



Using the Oracle Rapid Development Kit to Build and Deploy **Oracle Applications Cloud Simplified UIs**

Oracle Applications User Experience

ORACLE
APPLICATIONS
CLOUD

Copyright © 2013, 2016, 2017, Oracle and/or its affiliates. All rights reserved.

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.

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

Intel and Intel Xeon 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, Opteron, the AMD logo, and the AMD Opteron 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

About the Author5

Acknowledgements 6

Introduction7

Who Should Read This eBook.....7

Skills You Need.....7

Software You Need 8

How to Use This eBook 8

Support 9

My Oracle Support10

Documentation Accessibility10

About the Oracle Applications User Experience Team10

Comments and Suggestions 11

1. Rapid Development Kit.....12

AppsCloudUIKit Workspace.....13

Enterprise Oracle ADF architecture13

UIKitCommon14

DemoData.....15

DemoCRM (Sales Cloud)15

DemoFIN15

DemoHCM.....15

DemoMaster.....15

Running the Demo Application.....15

Home Experience: Springboard Page.....16

Unified Global Header18

Filmstrip18

Home Experience Infolet Page19

Sales Cloud Example Pages.....20

Human Capital Management Example Pages22

Accounting Example Pages24

Building With the Rapid Development Kit26

Build the Foundation26

Build a Single Work Area26

Build an Application.....26

Leverage the Demo Application.....26

2. Building a Local Application.....27

Design a Simple Work Area27

Use Design Patterns27

Use the Wireframe Template27

Build a Simple Work Area28

Set Up a Workspace28

Build the Data Layer	29
Build the User Interface	31
Run the Work Area.....	33
A Simulated Integration	33
Call REST Web Services.....	35
3. Deploying to the Oracle Java Cloud	36
Cloud Services	36
Set Up Database Cloud Service.....	36
Deploy to the Oracle Cloud	37
Create a Java Cloud Service Connection	38
Change the Connection Info	38
Configure Security	38
Deploy to the Cloud.....	38
Run Your Application in the Cloud	39
4. Integrating with Cloud Applications.....	40
About Cloud Applications Customization	40
Call Out from Oracle Applications Cloud	40
Call Out from the Navigator and Springboard	40
Call Out from a Page	41
Pass Parameters	41
Call Oracle Applications Cloud Web Services	41
Frequently Asked Questions	43
Can I use Eclipse for Oracle ADF development?	43
What if I have used an earlier version of the RDK?	43
Does the RDK use Oracle Alta UI?.....	43
Does the RDK contain Cloud Services?	43
Which Cloud Service should I use?.....	43
Can I use the RDK code samples in my production applications?	43
How much Oracle ADF do I need to know?	43
Where can I find additional resources on good user experience design?	44
Where can I find additional resources on Oracle ADF technology?	44
Where can I find additional resources on Oracle ADF architecture?	44
Where can I find examples of applications built with the RDK?	44

About the Author

Sten E. Vesterli picked up Oracle development as his first job after graduating from the Technical University of Denmark, and hasn't looked back since. He has worked with almost every development tool and server Oracle has produced in the last two decades, including Oracle Application Development Framework (Oracle ADF), Oracle JDeveloper, WebLogic, SQL Developer, Oracle Portal, Oracle BPEL, Collaboration Suite, Oracle Designer, Oracle Forms and Reports, and event Oracle Power Objects.

Today Sten is one of the world's leading experts on Oracle ADF. He has written two other books on Oracle ADF development: [*Oracle ADF Enterprise Application Development – Made Simple*](#) and [*Developing Web Applications with Oracle ADF Essentials*](#).

Sten loves sharing his knowledge: He started with a conference presentation in 1997. Since then he has also given more than 100 presentations at Oracle OpenWorld and at ODTUG, IOUG, UK-OUG, DOAG, DOUG, and other user group conferences around the world. He is a recipient of the ODTUG Best Speaker award, and he has written for the *ODTUG Technical Journal*, *Oracle Scene*, and *Oracle Profit*, and many other publications.

Oracle has recognized Sten's skills as an expert communicator by awarding him the prestigious Oracle ACE Director title. He is also a Fusion User Experience speaker, and he sits on the Oracle Usability Advisory Board and several partner advisory councils.

When not writing books or presenting, Sten teaches, mentors, and leads development projects. In his spare time, Sten enjoys triathlons. He completed his first Ironman in 2012. You can find his blog at <http://www.vesterli.com/>, send email to sten@vesterli.com, and follow him on Twitter: [@stenvesterli](https://twitter.com/stenvesterli).



Acknowledgements

It is great to see the [Oracle Applications User Experience Rapid Development Kit](#) (RDK) becoming such a powerful tool for every Oracle developer who wants to build great-looking user-friendly applications. In years past, Oracle developers like me often built humdrum, everyday applications because we didn't know there was any other way.

I would like to thank Oracle Applications User Experience Group Vice President Jeremy Ashley ([@jrwashley](#)) for his vision and support in bringing Oracle's internal best practice to the wider Oracle developer community. Also, a big thanks to Oracle Applications User Experience Senior Director Misha Vaughan ([@mishavaughan](#)) for shepherding this project through the many hurdles something like this has to overcome before it reaches the developer.

Oracle Applications User Experience Director Ultan O'Broin ([@ultan](#)) is a powerful voice for good user experience everywhere. He tirelessly travels the world spreading the user experience message. Ultan provided valuable guidance for this eBook and the entire RDK.

For the design part of the kit, including the easy-to-use Microsoft PowerPoint template, I want to thank Oracle Applications User Experience Senior Usability Engineer Julian Orr ([@Orr_UX](#)) who brought the necessary usability focus to the project.

Kudos for the code and demo application in the kit goes to Oracle Applications User Experience Senior User Experience Architect Lancy Silveira ([@LancyS](#)) and his team. As a developer, I appreciate the effort that went into producing good, running code that I and everybody else can copy from

Many others in the Oracle Applications User Experience team have contributed, including Principal User Experience Engineer Karen Scipi ([@KarenScipi](#)), who managed the process of getting various drafts worked into two coherent (and free!) useful Oracle Applications User Experience eBooks: this one and [Simplified User Experience Design Patterns for Oracle Applications Cloud Service](#).

Outside Oracle, several of my Oracle ACE Director friends have been part of the discussions that led to RDK. They have used various incarnations of it and provided great feedback while achieving great success for their customers. Thank you to Lonneke Dikmans ([@lonnekedikmans](#)) and Luc Bors ([@lucb](#)) from eProseed, Debra Lilley ([@debralilley](#)) from Certus Solutions, Lucas Jellema ([@lucasjellema](#)) from AMIS, and Basheer Khan ([@bkhan](#)) from Knex Technology.

Introduction

Your users and management have seen the modern, great-looking, user-friendly applications that Oracle demonstrates, and they are asking: “Why can’t our applications look and feel like that?”

Your answer should be a confident: “They can.”

This eBook will show you exactly how to build great applications like the Oracle Applications Cloud Simplified User Interface (UI). But we know you are busy and can’t be bothered to read long books. That’s why this eBook is short and comes as part of a whole Oracle Applications User Experience Rapid Development Kit (RDK), *including complete, running example applications*.

You can use the RDK in many ways. If you want to build a few screens to extend your Oracle Applications Cloud, you can use the UIKitCommon building blocks, including skins and components, to make your application look like the ones that Oracle Applications User Experience builds. If you want to build your own enterprise application, you can copy the example application, strip out the parts you don’t need, maybe modify some of the sample code, and add your own screens. The choices are yours.

Who Should Read This eBook

This eBook is intended for developers who have used Oracle Application Development Framework (Oracle ADF) to build basic applications. This eBook discusses how to build applications that look and feel like Oracle Applications Cloud. This eBook can also be used by Oracle partners, customers, and the wider Oracle ADF developer community.

The information in this eBook is useful:

- If you want to build stand-alone applications to take advantage of the user experience best practices Oracle has developed for the Oracle Applications Cloud
- If you want to build stand-alone PaaS applications intended for deployment to the Oracle Cloud, taking advantage of the user experience best practices Oracle has developed for Oracle Applications Cloud
- If you want to build SaaS applications to supplement Oracle Applications Cloud, offering your users the same experience as in the Oracle Applications Cloud

Skills You Need

In skilled trades (carpenter, mason, and so on), you start as an apprentice and progress to journeyman. Some progress further and become masters. This is a useful frame of reference for programming skills like Oracle ADF development as well.

- The apprentice is a programmer in training. He or she might have extensive skills from other frameworks and languages, but is new to Oracle ADF. The apprentice will know the most commonly used parts of Oracle ADF Business Components and Oracle ADF Faces and have a little project experience. An apprentice normally works under guidance of a master.

If you are an *apprentice*, you can use the RDK as a learning tool. Study the user experience design patterns and the code in the demo application to find out how to implement a good user experience in real life. There might be parts of the kit you don’t understand or features you are not quite able to make work.

- The journeyman is someone who has completed his or her apprenticeship and is fully educated, but not yet a master. The journeyman is familiar with all normal aspects of Oracle ADF, including the use of Oracle ADF libraries and various deployment options. He or she has worked with Oracle ADF on several projects and can work unsupervised, but will normally not supervise apprentices.

If you are a *journeyman*, you can use the RDK to build individual pages that will become part of a larger application. This can either be pages for a larger stand-alone application or pages used to extend Oracle Applications Cloud. You will understand and be able to use all parts of the parts of the UIKitCommon Oracle ADF library and can understand most of the code in the demo application.

- The master is someone with several years of experience with Oracle ADF after becoming a journeyman. The master is familiar with every aspect of Oracle ADF and has used it on many projects. He or she can serve as technical lead and supervises apprentices.

If you are a *master*, you can use the RDK to build whole stand-alone enterprise applications. You will be able to understand every part of the demo application and can take it apart and recombine the parts you need in order to quickly build a full enterprise application with the great user experience of the Oracle Applications Cloud Simplified UI.

Software You Need

In order to use the RDK, you will need Oracle JDeveloper 11.1.1.9.0 or later available from the Oracle Technology Network (OTN) [Oracle JDeveloper pages](#).

- If you want to deploy to the Oracle PaaS Cloud, you will need an active Oracle Java Cloud Service (JCS) and an active Oracle Database Cloud Service.
- If you want to include your application as part of an Oracle SaaS application, you will need an active Oracle Applications Cloud Service that includes the Oracle Java Cloud Service SaaS Extension (JCS-SX) Service.

Visit the Oracle Applications User Experience Rapid Development Kit page (<http://tinyurl.com/paas4saas>) to access links to Oracle Technology Network (OTN) and GitHub where you can download the latest version of the AppsCloudUIKit code and this eBook.

Visit the Oracle Applications User Experience Rapid Development Kit page (<http://tinyurl.com/paas4saas>) to access links to RDK resources. Visit Oracle Technology Network (OTN) (<http://www.oracle.com/technetwork/indexes/samplecode/app-cloud-accelerators-2882163.html>) or GitHub (<https://github.com/oracle/apps-cloud-ui-kit>) where you can download the latest version of the AppsCloudUIKit code and this eBook.

How to Use This eBook

This eBook describes the content of the RDK and explains how you can use it to build your own applications with the same look and feel as the Oracle Applications Cloud simplified user interface (UI).

In addition to this eBook, you will need to download (for free) the companion Oracle Applications User Experience eBook [Simplified User Experience Design Patterns for Oracle Applications Cloud Service](#).

This eBook contains the technical how-to information, while the design patterns eBook contains

user experience design patterns and guidelines that Oracle teams use to design and build simplified user interface applications for the Oracle Applications Cloud portfolio.

The user experience of an application used to be determined either by the application developers or by specially trained user interface designers. When developers specify the user experience, it is going to be easy to implement but not necessarily exactly what users want. On the other hand, a user experience designed by a user interface designer will match users' needs but might be prohibitively expensive to implement.

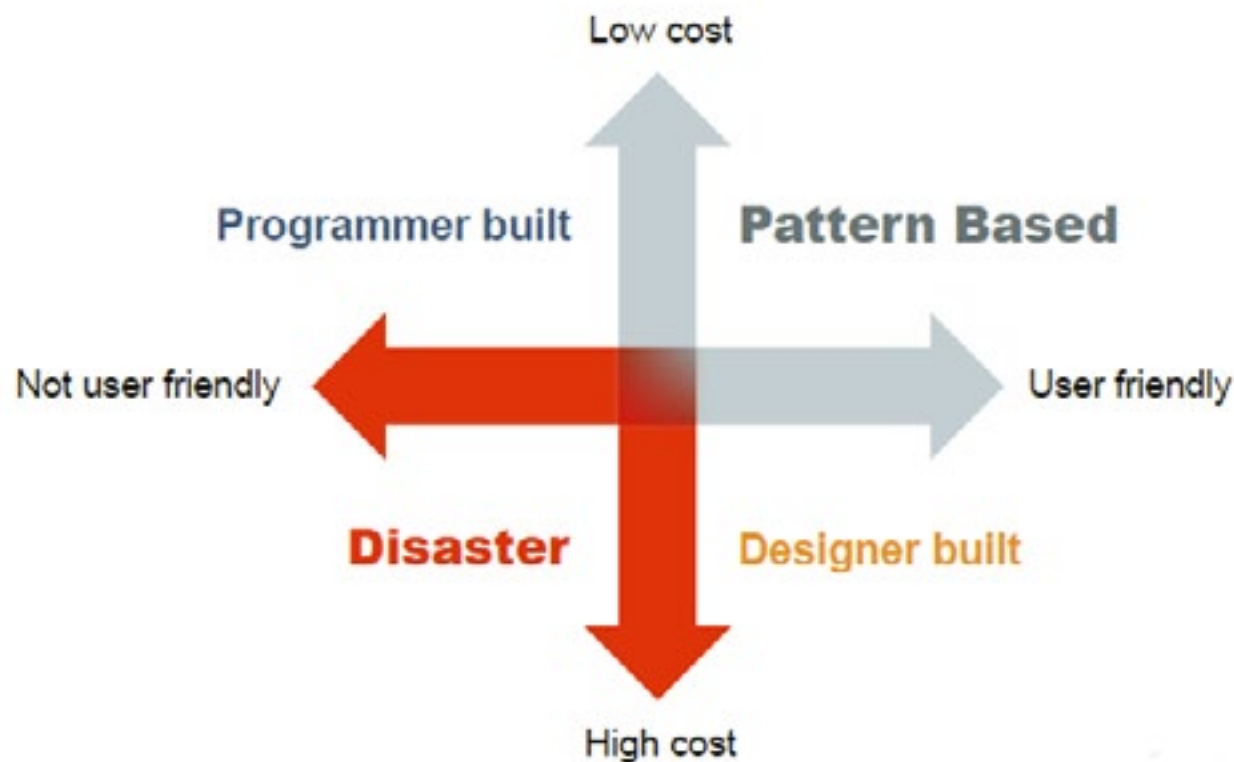


Figure I-1. Benefits of user experience design patterns

The Oracle Applications Cloud User Experience design patterns bring design skills and the latest user experience research together with the technical capabilities of Oracle ADF. This combination allows you to design user-friendly applications that can also be implemented quickly and efficiently.

Depending on your application development needs, you may need to reference content in one or more sections in this eBook.

The application examples in this eBook replicate the simplified user experience of the Oracle Applications Cloud Simplified UI.

Support

If you have questions about how to use this eBook or have questions about the content in this eBook, you can reach us on Twitter: @oauxcloud.

Here are a few resources that you might find useful as you design and build applications:

- Oracle Applications User Experience blogs:
<http://www.oracle.com/webfolder/ux/applications/blog/index.html>
- Oracle Applications User Experience (UX) Highlights:

<http://www.oracle.com/webfolder/ux/applications/successStories/oracleApplicationsCloud.html>

- Oracle Application Developer Framework - Oracle ADF pages:
<http://www.oracle.com/technetwork/developer-tools/adf/overview/index.html>

You can also enhance your user experience knowledge by attending events, such as webinars, workshops, and seminars that are led worldwide by the Oracle Applications User Experience (OAUX) team.

We do not provide support on learning or using Oracle ADF or developer tools. For more information:

- See the Oracle JDeveloper pages for documentation and tutorials:
<http://www.oracle.com/technetwork/developer-tools/jdev/overview/index.html>
- Join the Oracle JDeveloper and Oracle ADF space on the Oracle community for latest news and discussions:
https://community.oracle.com/community/java/java_development_tools/application_development_in_java/jdeveloper_and_adf
- Join the ADF Enterprise Methodology Group to discuss Oracle ADF architecture and best practice:
<https://groups.google.com/forum/#!forum/adf-methodology>
- Contact your Oracle ADF or community representative

My Oracle Support

Oracle customers that have purchased support for Oracle ADF have access to electronic support through My Oracle Support. For information, visit <http://www.oracle.com/pls/topic/lookup?ctx=acc&id=info> or visit <http://www.oracle.com/pls/topic/lookup?ctx=acc&id=trs> if you are hearing impaired.

Note. The RDK contains sample code that is available to anyone who downloads the kit. The sample code in the RDK is bound by the MIT Open Source License.

The RDK is not supported by Oracle Support.

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>.

About the Oracle Applications User Experience Team

User experience is the strategic differentiator for Oracle Applications Cloud. The Oracle Applications User Experience (OAUX) team invests heavily in researching, designing, and delivering a world-class user experience that delights, engages, empowers, and simplifies the lives of those who use Oracle Applications Cloud, no matter where or how they work.

The OAUX team continues to explore emerging technologies for ways to improve the user experience.

Learn more about OAUX outreach events and shared resources at www.oracle.com/usableapps.

Comments and Suggestions

As you use these design patterns and guidelines to build, personalize, extend, and integrate applications, we'd like to hear about your experiences. Share them with us on Twitter: @oauxcloud.

Tell us:

- Is there something that you'd like us to clarify?
- Would you like to see more user interface examples?
- Do you have an idea for a page type, component, or design pattern?
- Does the format of this eBook align well with your practice of building applications?
- What other information would help you build applications productively in the Cloud?
- Do you have a use case or task flow from PaaS or SaaS that you think should be included?

You can also share your experiences with us through the Oracle Applications User Experience blog at <https://blogs.oracle.com/oaux/> or through Twitter @oauxcloud.

1. Rapid Development Kit

The Oracle Applications User Experience [Rapid Development Kit \(RDK\)](#) contains everything you need to build great-looking, user-friendly applications that look and feel like the simplified user interface in the Oracle Applications Cloud portfolio.

Your users will love the application if you *build the right thing, and build the thing right*. The RDK helps you do both.

To build the right thing, you need to involve the users in the design, and follow the well-established science of good user experience. To help you with this, the RDK contains:

- A free companion eBook on design: [Simplified User Experience Design Patterns for Oracle Applications Cloud Service](#).
- An editable [Microsoft PowerPoint wireframing template](#) for designing high-fidelity wireframes that you can share with your users.

To build the thing right, you need to move from the wireframes to running code as quickly and efficiently as possible. To help you do this, the RDK contains:

- Oracle ADF overlays, explaining which Oracle ADF components are used to implement typical user experience design patterns
- Oracle ADF visual elements, templates, and special components that make it easy to implement the design
- This eBook, which gives you detailed how-to technical guidance
- A complete sample application with all the necessary software code to demonstrate how to implement a great-looking user-friendly application

The reusable code artifacts are packaged in an Oracle ADF Library that you can include in your applications.

Visit the [Oracle Applications User Experience Rapid Development Kit page](#) to access links to Oracle Technology Network (OTN) and GitHub where you can download the latest version of the AppsCloudUIKit code and this eBook.

The demo application:

- Demonstrates the modern user experience of the Oracle Applications Cloud Simplified UI
- Provides running code that you can read or even copy and paste into your own applications

For simplicity, the demo application uses non-persistent Java objects as data store, but the example in the next chapter shows how to build with the RDK on top of a regular database.

Note that *example* code is used, just like example code you might find on a blog or elsewhere on the Internet. Notice you didn't pay for it? That's why you can't open a Service Request with Oracle Support about the contents of the RDK.

AppsCloudUIKit Workspace

When you have unzipped the Rapid Development Kit zip file, you will find an AppsCloudUIKit folder. This folder contains an Oracle JDeveloper workspace with the following projects:

- DemoCRM (Sales Cloud)
- DemoData
- DemoFIN
- DemoHCM
- DemoMaster
- UIKitCommon

Additionally, you find the normal `src` directory that belongs at the root level of an Oracle JDeveloper application workspace, and a directory for Oracle ADF libraries (`libs`). Why this structure? Because it is best practice for enterprise Oracle ADF application development.

Enterprise Oracle ADF architecture

A good architecture for any Oracle ADF application is to have a foundation layer, a number of subsystems, and then a master application in which the functionality delivered by each subsystem is combined into a single application.



Figure 1-1. Enterprise Oracle ADF Architecture

The demo application is built in accordance with this architecture. The UIKitCommon as well as the DemoData is the foundation, DemoCRM, DemoFIN, and DemoHCM are subsystems, and DemoMaster is the master application.

The output of the foundation layer is one or more Oracle ADF libraries. The subsystems use these foundation libraries and each produces its own Oracle ADF Library. Finally, the master applica-

tion ties all the libraries together into the finished application.

All Oracle ADF library deployment profiles in the RDK are set to write the adflib files in the `libs` directory, and all projects consume their libraries from there.

For more information about Oracle ADF architecture, see the Oracle Technology Network (OTN) [Oracle ADF Architecture Square page](#) and the book [Oracle ADF Enterprise Application Development – Made Simple](#) (Vesterli, Sten E.).

UIKitCommon

The UIKitCommon project is part of the foundation and contains templates and components that you can use in your own simplified user interface applications. Normally, you will use the finished Oracle ADF Library built from this project, but if you want to see or even change the software code underlying the library, you can find it in this project.

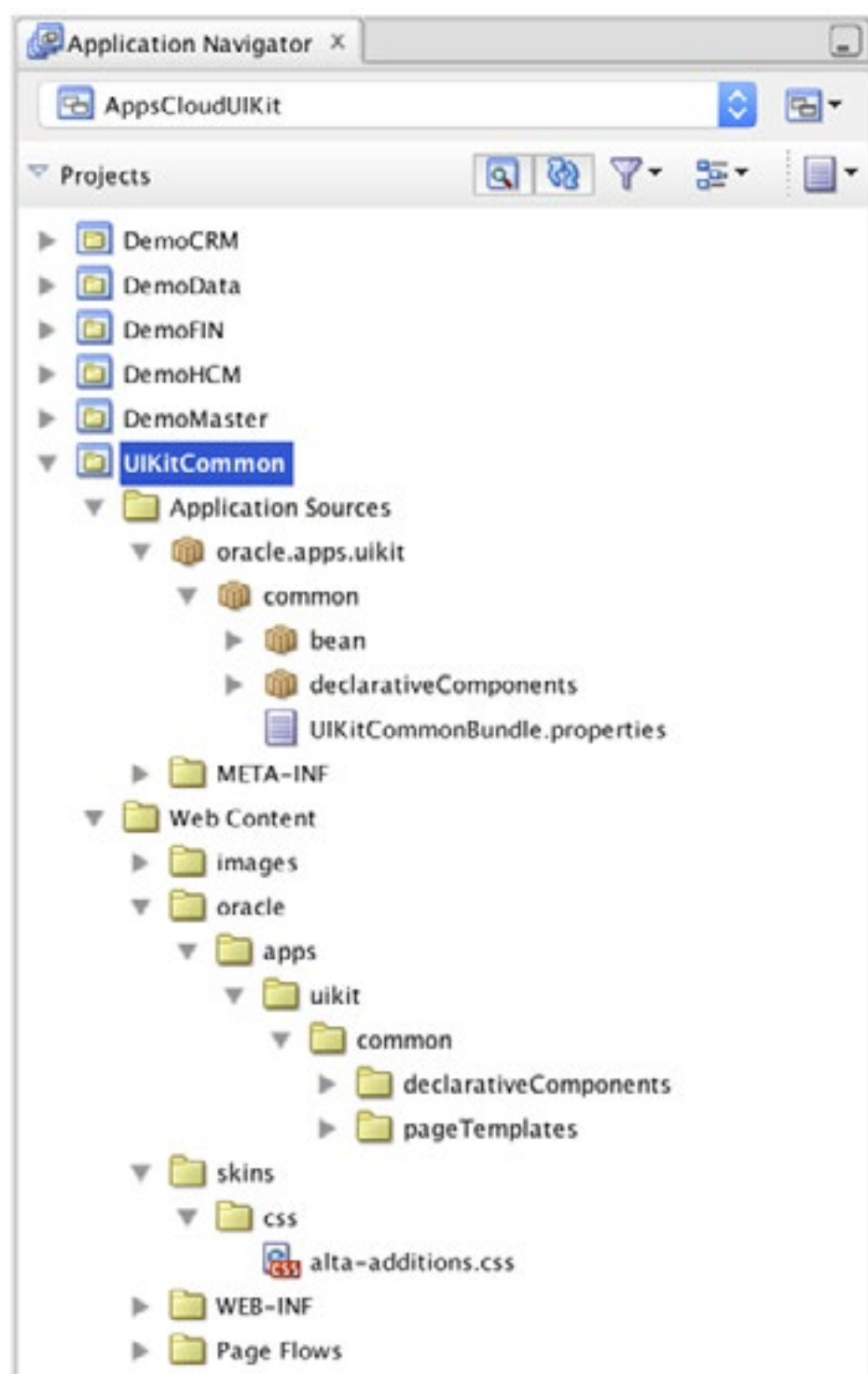


Figure 1-2. Structure for the UIKitCommon project

This project contains the simplified UI skin CSS file, page templates, and some declarative components. It also includes the Java code elements that the templates and declarative components need.

DemoData

The DemoData project is also part of the foundation and contains Java classes that serve as the data source for the demo application. These classes are accessed programmatically from the subsystems of the demo application; they are not published as data controls so they do not show up in the Data Controls palette in Oracle JDeveloper.

DemoCRM (Sales Cloud)

The DemoCRM project is a subsystem and contains the bounded task flow that implements the Contacts work area with a card view/list view landing page and pages for creating and editing contacts. The DemoCRM also contains bounded task flows that implement the Opportunities work area with a list view landing page and pages for creating and editing opportunities.

DemoFIN

The DemoFIN project contains a bounded task flow to implement the Financial Reports work area with a list search landing page and a page with an interactive sunburst visualization.

DemoHCM

The DemoHCM project contains a bounded task flow to implement the Team Performance work area with a simple list view landing page and information tiles on a details page. Additionally, the DemoHCM project includes bounded task flows that implement the My Team work area, which uses a card view/list view landing page.

DemoMaster

The DemoMaster project contains the master page that collects the bounded task flows from the DemoCRM, DemoFIN, and DemoHCM subsystems in accordance with Oracle ADF architecture best practice.

Running the Demo Application

To run the application, open the AppsCloudUIKit workspace in Oracle JDeveloper, open the DemoMaster project, and find the Welcome.jspx page.

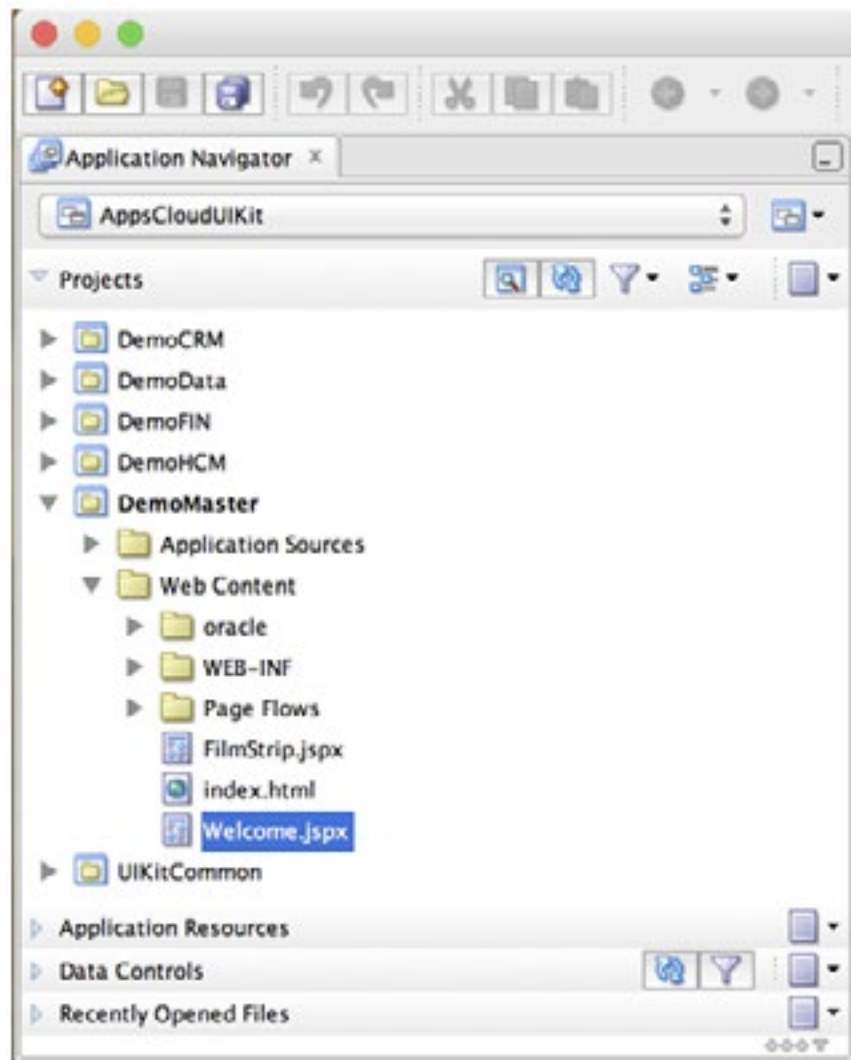


Figure 1-3. The Welcome.jspx page entry

Right-click this page and click Run to start the RDK demo application.

Home Experience: Springboard Page

When the simplified user interface opens for the first time, a splash screen appears over the home experience springboard page. After the splash screen is dismissed, the home experience springboard page appears. (The splash screen does not reappear when the application is opened again.)

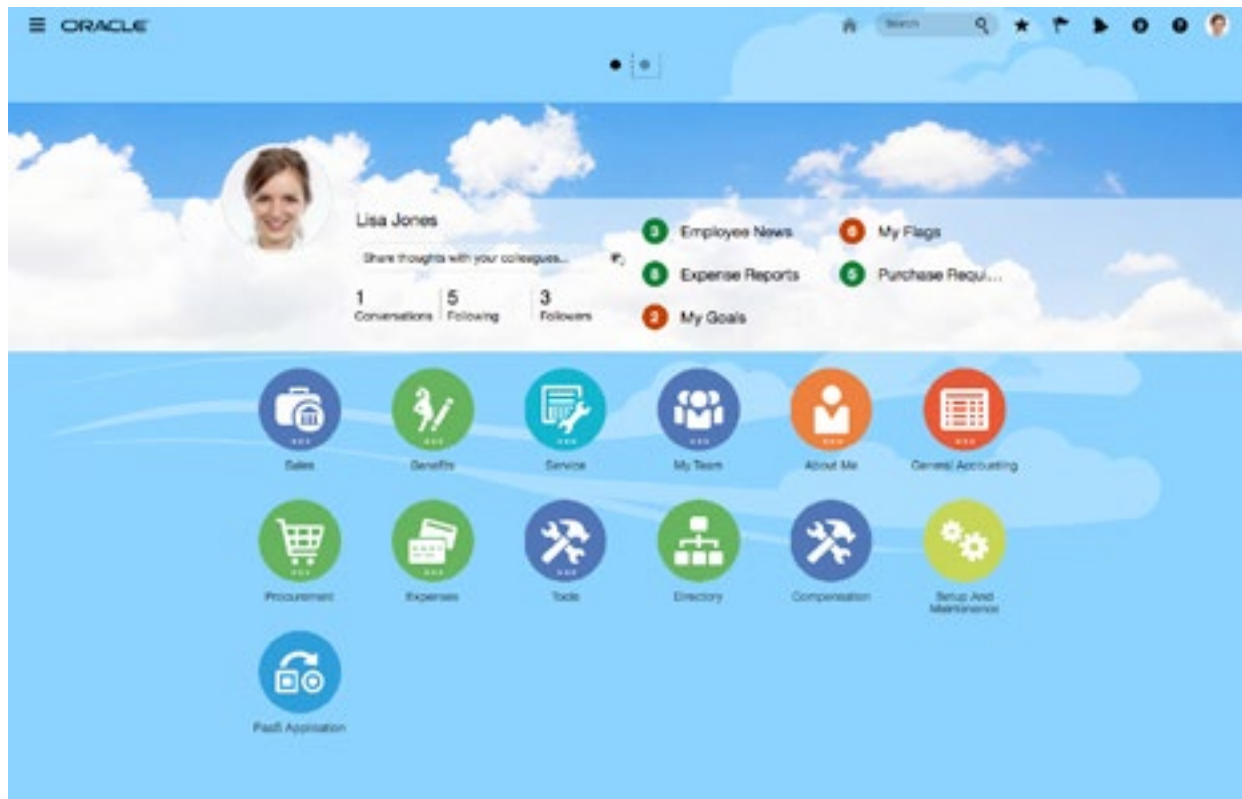


Figure 1-4. Home experience springboard page

The home experience springboard page is an example of the simplified user interface look you should aspire to in your own applications, whether you are building completely stand-alone applications or applications intended for integration into Oracle Applications Cloud. The home experience comprises a springboard page and any number of infolet pages that have been added. The springboard page contains a springboard, which is an iconic grid view of all work areas and dashboards that a user has access to.

If you are building an application, you can designate a home experience page. For example, a home experience page could be a springboard page, an infolet page, or even a work area page.

For more information about home experience pages, see the [Simplified User Experience Design Patterns for Oracle Applications Cloud Service](#) eBook.

The Sales, My Team, and General Accounting are functional grouping icon buttons on the springboard. When clicked, they expand to provide access to the individual work area and dashboard icon buttons.

If you are building an application, and the application contains more than 16 work areas and dashboards, the corresponding icons are grouped on the springboard.

For more information about groupings, see the [Simplified User Experience Design Patterns for Oracle Applications Cloud Service](#) eBook.

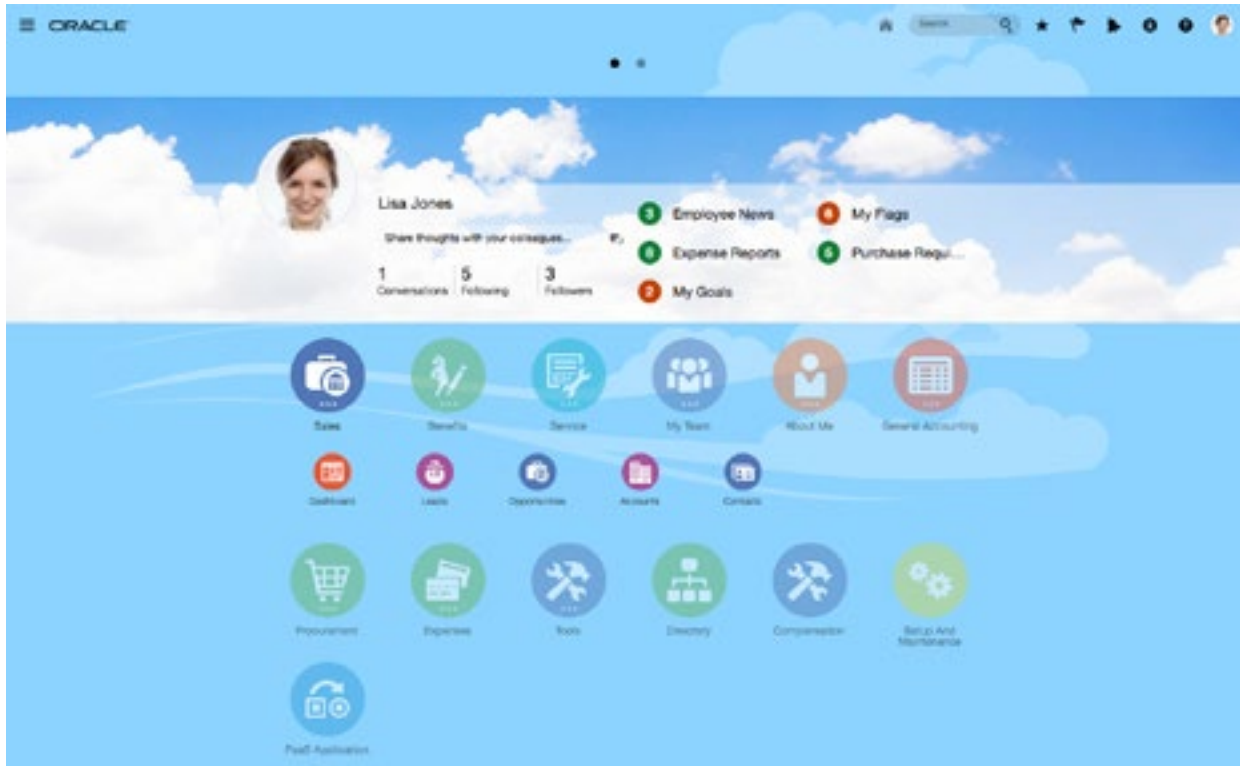


Figure 1-5. The home experience springboard page with an expanded Sales functional grouping icon that shows icon buttons for the work areas and dashboard in this grouping.

Unified Global Header

The Unified Global Header persists across the top of every page in the simplified user interface. When clicked, the Navigator icon opens a menu that provides access to work areas and dashboards.

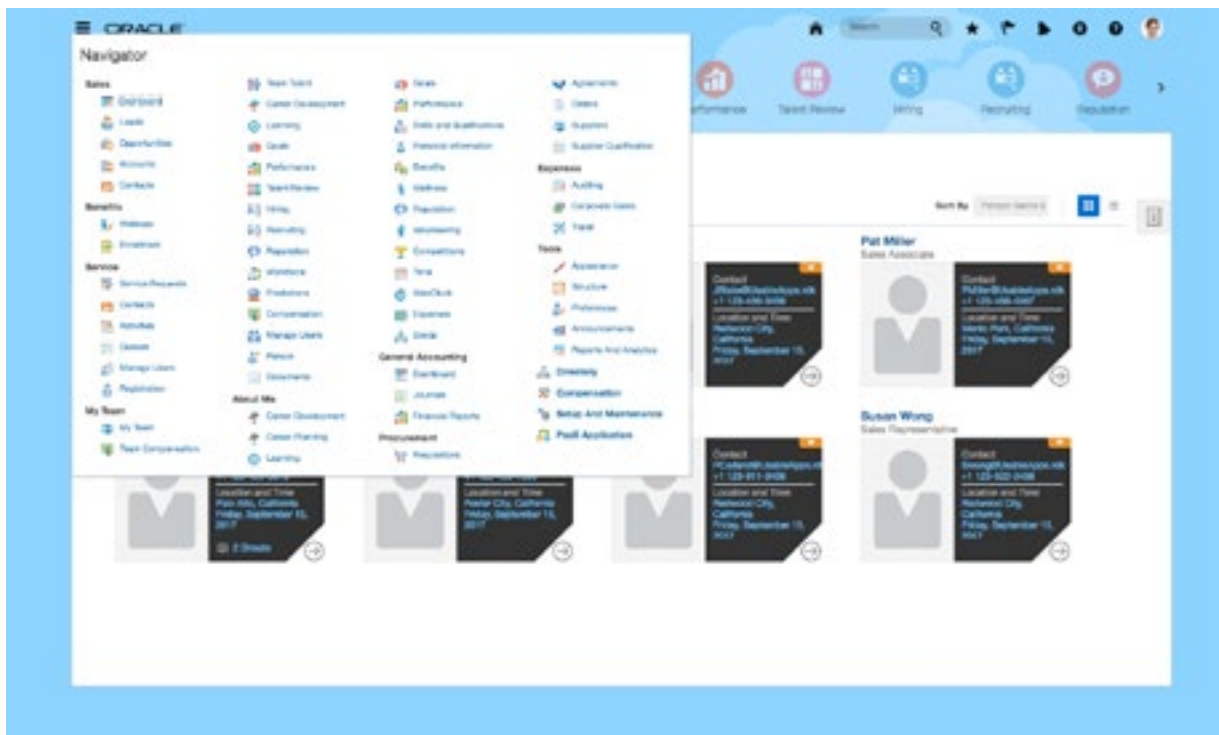


Figure 1-6. The Unified Global Header at the top of the My Team page. The Navigator menu shows the work areas and dashboards that the user has access to.

Filmstrip

Generally, the filmstrip is an iconic view of one grouping that appears below the Unified Global Header on a page. However, when a user has 16 or fewer icon buttons in total, across multiple groupings, all of the icon buttons in the springboard are displayed in one filmstrip, which can accommodate overflow situations when there are more icon buttons to display than fit the width of the page.

Users can click icon buttons in the filmstrip to navigate to work areas or dashboards.

For more information about the filmstrip, see the [Simplified User Experience Design Patterns for Oracle Applications Cloud Service](#) eBook.

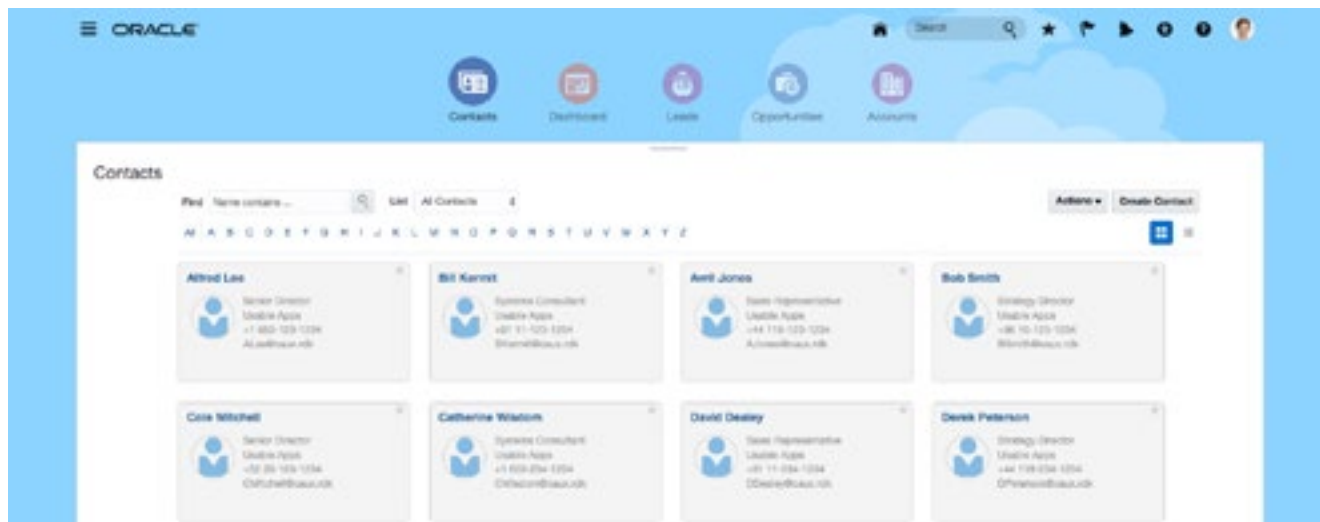


Figure 1-7. Filmstrip showing all icon buttons

Home Experience Infolet Page

If it makes sense in your application, you can add one or more infolet pages to the home experience. Users access infolet pages by clicking the second or subsequent pagination control above the springboard on the home experience springboard page.

An infolet page comprises a collection of infolets. Infolets are self-contained, interactive containers that use up to three views to display high-level, aggregated, essential information that can be quickly consumed at a glance and acted upon as needed.

The demo application contains a sample infolet page that showcases the infolet user experience and the code behind infolet pages.

For more information about the home experience infolet page and designing infolet pages and infolets, see the [Simplified User Experience Design Patterns for Oracle Applications Cloud Service](#) eBook.



Figure 1-8. Home experience infolet page. Users can navigate to the home experience springboard page by clicking the first pagination control.

Sales Cloud Example Pages

Click the Sales functional grouping icon to see a number of Sales work areas. The Opportunities and Contacts work areas are implemented in the demo application. The Opportunities work area landing page shows a simple list view and the work area shows an example of inline transactional pages (create and edit pages).

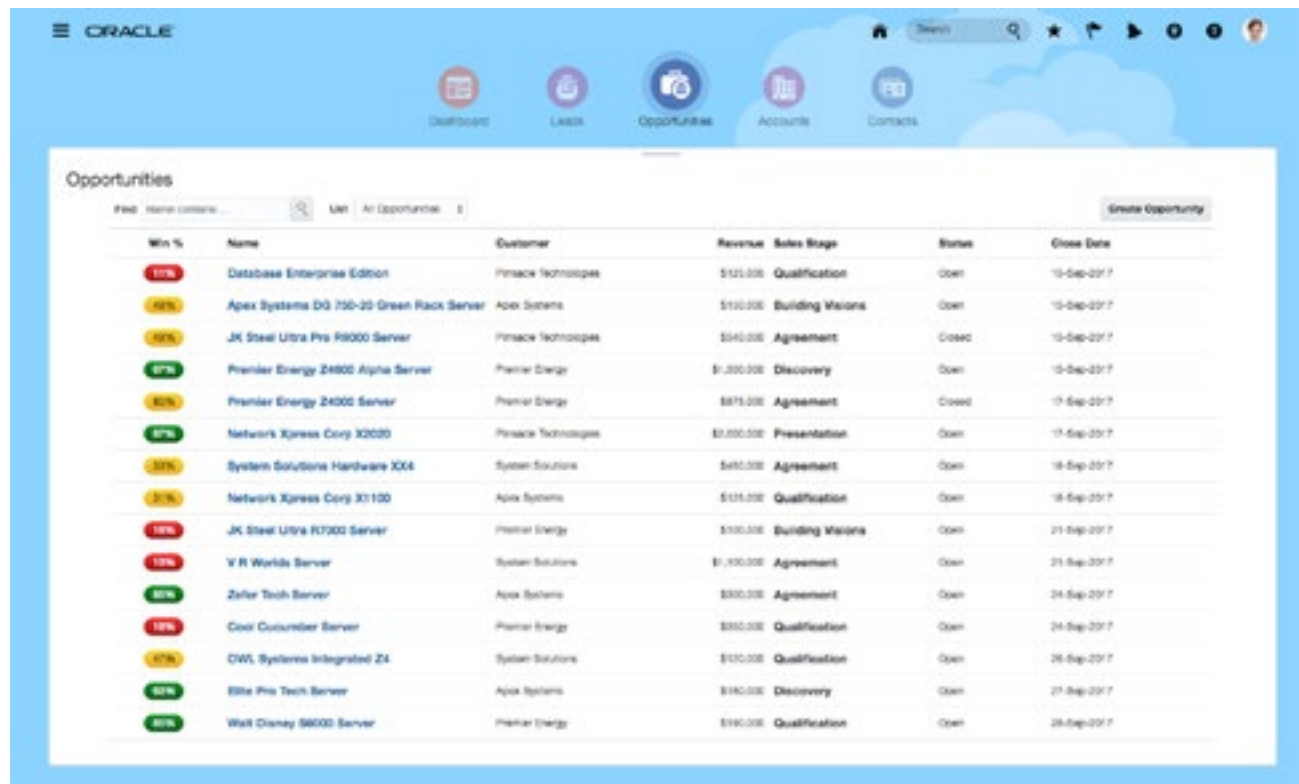


Figure 1-9. Opportunities list view landing page

Card view is commonly coupled with list view. Card view/list view display key metrics or summary information about a single or multiple objects, along with related detail about these objects in

two different views that the user can toggle between.

Card view shows richer detail and is best used when the default number of records is fewer. List view is optimized for viewing greater number of records simultaneously.

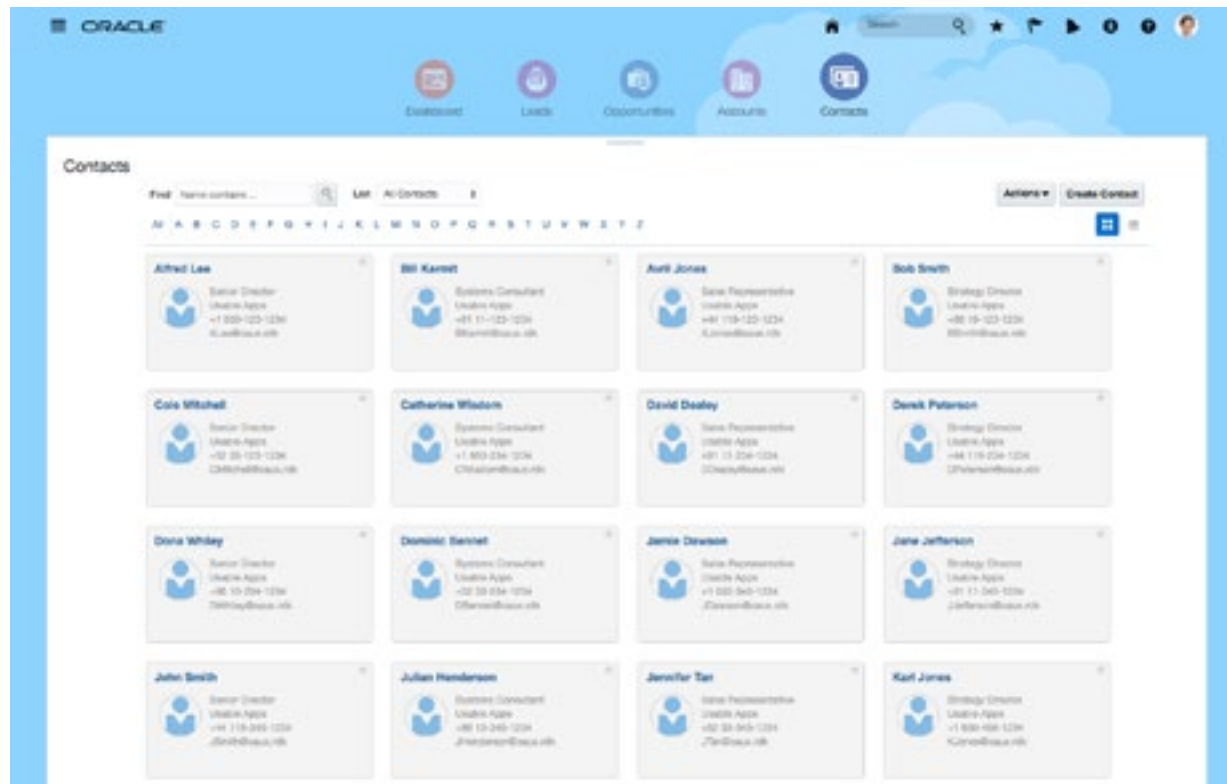


Figure 1-10. Contacts card view/list view landing page with view toggle buttons for switching between the card view and list view

For guidance on using a card view/list view landing page and design patterns, see the [Simplified User Experience Design Patterns for Oracle Applications Cloud Service](#) eBook.

The Opportunities and Contacts work areas demonstrate the following features:

- Card view/list view toggle buttons
- Favorites star icon
- Filter control: Show all or only favorites
- Search by name
- Action menu
- Multiple selection for an action (Select Contacts to Export)
- Create page
- Edit page

Create and edit pages demonstrate how data should be entered and edited in a simplified UI application. These pages are used for records with many attributes. Editing simple records can be done using inline editing in a table.

For guidance about when to use an edit in a popup pattern versus when to use an edit inline pattern, see the [Simplified User Experience Design Patterns for Oracle Applications Cloud Service](#) eBook.

Note that the demo application does not actually export any contacts, and because the demo data store is Java classes, there is no persistence.

Human Capital Management Example Pages

Click the My Team functional grouping icon to see the HCM work areas and dashboard. Click the My Team icon to open the My Team card view/list view work area landing page.

Consider using cards if the user needs a few key attributes or needs to browse multiple objects and their key attributes simultaneously in a summary card format. Cards can include text, links, and graphics. While the back side of a card is optional, it can be used to display more information about the objects without making the user drill into the full record.

The panel drawer is an optional component anchored to the right side of the UI Shell that consists of one or more vertical tabs, each of which opens a panel that contains supplementary content and actions. All content and actions should be relevant in the current context.

For information about using cards and panel drawers, and for design patterns for each, see the [Simplified User Experience Design Patterns for Oracle Applications Cloud Service](#) eBook.

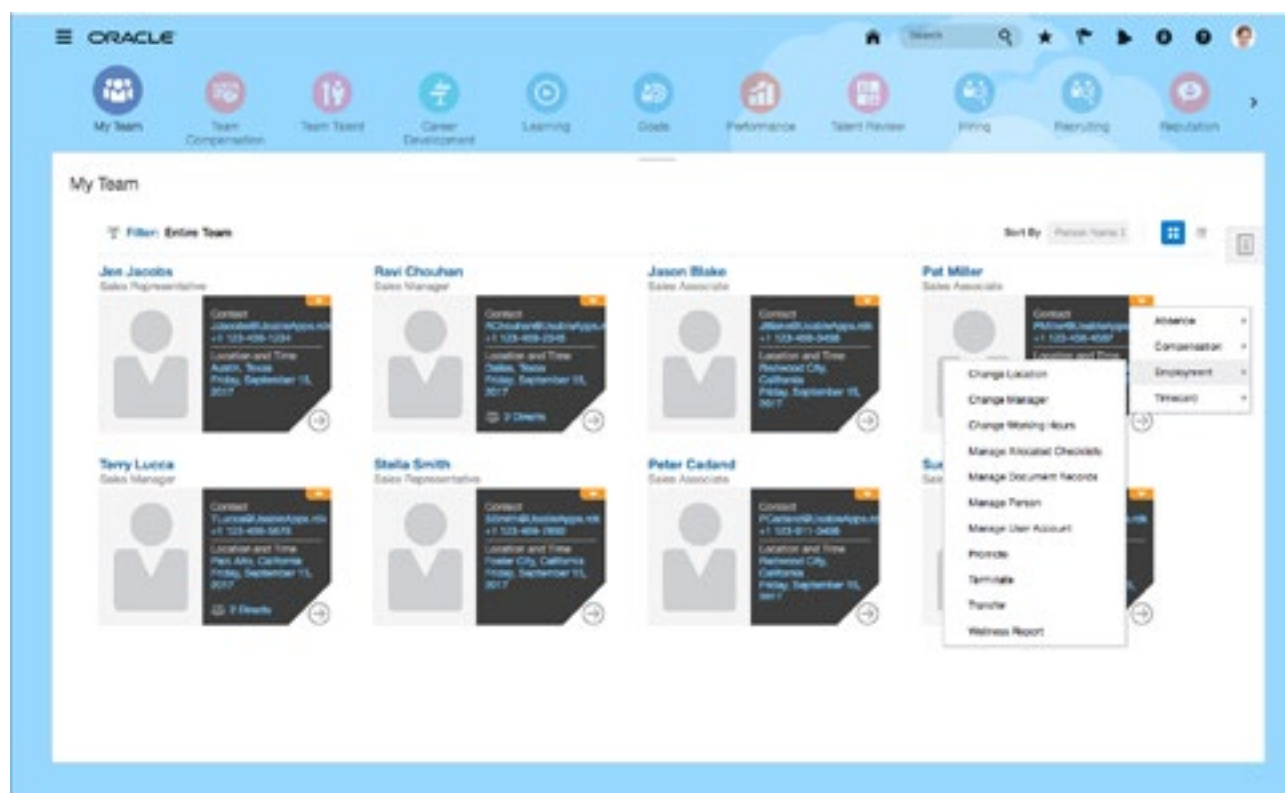


Figure 1-11. My Team card view/list view landing page with view toggle buttons for switching between the card view and list view and a panel drawer. Two cards show optional back sides, and one card shows an open Action menu.

The My Team work area demonstrates the following features:

- Card view/list view toggle buttons
- Filter control
- Cards with an optional back sides that provide additional details
- Action menu
- Panel drawer

To see an example of information tiles, click the Team Performance icon button, and then on the My Team’s Performance Evaluation page, click any of the FY2015 evaluation links.

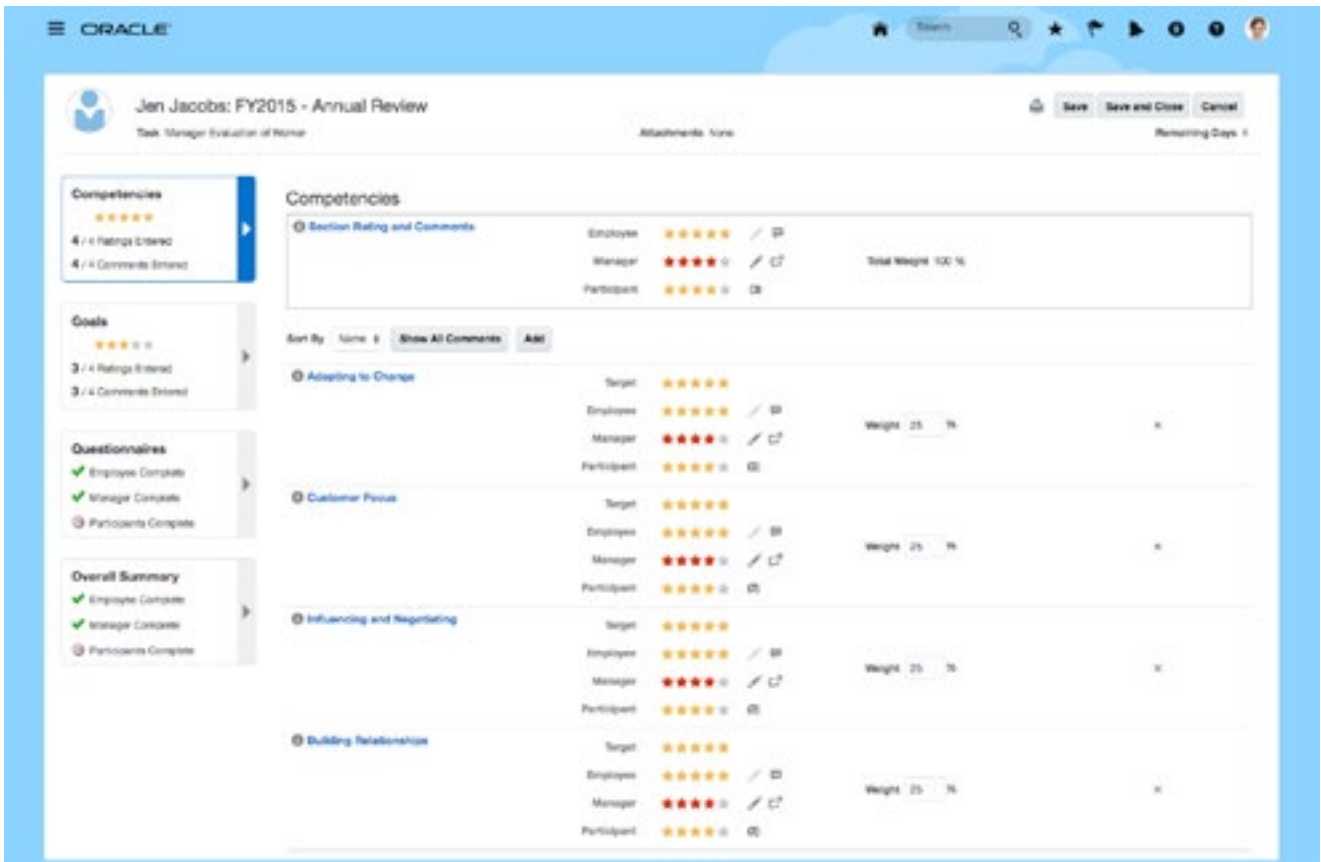


Figure 1-12. Information tiles displayed vertically on an evaluation page

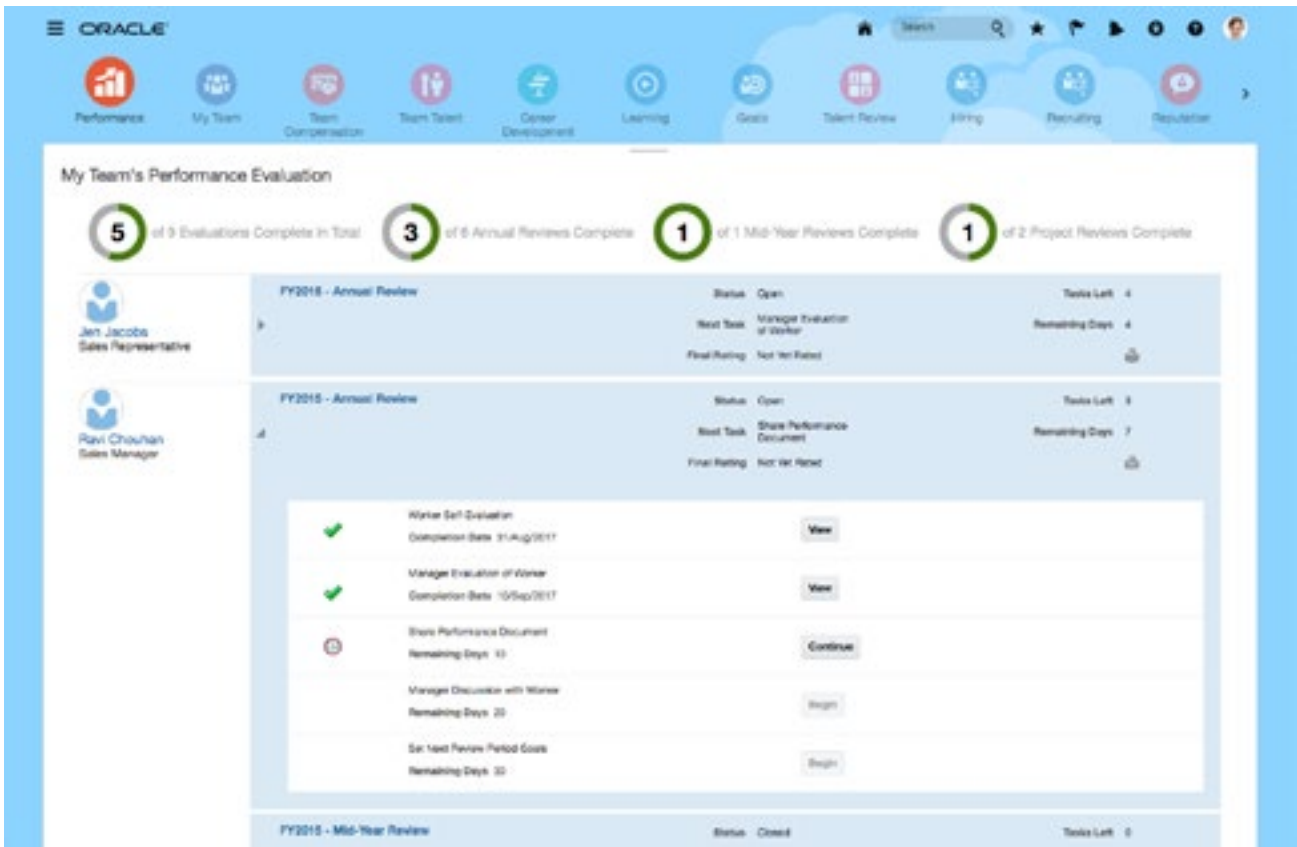


Figure 1-13. An in context, expanded view of the status of annual evaluation tasks for one employee

For information about information tiles, guidance on when to use them, and design patterns, see the [Simplified User Experience Design Patterns for Oracle Applications Cloud Service](#) eBook.

The Team Performance work area demonstrates the following features:

- Display corresponding details about the information in an information tile in the content area
- Display more details about the information on page in context
- Rating gauge control: Select a value by clicking a line of stars

Accounting Example Pages

Click the General Accounting functional grouping icon to see the General Accounting work areas and dashboard. Click the Financial Reports icon to open the Financial Reports work area list search landing page.



Figure 1-14. Financial Reports list search landing page

A list search landing page provides a query that enables user to quickly and effectively retrieve and explore information using autosuggest and filters. This design solution encourages the creation of a simple search page that supports users who need to interact with a single object, for example, a list of people.

For guidance on using a list search landing page and for design patterns, see the [Simplified User Experience Design Patterns for Oracle Applications Cloud Service](#) eBook.

Oracle ADF contains a long list of data visualization components, such as the sunburst chart, and the simplified UI makes extensive use of visualizations. It is hard to give formulaic rules about when to use which visualizations, but you should always consider whether a visualization might make the data more understandable to the user.

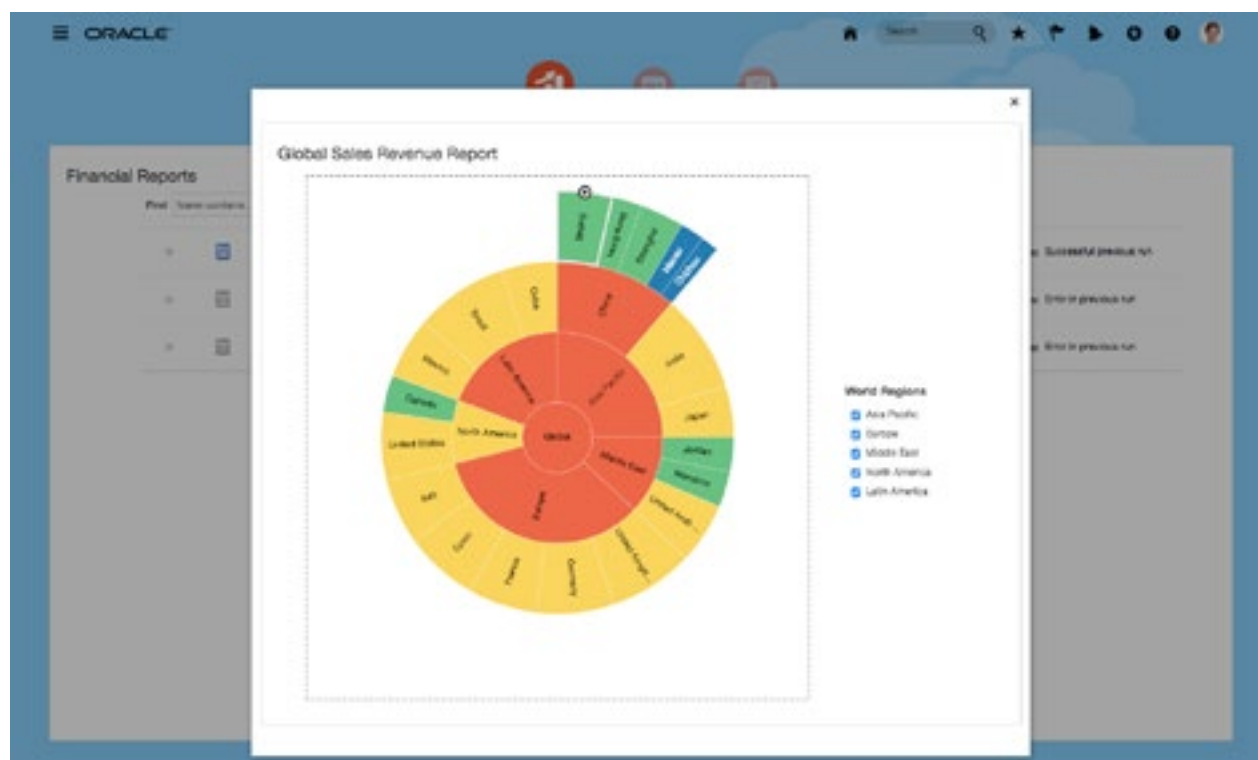


Figure 1-15. Example of a sunburst chart for an example Global Sales Revenue Report

Some examples of useful visualizations:

- For comparing values to each other, you might use a horizontal bar chart, possibly with the color-coded bars to draw attention to the values the user should focus on. Horizontal bar charts are better than vertical for non-time values because there is more room for longer labels to the left of a bar than under it.
- For comparing values to thresholds, you might use status meter gauges. These are very compact and work well in tables where you can have a visualization for every row.
- For showing discrete values over time, you might use a vertical bar chart.
- For showing continuous values over time, you might use a line chart.
- To show trends over time for many records at the same time, you might use the compact spark charts. Place one on every record in a table.
- To show correlation between two values, you might use a scatter (X-Y) chart.
- To show correlation between three values, you might use a bubble chart. This is like a scatter chart, but with the additional dimension visualized as a bubble instead of just a point. The size of the bubble indicates the third value.

The Financial Reports work area demonstrates the following features:

- List search landing page
- Interactive sunburst chart:
 - Double-click to drill into details
 - Hover mouse over outer element to see the plus icon to expand one item
 - Hover mouse over inner element to see the minus icon to collapse one item

- Breadcrumb navigation at the top to navigate up the hierarchy

Building With the Rapid Development Kit

When using the RDK to build your own applications, you can use the UIKitCommon as the foundation and refer to the demo applications to see how various features are implemented.

Build the Foundation

The foundation of an application based on the RDK will consist of the UIKitCommon library as well as any foundation elements you choose to build.

If you are using Oracle ADF Business Components in your application, your foundation layer should include framework extension classes and the entity objects for all relevant tables.

Build a Single Work Area

If you are building only a single work area for inclusion in an Oracle Applications Cloud instance, you should use the architecture described in the Building the Foundation section.

Your work area will be a subsystem containing a model project with your view objects and application module, as well as a view project with your bounded task flows and page fragments. It will include the Oracle ADF libraries from the foundation layer.

Your master application will simply be a wrapper around the subsystem if your application is going to appear in an Oracle Applications Cloud springboard.

Build an Application

If you are using the RDK to build an application that will be used outside of Oracle Applications Cloud, you will be building a foundation and work area subsystem as described in the Building the Foundation section. You will also need to build the home experience springboard page.

You build a springboard page in the master application. Refer to the code in the DemoMaster project to see how to build a springboard page, or use the code in this project as your own master application. You will need to build a filmstrip to support the functional grouping icons and groupings that you add. You may also want to include one or more optional infolet pages.

The RDK demo application contains code that shows how the filmstrip and infolet pages can be implemented.

Leverage the Demo Application

While the practice noted in this section is the normal and recommended enterprise application development approach, you can also simply take the demo application and make it your own: Remove the parts you don't need, modify the pages to match your needs, and add additional pages and functionality.

2. Building a Local Application

This section covers designing, building, and running an application locally on your development workstation.

Design a Simple Work Area

In order to build the right thing, you need understand what your users need, have well thought-out use cases, and sketch the flows that align with those use cases.

For an overview on selecting and modeling use cases, see the Oracle Applications User Experience video [Design the Right Thing & Designing the Thing Right](#).

Use proven user experience design patterns from the [Simplified User Experience Design Patterns for Oracle Applications Cloud Service](#) eBook to build an application your users can understand intuitively. And, if your application is intended as a companion to an Oracle Applications Cloud application, your users will have a consistent user experience.

In taking with your users, let's say, for example, that you discover they are clamoring for an application that displays a list of departments. Fortunately, we already have these in the standard HR demo schema that is delivered with the database.

Use Design Patterns

A landing page is an entry point for a work area that provides users with quick access to information and launch points for tasks within the work area. All work areas contain a landing page. The types of landing pages are:

- Information tiles
- Card view/list view
- List search

For more information about choosing the right landing page and design pattern for your use case, see the [Simplified User Experience Design Patterns for Oracle Applications Cloud Service](#) eBook.

Let's say that based on your users' use case, you find that there is no need for either information tiles or card views, so you choose the list search landing page. The search part of the list search design pattern is optional, and because you know there will be a limited number of departments, you decide not to implement the optional search panel.

In the sample application, you can find another example of a list search landing page by clicking the General Accounting functional grouping icon button on the springboard and then clicking Finance Reports.

Use the Wireframe Template

Use wireframes to document and review designs before you code. The RDK includes a wireframe

template that allows you to document your proposed solution using Microsoft PowerPoint.

The RDK wireframe template uses Microsoft PowerPoint because it is a familiar application to most users. Using Microsoft PowerPoint also reduces the need to cut and paste wireframes from one application to another for the purposes of design communication between the designer and stakeholders.

For an overview on selecting and modeling use cases, see the Oracle Applications User Experience video [Design the Right Thing & Designing the Thing Right](#).

Here is an example use case that contains a requirement for a department list page. The corresponding wireframe (designed using the RDK wireframe template) might look like this:



Figure 2-1. Wireframe example: List of departments

In this case, the use case indicates that the department identifier and name are more important than the manager name and location, so they are set in a larger font. This is an example of the information hierarchy used throughout the simplified UI. Be sure to consider the relative importance of values as you design and build pages.

Build a Simple Work Area

After you have agreement on the designs specified in your wireframes, the next step is to build the design. For this development task, you need Oracle JDeveloper.

Set Up a Workspace

Application development with the RDK is normal Oracle ADF development.

1. Create a normal application workspace in Oracle JDeveloper of type Fusion Web Application (Oracle ADF).
2. Include the UX RDK Oracle ADF library in your application. To do this, create a File Sys-

tem connection to the libs directory of the UX RDK.

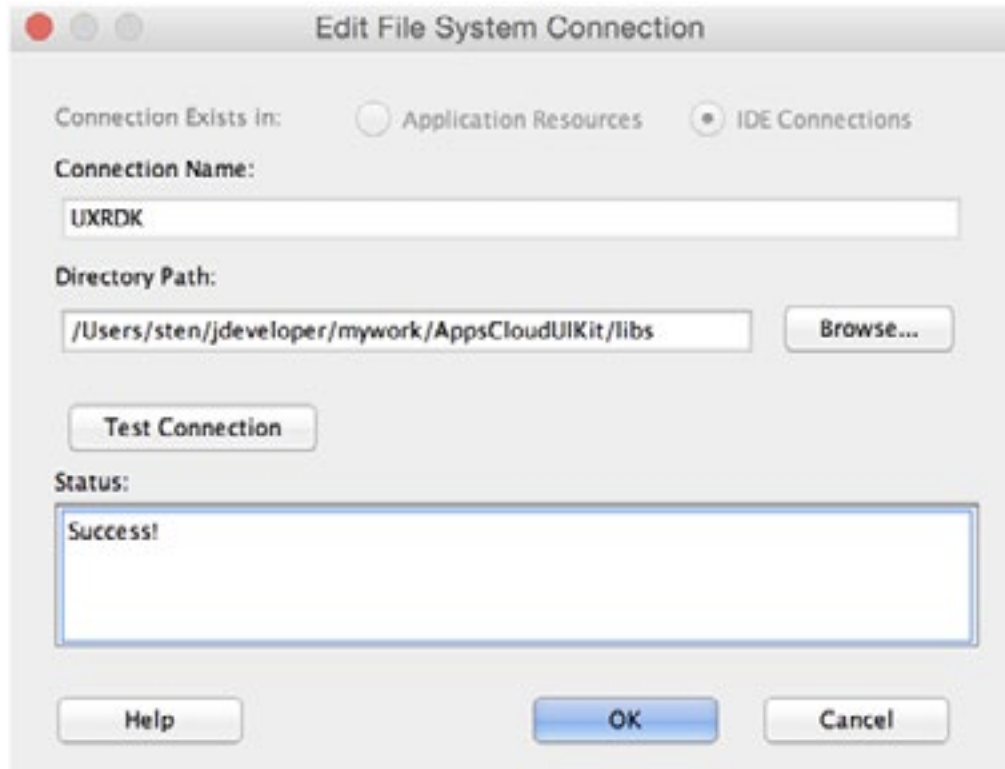


Figure 2-2. A file system connection

3. Select the view project in your own workspace, and then open the File System connection in the Resource Palette.
4. Find and right-click the adflibUIKitCommon.jar file, and then add it to your project.

This Oracle ADF library contains all of the common elements from the UIKitCommon project (templates, components, and so on).

Build the Data Layer

Based on the wireframe, it is clear that we need the following information:

- Department ID
- Department name
- Manager name
- Location

Based on the data model of the HR schema, we find that department ID and name are found in DEPARTMENTS, together with a manager ID and a location ID. Following the foreign keys in the database, we find that we need to join in EMPLOYEES to get the manager name and LOCATIONS to get the city.

Armed with this information:

1. Open the model project of your workspace and create entity objects for DEPARTMENTS,

EMPLOYEES, and LOCATIONS.

2. Create a DepartmentsVO view object that joins the data your need.

It should look something like this:

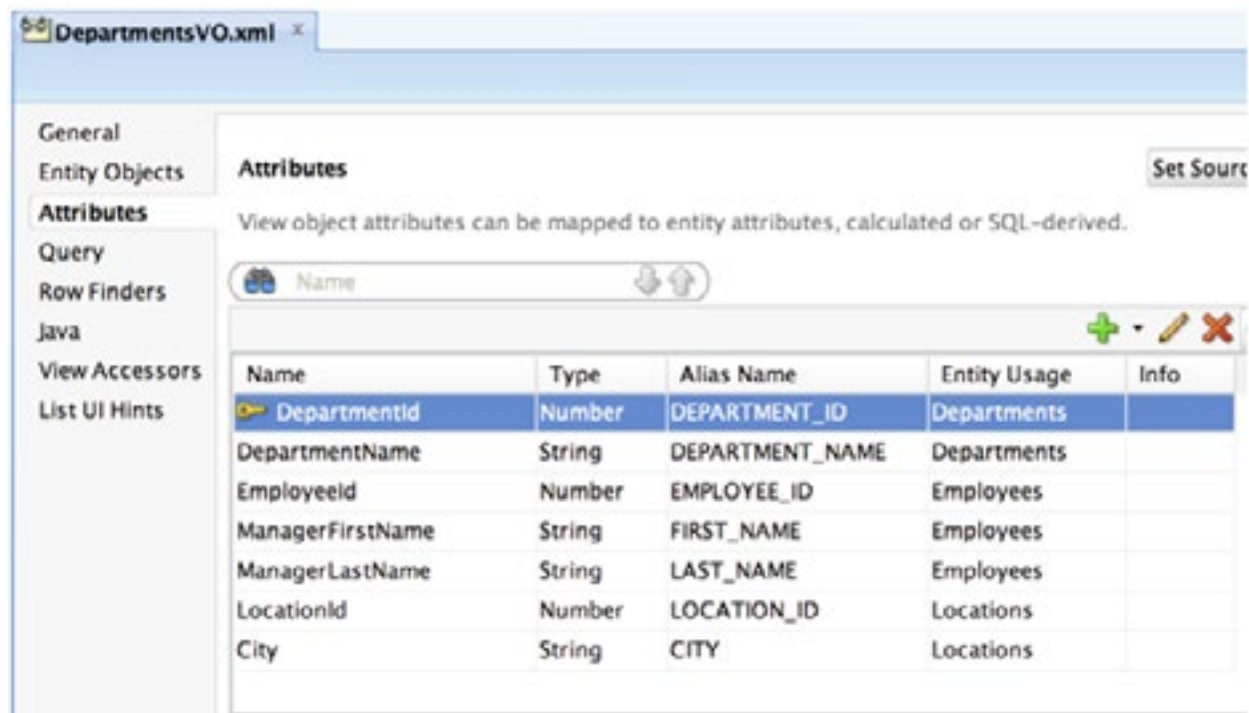


Figure 2-3. A DepartmentsVO view object

Notice that the attributes from Employees have been renamed to ManagerFirstName and ManagerLastName to indicate their meaning in this view object.

3. Create an application module and verify that you see the data that you expect to.

It should look something like this:

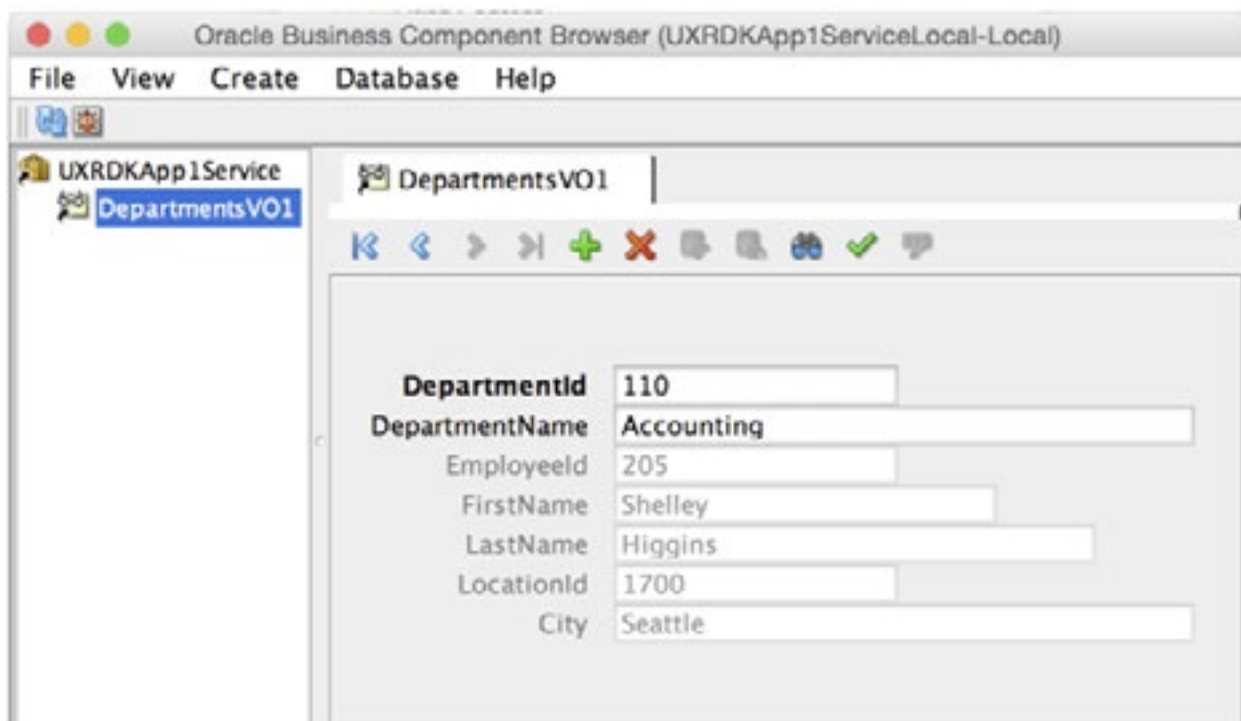


Figure 2-4. An application module example

Build the User Interface

All of the individual landing pages that you build must be built as bounded task flows with page fragments. This simple example work area contains only the landing page, but most of the design patterns involve several different pages that the user can navigate to and from the landing page. Your departments page flow should look like this:



Figure 2-5. Departments page flow

To create the page:

1. Double-click the DepartmentList view activity, and then select to base it on the MainPage-Template.

If you do not see MainPageTemplate in the list, you have not added the UX RDK Oracle ADF library (adflibUIKitCommon.jar) to your view project.

When the page opens, you will see that it contains several facets from the page template.

2. Place all of your content in the appropriately named pageContents facet.
3. Add a page title at the top and an Oracle ADF List below the header.
4. Drop a Panel Group Layout and set Layout to Vertical.
5. Add a standard page header.

The RDK offers an Oracle ADF declarative component that you can use for this. You can find this component in the Component Palette under AppsCloudUIKitDCs.

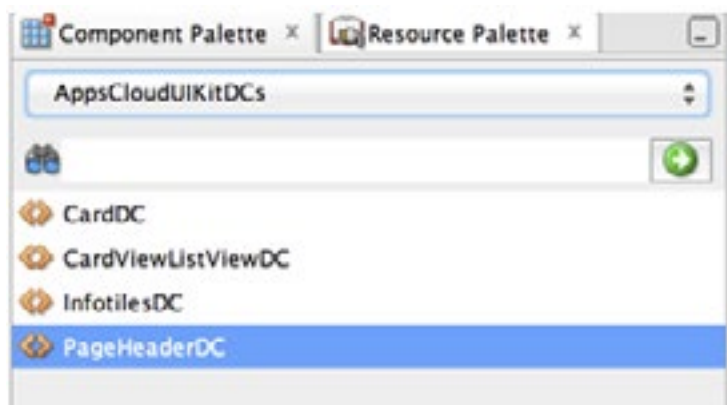


Figure 2-6. Oracle ADF Component Palette: PageHeaderDC

If you do not see this header, your view project does not include the adflibUIKitCommon.jar Oracle ADF Library that comes with the RDK. After you add the component, use the Property Palette to set the Page Title property to Department List.

6. Drop your Departments view object instance from the Data Controls palette onto the Panel Group Layout below the header as a Table/List View > ADF List View.
7. In the Create List wizard, select to use a Panel Grid Layout.

Looking at the layout, it is clear that we need two rows and 3 columns with the first two columns spanning both rows. Note that step 3 of the wizard allows you to define the cell spans:

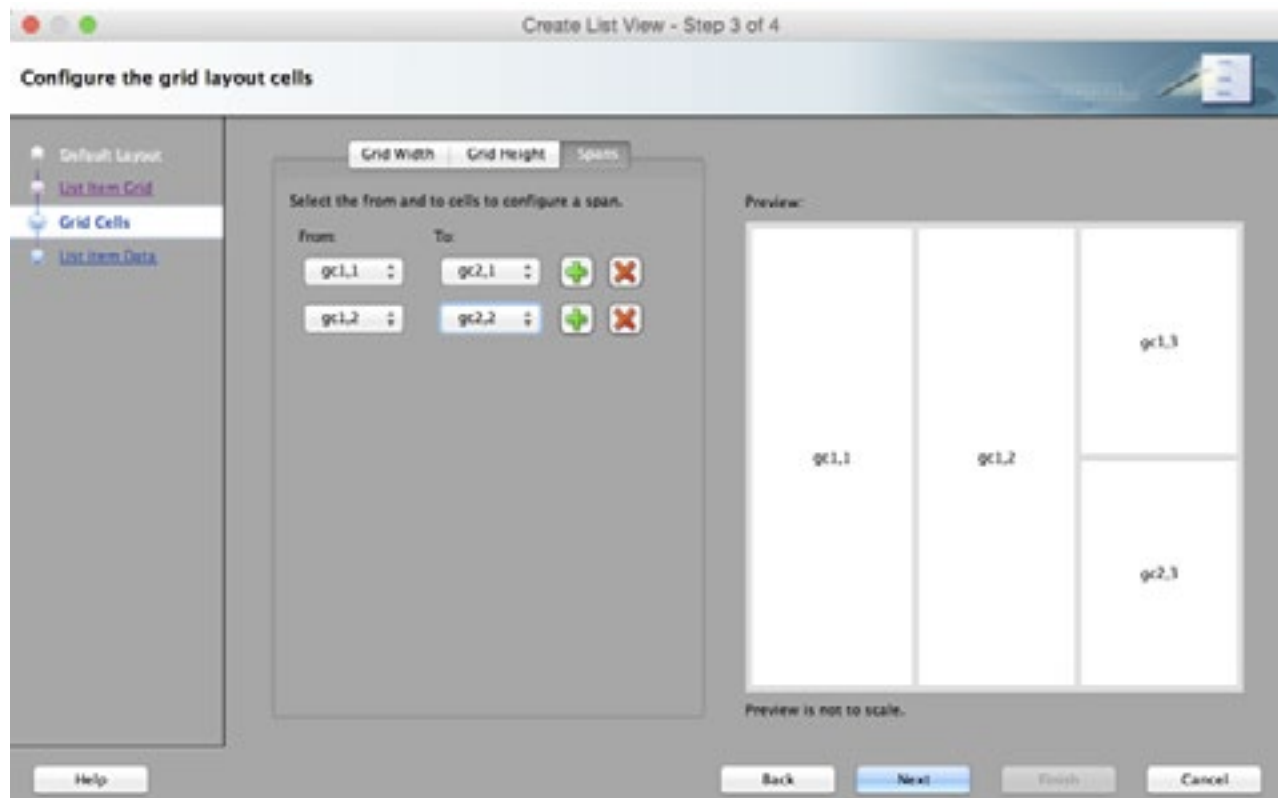


Figure 2-7. Create List View wizard: Panel grid layout, cell spans

8. Place the individual data items into the grid.

You can only place one item in each grid cell with the wizard, so you will have to add the manager last name later. Your field placement should look like this:

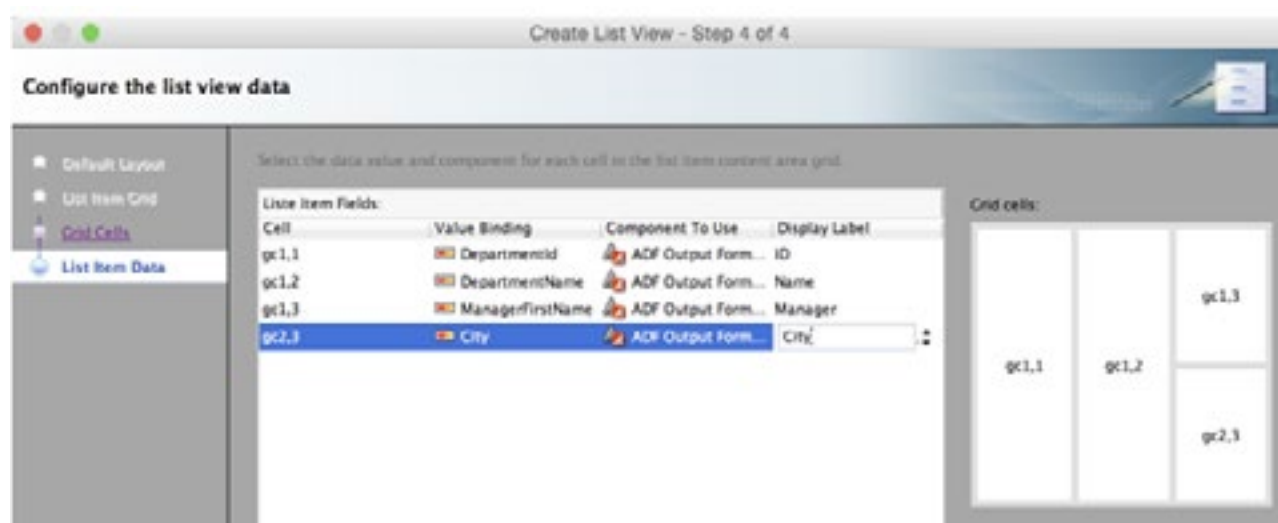


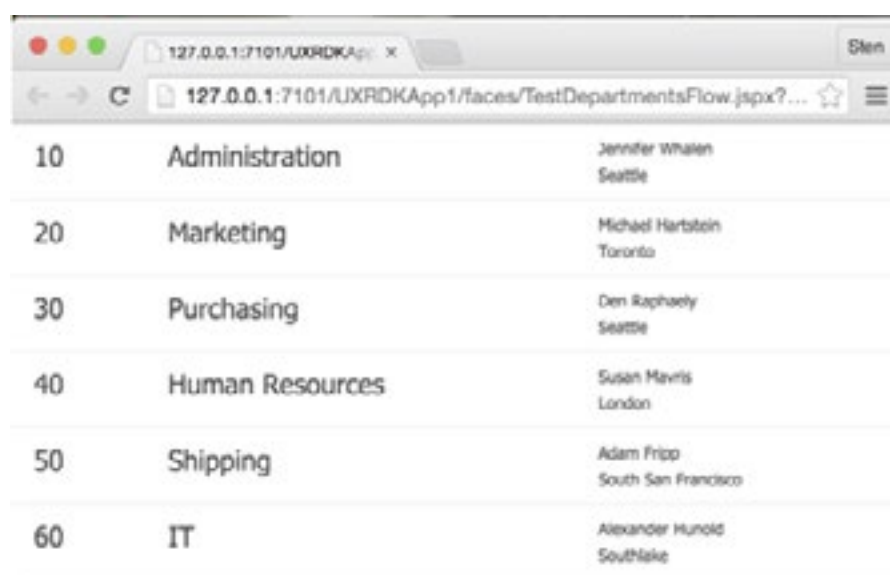
Figure 2-8. Create List View wizard: Field placement

After the page is created:

1. Open the Bindings tab and edit the tree binding for the Departments view object.
2. Shuffle the manager last name attribute to the right to include it under display attributes.
3. Use the Property Palette to change the Value property of the manager name field to include the last name (for example, `# {item.bindings.ManagerFirstName} # {item.bindings.ManagerLastName}`)
4. Change the visual properties of the department ID and name to 14px.
5. Select to use the Oracle Alta skin that simplified UI uses. To do this, simply open the trinidad-config.xml file in you view project and change skin-family to the value alta.

Run the Work Area

After you finish your bounded task flow, it is time to run and test it. Because all work areas are implemented as bounded task flows, they need a surrounding page to run in. Create a test page with a stretchable quick start layout and drop your task flow as a region in the pageContent facet. Then run your application. It should look something like this:



The screenshot shows a web browser window with the address bar displaying `127.0.0.1:7101/UXRDApp1/faces/TestDepartmentsFlow.jspx?...`. The browser window contains a table with the following data:

10	Administration	Jennifer Whalen Seattle
20	Marketing	Michael Hartstein Toronto
30	Purchasing	Den Raphaely Seattle
40	Human Resources	Susan Mavris London
50	Shipping	Adam Fripp South San Francisco
60	IT	Alexander Hunold Southlake

Figure 2-9. Test page

You can also test all task flows with the ADF EMG Task Flow Tester, which has been developed by members of the ADF Enterprise Methodology Group. The ADF EMG Task Flow Tester is a partner extension that you find in Oracle JDeveloper under Help > Check for Updates. Make sure to select the checkbox Open Source and Partner Extensions to see and install it.

A Simulated Integration

After you verify that your task flow works the way that you want it to, you can optionally place it in the demo application to make sure that it matches other work areas in the simplified UI.

To do this, you must first deploy your work area as an Oracle ADF Library.

1. Select the view project and create a deployment profile of type ADF Library JAR File.
2. In the deployment profile properties under Connections, change the setting to deploy Connection Name Only.
3. Use it to deploy your project to an Oracle ADF Library.
4. Place that library in the libs directory of the UX RDK.

The next step is to include your task flow in the demo master application. You might have noticed a PaaS Cloud icon on the springboard. You can add your own icon to the demo master application by opening the DemoData project in the AppsCloudUIKit workspace and finding the SessionState class. At the bottom of the `_buildNodeList()` method, you'll find the following code:

```
//Add Cloud Plug node

_nodeList.add(new Node(idx++, "PaaS Cloud", "cloudplug",
"WEB-INF/oracle/apps/uikit/flow/NotImplementedFlow.xml", "NotImplementedFlow"));
```

You can add a similar line with different title, icon, task flow path and task flow name, for example:

```
//Add Departments node

_nodeList.add(new Node(idx++, "Departments", "buildings",
"WEB-INF/oracle/apps/uikit/uxrdkapp1/dept/flow/DepartmentsFlow.xml",
"DepartmentsFlow "));
```

When done, you need to deploy the DemoData application to an Oracle ADF Library using the existing deployment profile. Then you should see your new icon show up when you run the demo master application.

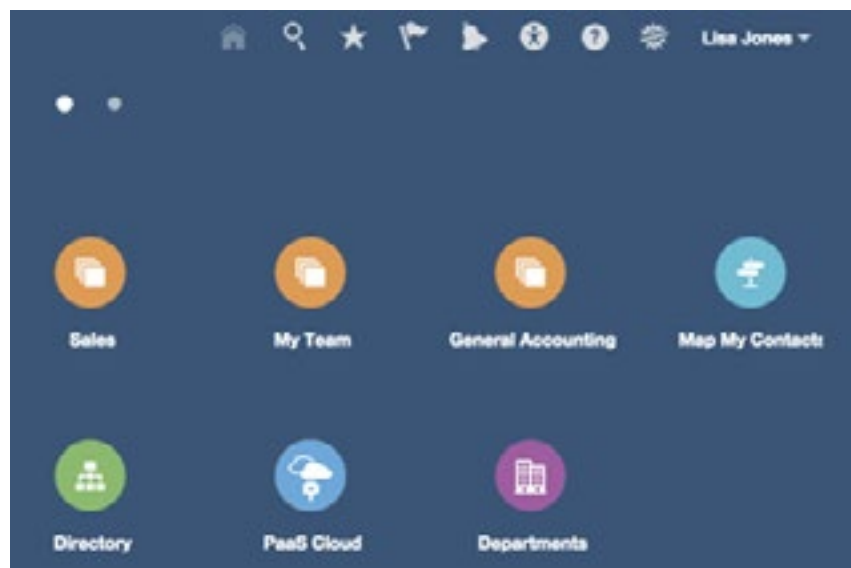


Figure 2-10. Springboard that shows the new Departments icon

When you click this icon, the bounded task flow from the new departments work area should appear:

ID	Department Name	Manager
10	Administration	Jennifer Whalen Seattle
20	Marketing	Michael Hartstein Toronto
30	Purchasing	Den Raphaely Seattle
40	Human Resources	Susan Mavris London

Figure 2-11. Departments list

Call REST Web Services

A very common requirement, especially when building add-ons to an existing Oracle Applications Cloud instance, is to be able to call REST web services.

Knowledge of and use of web services are essential activities when developing SaaS and PaaS solutions.

Calling web services is done from the server side of your application with normal Java calls. There are a number of Java frameworks available to support you writing this code.

For example, you could use Apache HttpComponents to help with the HTTP communication with the REST server, and JSON.simple for handling the JSON objects you will be receiving and possibly sending as well.

The AppsCloudUIKit comes with an example of a REST web service already implemented in the simplified UI for you. We have implemented a Google Maps service for the DemoCRM (Oracle Sales Cloud) contacts information. This service is accessed from the Map My Contacts icon on the simplified UI home experience.

Stay tuned to Usable Apps and other technical channels for examples of useful simple REST services and how you can add them.

3. Deploying to the Oracle Java Cloud

This section covers deploying your Oracle ADF application to the Oracle Java Cloud Service. For a stand-alone application, you can choose whether to deploy to an on-premise WebLogic instance or to a Java Cloud Service instance. If you want your application to be an extension of the Oracle Applications Cloud, deploy the application to a Java Cloud Service – SaaS Extension instance. The application can also be deployed to Oracle Java Cloud Service and linked to from Oracle Applications Cloud.

Stand-alone applications can be tested from your local development environment or an on-premise test environment before deployment. If your application integrates with the Oracle Applications Cloud, you can only perform component tests locally (using calling dummy test web services). A full integration test requires that your application can acquire an authentication token to present to Applications Cloud web services, so it can only be run once the application has been deployed to a Java Cloud Service – SaaS Extension instance.

For current Rapid Development Kit (RDK) deployment options for SaaS and PaaS solutions, contact the Oracle Applications User Experience RDK team through your partner enablement contacts.

To stay informed about the version of Oracle ADF running on the Java Cloud Service, see the [Oracle Java Cloud Service page](#). Click Learn More, and then FAQ.

Cloud Services

In order to run a database-based application in the Java Cloud, you will need:

- A Java Cloud Service
- An associated Database Cloud Service

Note that there are several different types of Java Cloud Service. If you want to build a stand-alone application, you should use a regular *Java Cloud Service* instance. If you intend your application to be integrated into Oracle Applications Cloud, you need to choose the type *Oracle Java Cloud – SaaS Extension*.

When you sign up for a Java Cloud Service trial of either type, an *Oracle Database Cloud Service – Schema* is automatically created for you, and the two services are associated. For security reasons, Oracle only allows a Java Cloud Service to access the Database Cloud Service it is connected to.

When you purchase a Java Cloud Service – SaaS Extension, you will need to also purchase an Oracle Database Cloud Service – Schema and associate them. The SaaS Extension Java cloud only works with the Schema database cloud.

Set Up Database Cloud Service

From the Cloud console, you can click on the Open Service Console link next to the database ser-

vice to get to your database console. If you have used Oracle Application Express 5.0, the user interface is familiar.

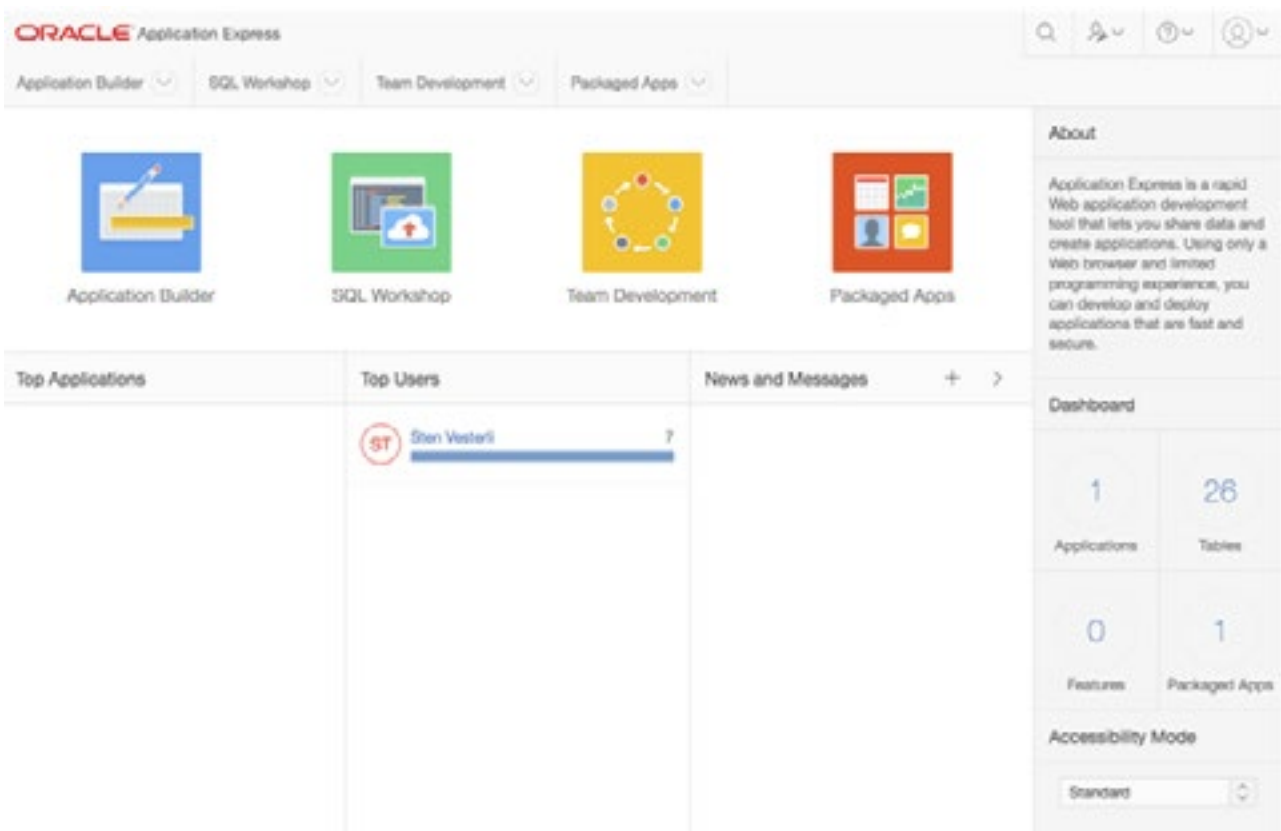


Figure 3-1. Oracle Application Express page

You will need to create a database user to hold the tables and other database objects your application needs. This is done from the administration menu in the top right, where you can select to manager users and groups.

The new user might be provisioned with some default demo tables like EMP, EMPLOYEES, DEPT, and DEPARTMENTS. These are the same as the tables in the normal SCOTT and HR demo schemas in an Oracle database. For training and demonstration purposes, you can just use these tables.

To upload your own tables and data, you use the Database Cart functionality of Oracle JDeveloper. This feature allows you to place tables in a “cart” and then transfer tables and optionally data to your cloud service. To use the Database Cart, Oracle JDeveloper needs various pieces of information about your service, such as URL, host, port, and so on. You can gather this information from the Database Cloud Services detail page.

Deploy to the Oracle Cloud

To get your application running in the Java Cloud Service, you must:

1. Create a Java Cloud Service connection from Oracle JDeveloper.
2. Modify your application database connection information.
3. Configure security.
4. Deploy the application.

Create a Java Cloud Service Connection

In Oracle JDeveloper, create a connection to your Java Cloud Service like any other Application Server connection. You need to provide the username and password for the Oracle.com account you use to log into your cloud services, as well as various connection information about your Java Cloud Service (data center, identity domain, name of service). All of this information is available on your Cloud Services dashboard.

Change the Connection Info

Your application needs to connect to the database service associated with the Java Cloud Service that you intend to deploy the application to. You can find the associations on the Cloud Services dashboard.

You need to use this as the JDBC Datasource name in your application. Datasources are configured in the Application Module configuration.

Configure Security

Depending on your needs, you can configure your application with one of the following security options:

- Internet public: Can be accessed by all users. Users are not prompted to log in
- Oracle public: Can be accessed by any user of Oracle cloud (in all identity domains).

This is the default security option. It does not pass the user identity to the application, so it is not possible for the application to implement any authorization controls.

- Tenant restricted: Can only be accessed by users in your identity domain. The authenticated user is passed to the application and role-based authorization is possible.

Like in other Oracle ADF applications, you can define application roles in the application and grant access to various parts of the application to specific application roles. When deploying the application, you can then map the application roles to the enterprise roles defined within the Oracle Java Cloud Service identity domain (possibly utilizing implicit mapping based on the role name).

Deploy to the Cloud

With a connection to the Java Cloud, the application configured to use the right datasource and security configured appropriately, you can deploy your application to the Java Cloud Service by selecting the Cloud application server connection.

Run Your Application in the Cloud

After deployment, your application should show up in the Java Console for your Java Cloud Service. You can access this from your Cloud dashboard.

If you click on the application, you can read the application URL that you can use to run the application. Note that this URL is for the application itself. You will need to add /faces/<your page> to the URL. For example, if your application URL is `http://<host>:<port>/AppsCloudUI-Kit`, and your page name is `MyPage.jspx`, your complete URL would be `https://javatrial1234.java.us2.oraclecloudapps.com/MyCloudApp/faces/MyPage.jspx`.

4. Integrating with Cloud Applications

This section covers integrating your own application with Oracle Applications Cloud. This eBook gives an overview of the procedures and technologies involved, but if this is your first integration with Cloud Applications, you will probably want to refer to the Oracle Applications Cloud documentation (<http://docs.oracle.com/cloud/latest/>) for details.

About Cloud Applications Customization

Oracle Applications Cloud can be customized in many different ways, including:

- Appearance (predefined themes)
- Logo (brand)
- Icon style (visuals styles, corner rounding)
- Colors (background, region, label, heading, link, button)
- User interface text (including names of functional areas and pages)
- By reordering or hiding functional areas and pages
- By changing pages by reordering or hiding fields
- By adding new fields to pages
- By adding buttons and links to connect to applications outside the Oracle Applications Cloud

All of these types of customizations are known as *runtime customizations* and are created using your web browser and the Oracle Applications Cloud customization tools (primarily the Structure page and the Oracle Composer).

A connection to an application outside Oracle Applications Cloud is a runtime customization, and this section is about creating these.

Call Out from Oracle Applications Cloud

You can call out from the Oracle Applications Cloud in two ways:

- From the Navigator and springboard
- From a page (only some applications)

Call Out from the Navigator and Springboard

You can add an icon for your own application to the Oracle Applications Cloud springboard next to the existing icons. To do this, you use the Structure page in Oracle Applications Cloud, like you do when you make other runtime customizations.

To add your own application, you create a page entry. This involves:

- Providing a page name and category
- Selecting an icon
- Selecting the role or roles allowed to access the page
- Specifying the application URL

You can create your own categories or place pages inside existing categories.

When you create a page entry, it will always show up in the Navigator menu. For each page, you can choose whether you want it to appear as an icon on the springboard as well.

If your category only contains one item and you choose to make it visible on the springboard, the icon appears at the top level, not inside the category. As soon as you add another item to the category, the category icon appears and the pages are shown inside the corresponding functional grouping.

If you have many related applications you want to add to the Oracle Applications Cloud, you can also add your own category. This category will appear to the user as a functional grouping icon. You can place your own pages inside this category just like the Oracle Applications Cloud does.

Call Out from a Page

In some Oracle Sales Cloud applications, you can also customize individual pages by adding buttons or links to call out to your own application. To do this, you need to use the Oracle Application Composer tool to create a new button or link.

Pass Parameters

If you want to pass parameters to your application from the Oracle Applications Cloud, the URL used to call your application must include the parameter values as part of the URL.

The ability to read parameter values from the current Oracle Applications Cloud context only exists when you add your own application to existing pages using Oracle Applications Composer, that is, only in customizations of Oracle Sales Cloud. For a list of the context values available for you to use in your integration, see the [Oracle Applications Cloud documentation](#).

Call Oracle Applications Cloud Web Services

If your application needs more information from the Oracle Applications Cloud than can be passed as a simple parameter, you need to call back into the Oracle Applications Cloud from your own application, using the SOAP web services that the Oracle Applications Cloud makes available.

However, the Oracle Applications Cloud will not accept unauthenticated requests. Therefore, you need to create a JSON Web Token into Oracle Applications Cloud before calling your own application, and then pass this token to your application.

Your application can then take this token and use it to authenticate to the Oracle Applications Cloud when calling web services. The Oracle Applications Cloud accepts JSON Web Tokens and will use that to identify a valid caller.

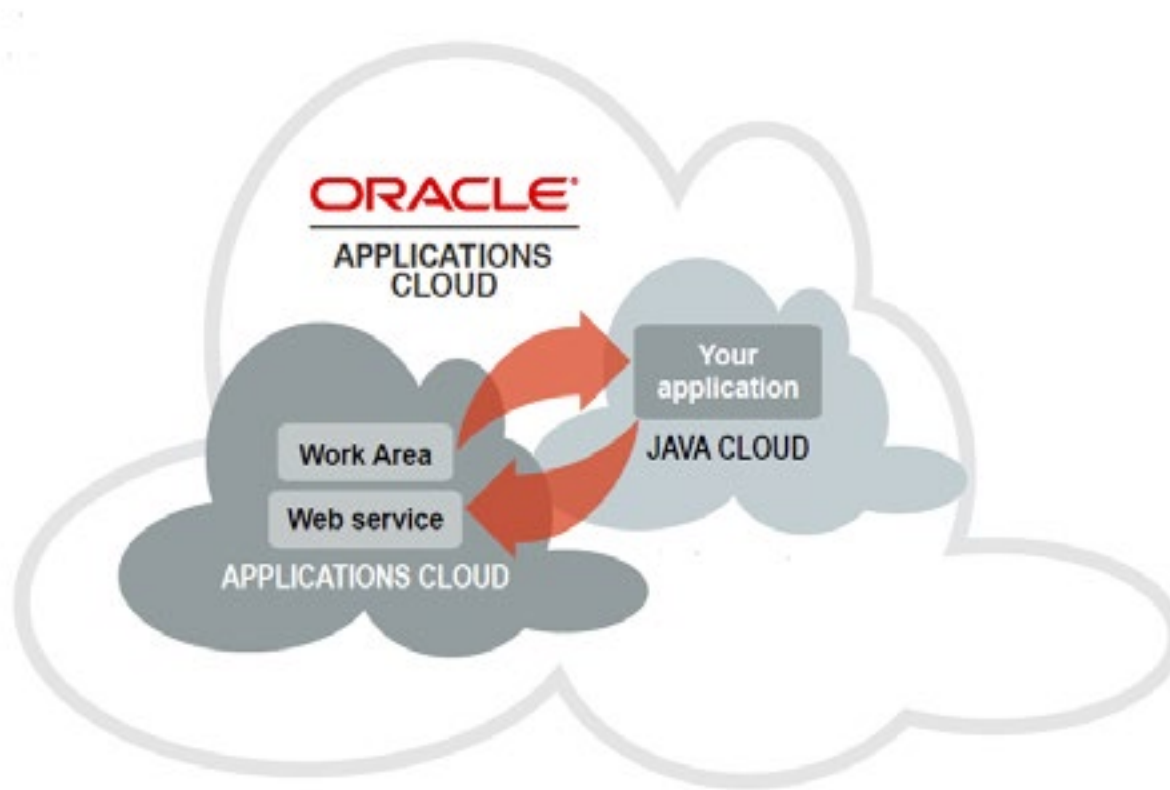


Figure 4-1. Passing authentication between Oracle Applications Cloud and Java Cloud

Frequently Asked Questions

This section answers some frequently asked questions about the Oracle Applications User Experience Rapid Development Kit (RDK).

For more information, see the [Oracle Applications User Experience Rapid Development Kit page](#).

Can I use Eclipse for Oracle ADF development?

Yes, using Oracle Enterprise Pack for Eclipse, you can develop Oracle ADF Faces applications using JPA/EJB business services, and you can use the AppsCloudUIKit Oracle ADF library in your application. The demo application, however, is delivered in an Oracle JDeveloper workspace format. If you want to examine the demo application, it will be easiest to simply install Oracle JDeveloper to look at the code, and then copy any code elements you want to reuse into use Eclipse.

What if I have used an earlier version of the RDK?

The RDK versions corresponding to Release 10 and Release 13 are based on Oracle JDeveloper 11.1.1.9.0 or later, and they come with a new code base. You can open applications built with earlier versions of the RDK in Oracle JDeveloper 11.1.1.9.0 and allow Oracle JDeveloper to update the application. In some cases, you might also need to make minor changes to your application in order to make it work with the RDKs for Release 10 and Release 13.

Does the RDK use Oracle Alta UI?

Yes, all the example applications in the RDK are based on the Oracle Alta UI.

Does the RDK contain Cloud Services?

No, the RDK does not contain any cloud services. It is meant to give examples of how to implement a user experience similar to the one found in Oracle Applications Cloud simplified user interface.

Which Cloud Service should I use?

If you want to build a stand-alone application that does not connect to the Oracle Applications Cloud, you should use a regular *Java Cloud Service* instance. If you intend your application to be integrated into Oracle Applications Cloud, you need to choose the type *Oracle Java Cloud – SaaS Extension*.

Can I use the RDK code samples in my production applications?

Sure, feel free. But realize that the RDK, including the demo application, is *example* code. There is no official support for the RDK, so please do not contact Oracle Support with questions about the RDK.

How much Oracle ADF do I need to know?

You do not need any extra Oracle ADF skills in order to use the RDK. On the other hand, the RDK does not magically imbue you with Oracle ADF programming skills.

If you can build a page without the RDK, you can use the RDK to build the same page but with a modern simplified user interface that makes it look great and fit in with the Oracle Applications Cloud.

Additionally, the RDK contains a lot of code samples that you can study in order to determine how to implement specific features.

Where can I find additional resources on good user experience design?

To learn more about user experience design patterns, see the Oracle Applications User Experience eBook [Simplified User Experience Design Patterns for Oracle Applications Cloud Service](#). Using proven design patterns will help you to quickly build user-friendly applications without having to spend months and years becoming a user experience expert.

Where can I find additional resources on Oracle ADF technology?

To learn more about Oracle ADF, see the Oracle Technology Network (OTN) [Oracle Application Development Framework - Oracle ADF pages](#).

If you have technical questions on Oracle ADF, you can post questions on the [OTN Oracle JDeveloper and Oracle ADF Forum](#).

Where can I find additional resources on Oracle ADF architecture?

To learn more about Oracle ADF architecture, see the Oracle Technology Network (OTN) [Oracle ADF Architecture Square page](#) and the book [Oracle ADF Enterprise Application Development – Made Simple](#) (Vesterli, Sten E.).

To ask questions about Oracle ADF Architecture and learn from experienced Oracle ADF developers and architects, join the [ADF Enterprise Methodology Group](#).

Where can I find examples of applications built with the RDK?

Examples of applications built with the RDK are available on the [Oracle Applications User Experience Rapid Development Kit page](#) under the Partner Successes section.