****

**Original Author: ZhengWei(HY)**

**Create Date: 2013-04**

1. **XJava是什么？**

XJava是Xml + Java的缩写。Xml + Java是一种流行的开发模式。现行流行的开发模式中Struts、Spring、Hibernate都是这样的。它们各有侧重点，各有优势。但，总体的来说都是将一部分可配置的内容使用XML文件配置实现，另一部分业务性强的使用Java语言开发实现。XJava就是这样的开发模式。

1. **XJava的优势**

在MVC模式中，XJava可十分容易的完成Controller层的 XML配置。因为这样的功能Struct、Spring中也有实现，这里就不多言了。

XJava还有更强大的功能，这也是Struts与Spring没有做的到功能。XJava还可以完成View层的构造，及View层中多个View的相互包含、相互嵌套、相互引用，这些复杂的关系全都能在XML配置完成。

XJava使我们轻松实现MVC中的View层和Controller层后，可将我们的关注点及精力重点放在具体业务功能的实现上。

XJava有强大的融合能力。XJava可与Hibernate融合，实现对数据库的访问。如果你感觉Hibernate达不到特定数据库的优化性能，那XJava可轻松的与你自己写的或是其它C3P0数据库访问框架相融合。

XJava在XML配置上比Spring更加灵活、更加多样。甚至于你自己写一个Java类，都可在XJava的XML配置中轻松的实现构造、属性赋值，还有方法的调用及方法返回结果的引用。

XJava的自适应性极强。某种程度上，你会发现XJava是Spring是超级父类，它能实现Spring反射机制，也能实现Spring实现不了的多种多样的关系构造。Struts、Spring的XML配置文件都有固定的DTD格式。但XJava可以不需要，它更加的灵活，它可以根据你的需要、你的Java类类型变化而变化XJava的XML配置文件。

XJava能最大限度的减少Java代码的编程。最大限度的提高XML配置的灵活性、自适应性。如，我们构造一个窗口，无论窗口的布局如何复杂、层次如何多样，当使用XJava配置后，你会惊奇的发现，实现这些功能都不需要写一行Java代码，只要一个XML配置文件时，你就会感觉到XJava的魅力所在。

1. **XJava开发说明**
   1. **Import标记**

此标记类似于Java语言中的引包。用于标记xml文件中统一节点名称的Java类类型。

Import标记的语法为：<import name=”uis” class=”java.util.ArrayList” />

* + 1. **Import : name属性**

此为Import节点标记的属性之一。表示xml文件中的统一节点名称。如<uis>…</uis>

* + 1. **Import : class属性**

此为Import节点标记的属性之一。表示xml文件中统一节点名称的Java类类型。如上面的例子中说明<uis>节点的Java类类型为java.util.ArrayList类型。

* 1. **Map : key 属性**

当节点的Java类类型为 java.util.Map类型时，可以使用此节点属性。Key属性表示其子节点的Java类类型的哪个方法名称作为Map集合put(k ,v)充填方法的关键字入参充填Map集合。

Key的值可以识别getter方法的全方法名称，也可以识别不含get前缀的getter方法名称 。

如果是getter方法的全方法名称，则大小写必须符合Java命令规范，即，必须与Java类中的方法名称一致。

如果是不含get前缀的getter方法名称，其首字母可以大写，也可以小写。

举例1.1：

<import name=”databases” class=”java.util.Hashtable”>

<import name=”database” class=”com.hy.common.db.Database”>

<databases **key=”toString”**>

<database>

<driver>oracle.jdbc.driver.OracleDriver</driver>

<url>jdbc:oracle:thin:@127.0.0.1:1521:ora92</url>

<user>xxx</user>

<password>xxx</password>

</database>

<database>

<driver>com.ibm.db2.jdbc.app.DB2Driver</driver>

<url>jdbc:db2://localhost:5000/sample</url>

<user>xxx</user>

<password>xxx</password>

</database>

</databases>

上面的举例中的<databases **key=”toString”**> 配置，也可以写<databases **key=” driver”**>配置或是<databases **key=” getDriver”**>配置都是可以的，因为<databases>的子节点<database>拥有<driver>节点，即，database的Java类类型的实例拥有一个方法名称为getDriver()的方法。

* 1. **List : add属性**

当节点的Java类类型为 java.util.List类型时，可以使用此节点属性，这是一个隐含属性，即，在实例开发中，不用标明此属性。它表示子节点使用父节点的add方法填充java.util.List集合。

举例1.2

<import name=”dates” class=”java.util.ArrayList”>

<import name=”date” class=”com.hy.common.Date”>

<**dates**>

<date id=”minDate”>

<year>1983</year>

</date>

<date id=”maxDate”>

<year>2100</year>

</date>

</**dates**>

上面的举例中<dates>是一个java.util.List集合。默认使用List.add(o)方法将<date>节点实例化的对象添加到集合中。

* 1. **Constructor标记**

复杂构造器的节点标记。当Java类型的实例在构造时，需要使用非默认（即有入参的）的构造时，使用<Constructor>标记来配置。

Constructor标记的语法为：

<XXX>< Constructor><param1>…</param1>…<paramN>…</paramN></ Constructor></XXX>

举例1.3

<import name=”dimension” class=”java.awt.Dimension” />

<dimension>

<**constructor**>

<int>40</int>

<int>36</int>

</**constructor**>

</dimension>

上面的举例翻译成Java代码如下：

java.awt. Dimension v\_Dimension = new java.awt.Dimension(40 ,36);

* 1. **XJava : class属性**

此属性是一个通用节点属性。表示当前xml节点的Java类类型。当不使用<import>引包节点时，对于个别节点可以使用此属性指明节点的Java类类型。

举例1.4

<dimension **class=”java.awt.Dimension”**>

<constructor>

<int>40</int>

<int>36</int>

</constructor>

</dimension>

上面的举例1.4与举例1.3是实现了相同的功能。但，我们没有对Dimension引包，当我们个别方法使用一下java.awt.Dimension类时，这样的写法更加的方便。

* 1. **XJava : setter属性**

此属性是个通用节点属性。表示父节点添加子节点的Java方法名称。入参数量只能是一个。其List:add属性就是XJava:setter的一种特殊化实现。

举例1.5

<import name=”XPanel” class=”javax.swing.JPanel” />

<import name=”XLabel” class=”javax.swing.JLabel” />

<import name=”XTextField” class=”javax.swing.JTextField” />

<import name=”flowLayout” class=”java.awt.FlowLayout” />

<XPanel **setter=”add”**>

<layout>

<flowLayout>

<alignment ref=”this.LEFT”>

</flowLayout>

</layout>

<XLabel>

<text>江南Style的作者是谁？</text>

</XLabel>

<XTextField>

<columns>16</columns>

<toolTipText>江南Style的作者是谁？</toolTipText>

</XTextField>

</XPanel>

上面的举例创建了一个面板。设置面板的布局为FlowLayout，对齐方式为左对齐。通过面板的add(o)方法添加了一个标签控件和一个文本框控件 。

* 1. **XJava : id属性**

此属性是个通用节点属性。表示节点构造后的Java实例的惟一标识，类似于变量名。有了这个ID变量名后，无论是在其后的xml文件中，还在Java代码中，都是方便的被引用。

举例1.6

<import name=”XPanel” class=”javax.swing.JPanel” />

<import name=”XLabel” class=”javax.swing.JLabel” />

<import name=”XTextField” class=”javax.swing.JTextField” />

<import name=”flowLayout” class=”java.awt.FlowLayout” />

<import name=”date” class=”com.hy.common.Date”>

<date **id=”minDate”**>

<year>1983</year>

</date>

<XPanel setter=”add”>

<layout>

<flowLayout>

<alignment ref=”this.LEFT”>

</flowLayout>

</layout>

<XLabel>

<text>开始时间</text>

</XLabel>

<XTextField>

<columns>16</columns>

<toolTipText>开始时间</toolTipText>

<text ref=”**minDate**.getFull” />

</XTextField>

</XPanel>

上面的举例1.6与举例1.5类似。创建了一个面板。设置面板的布局为FlowLayout，对齐方式为左对齐。通过面板的add(o)方法添加了一个标签控件和一个文本框控件 。不同是，实现了一个时间类型的变量minDate，并通过引用方式<text ref=”**minDate**.getFull” />初始化文本框控件的文本内容。

举例1.7

在Java代码中引用xml中定义的变量minDate的实现代码如下 。

Date v\_MinDate = (Date)XJava.getObject(“minDate”);

可以看，引用起来十分简单。并且可配置性很高，即不用修改Java代码，只配置xml文件就可以实现不同的功能。

* 1. **XJava : ref属性**

此属性是个通用节点属性。表示引用的对象实例是哪谁，与XJava:id属性配合使用。如举例1.6中就出现过XJava:ref属性，可以看到ref不光简单的引用对象minDate，还可以引用minDate.getFull()方法的返回值对象。这是个十分强大的功能。

举例1.8

<import name=”flowLayout” class=”java.awt.FlowLayout” />

<import name=”XPanel” class=”javax.swing.JPanel” />

<flowLayout **id=”LeftFL”**>

<alignment ref=”this.LEFT”>

</flowLayout>

<XPanel>

<layout **ref=”LeftFL”**>

… … … …

</XPanel>

上面举例1.8与举例1.6对比，可以看出，我们可以把布局对象放在<XPanel>的里面，也在放在<XPanel>的外面来实现。在一些通用性高的变量情况，常常使用这样的方式达到公共的目的。

* 1. **XJava : submit属性**

此属性是个通用节点属性。Submit属性值是一个被XJava定义过的XJava:id变量名称。它将这个变量名称下对应的所有子节点中有XJava:id属性的节点实例化的Java对象，保存在一个java.util.Map集合中，此集合的Map.key为XJava:id属性值，Map.value 为节点实例化的Java对象。然后将些Map集合通过Setter 方法传递给指定的对象方法。

举例1.9

<import name=”XPanel” class=”javax.swing.JPanel” />

<import name=”XLabel” class=”javax.swing.JLabel” />

<import name=”XTextField” class=”javax.swing.JTextField” />

<import name="XButton" class="com.hy.common.ui.JButton" />

<import name=”flowLayout” class=”java.awt.FlowLayout” />

<XPanel setter=”add” id=”**QueryPanel**”>

<layout>

<flowLayout>

<alignment ref=”this.LEFT”>

</flowLayout>

</layout>

<XLabel>

<text>开始时间</text>

</XLabel>

<XTextField id=”Param\_BeginDate”>

<columns>16</columns>

<toolTipText>开始时间</toolTipText>

</XTextField>

<XLabel>

<text>结束时间</text>

</XLabel>

<XTextField id=”Param\_EndDate”>

<columns>16</columns>

<toolTipText>结束时间</toolTipText>

</XTextField>

<XButton>

<text>查询</text>

<onClickTransaction>

<event class=”com.hy.ui.event.SubmitActionListener”>

<queryParam **submit=”QueryPanel”** />

</event>

</onClickTransaction>

</XButton>

</XPanel>

上面的举例创建了一个面板。设置面板的布局为FlowLayout，对齐方式为左对齐。通过面板的add(o)方法添加了两个标签控件、两个文本框控件和一个查询按钮控件 。对于查询按钮添加了点击事件处理类” SubmitActionListener”，此类中有一个方法为public void queryParam(Map i\_Map) ;的方法，通过此方法接收集合对象。此集合对象中有两个元素，分别是两个文本对象的实例化对象。

* 1. **Call标记**

方法调用节点标记。调用复杂Java方法或是需要获取Java方法返回值对象的显示时被使用。

Call标记的语法为：

<XXX>

<Call name=”java-method-name” returnid=”java-method-return-object-id”>

<param1>…<param1>

…… ……

<paramN>…</paramN>

</Call>

</XXX>

* + 1. **Call : name属性**

此为Call节点标记的属性之一。表示调用Java类实例的方法名称。也可以是Java类实例的方法的返回值对象实例的方法名称 ，如：name=”toString.trim”。

* + 1. **Call : returnid属性**

此为Call节点标记的属性之一。表示调用Java类实例的方法的返回值对象在XJava中的 id标识。有了此id标识，就可以在其后被引用。

举例1.10

<import name="XImageIcon" class="javax.swing.ImageIcon" />

<import name="XURL" class="java.net.URL" />

<XImageIcon>

<constructor>

<XURL>

<constructor>

<String>classpath:Ubuntu.png</String>

</constructor>

<XURL>

</constructor>

**<call name=” this.getImage.getScaledInstance” returnID=”ximageReturnID”>**

<int>24</int>

<int>24</int>

<int>1</int>

</call>

<image ref=” **ximageReturnID**” />

</XImageIcon>

上面的举例中，创建一个图片对象实例<XImageIcon>，在其构造时又用一个新创建出的<XURL>来构造。再<XImageIcon>构造完成后，调用自身的**” this.getImage.getScaledInstance”**方法，获取返回值对象并用**returnID=” ximageReturnID”**标识。再其后使用setImage(o)方法，将返回值对象填充给自身，完成一个大小为24\*24\*1按钮图片的创建。翻译上面的配置为Java代码如下 ：

ImageIcon v\_ImageIcon = new ImageIcon(new URL(“…/…/Ubuntu.png”));

Image v\_ReturnObj = v\_ImageIcon.getImage.getScaledInstance(24 ,24 ,1);

v\_ImageIcon.setImage(v\_ReturnObj);

有了上面的XJava配置，以后更换图片，调整图片显示大小，特别是整个应用的样式时，只要Copy一个XJava配置，简单修改一下就成，还能保存原有的样式。而不用修改Java代码。

* 1. **this关键字**

此为一个关键字，它即不xml文件中的节点，也不是节点的属性。而是用在节点属性值的关键字。在举例1.10中有出现过此关键字。它表示所在节点的父节点对应的Java类实例的引用。同Java语言中的this关键同样的含义。

* 1. **classpath关键字**

此为一个关键字，它即不xml文件中的节点，也不是节点的属性。而是用在节点属性值的关键字。在举例1.10中有出现过此关键字。它表示当前xml文件在Java工程中的包路径。通过它，我们可以获取同包下的其它资源或是上级包、下级包等相对包路径中的资源信息。

1. **XJava的样例演示**

演示应用1：XReport

应用界面如下 ：



XJava的配置如下 ：

<?xml version=*"1.0"* encoding=*"UTF-8"*?>

<config>

<uis>

<!-- 昨天的零点时间，用于开始时间 -->

<Date class=*"com.hy.common.Date"*>

<call name=*"getPreviousDay"* returnID=*"getPreviousDay"* />

</Date>

<!-- 明天的零点时间，用于结束时间 -->

<Date class=*"com.hy.common.Date"*>

<call name=*"getNextDay"* returnID=*"getNextDay"* />

</Date>

<!-- 工具栏按钮图标: 下一页 -->

<XImageIcon id=*"xiconNextPage"*>

<constructor>

<XURL>

<constructor>

<String>classpath:img/nextPage.png</String>

</constructor>

</XURL>

</constructor>

<call name=*"this.getImage.getScaledInstance"* returnID=*"ximageNextPage"*>

<int>24</int>

<int>24</int>

<int>1</int>

</call>

<image ref=*"ximageNextPage"* />

</XImageIcon>

<!-- 工具栏按钮图标: 最后一页 -->

<XImageIcon id=*"xiconLastPage"*>

<constructor>

<XURL>

<constructor>

<String>classpath:img/lastPage.png</String>

</constructor>

</XURL>

</constructor>

<call name=*"this.getImage.getScaledInstance"* returnID=*"ximageLastPage"*>

<int>24</int>

<int>24</int>

<int>1</int>

</call>

<image ref=*"ximageLastPage"* />

</XImageIcon>

<!-- 工具栏按钮图标: 导出 -->

<XImageIcon id=*"xiconExport"*>

<constructor>

<XURL>

<constructor>

<String>classpath:img/export.jpg</String>

</constructor>

</XURL>

</constructor>

<call name=*"this.getImage.getScaledInstance"* returnID=*"ximageExport"*>

<int>24</int>

<int>24</int>

<int>1</int>

</call>

<image ref=*"ximageExport"* />

</XImageIcon>

<!-- 主窗口中左则的树目录 -->

<XPanel id=*"xpReportList"*>

</XPanel>

<!-- 查询条件面板 -->

<XPanel id=*"xpSelect"* setter=*"add"*>

<layout>

<flowLayout>

<alignment ref=*"this.LEFT"* />

</flowLayout>

</layout>

<XLabel>

<text>时间类型</text>

</XLabel>

<XComboBox id=*"TimeType"*>

<call name=*"addItem"*><String>1</String><String>受理时间</String></call>

<call name=*"addItem"*><String>2</String><String>修复时间</String></call>

<call name=*"addItem"*><String>3</String><String>归档时间</String></call>

</XComboBox>

<XLabel>

<text>开始范围</text>

</XLabel>

<XTextField id=*"BeginTime"*>

<columns>11</columns>

<toolTipText>请输入开始时间</toolTipText>

<text ref=*"getPreviousDay.getFirstTimeOfDay"* />

<call name=*"setOnlyDateTime"* />

<call name=*"turnOnDateTime"* />

</XTextField>

<XLabel>

<text>~</text>

</XLabel>

<XTextField id=*"EndTime"*>

<columns>11</columns>

<toolTipText>请输入结束时间</toolTipText>

<text ref=*"getNextDay.getFirstTimeOfDay"* />

<call name=*"setOnlyDateTime"* />

<call name=*"turnOnDateTime"* />

</XTextField>

<XComboBox id=*"CityCode"*>

<call name=*"addItem"*><String></String> <String> 地市 </String></call>

<call name=*"addItem"*><String>029</String> <String> 西安 </String></call>

<call name=*"addItem"*><String>0919</String><String> 铜川 </String></call>

<call name=*"addItem"*><String>0917</String><String> 宝鸡 </String></call>

<call name=*"addItem"*><String>0910</String><String> 咸阳 </String></call>

<call name=*"addItem"*><String>0913</String><String> 渭南 </String></call>

<call name=*"addItem"*><String>0916</String><String> 汉中 </String></call>

<call name=*"addItem"*><String>0915</String><String> 安康 </String></call>

<call name=*"addItem"*><String>0914</String><String> 商洛 </String></call>

<call name=*"addItem"*><String>0911</String><String> 延安 </String></call>

<call name=*"addItem"*><String>0912</String><String> 榆林 </String></call>

</XComboBox>

<XComboBox id=*"ServiceClassID"*>

<call name=*"addItem"*><String></String> <String>业务类型</String></call>

<call name=*"addItem"*><String>8101</String><String>固定电话</String></call>

<call name=*"addItem"*><String>8105</String><String>专线</String></call>

<call name=*"addItem"*><String>8107</String><String>ADSL</String></call>

<call name=*"addItem"*><String>8108</String><String>LAN</String></call>

<call name=*"addItem"*><String>8115</String><String>ISDN</String></call>

<call name=*"addItem"*><String>8116</String><String>IPTV</String></call>

<call name=*"addItem"*><String>8127</String><String>政企障碍</String></call>

<call name=*"addItem"*><String>8129</String><String>PAS基站</String></call>

<call name=*"addItem"*><String>8117</String><String>光纤专线接入</String></call>

<call name=*"addItem"*><String>8118</String><String>电路连接类</String></call>

<call name=*"addItem"*><String>8120</String><String>小灵通</String></call>

<call name=*"addItem"*><String>8123</String><String>全球眼</String></call>

<call name=*"addItem"*><String>8124</String><String>网络电话</String></call>

<call name=*"addItem"*><String>8125</String><String>视频</String></call>

<call name=*"addItem"*><String>8126</String><String>系统集成(ICT)</String></call>

<call name=*"addItem"*><String>8102</String><String>互联网增值</String></call>

<call name=*"addItem"*><String>8119</String><String>号码信息增值</String></call>

<call name=*"addItem"*><String>8121</String><String>其它增值业务</String></call>

<call name=*"addItem"*><String>8122</String><String>其它</String></call>

</XComboBox>

<XComboBox id=*"FinishMode"*>

<call name=*"addItem"*><String>1</String><String>下派的障碍</String></call>

<call name=*"addItem"*><String>0</String><String>预处理的障碍</String></call>

</XComboBox>

<XComboBox id=*"RepeatAppeal"*>

<call name=*"addItem"*><String> </String><String> </String></call>

<call name=*"addItem"*><String>1</String><String>重复障碍</String></call>

</XComboBox>

<XComboBox id=*"OverTimeLen"*>

<call name=*"addItem"*><String> </String><String> </String></call>

<call name=*"addItem"*><String>1</String><String>超时障碍</String></call>

</XComboBox>

<XComboBox id=*"IsComplaint"*>

<call name=*"addItem"*><String> </String><String>工单性质</String></call>

<call name=*"addItem"*><String>0</String><String>1.普通障碍</String></call>

<call name=*"addItem"*><String>1</String><String>2.超时转投诉</String></call>

</XComboBox>

<XButton id=*"xbSubmit"*>

<text>查询</text>

<enabled>false</enabled>

<onClickTransaction>

<event class=*"com.hy.ui.event.SubmitActionListener"*>

<queryConditions submit=*"xpSelect"* />

</event>

</onClickTransaction>

</XButton>

</XPanel>

<!-- 查询工具栏面板 -->

<XPanel id=*"xpTools"* setter=*"add"*>

<layout />

<XButton id=*"xbNextPage"*>

<toolTipText>下一页</toolTipText>

<enabled>false</enabled>

<icon ref=*"xiconNextPage"* />

<call name=*"setBounds"*>

<int>0</int>

<int>0</int>

<int>24</int>

<int>24</int>

</call>

<onClickTransaction>

<event class=*"com.hy.ui.event.NextPageActionListener"* />

</onClickTransaction>

</XButton>

<XButton id=*"xbLastPage"*>

<toolTipText>最后一页</toolTipText>

<enabled>false</enabled>

<icon ref=*"xiconLastPage"* />

<call name=*"setBounds"*>

<int>26</int>

<int>0</int>

<int>24</int>

<int>24</int>

</call>

<onClickTransaction>

<event class=*"com.hy.ui.event.LastPageActionListener"* />

</onClickTransaction>

</XButton>

<XButton id=*"xbExport"*>

<toolTipText>导出</toolTipText>

<enabled>false</enabled>

<icon ref=*"xiconExport"* />

<call name=*"setBounds"*>

<int>52</int>

<int>0</int>

<int>24</int>

<int>24</int>

</call>

<onClickTransaction>

<event class=*"com.hy.ui.event.ShowExportActionListener"* />

</onClickTransaction>

</XButton>

<!-- 提示信息 -->

<XLabel id=*"xlHintInfo"*>

<call name=*"setBounds"*>

<int>150</int>

<int>0</int>

<int>500</int>

<int>24</int>

</call>

</XLabel>

</XPanel>

<!-- 查询结果列表 -->

<XTable id=*"xtDataList"*>

<showGrid>true</showGrid>

<rowSelectionAllowed>true</rowSelectionAllowed>

<selectionMode>1</selectionMode>

<autoResizeMode>0</autoResizeMode>

</XTable>

<!-- 查询结果滚动面板 -->

<XScrollPane id=*"xpResult"* setter=*"add"*>

<horizontalScrollBarPolicy>30</horizontalScrollBarPolicy>

<call name=*"this.getViewport.add"*>

<component ref=*"xtDataList"* />

</call>

</XScrollPane>

<!-- 工具栏与查询结果的分割 -->

<XSplitPane id=*"xsToolsResult"*>

<orientation ref=*"this.VERTICAL\_SPLIT"* />

<dividerSize>4</dividerSize>

<oneTouchExpandable>false</oneTouchExpandable>

<call name=*"setDividerLocation"*>

<int>25</int>

</call>

<call name=*"add"*>

<component ref=*"xpTools"* />

<object ref=*"this.TOP"* />

</call>

<call name=*"add"*>

<component ref=*"xpResult"* />

<object ref=*"this.BOTTOM"* />

</call>

</XSplitPane>

<!-- 查询条件与内容（工具栏及查询结果）的分割 -->

<XSplitPane id=*"xsContent"*>

<orientation ref=*"this.VERTICAL\_SPLIT"* />

<dividerSize>4</dividerSize>

<oneTouchExpandable>false</oneTouchExpandable>

<call name=*"add"*>

<component ref=*"xpSelect"* />

<object ref=*"this.TOP"* />

</call>

<call name=*"add"*>

<component ref=*"xsToolsResult"* />

<object ref=*"this.BOTTOM"* />

</call>

</XSplitPane>

<!-- 主窗口的左右分割 -->

<XSplitPane id=*"xsMain"*>

<orientation ref=*"this.HORIZONTAL\_SPLIT"* />

<dividerSize>0</dividerSize>

<oneTouchExpandable>false</oneTouchExpandable>

<call name=*"setDividerLocation"*>

<int>0</int>

</call>

<call name=*"add"*>

<component ref=*"xpReportList"* />

<object ref=*"this.LEFT"* />

</call>

<call name=*"add"*>

<component ref=*"xsContent"* />

<object ref=*"this.RIGHT"* />

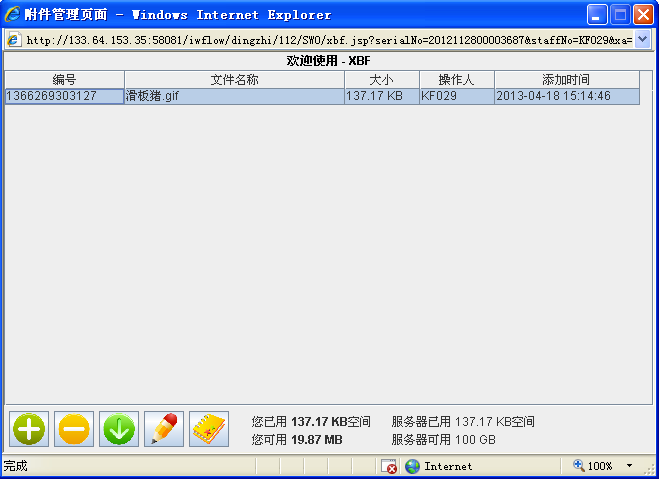
</call>

</XSplitPane>

</uis>

</config>

演示应用2：XBF



XJava的配置如下 ：

<?xml version=*"1.0"* encoding=*"UTF-8"*?>

<config>

<uis>

<!-- 内部参数信息 -->

<XParams key=*"name"*>

<XParam id=*"MaxUseSpaceSize"*>

<name>MaxUseSpaceSize</name>

<value>20</value>

<comment>每用户，最大可使用的空间大小(单位MB)</comment>

</XParam>

<XParam id=*"MaxFileCount"*>

<name>MaxFileCount</name>

<value>5</value>

<comment>每用户，最大可上传的文件个数</comment>

</XParam>

<XParam id=*"MaxSpaceSize\_All"*>

<name>MaxSpaceSize\_All</name>

<value>102400</value>

<comment>所有用户，最大空间大小(单位MB)</comment>

</XParam>

<XParam id=*"IsLog"*>

<name>IsLog</name>

<value>1</value>

<comment>是否记录日志</comment>

</XParam>

<XParam id=*"XWidth"*>

<name>XWidth</name>

<value>650</value>

<comment>窗体宽度</comment>

</XParam>

<XParam id=*"XHeight"*>

<name>XHeight</name>

<value>400</value>

<comment>窗体高度</comment>

</XParam>

</XParams>

<!-- 按钮图标: 上传按钮 -->

<XImageIcon id=*"xiconUpLoad"*>

<constructor>

<XURL>

<constructor>

<String>classpath:img/UpLoad.png</String>

</constructor>

</XURL>

</constructor>

<call name=*"this.getImage.getScaledInstance"* returnID=*"ximageUpLoad"*>

<int>32</int>

<int>32</int>

<int>1</int>

</call>

<image ref=*"ximageUpLoad"* />

</XImageIcon>

<!-- 按钮图标: 删除按钮 -->

<XImageIcon id=*"xiconDelete"*>

<constructor>

<XURL>

<constructor>

<String>classpath:img/Delete.png</String>

</constructor>

</XURL>

</constructor>

<call name=*"this.getImage.getScaledInstance"* returnID=*"ximageDelete"*>

<int>32</int>

<int>32</int>

<int>1</int>

</call>

<image ref=*"ximageDelete"* />

</XImageIcon>

<!-- 按钮图标: 下载按钮 -->

<XImageIcon id=*"xiconDownLoad"*>

<constructor>

<XURL>

<constructor>

<String>classpath:img/DownLoad.png</String>

</constructor>

</XURL>

</constructor>

<call name=*"this.getImage.getScaledInstance"* returnID=*"ximageDownLoad"*>

<int>32</int>

<int>32</int>

<int>1</int>

</call>

<image ref=*"ximageDownLoad"* />

</XImageIcon>

<!-- 按钮图标: 备注按钮 -->

<XImageIcon id=*"xiconComment"*>

<constructor>

<XURL>

<constructor>

<String>classpath:img/Comment.png</String>

</constructor>

</XURL>

</constructor>

<call name=*"this.getImage.getScaledInstance"* returnID=*"ximageComment"*>

<int>32</int>

<int>32</int>

<int>1</int>

</call>

<image ref=*"ximageComment"* />

</XImageIcon>

<!-- 按钮图标: 切换到日志界面的按钮 -->

<XImageIcon id=*"xiconOpenLog"*>

<constructor>

<XURL>

<constructor>

<String>classpath:img/OpenLog.png</String>

</constructor>

</XURL>

</constructor>

<call name=*"this.getImage.getScaledInstance"* returnID=*"ximageOpenLog"*>

<int>32</int>

<int>32</int>

<int>1</int>

</call>

<image ref=*"ximageOpenLog"* />

</XImageIcon>

<!-- 列表的标题 -->

<Titles id=*"titles"* class=*"java.util.ArrayList"*>

<String>编号</String>

<String>文件名称</String>

<String>大小</String>

<String>操作人</String>

<String>添加时间</String>

</Titles>

<!-- 列表的大小 -->

<Titles id=*"titles\_Size"* class=*"java.util.ArrayList"*>

<String>120</String>

<String>220</String>

<String>70</String>

<String>70</String>

<String>145</String>

</Titles>

<!-- 提示信息 -->

<XLabel id=*"xlHintInfo"*>

<horizontalAlignment>0</horizontalAlignment>

<text>欢迎使用</text>

</XLabel>

<!-- 顶部面板 -->

<XPanel id=*"xpTop"*>

<layout>

<borderLayout />

</layout>

<call name=*"add"*>

<String>Center</String>

<component ref=*"xlHintInfo"* />

</call>

</XPanel>

<!-- 数据列表 -->

<XTable id=*"xtDataList"*>

<showGrid>true</showGrid>

<rowSelectionAllowed>true</rowSelectionAllowed>

<selectionMode>1</selectionMode>

<autoResizeMode>0</autoResizeMode>

<call name=*"this.getSelectionModel.addListSelectionListener"*>

<event class=*"com.hy.ui.event.TableSelectionListener"* />

</call>

<call name=*"this.addMouseMotionListener"*>

<event class=*"com.hy.ui.event.TableMouseMotionListener"* />

</call>

</XTable>

<!-- 数据列表的滚动面板 -->

<XScrollPane id=*"xpDataList"* setter=*"add"*>

<horizontalScrollBarPolicy>30</horizontalScrollBarPolicy>

<toolTipText>拖放文件到此，可上传多个文件。</toolTipText>

<call name=*"this.getViewport.add"*>

<component ref=*"xtDataList"* />

</call>

</XScrollPane>

<!-- 按钮的容器边界 -->

<Insets id=*"Insets"* class=*"java.awt.Insets"*>

<constructor>

<int>0</int>

<int>0</int>

<int>0</int>

<int>0</int>

</constructor>

</Insets>

<!-- 按钮的大小 -->

<Dimension id=*"Dimension"* class=*"java.awt.Dimension"*>

<constructor>

<int>40</int>

<int>36</int>

</constructor>

</Dimension>

<!-- 操作面板。如浏览、添加操作 -->

<XPanel id=*"xpOperate"* setter=*"add"*>

<layout>

<flowLayout>

<alignment ref=*"this.LEFT"* />

</flowLayout>

</layout>

<XButton id=*"xbSubmitFile"*>

<toolTipText>上传按钮。请选择要上传的文件。拖放文件到此，可上传多个文件。</toolTipText>

<enabled>false</enabled>

<icon ref=*"xiconUpLoad"* />

<margin ref=*"Insets"* />

<preferredSize ref=*"Dimension"* />

<onClickTransaction>

<event class=*"com.hy.ui.event.SubmitActionListener"* />

</onClickTransaction>

</XButton>

<XButton id=*"xbDeleteFile"*>

<toolTipText>删除按钮。删除列表选中的文件。支持多选。</toolTipText>

<enabled>false</enabled>

<icon ref=*"xiconDelete"* />

<margin ref=*"Insets"* />

<preferredSize ref=*"Dimension"* />

<onClickTransaction>

<event class=*"com.hy.ui.event.DeleteActionListener"* />

</onClickTransaction>

</XButton>

<XButton id=*"xbDownLoadFile"*>

<toolTipText>下载按钮。下载列表选中的文件。支持多选。</toolTipText>

<enabled>false</enabled>

<icon ref=*"xiconDownLoad"* />

<margin ref=*"Insets"* />

<preferredSize ref=*"Dimension"* />

<onClickTransaction>

<event class=*"com.hy.ui.event.DownLoadActionListener"* />

</onClickTransaction>

</XButton>

<XButton id=*"xbCommentFile"*>

<toolTipText>备注按钮。对列表选中的文件添加或编辑备注文字。支持多选。</toolTipText>

<enabled>false</enabled>

<icon ref=*"xiconComment"* />

<margin ref=*"Insets"* />

<preferredSize ref=*"Dimension"* />

<onClickTransaction>

<event class=*"com.hy.ui.event.ShowCommentActionListener"* />

</onClickTransaction>

</XButton>

<XButton id=*"xbOpenLog"*>

<toolTipText>日志按钮。查看各种操作的操作日志。</toolTipText>

<enabled>false</enabled>

<visible>false</visible>

<icon ref=*"xiconOpenLog"* />

<margin ref=*"Insets"* />

<preferredSize ref=*"Dimension"* />

<onClickTransaction>

<event class=*"com.hy.ui.event.ShowLogActionListener"* />

</onClickTransaction>

</XButton>

<!-- 每位用户的使用空间信息 -->

<XLabel id=*"xlUseSpaceInfo"*>

<horizontalAlignment>2</horizontalAlignment> <!-- SwingConstants.LEFT -->

<font>

<constructor>

<String ref=*"xlHintInfo.getFont.getName"* />

<int>0</int>

<int>12</int>

</constructor>

</font>

<call name=*"setSize"*>

<int>500</int>

<int>80</int>

</call>

</XLabel>

</XPanel>

<!-- 主面板 -->

<XPanel id=*"xpMain"*>

<layout>

<borderLayout />

</layout>

<call name=*"add"*>

<String>North</String>

<component ref=*"xpTop"* />

</call>

<call name=*"add"*>

<String>Center</String>

<component ref=*"xpDataList"* />

</call>

<call name=*"add"*>

<String>South</String>

<component ref=*"xpOperate"* />

</call>

</XPanel>

</uis>

</config>