Development Workbench - Screen Development II
Oracle FLEXCUBE Universal Banking
Release 14.8.3.0.0
Part No. - G61377-01
October 2026

Contents

1 Preface 3

1.1 Audience 3

1.2 Related Documents 3

2 Introduction 4

3 Generated Files 4

3.1 Front-End Files 5

3.2 System Packages 7

3.3 Hook Packages 7

3.4 Meta Data 8

3.5 Others 10

4 Deployment 10

4.1 Prerequisites: 10

4.2 Deploy Files: 12

5 Release 13

5.1 SVN Integration 13

1 Preface

This document describes the generation, deployment and release of units from Oracle FLEXCUBE Development Workbench for Universal Banking. This document explains the process of generating and deploying files from Development Workbench for a function id

1.1 Audience

This document is intended for FLEXCUBE Application developers/users that use Development Workbench to develop various FLEXCUBE components.

To Use this manual, you need conceptual and working knowledge of the below:

Proficiency	*Resources*
FLEXCUBE Functional Architecture	Training programs from Oracle Financial Software Services.
FLEXCUBE Technical Architecture	Training programs from Oracle Financial Software Services.
FLEXCUBE Object Naming conventions	Development Overview Guide
Working knowledge of Web based applications	Self Acquired
Working knowledge of Oracle Database	Oracle Documentations
Working knowledge of PLSQL & SQL Language	Self Acquired
Working knowledge of XML files	Self Acquired
Working Knowledge of Development Workbench for Universal Banking	User Manuals

1.2 Related Documents

Oracle FLEXCUBE Enterprise Limits and Collateral Management ODT Screen Development

Development Workbench – Notifications

2 Introduction

This document gives information on the following topics:

- Chapter 2 , “Introduction”
- Chapter 3 , “Generated Files"
- Chapter 4 , “Deployment”
- Chapter 5 , "Release”

3 Generated Files

Development Workbench generates following files as shown in the figure below. This screen can be launched by clicking on  icon from the function generation screen . This has to be done after design of screen is completed. Refer *Oracle FLEXCUBE Enterprise Limits and Collateral Management ODT Screen Development* for detailed explanation on screen development

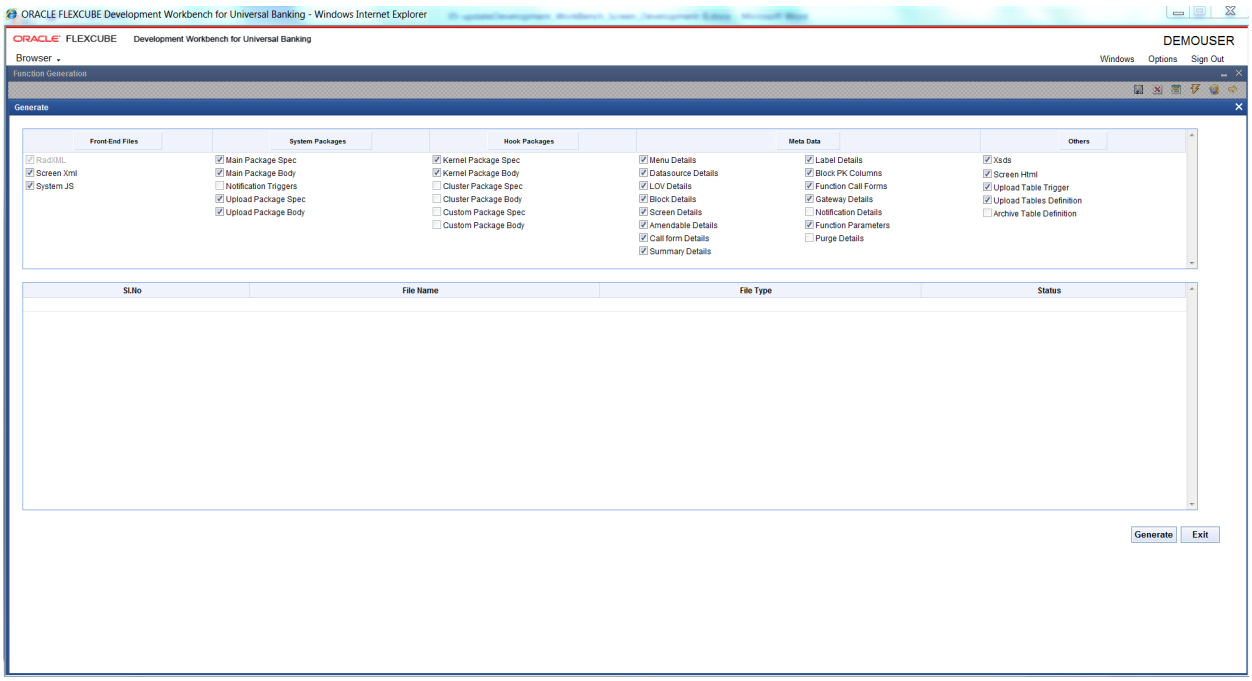


Fig 3.1: File Generation Screen

For action New , Workbench automatically selects the all files as shown in figure. If function ID has been loaded .then hook packages will not be checked

Click on Generate button to generated the files selected

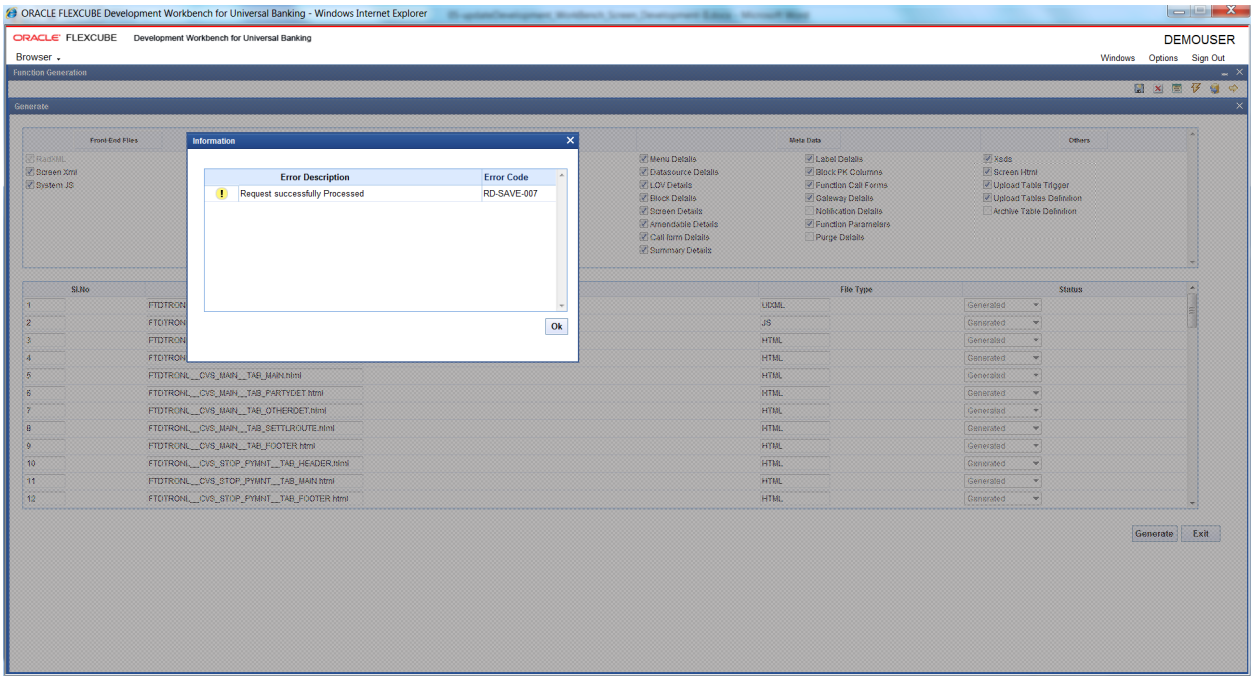


Fig 3.2: Files Generated message

3.1 Front-End Files

These files need to be copied to application server in respective folders.

- **RADXML** → This file contains the Data Used to Design screen. Radxml is Input to Workbench.
  File Name : <Function_ID>_RAD.xml

- **Screen Xml** → This file is an XML markup of presentation details, for the designed Function ID specific to a language. XSL Transformation is applied to this XML file by linking it to an XSL file. This results in HTML Markup which is displayed by the browser.
  File name : <Function_ID>.xml

- **System JS** → This JavaScript file mainly contains a list of declared variables used by FLEXCUBE Infra.

  - msgxml: - This variable is used by the system to build FCUBS Request XML
  - dataSrcLocationArray: - This variable is an array of DATA BLOCKS
  - relationArray:-This array contains relation and relation type details of blocks.This is used for data binding

- retflds and bndFlds:- These arrays contains LOV information
- CallFormArray, CallFormRelat, CallRelatType:- These arrays contains callform details, call form relation and relation type
- actionsAmmendArray: - This array contains information for enabling fields based on actions

File name : <Function_ID>_SYS.js

- **Release Specific JS →**

Note: Release Specific JavaScript File will not be generated by Workbench. This file, if required, has to be created by developer.

This JavaScript file allows developer to add functional code and is specific to Release.

The functions in this file are generally triggered by screen events. A developer working in KERNEL release would add functions based on two categories:

- Functions triggered by screen loading events
  Eg: fnPreLoad_KERNEL(),fnPostLoad_KERNEL()
- Functions triggered by screen action events
  Eg: fnPreNew_ KERNEL (),fnPostNew_ KERNEL ()

A developer working in CLUSTER release would add functions based on two categories:

- Functions triggered by screen loading events
  Eg: fnPreLoad_CLUSTER(),fnPostLoad_CLUSTER()
- Functions triggered by screen action events
  Eg: fnPreNew_ CLUSTER (),fnPostNew_ CLUSTER ()

In case if any function in KERNEL JavaScript file has to be modified, this can be achieved by overriding the function in CLUSTER JavaScript file.

The functions in this file are generally triggered by screen events. A developer working in CUSTOM release would add functions based on two categories:

- Functions triggered by screen loading events
  Eg: fnPreLoad_CUSTOM(),fnPostLoad_CUSTOM()
- Functions triggered by screen action events
  Eg: fnPreNew_ CUSTOM (),fnPostNew_ CUSTOM ()

In case if any function either in KERNEL JavaScript file or CLUSTER JavaScript file has to be modified, this can be achieved by overriding the respective function in CUSTOM JavaScript file

File name : <Function_ID>_<RELEASE_TYPE>.js

3.2 System Packages

- **Main Package Specification:** The Main Package Spec Contains all the Declarations.
- **Main Package Body:**

  Main Package would contain the all the business logic and persistence code for the functioned. It will also contain call to hook packages .Different flavors of main Package generated are

  1) Maintenance FunctionId
  2) Transaction FunctionId
  3) Others Category FunctionId
  4) Report Parameter Screen
  5) Notification
  6) Purge Entity

  The main package has the below stages in case of a maintenance function:

  - Converting Ts to PL/SQL Composite Type
  - Checking for mandatory fields
  - Defaulting and validating the data
  - Writing into Database
  - Querying the Data from database
  - Converting the Modified Composite Type again to TS

  Each of these stages has a 'Pre' and 'Post' hooks in the Release Specific Packages.

Note : Main Package has the system-generated code and should not be modified by the developer.

- **Upload Package Specification**
- **Upload Package Body**

  Upload Packages would support adapter upload flow of any Function ID. The package will contain code to convert upload table data to function id specific PL/SQL composite type. The main package of the functioned would be called to upload the data

- **Notification Triggers**
  This will be generated only for notification trigger screens. Refer *Development Workbench – Notifications* for further reference

3.3 Hook Packages

Kernel, Cluster and Custom Packages are the packages where the respective team can add business logic in appropriate functions using the Pre and Post hooks available.

- **Kernel Package Spec**
- **Kernel Package Body**

  The Kernel package is solely for the Kernel Team to modify. The Main package has designated calls to the Kernel package for executing any functional checks or validations included in the Kernel Package.

  All the user level validations and conditional operations should be included in Fn_Post_Default_and_Validate. This function is called from the Main Package after the execution of Fn_Default_and_Validate. User should avoid putting validations or code in any other function.

  In case user needs to add a separate function, the existing Workbench generated structure should not be changed. Instead the user can create a new package e.g. <MODULE>PKS_<FID>_UTILS package. The desired function can be included in this package and the call can be made from the Kernel Package.

- **Cluster Package Spec**
- **Cluster Package Body**

  The Cluster package is available to the Cluster Team to add any validations or Checks specific to the Cluster Team over and above the Kernel Team. The Kernel Team or the Custom Team should not modify the contents of this package.

- **Custom Package Spec**
- **Custom Package Body**

  The Custom package is available to the Custom Team only to add any validations or Checks over and above those already present in the Kernel and Cluster Packages.

3.4 Meta Data

Workbench generates **INC** files which contains insert scripts. These are static data required for the functioning of the function id. Developer needs to deploy generated INC's in FLEXCUBE schema.

- **Menu Details →** Under menu Details below 4 table entry Insert Scripts will get generated.

  SMTB_MENU
  SMTB_FUNCTION_DESCRIPTION
  SMTB_ROLE_DETAIL
  SMTB_FCC_FCJ_MAPPING

  If any one of table entry missing while designing new function ID, FLEXCUBE won't allow launching function ID.

- **Data source Details →** Generates INC for CSTB_FID_DATA_SOURCES. This data is required for uploading data to FLEXCUBE for the function id through excel.
- **Lov Details →** Generates INC for CSTB_LOV_INFO. FLEXCUBE will be using this information to get data for List of values provided in the screen
- **Block Details →** Generates INC for CSTB_FID_DATA_BLOCKS. This data is required for uploading data to FLEXCUBE for the function id through excel.
- **Screen Details →** Generates INC for CSTB_FID_SCREENS.
- **Amendable details →** Generates INC for below table. These scripts provides information regarding the amendable fields in the screen .Entry has to be present for the required field in these tables in order to make that field amendable through FLEXCUBE and gateway operation.

  GWTM_AMEND_MASTER
  GWTM_AMEND_NODES
  GWTM_AMEND_FIELDS
  GWTB_AMEND_NODES
  GWTB_AMEND_FIELDS
- **Call form Details →** Generates INC for CSTB_CALL_FORM_NODES. These will be generated only if the screen is a Call Form. It is required for attaching the particular Call form to any other function id.
- **Summary Details →** Generates INC for CSTB_SUMMARY_INFO. Dat in this table is required for fetching records in the summary screen of the particular function id .
- **Label Details →** Generates INC for CSTB_FIELD_LABELS and CSTB_OTHER_LABELS. Labels can be used for Language translation of screens. These scripts provides information regarding the label codes attached to each field. These are useful while developing language packs for FLEXCUBE

  CSTB_FIELD_LABELS → Contains Field Label details.
  CSTB_OTHER_LABELS → Contains SCREEN, TAB, FIELDSET, BLOCK etc label details.

  Note that Workbench does not generate scripts for CSTB_LABELS. Data in CSTB_LABELS is maintenance required for the Tool.
- **Block Pk Columns →** Generates INC for STTB_AUDIT_PK_COLS.
- **Function Call Forms →** Generates INC for CSTB_FID_CALLFORMS. This data is required for uploading data to FLEXCUBE for the function id through excel.
- **Gateway Details →** Generates INC for below tables.
  Data in these tables are required for gateway operations (web services) for the particular screen

  GWTM_FCJ_FUNCTIONS
  GWTM_OPERATIONS_MASTER
- **Notification Details →** Generates INC for below tables. Notification INCs will get generated only for Notification screens

  GWTM_NOTIFICATION_TRIGGERS

GWTM_NOTIFICATIONS_MASTER

- **Function parameters →** Generates INC for below tables. This INC gets generates only for Reports screens (i.e. Function Category – Report).

  AETB_FUNC_MASTER

  AETB_FUNC_DETAIL
- **Purge Details →** Generates INC for below tables . These scripts will be generated only for Purge Entity Definitions screens

  STTM_PURGE_MASTER

  STTM_PURGE_TBL_DETAILS

  STTM_PURGE_FILTERS

3.5 Others

- **Xsds →** The XSD files are used for the creation of xmls, which are in turn used during the execution of operations through Gateway.
- **Screen html →** The HTML files are used to take screen shots by converting them to images, which in turn used for Document preparation.
- **Upload Table Trigger →** Trigger will be generated for inserting records into adapter upload process tables (CSTB_EXT_CONTRACT_STAT ,STTB_UPLOAD_MASTER) from upload tables.
- **Upload Table definition →** DDL for upload tables would be generated. If existing upload tables are being re used , the generated DDL's can ignored
- **Archive Table definition →** DDL for archival tables would be generated for Purge Entity Radxml. Synonyms for the tables would also be generated.

  Note: For new screen all file options will be checked automatically. For modification only system generated files will be checked, based on user requirement can opt for other file generation.

4 Deployment

Workbench generated files can be deployed directly to the application server and schema. Front-end files get copied to desired location which user has maintained while adding environment details. Database files get compiled to FLEXCUBE schema which is specified in the working environment.

4.1 Prerequisites:

The below maintenance is required to use the deploy option.

In environment user has two options to copy files based on the operating system FLEXCUBE application is running.

- **Windows : File Copy / FileManager**

- **Unix : FileManager**
  FileManager Servlet needs to deploy in application server where FLEXCUBE is running. Refer File manager section in *Development Workbench – Administration* to get more details about file manager.

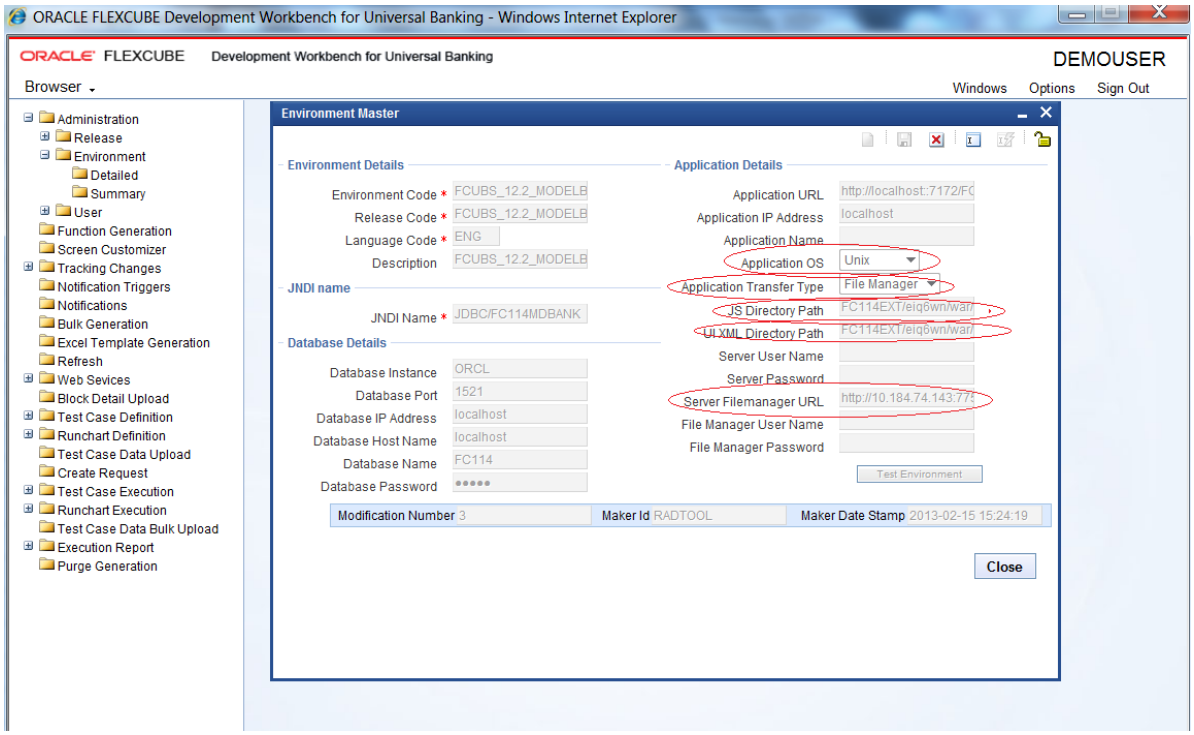


Fig 4.1: The Environment Details

In environment creation below details are mandatory to use deploy option.

- **Application Operating System** → Unix/Windows
- **Application Transfer Type** → Windows → File Copy/FileManager
  UNIX → FileManager
- **JS Directory Path :** Enter the shared path of JS files.
- **Uixml Directory Path:** Enter Shared path of Language Xml files.
  Note: If appliaction server is windows user can enter absolute path.

4.2 Deploy Files:

Click Deploy button as shown in below image.

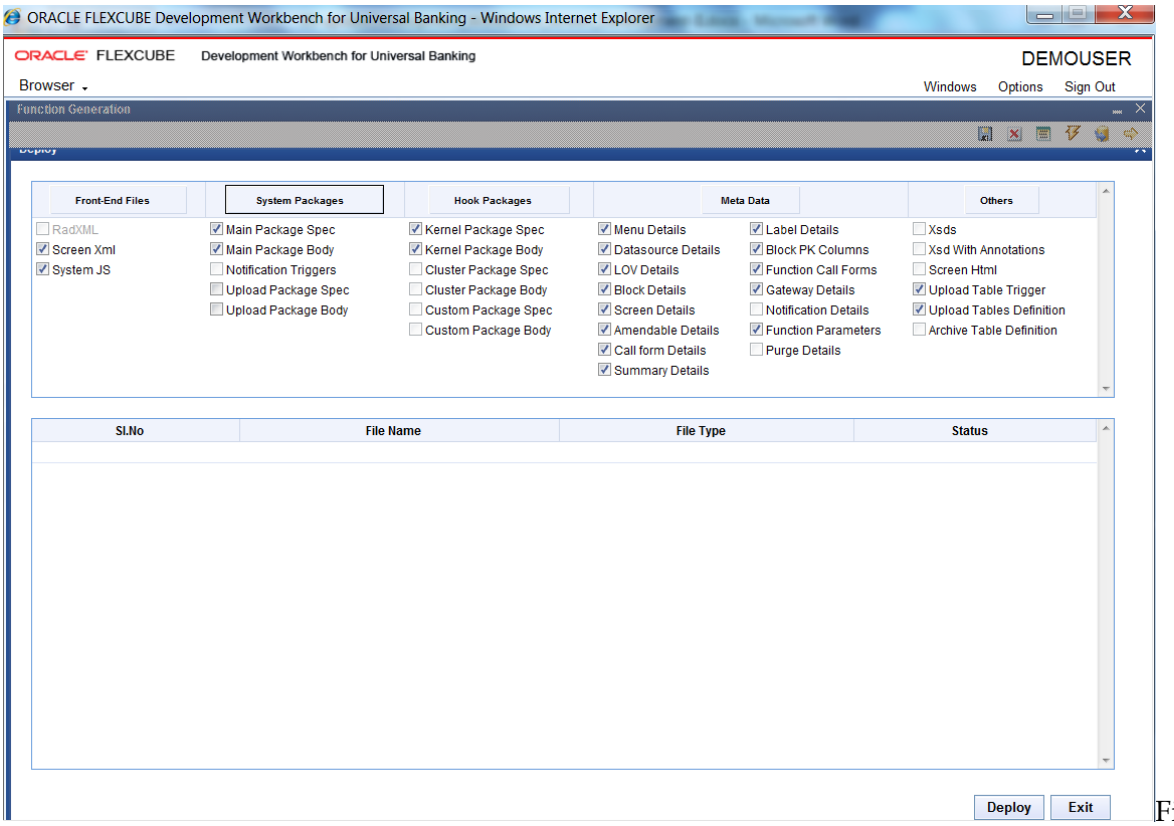


Fig 4.2: The deployment Screen

Select required files to deploy .Click deploy button. On success full deployment user will get status as Deployed.

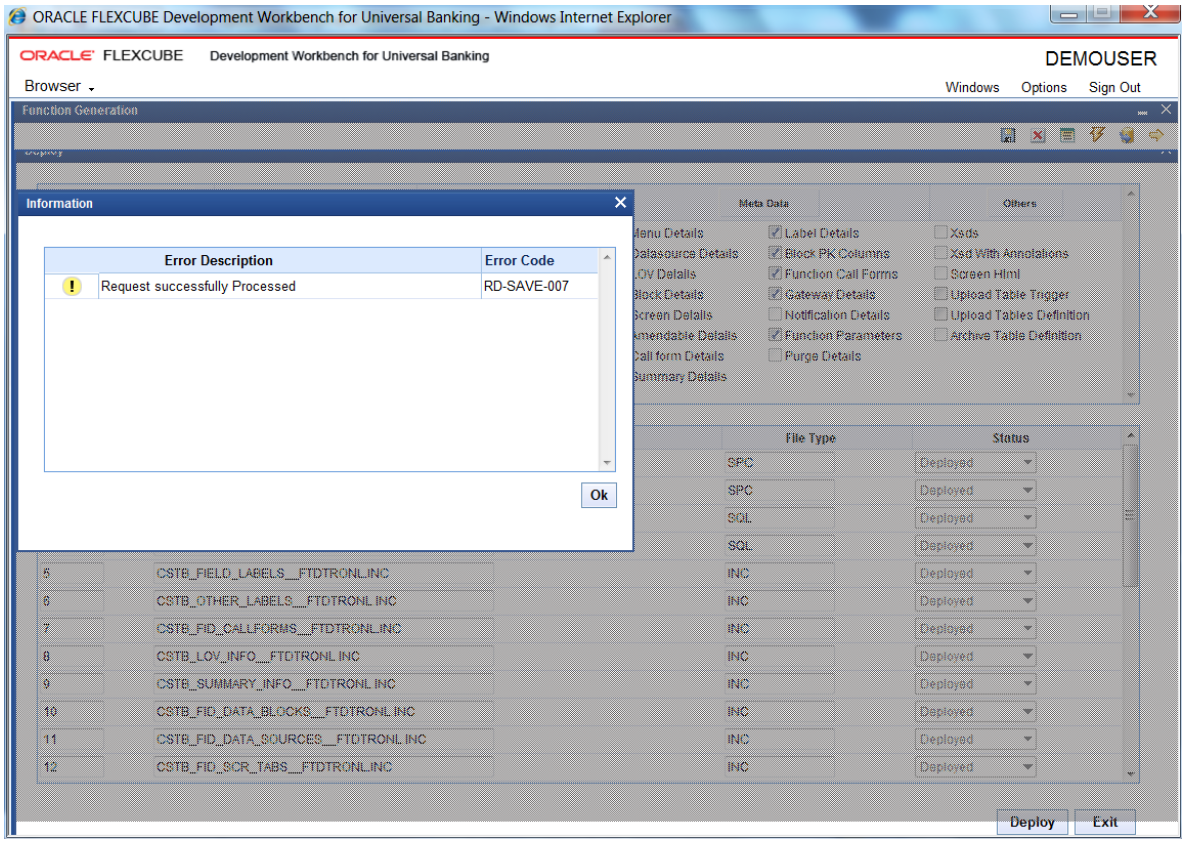


Fig 4.3: Deployment of selected files

5 Release

Release feature of Workbench is applicable only for developers within Oracle Financials. It involves the integration the Tool with other in house tools (repository/version control tools)

Oracle FLEXCUBE Development Workbench is integrated with SVN.

5.1 SVN Integration

Now, Workbench provides an interface to release the units generated to SVN

The following prerequisite should be adhered to before attempting to release units to SVN

1) SVN Repository URL should be provided in the Preferences Node of Function Id against field SVN Repository URL. Path till the module name has to be given.
   Example:- For an FT module RADXML path can be given as
   Z:\FCUBS_11.3.0\Soft\MAIN\FT

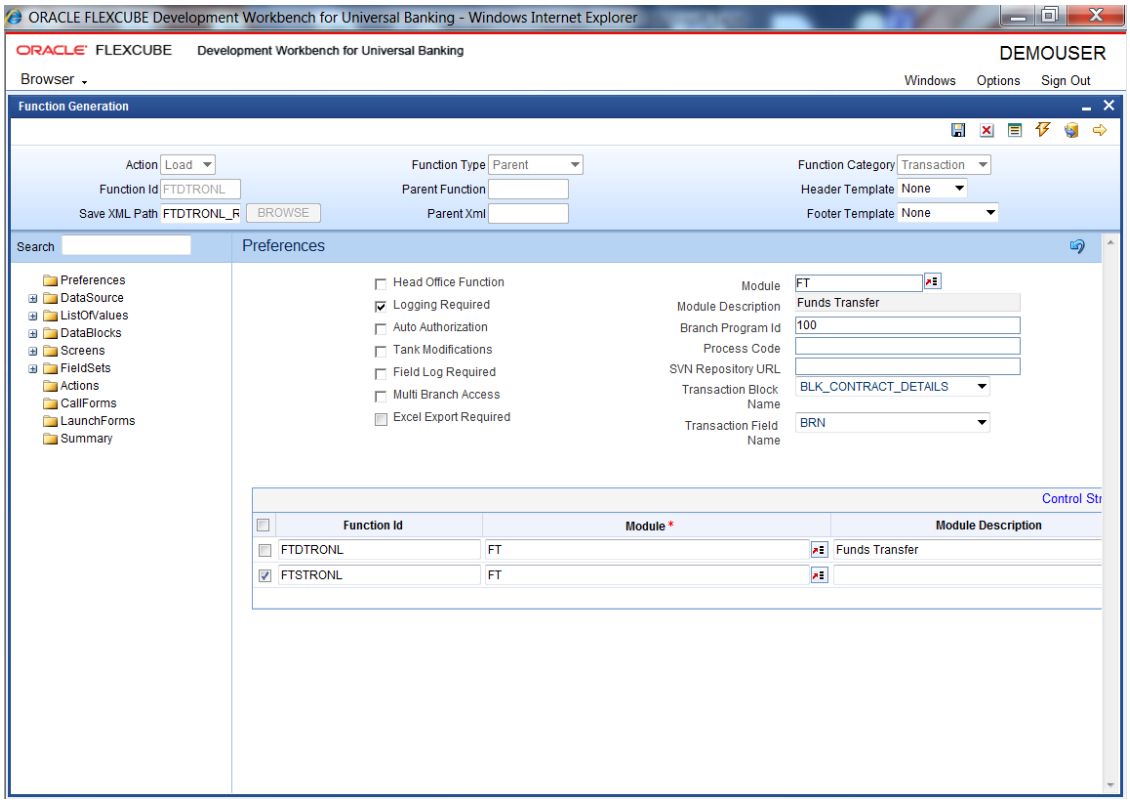


Fig 5.2.1: Preferences Screen highlighting SVN Repository URL

The following files can be released to SVN through Development Workbench:

1) RADXML
2) Main Package(spc and sql)

From the release screen check on the units to be released to SVN and click OK .SVN credentials has to be provided in the screen. SFR number has to be provided if it is not a simple check in. If checked in successfully, status will be displayed as released.



Development Workbench – Screen Development II
October 2026
Version 14.8.3.0.0

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/

Copyright © 2007, 2026, Oracle and/or its affiliates. All rights reserved.

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

U.S. GOVERNMENT END USERS: Oracle programs, including any operating system, integrated software, any programs installed on the hardware, and/or documentation, delivered to U.S. Government end users are "commercial computer software" pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, use, duplication, disclosure, modification, and adaptation of the programs, including any operating system, integrated software, any programs installed on the hardware, and/or documentation, shall be subject to license terms and license restrictions applicable to the programs. 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 failsafe, 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.

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.

This software or hardware and documentation may provide access to or information on 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. 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.