Oracle Banking Extensibility Workbench

User Guide

Release 14.3.0.0.0

Part No. F25382-01

December 2019

Oracle Financial Services Software Limited

Oracle Park

Off Western Express Highway
Goregaon (East)
Mumbai, Maharashtra 400 063
India
Worldwide Inquiries:
Phone: +91 22 6718 3000
Fax: +91 22 6718 3001
www.oracle.com/financialservices/

Contents

Introduction

This guide contains the steps for the user of the Oracle Banking Extensibility workbench to get familiarize with the OBX tool.

Abbreviations

Below are set of abbreviations used in the guide:

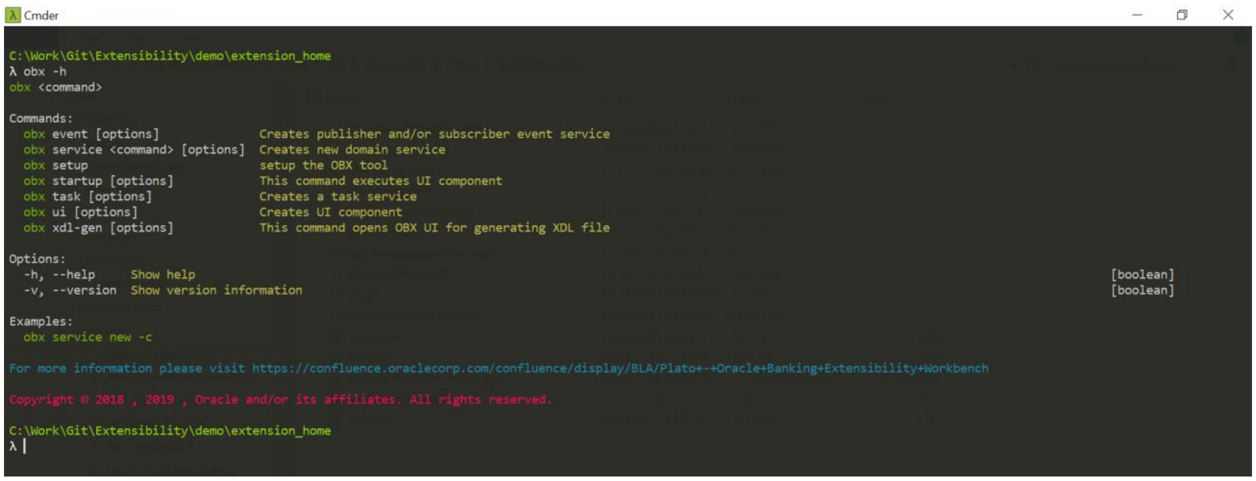
- OBX – Oracle Banking Extensibility Workbench
- WC – Web Components
- SMS – Security Management System
- XDL – Extensibility Domain Language
- CMC- Common Core Management

Pre-requisite

- It is assumed that OBX installation guide has been followed and all the pre-requisite mentioned in the document are done.

OBX Tool

Once all the installation and pre-requisite is successful done we will be able to see the obx tool running using command obx -h



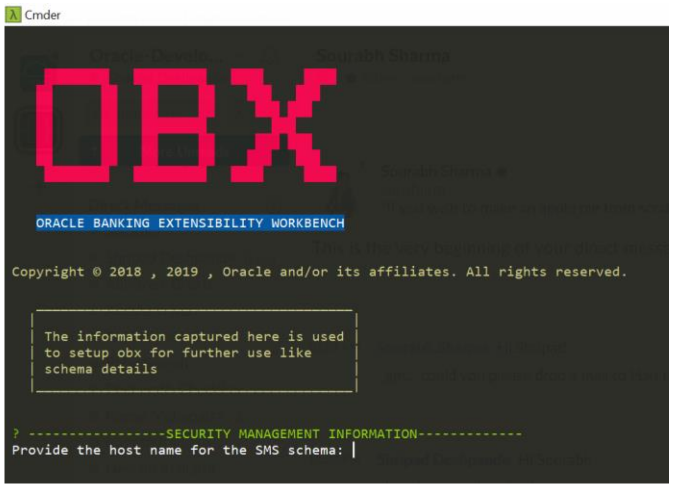
```
C:\Work\Git\Extensibility\demo\extension_home
λ obx -h
obx <command>

Commands:
  obx event [options]              Creates publisher and/or subscriber event service
  obx service <command> [options]  Creates new domain service
  obx setup                        setup the OBX tool
  obx startup [options]            This command executes UI component
  obx task [options]               Creates a task service
  obx ui [options]                 Creates UI component
  obx xdl-gen [options]            This command opens OBX UI for generating XDL file

Options:
  -h, --help     Show help                                                                                  [boolean]
  -v, --version  Show version information                                                                   [boolean]

Examples:
  obx service new -c

For more information please visit https://confluence.oraclecorp.com/confluence/display/BLA/Plato+-+Oracle+Banking+Extensibility+Workbench

Copyright © 2018 , 2019 , Oracle and/or its affiliates. All rights reserved.

C:\Work\Git\Extensibility\demo\extension_home
λ |
```

OBX Setup Command

- Make sure you have JAVA_HOME system variable in the environment variables. Also add the jdk path till bin folder in the PATH Variable of environment variables.
- Before proceeding further and firing obx setup command one should know about the Schema details from product processor or base application. These schemas will be **SMS**, **Common Core**, **Plato UI Config** and **Entity**. Entity schema should be created separately as product processor will not have it and it will comprise of all the extensions related tables.
- Navigate to URL and download https://chromedriver.chromium.org/downloads for the version ChromeDriver 78.0.3904.105. This will be used for selenium testing for UI
- Unzip the chrome driver and add it into the environment variable system path

Steps

- First and foremost, step for creating any extension is to setup OBX tool, this will capture some important information which we need repeatedly during the course of extension creation and integration with main application
- To setup OBX execute the command '**obx setup**' from extension_home folder using **cmder** in admin mode, this will open the following prompts in the console emulator



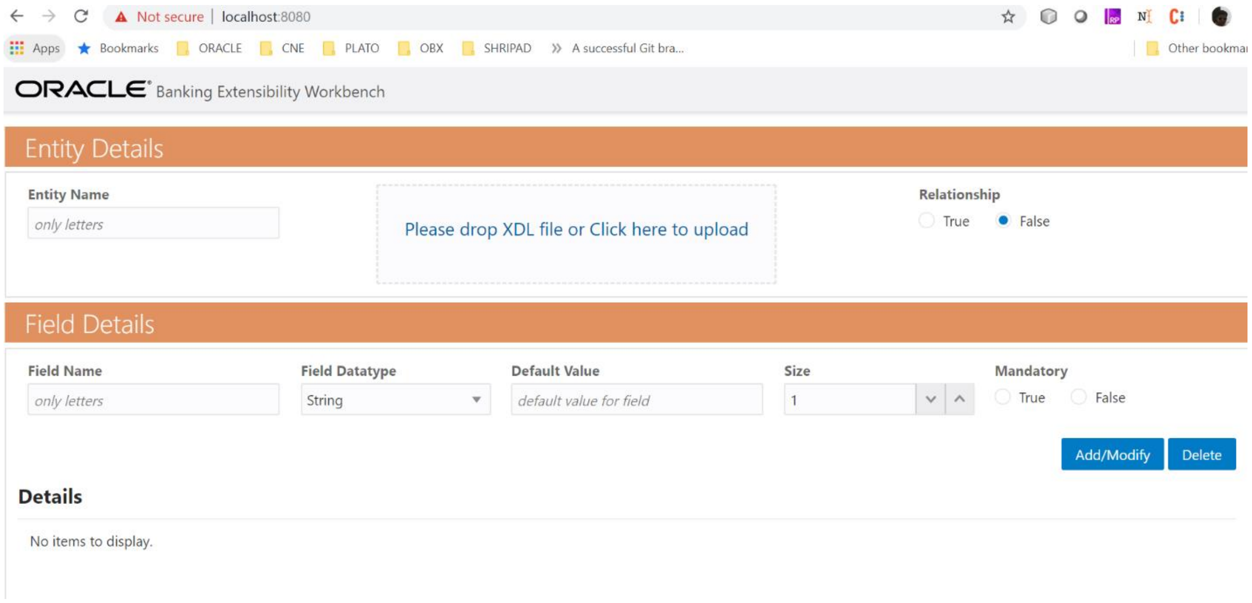
```
ORACLE BANKING EXTENSIBILITY WORKBENCH

Copyright © 2018 , 2019 , Oracle and/or its affiliates. All rights reserved.

 ________________________________________
|                                        |
| The information captured here is used  |
| to setup obx for further use like      |
| schema details                         |
|________________________________________|

? ----------------SECURITY MANAGEMENT INFORMATION------------
Provide the host name for the SMS schema: |
```

- Answer the following question with below answers without any trailing or leading spaces:
  - Provide the host name for the SMS schema: **host**
  - Provide the port for the SMS schema: **1521**
  - Provide the sid for the SMS schema if any (press enter if no sid needed): **<<BLANK>>**
  - Provide serviceName for the SMS schema if any (press enter if sid provided): **PDBs**
  - Provide the host name for the CMC schema: **host**
  - Provide the port for the CMC schema: **1521**
  - Provide the sid for the CMC schema if any (press enter if no sid needed): **<<BLANK>>**
  - Provide serviceName for the CMC schema if any (press enter if sid provided): **PDB**
  - Provide the host name for the ENTITY schema: **host**
  - Provide the port for the ENTITY schema: **1521**
  - Provide the sid for the ENTITY schema if any (press enter if no sid needed): **<<BLANK>>**
  - Provide serviceName for the ENTITY schema if any (press enter if sid provided): **PDB**

- Provide the host name for the UI_CONFIG schema: **host**
  - Provide the port for the UI_CONFIG schema: **1521**
  - Provide the sid for the UI_CONFIG schema if any (press enter if no sid needed): **<<BLANK>>**
  - Provide serviceName for the UI_CONFIG schema if any (press enter if sid provided): **PDB**
- Once all the answers are provided, it will create a file with name **db_props.json** inside extension_home folder, verify all the answers provided above

Simple domain service creation

- In this section we will be creating a simple domain micro-service
- To start creating the micro service we need to first generate the XDL (Extensibility Domain Language) file, which is understandable by the tool. For XDL file generation we have OBXUI to help us in generating the same
- From the **extension_home** folder execute the command '**obx xdl-gen**', this will start the OBX tool and we will be able to see the OBXUI up and running on https://localhost:8080



Entity Details

Entity Name: *only letters*

Please drop XDL file or Click here to upload

Relationship: True [✓ False]

Field Details

Field Name	Field Datatype	Default Value	Size	Mandatory
only letters	String	*default value for field*	1	True False

Add/Modify Delete

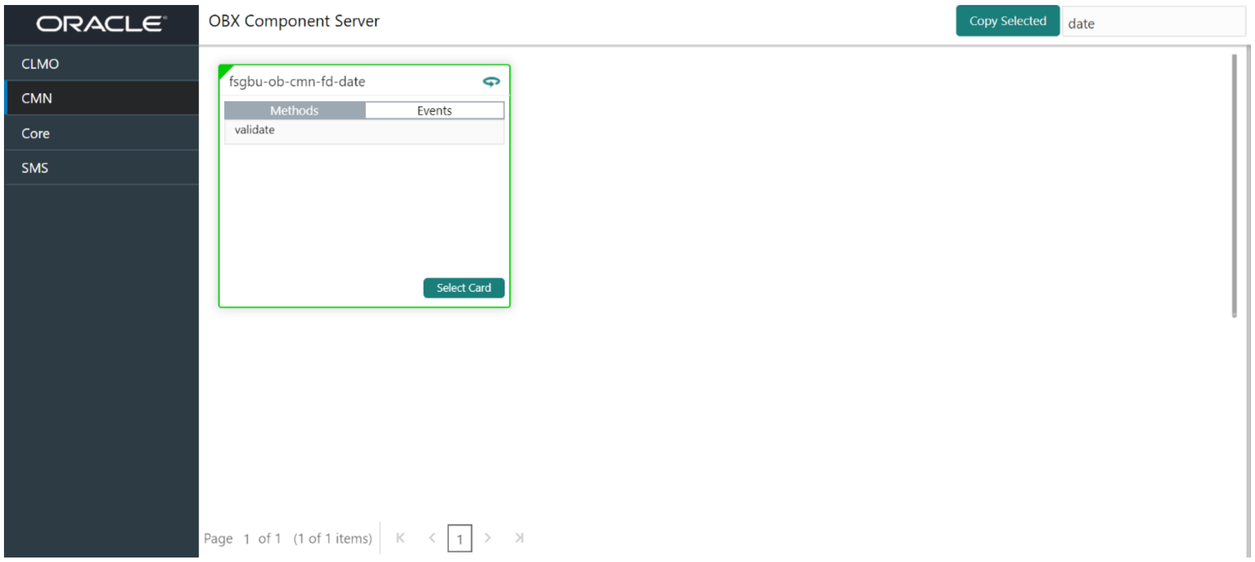
Details

No items to display.

- In that UI capture all information related to your entity along with the fields
- Once all the details are captured click on Save XDL button. This will allow you to save the XDL file on local machine. Please save the file in extension_home folder with proper name
- Now come back to cmder where it is waiting for your response for the Question **Did you generate the xdl file?** Mark it yes and proceed.
- It will prompt you question what do you want to create. Select the second option of **Domain Service with Optional UI component** and proceed
- This will take you to next section where it will prompt other set of questions related to the service generation. Answer them all and proceed
- After answering all the questions, it will generate the code for service and will prompt **Do you want to create UI component for this service?** You may select yes if you want to generate the UI the component or else if select no it will only generate the service

Simple UI(WC) creation

- Once selected yes to above section, here we will creating a web component, answer the below question as per your service:
  - Enter Virtual Page Name (I'll prepend obx-vp- to it): **<<component-name>>**
  - Enter the Service URL, from where Virtual Page will be communicating (I'll append /service_name/version/resource_name to it): **<<host:port>>**
- Once above questions are answered it will start generating the libraries followed by component server.
- Once all the internal processes are over, it will wait for user response to modify the metadata file. This metadata file contains the components list which user want to reuse using component server. For that we will first open component server which is running on http://localhost:8002



- Post saving the file come back to cmder and answer the question **Did you modify Metadata? If not please change it first before proceeding** as Yes
- Next it will ask question **Do you want to integrate the service and ui with App-shell?**
- If selected No, your component should be running locally on http://localhost:8001

Generating the Extended Component war

For generating the extended components war execute the following commands:

- Make you sure you have a folder created/cloned with structure extended-components/js/components in your extension_home folder
- Go to generated web component folder and execute command sh buildExtendedComponent.sh
- Come back to extension_home folder and run the command obx build-cca
- This will generate the extended-component.war in extension_home folder, which can be deployed on the application server

Integration of Web Component with Application Shell

In this step we are going to integrate the UI component generated for loan enquiry service with running application shell. To do that follow the below steps:

- ***Take the config_extendend.sql from component/db/ui-config and execute it in PLATO_UI_CONFIG schema*
- ***Take the scripts.sql from component/db/sms and execute it in SMS schema*
- Create New Role Code relevant to your service and web component and assign it to user

Maintenance Domain Service creation

- For generating the Maintenance type of domain service, we will again generate the xdl from extension_home folder
- Execute the command **'obx xdl-gen'** from extension_home folder and wait for the OBXUI to start
- Once started on https://localhost:8080 give all relevant information corresponding the service you wish to create
- Save the XDL file in the extension_home folder
- Now from cmder proceed by giving yes to **Did you generate the xdl file? (Y/n)**
- Select the option **Maintenance domain service with optional detail & summary UI components** for question **Do you want to create? Give Yes and proceed**
- This will take you to next section where it will prompt other set of questions related to the service generation. Answer them all and proceed
- After answering all the questions, it will generate the code for service and will prompt **Do you want to create a Maintenance and Summary Components for this service?** Give yes to it
- Once done we will start creating UI components for same

Detail and Summary WC creation

- Now it will start asking questions related to UI component. Answer them like below:
    - Enter the Service URL, from where Virtual Page will be communicating (I'll append /service_name/version/resource_name to it): **localhost:7001**
- Once above questions are answered it will start generating the libraries followed by component server
- Once all the internal processes are over, it will wait for user response to modify the metadata file. This metadata file contains the components list which user want to reuse using component server
- The above process will run twice as we have to create two components that is Detail Screen and Summary Screen
- On successful process completion we will have to components created inside extension_home folder one for summary and other for detail

Generating the Extended Component war

For generating the extended components war execute the following commands:

- Make you sure you have a folder created/cloned with structure **extended-components/js/components** in your extension_home folder
- Go to generated web component folder and execute command **sh buildExtendedComponent.sh**
- Come back to **extension_home** folder and run the command **obx build-cca**
- This will generate the **extended-component.war** in extension_home folder, deploy it on your application server like weblogic

Creation of Master Resource/Data Segment

In this section, we will be generating the data segment from scratch for the maintenance type of screens. This will involve generation of domain service and generating the UI component along with it. To start with we will first generate the domain service with UI using the OBX tool.

Domain Service with UI Component

- To generate the master domain service, navigate to extension_home folder from cmder
- Execute the command obx xdl-gen, this we will bring the OBX UI running on the https://localhost:8080
- Provide all the information related to your data segment and relationship as **false**
- Also capture all the fields details from the screen
- Save the xdl file in the extension_home folder by clicking on save xdl button
- Now come back to cmder and proceed by giving **Y** to question **Did you generate the xdl file?**
- Select the option of **Data-segment service with optional UI component** from the selection **Do you want to create: (Use arrow keys)**
- This will take you to next section where it will prompt other set of questions related to the service generation. Answer them all and proceed
- Once all the answers provided it will generate the code for the service and will ask the question **Do you want to create a Data Segment for this service?** Select Y and proceed
- Now it will prompt questions related to the UI component. Provide the following details:
    - Enter the Service URL, from where Virtual Page will be communicating (I'll append /service_name/version/resource_name to it): **localhost:7001**
- Once done it will generate the UI code, with Libraries and component server and will wait for the response **Did you modified Metadata? If not please change it first before proceeding** select Yes and proceed
- It will complete the process will prompt **Do you want to integrate the service and ui with App-shell?** Give **No** to it and your process will be completed with component and service created

Generating the Extended Component war for Data Segment

For generating the extended components war execute the following commands:

- Go to folder component folder and execute command **sh buildExtendedComponent.sh**
- Come back extension_home folder and run the command **obx build-cca**
- This will generate the extended-component war deploy it on application server like weblogic

Creation of Child Resource/Data Segment

Now we already have a master data segment created in the above section, in this part we will be creating a child data segment which will get attached to the existing to flow with master data segment. To do that we will follow the below process:

- To generate the child data segment domain service, navigate to extension_home folder from cmder
- Execute the command obx xdl-gen, this we will bring the OBX UI running on the https://localhost:8080
- Provide all the information related to your data segment and relationship as false
- Also capture all the fields details from the screen
- Save the xdl file in the extension_home folder by clicking on save xdl button
- Now come back to cmder and proceed by giving Y to question Did you generate the xdl file?
- Select the option of Data-segment service with optional UI component from the selection Do you want to create: (Use arrow keys)
- This will take you to next section where it will prompt other set of questions related to the service generation. Answer them all and proceed. Here select, the type of segment as Child
- Once all the answers provided it will generate the code for the service and will ask the question Do you want to create a Data Segment for this service? Select Y and proceed
- Now it will prompt questions related to the UI component. Provide the following details:
    - Enter the Service URL, from where Virtual Page will be communicating (I'll append /service_name/version/resource_name to it): localhost:7001
- Once done it will generate the UI code, with Libraries and component server and will wait for the response Did you modified Metadata? If not please change it first before proceeding select Yes and proceed
- It will complete the process will prompt Do you want to integrate the service and ui with App-shell? Give No to it and your process will be completed with components created in the folder along with the service.

Modification of Base Component

In this section we will generate an extended component from which we will modify the base component fields. To start the process, follow the below given steps:

Generating extended component

- Navigate to the extension_home folder and run the command obx ui –mb
- After running the command one prompt will appear on screen where you need to provide the name of the base component which you are going to modify.
- Enter Base Web component name which you want to modify (I'll append -extended to it): <<Base Component Name>>
- After answering the prompt, the extended Ui component will get generated.

Event Hub Integration

In this section we will generate an event service which can be a consumer/producer or both. Please follow the steps below to generate the event service.

- Event Service Generation
- Navigate to the extension_home folder using cmder.
- Now create a folder with the name of service you wish to create
- Change your directory to newly created
- To generate the service code run the command obx event -c
- Once the command is fired, the OBX tool will prompt some questions related to the event service. Please provide the following details for the prompts:
    - Enter name of service: **<<name of service>>**
    - Enter the hostname for kafka server: localhost (here you can provide the host where kafka server is running)
- Enter the port for kafka server: Port
- Enter the hostname for zookeeper server: localhost
- Enter the port for zookeeper server: port
- Enter number of events: 2
- Please Select the Type of event/stream you wish to create: publisher
- Enter the name of event/stream 1: <<Name of Publisher Event/Stream>>
- Enter topic name for the selected event/stream: <<Topic Name>>
- Enter avro schema name for the selected event: <<Avro Schema Name >>
- Please Select the Type of event/stream you wish to create: subscriber
- Enter the name of event/stream 2: <<Name of Subscriber Event/Stream>>
- Enter topic name for the selected event/stream: <<Topic Name>>
- Enter avro schema name for the selected event: <<Avro Schema Name >>
- Once all the answers are provided the event service with one subscriber and one publisher will gets generated.

Batch Hub Integration

In this section we will generate a batch service. We won't be using any xdl file for generating the batch file because it generates a standard code for batches. You can modify the code based on your requirements. Please follow the steps below to generate the event service-

- Navigate to the extension_home folder using cmder.
- Now create a folder for batch service and navigate into it
- To generate the service code run the command obx task –c
- After firing the command 2 questions are prompted. Provide the answers for those prompts as given below-
  - Enter the product family: <<Product name>>
  - Enter the task name: <<Task Name>>
- After answering these two prompts your service will get generated.
- Now run the command gradlew clean build –x test to build the service.