AIM

MD.070 APPLICATION EXTENSION TECHNICAL DESIGN

Oracle Application Framework Extension

Drilldown to Detail Page

Author:	Arone.zhang
Creation Date:	April 24, 1999
Last Updated:	March 5, 2009
Document Ref:	MD070_OAExtension _TB004
Version:	1.0

Approvals:

<Approver 1> Arone.zhang

<Approver 2>

Copy No. _____

Document Control

Change Record

Date	Author	Version	Change Reference
24-Apr-99	Arone.zhang	1.0	No Previous Document

Reviewers

Name	Position
Arone.zhang	Project Leader

Distribution

Copy No.	Name	Location
1	Library Master	Project Library
2		Project Manager
3		
4		

Note To Holders:

If you receive an electronic copy of this document and print it out, please write your name on the equivalent of the cover page, for document control purposes.

If you receive a hard copy of this document, please write your name on the front cover, for document control purposes.

Contents

课程介绍

概述

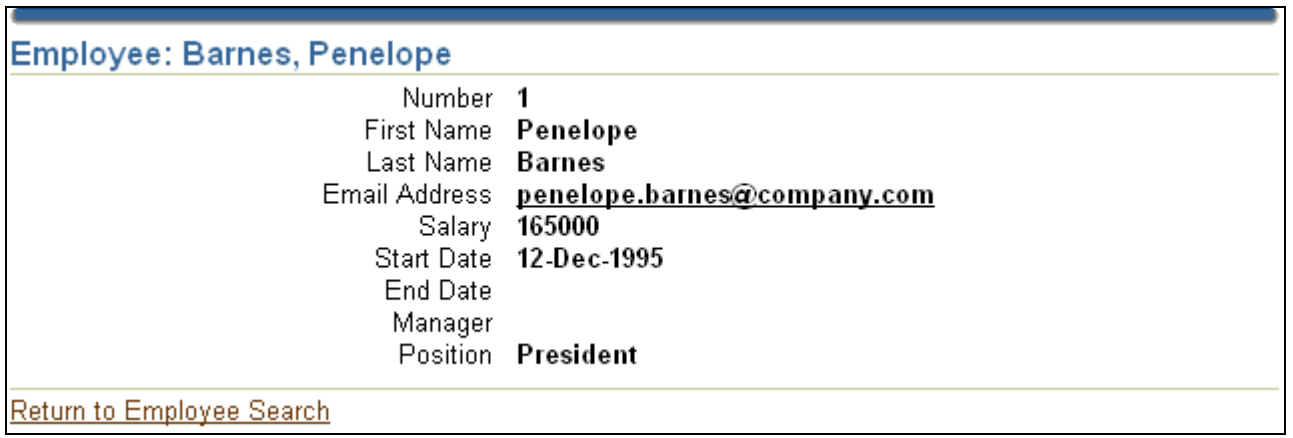
本课程扩展了上一课程中的查询页，包括了钻取到明细页查询员工详细信息。

前提：已经完成了 Search 的课程，如果还没有进行相关的学习，请参照 Search 文档进行之前的工作

图1：在查询页中链接员工名称到明细页中

Number	Name	Manager	Position
1	Barnes, Penelope		President

图2：完成后的明细页



Employee: Barnes, Penelope

Number	**1**
First Name	**Penelope**
Last Name	**Barnes**
Email Address	**penelope.barnes@company.com**
Salary	**165000**
Start Date	**12-Dec-1995**
End Date	
Manager	
Position	**President**

Return to Employee Search

课程的目标

完成此课程之后，需要掌握如下的课题：

- 保留一个应用模块，作为页之间互相导航的UI 应用模块根节点
- 通过编写必要的程序来实现上下文相关的文本信息
- 创建一个典型的单行只读的详细页
- 启用 bread crumbs，在明细页中配置使用“Return To”链接
- 配置一个具有导航链接能力的项目(HTTP GET请求)，并带有请求的参数信息到 URL 参数中
- 配置一个具有“mailto”能力的项目
- 在明细页中实现数据初始化工作

步骤1：创建明细页的视图对象

任务1.1: 创建 EmployeeFullVO 视图对象

创建一个明细级别的视图对象，它包括了所有在后面需要使用的员工属性，这个视图对象是建立在 Search课程中所创建的 EmployeeEO 和 EmpToMgrAO 对象之上的

- 在导航栏中选择 **oracle.apps.ak.employee.server** BC4J 包，右键从上下文菜单中选择 **New View Object...** 打开创建视图对象向导
- 跳过欢迎页面
- 在 **Name** 页中，设置 **Name** 的值为 **EmployeeFullVO**，并确保 Package 的值为 **oracle.apps.ak.employee.server**
- 选择 **下一步** 继续
- 在 **Entity Object** 页中，从 **Available** 列表中选择 **EmployeeEO** 两次(一个为员工，另一个为主管)到 **Selected** 列表中
- 在 **Selected List** 列表中选择 **EmployeeEO1** ，验证和设置如下属性值：

  Association End 设置为 **ManagerIdEmployeeEO**
  Read Only 选中
  Reference 选中

- 选择 **下一步** 继续
- 在 **Attributes** 页中，从 **Available** 列表中选择如下属性到 **Selected** 列表中：
  - 从 **EmployeeEO** 中选择

    EmployeeId
    FirstName
    LastName
    FullName
    Salary
    StartDate
    EndDate
    EmailAddress
    PositionCode
    ManagerId

  - 从 **EmployeeEO1** 中选择

EmployeeId(JDeveloper自动命名为**EmployeeId1**)
FullName(JDeveloper自动命名为**FullName1**)
EmailAddress(JDeveloper自动命名为**EmailAddress1**)

- 选择 **下一步** 到 **Attribute Settings** 页中，设置如下的属性值：

属性	Name 域值	Query Column Alias 域值
EmailAddress	EmployeeEmail	EMPLOYEE_EMAIL
FullName	EmployeeName	EMPLOYEE_NAME
FullName1	ManagerName	MANAGER_NAME
EmailAddress1	ManagerEmail	MANAGER_EMAIL

- 选择下一步，在 Query 页中看到如下的查询语句

```
SELECT EmployeeEO.EMPLOYEE_ID,
       EmployeeEO.FIRST_NAME,
       EmployeeEO.LAST_NAME,
       EmployeeEO.FULL_NAME AS EMPLOYEE_NAME,
       EmployeeEO.EMAIL_ADDRESS AS EMPLOYEE_EMAIL,
       EmployeeEO.MANAGER_ID,
       EmployeeEO.POSITION_CODE,
       EmployeeEO.SALARY,
       EmployeeEO.START_DATE,
       EmployeeEO.END_DATE,
       EmployeeEO1.EMPLOYEE_ID AS EMPLOYEE_ID1,
       EmployeeEO1.full_name AS MANAGER_NAME,
       EmployeeEO1.email_address AS MANAGER_EMAIL
FROM FWK_TBX_EMPLOYEES EmployeeEO,
     FWK_TBX_EMPLOYEES EmployeeEO1
WHERE EmployeeEO.manager_id = EmployeeEO1.employee_id
```

- 使用 Expert Mode 功能修改查询语句，添加 MANAGER_ID / EMPLOYEE_ID 之间的外链接(+)，并连接 FWK_TBX_LOOKUP_CODES_VL 视图取得员工职位的名称

- 完成并验证通过后，应该有下列一样的SQL语句

```
SELECT EmployeeEO.EMPLOYEE_ID,
       EmployeeEO.FIRST_NAME,
       EmployeeEO.LAST_NAME,
       EmployeeEO.FULL_NAME AS EMPLOYEE_NAME,
       EmployeeEO.EMAIL_ADDRESS AS EMPLOYEE_EMAIL,
       EmployeeEO.MANAGER_ID,
       EmployeeEO.POSITION_CODE,
       EmployeeEO.SALARY,
       EmployeeEO.START_DATE,
       EmployeeEO.END_DATE,
       EmployeeEO1.EMPLOYEE_ID AS EMPLOYEE_ID1,
       EmployeeEO1.full_name AS MANAGER_NAME,
       EmployeeEO1.email_address AS MANAGER_EMAIL,
       flkp.meaning AS POSITION_DISPLAY
 FROM FWK_TBX_EMPLOYEES EmployeeEO,
     FWK_TBX_EMPLOYEES EmployeeEO1,
     fwk_tbx_lookup_codes_vl flkp
 WHERE EmployeeEO.manager_id = EmployeeEO1.employee_id (+)
 and EmployeeEO.position_code = flkp.lookup_code
 and flkp.lookup_type = 'FWK_TBX_POSITIONS'
```

- 选择两次 下一步 到 **Java** 页中，选择页中的两个选项框：**View Object Class: EmployeeFullVOImpl > Generate Java File；View Row Class: EmployeeFullVORowImpl > Generate Java File**

- 选择完成创建 VO

任务1.2: 添加视图对象到 根 UI 应用模块

明细页共享 Search 页所使用的根 UI 应用模块，所以需要添加页视图对象到 **EmployeeAM** 应用模块中

- 从导航栏中选择 **EmployeeAM** 应用模块，右键从上下文菜单中选择 **Edit EmployeeAM...** 打开 **Application Module** 编辑向导

- 导航到 **Data Model** 页中，从 **Available View Objects** 列表中选择 **EmployeeFullVO** 到列表中

- 选择 **确定** 完成

步骤2：创建明细页面 UI

本步骤创建一个简单的只读页，页中的员工明细信息在用户点击员工姓名的时候系统自动查询得出。

任务2.1: 创建页

在 **oracle.apps.ak.employee.webui** 包中创建一个新的 UI 页面

- 从导航栏中选择 SearchOAProject.jpr 项目，右键从上下文菜单中选择 **New...**
- 在 **New** 窗口中，展开 **Web Tier** 层次选择 **OA Components**
- 在 OA Components **Items** 列表中选择 **Page**
- 选择 **确定** 继续
- 在 **New Page** 对话框中，输入 **Name** 的值为 **EmpDetailsPG**；**Package** 的值为 **oracle.apps.ak.employee.webui**
- 选择 **确定** 创建页面

任务2.2: 更改 pageLayout 区域

- 在结构窗口中选择 **EmpDetailsPG**中的 **region1**

- 设置和验证以下的属性值

属性	值
ID	PageLayoutRN
Region Style	pageLayout
AM Definition	oracle.apps.ak.employee.server.EmployeeAM
Window Title	Framework ToolBox Tutorial: Labs Arone
Title	Employees: (后面将添加员工号码)
AutoFooter	True

任务3.3: 添加一个产品商标图片

每个 Oracle 应用页面都要求有一个商标图片

- 在结构窗口中选择 **PageLayoutRN** 区域，右键从上下文菜单中选择 **New…productBranding**
- JDeveloper 会自动创建一个包括了**productBranding**(名为 **Item1**) 图形项目的 **pageLayoutComponents** 文件夹，选择其中的项目进行下面的属性设置

属性	值
ID	ProdBrand
Image URI	FNDTAPPBRAND.gif
Additional Text	OA Framework ToolBox Tutorial Arone

任务2.4: 创建主内容区域

为了使项目显示出合适的缩进，在 **PageLayoutRN** 区域中添加 **defaultSingleColumn** 区域，由于此区域中所有的项目都和 **EmployeeFullVO1** 视图实例相对应，使用区域向导快速创建。

- 在结构窗口中选择 **PageLayoutRN** 区域，右键从上下文菜单中选择 **New > Region Using Wizard**
- 跳过欢迎页
- 在 **BC4J Objects** 页中选择 **EmployeeAM(oracle.apps.ak.employee.server.EmployeeAM)**，并从 **Available View Objects** 列表中选择 **EmployeeFullVO1** 视图对象实例
- 选择 下一步 到 **Region Properties** 页，设置 **Region ID** 的值为 **MainRN**；**Region Style** 的值为 **defaultSingleColumn**
- 下一步到 **View Attributes** 页中，从 **Available View Attributes** 列表中选择如下属性列到 **Selected View Attributes** 列表中

  EmployeeId
  FirstName
  LastName
  EmployeeEmail
  PositionDisplay
  ManagerName
  Salary
  StartDate
  EndDate

- 选择 下一步 到 **Region Items** 页，设置如下的项目属性：

ID	Prompt	Style	Data Type	Attribute Set
EmpNum	Number	messageStyledText	NUMBER	/**< base path>**/EmployeeId_Number
FirstName	First Name	messageStyledText	VARCHAR2	/**< base path>**/FirstName
LastName	Last Name	messageStyledText	VARCHAR2	/**< base path>**/LastName
EmailAddress	Email Address	messageStyledText	VARCHAR2	/**< base path>**/EmailAddress
Position	Position	messageStyledText	VARCHAR2	/**< base path>**/Position
MgrName	Manager	messageStyledText	VARCHAR2	/**< base path>**/FullName_Manager
Salary	Salary	messageStyledText	NUMBER	/**< base path>**/Salary
StartDate	Start Date	messageStyledText	DATE	/**< base path>**/StartDate
EndDate	End Date	messageStyledText	DATE	/**< base path>**/EndDate

<base patch> 为 **/oracle/apps/fnd/framework/toolbox/attributesets/FwkTbxEmployees**

- 确定创建数据表
- 在结构窗口中选择 **MainRN**，设置 Hide Header 属性值为 **True**；Text 属性值留空

任务2.5: 设置 MainRN 区域的属性

大部分 **MainRN** 区域的属性在创建 **defaultSingleColumn** 的时候就已经设置好了，现在需要设置 EmailAddress 和 MgrName 两个项目的具有“mailto”链接的功能，并使用符合规范的 CSS Class

mailto:{@<*ViewObjectAttributeName*>} 的语法格式，在运行的时候，OA Framework 自动替换在两个大括号之间的视图属性的具体值

打开 **MainRN** 区域下的项目进行如下的属性值设置：

EmpNum

属性	值
CSS Class	OraDataText

FirstName

属性	值
CSS Class	OraDataText

LastName

属性	值

属性	值
CSS Class	OraDataText

EmailAddress

属性	值
Destination URI	mailto:{@EmployeeEmail}
CSS Class	OraDataText

Position

属性	值
CSS Class	OraDataText

MgrName

属性	值
Destination URI	mailto:{@ManagerEmail}
CSS Class	OraDataText

Salary

属性	值
CSS Class	OraDataText

StartDate

属性	值
CSS Class	OraDataText

EndDate

属性	值
CSS Class	OraDataText

任务2.6: 改变区域的样式

选择 **MainRN defaultSingleColumn** 区域，改变它的 Region Style 属性值为 **messageComponentLayout** 。JDeveloper 将显示如下的警告信息，选择 **是** 继续，区域中的项目不会受到影响。

提示：**messageComponentLayout** 是 11.5.10 新发布的组件，它替换了所有以 **default*** 的区域。但是"region using wizard"的方式不支持直接创建 **messageComponentLayout** ，当单行区域的所有项目是和试图对象绑定在一起的时候，通过这种方式来创建这种区域是最方便有效的。当然等到"region using wizard"的方式支持直接创建 **messageComponentLayout** 的时候，我们就没有必要采取这种方式了。

任务2.7: 增加一个 "Return to Employee Search" 链接

在页面的底部增加一个"Return to Employee Rearch"链接

- 在结构窗口中选择 **PageLayoutRN**，右键选择 **New > returnNavigation**

- 选择自动创建在 **pageComponents** 文件夹下的 **returnNavigation link** 项目，设置如下的属性值：

属性	值
ID	ReturnLink
Destination URI	OA.jsp?page=/oracle/apps/ak/employee/server/webui/EmpSearchPG&retainAM=Y
Text	Return to Employee Search

步骤3：实现试图对象查询

任务3.1: 添加一个 initQuery 方法到 EmployeeFullVOImpl 类中

此方法适用一个 employeeId 参数，并将参数传入查询语句中的 WHERE 子句 **EMPLOYEE_ID = :1** ，然后执行查询。

提示：当你需要初始化一个 Number 类型的变量时，记得导入 oracle.jbo.domain.Number；否则在运行的时候会抛出 java.lang.Number 错误。

```
import oracle.jbo.domain.Number;
import oracle.apps.fnd.framework.OAException;
...


 public void initQuery(String employeeNumber)
 {
   if ((employeeNumber != null) &&
       (!("".equals(employeeNumber.trim()))))
```

```
    {

     // Do the following conversion for type consistency.
     Number empNum = null;

     try
     {
      empNum = new Number(employeeNumber);
     }
     catch(Exception e)
     {
      throw new OAException("AK", "FWK_TBX_INVALID_EMP_NUMBER");
     }
     setWhereClause("EMPLOYEE_ID = :1");
     setWhereClauseParams(null); // Always reset
     setWhereClauseParam(0, empNum);
     executeQuery();

    }
  } // end initQuery()
```

任务3.2: 添加 initDetails() 方法到 EmployeeAMImpl 类中

后面将会从 UI 控制器调用此方法(OA Framework的编码标准：直接与在控制器中的 **OAApplicationModule** 接口进行交互，不要直接和视图对象进行交互)，此方法委派你在 **EmployeeFullVOImpl** 类中创建的 **initQuery** 方法

```
/*******************************************************************************
 * Initializes the detail employee query.
 *******************************************************************************
 */
 public void initDetails(String employeeNumber)
```

```
 {
  EmployeeFullVOImpl vo = getEmployeeFullVO1();
  if (vo == null)
  {
   MessageToken[] errTokens = { new MessageToken("OBJECT_NAME", "EmployeeFullVO1")};
   throw new OAException("AK", "FWK_TBX_OBJECT_NOT_FOUND", errTokens);
  }
  vo.initQuery(employeeNumber);

 } // end initDetails()
```

任务3.3: 保存并编译

- 选择 JDeveloper 主菜单 **File > Save All**
- 选择项目，右键选择 **Rebuild SearchOAProject.jpr** 编译完成的所有

步骤4：实现下钻到 EmpDetailsPG

本步骤中，在上一课中学习的查询结果表中 EmpName 列上面添加一个链接，链接导航到一个新的明细页。

任务4.1: 配置 EmpName 项目成为一个链接

当用户选择员工姓名链接时，显示一个新的明细页。这是属于一个 GET 请求，我们需要做：

- 添加 **employeeNumber** 和 **employeeName** 参数到 URI，这些参数的值来源于 **EmployeeFullVO1** 中的 **EmployeeId** 和 **EmployeeName**

- 意味着在 **EmpDetailsPG** 打开使用过程中， **EmpSearchPG** 所属的应用模块实例必须保持原有的状态，保证这些页面可以共享一个根应用模块

选择 ResultsTable 区域中的 EmpName 项目，设置 Destination URI 的属性值为：

OA.jsp?page=/oracle/apps/ak/employee/webui/EmpDetailsPG&employeeNumber={@EmployeeId}&employeeName={@EmployeeName}&retainAM=Y&addBreadCrumb=Y

在运行的时候，OA Framework 自动替换在两个大括号之间的视图属性的具体值。例如：&employeeNumber={@EmployeeId} 变为 &emloyeeNumber=1234

任务4.2: 为明细页创建控制器

- 在结构窗口中选择 **EmpDetailsPG** 中的 **PageLayoutRN** ，右键从上下文菜单中选择 **Set New Controller ...**
- 在 **New Controller** 对话框中，指定 **Package Name** 的值为 **oracle.apps.ak.employee.server.webui** ； **Class Name** 的值为 **EmployeeDetailsCO**
- 选择 **确定** 按钮完成控制器创建

任务4.3: 增加控制器逻辑以在页面被请求的时候初始化员工查询

为了使 **EmpDetailsPG** 页面在被请求的时候能够自动的查询 **EmployeeFullVO** 试图对象下的数据，需要添加如下的代码到 **EmployeeDetailsCO.processRequest()** 方法中。

其中引用了 employeeNumber 参数的值

```
import java.io.Serializable;
import oracle.apps.fnd.framework.OAException;
import oracle.apps.fnd.framework.OAApplicationModule;
...
```

```
public void processRequest(OAPageContext pageContext, OAWebBean webBean)
{
  // Always call this first.
  super.processRequest(pageContext, webBean);

  // Get the employeeNumber parameter from the URL
  String employeeNumber = pageContext.getParameter("employeeNumber");

  // Now we want to initialize the query for our single employee
  // with all of its details.
  OAApplicationModule am = pageContext.getApplicationModule(webBean);
  Serializable[] parameters = { employeeNumber };
  am.invokeMethod("initDetails", parameters);

}
```

任务4.4: 通过编程来实现页标题文本的动态显示

按照 BLAF UI Guidelines on Header Components 准则，需要在明细页的标题上显示当前被选择的员工名称，为此需要在 **processRequest()** 方法中添加如下的代码

假设 FWK_TBX_T_EMP_HEADER_TEXT 消息已经在系统中创建，其消息定义的内容为 **Employee: &EMP_NAME**

```
import oracle.apps.fnd.common.MessageToken;
import oracle.apps.fnd.framework.webui.beans.layout.OAPageLayoutBean;

...

 public void processRequest(OAPageContext pageContext, OAWebBean webBean)
 {
   // Code from Step 4.3 ommitted for clarity... The following should be
   // added after the code you added above.
```

```
  // Get the employeeName parameter from the URL
  String employeeName = pageContext.getParameter("employeeName");

  // Always use a translated value from Message Dictionary when setting
  // strings in your controllers.
  // Instantiate an array of message tokens and set the value for the
  // EMP_NAME token.
  MessageToken[] tokens = { new MessageToken("EMP_NAME", employeeName)};

  // Now, get the translated message text including the token value.
  String pageHeaderText =
    pageContext.getMessage("AK", "FWK_TBX_T_EMP_HEADER_TEXT", tokens);

  // Set the employee-specific page title (which also appears in
  // the breadcrumbs).  Note that we know this controller is
  // associated wit the pageLayout region, which is why we cast the
  // webBean to an OAPageLayoutBean before calling setTitle.

  ((OAPageLayoutBean)webBean).setTitle(pageHeaderText);

 }
```

任务4.5: 保存测试

Open and Closed Issues for this Deliverable

Open Issues

ID	Issue	Resolution	Responsibility	Target Date	Impact Date

Closed Issues

ID	Issue	Resolution	Responsibility	Target Date	Impact Date