Quick answers to common problems

Oracle E-Business Suite R12 Integration and OA Framework Development and Extension Cookbook

A practical step-by-step guide to develop end-to-end extensions to Oracle E-Business Suite Release 12, with detailed illustrations and explanations

Andy Penver

[PACKT] PUBLISHING enterprise professional expertise distilled

Oracle E-Business Suite R12 Integration and OA Framework Development and Extension Cookbook

A practical step-by-step guide to develop end-to-end extensions to Oracle E-Business Suite Release 12, with detailed illustrations and explanations

Andy Penver

[PACKT] PUBLISHING enterprise
professional expertise distilled

BIRMINGHAM - MUMBAI

Oracle E-Business Suite R12 Integration and OA Framework Development and Extension Cookbook

Every effort has been made in the preparation of this book to ensure the accuracy of the information presented. However, the information contained in this book is sold without warranty, either express or implied. Neither the author, nor Packt Publishing, and its dealers and distributors will be held liable for any damages caused or alleged to be caused directly or indirectly by this book.

Packt Publishing has endeavored to provide trademark information about all of the companies and products mentioned in this book by the appropriate use of capitals. However, Packt Publishing cannot guarantee the accuracy of this information.

First published: March 2013

Production Reference: 1140313

Published by Packt Publishing Ltd.
Livery Place
35 Livery Street
Birmingham B3 2PB, UK.

ISBN 978-1-84968-712-6

`www.packtpub.com`

Cover Image by Neha Rajappan (`neha.rajappan1@gmail.com`)

Credits

Author
Andy Penver

Reviewers
Parvez Achhwa
Brian Badenhorst
Stephen Phillips

Acquisition Editor
Rukhsana Khambatta

Lead Technical Editors
Suchi Singhal
Sruthi Kutty

Technical Editors
Prasad Dalvi
Amit Ramadas

Project Coordinator
Anish Ramchandani

Proofreaders
Aaron Nash
Linda Morris

Indexer
Hemangini Bari

Production Coordinator
Conidon Miranda

Cover Work
Conidon Miranda

About the Author

Andy Penver currently lives and works in the U.K. as a solution architect for a large public sector client. He studied at Christ Church (University of Kent). He has over 18 years of experience in working with Oracle E-Business Suite. He has worked in both the private and public sectors, and has a strong technical background. He has led and managed teams of consultants through the full project lifecycles on some very large programs throughout the U.K. and Europe. Andy has been heavily involved in two large-scale, award-winning implementations of a shared service centre.

Andy is currently the Managing Director of his own consultancy, NU-TEKK Limited.

Andy has previously authored and published a book called *Oracle E-Business Suite R12 Core Development and Extension Cookbook*.

I would like to thank my parents, my wife and three children, SJ, James, and Jack for their understanding and support while I spent many hours and late nights on the book. I would also like to thank my colleagues for their help in reviewing and commenting on the material in the book. In particular, I would like to thank Brian Badenhorst, Steve Phillips, and Parvez who all have helped me in some way to produce this book.

About the Reviewers

Parvez Achhwa has more than 22 years of experience in information systems technologies, with the last 17 years in Oracle Apps 11*i*/12*i* (CRM and Financials) implementation, BI, Business Process Automation, and management of large-scale projects. He is a rare breed to possess both functional and technical skills at various landscapes. In recent years, he has also developed Siebel implementation skills.

Presently, he is working with DAMAC Holding as Oracle ERP Manager.

> I would like to thank Andy for giving me the opportunity to review this book.

Brian Badenhorst works as an Oracle E-Business Consultant and has over 10 years of experience in designing and developing robust solutions for a range of clients in both the UK private and public sectors. Backed by an engineering background, he has the ability to put forward strategic solutions and he has significant experience in fulfilling bespoke requirements.

Brian works as a freelance consultant and is the Managing Director of his own business.

This is the first book that Brian has been involved in.

Stephen Phillips was born in London in 1959 and moved to Australia in 1969 when his family emigrated. He studied Computer Science at the University of Melbourne and has worked in IT in both Australia and the U.K. for over 30 years. He began his career at the Government Computing Service (GCS) in Melbourne, Australia in 1981, working on Burroughs mainframes. Since that time he has worked in all scales of equipment and on projects ranging from large-scale implementations of Oracle HRMS to small PC-based systems for a single client.

Stephen has worked extensively with Oracle software since 1989 and moved back to London in 1994 in order to broaden his experience.

Since his return to the U.K. he has worked for a number of prestigious clients, including Oracle UK, Reuters, Lloyds TSB, Walkers Snack Foods, The John Lewis Partnership, and the Ministry of Justice. Stephen has worked on Data Migration projects with a number of these clients, as part of their implementation of Oracle HRMS/Payroll.

In 2003, Stephen completed the MSc degree with the Open University. The major area of his dissertation was artificial intelligence, as applied to the compiling of public transport timetables.

Table of Contents

Preface

Extending Oracle E-Business Suite

This book is about the ways in which we can extend Oracle E-Business Suite (EBS) and focuses on more recent tools and technology. Its primary focus is to show how we can integrate with EBS, personalize and develop OA Framework pages with EBS, and to show how we can use BI Publisher to create and mail merge documents within EBS. The book has many detailed examples to work through with tips and explanations about how various components can be configured and how we can extend standard functionality and the various ways in which we can do it. It complements my first book, *Oracle E-Business Suite R12 Core Development and Extension Cookbook*, which focuses on writing concurrent programs, the personalization and development of professional forms, creating workflows, and using common utilities and scripts within EBS.

This book is aimed at developers who are new to E-Business Suite, or those who are strong in one particular area and need to expand their knowledge in other areas. Experienced developers may also use the book to brush up on their skills, or to pick up tips that may help them. The book focuses on the newer technologies being used in the more recent releases of EBS. There are plenty of detailed screenshots throughout each chapter giving clear instructions. This helps to provide a clear and full understanding of what we are doing and why. Each topic will develop a solution to a scenario, and will show how we need to set up our development tools right through to deploying it within EBS.

At the end of each chapter, the reader should have developed a good understanding of the topic and will be able to take away the knowledge gained and start using it in practice. The book also comes with complete, fully-tested code and scripts that can be downloaded. The examples have been developed using a Vision instance of Oracle E-Business Suite Release 12 (12.1.1). It would be helpful to know some SQL, PL/SQL, XML, and Java, but it is not essential, as the code is already written and fully documented. We will be going through each chapter example step-by-step, so you will not be expected to write any code that is not detailed in the book.

If you are new to Oracle EBS or a consultant who has worked with Oracle E-Business Suite before, you will be aware that there are many technical components to consider; there are many features using a range of different tools and the footprint seems to be expanding all the time. One of the key features of EBS is its flexibility. Not all organizations are the same and there are many ways you can configure EBS to make it specific to an organization's needs. Whenever possible, an organization should always attempt to use standard functionality to meet their requirements. However, there are occasions where business requirements cannot be met using the standard Oracle-provided functionality, and this can be anything from renaming a label on a screen, to automating a process that would take many hours for someone to process manually.

There are various ways in which you can change behavior in Oracle EBS. The simplest way to change behavior is through configuration. For example, extending Oracle through profile options, value sets, descriptive flexfields are ways to extend applications through configuration. In addition, Oracle also gives us the ability to change Oracle OA Framework pages through personalization. It is a powerful feature that allows us to change behavior without needing to change any standard objects. These are mechanisms provided by Oracle to change how the application looks, or how the application behaves. On the other hand, there are some extensions that require writing code, creating new objects, or even extending or replacing existing objects. These types of extensions are in addition to the code or objects that Oracle delivers.

A powerful feature of Oracle is the ability to extend EBS, but when doing this, there is a strict set of development standards that must be adhered to. There are two primary ways to modify Oracle EBS; the first is customization by extension, as we have just described, and the second is customization by modification. Customization by modification is where standard objects are changed, meaning that the change needs to be reapplied when a newer version of the object is released by Oracle.

Customizations by modification are not supported by Oracle and should always be avoided. There are rarely occasions where there is a real business justification for a customization by modification. There are often alternatives, such as exploring other solutions or looking at ways to change the requirements or processes. It is important to understand the difference between customization by extension and customization by modification, as the two terms are often used loosely. When we extend Oracle EBS by supported methods, the standard functionality is still supported by Oracle as it has not been amended by the extension, whereas customization by modifying a standard object means the standard object also becomes unsupported by them.

A worthwhile feature of Oracle EBS is that it uses a common toolset and also has a vast amount of documentation written about the supported ways in which you can extend the product. The first place to start would be the Oracle Release 12 Documentation Library. This provides many documents, all relating to Oracle EBS. Another essential resource is Oracle Support (previously known as Metalink), which is a portal provided by Oracle for support, documentation, white papers, and patches among many other things related to EBS. To access the portal, you need to register and you also need to provide a support identifier, which is only provided when there is a support contract with Oracle Support. So, if you are new to EBS or if you have been around for a while, you can nearly always find examples or documentation relating to your specific requirement. There are nearly always numerous business processes and solutions available to satisfy it. The best way to provide a solution is to have an understanding of what options are available. Having knowledge of the various ways in which we can extend EBS will give us a much better chance of coming up with better solutions. This book will provide recipes that will cover some simple and some more complex solutions. It will utilize a majority of the Oracle toolset and will, hopefully, broaden your knowledge. Expanding our knowledge of the toolset will allow us to provide a more varied set of solutions, resulting in having a better chance of providing a better, more robust solution for a given problem.

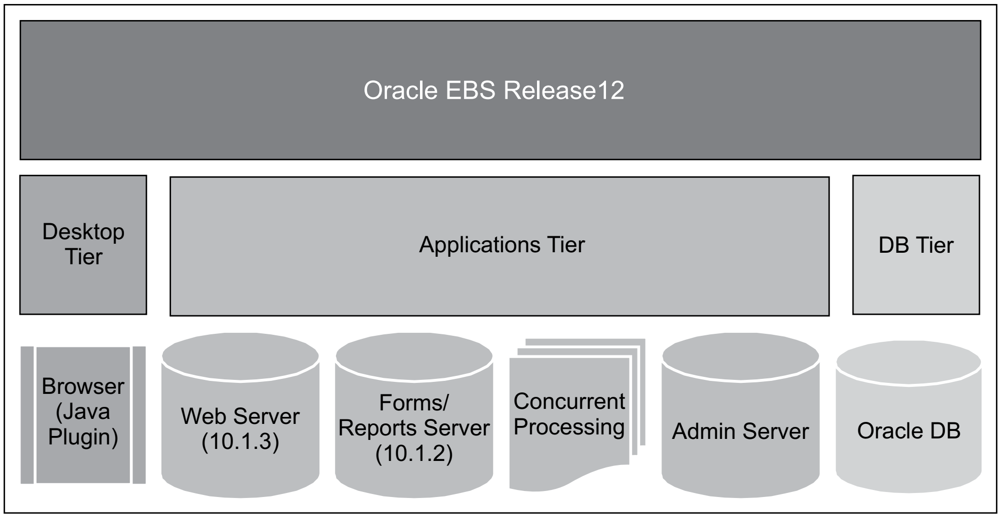
Understanding the EBS architecture

Before we really get into the book it is important to understand the E-Business Suite architecture in Release 12. We need to understand where the files are kept and how it is installed. If you are a beginner, you will need this information to understand some key concepts and pick up some terminology that is often used when we discuss Oracle EBS.

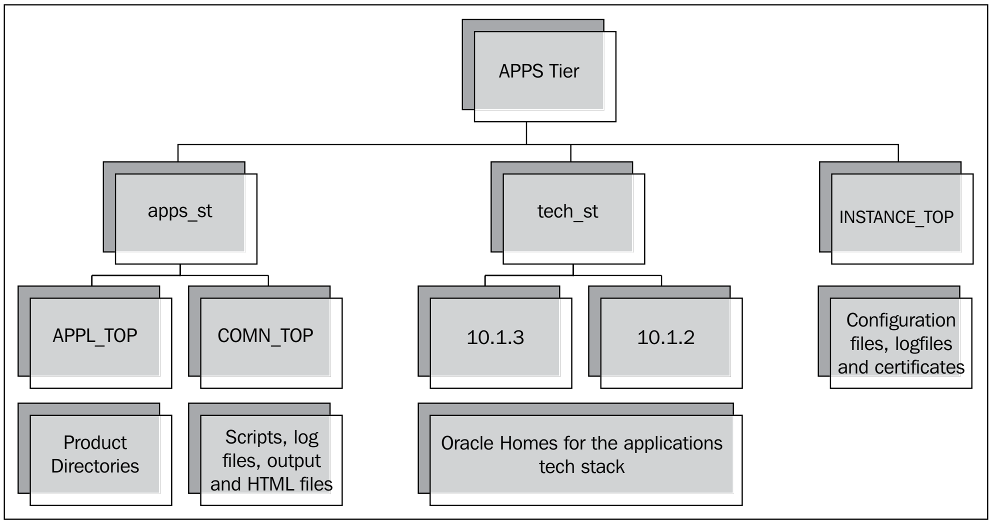
In Release 12, there has been a significant change in the file system when it is installed. The change quite simply is used to segregate code, data, and configuration. This makes maintenance much easier. There will be a number of terms that you will encounter regularly when we discuss EBS. A server is a term for a number of processes that provide specific functionality on a single machine. A tier is a term used to describe a logical group of services, which can be on one or more physical machine. A machine is used to describe a computer or group of computers. A node is a group of computers that work closely together as a cluster.

Essentially, there are three tiers per instance of EBS. The Database Tier (DB Tier), the Application Tier (APPS Tier), and the Desktop Tier. The desktop tier is the client interface where users will connect to Oracle EBS through a web browser. When Oracle is first used, the browser will install a J2SE plugin which will use Oracle’s own Java Virtual Machine (JVM) rather than the browser's own JVM. The application tier (also known as the middle tier) processes all of the business logic. It comprises of three servers or service groups. These are Web Services which process requests from the desktop client. Then there are the Forms services which manage all of the listening and secure requests for Oracle forms. Then there is the concurrent processing server, which processes concurrent requests that are submitted.

When Oracle EBS is installed on all variants of UNIX, the install is performed by the root user. However, as part of the installation process there will be two OS users. One that will own the application node file system (the applmgr user) and the other will own the database node file system (the oracle user). When installed on Windows, there is one OS user that owns the file system. The following diagram shows the basic architecture:



The directory structure of the Application Tier and Database Tier will help to understand where the files are stored. The following directory structure should help to understand where various files are stored within EBS:


APPS Tier
apps_st
tech_st
INSTANCE_TOP
APPL_TOP
COMN_TOP
10.1.3
10.1.2
Configuration files, logfiles and certificates
Product Directories
Scripts, log files, output and HTML files
Oracle Homes for the applications tech stack

The following shows the directory structures of some core directories on the Applications Tier:

Profile	Directory	Description
`APPL_TOP`	`/apps_st/appl`	This directory is the base directory where all of the product tops are stored.
`AU_TOP/forms/US`	`/apps_st/appl/au/12.0.0/forms/US`	This directory is where all forms are stored.
`AU_TOP/resource`	`/apps_st/appl/au/12.0.0/resource`	This directory is where all libraries are stored.
`COMMON_TOP`	`/apps/apps_st/comn`	This is the base directory for scripts, log files, output files, and HTML files.
`ORACLE_HOME`	`/apps_st/10.1.2`	This is the directory structure for the forms' server files.
`iAS_ORACLE_HOME`	`/apps_st/10.1.3`	This is the directory structure for the web server files.
`INSTANCE_TOP`	`/inst/apps/<CONTEXT_NAME>`	This is the top directory for the configuration files.
`ADMIN_SCRIPTS_HOME`	`/inst/apps/<CONTEXT>/admin/scripts`	This is where admin scripts are stored to start and stop services.

The following table describes where the files are stored on the Database Tier:

Profile	Directory	Description
`ORACLE_HOME`	`/db/tech_st/10.2.0`	This directory is the base directory where all of the database files are stored.
Data Files	`/db/apps_st/data`	This directory is the base directory where all of the data files are stored.

Within EBS, each product has its own database user within Oracle. The user will own the objects that store data, such as tables. This is in the form of an abbreviation of the product (AP for Account Payables and HR for Human Resources for example) and is referred to as a schema. The passwords are, by default, the same as the schema name, but this is nearly always changed after install. On the file system on the APPS tier, each product has its own file system. The file system has the same root directory as the product \ user name in Oracle. For example, there will be a directory structure for Payables (AP) that has a root directory of AP_TOP. There is one very important schema called the APPS schema. The apps schema owns all of the code, such as packages, triggers, views, java classes. Each product user has grants providing access to the APPS schema for all of its objects. Therefore, the APPS schema has access to all of the objects, that is the code and the objects owned by the product schemas. It is important that as a developer, you can access the APPS database schema. All objects that you will require access to are accessible from this schema. When we create custom objects, we need to create a custom schema and all of the data storing objects will reside in this schema. Usually, an on-site DBA will perform this task, but we will go through this in one of the recipes we have.

If you would like to understand more relating to the core concepts of EBS, you can get more detailed information from the Oracle Applications Concepts Release 12 from the Release12 Documentation Library. You can get the documentation from Oracle by following this link `http://docs.oracle.com/cd/B53825_08/current/html/docset.html`.

Before we start

One final word before we get into the recipes. Hopefully, you will find the book extremely useful. There are thousands of pages written about how to extend and customize Oracle E-Business Suite. This book is not going to cover every solution to every problem. In fact, it will not even come close. The recipes are designed to take us through the different features of the tools and extensions that you may need to utilize in your own solutions. Use the book to get a feel for how each tool and product can be used. It will provide a core understanding of how things can be done and the standards that we need to adhere to. You will need to expand on what you learn here and apply the knowledge you gain to design a solution or solutions to your own scenarios. I personally am a great believer of learning through practice. Expanding your knowledge by understanding the fundamentals is essential to providing the best solutions. Oracle uses many tools and technologies in its suite of applications, of which there is too much to cover in this book.

What this book covers

Chapter 1, *Personalizing OA Framework Pages*, looks at some recipes that will implement personalizations to an OA framework page. We will look at a number of different types of personalization and the levels at which we can create personalizations. At the end of the chapter, readers will have an understanding of how we can personalize OA framework pages in EBS. Readers will understand which responsibilities are used to administer personalizations and which profile options need to be changed to view the personalization links. Throughout the chapter, we will be creating a variety of different examples that will give a broad understanding of what we can achieve through personalizing OA framework pages.

Chapter 2, *Getting Started with OA Framework Pages*, deals with the creation of OA Framework pages. We start off by looking at the architecture of OA Framework pages as it is important to understand this before we start any development. Further, we will go through the process of setting up our development environment and what we will need to do to get up and running. Towards the end of the chapter readers will have an understanding of the components required to develop and test an OA Framework page.

Chapter 3, *Creating a Master Detail Page in OA Framework*, will explain the process of creating a master detail page in the OA Framework. Readers will have an understanding of how we can create master detail pages in the OA framework. Finally, we will go through the process of deploying our page in Oracle E-Business Suite (EBS) and running it through the application.

Chapter 4, *Adding a Creation Page and LOV Region in OA Framework*, explains how to make changes to the OA Framework pages we have created. This chapter gives the readers an understanding of how they can create pages that interact with the database. We will also know how to use the debugger that is provided in JDeveloper.

Chapter 5, *Advanced OA Framework*, helps in learning some more advanced features of OA Framework pages which includes a lot more coding. The chapter will continue to develop the pages we have worked on so far.

Chapter 6, *BI Publisher*, looks at recipes that will explain how BI Publisher integrates with E-Business Suite to allow us to generate formatted data with a very feature rich output. We will look at how it integrates with commonly used desktop applications, such as Microsoft Word and Excel. At the end of the chapter, readers will understand how to set up our development environments. We will learn to create our report templates, and how to generate the XML data that the template requires to produce a report output in a number of different formats. We will look at how the report mail merges the data with Word documents, and how we can use a feature called Bursting to send formatted documents out by e-mail.

Chapter 7, *Desktop Integration*, looks at how we can integrate with E-Business Suite using the desktop integrator. Oracle E-Business Suite provides a framework that allows us to integrate with Microsoft Office-based tools. It is called Web ADI (Application Desktop Integrator). Users tend to be in favor of using the tools, as it integrates with the Microsoft Office tools we are all used to working with. At the end of this chapter readers will learn how we can use Excel-based spreadsheets to view and upload data to EBS. We will understand how data can be validated, and how we can include structured data entry methods, such as lists of values and drop down lists to spreadsheets. We will understand how we can create a new integrator to upload data and how to assign an integrator to a responsibility.

Chapter 8, *Utilities*, looks at a number of utilities that you will find extremely useful. We will show how to create a custom schema and how to configure it on EBS. We will also look at ways to use common utilities, which will reduce risk and save enormous amounts of time. Usually, the biggest reason for problems when migrating extensions or configuration is when there are human tasks. We will also look at how to start and stop an environment.

What you need for this book

Oracle uses many great tools to develop extensions. The tools we will mostly focus on in this book are the following:

- Oracle JDeveloper with OA Extension
- SQL Developer
- BI Publisher
- Microsoft Excel

Microsoft Word

Other utility programs we will be using are:

- WinSCP — a windows based FTP frontend
- PuTTY — a UNIX command line window
- JAD - a Java decompiler

Developing code in EBS

There are a number of languages used for programming and writing scripts that Oracle supports. We are going to primarily focus on the following in this book:

- PL/SQL
- Java
- SQL
- XML

UNIX shell scripting

Other languages are also less commonly used within EBS and fall outside the scope of this book.

Who this book is for

This book is written for individuals who want to learn how to develop extensions in Oracle E-Business Suite. If you are involved in development or supporting an E-Business Suite implementation, you should find this book very useful. The book gives detailed explanations, so minimal technical expertise is required. It is suitable for beginners who have little experience, or developers who may want to use the book to brush up on their skills.

Conventions

In this book, you will find a number of styles of text that distinguish between different kinds of information. Here are some examples of these styles, and an explanation of their meaning.

Code words in text are shown as follows: "You may also notice that there is also a ZIP file called `oafch2.zip` provided in the code bundle."

A block of code is set as follows:

```
SELECT flv.lookup_code,
       flv.meaning
  FROM fnd_lookup_values flv
 WHERE flv.lookup_type = ‘XXHR_SOCIETY_LOV’
```

Any command-line input or output is written as follows:

```
$JAVA_TOP/<customFolder>/oracle/apps/<ApplicationShortName>/<moduleName>/
<subModuleName>/webui
```

New terms and **important words** are shown in bold. Words that you see on the screen, in menus or dialog boxes for example, appear in the text like this: "Obtain the version of EBS you are on. To do this, log on to EBS and click on the **About this Page** link.".

> Warnings or important notes appear in a box like this.

> Tips and tricks appear like this.

Reader feedback

Feedback from our readers is always welcome. Let us know what you think about this book—what you liked or may have disliked. Reader feedback is important for us to develop titles that you really get the most out of.

To send us general feedback, simply send an e-mail to `feedback@packtpub.com`, and mention the book title via the subject of your message.

If there is a topic that you have expertise in and you are interested in either writing or contributing to a book, see our author guide on `www.packtpub.com/authors`.

Customer support

Now that you are the proud owner of a Packt book, we have a number of things to help you to get the most from your purchase.

Downloading the example code

You can download the example code files for all Packt books you have purchased from your account at `http://www.packtpub.com`. If you purchased this book elsewhere, you can visit `http://www.packtpub.com/support` and register to have the files e-mailed directly to you.

Errata

Although we have taken every care to ensure the accuracy of our content, mistakes do happen. If you find a mistake in one of our books—maybe a mistake in the text or the code—we would be grateful if you would report this to us. By doing so, you can save other readers from frustration and help us improve subsequent versions of this book. If you find any errata, please report them by visiting `http://www.packtpub.com/submit-errata`, selecting your book, clicking on the **errata submission form** link, and entering the details of your errata. Once your errata are verified, your submission will be accepted and the errata will be uploaded on our website, or added to any list of existing errata, under the Errata section of that title. Any existing errata can be viewed by selecting your title from `http://www.packtpub.com/support`.

Piracy

Piracy of copyright material on the Internet is an ongoing problem across all media. At Packt, we take the protection of our copyright and licenses very seriously. If you come across any illegal copies of our works, in any form, on the Internet, please provide us with the location address or website name immediately so that we can pursue a remedy.

Please contact us at `copyright@packtpub.com` with a link to the suspected pirated material.

We appreciate your help in protecting our authors, and our ability to bring you valuable content.

Questions

You can contact us at `questions@packtpub.com` if you are having a problem with any aspect of the book, and we will do our best to address it.

1
Personalizing OA Framework Pages

In this chapter we will cover:

- Getting started with personalizations
- Setting the personalization profile options
- Discovering information about a page
- Clearing the cache
- Creating a user-level personalization
- Adding tool tips
- Creating an item
- Re-ordering items on a page
- Adding a button to a page
- Hiding a button on a page
- Making a field required
- Adding a flexfield to a page
- Using SPEL to trigger personalizations
- Deactivating personalizations
- Deleting personalizations

Introduction

OA Framework pages have been implemented using an object-oriented design approach that allows us to be pretty flexible when it comes to making changes to the user interface. The objects that we see on each page are made up of smaller objects that are stored independently in a repository called Meta Data Service (MDS).

In this chapter, we will discuss how to make changes to the user interface through personalization. When we personalize a page, we are altering the declaration of a page's interface. At runtime, objects are loaded from the repository and are rendered in a browser. What we see is determined by the metadata definition, which specifies how objects are displayed on the page. We can create a personalization at multiple levels such as site, function, organization, or responsibility. The good thing is that they can be configured directly from the UI page and are likely to survive an upgrade, as opposed to extensions. One thing you cannot modify with personalizations is the business logic.

At the end of this chapter, readers will have an understanding of how we can personalize OA Framework pages in EBS. Readers will understand which responsibilities are used to administer personalizations and which profile options need to be changed to view the personalization links. Throughout this chapter, we will be creating a variety of different examples that will give a broad understanding of what we can achieve through personalizing OA Framework pages.

Getting started with personalizations

It is important that we understand some common terminology and gain an understanding of how an OA Framework page is constructed before we get started. Let's start off by understanding that OA Framework has been developed by Oracle and is based upon a **Model View Controller** (**MVC**) architecture developed using Java 2 Platform, Enterprise Edition (J2EE).

- **Model**: A model can be thought of as the data and is implemented using **Oracle Business Components for Java** (**BC4J**). This comprises three layers, which are the **entity object** (**EO**), **view object** (**VO**), and **application module** (**AM**).
    - **Entity object**: This is based upon a database table and all of the database transactions will go through the EO.
    - **View object**: This is based upon any number of entity objects or it can be an SQL statement, if the page is read only.
    - **Application module**: This is a container for the view object and it manages transactions that occur. Each page has a root AM to maintain the transaction context.
- **View**: A view defines a user interface that is rendered as an HTML page. This is implemented with **user interface XML** (**UIX**).

- **Controller**: A controller is java code that handles events when they occur at runtime. There are two methods that are called when the page runs.

In this recipe, we are going to set up some records that we will use throughout the chapter. We are going to demonstrate our personalizations in the manager self-service screens. We will need to perform the following tasks before we get started with personalizing OA Framework pages:

- Creating a custom responsibility
- Creating a new user
- Creating an employee record
- Assigning a manager
- Attaching an employee to a user record

Creating a custom responsibility

Now to create our new responsibility that we will use to access core HR screens, we will create a test employee record and ensure that the employee is a manager of other employees. This will be used to access some of the self-service screens where we will create some personalizations in the upcoming recipes.

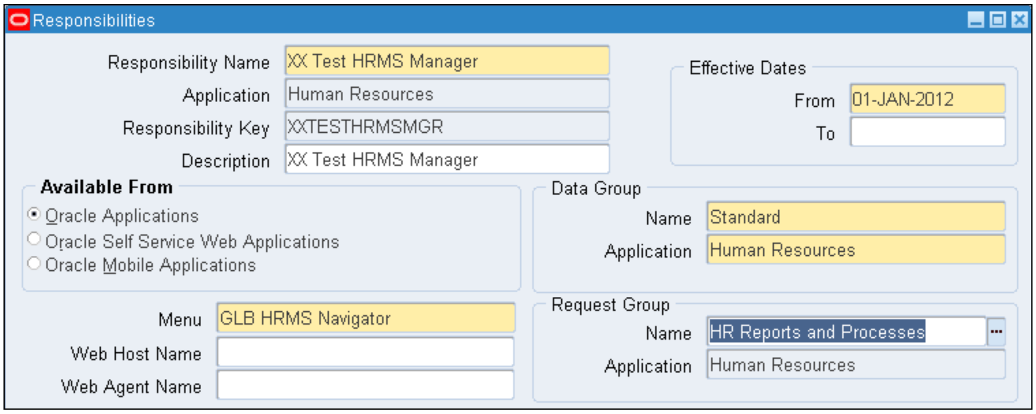
How to do it...

Perform the following steps to create a new responsibility called `XX Test HRMS Manager`:

1. Log in to Oracle E-Business with the **System Administrator** responsibility.
2. Navigate to **Security** | **Responsibility** | **Define** and the **Responsibilities** window will open.
3. Enter the required data as given in the following table:

Item name	Item value
Responsibility Name	`XX Test HRMS Manager`
Application	`Human Resources`
Responsibility Key	`XXTESTHRMSMGR`
Description	`XX Test HRMS Manager`
Data Group	
Name	`Standard`
Application	`Human Resources`
Menu	`GLB HRMS Navigator`
Request Group	
Name	`HR Reports and Processes`

4. Click on the **Save** button in the toolbar (or press *Ctrl* + *S*) to save the record.
5. The **Responsibilities** screen should now appear as shown in the following screenshot:



Responsibility Name: XX Test HRMS Manager
Application: Human Resources
Responsibility Key: XXTESTHRMSMGR
Description: XX Test HRMS Manager

Effective Dates
From: 01-JAN-2012
To:

Available From
● Oracle Applications
○ Oracle Self Service Web Applications
○ Oracle Mobile Applications

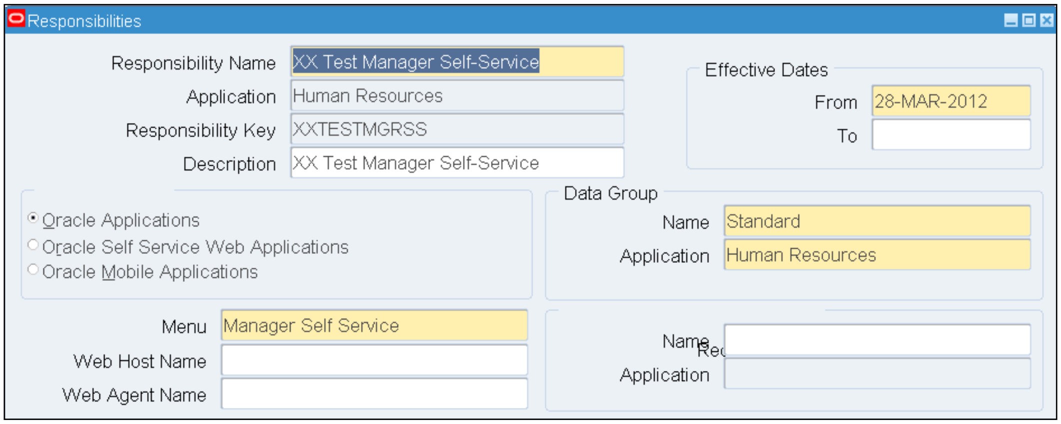
Data Group
Name: Standard
Application: Human Resources

Menu: GLB HRMS Navigator
Web Host Name:
Web Agent Name:

Request Group
Name: HR Reports and Processes
Application: Human Resources

6. Now create a test manager self-service responsibility.
7. Click on the new record icon.
8. Enter the required data as given in the following table:

Item name	Item value
Responsibility Name	`XX Test Manager Self-Service`
Application	`Human Resources`
Responsibility Key	`XXTESTMGRSS`
Description	`XX Test Manager Self-Service`
Data Group	
Name	`Standard`
Application	`Human Resources`
Menu	`Manager Self Service`
Request Group	
Name	



Responsibility Name: XX Test Manager Self-Service
Application: Human Resources
Responsibility Key: XXTESTMGRSS
Description: XX Test Manager Self-Service

Effective Dates
From: 28-MAR-2012
To:

● Oracle Applications
○ Oracle Self Service Web Applications
○ Oracle Mobile Applications

Data Group
Name: Standard
Application: Human Resources

Menu: Manager Self Service
Web Host Name:
Web Agent Name:

Req
Name:
Application:

9. Click on the **Save** button in the toolbar (or press *Ctrl* + *S*) to save the record.
10. Exit the form.

How it works...

We have now created a new responsibility where we can access the HRMS screens.

Creating a new user

We are going to create a user called XXUSER that we will use throughout the chapter for our personalizations. We will add the responsibilities, that we will use throughout the chapter. Also, the user will be assigned an employee record so that when we access the manager self-service screens, there is relevant data available for our personalizations that we are going to create.

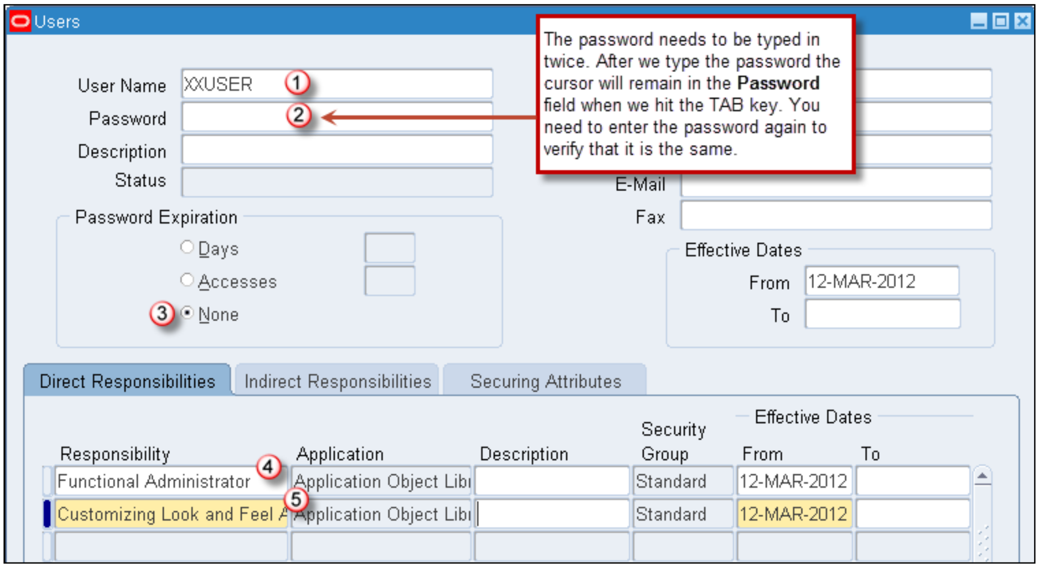
How to do it...

To create the user, perform the following steps:

1. Log in to Oracle and select the **System Administrator** responsibility.
2. Navigate to **Security** | **User** | **Define**.
3. Enter `XXUSER` in the **User Name** field.
4. Enter a password in the **Password** field and press the *Tab* key.
5. Enter the password again and press the *Tab* key.
6. Set **Password Expiration** to **None**.

7. Navigate to the **Direct Responsibilities** tab.
8. Add the following responsibilities:
    - Application Developer
    - Functional Administrator
    - Manager Self-Service
    - System Administrator
    - XX Test HRMS Manager
    - XX Test Manager Self-Service

   The **Users** screen should now appear as shown in the following screenshot:



Users

The password needs to be typed in twice. After we type the password the cursor will remain in the **Password** field when we hit the TAB key. You need to enter the password again to verify that it is the same.

User Name: XXUSER
Password:
Description:
Status:

Password Expiration
○ Days
○ Accesses
● None

E-Mail:
Fax:

Effective Dates
From: 12-MAR-2012
To:

Direct Responsibilities | Indirect Responsibilities | Securing Attributes

				Effective Dates	
Responsibility	Application	Description	Security Group	From	To
Functional Administrator	Application Object Libı		Standard	12-MAR-2012	
Customizing Look and Feel A	Application Object Libı		Standard	12-MAR-2012	

9. Save the form.

How it works...

We have now created our user called `XXUSER`. The user has access to the responsibilities, which we are going to need throughout the chapter.

Creating an employee record

Now we are going to create an employee called **Employee Manager**, which we will use throughout the chapter for our personalizations. As suggested by the name, this employee will be the manager of other employees.

How to do it...

To create the employee, perform the following steps:

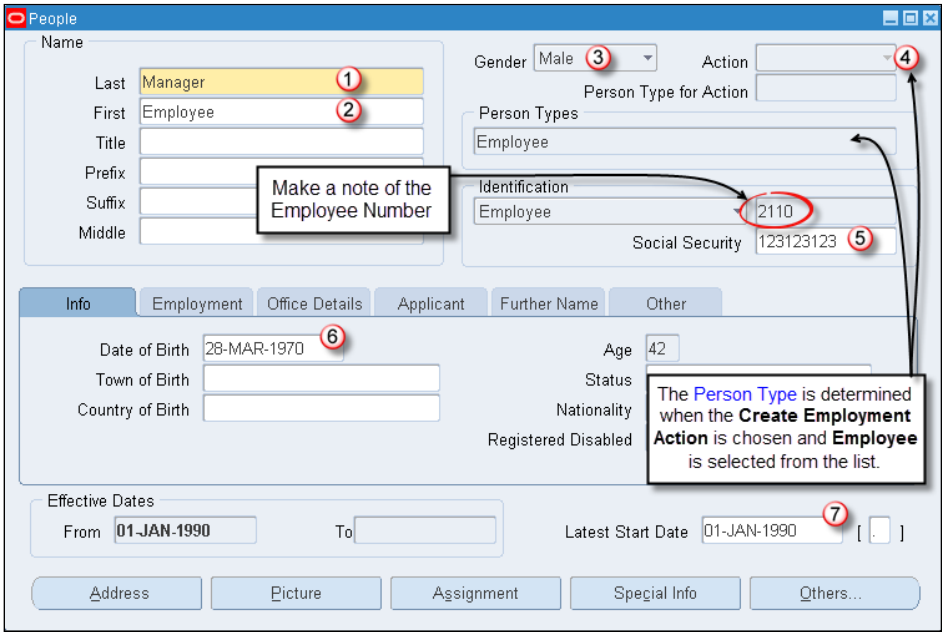
1. Log in to Oracle with the `XX Test HRMS Manager` responsibility.
2. Navigate to **People** | **Enter and Maintain**.
3. When the **Find Person** screen opens, click on the **New** button.
4. Enter the required data as given in the following table:

Item name	Item value
Last	`Manager`
First	`Employee`
Gender	**Male** (or **Female**)
Action	**Create Employment**
Person Type	`Employee`
Social Security	`123123123`
Date of Birth	`28-MAR-1970`

5. Update the **Latest Start Date** field to `01-JAN-1990`.
6. Save the record, and when prompted, click on the **Correction** button.
7. Click on **OK**, when prompted with the message **The original hire date will be updated to be the same as the start date**.
8. Make a note of the employee number that has been automatically generated.

> If the employee number has not been automatically generated, type one manually and save the record as a **Correction**.

The data entered should look similar to the following screenshot:



Last: Manager

First: Employee

Gender: Male

Person Types: Employee

Identification: Employee 2110

Social Security: 123123123

Date of Birth: 28-MAR-1970

Age: 42

Effective Dates From: 01-JAN-1990

Latest Start Date: 01-JAN-1990

Make a note of the Employee Number

The Person Type is determined when the **Create Employment Action** is chosen and **Employee** is selected from the list.

How it works...

We have now created an employee record that we will configure as the manager of other employees. We need to do this so that the self-service screens we are going to personalize will have some data returned in them and we can complete some of the upcoming recipes.

Assigning a manager

We are now going to update the assignment record of an existing employee so that the employee we have just created is the manager of an existing hierarchy and when we perform some personalizations, there is some data returned in the forms we are going to personalize.

How to do it...

To assign a manger, perform the following steps:

1. In the **Enter and Maintain person** screen, click on the **Find Person** (torch) icon in the toolbar.
2. Enter `1197` for the employee number.

3. Click on the **Assignment** button.
4. Click on the **Supervisor** tab.
5. Enter our employee number in the **Worker Number** field. (The employee number is `2110` for the employee that was created in the preceding recipe. It may be a different number on the environment you are using.)
6. Tab out of the field and select the **Correction** button, if prompted.
7. The **Name** fields will automatically populate to **Manager**, **Employee**, and so on (our test employee), when we tab out of the **Worker Number** field.
8. Save the form and close it.

How it works...

We have assigned our employee as a manager of another employee. This means that we will then inherit the hierarchy of the employee we became the manager of and hence we will have data in the self-service screens that we are going to personalize.

Attaching an employee to a user record

We are going to associate the employee we have just created to our user called `XXUSER`. When we log in as the `XXUSER` responsibility, the self-service screen will display data on the screens appropriate to the employee record that we have associated with our user. As we have assigned our employee record as the manager of other employees, we will see the data of the employees in the screens we are going to access in manager self-service.

How to do it...

To create the user, perform the following steps:

1. Select the **System Administrator** responsibility.
2. Navigate to **Security** | **User** | **Define**.
3. Press the *F11* key to enter a query.
4. Enter `XXUSER` in the **User Name** field and press *Ctrl* + *F11* to execute the query.
5. In the **Person** field enter `Manager` and `Employee`, and press the *Tab* button.
6. Save the form.

How it works...

We have now associated our employee with our user. Therefore, when we log in as our user `XXUSER`, we will be able to access the self-service screens as a manager of other employees.

Setting the personalization profile options

There are three profile options that allow us to personalize self-service screens. The three profile options are as follows:

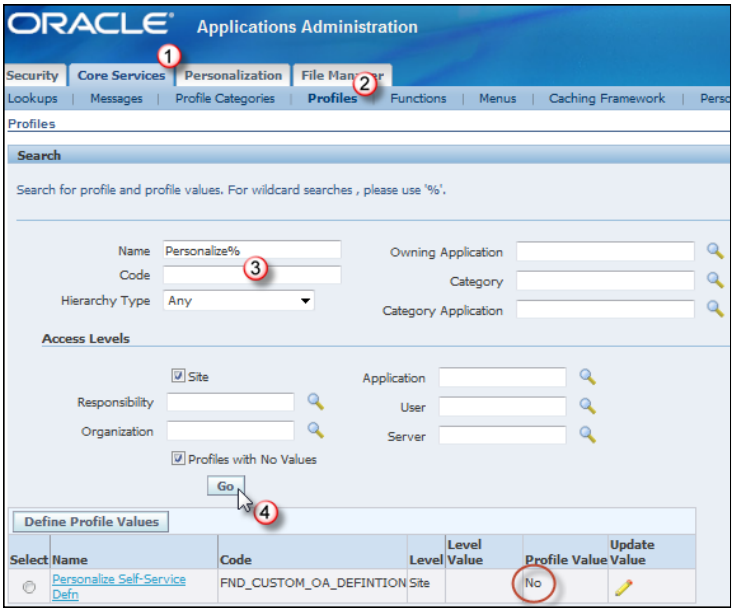
- **Personalize Self-Service Defn**: When set to **Yes**, a **Personalize Page** link will appear at the top of each self-service page
- **FND: Personalization Region Link Enabled**: When set to **Yes**, a **Personalize Region** link will appear on each region of a self-service page
- **FND: Diagnostics**: When set to **Yes**, an **About this Page** link will appear at the bottom of each self-service page

How to do it...

To set the three profile options, perform the following steps:

1. Log in to Oracle with `XXUSER` and select the **Functional Administrator** responsibility.
2. Navigate to **Core Services** | **Profiles**.
3. In the **Name** field of the **Search** screen, type `Personalize%`.
4. Click on the **Go** button.

You will see that the **Personalize Self-Service Defn** profile option value is set to **No** at site level, as shown in the following screenshot (circled):



Profiles

Search

Search for profile and profile values. For wildcard searches , please use '%'.

Name Personalize%

Code

Hierarchy Type Any

Owning Application

Category

Category Application

Access Levels

☑ Site

Responsibility

Organization

Application

User

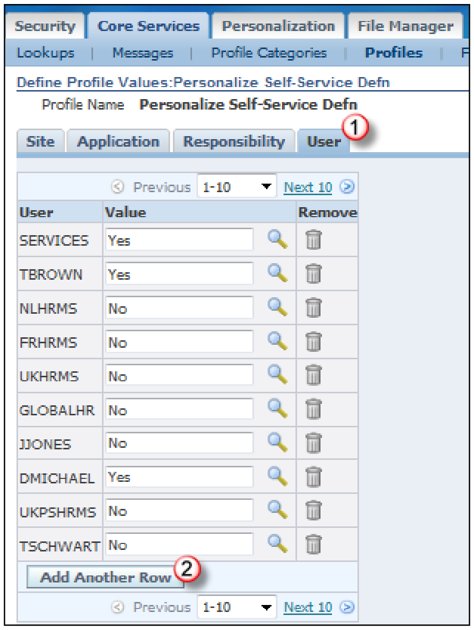
Server

☑ Profiles with No Values

Select	Name	Code	Level	Level Value	Profile Value	Update Value
○	Personalize Self-Service Defn	FND_CUSTOM_OA_DEFINTION	Site		No	✎

We are now going to update the profile option to **Yes** at user level for the `XXUSER user`:

1. Click on the **Update Value** (pencil) icon.
2. Click on the **User** tab and click on the **Add Another Row** button:

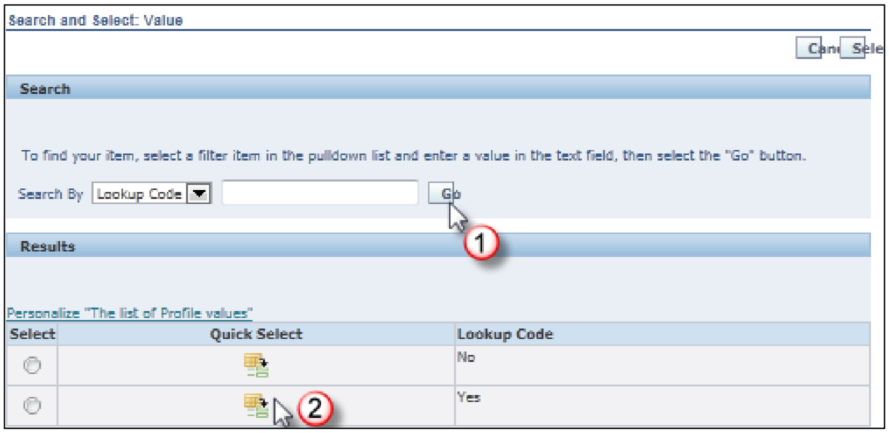


Define Profile Values:Personalize Self-Service Defn

Profile Name **Personalize Self-Service Defn**

User	Value	Remove
SERVICES	Yes	
TBROWN	Yes	
NLHRMS	No	
FRHRMS	No	
UKHRMS	No	
GLOBALHR	No	
JJONES	No	
DMICHAEL	Yes	
UKPSHRMS	No	
TSCHWART	No	

3. In the **User** field, enter the name of our user, `XXUSER`.
4. Navigate to the **Value** field and click on the list icon.
5. Click on the **Go** button.
6. Click on the **Quick Select** icon for the **Yes** value as shown in the following screenshot:



Search and Select: Value

To find your item, select a filter item in the pulldown list and enter a value in the text field, then select the "Go" button.

Search By Lookup Code

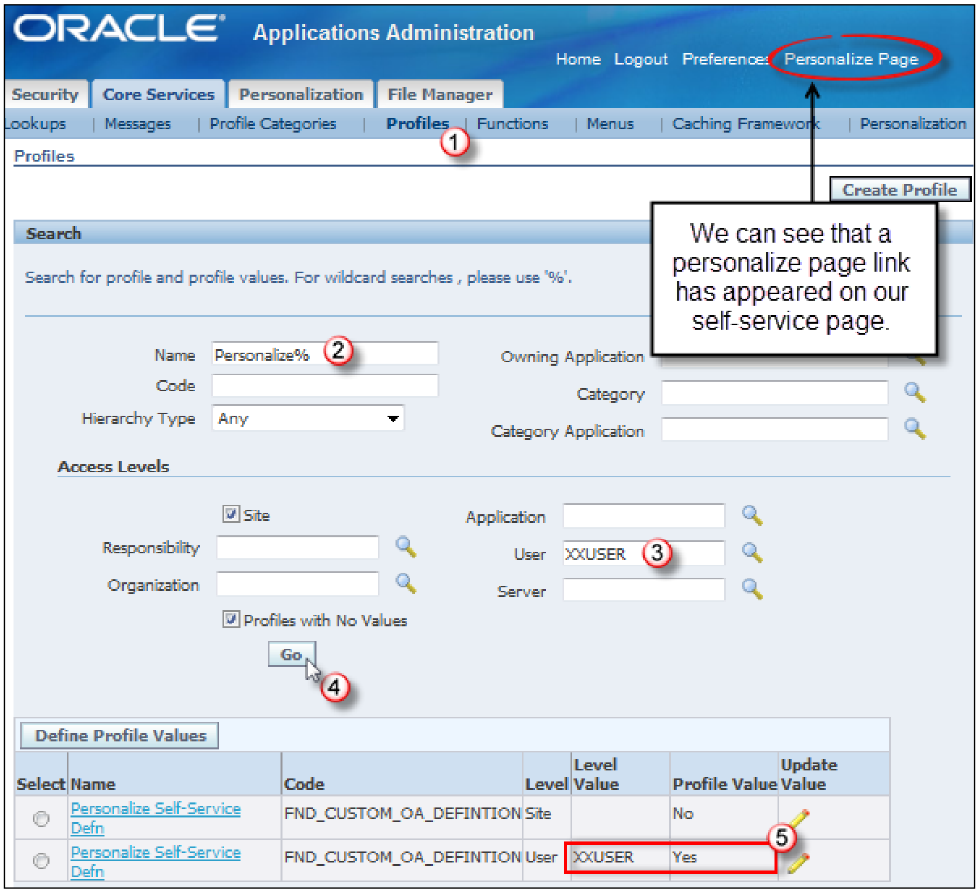
Select	Quick Select	Lookup Code
○		No
○		Yes

7. When you are returned to the **Profiles** screen, click on the **Update** button and wait for the confirmation that the record has been saved.
8. Now navigate to **Core Services** | **Profiles** to return to the profiles screen.

We will now check to see if the value has been set. To do this, perform the following steps:

1. In the **Name** field of the **Search** screen, type `Personalize%`.
2. In the **Access Levels** region, enter `XXUSER` in the **User** field.
3. Click on the **Go** button.

The value we set for our user will be displayed as shown in the following screenshot. We can see that there is a **Personalize Page** link that has now appeared as a result of setting the profile option to **Yes**.



We can see that a personalize page link has appeared on our self-service page.

Search for profile and profile values. For wildcard searches , please use '%'.

Name	Personalize%	Owning Application	
Code		Category	
Hierarchy Type	Any	Category Application	

Access Levels

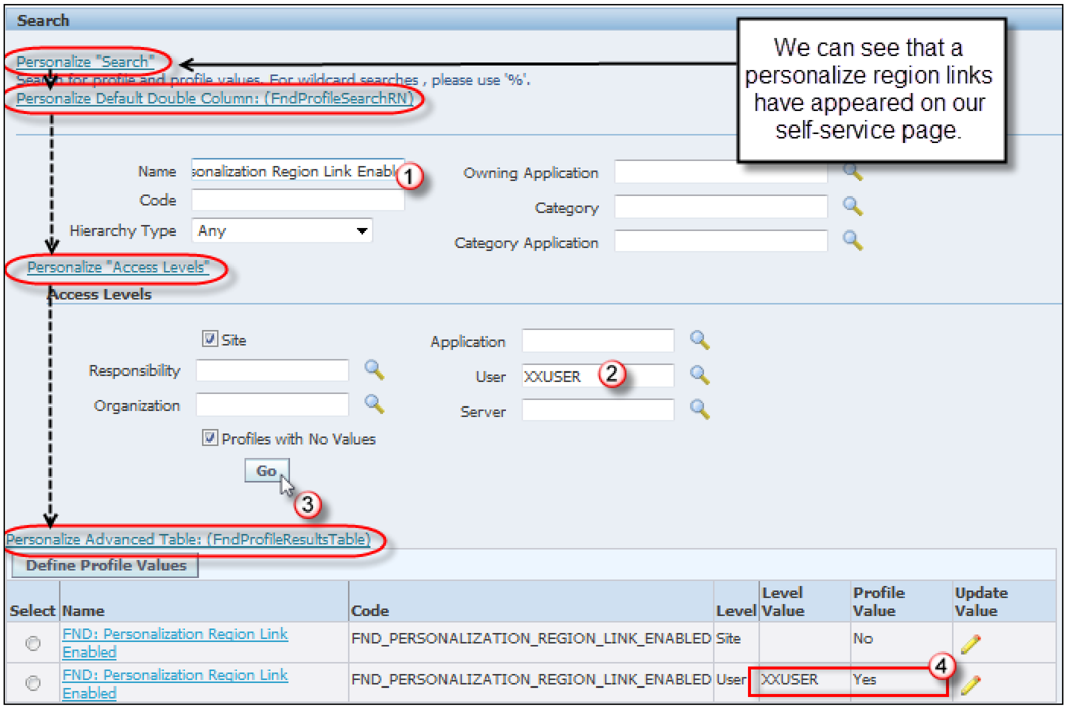
	☑ Site	Application	
Responsibility		User	XXUSER
Organization		Server	
	☑ Profiles with No Values		

Select	Name	Code	Level	Level Value	Profile Value	Update Value
○	Personalize Self-Service Defn	FND_CUSTOM_OA_DEFINTION	Site		No	✎
○	Personalize Self-Service Defn	FND_CUSTOM_OA_DEFINTION	User	XXUSER	Yes	✎

We are now going to set the values for the other two profiles options at user level in the same way.

Repeat these steps but this time set the **FND: Personalization Region Link Enable** profile option to **Yes**.

We can now see that the personalize region links are now displayed in the self-service page, as shown in the following screenshot:



Search

Personalize "Search"

Search for profile and profile values. For wildcard searches , please use '%'.

Personalize Default Double Column: (FndProfileSearchRN)

We can see that a personalize region links have appeared on our self-service page.

Name	sonalization Region Link Enabl	Owning Application	
Code		Category	
Hierarchy Type	Any	Category Application	

Personalize "Access Levels"

Access Levels

☑ Site

Responsibility		Application	
Organization		User	XXUSER
		Server	

☑ Profiles with No Values

Go

Personalize Advanced Table: (FndProfileResultsTable)

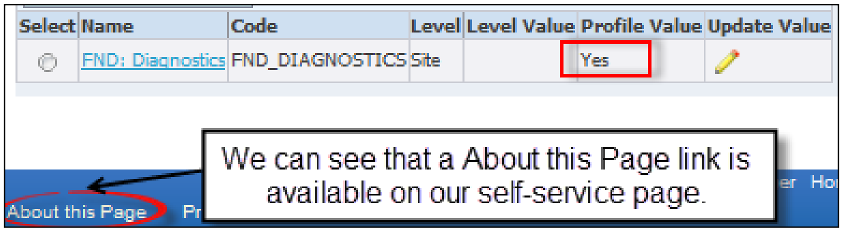
Define Profile Values

Select	Name	Code	Level	Level Value	Profile Value	Update Value
○	FND: Personalization Region Link Enabled	FND_PERSONALIZATION_REGION_LINK_ENABLED	Site		No	✎
○	FND: Personalization Region Link Enabled	FND_PERSONALIZATION_REGION_LINK_ENABLED	User	XXUSER	Yes	✎

We are now going to set the value of the `FND: Diagnostics` profile option at site level in the same way:

1. In the **Name** field of the **Search** region, enter `FND: Diagnostics`.
2. Click on the **Go** button.
3. Click on the **Update Value** (pencil) icon.
4. Under the **Site** tab, set the **Site Value** field to **Yes**.
5. Click the **Update** button and wait for the confirmation that the record has been saved.
6. Now navigate to **Core Services** | **Profiles** to return to the **Profiles** screen.
7. In the **Name** field of the **Search** screen, type `FND: Diagnostics`.
8. Click on the **Go** button.

We can now see that the **About this Page** link is now displayed in the self-service page, as shown in the following screenshot:



Select	Name	Code	Level	Level Value	Profile Value	Update Value
○	FND: Diagnostics	FND_DIAGNOSTICS	Site		Yes	✎

We can see that a About this Page link is available on our self-service page.

How it works...

We have set the profile options that enable links on self-service pages. This will provide access to the required links on the screen so that we can create our personalizations.

Discovering information about a page

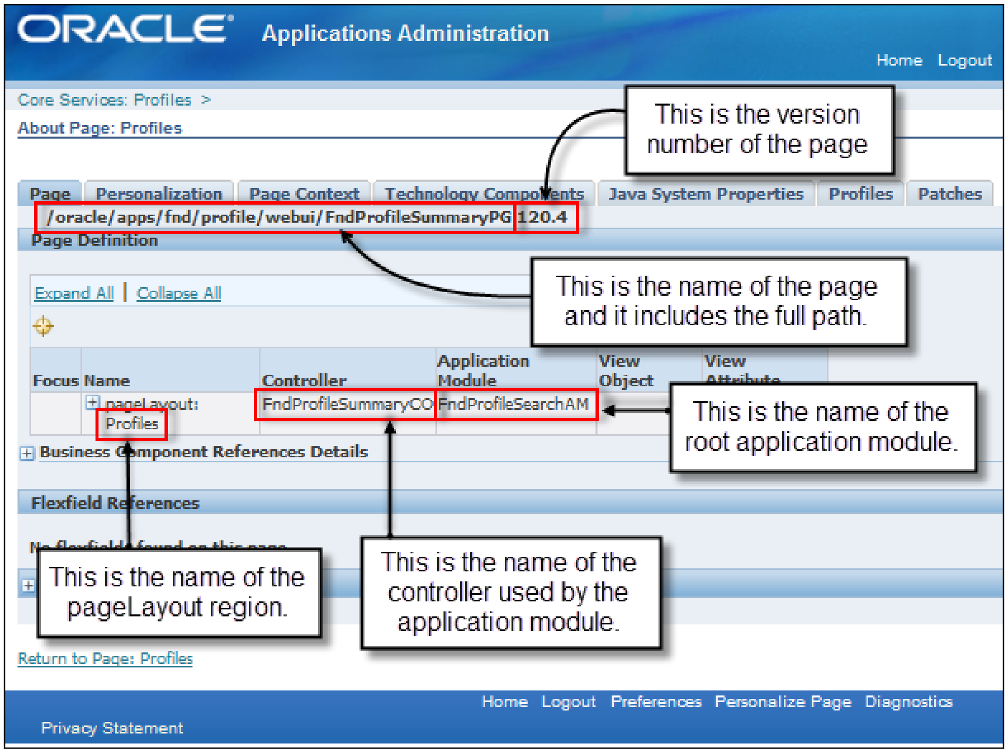
The **About this Page** link is used to find key information about the objects and the structure of the page. It is essential when we want to create personalizations and also if we need to extend a page.

How to do it...

To look at the information available on the **About this Page** link, perform the following steps:

1. Log in to Oracle with the `XXUSER` and select the **Functional Administrator** responsibility.
2. Navigate to **Core Services** | **Profiles**.
3. Scroll to the bottom of the page and select the **About this Page** link.

   If we look at the page, we can identify some key information as shown in the following screenshot:



/oracle/apps/fnd/profile/webui/FndProfileSummaryPG 120.4

Focus	Name	Controller	Application Module	View Object	View Attribute
	pageLayout: Profiles	FndProfileSummaryCO	FndProfileSearchAM		

This is the version number of the page

This is the name of the page and it includes the full path.

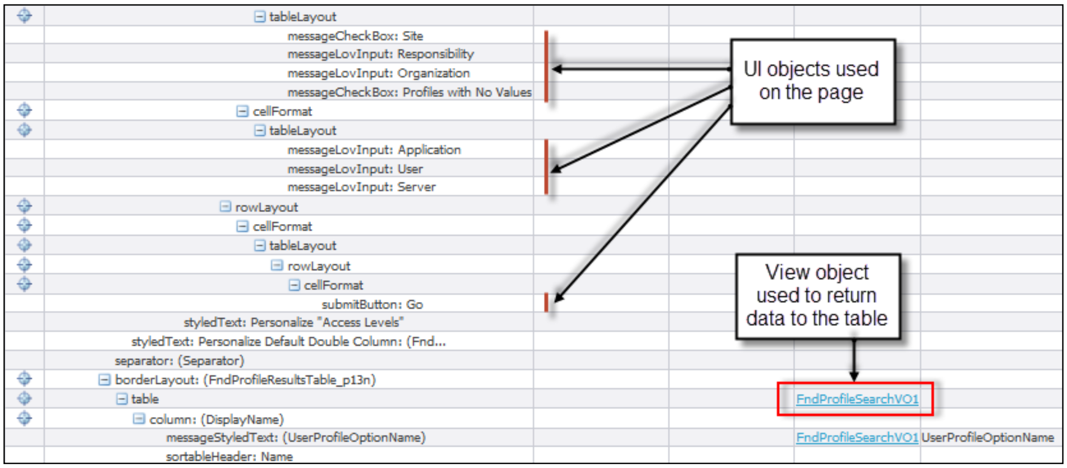
This is the name of the root application module.

This is the name of the pageLayout region.

This is the name of the controller used by the application module.

4. Click on the **Expand All** link and scroll down the page.

We can see the objects that are displayed on the page such as items and buttons. We can also see the view object used for the table, as shown in the following screenshot:



```
            tableLayout
                messageCheckBox: Site
                messageLovInput: Responsibility
                messageLovInput: Organization
                messageCheckBox: Profiles with No Values
          cellFormat
            tableLayout
                messageLovInput: Application
                messageLovInput: User
                messageLovInput: Server
        rowLayout
          cellFormat
            tableLayout
              rowLayout
                cellFormat
                  submitButton: Go
      styledText: Personalize "Access Levels"
    styledText: Personalize Default Double Column: (Fnd...
  separator: (Separator)
  borderLayout: (FndProfileResultsTable_p13n)                        FndProfileSearchVO1
    table                                                            FndProfileSearchVO1
      column: (DisplayName)
        messageStyledText: (UserProfileOptionName)                   FndProfileSearchVO1 UserProfileOptionName
        sortableHeader: Name
```

UI objects used on the page

View object used to return data to the table