开发所有联系人页面即AllContacts页面

需求：加载页面后，从sl\_employee表中加载所有员工信息到datagrid中。

sl\_employee表脚本：

CREATE TABLE `sl\_employee` (

`EMPLOYEE\_ID` int(11) NOT NULL AUTO\_INCREMENT,--员工Id

`EMPLOYEE\_CODE` varchar(20) DEFAULT NULL,--员工编号

`USER\_NAME` varchar(20) DEFAULT NULL,--系统Id

`DEPT\_ID` int(11) DEFAULT NULL,--部门编号

`EMPLOYEE\_NAME` varchar(16) DEFAULT NULL,--员工姓名

`SEX` bit(1) NOT NULL,--性别

`BIRTHDAY` datetime DEFAULT NULL,--生日

`MARRIED` bit(1) NOT NULL,--婚否

`SALARY` decimal(10,2) DEFAULT NULL,--薪水

`DEGREE` varchar(30) DEFAULT NULL,--学历

`EMAIL` varchar(50) DEFAULT NULL,--电子邮件

`WEB` varchar(50) DEFAULT NULL,--个人主页

`CMNT` varchar(255) DEFAULT NULL,--备注

`IMAGE` varchar(50) DEFAULT NULL,--照片

`PASSWORD` varchar(45) DEFAULT NULL,--密码

`ANIMAL` varchar(45) DEFAULT NULL,--属相

`PHONE` varchar(45) DEFAULT NULL,--座机

`MOBILE` varchar(45) DEFAULT NULL,--移动电话

`POSITION` varchar(45) DEFAULT NULL,--职务

PRIMARY KEY (`EMPLOYEE\_ID`),

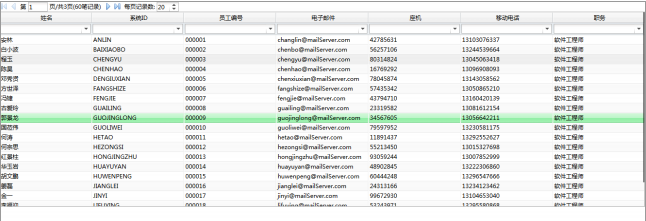
KEY `fk\_dept\_id` (`DEPT\_ID`),

CONSTRAINT `fk\_dept\_id` FOREIGN KEY (`DEPT\_ID`) REFERENCES `sl\_dept` (`dept\_id`) ON DELETE

NO ACTION ON UPDATE NO ACTION

) ENGINE=InnoDB AUTO\_INCREMENT=62 DEFAULT CHARSET=utf8;

效果如下：



## 1、准备工作

涉及到sl\_employee表，因此需要在Common.model.xml中新建DataType。新建DataType，在Common.model.xml文件里，Model节点里新建DataType，设置属性：

|  |  |  |
| --- | --- | --- |
| 属性 | 值 | 描述 |
| name | SlEmployee | DataType的名称 |
| matchType | com.bstek.dorado.sample.standardlesson.entity.SlEmployee | 该DataType可匹配的Java Bean的类型 |

右键[通过简单Java对象自动创建PropertyDefs]。IDE会自动根据这个DataType所设定的matchType定义的POJO对象映射生成DataType的PropertyDefs信息，设置每个字段的中文说明，分别设置PropertyDef的label属性。

## 2、开发步骤

### 2.1全部查询

1）创建视图控件

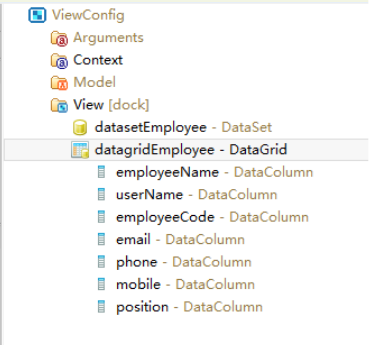
新建AllContacts.view.xml，在[view]节点下添加一个DataSet控件，设定DataSet的属性如下：

|  |  |  |
| --- | --- | --- |
| 属性 | 值 | 说明 |
| id | datasetEmployee | 控件的id |
| DataType | [SlEmployee] | DataSet的数据类型 |
| dataProvider | employeeService#getAll | 由后台employeeService这个beat中的getAll方法为DataSet提供数据 |

为展现其中的数据需要添加一个数据敏感型控件对象DataGrid，DataGrid就是我们日常开发中使用的表格控件。在视图中添加DataGrid控件并设置属性：

|  |  |  |
| --- | --- | --- |
| 属性 | 值 | 说明 |
| id | datagridEmployee | 控件的id |
| dataSet | datasetEmployee | 通过dataSet属性告知DataGrid绑定是哪些数据，以便在DataGrid中展示和处理 |

然后右键选择[Generate DataColumns]，生成相关字段。调整字段显示顺序，删除多余字段，保留employeeName，userName，employeeCode，email，phone，mobile，position这几个字段：



（注：假如不生成DataGrid Columns，那么DataGrid将显示绑定DataSet的DataType的所有字段）

2）准备后台服务代码

@Component

public class EmployeeService {

@Resource

private SlEmployeeDao slEmployeeDao;

@DataProvider

public Collection<SlEmployee> getAll() {

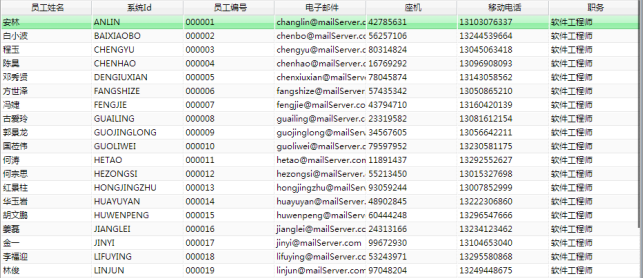
return slEmployeeDao.getAll();

}

}

3）查看运行效果

<http://localhost:8080/standard-lesson/com.bstek.dorado.sample.standardlesson.junior.contacts.AllContacts.d>



### 2.2分页查询

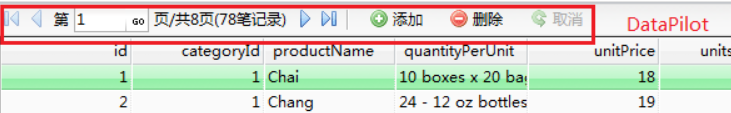
通过全部查询可以发现，DataGrid将所有员工信息直接显示出来，而实际开发中一旦数据量大的时候可能会有性能隐患，因此我们需要对其进行分页显示。对View进行修改，使之能够进行分页显示。

1）修改控件设置

修改datasetEmployee的属性：

|  |  |  |
| --- | --- | --- |
| 属性 | 值 | 说明 |
| pageSize | 15 | 装载数据时使用的分页大小，即按照每页多少条记录来进行分页装载 |
| dataProvider | employeeService#getAllForPage | 通过EmployeeService类中的getAllForPage方法为DataSet提供数据加载服务 |

新增分页控件，DataPilot是数据导航条控件，主要用于数据分页导航显示等。



在[View] 节点下添加一个DataPilot控件，将其移动到datagridEmployee控件上方，并设置属性：

|  |  |  |
| --- | --- | --- |
| 属性 | 值 | 说明 |
| id | datapilotEmployee | 控件id |
| dataSet | datasetEmployee | 控件绑定的DataSet |
| itemCodes | pages,pageSize | 用于定义数据导航条中要显示那些子对象的表达式。  pages用于简单翻页按钮定义的特殊代码，表示|<,<,goto,>,>|这5个子对象的组合。  pageSize显示和指定每页记录数的编辑框。 |

2）准备后台服务

在EmployeeService类中添加getAllForPage方法：

@DataProvider

public void getAllForPage(Page<SlEmployee> page) {

slEmployeeDao.getAll(page);

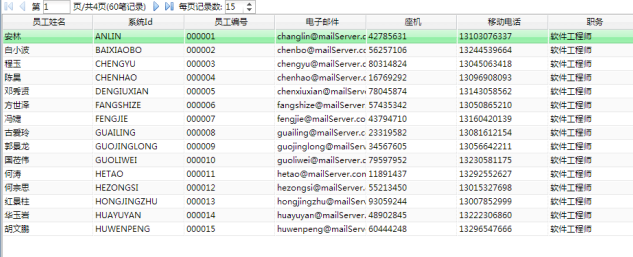
}

对于处理分页的DataProvider方法，要求必须包含一个Page参数，Page参数是Dorado7中支持分页的一个工具类，采用分页技术处理后，每次请求一个新的分页数据都会产生一个DataProvider的AJAX请求，在这个请求动作中，客户端会自动上传每页需要加载的数据时多少条（pageSize）和现在需要加载的数据是第几页的（pageNo），这些信息都会由DataProvider处理机制自动将信息放在Page对象中，并激活DataProvider方法。这样我们就可以在DataProvider方法中通过page对象的getPageSize()和getPageNo()方法获取客户端分页请求的基本分页信息。并交给后台做数据查询的DAO做进一步的处理。处理后台的数据存放在Page对象中，并传递到前台进行展示。

（可以在getAllForPage方法内部添加断点，并在每一次翻页的时候注意查看page中的pageSize和pageNo信息。）

3）查看运行效果

<http://localhost:8080/standard-lesson/com.bstek.dorado.sample.standardlesson.junior.contacts.AllContactsForPage.d>



### 2.3过滤栏查询

可以通过打开DataGrid控件的过滤栏属性，使之可以通过过滤栏进行数据查询，降低开发的工作量。

1）修改控件设置

修改[datagridEmployee]控件设置，默认情况下过滤栏不显示：

|  |  |  |
| --- | --- | --- |
| 属性 | 值 | 说明 |
| showFilterBar | true | 设置DataGrid的过滤栏显示与否（默认不显示） |
| filterMode | serverSide | 设置DataGrid的过滤模式为服务端过滤 |

修改datasetEmployee设置，设置属性如下：

|  |  |  |
| --- | --- | --- |
| 属性 | 值 | 说明 |
| dataProvider | employeeService#getAllForFilter | 由后台employeeService这个bean中的getAllForFilter方法为DataSet提供数据 |

2）准备后台服务

在EmployeeService类中增加新方法getAllForFilter：

@DataProvider

public void getAllForFilter(Page<SlEmployee> page, Criteria criteria)

throws Exception {

DetachedCriteria detachedCriteria = DetachedCriteria

.forClass(SlEmployee.class);

if (criteria != null) {

slEmployeeDao.find(page,

HibernateUtils.createFilter(detachedCriteria, criteria));

} else {

slEmployeeDao.getAll(page);

}

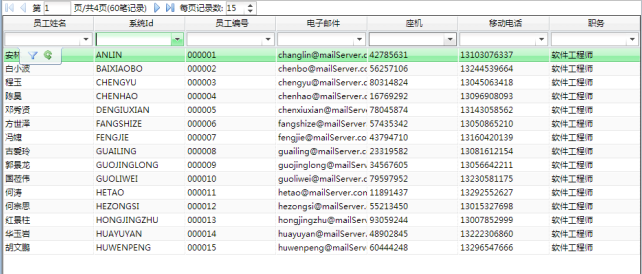
}

新传入creteria参数，Criteria对象（com.bstek.dorado.data.provider.Criteria），这是Provider提供的一个封装对象，用于描述Grid中过滤栏中的查询条件。通过传入查询条件组装插叙语句，最终获得查询结果。

Criteria通常有多个列，如果在多个列都中都输入值，则可以通过其criteria对象的criterions集合获取多个查询条件，并通过集合中的SingleValueFilterCriterion获取查询条件和值。

3）查看运行效果

<http://localhost:8080/standard-lesson/com.bstek.dorado.sample.standardlesson.junior.contacts.AllContactsShowFilterBar.d>



## 3、知识点

* AutoForm控件的使用和布局
* 了解获取数据库中表单的多条记录的方式
* 进一步熟悉DataSet和DataType的概念