Yigo二次开发

Yigo二次开发

目录

第 1 部分

目录

第 1 章 客户端公式

目录

1.1:操作步骤

步骤一：创建客户端二次开发类并继承IFunctionProvider接口，并注册二次开发方法类。如下：

```
package com.bokesoft.yes.scm.function;

import com.bokesoft.yigo.parser.IFunImplCluster;
import com.bokesoft.yigo.parser.IFunctionProvider;

public class SCMUIFunctionProvider implements IFunctionProvider {
    @Override
 public IFunImplCluster[] getClusters() {
  return new IFunImplCluster[] {
    new SCMUIFunction(),
  };
 }
}
```

步骤二：创建二次开发方法类SCMUIFunction，并继承BaseViewFunctionImpl接口。具体如下：

```
public class SCMUIFunction extends BaseFunImplCluster {

 public SCMUIFunction() {
  super();
 }

 @Override
 public Object[][] getImplTable() {
  return new Object[][] { { "SCM_Load", new SCM_LoadImpl() } };
 }

 private class SCM_LoadImpl extends BaseViewFunctionImpl {
  @Override
  public Object evalImpl(String name, ViewEvalContext context,
    Object[] args, IExecutor executor) throws Throwable {
   return true;
  }
 }
}
```

步骤三：在SCMUIFunction类中的getImplTable方法中注册公式，公式类继承BaseViewFunctionImpl接口，并在evalImpl方法中实现。如上。

步骤四： 在Enhance.xml中注册

```
<ExtUIFunction>
<UIFunction Description="SCM 客户端扩展公式" Provider="com.bokesoft.yes.scm.function.SCMUIFunctionProvider"/>
</ExtUIFunction>
```

第 2 章 中间层公式

目录

2.1:操作步骤

步骤一：创建中间层Provider类并继承IFunctionProvider接口，并注册二次开发方法类。如下：

```
package com.bokesoft.yes.scm.function;

import com.bokesoft.yigo.parser.IFunImplCluster;
import com.bokesoft.yigo.parser.IFunctionProvider;

public class SCMMidFunctionProvider implements IFunctionProvider {
 @Override
 public IFunImplCluster[] getClusters() {
  return new IFunImplCluster[] {
    new SCMMidFunction()
  };
 }
}
```

步骤二：创建中间层二次开发方法类例如SCMMidFunction继承BaseFunImplCluster接口。

```
public class SCMMidFunction extends BaseFunImplCluster {

 public SCMMidFunction() {
  super();
 }

 @Override
 public Object[][] getImplTable() {
  return new Object[][] { { "SCM_Depart", new SCM_DepartImpl() } };
 }

 private class SCM_DepartImpl extends BaseMidFunctionImpl {
  @Override
  public Object evalImpl(String name, DefaultContext context,
    Object[] args, IExecutor executor) throws Throwable {
   return true;
  }
 }
}
```

步骤三：在中间层二次开发方法类中的getImplTable方法中注册公式，公式类继承BaseMidFunctionImpl接口，并在evalImpl方法中实现。 如上。

步骤四： 在Enhance.xml中注册

```
<ExtMidFunction>
<MidFunction Description="SCM 中间层扩展公式" Provider="com.bokesoft.yes.scm.function.SCMMidFunctionProvider"/>
</ExtMidFunction>
```

第 3 章 中间层服务

目录

3.1:操作步骤

步骤一：创建中间层服务类并继承IExtService接口，在方法doCmd中做具体实现。

步骤二： 在Enhance.xml中注册

```
<ExtService>
<Service Name="DivideDataService" Description="SCM分量/分段服务" Impl="com.bokesoft.yes.scm.service.DivideDataService"/>
</ExtService>
```

步骤三：可使用公式InvokeService调用服务，InvokeService参数如下：

参数一：serviceName 服务名

参数二~参数N：paras 服务运行所需要的参数

步骤四：可在客户端公式中调用服务，如下：

```
IForm form = context.getForm();
IServiceProxy proxy = ServiceProxyFactory.getInstance().newProxy(form.getVE());
String serviceName = TypeConvertor.toString(args[0]);
ArrayList<Object> paras = new ArrayList<Object>();
proxy.invokeService(serviceName, paras);
```

第 4 章 中间层下推后的数据处理

目录

4.1:操作步骤

适用于下推之后需要对下推后的数据做自定义修改的情况

步骤一：创建中间层二次开发类并继承IServiceProcess<DefaultContext>接口。具体如下：

```
package com.bokesoft.yes.scm.process;

import com.bokesoft.yigo.mid.base.DefaultContext;
import com.bokesoft.yigo.mid.service.IServiceProcess;
import com.bokesoft.yigo.struct.datatable.DataTable;
import com.bokesoft.yigo.struct.document.Document;

public class MapProcess implements IServiceProcess<DefaultContext> {
 @Override
 public void process(DefaultContext context) throws Throwable {
  Document document = context.getDocument();
  DataTable dataTable = document.get("TarMapDetail");

  dataTable.beforeFirst();
  while(dataTable.next()) {
  }
 }
}
```

步骤二：在映射关系文件中的Map节点中，设置属性
PostProcess="com.bokesoft.yes.scm.process.MapProcess"

第 5 章 下推分组策略

目录

5.1:操作步骤

适用于一张上游单据自动生成多张下游单据并需要对下推的内容做自定义拆分的情况。

步骤一：创建中间层二次开发方法并继承IMapSplitProxy接口，实现doMapSplit方法。具体实现如下：

```
public class SCMMapGroupSplitProxy implements IMapSplitProxy {

 @Override
 public ArrayList<MapSplitInfo> doMapSplit(MetaSplit metaSplit,
   Document doc, String primaryKey) {
  DataTable mainTable = doc.get(primaryKey);
  HashMap<String, MapSplitInfo> map = new HashMap<String, MapSplitInfo>();
  mainTable.beforeFirst();
  while (mainTable.next()) {
   String addressKey = mainTable.getString("LOL_ADDRESS_KEY");
   MapSplitInfo info = null;
   if (map.containsKey(addressKey)) {
    info = map.get(addressKey);
   } else {
    info = new MapSplitInfo();
    map.put(addressKey, info);
   }
   info.addBookMark(mainTable.getBookmark());
  }
  ArrayList<MapSplitInfo> array = new ArrayList<MapSplitInfo>();
  array.addAll(map.values());
  return array;
 }
}
```

步骤二：在映射关系文件中的Map节点下，添加Split子节点，其中Type属性为Custom。并再SplitProcess子节点中注册二次开发类。具体如下：

```
<Split Type="Custom">
 <SplitProcess Description="收装信息分组" Impl="com.bokesoft.yes.scm.impl.SCMMapGroupSplitProxy">
   </Split>
```

第 6 章 登录界面自定义字段处理

目录

6.1:操作步骤

适用于自定义登录界面，并需要将自定义字段作为全局变量存储的情况

步骤一：创建中间层间二次开发类并继承ISessionParaItemsProvider接口。实现其中loadItems方法，其中user为登录名。具体如下：

```
package com.bokesoft.yigo.mid.session;

import com.bokesoft.yigo.common.struct.PairItem;
import com.bokesoft.yigo.common.struct.PairItemList;
import com.bokesoft.yigo.mid.base.DefaultContext;

public class SessionParaItemsProvider implements ISessionParaItemsProvider {
 @Override
 public PairItemList loadItems(DefaultContext context, String user)
   throws Throwable {
  PairItemList items = new PairItemList();
  if ( "admin".equalsIgnoreCase(user) ) {
   items.add(new PairItem(10000, "机构1"));
   items.add(new PairItem(10001, "机构2"));
  } else {
   items.add(new PairItem(10000, "机构1"));
  }
  return items;
 }
}
```

步骤二：自定义登录界面，继承DefaultLogin类，其中

方法一：isEnableSessionPara 是否启用全局变量字段。

具体实现如下：

```
package com.bokesoft.yigo.fxapp.ui.auth;

public class SessionParaLogin extends DefaultLogin {
 public SessionParaLogin() {
  super();
 }

 @Override
 public boolean isEnableSessionPara() {
  return true;
 }
}
```

步骤三：在Setting.xml中注册:

1、注册自定义字段下拉选项的回调函数

<Session SessionParaKey="key" SessionParaItemsProvider="com.bokesoft.yigo.mid.session.SessionParaItemsProvider"/>

2、自定义登录界面类

<Login NormalAppLogin="com.bokesoft.yigo.fxapp.ui.auth.SessionParaLogin"/>

步骤四：获取定义的全局变量:

1、二次开发从 Env.get(String key) 中获取

2、中间层/客户端表达式 GetSessionPara(String key)

PS: key为 Setting文件中Session节点中定义的 SessionParaKey

步骤三：添加i18n目录 根据语种添加界面显示标题.