

Oracle® Banking Digital Experience Cloud Service User Interface Guide



Release 25.1.1.0.0
G46579-01
October 2025

ORACLE®

Copyright © 2015, 2025, Oracle and/or its affiliates.

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

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

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

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

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

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

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

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

Contents

Preface

Purpose	i
Audience	i
Documentation Accessibility	i
Diversity and Inclusion	i
Conventions	ii
Related Resources	ii
Screenshot Disclaimer	ii
Acronyms and Abbreviations	ii

1 Pre-requisite

2 User Interface Build

3 UI deployment

4 Configuration to run UI on Oracle HTTP Server

5 Oracle HTTP Server Commands

5.1 Starting Oracle HTTP Server Instances from the Command Line	1
5.2 Stopping Oracle HTTP Server Instances from the Command Line	1

6 Configuring User Interface

7 Language Pack

7.1 Adding New Language	1
Column Explanation	1

	Rules for Locale Flags (No Manual DB changes is expecting from the user)	2
	Best Practices for Managing Locales (Always Use Locale Maintenance Screen)	2
7.2	Deployment of the language pack	2

8 Configuring Different URL's on the Basis of Enterprise Roles

Index

Preface

- [Purpose](#)
- [Audience](#)
- [Documentation Accessibility](#)
- [Diversity and Inclusion](#)
- [Conventions](#)
- [Related Resources](#)
- [Screenshot Disclaimer](#)
- [Acronyms and Abbreviations](#)

Purpose

This guide is designed to help acquaint you with the Oracle Banking application. This guide provides answers to specific features and procedures that the user need to be aware of the module to function successfully.

Audience

This document is intended for the following audience:

- Customers
- Partners

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 customer access to and use of Oracle support services will be pursuant to the terms and conditions specified in their Oracle order for the applicable services.

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.

Conventions

The following text conventions are used in this document:

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

Related Resources

For more information on any related features, refer to the following documents:

- Oracle Banking Digital Experience Installation Manuals
- Oracle Banking Digital Experience Licensing Manuals

Screenshot Disclaimer

Personal information used in the interface or documents is dummy and does not exist in the real world. It is only for reference purposes; actual screens that appear in the application may vary based on selected browser, theme, and mobile devices.

Acronyms and Abbreviations

The list of the acronyms and abbreviations used in this guide are as follows:

Table 1 Acronyms and Abbreviations

Abbreviation	Description
OBDX	Oracle Banking Digital Experience

1

Pre-requisite

This topic provides information on **Pre-requisite**.

OHS software along with instance should be available for use.

For further detailed configuration of Oracle HTTP Server, please refer to [**https://docs.oracle.com/middleware/12213/webtier/administer-ohs/toc.htm**](https://docs.oracle.com/middleware/12213/webtier/administer-ohs/toc.htm)

First make sure that NodeJS is installed on the machine.

2

User Interface Build

This topic provides information on **User Interface Build**.

The current GUI build is based on Webpack.

Webpack is a free, open-source JavaScript module bundler. It can also be used with HTML and CSS. Webpack is primarily used for JavaScript, but it can also transform front-end assets like HTML, CSS, and images.

The tasks performed during a typical GUI build are:

- Toolkit Component generation from metadata
- Pre Build checks (For some development rules)
- ESLint for the JS files.
- SCSS compilation to CSS
- CSS optimization
- HTML validation
- JS minification and bundling.

App Shell Lib dependencies:

The UI patch relies on two modules: app-shell and cmc-component-server. These components are necessary to load OBRH (Routing Hub) in the OBDX UI for admin. Once installed, additional access control can be managed through Role Maintenance.

The patch will have Appshelllib.zip.

Unzip that at channel level.

Running UI Build:

Follow steps below to run UI Build:

Ensure that the Swagger documentation (JSON) is hosted and accessible on a server.

In the channel directory, locate the swagger/mapping.json file and change all occurrences of the Swagger documents to the URL of the locally hosted server.

Place obdx-ui-workbench-code-generator, obdx-ui-workbench-core as sibling directory to channel, thus making all three directories in the same level.

Open a terminal inside obdx-ui-workbench-core directory and run following commands.

- *npm install.*
- *npm run-script build.*
 - In obdx-ui-workbench-code-generator directory open the package.json file and remove dependency of @obdx/uiworkbench-core.
 - Inside obdx-ui-workbench-code-generator directory run the following commands.
- *npm install.*

- `npm link ../obdx-ui-workbench-core`.

In the channel/package.json file, remove @obdx/obdx-ui-workbench-code-generator, @obdx/uiworkbench-core, @obdx/app-shell, and @obdx/cmc-component-server.

Initialize all the Node package dependencies by running the following command at the channel level:

`npm install` or `npm i`

`npm link ../obdx-ui-workbench-core ../obdx-ui-workbench-code-generator ../app-shell ../cmc-component-server`.

For Build run following command.

`npm run build`

It runs all the required commands for build and output is stored in dist folder.

The other commands are available for build if user wants to run individual commands

`npm run start`

It is used in development workspace for developer. It builds all the resources and opens a dev server for the development.

`npm run codegen`

It generates delta component from last build from toolkit manifest.

`npm run codegen-all`

It generates all the components from toolkit manifest.

`npm run webpack-build`

Run webpack build in production mode.

`npm run webpack-dev`

Run webpack build in development mode.

`npm run lint`

Run all the lint tasks such as eslint, html-validate and pre build checks

`npm run eslint`

Run the eslint task for manual components.

`npm run eslint-toolkit`

Run eslint task for toolkit components

`npm run html-validate`

Run HTML validate task.

npm run widget-manifest-gen

Generates widget manifest from all widgets component.

Webpack configurations are maintained under following files:

- `scripts/webpack/webpack.common.js`

All the common webpack configurations applicable in all the build.

- `scripts/webpack/webpack.prod.js`

Webpack Configuration applicable for production build.

`scripts/webpack/webpack.dev.js`

Webpack Configuration applicable for development build.

For detail webpack configuration please refer: <https://webpack.js.org/concepts/>

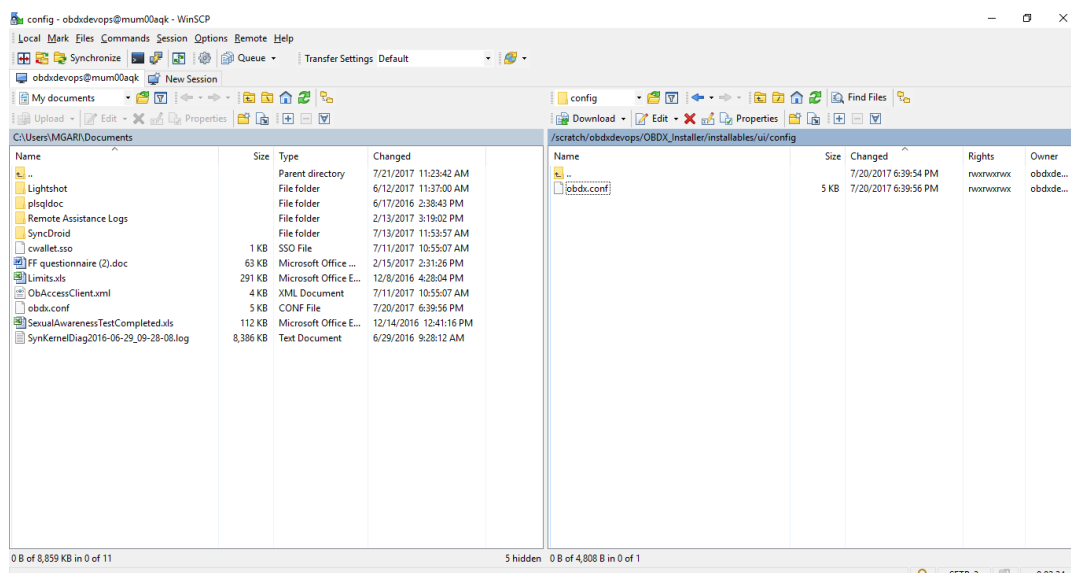
3

UI deployment

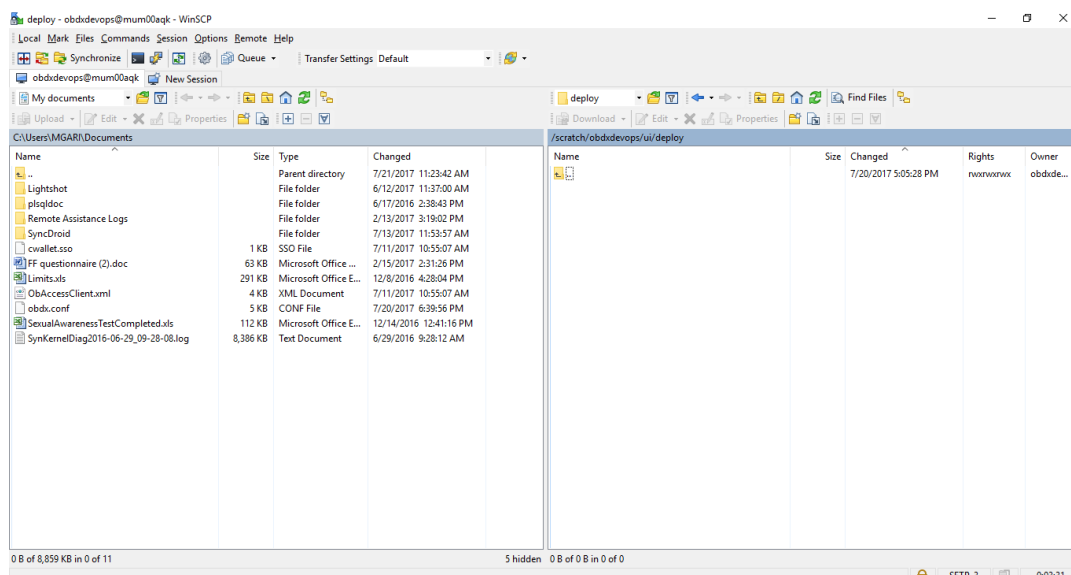
This topic describes the systematic instruction to **UI deployment** option.

Below steps needs to be performed for UI deployment on OHS server.

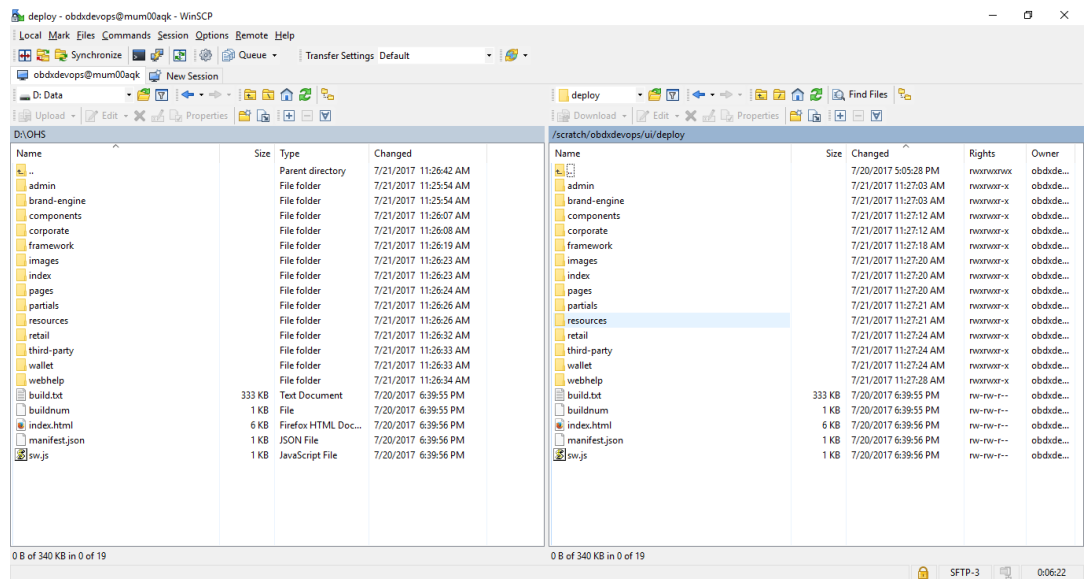
- Copy the obdx.conf from OBDX_Installer/installables/ui/config directory into the instance config directory (where httpd.conf is present). httpd.conf file is present at {DOMAIN_HOME}/config/fmwconfig/components/OHS/{componentName}



- Create a directory where obdx UI files would be deployed on OHS server.



- Copy all files / directories from OBDX_Installer/installables/ui/deploy into newly created directory.



4

Configuration to run UI on Oracle HTTP Server

This topic provides information on **Configuration to run UI on Oracle HTTP Server**.

Make sure following OHS modules must be loaded

- mod_rewrite.so
- mod_deflate.so
- mod_expires.so
- mod_mime.so
- mod_headers.so

Following are the changes needed to be done in the obdx.conf file and place this file in same folder where httpd.conf file exists.

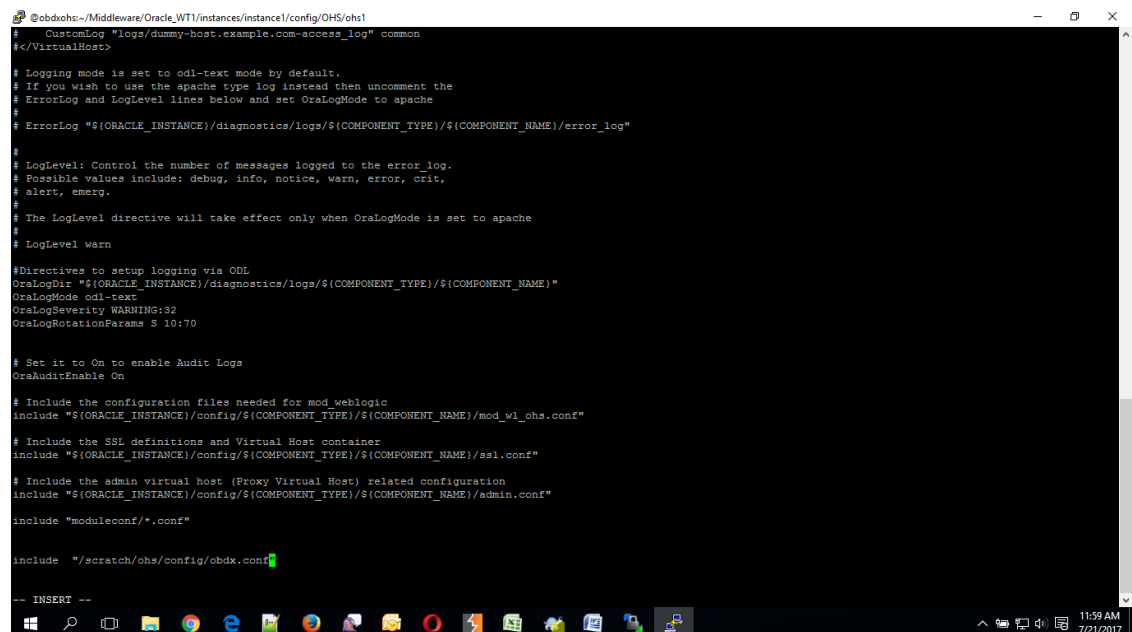
1. Replace the <CHANNEL_PATH> (all occurrences) with the newly created directory (from previous UI deployment step).
2. Configuration for Content Security Policy, refer to the below document

Oracle Banking Digital Experience Security Guide

Include the obdx.conf into httpd.conf using below configuration

include "obdx.conf" (needs to be added in httpd.conf)

Read obdx.conf for inline documentation.



```
@obdxohs-~/Middleware/Oracle_WT1/instances/instance1/config/OHS/ohs1
# CustomLog "logs/dummy-host.example.com-access_log" common
#</VirtualHost>

# Logging mode is set to odl-text mode by default.
# If you wish to use the apache type log instead then uncomment the
# ErrorLog and LogLevel lines below and set OraLogMode to apache
#
# ErrorLog "${ORACLE_INSTANCE}/diagnostics/logs/${COMPONENT_TYPE}/${COMPONENT_NAME}/error_log"
#
#
# LogLevel: Control the number of messages logged to the error_log.
# Possible values include: debug, info, notice, warn, error, crit,
# alert, emerg.
#
# The LogLevel directive will take effect only when OraLogMode is set to apache
#
# LogLevel warn

#Directives to setup logging via ODL
OraLogDir "${ORACLE_INSTANCE}/diagnostics/logs/${COMPONENT_TYPE}/${COMPONENT_NAME}"
OraLogMode odl-text
OraLogSeverity WARNING:32
OraLogRotationParams S 10:70

# Set it to On to enable Audit Logs
OraAuditEnable On

# Include the configuration files needed for mod weblogic
include "${ORACLE_INSTANCE}/config/${COMPONENT_TYPE}/${COMPONENT_NAME}/mod_wl_ohs.conf"

# Include the SSL definitions and Virtual Host container
include "${ORACLE_INSTANCE}/config/${COMPONENT_TYPE}/${COMPONENT_NAME}/ssl.conf"

# Include the admin virtual host (Proxy Virtual Host) related configuration
include "${ORACLE_INSTANCE}/config/${COMPONENT_TYPE}/${COMPONENT_NAME}/admin.conf"

include "moduleconf/*.conf"

include "/scratch/ohs/config/obdx.conf"

-- INSERT --
```

Following are the changes need to be done in mod_wl_ohs.conf which is present at {DOMAIN_HOME}/config/fmwconfig/components/OHS/{componentName}

Copy below configuration into mod_wl_ohs.conf

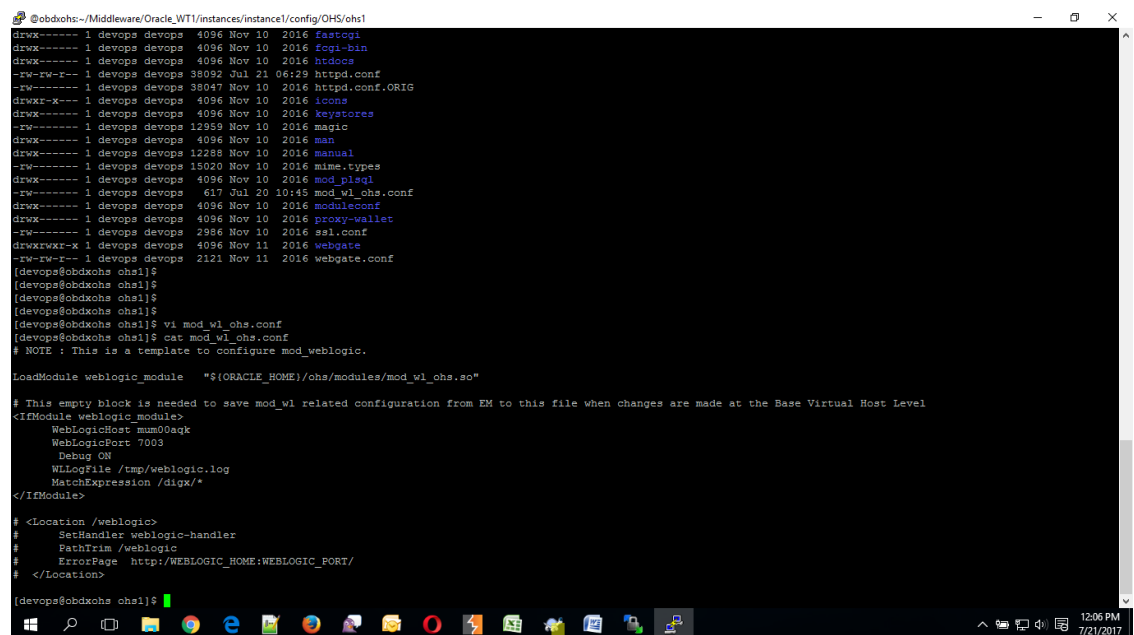
```
<IfModule weblogic_module>
WebLogicHost HOSTNAME
WebLogicPort MANAGE_SERVER_PORT
Debug ON
WLLogFile DIR/FILENAME
MatchExpression /digx*
</IfModule>
```

Configure below properties

1. HOSTNAME – Weblogic server hostname (where OBDX weblogic domain is configured)
2. MANAGE_SERVER_PORT – Weblogic manage server port (where OBDX application is deployed)
3. DIR / FILENAME – Path where log file should be generated

Sample configuration (for reference purpose only)

```
<IfModule weblogic_module>
WebLogicHost wls_server1
WebLogicPort 7003
Debug ON
WLLogFile/tmp/weblogic_obp.log
MatchExpression/digx/*
</IfModule>
```



```
@obdxohs:~/Middleware/Oracle_WT1/instances/instance1/config/OHS/ohs1
drwx----- 1 devops devops 4096 Nov 10 2016 fastcgi
drwx----- 1 devops devops 4096 Nov 10 2016 fcgi-bin
drwx----- 1 devops devops 4096 Nov 10 2016 httdocs
-rw-rw-r-- 1 devops devops 38092 Jul 21 06:29 httpd.conf
-rw----- 1 devops devops 38047 Nov 10 2016 httpd.conf.ORIG
drwxr-x--- 1 devops devops 4096 Nov 10 2016 icons
drwx----- 1 devops devops 4096 Nov 10 2016 keystores
-rw----- 1 devops devops 12989 Nov 10 2016 magic
drwx----- 1 devops devops 4096 Nov 10 2016 man
drwx----- 1 devops devops 12288 Nov 10 2016 manual
-rw----- 1 devops devops 15020 Nov 10 2016 mime.types
drwx----- 1 devops devops 4096 Nov 10 2016 mod_plsql
-rw----- 1 devops devops 617 Jul 20 10:45 mod_wl_ohs.conf
drwx----- 1 devops devops 4096 Nov 10 2016 moduleconf
drwx----- 1 devops devops 4096 Nov 10 2016 proxy-wallet
-rw----- 1 devops devops 2986 Nov 10 2016 ssl.conf
drwxrwxr-x 1 devops devops 4096 Nov 11 2016 webgate
-rw-rw-r-- 1 devops devops 2121 Nov 11 2016 webgate.conf
(devops@obdxohs ohs1)$
(devops@obdxohs ohs1)$
(devops@obdxohs ohs1)$
LoadModule weblogic_module "${ORACLE_HOME}/ohs/modules/mod_wl_ohs.so"
# This empty block is needed to save mod_wl related configuration from EM to this file when changes are made at the Base Virtual Host Level
<IfModule weblogic_module>
WebLogicHost mm00agk
WebLogicPort 7003
Debug ON
WLLogFile /tmp/weblogic.log
MatchExpression /digx/*
</IfModule>
# <Location /weblogic>
#   SetHandler weblogic-handler
#   PathTrim /weblogic
#   ErrorPage http://WEBLOGIC_HOME:WEBLOGIC_PORT/
# </Location>
(devops@obdxohs ohs1)$
```

5

Oracle HTTP Server Commands

- [Starting Oracle HTTP Server Instances from the Command Line](#)
This topic provides information on **Starting Oracle HTTP Server Instances from the Command Line**
- [Stopping Oracle HTTP Server Instances from the Command Line](#)
This topic provides information on **Stopping Oracle HTTP Server Instances from the Command Line**.

5.1 Starting Oracle HTTP Server Instances from the Command Line

This topic provides information on **Starting Oracle HTTP Server Instances from the Command Line**

You can start up Oracle HTTP Server instances from the command line via a script.

1. Ensure that Node Manager is running.
2. Enter the following command:

Linux or UNIX: `$DOMAIN_HOME/bin/startComponent.sh componentName`
Windows: `%DOMAIN_HOME%\bin\startComponent.cmd componentName`

For example:

```
$DOMAIN_HOME/bin/startComponent.sh ohsl
```

The startComponent script contacts the Node Manager and runs the nmStart() command.

When prompted, enter your Node Manager password. The system responds with these messages:

```
Successfully started server componentName...  
Successfully disconnected from Node Manager...  
Exiting WebLogic Scripting Tool.
```

5.2 Stopping Oracle HTTP Server Instances from the Command Line

This topic provides information on **Stopping Oracle HTTP Server Instances from the Command Line**.

You can stop Oracle HTTP Server instances from the command line via a script.

Enter the following command:

```
Linux or UNIX: $DOMAIN_HOME/bin/stopComponent.sh componentName  
Windows: %DOMAIN_HOME%\bin\stopComponent.cmd componentName
```

For example:

```
$DOMAIN_HOME/bin/stopComponent.sh ohs1
```

This command invokes WLST and executes the nmKill() command. The stopComponent command will not function if the Node Manager is not running.

For more commands refer the following

<https://docs.oracle.com/middleware/1221/webtier/administer-ohs/getstart.htm>

6

Configuring User Interface

This topic describes the systematic instruction to **Configuring User Interface** option.

All the UI configurations are available in config.js while which is present under the <CHANNEL_PATH>\framework\js\configurations directory. JavaScript object for the configuration is declare by the name "configuration". Application freeze this object so its value cannot be change in running memory.

Category of the configuration:

i18n: All the internalization specific configuration mentioned in this. Currently this category have list of rtl locales

```
i18n: {  
    rtlLocales: ["ar", "he", "ku", "fa", "ur", "dv", "ha", "ps", "yi"]  
}
```

Sharding: Domain sharding is a technique used to increase the amount of simultaneously downloaded resources for a particular website by using multiple domains. This allows websites to be delivered **faster** to users as they do not have to wait for the previous set of resources to be downloaded before beginning the next set. Implementer can introduce 3 additional domains for the UI

1. **apiBaseURL:** If the HTTP server and the application server are on same host, the property is set as "" otherwise set to host name and port of the application server.
imageResourcePath: The base path from which the image resources are to be fetched. It can also be a relative path pointing to the same domain the page is running on or a fully qualified path to different server on which images are hosted

```
sharding:  
{  
    apiBaseURL: ""  
}
```

Authentication: OBDX product ships with two type of authentication methods:

1. OAM Authentication
2. Non OAM Authentication (OBDXAuthenticator)
3. JWT Authenticator (JWTAuthenticator)

Configuring OAM Authentication set type as OAM and also provide the provider URL of OAM in providerURL property.

For Non OAM set type as OBDXAuthenticator or JWTAuthenticator based on requirement.

In the application, setting secure and public page is required. For this two properties are exposed as `pages.securePage` and `pages.publicPage`. As name suggest `pages.securePage` have the pathname of secure page and `pages.publicPage` have the pathname of public/unsecure page.

```
authentication: {
  type: "OBDXAuthenticator",
  providerURL: "",
  pages: {
    securePage: "home.html",
    publicPage: "index.html"
  }
}
```

Third Party API's: Some of the application module required integration with third party provider like facebook, linkedin, google etc. So in this category we maintained all the sdk url, api keys and provider url of third party api's

```
thirdPartyAPIs: {
  facebook: {
    url: "",
    sdkURL: "",
    apiKey: ""
  },
  linkedin: {
    sdkURL: "",
    apiKey: ""
  },
  googleMap: {
    url: "",
    sdkURL: "",
    apiKey: ""
  }
}
```

API Catalogue: This category used for several context root available in OBDX API's and their default versions. This is maintained at `<CHANNEL_PATH>/framework/js/api-catalogue`

```

apiCatalogue: {
  base: {
    contextRoot: "digx",
    defaultVersion: "v1"
  },
  extended: {
    contextRoot: "digx/ext",
    defaultVersion: "v1"
  },
  social: {
    contextRoot: "digx-social",
    defaultVersion: "v1"
  },
  "digx-auth": {
    contextRoot: "digx-auth/ext",
    defaultVersion: "v1"
  },
  "digx-auth-extended": {
    contextRoot: "digx-auth",
    defaultVersion: "v1"
  }
}

```

System Configuration: This category of configuration is used for system level properties. Brief description of properties are below:

componentAccessControlEnabled: Component access check(through role transaction mapping) is enabled or not. Depending of this property menu or link will filtered.

requestThrottleSeconds: OBDX UI can cached service responses and it also distribute one API response to several caller. For example if 3 widgets calling same API, in this case application fire only one API and distribute its response to all the callers. requestThrottleSeconds property used for caching time of the response. Unit is in second. It means if you set requestThrottleSeconds as 5(second) it means if application fire same API within 5 second application return the same response which it fire earlier.

defaultEntity: Default entity if entity cannot be derived.

sslEnabled: SSL is enabled or not.

loggingLevel: Logging level of OBDX UI.

```

system:
{
  componentAccessControlEnabled: true,
  requestThrottleSeconds: 5,
  defaultEntity: "",
  sslEnabled: true,
  loggingLevel: "LEVEL_ERROR",
}

```

Development Configuration: This category of configuration is used during development phase. In this category we also have property for enabling accessibility checks during run time.

```
development:
{
checkAccessibility: false,
axeUrl: "https://cdnjs.cloudflare.com/ajax/libs/axe-core/3.3.2/
axe.min.js"
}
```

Domain Deployment: This flag is set enable true or false based on services deployment strategy.

Overriding Configurations:

If User wants to override any configuration available in config.js. They can do by putting all the modified properties in scripts/webpack/.obdx-config-override.json.

Please make sure any properties maintained here will be add and updated in original config.js

7

Language Pack

This topic provides information on **Language Pack**.

OBDX is pre-configured with the following locale:

('en', 'ar', 'es', 'pt', 'fr', 'zh_CN', 'zh_TW')

Note

By default, only English is enabled and set as the default language. The other languages are disabled. You can enable other languages if the scripts for those languages, provided by the translation pack, are installed on your system.

- [Adding New Language](#)
This topic provides information on **Adding New Language**.
- [Deployment of the language pack](#)
This topic provides information on **Deployment of the language pack**.

7.1 Adding New Language

This topic provides information on **Adding New Language**.

To add a new language other than the ones provided, follow these steps:

1. **Determine the locale code:** Identify the locale code for the new language you want to add. For example, 'de' for German.
2. **Insert a new row into the `DIGX_FW_LOCALE` table:** Use the following SQL statement to insert a new row into the `DIGX_FW_LOCALE` table:

```
```sql INSERT INTO DIGX_FW_LOCALE (CODE, DESCRIPTION, IS_ENABLED, VALUE,  
IS_INSTALLED, IS_DEFAULT) VALUES ('de', 'Deutsch', 'N', 'de', 'N', 'N'); ```
```

### Note

Replace 'de' with the locale code for the new language and 'Deutsch' with the description of the language.

## Column Explanation

1. **CODE:** The code for the locale.
2. **DESCRIPTION:** The description of the locale.
3. **IS\_ENABLED:** Flag indicating whether the locale is enabled ('Y' or 'N'). A locale can be enabled only if `IS\_INSTALLED` is 'Y'.
4. **VALUE:** The value of the locale.

5. **IS\_INSTALLED**: Flag indicating whether the locale is installed ('Y' or 'N'). This flag is automatically set by the Translation Drop generator job for the locale which are supported by OBDX using translation pack.
6. **IS\_DEFAULT**: Flag indicating whether the locale is the default ('Y' or 'N'). English is the default language. (Note: This is the default fallback locale system will use in case the locale code from the request is not maintained in the database)

## Rules for Locale Flags (No Manual DB changes is expecting from the user)

The following rules apply to the locale flags:

1. **IS\_INSTALLED** is automatically set by the Translation Drop generator job. It is not recommended to manually set this flag to 'Y' unless you are certain that the necessary SQL scripts have been executed properly.
2. To enable a locale, set **IS\_ENABLED** to 'Y', but only if **IS\_INSTALLED** is 'Y'.
3. If **IS\_ENABLED** is 'Y' without **IS\_INSTALLED** being 'Y', the locale will not be displayed on the screen.
4. To set a locale as default, both **IS\_ENABLED** and **IS\_INSTALLED** must be 'Y'.
5. At least one locale should be enabled, and one should be set as default.

## Best Practices for Managing Locales (Always Use Locale Maintenance Screen)

1. Allow the Translation Drop generator job to automatically manage the **IS\_INSTALLED** flag.
2. Locales will be available for use only after they have been installed (**IS\_INSTALLED** = 'Y'), and this should be done using the Locale Maintenance Screen.
3. At least one locale is enabled, and one should be set as default. (System Constraints already enforced on the screen).

## 7.2 Deployment of the language pack

This topic provides information on **Deployment of the language pack**.

Language pack can be classified in the following types

### Database Scripts:

1. Login to OBDX Schema
2. Execute following SQL files :

```
OBDX_<VERSION>_TRANSLATION_PACK\<LOCALE>\seed\digx_fw_error_messages.sql ,
OBDX_<VERSION>_TRANSLATION_PACK\<LOCALE>\seed\digx_fw_info_messages.sql |
```

3. Commit the changes

```
commit;
```

**Weblogic Configuration:**

1. Copy all files/ directories from  
OBDX\_<VERSION>\_TRANSLATION\_PACK\<LOCALE>\config to \${OBDX\_HOME}\config  
hosted on Weblogic Server

**Note**

The path for \${OBDX\_HOME}\config can be found under Managed Server classpath which is accessible via Weblogic Administration.

**UI Configuration:**

1. Copy complete  
OBDX\_<VERSION>\_TRANSLATION\_PACK\<LOCALE>\channel\resources\nls\<LOCALE>  
directory to <CHANNEL\_PATH>/resources/nls/
2. Create a new <LOCALE> directory in <CHANNEL\_PATH>/partials/help
3. Copy all existing files from <CHANNEL\_PATH>/partials/help to <CHANNEL\_PATH>/partials/  
help/<LOCALE>
4. Override all help files from  
OBDX\_<VERSION>\_TRANSLATION\_PACK\<LOCALE>\channel\partials\help\<LOCALE> to  
<CHANNEL\_PATH>/partials/help/<LOCALE>
5. Validate required language entry is maintained at <CHANNEL\_PATH>scripts/webpack/  
loaders/resource-bundle-loader.js in supportedLang variable.
6. Run the GUI Build.

**Configure below properties:**

1. Open file digx-shared-libs/WEB-INF/classes/Preferences.xml
2. Add following lines

```
<Preference name="ErrorMessagesConfig_<LOCALE>"
PreferencesProvider="com.ofss.digx.infra.config.impl.DBBasedPropertyProvide
r"
parent="jdbcpreference" propertyFileName="select ERROR_CODE,
ERROR_MESSAGE from DIGX_FW_ERROR_MESSAGES where USER_LOCALE = '<LOCALE>' "
syncTimeInterval="0" />
```

```
<Preference name="InfoMessagesConfig_<LOCALE>"
PreferencesProvider="com.ofss.digx.infra.config.impl.DBBasedPropertyProvide
r"
parent="jdbcpreference" propertyFileName="select INFO_CODE,
INFO_MESSAGE from DIGX_FW_INFO_MESSAGES where USER_LOCALE = '<LOCALE>' "
syncTimeInterval="0" />
```

# 8

## Configuring Different URL's on the Basis of Enterprise Roles

This topic provides information on **Configuring Different URL's on the Basis of Enterprise Roles**.

To enable URL separation based on enterprise roles using custom header name and value, the following queries needs to be executed in **DIGX\_FW\_CONFIG\_ALL\_B** table

```
Insert into DIGX_FW_CONFIG_ALL_B
(PROP_ID, CATEGORY_ID, PROP_VALUE,FACTORY_SHIPPED_FLAG,
PROP_COMMENTS, SUMMARY_TEXT, CREATED_BY, CREATION_DATE, LAST_UPDATED_BY,
LAST_UPDATED_DATE, OBJECT_STATUS, OBJECT_VERSION_NUMBER)

values ('IS_LOGIN_SEPARATION_ENABLED','SecurityConstants','true','N',null,
'Is login separation enabled','ofssuser',sysdate,'ofssuser',sysdate,'Y',1);
```

This query enables the URL separation mechanism. By default the URL separation mechanism is not enabled.

```
Insert into DIGX_FW_CONFIG_ALL_B (PROP_ID, CATEGORY_ID, PROP_VALUE,
FACTORY_SHIPPED_FLAG, PROP_COMMENTS, SUMMARY_TEXT, CREATED_BY, CREATION_DATE,
LAST_UPDATED_BY, LAST_UPDATED_DATE, OBJECT_STATUS, OBJECT_VERSION_NUMBER)

values ('LOGIN_HEADER_NAME','SecurityConstants',<HEADER_NAME>,'Y',null,
'Header name for login
separation','ofssuser',sysdate,'ofssuser',sysdate,'Y',1);
```

This query is used to provide entry for the custom header name.

```
Insert into DIGX_FW_CONFIG_ALL_B (PROP_ID, CATEGORY_ID, PROP_VALUE,
FACTORY_SHIPPED_FLAG, PROP_COMMENTS, SUMMARY_TEXT, CREATED_BY, CREATION_DATE,
LAST_UPDATED_BY, LAST_UPDATED_DATE, OBJECT_STATUS, OBJECT_VERSION_NUMBER)

values (<HEADER_NAME>,'SecurityConstants',<HEADER_VALUE>,'Y',null,'login
separation header name
and value pair','ofssuser',sysdate,'ofssuser',sysdate,'Y',1);
```

This query is used for mapping the custom header name with its corresponding value.

```
Insert into DIGX_FW_CONFIG_ALL_B (PROP_ID, CATEGORY_ID, PROP_VALUE,
FACTORY_SHIPPED_FLAG, PROP_COMMENTS, SUMMARY_TEXT, CREATED_BY, CREATION_DATE,
LAST_UPDATED_BY, LAST_UPDATED_DATE, OBJECT_STATUS, OBJECT_VERSION_NUMBER)
```



```
values (<HEADER_VALUE>,'SecurityConstants',<ENTERPRISE_ROLE>,'Y',null,
'Enables login separation for given enterprise
role','ofssuser',sysdate,'ofssuser',sysdate,'Y',1);
```

This query is used for mapping the custom header value with the enterprise role for which the URL separation has to be achieved.

In the above queries, <HEADER\_NAME> field denotes the custom header name, <HEADER\_VALUE> denotes the custom header value, and <ENTERPRISE\_ROLE> field denotes the enterprise role. These fields need to be replaced with own custom values before executing the queries.

### OHS Configuration:

To support it OHS needs to send an additional header to Weblogic server. To enable this implementer needs to configure a new port and create a virtual host where that custom header is added in the request.

Sample snippet is below

```
Listen PORT_NO<VirtualHost *:PORT_NO >
RequestHeader add <HEADER_NAME> "<HEADER_VALUE> "
<Location /digx>
SetHandler weblogic-handler
WebLogicCluster WEBLOGIC_HOST:WEBLOGIC_PORT
</Location>
</VirtualHost>
```

# Index

## A

---

Adding New Language, [1](#)

## C

---

Configuration to run UI on Oracle HTTP Server, [1](#)

Configuring Different URL's on the Basis of  
Enterprise Roles, [1](#)

Configuring User Interface, [1](#)

## D

---

Deployment of the language pack, [2](#)

## L

---

Language Pack, [1](#)

## P

---

Pre-requisite, [1](#)

## S

---

Starting Oracle HTTP Server Instances from the  
Command Line, [1](#)

Stopping Oracle HTTP Server Instances from the  
Command Line, [1](#)

## U

---

UI deployment, [1](#)

User Interface Build, [1](#)