [Configure Row Finders for a Business Object](#).

3. Select a row finder and click **OK**.
When you download data to your workbook, the row finder filters the data based on the filter criteria. If the row finder requires input from the user, you'll be prompted to provide a value for the row finder parameter.

For more information about configuring the Oracle business object REST API service that supports finders, refer to these resources in *Developing Fusion Web Applications with Oracle Application Development Framework*:

- About RESTful Web Services and ADF Business Components
- Filtering a Resource Collection with a Row Finder

Use Search Parameters to Limit Downloaded Data

You can specify additional search parameters to add to the REST endpoint URL on download.

Many REST services support the addition of various parameters in the query string portion of the URL, as in this example:

```
GET.../orderReleaseLines?q=ID=1234
```

where:

- `q` is an optional parameter supported by the `orderReleaseLines` service
- `ID` is the name of a field that supports query
- `1234` is the search value

Each service defines which parameters can be used for search. Likewise, each service defines the required and supported syntax for the expression that appears on the right-hand side of the assignment operator (=).

The add-in does not know which parameters are useful for search. Likewise, the add-in does not know the proper syntax for the parameter values. So, you will need to consult the API documentation for the service you are using to identify whether to use "q" for the parameter name and how to formulate the search expression properly.

There is no validation in this editor or at download time. If you enter invalid information, you may get a bad request error.

 **Tip:**

To troubleshoot a particular combination of search settings, open the Network Monitor and download data. Then, inspect request details in the Network Monitor's window. This information may help you refine search settings.

When the user clicks the Download Data button, the search parameters are added to the appropriate URL along with the other search options (if applicable).

The add-in applies URL encoding to the parameter value at download time. So, you should not enter URL-encoded values. The search parameter name is not encoded.

If the service supports complex searches, you can create complex searches in the layout, as shown in the following example:

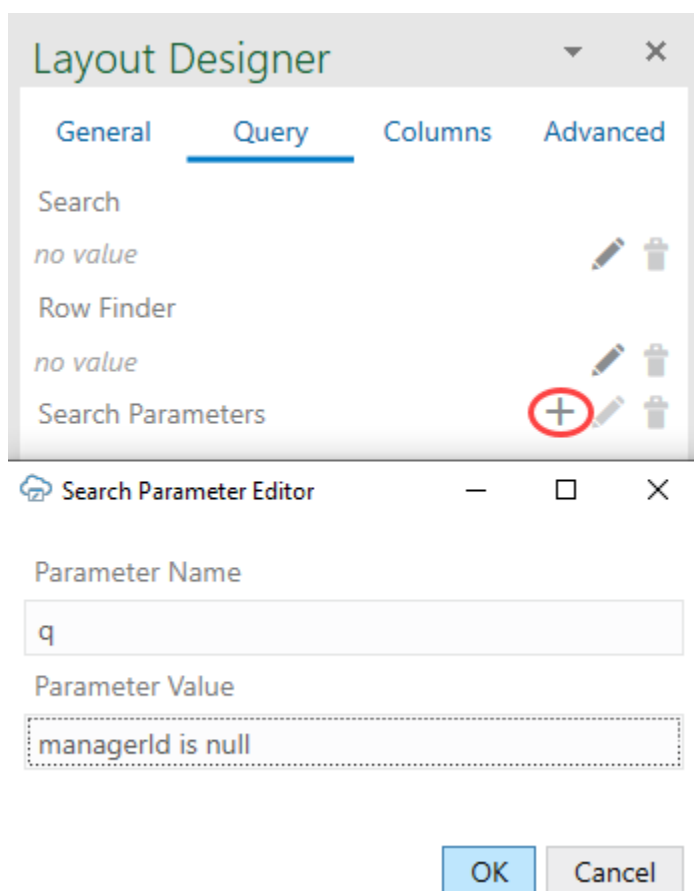
```
q=((firstName LIKE '*es*') or ((hireDate< "2001-01-13") and (department = 10)))
```

The Search Parameters property does not support "dynamic" search values. Using the above example, the value, 1234, is specified in the Layout Designer as a constant.

The Search Parameters property works in combination with the other two properties (where applicable). They are not mutually exclusive. However, some combinations may work where others may not. If you choose to configure multiple search options, you must ensure that the service supports that combination.

To add search parameters to your layout:

1. In the Excel ribbon, click **Designer**.
2. In the Layout Designer's Query tab, click the Add or Edit Search Parameter icon next to the Search Parameters property to open the Search Parameter Editor where you add or edit search parameters.



3. Click **OK** to close the open windows.

Download Behavior in Layouts that Use Search and Row Finders

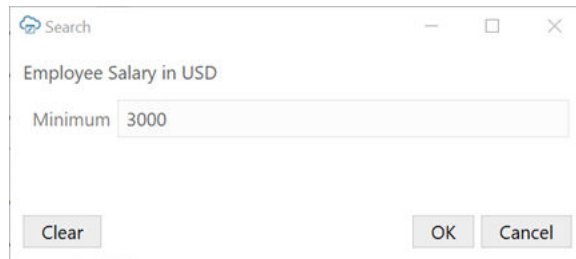
If you configured a value for the Row Finder property, the behavior of the add-in depends on the configuration of the finder exposed by the REST service:

- If there are no parameters, there is no prompt.
- If there is exactly one parameter, a prompt appears that allows the user to specify a value for the parameter.
- If the finder has more than one parameter, a prompt appears that allows the user to specify values for each parameter.



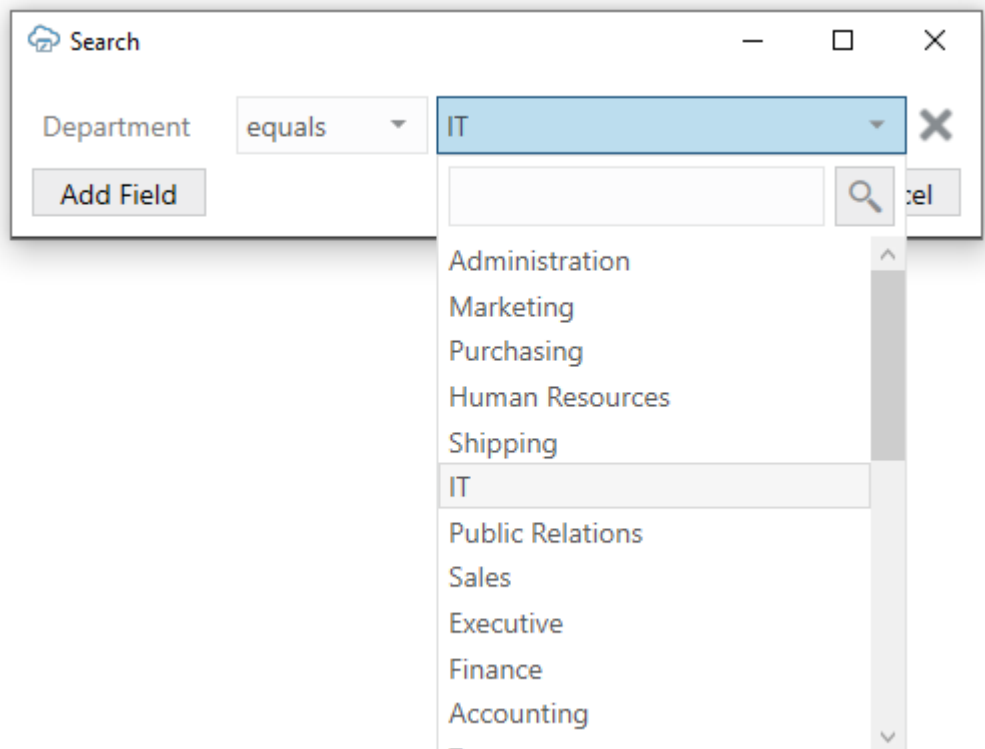
Note:

Only Oracle business object REST API services support row finders.



A screenshot of a search dialog box titled "Search". The dialog contains the text "Employee Salary in USD" and a text input field labeled "Minimum" with the value "3000". At the bottom, there are three buttons: "Clear", "OK", and "Cancel".

If you added search fields to the Search property, the add-in prompts users who download employee data using the **Download Data** button to enter values in the search field, as shown here:



A screenshot of a search dialog box titled "Search". The dialog has a "Department" dropdown menu set to "equals" and a search input field containing "IT". A list of departments is displayed below the input field, including Administration, Marketing, Purchasing, Human Resources, Shipping, IT (highlighted), Public Relations, Sales, Executive, Finance, and Accounting. There is an "Add Field" button on the left and a "Cancel" button on the right.

8

Custom Actions

Workbooks that integrate with Oracle business object REST API services can allow the user to perform custom actions on rows of data.

For example, an expense report business object might support a custom action called "reject". Users could download many expense reports and reject many of them in a single upload.

Use Custom Actions

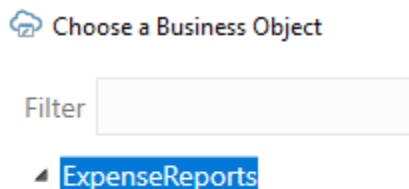
Let's take a look at how you can invoke a custom action in your Excel workbook, for example, the "reject" custom action that's exposed by an expense report business object to reject expenses.



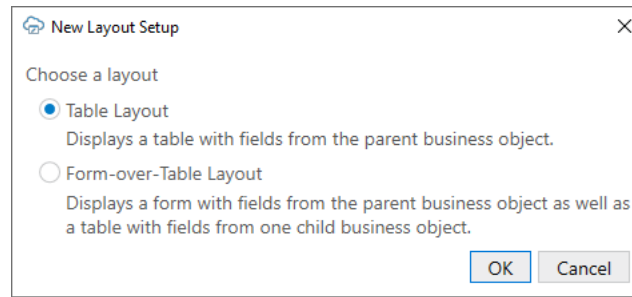
Note:

Your REST API must expose custom actions on the item path for your resource, something like a POST to `/contextRoot/v1/myBusObj/{itemId}/action/doMyAction`. Actions defined on the collection path are not supported.

1. Create a layout for the custom action, for example, a Table layout for the "reject" custom action.
 - a. In the Oracle Visual Builder tab, click **Designer**.
 - b. When prompted, provide the service description document.
 - c. Choose a business object, for example, `ExpenseReports`.

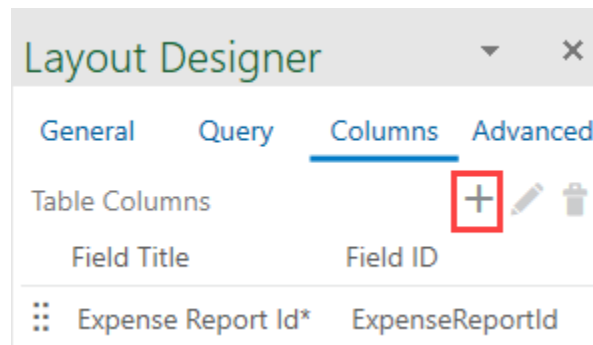


- d. Select a layout and click **OK**.

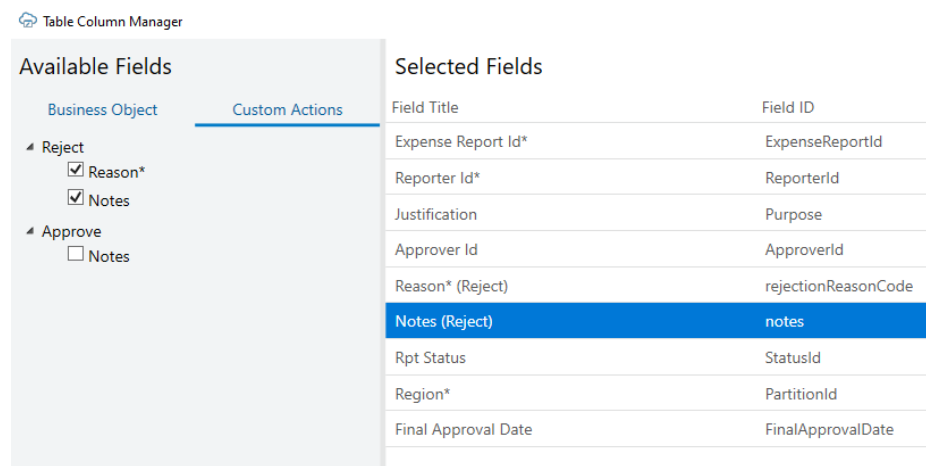


In our example, a Table layout that includes column headers and a placeholder data row is created in the Excel workbook.

2. Create new columns in the layout to pass payload field values that the custom action needs. In this example, we'll add columns to show the Reason and Notes payload fields.
 - a. In the Layout Designer, click **Columns**, then click Manage Columns (+).



- b. Click **Custom Actions** in the Table Column Manager.
 - c. In the Selected Fields pane, select the location where you want the payload field columns to be added. The columns will be inserted before the selected field.
 - d. In the Available Fields pane, select the payload fields (**Reason** and **Notes**).



- e. Click **Done**.

f. Click **Redraw Layout**.

The newly added columns show in the worksheet table.

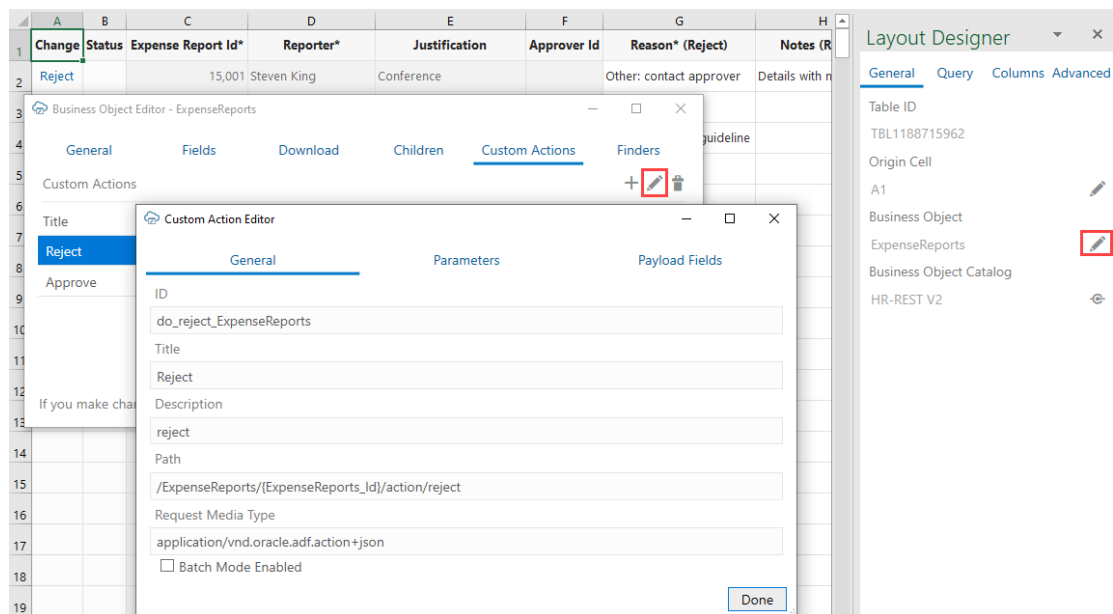
3. Mark rows for the custom action. You can mark rows by entering values in the payload field columns or by using the ribbon command. See Perform Custom Actions in Table and Form-over-Table Layouts in *Managing Data Using Oracle Visual Builder Add-in for Excel*.
4. Click **Upload Changes** to send your changes to the REST endpoint and invoke the custom action on the marked rows.

The result of the upload, including any warnings or custom action responses, is shown in the Status Viewer. See [Upload Changes for Custom Actions](#).

Edit Custom Actions

Custom actions exposed by the service that your workbook uses can be viewed and edited in the **Custom Actions** tab of the Business Object Editor.

Properties such as Title (which appears in the UI that the workbook user sees) and Description can be edited as needed in the Custom Action Editor. Other properties of the custom action should generally be left as is.



Payload fields can be edited on the **Payload Fields** tab of the Custom Action Editor. Properties such as the Title, Data Type, and Required can be changed. For example, you can change the Data Type of a payload field from String to Integer, as long as the change is compatible with the service. You can also update the Required property to mark the payload field as required or not:

The screenshot shows a window titled "Custom Action Field Editor" with two tabs: "General" and "List of Values". The "General" tab is selected. It contains the following fields and options:

- ID:** rejectionReasonCode
- Title:** Reason
- Data Type:** String
- Tooltip:** (empty text box)
- Required
- Omit from payload if value is empty
- Done** button

You can also use the Tooltip field to add a description of the payload field to help your users determine what information or values are expected. Users can display the tooltip in the worksheet by selecting the column header for a payload field.

You can also define a list of values for the payload field (see [Configure a List of Values for a Custom Action Field](#)).

Service Description for Custom Actions

If a given REST API supports custom actions, they are described in the OpenAPI v3 service description document generated by the Oracle business object REST API service. For example, a `reject` custom action would appear in the `paths` collection:

```
// Note: some JSON content has been omitted for brevity/clarity

"/ExpenseReports/{ExpenseReports_Id}/action/reject": {
  "parameters": [
    {
      "$ref": "#/components/parameters/ExpenseReports_Id"
    }
  ],
  "post": {
    "summary": "reject",
    "description": "reject",
    "operationId": "do_reject_ExpenseReports",
    "responses": {
      "default": {
        "description": "The following table describes the default
response for this task.",
        "content": {
          "application/vnd.oracle.adf.actionresult+json": {
            "schema": {
              "type": "object",
              "properties": {
                "result": {
                  "type": "string"
                }
              }
            }
          }
        }
      }
    }
  }
}
```


To enable the add-in to process changes in batches, select the **Batch Mode Enabled** check box in the Custom Action Editor.

If the custom action doesn't work properly in batches, clear the check box to disable batch processing. If the check box is cleared, the add-in invokes the custom action one row at a time.

 **Note:**

The "batchEnabled" : false parameter in the service description only determines the default value of the **Batch Mode Enabled** check box in the Custom Action Editor. If the parameter is not configured, the **Batch Mode Enabled** check box is selected by default.

After the catalog is created, the check box setting controls how the add-in behaves.

Custom Action Limitations

Here are some things to keep in mind when using custom actions:

- Custom actions that correspond to view object methods (as opposed to view object row methods) are not currently supported.
- Custom actions are not supported for pending Create rows or form in Create mode.
- Custom actions defined in the OpenAPI 3 service description document that have request payload schema members that match business object fields are unlikely to function properly.
- Oracle business object REST API service OpenAPI 3 service descriptions do not indicate which custom action request payload fields are required. You can use the Business Object Editor to adjust the **Required** property on payload fields.
- The results of custom actions that return complex object values for the "result" member in the JSON response payload are not shown in the Status Viewer.

9

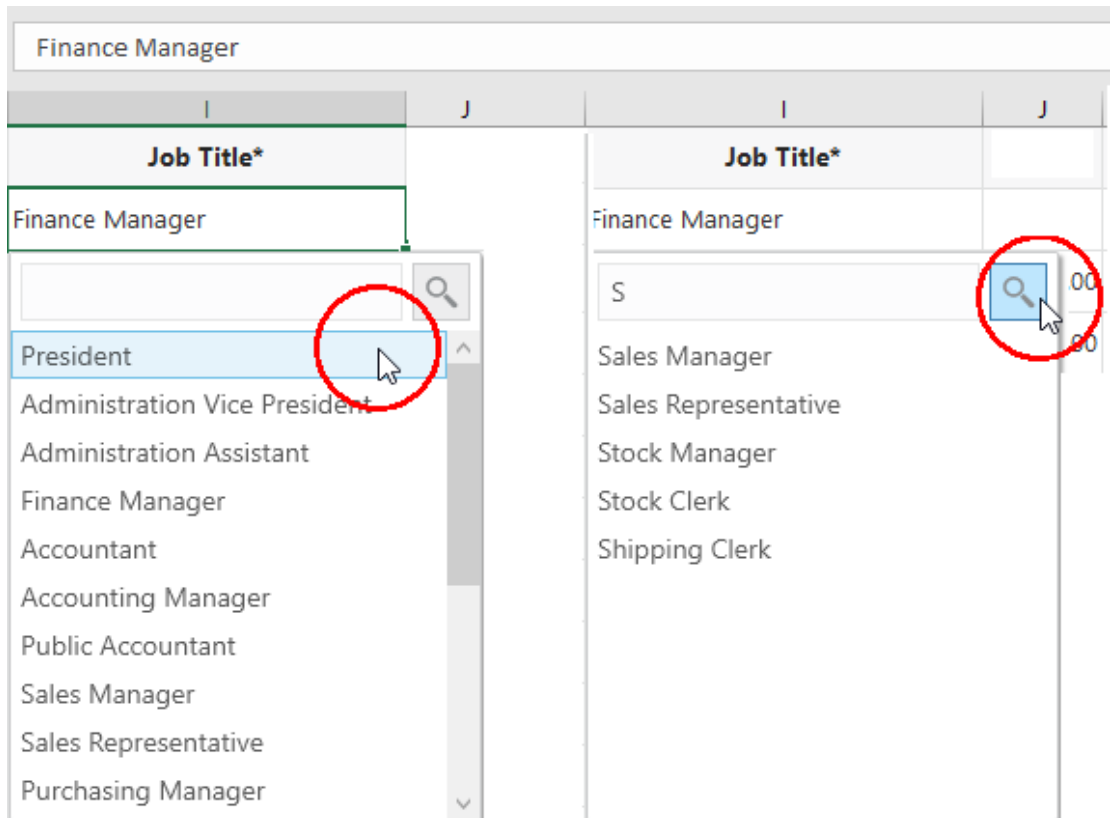
Use Lists of Values in an Excel Workbook

Oracle Visual Builder Add-in for Excel enables you to associate a **list of values** with a business object field, a custom action payload field, or a row finder variable.

A list of values is the set of valid values for a business object field and represents a relationship between the field in one business object and another referenced business object.

When a user selects a cell with a properly configured list of values, a search-and-select window shows a list of values for the user to choose from. Alternatively, the user can enter some text and click the Search icon to filter and show values based on the user input, for example, to show values that start with the user-specified text. See [Configure Search for a List of Values](#).

This image illustrates both these scenarios. For the latter scenario, the user entered S in the search box to filter values from the list that begin with S (Sales Manager, Sales Representative, and so on). Users can also enter the full value in the search box, assuming it's a valid value such as Sales Manager.



Each item in the list of values has a **display** value (which appears in the Excel workbook) and an **identity** value that is retrieved and posted to the business object field. For example, an Employee business object's Job attribute might contain these display and identity values:

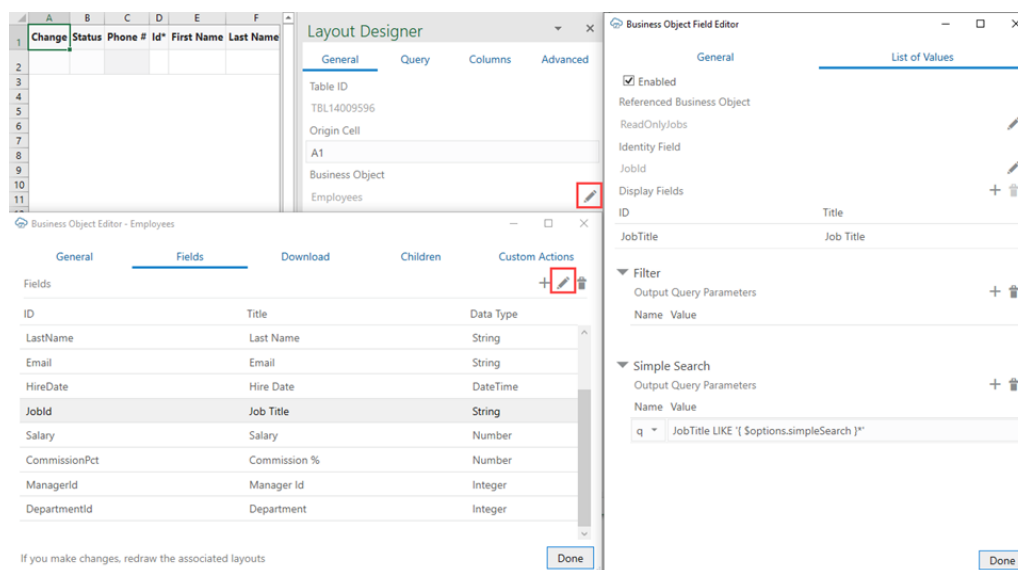
Display value	Identity value
President	PRES
Finance Manager	FIN_MGR
Sales Manager	SAL_MGR

On download, the identity values are replaced by the display values; On upload, the display values are replaced by the identity values.

Configure a List of Values for a Business Object Field

You can use the Business Object Field Editor to add or modify the list of values configuration for a field.

To access this editor, click the Edit icon next to the Business Object field in the Layout Designer. Once you open the Business Object Editor, click the **Fields** tab and select the business object's field. Then, click the Edit icon in the Business Object Editor to open the Business Object Field Editor, as shown in the following image where the Business Object Field Editor is open for the Employee business object's Job Title field.



The List of Values tab in the Business Object Field Editor shows the following properties:

- **Enabled:** Select to enable list of value functionality for this field; deselect to disable
- **Referenced Business Object:** This business object provides the display values for the corresponding identity values
- **Identity Field:** The field in the referenced business object used to look up the display values for the identity values in the current field
- **Display Fields:** These fields come from the referenced business object and are shown instead of the identity values where this field is used in a layout

To configure a list of values for a business object field that does not have one yet:

1. Select the **Enabled** check box on the **List of Values** page.
2. Select the appropriate business object using the Edit icon next to the **Referenced Business Object** field. Pick a business object from the same catalog used to create the layout.
3. Choose the appropriate identity field from the referenced business object.
4. Choose the desired display field from the referenced business object. You can choose multiple display fields for one list of values.
5. Configure Filter and Simple Search as desired (see subsequent sections for details).
6. Click **Done**.

 **Tip:**

If the catalog is missing the desired business object, you can add a new business object to an existing catalog (see [Add a Business Object to an Existing Catalog](#)).

If you want to reference a business object with a different base path, you can create or import a business object into the current catalog and then override the base path (see [Override a Business Object's Base Path](#)).

The add-in [caches the data of list of values](#) in the workbook. After you modify the configuration of any list of values, click **Redraw Layout**. Then, click **Clear List of Values Cache** from the **Advanced** menu in the Misc ribbon tab.

Configure a List of Values for a Custom Action Field

Define a list of values for a custom action's payload field, just as you'd configure a list of values for a business object field.

Use the Business Object Editor to access a business object's custom actions and associated payload fields. You can then edit a payload field to define a list of values:

The List of Values tab shows the following properties:

- **Enabled:** Select to enable list of value functionality for this field; deselect to disable
- **Referenced Business Object:** This business object provides the display values for the corresponding identity values
- **Identity Field:** The field in the referenced business object used to look up the display values for the identity values in the current field
- **Display Fields:** These fields come from the referenced business object and are shown instead of the identity values where this field is used in a layout

To configure a list of values for a payload field that does not have one yet:

1. Select the **Enabled** check box on the **List of Values** page.
2. Select the appropriate business object using the Edit icon next to the **Referenced Business Object** field. Pick a business object from the same catalog used to create the layout.
3. Choose the appropriate identity field from the referenced business object.
4. Choose the desired display field from the referenced business object. You can choose multiple display fields for one list of values.
5. Configure Filter and Simple Search as desired (see subsequent sections for details).
6. Click **Done**.

Tip:

If the catalog is missing the desired business object, you can add a new business object to an existing catalog (see [Add a Business Object to an Existing Catalog](#)).

If you want to reference a business object with a different base path, you can create or import a business object into the current catalog and then override the base path (see [Override a Business Object's Base Path](#)).

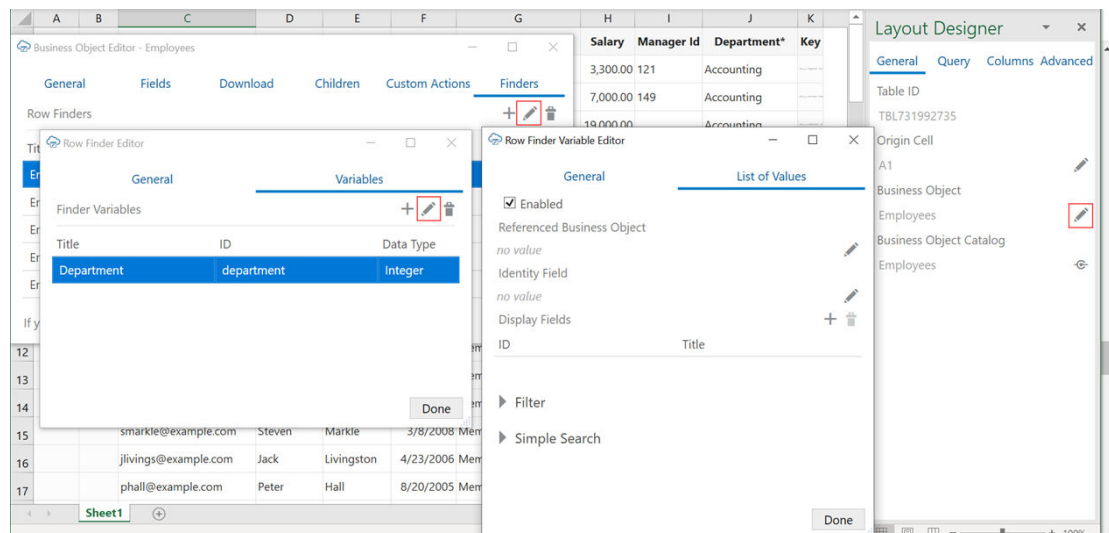
Once you define a list of values, the choice list will appear wherever that payload field appears, as shown in the preceding image where the referenced business object's list of values appear for the **Reason (Reject)** column.

The add-in [caches the data of list of values](#) in the workbook. After you modify the configuration of any list of values, click **Redraw Layout**. Then, click **Clear List of Values Cache** from the **Advanced** menu in the Misc ribbon tab.

Configure a List of Values for a Row Finder Variable

Define a list of values for a row finder variable, just as you'd configure a list of values for a business object field.

Use the Business Object Editor to access a business object's row finder variables. You can then edit a variable to define a list of values:



The List of Values tab shows the following properties:

- **Enabled:** Select to enable list of value functionality for this field; deselect to disable
- **Referenced Business Object:** This business object provides the display values for the corresponding identity values
- **Identity Field:** The field in the referenced business object used to look up the display values for the identity values in the current field

- **Display Fields:** These fields come from the referenced business object and are shown instead of the identity values where this field is used in a layout

To configure a list of values for a row finder variable that does not have one yet:

1. Select the **Enabled** check box on the **List of Values** page.
2. Select the appropriate business object using the Edit icon next to the **Referenced Business Object** field. Pick a business object from the same catalog used to create the layout.
3. Choose the appropriate identity field from the referenced business object.
4. Choose the desired display field from the referenced business object. You can choose multiple display fields for one list of values.
5. Configure Filter and Simple Search as desired (see subsequent sections for details).
6. Click **Done**.

 **Tip:**

If the catalog is missing the desired business object, you can add a new business object to an existing catalog (see [Add a Business Object to an Existing Catalog](#)).

If you want to reference a business object with a different base path, you can create or import a business object into the current catalog and then override the base path (see [Override a Business Object's Base Path](#)).

Once you define a list of values, the choice list will appear wherever that row finder variable appears during download.

The add-in [caches the data of list of values](#) in the workbook. After you modify the configuration of any list of values, click **Redraw Layout**. Then, click **Clear List of Values Cache** from the **Advanced** menu in the Misc ribbon tab.

Configure Filters for a List of Values

You can configure a filter for a list of values so that users see only certain values in the list when they select the field in a layout. You can define these filters using name-value query parameters.

The query parameter's value is a string that could include the expression `{ $options.fieldValues['FieldId'] }`, where *FieldId* is the ID property of another field in the current business object. The add-in replaces this expression with the corresponding field value (specifically, the identity value) when sending the request for the referenced business object's list of values.

 **Note:**

Expressions like `{ $options.fieldValues['FieldId'] }` cannot be used in filters for custom action fields or row finder variables.

For example, when you work with a Table layout for departments and want users to see only the current department's employees, you can specify a configuration to filter the Employees list with a Row Finder based on the current row's DepartmentId, if the REST service supports finders.

 **Note:**

The filter you define determines which items are fetched and displayed in the search-and-select window. This feature is different from the simple search query. When you configure a simple search for a list of values, the user can enter a search term in the search-and-select window and return values for the list that match the term. See [Configure Search for a List of Values](#) for more information on the simple search feature.

1. In the Business Object Field Editor's List of Values tab, click the Add Query Parameter icon in the Filter section.
2. In the **Name** list, select **finder** as the query parameter.
3. In the **Value** text box, enter an expression similar to:

```
EmpsByDeptIdFinder;DeptId={ $options.fieldValues['DepartmentId'] }
```

where `EmpsByDeptIdFinder` is the finder's name, `DeptId` is the finder's bind variable, and `{ $options.fieldValues['DepartmentId'] }` is the expression for the current row's `DepartmentId` field value.



If the list of values' default order is unclear, you can add the **orderBy** parameter if the service supports it.

Configure Search for a List of Values

You can allow your users to enter a string to search for a value in a list of values by defining search parameters.

The query parameter value is a string expression that may include the expression `{ $options.simpleSearch }`. The add-in replaces this expression with the input that the user enters in the search box within the search-and-search window. The add-in does not validate the values, so if the user enters invalid information, the add-in will make invalid requests that return errors.

The syntax for the simple search query varies based on the REST service type and expected query syntax for a business object.

Simple Search parameters are not applied when the add-in retrieves the initial set of values to show in the search-and-select window next to the cell in the worksheet. They are applied only when the user clicks the Search icon (after the initial set of values are already retrieved).

If your list of values includes more than one display field, you can configure a simple search query that returns matches from any of the display fields. For example, to match text to either an employee's first or last name, create a query with the "OR" operand like this: `FIRST_NAME LIKE '{ $options.simpleSearch }*' OR LAST_NAME LIKE '{ $options.simpleSearch }*'.` See [Configure a List of Values with Multiple Display Fields](#).

 **Note:**

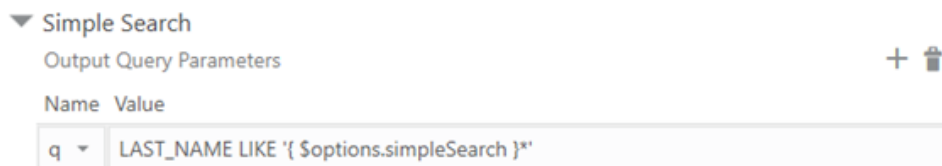
When both filter and simple search parameters are configured, the filter parameters are applied first and then the simple search parameters. The filter determines which items are fetched and displayed in the search-and-select window. Simple search only searches within the filtered list. If multiple values are specified for one parameter, the last value takes effect. See [Configure Filters for a List of Values](#) for more information on the filter feature.

1. In the Business Object Field Editor's List of Values tab, click the Add Query Parameter icon in the Simple Search section.
2. Add values for the Name and Value properties based on what the service supports. Many REST services support a specific query parameter such as `q` for search, as in the following example:

```
GET.../employees?q=LAST_NAME LIKE 'Jones*'
```

Select the appropriate query parameter in the Name list; in our example, this is `q`. In the Value text box, provide a search expression that is appropriate for the service and include the `{ $options.simpleSearch }` text placeholder where appropriate. In our example, the Value is:

```
LAST_NAME LIKE '{ $options.simpleSearch }*'.
```



Now, if the user enters `Jo` in the search box and clicks the Search icon, the resulting request to the referenced business object will be:

```
GET.../employees?q=LAST_NAME LIKE 'Jo*'.
```

 **Note:**

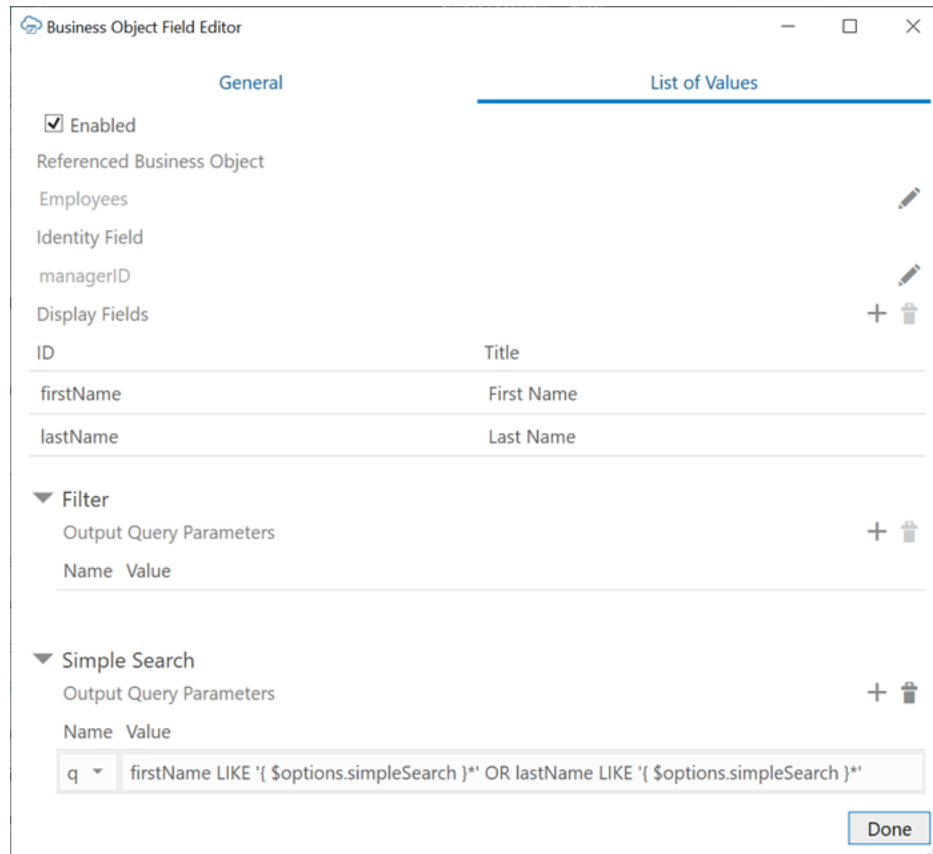
Consult the API documentation for your service to determine the appropriate search syntax to use in the Value column. For example, if you use an Oracle business object REST API service, consult *Understanding Framework Support for Query Syntax in Accessing Business Objects Using REST APIs*. Remember that the add-in uses REST framework version 6 when making requests to a business object REST API service.

Configure a List of Values with Multiple Display Fields

You can configure a list of values that shows more than one display field in the search-and-select window when the user selects a cell.

When you do so, Oracle Visual Builder Add-in for Excel shows the display fields in separate columns but concatenates these values when shown in the Excel cell.

For example, you may want to have a list of values showing managers by first and last name. Configure the list of values as you would normally but include both name fields (firstName and lastName) in the Display Fields table.



When a user selects a cell in the Manager column, the add-in displays a search-and-select window that the user can choose from, showing both first and last name in separate columns as shown here.

Manager*	Department*	Key
	Research	

First Name	Last Name
Sophie	Ren
Dave	Brown
Julia	Nayer
Steven	King
Neena	Kochhar
Lex	De Haan
Peter	Tucker
Irene	Mikkilineni
Alyssa	Hutton
James	London

You can also configure a simple search query that returns matches from either of the display fields. For example, to match text to either the employee's first or last name, create a query with the "OR" operand like this:

▼ Simple Search

Output Query Parameters + 🗑️

Name	Value
q	firstName LIKE '{ \$options.simpleSearch }*' OR lastName LIKE '{ \$options.simpleSearch }*'

Note:

The syntax for the simple search query varies based on the REST service type and expected query syntax for a business object.

When a user types in the search box and clicks the Search icon, the add-in displays all first or last names that match the string. In this example, the add-in matches the search string "Jo" to the first name "John" but also the last name "Jones".

Manager*	Department*	Key
	Research	

First Name	Last Name
John	McGahern
Jonathon	Taylor
Charles	Johnson
John	Russell
Vance	Jones
John	Seo
John	Chen
Joshua	Patel
Jose Manuel	Urman

 **Note:**

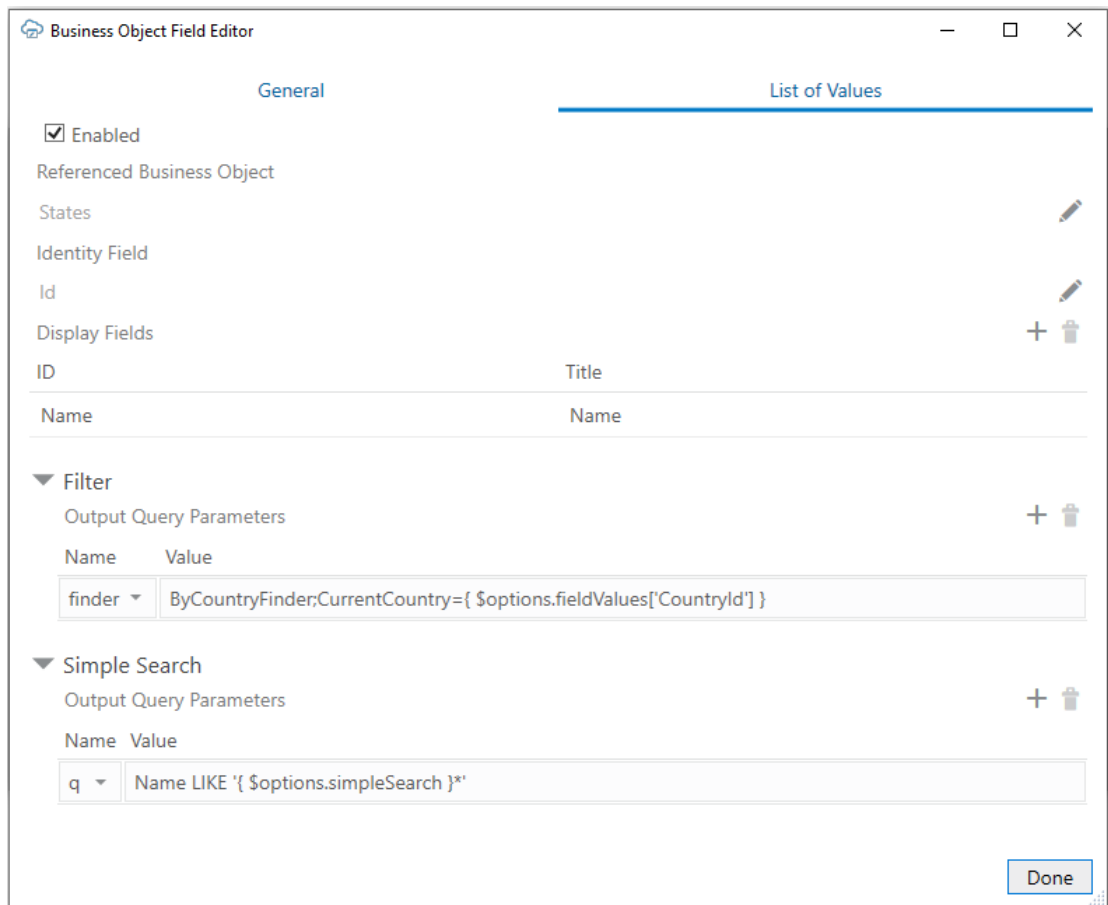
For a list of values with multiple display fields, if you enter a value directly in the cell that is not yet cached, the value will be considered invalid initially. Use the search-and-select window instead to select the value. The default search is specific to the first display field. You can [modify the search expression](#) from the List of Values page.

Configure a Cascading List of Values in a Layout

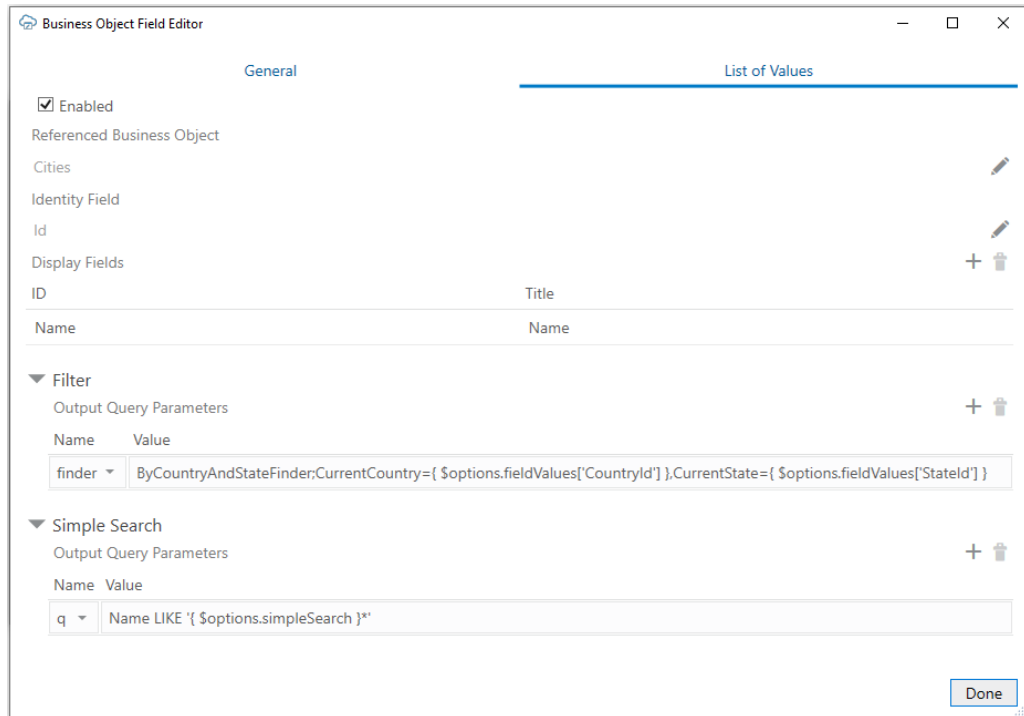
You can configure a **cascading list of values** if the REST service for your Excel workbook supports it.

In a cascading list of values, the value selected in one list determines the range of values that users can select from subsequent lists. For example, suppose a table displays columns with lists of values for Countries, States, and Cities. The value that a user chooses in the Countries list determines the values that appear in the list for States, and so on.

Here's how you use the Filter section in the Business Object Field Editor to configure a list of values for Countries, States, and Cities. To filter the list of values by the current country:



To filter the list of values by Country and State:



In a scenario like this with three fields ("fieldA", "fieldB", and "fieldC"), remember that fieldC depends on both fieldB and fieldA. So when you create the filter for fieldC's list of values, also include fieldA's value expression. In general, all of fieldB's dependencies must also be fieldC's dependencies.

Note the following name-value parameters in the Filter sections:

Table 9-1 Example Name-Value Parameters in Filter

Field	Name	Value
StateId	finder	<code>ByCountryFinder;CurrentCountry={ \$options.fieldValues['CountryId'] }</code>
CityId	finder	<code>ByCountryAndStateFinder;CurrentCountry={ \$options.fieldValues['CountryId'] },CurrentState={ \$options.fieldValues['StateId'] }</code>

If the service has complete metadata for a cascading list of values, the Filter section of the corresponding field's list of values is automatically configured.

If the service does not have complete metadata for a list of values but you know it supports search syntax that makes for a cascading list of values, you can use the Filter to manually configure a list of values. See [Configure Filters for a List of Values](#).

Configure the Bind Parameters for a Descriptive Flexfield's List of Values

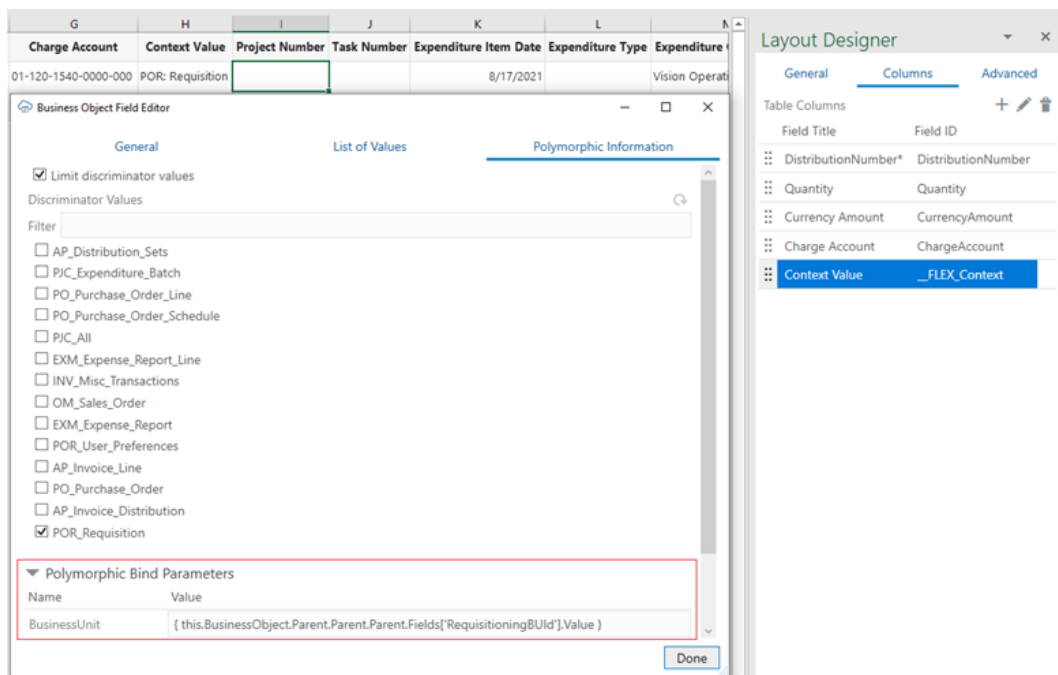
You can use the Business Object Field Editor to configure the bind parameters used by segments in a descriptive flexfield (DFF).

Oracle Visual Builder Add-in for Excel currently only supports configuring the polymorphic bind parameters. Other details of the list of values for DFFs must come from the metadata and are not configurable.

To configure polymorphic bind parameters (also known as "flex bind variables"), open the Business Object Field Editor for a DFF from the Layout Designer, and then configure the bind parameters in the Polymorphic Bind Parameters area of the Polymorphic Information tab.

For example, suppose you want to configure the bind parameter, **BusinessUnit**, for a list of values on a polymorphic field set (**Context Value** column in the Layout Designer). To configure the polymorphic bind parameters in this case:

1. From the Layout Designer, select the Columns tab and then select the **Context Value** column.
2. Click the Edit icon in the Business Object Editor to open the Business Object Field Editor.
3. From the Polymorphic Information tab, configure the bind parameters, as shown in the following image:



- Value of a bind parameter can be a literal value like 100001, Requisition001, or a string that contains one or more expressions surrounded by curly braces { }
- An expression can refer to the value of a field in any business object in the current business object hierarchy.
 - there must be a Form-over-Table layout configured for that business object

- the field referred to must be exposed in the form of the Form-over-Table layout
- In this example, the value is a single expression, where
 - *this* is the polymorphic field set
 - *this.BusinessObject* is the business object that owns the polymorphic field set (the DFF business object)
 - *Parent* gets the parent business object in the hierarchy (bottom-up: RequisitionDistributions/projectDFF, RequisitionDistributions, RequisitionLines, PurchaseRequisitions)
 - *this.BusinessObject.Parent.Parent.Parent* is the PurchaseRequisitions business object that owns the field. *RequisitioningBUID*
 - *Fields['<field ID>'].Value* gets the current field value of the field with the given field ID. Note the single quotes (required) around the ID.
- When the value is a compound string such as { *expr1* } is the capital of { *expr2* }, and *expr1* resolves to *Paris* and *expr2* resolves to *France*, the resolved value of the whole string will be "Paris is the capital of France".

Limitations

- This feature only supports lists of values in context-sensitive DFF fields and not "global" DFF fields.
- Only binds of data type "string" and "number" are supported.
- The add-in uses client binds if and only if the OpenAPI 3 document provides proper metadata.
- Expression for the bind parameter can only refer to field value of a form field in Form-over-Table layout. Referring to field value of a table (table portion of Form-over-Table layout, or Table layout) is not supported.
- Expression syntax is limited to what's shown in the example (key parts: *this*, *BusinessObject*, *Parent*, *Fields[<field ID>].Value*).

Clear Cache for a List of Values

A list of values' choices are always cached in the workbook.

The cache contains up to 300 items, plus all used items. It is populated during the first download or the first time the search-and-select window is used. The search-and-select window shows the cached list of values, if available. An upload also uses cached data.

As a workbook developer, remember to click **Clear List of Values Cache** from the **Advanced** menu:

- Whenever you change any list of values configuration, and always when you publish the workbook
- When the service host is changed.
- When cached data is not what the user expects.

Notes and Limitations for Lists of Values

Review the notes and limitations listed here when planning to use lists of values:

- For Oracle business object REST API services, "row context" list of values is not supported.
- Only one identity field is supported in a list of values.
- Display values support strings; identity values support integers or strings. Decimal numbers, dates, and date-time values are not supported.
- For any read-only field (column or form), the add-in still swaps identity values for display values. However, the search-and-select window does not appear when the cell is selected.
- Excel drop-down lists using data validation may not be compatible with the add-in.
- For Oracle business object REST API services, if the catalog includes metadata for an LOV, the add-in reads the metadata and configures the LOV automatically. Supported ControlTypeHints: "combo_lov", "choice", and "input_text_lov".

10

Appearance of an Integrated Excel Workbook

Oracle Visual Builder Add-in for Excel automatically manages the appearance of an integrated Excel workbook through built-in styles and data format types.

The add-in automatically formats cells in a workbook by creating a set of named Excel styles when the workbook is first initialized. The styles created by the add-in are consistent with Oracle Alta UI standards and Oracle accessibility guidelines. The format portion of the styles is sensitive to the user's preferences as defined in the Windows region settings. So a US user will see US dates and a French user will see French dates.

Once the styles are initialized, the add-in applies those styles to the integrated cells, according to the field's properties, at key points (such as during a data download). Typically, the styles are created once for a workbook and are never updated, but you can take advantage of the options described in this chapter.

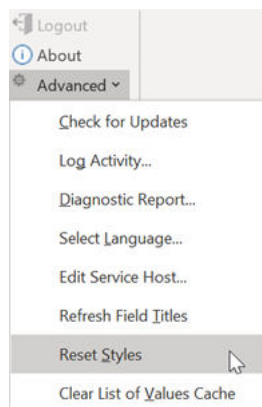
Reset Workbook Styles

You have the option to reset styles in an Excel workbook when a new add-in version has updated style definitions and you want to use those styles in an older workbook, or when a user has changed style definitions in a workbook and wants to revert to the current definitions.

To reset the style definitions in a workbook, select **Reset Styles** from the Advanced drop-down.

Note:

The **Reset Styles** command is only available when the designer tools have been enabled.



You are prompted to confirm. After confirmation, each style definition used by the add-in is updated. The list of styles includes any style that would be created by the current add-in, including the Normal style.

Choose Field Formats

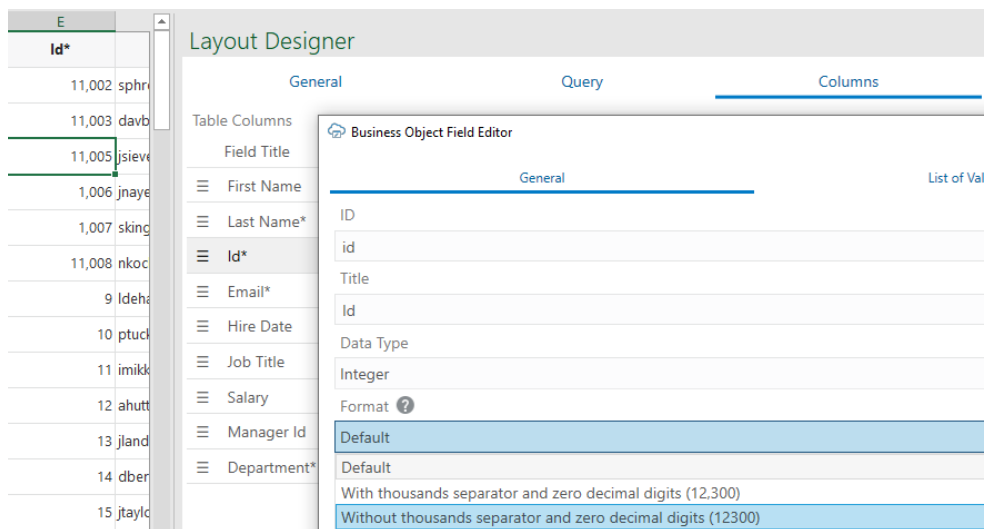
Cell formats in an Excel workbook are applied by the add-in according to a field's data type and based on whether the field is editable (create or update). While this behavior works for most scenarios, sometimes you might want to extend a field's formatting options.

Let's consider the example of an employee workbook that includes ID and Salary fields, both of which are integers. A value of 10,000 for the Salary field can be formatted with the thousands separator (10,000 versus 10000). But an ID value of, say, 12300 seems odd with the thousands separator (12,300). In this case, you can override the ID field's default format to choose a format without the thousands separator.

To override the default format for a field:

1. In the Layout Designer's Columns tab, select the field you want to update and click the Edit icon.
2. In the Business Object Field Editor that appears, select an option in the **Format** drop-down. The default value is `Default`, which tells the add-in to apply the usual automatic styles without any override.

Remember, the format types shown to you are based on what's appropriate for the field's data type. In this image of an employee's workbook, the ID field uses the Integer data type and the available options let you format the ID with or without the thousands separator. The options also follow user preferences as defined in the Windows Region settings, so a US user will see a decimal point ($\pi = 3.14$) while a French user will see a decimal comma ($\pi = 3,14$).



3. Click **Done**.
4. Click **Redraw Layout** or **Download Data** to process the change.

Add Tooltips to Your Workbook

You can add tooltips to the form field labels or table column headers of your workbook to provide help to your users. You can also add tooltips to custom action fields.

To add a tooltip, open the Business Object Field Editor for a field and type help text in the Tooltip property on the General tab.

The screenshot shows the 'Business Object Field Editor' dialog box with the 'General' tab selected. The 'ID' field is 'POHeaderId', the 'Title' is 'PO Header Id', the 'Data Type' is 'Integer', and the 'Format' is 'Default'. The 'Description' field is empty. The 'Tooltip' field is highlighted with a red box and contains the text 'This is the Purchase Order Header ID'. Below the tooltip field are several checkboxes: 'Required for update', 'Required for create', 'Editable on update', 'Editable on create', 'Searchable' (checked), and 'Omit from payload if value is empty'. A 'Done' button is at the bottom right.

The tooltip is displayed when the user selects the form field label or table column header.

The screenshot shows a table with a tooltip displayed over the 'PO Header Id' column header. The table has a header row and four data rows. The tooltip contains the text 'This is the Purchase Order Header ID'.

	A
1	Purchase Orders
2	PO Header Id
3	PO Header Id This is the Purchase Order Header ID
4	

11

Upload Changes

When you are done making changes in an Excel workbook, you are ready to upload the changes to the REST service.

Upload Changes from a Table Layout

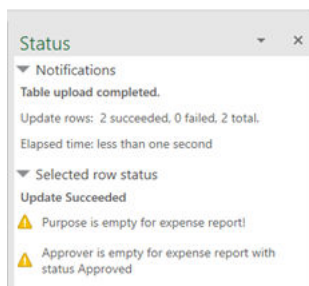
When a user clicks the **Upload Changes** button for a Table layout, here's what happens:

1. The add-in checks the table for pending changes. If there are no pending changes, upload is skipped.
2. If a Pre-Upload macro is configured, it is invoked. If the macro throws an exception or returns any value other than `true`, the upload process quits.
3. For pending Create and Update operations, the rows are first validated locally, for example, data type, required, and so on (see Data validation in *Managing Data Using Oracle Visual Builder Add-in for Excel*). Any failures are marked as failed and skipped (these rows do not produce requests on the business object service).
4. Pending changes are generally processed as follows:
 - a. Updates result in a PATCH or PUT request on the item path.
 - b. Creates result in a POST request on the collection path.
 - c. Deletes result in a DELETE request on the item path.
 - d. Rows marked for action result in a POST request on the item action path.

 **Note:**

Batch processing is handled differently. See [Upload Changes Using Batch Requests](#).

5. Success and failure is noted in the Status column. Errors and warnings are cached and displayed in the Status Viewer when the user clicks on a given row. Here is a sample of the Status Viewer showing two warnings on a successful update:



6. Successful Create rows are updated from the service and converted into existing rows that can be edited.
7. Successful Delete rows are removed from the Excel worksheet.

The request payload for Create and Update operations includes a value (possibly empty) for every column in the table, except those for read-only fields and custom action payload fields (see [Upload Changes for Custom Actions](#)). If configured, empty values are skipped. See [Omit Empty Values During Upload](#).

The add-in does not track which columns have been altered, it always sends values for the entire row.

The add-in may send up to four requests to the service at a time for improved performance (on different threads). As a result, the order in which the rows are sent to the server is non-deterministic. It is not guaranteed to be in the same order as in the table.

Upload Changes from a Form-Over-Table Layout

When a user clicks the **Upload Changes** button for a Form-Over-Table layout, here's what happens:

1. The add-in checks the form for a pending Update or pending Create operation and the table for pending changes. If there are no pending form or table changes, upload is skipped.
2. If a Pre-Upload macro is configured, it is invoked. If the macro throws an exception or returns any value other than `true`, the upload process quits.
3. When the form has a pending Update or pending Create operation:
 - For a pending Update:
 - a. A GET request is sent to the parent's item path.
 - b. Form field values (for example, data type, required, and so on) are validated; read-only fields are skipped. If validation failures exist, the upload is stopped and no subsequent REST requests are made.
 - c. When all form fields are valid, a request (PATCH/PUT) is sent to the parent's **item** path. The payload for this request contains the values of all form fields that have been changed. If configured, empty values are skipped. See [Omit Empty Values During Upload](#).

For a pending Create:

- a. Form field values (for example, data type, required, and so on) are validated (see *Data validation in Managing Data Using Oracle Visual Builder Add-in for Excel*); read-only fields are skipped. If validation failures exist, the upload is stopped and no subsequent REST requests are made.
- b. A POST request is sent to the parent's **collection** path. The payload for this request contains a value (possibly empty) for every editable form field in the form. If configured, empty values are skipped. See [Omit Empty Values During Upload](#).

The child table is not involved in this step.

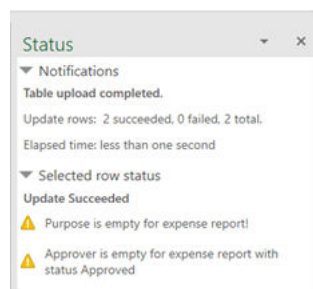
4. The results of the form upload are reflected in the status viewer immediately.
5. If the form upload fails, the add-in stops and does not attempt an upload on the child table.

6. If the form upload succeeds, the add-in proceeds with the child table as follows:
 - a. Checks the child table for pending changes, creates, deletes, custom actions, etc.
 - b. If changes are found, the child table upload proceeds in the same manner as a Table layout upload with one important difference: the child business object's paths are used for each request.
The add-in ensures that the parameters in the child business object paths are replaced with the correct values during the table upload operation.

If the form and table both have pending changes, there are a minimum of two requests: one for the form and one (or more) for the child table.

The form and child table changes are never sent in a single request. In particular, sending a single request where the payload contains values from the parent form fields along with an array of values from the child table rows is not supported.

Success and failure is noted in the Status column. Errors and warnings are cached and displayed in the Status Viewer when the user clicks on a given item. Here is a sample of the Status Viewer showing two warnings on a successful update:



Upload Changes for Custom Actions

For Oracle business object REST API services that expose custom actions on the item path, the user can mark rows so that the custom action is called on those rows during the upload. For custom actions that require payload fields, table columns that correspond to those fields must be added.

During the upload operation (which may also include update, create, and delete tasks), the add-in performs the following steps for each marked row:

1. Creates the payload by collecting the cell values for each custom action field column and adding the value to a simple JSON object (member name/value pairs) in the payload. If configured, empty values are skipped. See [Omit Empty Values During Upload](#).
The entire payload body follows this example format:

```
{
  "rejectionReasonCode": "Other: contact approver",
  "notes": "Details with manager"
}
```

- There is no other content in the POST request body (no action name, no array of argument values).

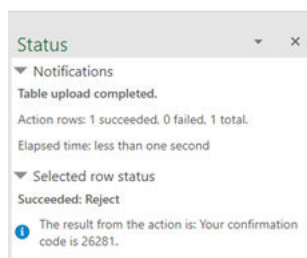
- If any values from these columns are invalid (missing when required, incorrect data type, Excel formula error), the row is omitted from the Upload and marked as failed. See Data validation in *Managing Data Using Oracle Visual Builder Add-in for Excel*
2. Prepares the request:
 - REST-Framework-Version header added (see [Configure the REST-Framework-Version](#))
 - Content-Type header added based on each custom action's Request Media Type property on the **Custom Action Editor** (available from the **Business Object Editor > Custom Actions** tab)
 3. Makes the request:
 - Sends the POST to the custom action path (POST is the only HTTP method supported for invoking custom actions)
 - See the note below about batch requests.
 4. Processes the response:
 - For 200 response status, the row is marked as succeeded.
 - For 400 response status, the row is marked as failed, and the response payload is parsed for Oracle business object REST API service error content; error details can be seen in the Status Viewer pane.
 - A 412 response status indicates that the row was modified by some other agent or user after it was downloaded to the Excel table; such a status is treated as a row-level error

Cell values in action rows are not refreshed. If the custom method logic in the service has altered any values in the row, those changes will not be reflected in the table row until the next download.

If the response payload for the Custom Action request includes a "result" member with a simple value such as a string or number, then the add-in displays that result in the Status Viewer. The response payload for a successful invocation should have a result member in the following format:

```
{  
  "result": "<return-value-from-custom-action-method>"  
}
```

For example, after invoking the custom action "Reject" on an Expense Report table row, the REST service may return a result such as "result": "Your confirmation code is 26281.". This result is displayed in the Status Viewer, as shown here:



See [Executing a Custom Action](#) in the *Developing Fusion Web Applications with Oracle Application Development Framework*.

For information on custom actions, see [Custom Actions](#).

Upload Changes Using Batch Requests

Oracle Visual Builder Add-in for Excel supports the use of batch requests for uploading changes to Oracle business object REST API services.

During an upload operation, the add-in sends multiple rows per request using a batch API. Rows marked for Update, Create, Delete, and custom actions are included in the batch requests.

For other REST service types, the add-in sends just one row per request.

The add-in uses an algorithm to determine the optimal size of the payload (number of rows) for a batch request as the upload operation proceeds. This is done to optimize performance and avoid timeouts. It works like this:

1. The add-in sends an initial set of batch requests using 10 rows per batch.
2. The add-in measures the response times for those initial requests and calculates the approximate number of rows that can be processed in 10 seconds.
3. Using the new value, the add-in sends one or more batches to the service.
4. The add-in recalculates the number of rows after each set of batch requests, using the target of 10 seconds per request, until all rows have been processed.



Note:

The target of 10 seconds is not guaranteed, since actual response times vary depending on the REST service business logic, server load, network latency, and so on.

If a batch request contains one or more errors, no changes are made by the service. In that case, a second batch request containing only the rows that succeeded during the first batch request is sent. If the second batch request fails, the add-in falls back to sending one row per request. For more information, see [Making Batch Requests](#).

If a set of changes includes custom actions and those actions have batch enabled, the changes are included in a batch request. However, if one or more rows is marked for a custom action that has batch disabled, the add-in reverts to sending the changes row by row. (See also [Batch Mode for Custom Actions](#).)

If you configure the add-in for parallel requests, it sends up to 4 batch requests in parallel depending on your settings. See [Configure Parallel Requests During Upload](#).

Upload Changes Using Upsert Mode

Oracle business object REST API services support an optional "upsert" mode that you can use when uploading changes to your REST service.

Ensure that your REST services business object supports upsert mode before you enable it.

When upsert mode is enabled, the REST service should behave as follows: first, it examines the incoming payload of a create row request. Next, if the payload contains enough information to match an existing row, the operation updates that existing item. If there is no match, the operation should create a new row. See [Updating or Creating Resource Items \(Upsert\)](#) for more information.



Note:

The exact behavior of upsert mode is determined by the service.

You enable upsert mode by selecting the **Use Upsert Mode for Create** from the General tab of the Business Object Editor. After you enable it, refresh any layouts in the workbook that are based on that business object.

Business Object Editor - Employees

General Fields Download Children Custom Actions Finders

Title
Employees

Description

Parent Business Object
Top-level business object (no parent)

Base Path Override

Use Upsert Mode for Create

Collection Path
/Employees Edit

Item Path
/Employees/{Employees_Id} Edit

If you make changes, redraw the associated layouts Done

When upsert mode is enabled, the add-in sends the HTTP header, `Upsert-Mode: true`, with POST requests for single-row creates. For batch mode, the add-in sets the operation value for each batch part to "upsert" instead of "create".

Omit Empty Values During Upload

You can configure Oracle Visual Builder Add-in for Excel to omit fields with empty values during upload.

Request payloads for Create and Update operations normally include a value for every column in the table (except those for read-only fields and custom action payload fields) even if the value is empty or null. Null values can potentially cause validation errors.

To avoid these errors, you can configure the add-in to omit null values by selecting the **Omit from payload if value is empty** check box in the Business Object Field Editor.

Business Object Field Editor

General List of Values

ID
department

Title
Department

Data Type
Integer

Format
Default

Description

Tooltip

Required for update
 Required for create
 Editable on update
 Editable on create
 Searchable
 Omit from payload if value is empty

Done

When selected, the add-in won't include the field in the payload during upload if the value in the field is empty.

The add-in sets the default value for the **Omit from payload if value is empty** check box based on the "nullable" field in the OpenAPI v3 (OA3) document. If "nullable" is true, the check box is deselected and null values are uploaded; if "nullable" is false (or the field is not present), the check box is selected and null values are omitted from the upload.

 **Note:**

If this check box is selected (null values are skipped), you won't be able to change a "non-null" value to a null value for a field in an existing row.

Data Consistency

When a workbook uses an Oracle business object REST API service that supports data consistency verification using an entity tag (Etag) mechanism, the add-in detects and reacts to the following scenario:

1. User A downloads information from a business object into a table in their integrated workbook.
2. User B downloads the same information into a table in their integrated workbook, edits it, and uploads changes.
3. User A then edits the same information (downloaded in Step 1) and uploads the changes.

4. The add-in provides the service with the necessary information (entity tags) to prevent User A's changes from overwriting those changes made by User B. Instead, when the server detects such a change, its response allows the add-in to display an error message similar to the following for any such rows in the table:

This row has been modified by another user. Please download before editing.

If you see this message, you'll have to discard your changes by downloading the latest data and then redoing your changes as needed.

For information about the entity tag (ETag) mechanism, see Data Consistency Tasks in *Accessing Business Objects Using REST APIs*.

If your workbook uses other types of REST services, the last writer wins. So, for the scenario just outlined, User A's changes in Step 3 will overwrite the changes of User B in Step 2.

Configure Parallel Requests During Upload

You can configure Oracle Visual Builder Add-in for Excel to send multiple requests in parallel during upload to improve performance and shorten the overall time to upload a large set of changes.

The add-in allows you to set from 1 to 4 requests in parallel. If you leave the setting on the default (4), the add-in sends four requests at the same time during upload instead of one request after another. If you choose a value of "1", the add-in sends only one request at a time during upload.

This feature is available for both layout types and can be set independently for each one.

Parallel requests behave differently than batch requests. With a batch request, the add-in sends the changes for multiple rows in each request; With parallel requests, the add-in sends multiple requests at the same time, regardless of whether the request contains one row or multiple rows.

Note:

While the add-in collects batches of rows in "table order" (from top to bottom), the threading used for parallel requests may result in batches of rows arriving at the REST service endpoint "out of order". For example, batch 7 may arrive at the endpoint before batch 6.

1. Open the Layout Designer for a worksheet and then click the Advanced tab.
2. Under Upload, drag the **Parallel Requests** slider to your desired setting.

We recommend setting the value to 4 parallel requests (the default) during upload. However, some services are not compatible with parallel requests due to their internal business logic. For such services, set this value to "1". Keep in mind that this setting generally results in slower upload performance.

 **Note:**

Take care to test thoroughly in a realistic environment when adjusting this value. Performance may actually degrade if you have too many parallel requests, as may happen if you have multiple users performing uploads at about the same time.

12

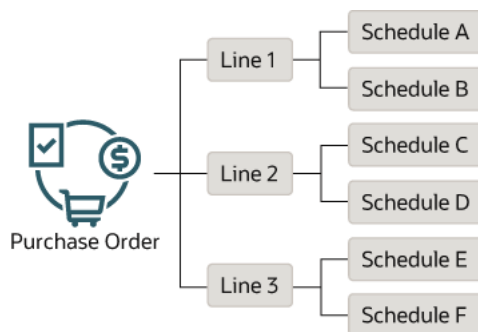
Use Multiple Layouts for Multi-level Business Objects

When business objects in your REST service have a parent-child relationship, you can create a set of dependent layouts and then perform operations, such as downloading data and uploading your changes, to all your layouts at once.

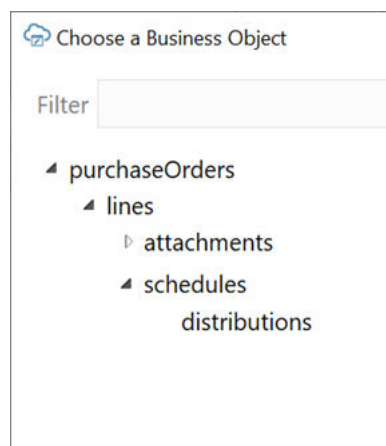
Business Object Hierarchies

Consider an example hierarchy of business objects where **purchaseOrders** is the parent, **lines** is the child, and **schedules** is the grandchild.

In this hierarchy, **purchaseOrders** is a collection of top-level purchase orders each with one or more lines for managing the details of each order. Each of these lines may include one or more schedules for tracking shipping details.



A hierarchy of business objects can go to even more levels ("great-grandchildren" and "great-great-grandchildren") or have more than one business object at each level ("siblings"). For example, the following hierarchy has "sibling" grandchildren (**attachments** and **schedules**) and a "great-grandchild" (**distributions**) under **schedules**.

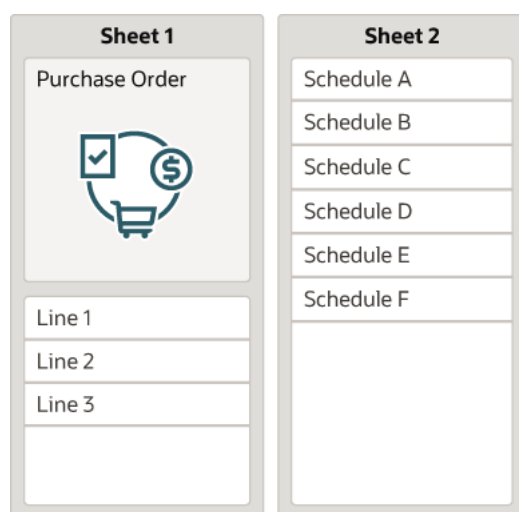


Configuring Dependent Layouts

To create a set of dependent layouts, create either a Form-over-Table or a Table layout as your top-level or "primary" layout on the first worksheet. Then create Table layouts for each subordinate level on separate worksheets. When you are done, link each dependent layout to its direct parent layout.

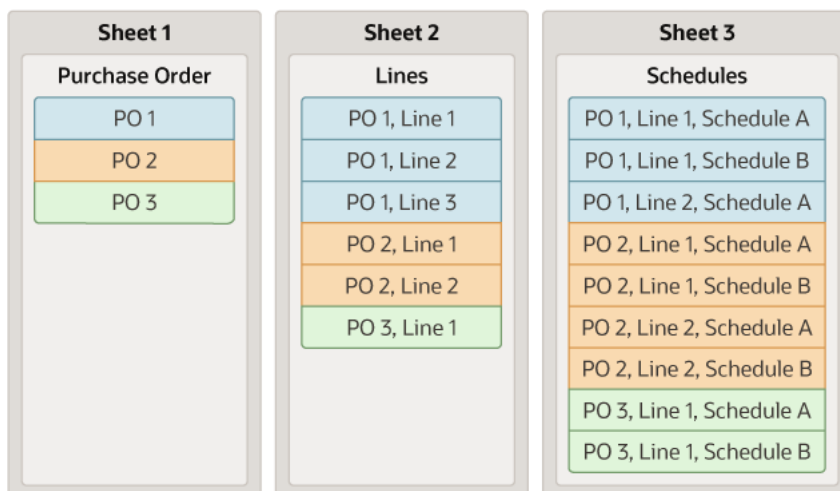
If you want to show a single purchase order and all its associated lines and schedules, create a Form-over-Table layout for the first two levels (**purchaseOrders** and **lines**) and a Table layout for the third level (**schedules**). You can then configure a search to prompt the user to enter a purchase order ID.

When the layouts are populated, the form part of the Form-over-Table layout displays details for the purchase order, the table part shows all the lines for the purchase order, and the dependent Table layout shows all the schedules for the associated lines.



You can also use a Table layout as your primary layout instead of a Form-over-Table layout. In this case, create Table layouts for each level in the hierarchy (**purchaseOrders**, **lines**, and **schedules**) on separate worksheets. As before, link each dependent layout back to its parent.

In this configuration, you can display one or more purchase orders in the primary Table layout, and all associated lines and schedules in the subsequent Table layouts.



 **Note:**

Keep in mind that a single parent may have multiple children with each of these having multiple grandchildren. Consequently, a large number of results in the primary layout may result in very large volumes of data in the subordinate layouts. To avoid performance issues when downloading and uploading data, Oracle recommends that you configure an appropriate query to limit the number of results in the primary layout. See [Use Search Parameters to Limit Downloaded Data](#).

"Siblings" are business objects at the same level and can be at the second level or deeper in your hierarchy. For example, you may have a parent business object (**Expenses**) at the top level and two children (**Itemizations** and **Distributions**) at the second level. In this example, **Itemizations** and **Distributions** are siblings of each other.

To create a set of dependent layouts for this hierarchy, you have two options. You can create a Form-over-Table layout for the parent and one of the children (for example, **Expenses** and **Itemizations**) on one worksheet and a Table layout for the other child (**Distributions**) on a second worksheet. Or you can create Table layouts for each business object (**Expenses**, **Itemizations**, and **Distributions**) on separate worksheets. Again, use a Form-over-Table layout if you want to show a single purchase order in the form and list all associated lines and attachments in the tables.

When you've created all the necessary layouts, remember to link each dependent layout back to the parent.

Once the dependencies are established, download, upload, and clear operations act on all the linked layouts, starting from the parent layout, followed by the child layouts, the grandchildren layouts, and so on.

If you allow your users to create new rows in a layout (**Create Enabled** is selected in the Table's capabilities), you must add one or more columns from the parent layout to uniquely identify the parent of the new child item. To create a new distribution record, for example, the user must be able to specify the correct line and schedule for the new distribution.

 **Note:**

You don't have to add columns to a second-level sibling layout if the parent is in a Form-over-Table layout since the sibling is linked to a unique parent in the form part of the layout.

Create a Set of Dependent Layouts

Create a set of dependent layouts for a hierarchy of business objects and link the layouts together.

To create a set of dependent layouts, create either a Form-over-Table or a Table layout as your primary layout on the first worksheet and then create Table layouts for the other business objects in your hierarchy on separate worksheets. When you are done, link each dependent layout to its direct parent layout. Finally, configure an appropriate query to limit the number of results in the primary layout.

For each dependent Table layout you create, you can include a column from the immediate parent layout to help your users keep track of the parent record for each row in the table. For example, if "ScheduleNumber" is displayed as a column in the schedules layout, you can add it to the distributions layout to see which distributions are associated with this schedule.

To enable the creation of items in a Table layout, you may need to add one or more columns to the layout from the layout's immediate parent. Including parent columns in the layout allows your business users to specify a unique parent for any new items they create. See [Support for Creating Items](#).

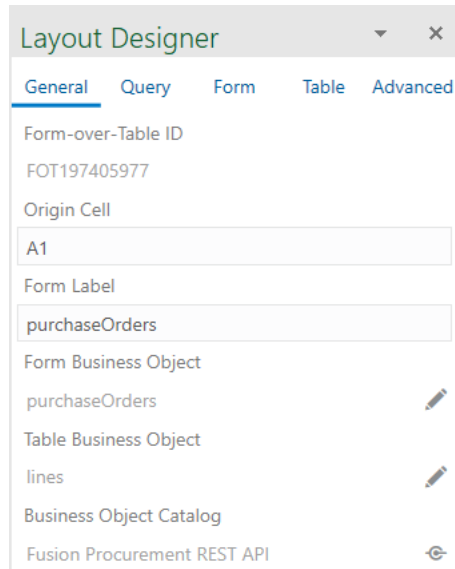
Let's use the example in this section to create a hierarchy of dependent layouts that mirrors your business object hierarchy.

 **Note:**

The primary layout in a set of dependent layouts can be a Form-over-Table layout or a Table layout. This topic covers how to create a set of dependent layouts that uses a Form-over-Table as the primary layout, but the principle is the same for using a Table layout as the primary layout.

1. Create a Form-Over-Table layout for the first two levels in the business object hierarchy.
 - a. In the Oracle Visual Builder tab, click **Designer**.
 - b. When prompted, provide the service description document.
 - c. Choose the parent business object. For example, select **purchaseOrders** to create a layout for purchase orders over lines (the first two levels in our hierarchy) and click **OK**.
 - d. Select **Form-over-Table Layout** as the new layout and click **OK**.
 - e. Choose the child business object. For example, select **lines** to complete the purchase orders over lines layout and click **OK**.

A Form-over-Table layout is created in the worksheet, where **purchaseOrders** is the form business object and **lines** is the table business object. This worksheet is now your primary layout.

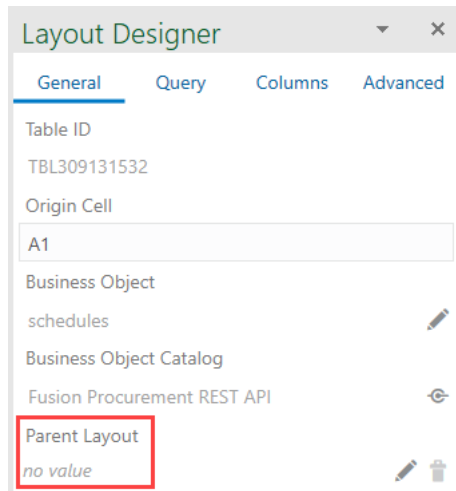


The screenshot shows the 'Layout Designer' window with the 'General' tab selected. The configuration is as follows:

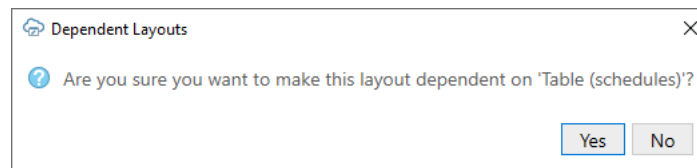
Property	Value	Action
Form-over-Table ID	FOT197405977	
Origin Cell	A1	
Form Label	purchaseOrders	
Form Business Object	purchaseOrders	Edit
Table Business Object	lines	Edit
Business Object Catalog	Fusion Procurement REST API	Refresh

2. Create a Table layout for each of the other business objects in your hierarchy. Ensure that you select the same business object catalog that is used by your primary Form-Over-Table layout.
 - a. Click the New Sheet icon to add a new worksheet.
 - b. Click **Designer**.
 - c. Choose the business object that's next in the hierarchy, for example, **attachments**, and click **OK**.
 - d. Select **Table Layout** as the new layout and click **OK**.
 - e. Repeat steps **a** to **d** to create Table layouts for all the other business object in your hierarchy. Continuing our example, create a Table layout for the sibling, **schedules**, and one for **distributions**.

A Table layout is created for each business object in your hierarchy. Notice the **Parent Layout** field in the Layout Designer's General tab, shown here for the schedules layout. This field shows only in layouts where the business object is a child of another business object in the same business catalog.



3. Set the parent layout for each Table layout in the hierarchy starting with the lowest level.
 - a. On the worksheet for the last item in the hierarchy (**distributions** in our example), click the Choose Parent Layout icon (✎) in the Layout Designer's General tab.
 - b. Select the appropriate layout from the Dependent Layouts window and click **OK**. If there is only one possible parent layout, a prompt appears asking you to confirm a parent for the layout. Click **Yes** to confirm the parent layout in the hierarchy, for example:

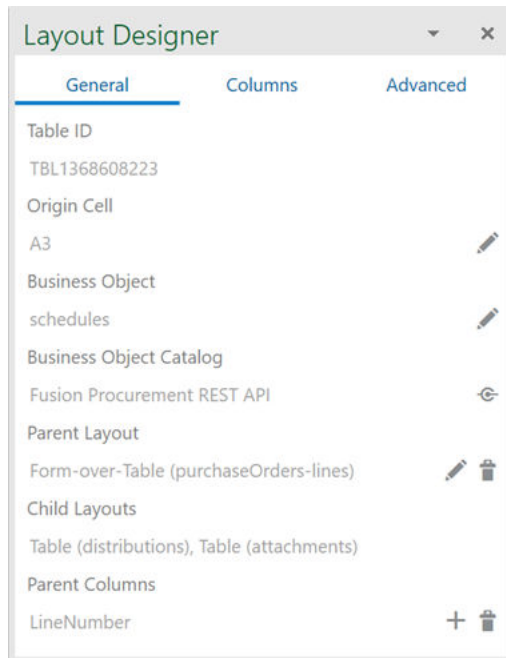


- c. Repeat this step for each Table layout in your business object hierarchy. Continuing our example, the parent layout for schedules is purchaseOrders-lines (which is also the primary layout). You don't need to set a parent for the primary layout.

 **Note:**

If you see a message, "No layouts found that this layout can depend on", it may mean the layout is not part of the same catalog or that the sibling business object you are trying to link has already been used in the table part of a Form-over-Table layout.

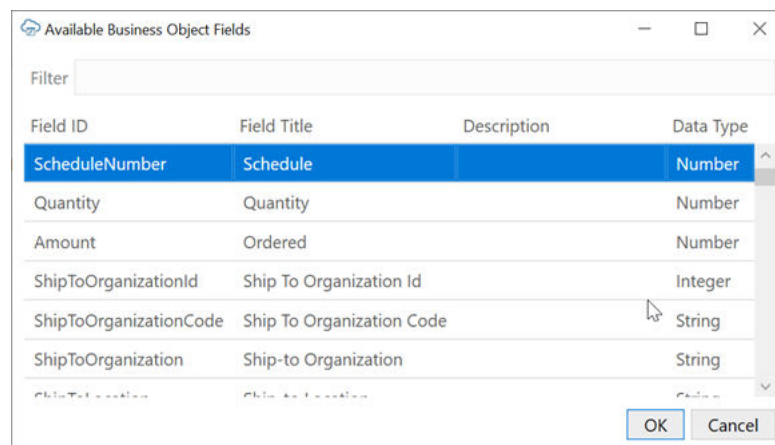
Once a layout has its **Parent Layout** defined, you'll notice a **Child Layout** field in its parent's Layout Designer. Also, search specifications (defined in the Query tab) are no longer available for any dependent layout that is not the primary layout. Here's the Layout Designer for the schedules Table layout, whose parent is purchaseOrders-lines and children are distributions and attachments.



4. If you plan to enable the creation of new items on layouts below the primary layout (for example, schedules and distributions), you may need to choose one or more columns from the parent layout to display in the layout.

If the desired parent field is already displayed as a column in the child layout, remove that column from the layout and instead add it as a parent column as described in this step.

- a. From the worksheet (distributions in our example), click the Add Parent Column icon (+) in the Layout Designer's General tab.
- b. From the Available Business Object Fields window, select a field from the parent layout and click **OK**. The field must be exposed as a column in the parent table. If you don't see the field you want to use, you'll need to add it as a column in the parent table.

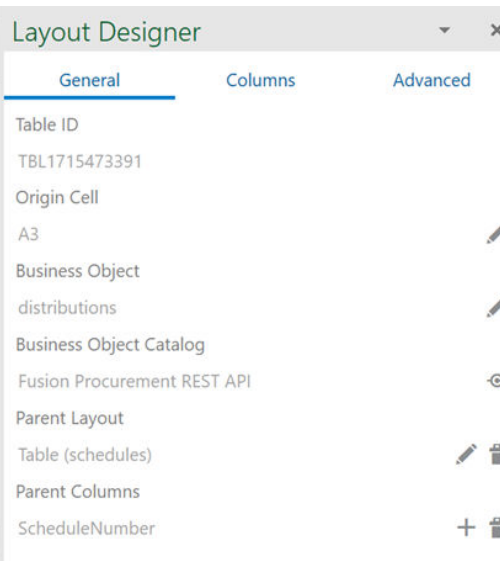


- c. Repeat steps **a** and **b** to add additional parent columns to the child layout as required.

- d. Repeat steps a to c to add parent columns to other Table layouts in the hierarchy.
5. Click **Redraw Layout** for each of the modified worksheets in your hierarchy.

A column for each parent field added is created in the child layout, for example, the **Schedule (schedules)** column as shown here:

C	D	E	F
Schedule (schedules)	Distribution	Quantity	Ordered
1.00	1.00	15.00	
1.00	1.00	15.00	
1.00	1.00	15.00	



Configuration for your dependent layout is complete. You can choose to configure the workbook further to limit the data that is downloaded to the primary layout. See [Configure Search Options for Download](#).

Before you publish and distribute your workbook to users, test the workbook to ensure that download, upload, or clear operations work on all layouts in the hierarchy. See Manage Data in a Dependent Layout in *Managing Data Using Oracle Visual Builder Add-in for Excel*.

Download, Upload, and Clear Operations on Dependent Layouts

When you download, upload, or clear data for a layout in a dependent hierarchy, the operation takes effect on all layouts in the hierarchy, starting with the primary layout, progressing to the next layout in the hierarchy, and continuing down until the last level in the hierarchy.

If the layout is not part of a set of dependent layouts, the operation is performed on the active layout only.

Settings such as those for queries and macros only apply to the primary layout and are not enforced on other layouts in the hierarchy.

 **Note:**

The "Download All/Stop Now" prompt also only applies to the primary layout. This prompt appears during download when the number of downloaded rows in the primary layout reaches the configured limit. Other pagination settings apply to all layouts in the hierarchy. For more information about the Limit Parameter settings, see [Configure Pagination for a Business Object](#)

Downloading Data

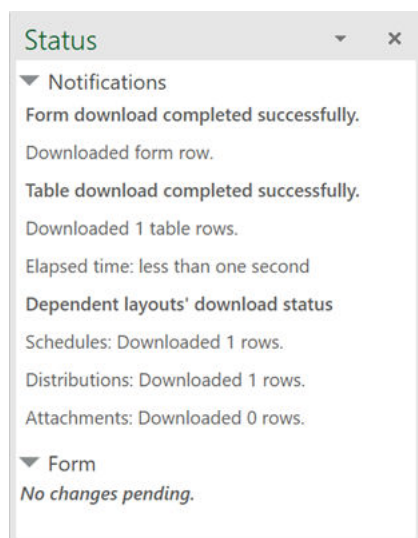
On download, the add-in first checks all layouts in the hierarchy for any pending changes. If there are changes pending, the user is prompted to confirm the download operation. If the user chooses to proceed, all pending changes are lost.

During download, the add-in first retrieves the values for the primary layout from the REST service. After the primary layout is populated, the add-in makes the next worksheet in the hierarchy active and retrieves all the appropriate items.

All items for all rows from the parent layout are downloaded at each level. (Any search specifications, if configured, apply only to the primary layout.) For example, when Sheet 1 in your workbook contains Purchase Orders as the parent and Lines as the child (containing, say, 10 Lines) and Sheet 2 contains Schedules as the grandchild, the Schedules table is populated with all Schedule items for all Lines. If each of the 10 Lines had two Schedules, the Schedules table would download 20 Lines.

The download operation proceeds through the rest of the table layouts in the hierarchy, retrieving all items for all rows from the parent layout.

When the operation finishes, the primary layout becomes active and the Status Viewer shows results for the primary layout as well as a summary for each layout in the dependent hierarchy, as shown in this example for a download operation:



Following a download, you can edit data much as you would in a Table or a Form-over-Table layout.

Uploading Changes

On upload, the add-in makes the primary layout active and sends all updates. If the primary layout uses a Form-over-Table, changes are sent first from the form and then from the table. The add-in then moves to the worksheet with the first dependent table layout and uploads changes before proceeding to the next layout.

Pending changes may include creation of new items, update or deletion of existing items, and invocation of actions on items. For rows pending Update, values in the parent column cells are not uploaded.

For new items on layouts below the primary layout, values in the parent columns must match a row in the parent layout. For example, to create a new distribution, you must specify an existing line and schedule in the parent columns with which to associate the new item. Empty parent column cells in the dependent layout or in its immediate parent layout result in creation failing.

The match is performed across all parent column cells. If one row in the parent table matches, it is used as the parent row. If more than one row matches, the first matching row is used. If no rows match, then the row is marked as "Create Failed".

When the operation finishes, the primary layout becomes active and the Status Viewer shows results for the primary layout as well as a summary for each layout in the dependent hierarchy.

Clear

When the clear operation is invoked, data is cleared from all the layouts in the dependent hierarchy.

Support for Creating Items

To enable the creation of items on layouts below the primary layout, add one or more columns to the layout from the layout's immediate parent. Including parent columns in the layout allows your business users to specify a unique parent for any new items they create.

The parent columns you choose must uniquely identify the parent row and must be exposed in the parent table. For example, to create a new distribution, a user must be able to specify an existing line and schedule with which to associate the new item. Therefore, you would want to add columns such LineNumber and ScheduleNumber to both the distributions and schedules layouts.

If no parent columns are configured, the table cannot support item creation and rows inserted into the table are ignored during upload.

Note:

You don't have to add columns to a second-level sibling layout if the parent is in a Form-over-Table layout since the sibling is linked to a unique parent in the form part of the layout.

Add a parent column to a dependent layout using the Add Parent Column icon (+) in the Layout Designer's General tab.

The field must be exposed as a column in the parent table. If you don't see the field you want to use in the Available Business Object Fields window, you'll need to add it as a column in the parent table.

Repeat the step to add additional columns from the parent layout and then redraw the layout when you're done. The parent columns appear after the Status column in the layout in the order you added them. The table header for the parent column uses the format "<field title> (<parent business object title>)" such as "Line Number (lines)".

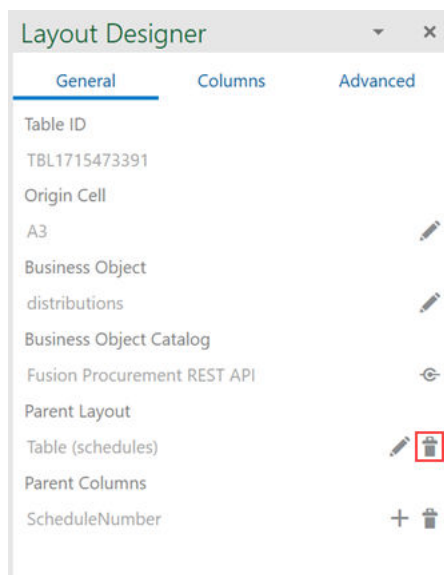
To clear the set of parent columns, use the Clear Parent Columns icon (🗑️) next to the Parent Columns property.

Delete a Dependent Layout

When your layout is part of a hierarchy of dependent layouts, the layout cannot be deleted without first removing its dependency in the layout hierarchy.

To delete a dependent layout:

1. Open the Layout Designer of the Excel worksheet whose layout you want to delete.
2. In the General tab, click the Remove Dependency icon (🗑️) next to Parent Layout.



3. When prompted, click **Yes** to remove the dependency.
4. Click **Delete Layout**, then confirm your selection.
5. On the other layouts in the workbook, click **Redraw Layout**.

13

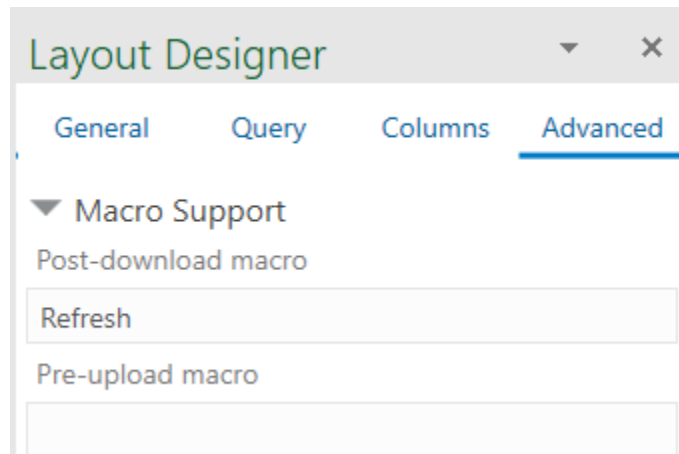
Use Macros in an Integrated Excel Workbook

You can configure macros that Oracle Visual Builder Add-in for Excel runs at specific points in the lifecycle of an integrated Excel workbook.

Use of this functionality requires you to use the Excel macro-enabled workbook type (.XLSM) and create your macros in a macro module. For more information about creating macros, see Microsoft documentation; describing how to create macros in an Excel workbook is outside the scope of this guide.

Some companies block the usage of Excel macros because they do not think macros are sufficiently secure. Consider your intended audience before you add a macro. You are also responsible for the security risks involved in using macros. So research the risks thoroughly before you deliver an integrated workbook to your customers. After creating a macro, take steps to protect the macro both from malicious and accidental alterations that might produce unexpected or harmful results. If a macro results in changes that are incompatible with the add-in or results in undesirable behavior, change the macro to avoid this behavior.

The Layout Designer's Advanced tab provides two properties where you can specify macros: the **Post-Download Macro** to run after download completes and the **Pre-Upload Macro** to run before an upload begins. Provide your macro names as the values of these properties. For example, when you've created a Refresh macro that you want to run after an upload, enter `Refresh` as the value of the **Post-Download Macro** property.



Tip:

Do not include the parentheses when specifying the name of the macro.

The macro that you specify for the **Post-Download Macro** property is not used if the user cancels download, if the table or form is empty, or in the event of an unexpected error. The macro that you specify for the **Pre-Upload Macro** property is used just before an upload. If the macro returns any value other than `true`, the upload operation quits and a notification appears in the Status Viewer. If the macro returns `true`, upload proceeds normally. To return

a true or false value from a macro, define a Boolean Function. See Microsoft documentation for details.

Here's example logic of an `IsUploadReady` function for a Pre-Upload Macro:

```
Function IsUploadReady() As Boolean
    Dim returnVal As Boolean

    On Error GoTo ErrHandler:

    Dim table As Range
    Set table = Sheets("Sheet1").Range("TBL349543489")
    ' The named range, TBL349543489, is managed automatically by the add-
in

    returnVal = True

    Dim cRows As Long
    cRows = table.Rows.Count
    Dim currentTableRow As Long
    Dim amount As Long
    For currentTableRow = 2 To cRows ' start with 2 to skip header row
        amount = table(currentTableRow, 10) ' Amount is the tenth column
in the table
        If amount < 0 Then
            returnVal = False
            Debug.Print "Found negative amount = "; amount
        End If
    Next

    IsUploadReady = returnVal
    Exit Function

ErrHandler:
    Dim failureMessage As String
    failureMessage = Err.Description
    MsgBox failureMessage
    IsUploadReady = False
    Exit Function
End Function
```

When an error occurs during the execution of a macro, Excel displays a Microsoft Visual Basic window to the user. We recommend that you implement a robust error handling strategy so that the window displays a useful message to the user who encounters an error during macro execution. The following is a simplistic example. The appropriate error handling strategy for a given macro depends on the logic in the macro.

```
Sub Refresh()

    On Error GoTo ErrHandler:

    ActiveWorkbook.RefreshAll
    Exit Sub
```



```
ErrorHandler:
    Dim failureMessage As String
    failureMessage = Err.Description
    MsgBox "Unable to refresh. Details: " & failureMessage
    Exit Sub
End Sub
```

 **Tip:**

The add-in creates and maintains named ranges for the data table. Your macros should never modify these named ranges. However, your macros can access the named range to locate the data table on a dynamic basis.

 **Note:**

Macro recording is incompatible with add-in features such as download and upload and is not supported. Do not attempt to record any add-in features. In some cases, you may see unexpected exceptions.

Do not leave the Excel Visual Basic editor's break mode on when you use **Download Data** or **Upload Changes**. It is not supported and can result in an unexpected exception.

Publish an Integrated Excel Workbook

Once you complete configuring an Excel workbook, you can publish it for users who perform data entry. Publishing creates a copy of the workbook and prepares it for distribution; for example, it lets you hide the Design tools and turn on worksheet protection for each worksheet with a layout.

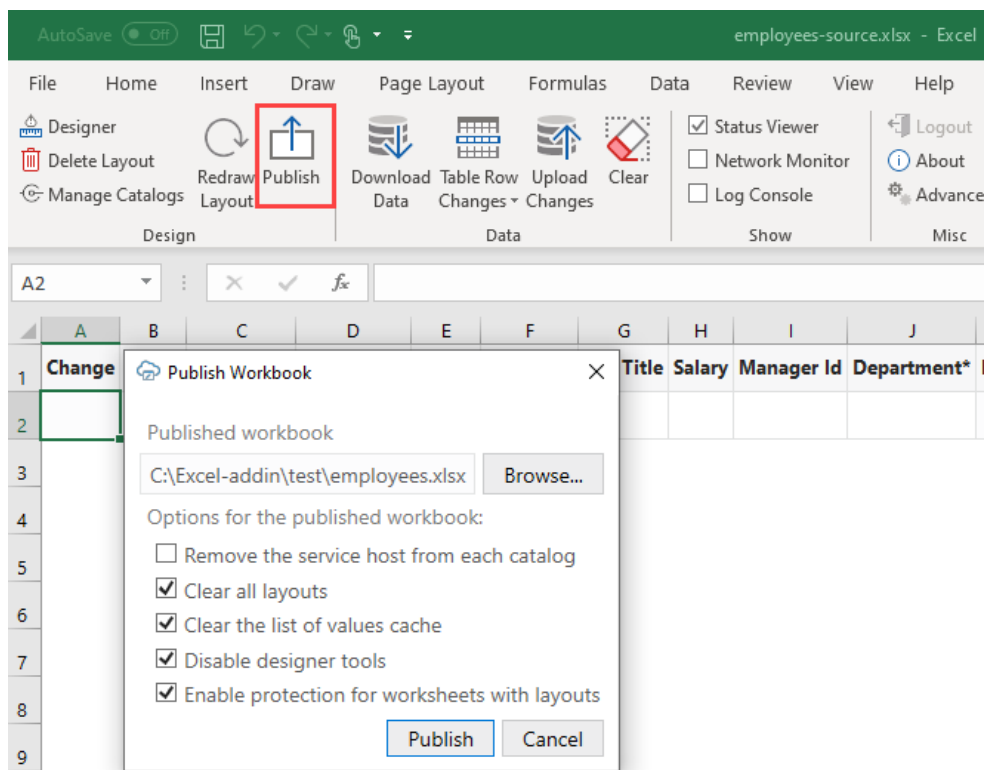


Note:

Publishing is optional. All data editing features of an integrated workbook are available in both published and unpublished copies of the workbook.

The recommended steps to take before you publish an Excel workbook are:

1. Complete configuration of the workbook.
2. Test the configuration thoroughly.
3. Use Excel's Inspect Workbook feature to review and remove personal information from the workbook.
Access the Inspect Workbook feature from Excel's File menu. When you use the Document Inspector to choose content to inspect and potentially remove, ensure that the Hidden Worksheets check box is not selected. You must not remove hidden worksheets from workbooks that you distribute because the add-in uses hidden worksheets to integrate the Excel workbook with the REST service.
4. In the Oracle Visual Builder tab, click **Publish**.



The Publish Workbook window opens. If the name of the original workbook ended with `-source` (for example, `employees-source.xlsx`), the add-in will offer the same name without `-source`.

5. To change the published workbook's directory and file name, enter the desired name and location in the Publish Workbook window, or click **Browse**. Make sure the published workbook doesn't use the same name as any open workbook (Excel won't let you use the same name for different workbooks, even if the file paths are different).
6. In the Publish Workbook window, choose other options for your published workbook:
 - If you want users to enter the service host when they open the published workbook, select **Remove the service host from each catalog**. This option lets users connect to another service host to access the REST service.
 - Select **Clear all layouts** to clear data downloaded to each layout in the published workbook. This option lets users download the latest data from the REST service in the published workbook.
 - Select **Clear List of Values Cache** to clear the list of values cache in the published workbook. This option is important if users might have different choices for a list of values in the published workbook.
 - If you don't want users of the published workbook to modify its layout, select **Disable Designer tools**. This option hides the design tools in the published workbook. (If the add-in was installed with the design tools already disabled, this option has no additional effect.)
 - If you want to prevent users from modifying read-only fields and performing Excel actions in the published workbook, select **Enable protection for**

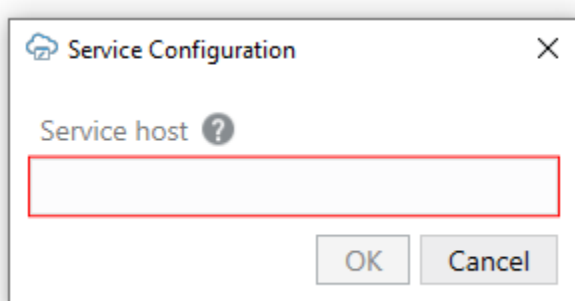
worksheet with layouts. This option enables Excel's worksheet protection for each worksheet with a layout.

7. Click **Publish**.
You'll see a notification in the Status Viewer that the workbook has been published to the specified directory.
8. Save the source version of the workbook, in case you need to make configuration changes post-publication.

Now that your workbook is published, you can distribute it to users for data entry.

There are a number of differences with the source workbook:

- If the service host value was removed for a workbook, users who open that workbook and perform an action that requires access to the REST service are prompted to enter the service host value, as shown here:

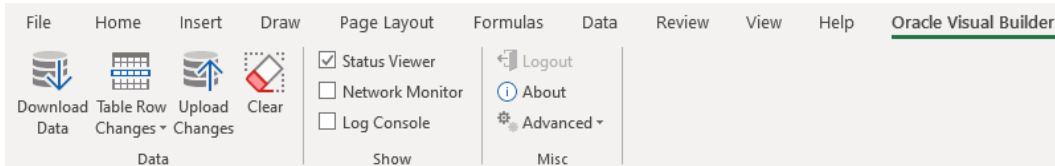


Actions that require access to the service include the Download Data and Upload Changes commands.

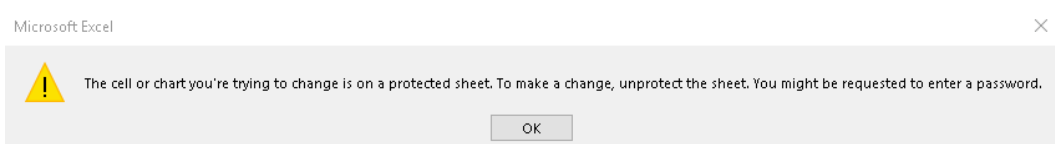
 **Tip:**

To change the URL of the service host when you don't have the Design tools installed, use the **Edit Service Host** option in the Advanced menu.

- If the Designer tools were disabled, tools such as the Designer, Delete Layout, and Publish do not appear in the Oracle Visual Builder tab, as shown here:



- If worksheet protection was enabled and a user tries to modify a read-only field, a message similar to the following image is shown:



Worksheet protection also prevents the user from performing Excel actions that might disrupt the workbook's integration with the service.

REST Service Support

This chapter provides additional technical details about how Oracle Visual Builder Add-in for Excel supports integration with REST services. It also provides information about technical known issues and limitations.

Service Types

Oracle Visual Builder Add-in for Excel provides support for the following service types:

- [Oracle business object REST API services](#)
- [Oracle REST Data Services \(ORDS\)](#)
- [Other services](#)

Supported Data Types

The add-in supports a variety of data types exposed by business objects in web applications developed using Visual Builder and data types exposed by REST services.

The add-in supports the following OpenAPI data types (derived from the JSON Schema Specification):

- Boolean
- integer
- object
- number
- string

In addition, the add-in recognizes the optional modifier property "format", when it is applied to values of type string. The two formats recognized are "date-time" and "date". There is no support for time, binary, or byte formats or for array-valued fields

The add-in ignores fields with unsupported data types when you create a Table layout or Form-over-Table layout in the Excel workbook. If, for example, a service that you use to retrieve data includes the `attachment` attribute data type, the add-in ignores it and does not create a column in the data table for this attribute type.

For more information, see the specifications for OpenAPI and JSON Schema:

- OpenApi: <https://github.com/OAI/OpenAPI-Specification/blob/master/versions/3.0.0.md#dataTypes>
- JSON Schema: <https://tools.ietf.org/html/draft-wright-json-schema-00#section-4.2>

Required Fields

When a business object field is created from a service description, the initial values of the **Required for Update** and **Required for Create** properties are set based on the following OpenApi property:

- The **Required for Update** value is determined by the request body schema of the PUT (or PATCH) operation for the item path, along with the OpenAPI [Required](#) array property.
- The **Required for Create** value is determined by the request body schema of the POST operation for the collection path, along with the OpenAPI [Required](#) array property.

When these PUT/PATCH or POST operations are not available, the Required properties are set using the response body schema of the collection GET operation and the OpenAPI Required array property. If the OpenAPI Required array property is not specified for a schema, the Required properties default to `false`.

You can always edit the **Required for Update** and **Required for Create** values in the Business Object Field Editor. See [Edit Service Descriptions and Business Objects](#).

REST Operations

Table and Form-over-Table capabilities are enabled for PUT, PATCH, POST, and DELETE operations as follows:

- Existing row updates are enabled if the item path has either a PUT or PATCH operation. For PATCH operations, the add-in includes all data values from editable cells for the changed row(s) on upload, regardless of whether the data value was edited since download.
- Create new rows is enabled if the collection path has a POST operation
- Delete existing rows is enabled if the item path has a DELETE operation



Note:

Even if the business object supports an operation, you can still choose to disable a layout's capability by deselecting it in the Layout Designer's Advanced tab.

Language Support

For every request that the add-in makes to the REST service, Oracle Visual Builder Add-in for Excel automatically adds the `accept-language` header.

By default, the value sent with the `accept-language` header is the language/culture code that Excel is currently configured to use. You can change the language as described in [Change the Add-in's Language](#).

Each REST service determines how and whether it will react to the `accept-language` header.

Oracle Business Object REST API Services

Oracle business object REST API services provide rich service descriptions that Oracle Visual Builder Add-in for Excel can analyze to provide a business object catalog with many details already filled out.

Oracle business object REST API services include many services offered by Oracle Cloud Applications as well as custom business objects in Visual Builder. When the add-in integrates an Excel workbook with Oracle business object REST API services, it supports special features such as:

- a [graphical search editor](#);
- [row finders](#);
- [batch upload](#);
- [polymorphic business objects](#);
- [custom actions](#) and,
- [upsert mode](#).

See [Accessing Business Objects Using REST APIs](#) and [Creating ADF RESTful Web Services with Application Modules](#).

Oracle REST Data Services

Oracle Visual Builder Add-in for Excel supports Oracle REST Data Services when you provide an OpenAPI description of an ORDS service. ORDS with AutoREST can provide an OpenAPI service description.

For example, use `http(s)://myhost.example.com:8888/ords/hr_demo/open-api-catalog/employees/` where:

- `myhost.example.com:8888` is the host and domain portion
- `hr_demo` is the schema/application
- `employees` is the database table

For other ORDS endpoints, you need to find or create an OpenAPI service description. See [Create a Service Description in Oracle Visual Builder](#). For information about AutoREST, see Automatic Enabling of Schema Objects for REST Access (AutoREST) in *Oracle REST Data Services Installation, Configuration, and Development Guide*.

Only basic authentication is supported when working with ORDS services.

After importing an ORDS service description, you can use the Business Object Field Editor to provide additional information about each field to improve the overall user experience. For example:

- Edit the field titles
- Designate certain fields as required
- Define lists of values. See [Use Lists of Values in an Excel Workbook](#).

Known Issues with ORDS

Refer to these known issues when planning to use Oracle REST Data Services.

- For some row-level errors, the ORDS server does not provide a specific reason for the error.
- In some cases, the ORDS server returns Create Failed for rows, when in fact the Create operation was successful. Re-downloading rows into the table will show the created rows.
- With an ORDS service, the PUT operation on the item path performs an "upsert" (see Update/Insert Table Row in *Oracle REST Data Services Installation, Configuration, and Development Guide*). So if you are about to update an existing row and someone else deletes that row, your update attempt may re-create that row. There's no warning or notice when this behavior occurs.

Other Services

Oracle Visual Builder Add-in for Excel can also be used with other service types as long as the service behaves as the add-in expects.

When using another REST service, provide an OpenApi service description as you would for other service types. See [REST Service Support Limitations](#) for more information on the add-in's expectations for any service.



Note:

The graphical search editor is not available for these services but search parameters can be used. See [Use Search Parameters to Limit Downloaded Data](#).

Object-typed Fields and Sub-fields

Oracle Visual Builder Add-in for Excel supports fields of type `Object`. These fields may expose sub-fields, also known as nested fields.

If, for example, you have an Employee business object with the following fields:

- First Name (type: String)
- Last Name (String)
- Address: (Object)
 - Street (String)
 - City (String)
 - State (String)
 - Zip (String)
 - GPS Coordinates (Object)
 - * Latitude (Number)

- * Longitude (Number)
- Hire Date (Date)

In this example, the type of the Address field is `Object` and it contains sub-fields. Object fields should not be confused with arrays. In this example, an Employee has only one Address. The add-in does not support fields that are typed as arrays.

The add-in handles Object fields and their sub-fields in the following manner:

1. First, in the Business Object editor **Fields** tab, only the top-level fields are listed. In this example, the top-level fields are: First Name, Last Name, Address, and Hire Date. To edit the properties for sub-fields of Address, edit Address and find the Sub-fields list on the Field Editor window. The direct sub-fields for Address are Street, City, State, Zip, and GPS Coordinates. Since GPS Coordinates is of type `Object`, its field editor will show its sub-fields (Latitude, Longitude).
2. Next, when creating a Table layout from a business object's fields, the add-in promotes the sub-fields and creates columns for each (leaf) sub-field. This maintains a regular, rectangular structure for the table in the worksheet. So, the above example generates a table with these columns:
 - First Name
 - Last Name
 - Address / Street
 - Address / City
 - Address / State
 - Address / Zip
 - Address / GPS Coords / Latitude
 - Address / GPS Coords / Longitude
 - Hire Date

REST Service Support Limitations

Refer to these limitations when planning to integrate a workbook with a REST service using Oracle Visual Builder Add-in for Excel.

Many different request and response schema types are possible and we cannot list all that are compatible with the add-in. If a particular structure is not listed explicitly as supported, it may not work.

- Only REST API services that return `application/json` media types as response payloads are supported. Other media types (for example, XML and `octet-stream` responses) are not supported.
- Child business objects cannot be used for the form part of a Form-over-Table layout.
- Polymorphic business objects cannot be used to create a layout.
- Asymmetrical field lists. Since download, editing, and upload all occurs in the same Excel rectangular grid, the add-in counts on having a single set of field IDs (JSON member names) for both download and upload. If the REST service uses different field IDs for the same information when completing different operations, it cannot be used effectively with the add-in.
- Fields with forward slash (`/`) in the member name:

- OpenApi documents contain schema properties that represented in JSON as something like "memberName" : { . . . properties describing the field ... }
- When creating the business object field from the JSON member, the add-in uses the member name as the field ID.
- Field IDs that include the / character are incompatible with the add-in, so such members will not be represented as fields in the business object.
- When providing a URL for an OpenAPI service description, ? metadataMode=minimal is not supported.
- Remote references are not supported in OpenApi service descriptions.

If a REST service owner makes significant changes to the service after the workbook is configured to integrate with the service, the integration may not function as expected. In such cases, you can either re-import the service description and create a new layout, or refresh the business object catalog. If the change is minor, you can update the business object details to match the change in the service. See [Edit Service Descriptions and Business Objects](#).

16

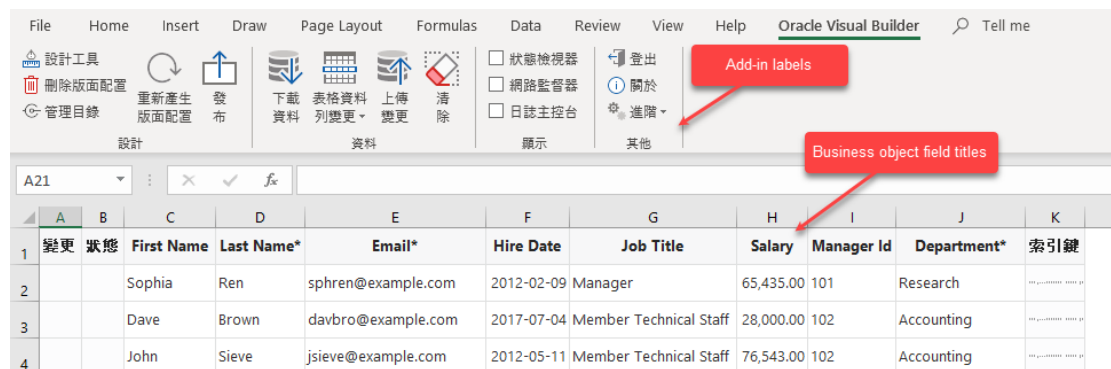
Internationalization

Oracle Visual Builder Add-in for Excel is available in various languages. It automatically detects the user's preferred language from Microsoft Excel and uses that language where possible.

The date, date-time, and number formats used by the add-in are culture-sensitive. See also [Appearance of an Integrated Excel Workbook](#).

When using the add-in, you work with:

- Labels visible on the Oracle Visual Builder ribbon and in various windows displayed by the add-in. These labels, known as the **add-in labels**, are owned by the add-in and are localized.
- Labels visible as column headers and field labels are known as **business object field titles**. These titles are owned by the REST service.



The add-in sends the `accept-language` header to the service on every request. The language setting specified for Excel is used for the add-in labels and for requests to the service, including the describe requests that fetches the initial business object field titles.

Note:

Because business object field titles are owned by the service, contact the service owner for any missing translations or languages.

Change the Add-in's Language

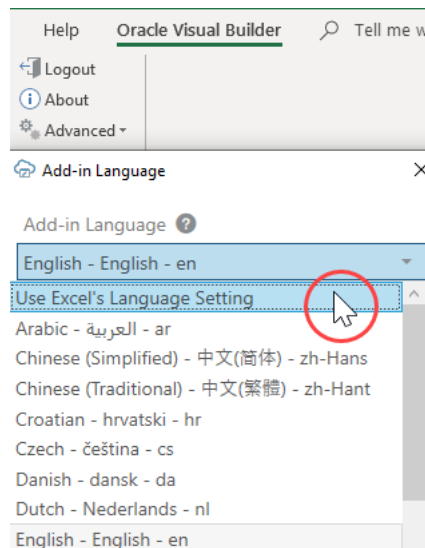
You can change the language that the Excel add-in uses. Do this if you want to evaluate your integrated workbook with different languages.

1. In Excel, click the **Oracle Visual Builder** tab.
2. Choose **Select Language** from the Advanced drop-down.
3. In the Add-in Language drop-down list that appears, select the language you want to use. The drop-down list displays the languages that the add-in supports.

4. Click **OK**.
5. Restart Excel to make your changes take effect.

The add-in's user interface elements (**Download Data** and so on) now use the language you selected. If the selected language uses a right-to-left writing system, the add-in's user interface elements appear in right-to-left mode. The language that Excel uses remains unchanged, as does the format used for dates, times, and numbers. See Excel or Windows options to change Excel's language and formats for dates, times, and numbers.

The language that you choose for the add-in language is stored in a local file in the Windows user profile. You can select the **Use Excel's Language Setting** option in the Add-in Language drop-down list to remove this setting for the current user.



Refresh All Field Titles

If your integrated workbook uses an Oracle business object REST API service, it is possible to refresh all field titles.

This functionality is handy when switching languages. Here's an example of when to reset field titles: Imagine an invoices workbook was configured and tested with English as the current language, so all field titles are in English and this information is saved with the workbook. Now imagine this workbook was sent to someone in France for data entry. That person has Windows and Excel configured for French, and she would like to see field titles in French as well. Resetting the field titles enables this data entry operator to see field titles in French.

▲ Caution:

A reset wipes out any local changes made to field titles using the Business Object Field Editor.

1. In Excel, click the **Oracle Visual Builder** tab.
2. Choose **Refresh Field Titles** from the Advanced drop-down.

3. In the Field Titles window, click **Yes**.

 **Note:**

The reset functionality relies on the `/describe` response from the Oracle business object REST API service, with the add-in sending the current language preference in the request. A given service may or may not have translations for the requested language. For any missing translations, contact the owner of the service.

Security Best Practices

When using Oracle Visual Builder Add-in for Excel, follow these security-related best practices and recommendations.

Security Guidelines

Follow these best practices:

- Update the add-in to the latest version available.
- Restrict access to Excel documents containing sensitive data.
- Consider adding passwords to workbooks to further reduce exposure.
- Always use HTTPS endpoints instead of HTTP.
- Do not use basic authentication.
- Ensure that the latest Windows updates and security patches have been applied to the computers where you install the add-in.
- Turn off older obsolete security protocols, such as SSL.
- Consider using Excel's Inspect Workbook feature (available on Excel's File menu) to review and remove personal information from the workbook before you distribute it. When you use the Document Inspector, make sure the Hidden Worksheets check box is not selected. You must not remove hidden worksheets, because the add-in uses hidden worksheets to integrate a workbook with the REST service.

Basic Authentication

The add-in supports basic authentication:

- When using REST service endpoints protected by basic authentication, the user is prompted for credentials when the add-in connects to the endpoint.
- When used with HTTP, basic authentication is not secure. Basic authentication should only be used with HTTPS, and preferably only in non-production environments.

JSON Web Token

In addition to basic authentication, the add-in also supports authentication for REST services exposed by Oracle Cloud applications that use the JSON Web Token (JWT) relay servlet. No configuration is required by you. The add-in automatically detects whether the Fusion application's service has the `/anticsrf` and `/tokenrelay` endpoints configured.

During authentication with JWT, the add-in uses the embedded browser to display a pop-up browser window and navigate to the hosting web application's log-in page. See [Embedded Browsers](#).

When the user provides valid credentials, the pop-up automatically closes and access to the service can proceed using the token obtained during the log-in sequence.

Use of the JSON Web Token (JWT) relay servlet is only available for Fusion applications, as the path to the token relay service that the add-in uses is specific to Fusion applications.

 **Note:**

In this release of the add-in, using **self-signed** certificates with the JWT relay servlet will not work. A valid certificate issued from a well-known root certificate authority should work fine with the JWT relay servlet.

TLS Security

When the add-in connects to a REST endpoint using HTTPS, the add-in enables the system default behavior for TLS to determine which TLS protocol is to be used. Because the add-in runs within the Excel process, it cannot rely entirely on the .NET 4.7 default setting to do this. To ensure that the system default behavior is in effect, the add-in sets the *AppContext.DontEnableSystemDefaultTlsVersions* property to false for the current app domain.

See the following Microsoft documentation:

- [If your app targets .NET Framework 4.7 or later versions](#)
- [Configuring security via AppContext switches \(for .NET Framework 4.6 or later versions\)](#)

Troubleshoot Excel Workbooks

If you experience issues with Oracle Visual Builder Add-in for Excel, follow the steps here to identify and resolve issues. If you still can't resolve your issue, contact [Oracle Support](#).

- Review the documentation to make sure the desired operation is supported.
- Download and run the [Client Health Check Tool](#).
- Make sure you're on a [supported platform](#).
- Upgrade to the [latest version](#) of the add-in.
- Apply available [Microsoft updates](#).
- Close all workbooks, exit Excel, and try again with simple steps.
- Generate an add-in log for review. See [Logging](#). If you contact Oracle Support, you may be asked to provide this log.

Caution:

Log files may include personal information from your computer including, but not limited to, the computer's name and your Windows profile name. Be sure to select the appropriate option when uploading files with personal information to a service request so that the file access can be restricted as needed.

- Generate a diagnostic report if required. See [Diagnostic Report](#).

Note:

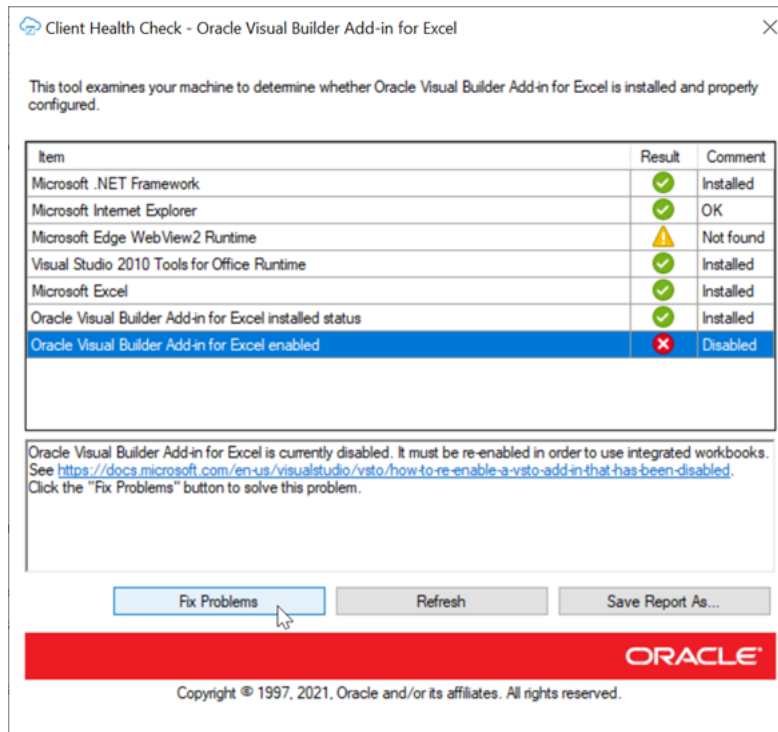
If you are experience network issues such as "bad request" errors during normal operations, there may be an issue with the REST service. Contact the owners of the REST service to determine whether the service is providing the expected response. You can use the Network Monitor to capture details of the request-response pairs to share with the REST service owners. See [Network Monitor](#) for the steps.

Check Your Environment

Run the Client Health Check Tool to check if the desktop configuration and environment are suitable for Oracle Visual Builder Add-in for Excel and to resolve issues.

Download the latest version of the Client Health Check tool (`vbafe-health-check.exe`) from the Oracle [Downloads](#) page.

1. Run `vbafe-health-check.exe` and review the result for each examined item.



Select each failed item (✗) for more details, including the steps necessary to resolve the issue. If the add-in can resolve the issue, the **Fix Problems** button is enabled.

Note:

The Client Health Check also shows a warning (⚠) for any optional software that is missing. You don't have to install this software to use the add-in.

2. Click **Fix Problems** if available or follow the instructions to resolve the issue.
3. If Oracle Support requests a copy of the report:
 - a. click **Save Report As...** and choose a name and location for the report.
 - b. Review the report and remove any sensitive information.
 - c. Send the report to Oracle Support.

Apply Microsoft Updates

When troubleshooting issues with Oracle Visual Builder Add-in for Excel, we recommend first applying all pending Windows, Office, and Internet Explorer updates before reproducing the issue. A Microsoft patch may resolve your problem.

1. From the Windows Start menu, select **Settings, Update & Security**, and then **Windows Update**.
2. If updates are available on the Windows Update page, review the updates and click **Install Now**.

 **Note:**

The details of applying Windows updates can vary from version to version and also according to your company's IT policy. Check with your system administrator for assistance, if needed.

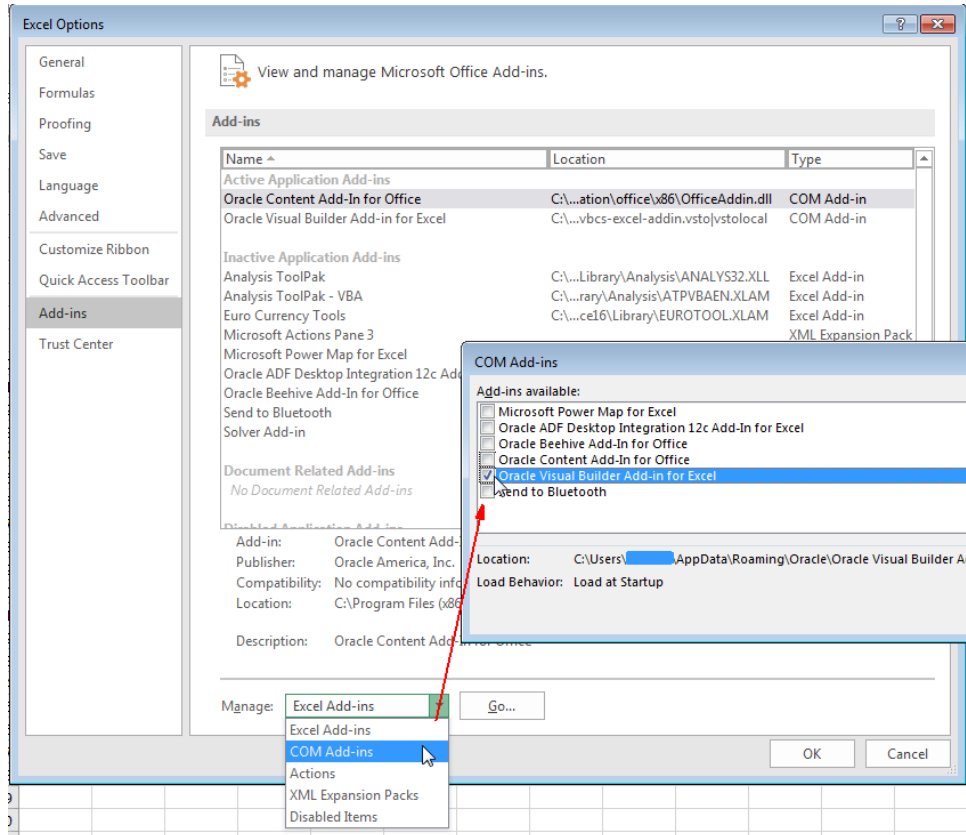
3. If you are using Internet Explorer (IE) version 10 or earlier, upgrade to version 11 now. The add-in uses Internet Explorer for some of its functionality. Microsoft only supports Internet Explorer version 11 on the platforms that the add-in supports. See Embedded Browser Notes.

Re-Enable Oracle Visual Builder Add-in for Excel

If your add-in becomes disabled and you are unable to use the [client health check tool](#), you can re-enable Oracle Visual Builder Add-in for Excel through Microsoft Excel.

1. In Excel, click **File > Options > Add-Ins**.
2. Select **COM Add-ins** in the Manage drop-down list and click **Go**.
3. Clear the **Oracle Visual Builder Add-in for Excel** check box and click **OK**.
4. Restart Excel.
5. Enable the add-in by repeating the steps and instead selecting the **Oracle Visual Builder Add-in for Excel** check box from the **Add-ins available** list in the COM Add-ins

dialog.



Network Monitor

Use the Network Monitor window to inspect the content of REST service calls between your Excel workbook and the REST service if you encounter unexpected behavior.

The Network Monitor window provides information such as the start time, the elapsed time, and response for each REST call that originates from the workbook. In addition, it provides the JSON payloads that the Excel workbook and the service exchange.

If you encounter issues with the REST service such as "bad request" errors, you can capture information about the error in the Network Monitor window and share this information with the owner of the REST service.

The Network Monitor window generally goes to the background while you perform the steps of your use case. Bring the window forward to see the details of each request and response. You can select and copy information from the window. You may need this detail when troubleshooting problems with the service owner. The window shows up to 100 request-response events. Older events are discarded as new ones are added.

1. In Excel, click the **Oracle Visual Builder** tab.
2. Select **Network Monitor**.
The Network Monitor window displays.
3. Repeat the steps that lead to the issue.
4. Review the details of each request and response.

5. Optionally, right-click an entry for a REST service call in the upper table, and choose the **Save request-response details as ...** option to save the information to a file.

 **Caution:**

Request and response payloads may include sensitive information, including actual data and personally identifiable information. Be sure to handle these payloads with due care.

Logging

When reporting an issue about the add-in, generate a detailed log file that captures the steps that lead to the problem you want to report.

 **Caution:**

Log files may include personal information from the user's computer including, but not limited to, the computer's name and the end user's Windows profile name. Be sure to select the appropriate option when uploading files with personal information to a service request so that file access can be restricted as needed.

1. In Excel, click the **Oracle Visual Builder** tab.
2. Select **Log Activity** from the Advanced drop-down list to specify a directory location and file name for the log file. This starts the logging session.
3. Repeat the steps that lead to the issue.
4. Exit Excel completely to stop the logging session and before you access the log file.

 **Note:**

The next time you run Excel logging will no longer be enabled.

The log file that you generate captures information about steps during an Excel session.

Logging Console

The Logging console displays log messages based on the actions performed. If you encounter any issues, view the logging messages to troubleshoot and diagnose issues.

1. In Excel, click the **Oracle Visual Builder** tab.
2. Select **Log Console**.
The Logging Console window displays.
3. Repeat the steps that led to the issue.
4. Review the logged messages.
5. Do one or more of the following as required:
 - Type a word or phrase in the **Filter** box to display matching log entries.

- To display more details such as Time, Thread, and Level, click **Show Event Details**.
- For verbose logging, click **Enable Verbose Output** and repeat the steps that led to the issue.
- Click **Clear** to discard all log entries.

Diagnostic Report

The diagnostic report contains information that can help resolve issues. Provide a diagnostic report when reporting a problem with Oracle Visual Builder Add-in for Excel.

1. In Excel, click the **Oracle Visual Builder** tab.
2. Select **Diagnostic Report** from the Advanced drop-down.
3. Save the diagnostic report to a directory location with a file name of your choice.
4. Review the content of the diagnostic report to remove any sensitive information that you do not want to share before you use it to report an issue.

19

Migration

You can migrate an Excel workbook created or modified with version 2.x of the add-in to use the current version of the add-in.

Workbooks created or modified with version 2.x of the add-in migrate seamlessly to the latest version of the add-in. No special steps are required, other than the usual upgrade recommendations (see [Upgrade to the Latest Version](#)).

In general, your workbook should continue to function after the upgrade as before. If you want to take advantage of new add-in features, you may need to make some changes to the workbook configuration. However, once you configure your workbook to use the latest features, it may no longer be compatible with the older version of the add-in. So before distributing your updated workbook, make sure your target audience has access to the corresponding add-in version.

Embedded Browsers

Oracle Visual Builder Add-in for Excel uses an embedded web browser to display web pages from inside Microsoft Excel.

Specifically, the embedded web browser is used to display the log-in web page when authenticating using JSON Web Token (JWT). See [JSON Web Token](#).

The add-in uses one of the following two embedded browsers:

- WebView2 based on Edge/Chromium
- The .NET WebBrowser control based on Microsoft Internet Explorer technology

The add-in uses WebView2 if the appropriate runtime is detected on the local computer. Otherwise, it uses the .NET WebBrowser control. There is no option to choose the embedded web browser. Your default web browser setting in Windows Settings has no effect on the add-in.

WebView2

WebView2 is an embedded web browser based on Edge/Chromium. In order for the add-in to use WebView2, the WebView2 runtime must be installed on each computer where the add-in runs.

You can download the runtime from here: <https://developer.microsoft.com/en-us/microsoft-edge/webview2/#download-section>.

You should choose one of the "evergreen" installers. Do not choose the "Fixed Version" option.



Note:

The WebView2 runtime may already be present on your computer if you have Microsoft 365 Apps installed. See [Microsoft Edge WebView2 and Microsoft 365 Apps](#).

The .NET WebBrowser Control

If the add-in cannot find a compatible WebView2 runtime, it uses the .NET WebBrowser control instead. The .NET WebBrowser control is based on Microsoft Internet Explorer technology.

There is no special installation required. This control is part of Microsoft .NET.

Since the .NET WebBrowser control is based on Microsoft Internet Explorer, make sure that Internet Explorer has been upgraded to version 11 and that all Microsoft updates and security patches have been applied.

 **Note:**

Support for the .NET WebBrowser Control is deprecated and may be removed in a future release. You should install the WebView2 runtime as soon as possible.

How to avoid using Internet Explorer

If you wish to avoid using Internet Explorer with the add-in, make sure that the WebView2 runtime is installed as described above.

Technical Notes Regarding WebView2

If there is any problem with the log-in web page, please contact the page owner. Such pages are outside the scope of the add-in. Let the page owner know that the page needs to be compatible with Edge/Chromium.

When the add-in uses the WebView2 browser control, the browser's "SmartScreen" feature is disabled. See the [Microsoft Defender SmartScreen Frequently Asked Questions](#) or the [documentation](#) for more information

The WebView2 browser control uses a user data folder on the local computer to store browser data, such as cookies, permissions, and cached resources. This folder can be found under:

```
%LocalAppData%\Oracle\Visual Builder\  
For example, C:\Users\username\AppData\Local\Oracle\Visual  
Builder\EBWebView
```

See [Manage the User Data Folder](#) in the Microsoft Edge documentation.

Technical Notes Regarding the .NET WebBrowser Control

If there is any problem with the log-in web page, please contact the page owner. Such pages are outside the scope of the add-in. Let the page owner know that the page needs to be compatible with Microsoft Internet Explorer.

When troubleshooting a problem, be sure to pay special attention to your IE configuration. There are key settings under Internet Options that can influence the behavior of the add-in. These settings include the LAN settings, Security settings, Languages settings, and Certificates settings.

 **Note:**

The add-in installer sets the `FEATURE_BROWSER_EMULATION` feature control key for Excel in the registry to IE 11 mode. See [Browser Emulation](#) in the Microsoft Internet Explorer documentation.

Third Party Licenses

Oracle Visual Builder Add-in for Excel includes the following third-party software.

Microsoft.OpenApi, Version 1.2.3

Copyright (c) Microsoft Corporation. All rights reserved.

MIT License

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED *AS IS*, WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

© 2019 GitHub, Inc.

Microsoft.Web.WebView2, Version: 1.0.705.50

Copyright (c) Microsoft Corporation. All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

- Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
- Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
- The name of Microsoft Corporation, or the names of its contributors may not be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT

LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

NewtonSoft.Json, Version 12.0.3

The MIT License (MIT)

Copyright (c) 2007 James Newton-King

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

SharpYaml, Version 1.6.6

Copyright (c) 2013-2016 SharpYaml - Alexandre Mutel

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

 SharpYaml is a fork of YamlDotNet <https://github.com/aubry/YamlDotNet> published with the following license:

Copyright (c) 2008, 2009, 2010, 2011, 2012 Antoine Aubry

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.