## 1、JavaScript Controller

### 1）JavaScript Controller是一种新的代码管理方式。

要看懂一个view.xml文件中究竟定义了哪些事件声明是一件麻烦的事情，必须不断的点击各个控件去查看各个事件下的代码。

JavaScript Controller的做法是把所有的事件代码集中到一个和view.xml同名位置的js文件中，同时利用JavaScript中的“Annotation”来完成事件的挂接。

/\*\* @Bind #buttonOK.onClick \*/

function buttonOK() {

alert("buttonOK Clicked.");

}

/\*\* @Bind #dsPeople.onReady \*/

function dsPeopleReady(dsPeople) {

dsPeople.insert();

}

@Bind后面的表达式与View.get()方法中的表达式高度相似，可以利用这种“Annotation”批量的为一组控件定义事件：

/\*\* @Bind ^readOnlyEditor.onCreate \*/ 绑定tags=readOnlyEditor的控件

function setReadOnly(self) {

self.set("readOnly", true);

}

并不只是与View同名的js文件可以成为JavaScript Controller，所有通过View的javascript属性装载的js文件都支持这种声明。

### 2）匿名事件监听器

有时候为每一个事件监听器命名是一件费心的事，而且在大部分情况下这个方法的名称是没有任何实际作用的。JavaScript Controller提供了对匿名事件监听器的支持。

/\*\* @Bind #buttonOK.onClick \*/

!function() {

alert("buttonOK Clicked.");

}

/\*\* @Bind ^readOnlyEditor.onCreate \*/

!function(self) {

self.set("readOnly", true);

}

/\*\* @Bind view.onReady \*/

!function() {

alert("${request.getRequestURI()}");

}

### 3）声明式公用方法

有时候需要在JavaScriptController文件中定义一些公用的方法。

默认情况下JavascriptController中的所有方法都是私有的，外部不可见的。同时Dorado7.1提供了两个"Annotation"用于声明方法的可见性——@View和@Global。@View用于标注某个方法在View中可见，@Global用于标注某个方法全局可见。例如：

/\*\* @View \*/

function showMessage(text) {

dorado.MessageBox.alert(text);

}

此时showMessage()已经被注册成了档期啊View对象的一个方法，可以在其他view.xml和js文件中通过view.showMessage()来调用它。

## 2、Dorado对象监听器

### 1）普通的对象监听器

Dorado7中的很多对象都支持名为listener的属性，可以在属性中定义一到多个的Listener用于监听对象的初始化过程。目前支持此类监听的对象包括：

* DataType
* DataProvider
* DataResolver
* ViewConfig
* View
* Component（即所有的控件）

可监听的动作包括：

* beforeInit在对象将要被初始化之前触发，此时系统为尚未将XML中配置的信息初始化到对象中。
* onInit在对象将被初始化之后触发，此时系统已将XML配置中的信息初始化到对象中。

定义一个监听器的方法与大多数自动方法适配相仿，同时Dorado还在此处提供了自动根据方法名监听动作的功能。具体的规则是如果方法名是以before开头的，那么Dorado会认为监听beforeInit动作，否则方法监听到的将是onInit动作。例如下面的方式将可以监听某按钮的beforeInit动作：

public void beforeInit(Button button) {

// your code

}

（注：对于beforeInit的监听器而言，我们还可以为监听方法提供一个逻辑型的返回值，当方法返回false时，表示将要终止该对象的初始化动作，其后的Listener也将被一同跳过。如果方法不支持返回值或者返回的不是逻辑型的false值则表示继续执行后续的初始化动作。）

### 2）全局对象监听器

全局监听器可以批量的监听上面列出的所有对象或其中的一部分对象。不支持方法自动适配，要定义一个全局对象监听器必须继承Dorado7提供的com.bstek.dorado.data.listener.GenericObjectListener<T>类。该类包含3个属性：

* order-监听优先级，数字越小越先被触发，系统默认的order是999。
* pattern-要监听的范围，可使用通配符，不定义则表示全部拦截。
* excludePattern-不监听的范围，可使用通配符。

通过pattern和excludePattern并不是唯一的用来设定监听范围的方法，更常用的是利用GenericObjectListener类的泛型。如下确定该监听器只能监听系统中所有View的初始化动作：

public class ViewListener extends GenericObjectListener<View> {

@Override

public boolean beforeInit(View view) throws Exception {

// your code

}

@Override

public void onInit(View view) throws Exception {

// your code

}

}

向系统中注册一个全局对对象监听器的方法：

<bean parent="dorado.genericObjectListenerRegister">

<property name="listener">

<bean class="test.ViewListener" >

<property name="order" value="1" />

<property name="pattern" value="\*" />

<property name="excludePattern" value="com.bstek.public.\*" />

</bean>

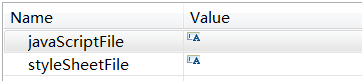
</property>

</bean>

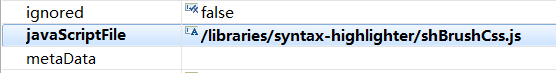
## 3、View中如何引入第三方的资源文件

### 1）方法1：javascriptFile和styleSheetFile

对于一个dorado的view，可以通过View提供的两个属性引入外部第三方库的js库以及对应css等资源文件。



例如在sample-center的view中引入一个js：

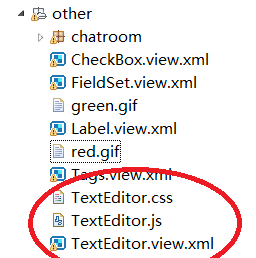


如果需要引入多个js，逗号分隔就可以。

资源文件也可以放在src下，配置如下：



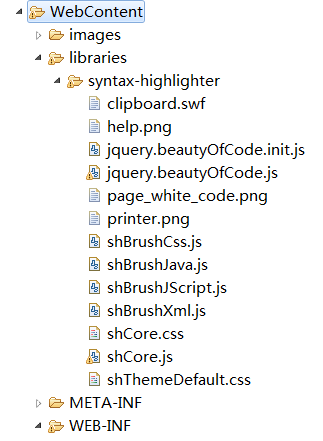
（注：上面所说的是第三方资源，如果仅仅是想给当前View添加自定义的js或css，则可以采用同名规则，View会自动加载）



### 2）方法2：packages-config.xml

多个View需要引入相同的js和css配置文件，并且这些js和css文件并不止一个。这种情况使用packages-config.xml配置方式。该文件主要用于系统资源文件的打包处理。

如sample-center中在根目录中添加了如下的syntax-highlighter资源目录，把相关的js和css都复制于此。



在packages-config.xml中做出如下配置：

<Packages>

<Package name="common" pattern="js"></Package>

<Package name="syntax-highlighter-css" pattern="css"

baseUri=">libraries/syntax-highlighter">

/libraries/syntax-highlighter/shCore,

/libraries/syntax-highlighter/shThemeDefault

</Package>

<Package name="syntax-highlighter" pattern="js"

depends="jquery,syntax-highlighter-css">

/libraries/syntax-highlighter/jquery.beautyOfCode,

/libraries/syntax-highlighter/jquery.beautyOfCode.init,

/libraries/syntax-highlighter/shCore,

/libraries/syntax-highlighter/shBrushJava,

/libraries/syntax-highlighter/shBrushJScript,

/libraries/syntax-highlighter/shBrushXml

</Package>

</Packages>

文件中添加了两个package（名称为syntax-highlighter-css和syntax-highlighter），其pattern用于定义资源类型，分为为css和js。

view如何引用其中的资源：



view提供了packages属性，用于引入packages-config.xml中配置的资源包，如果有多个资源包，逗号分隔。

depends属性

package节点中depends属性的含义：表示这个packages依赖jquery和syntax-highlighter-css两个package，其中jquery是dorado核心包默认提供的，直接使用就可以了。这样当一个View通过packages属性引入一个资源包的时候，会根据depends设定自动加载依赖的资源包。

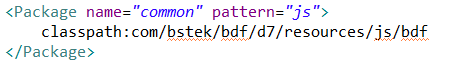
（注：javaScriptFile与styleSheetFile中的资源会被合并到页面中，而采用package机制引入的资源不会合并，而是作为页面的资源引用机制加载。）

baseUri属性

package内部定义的资源默认采用Resource机制加载，如果我们定义了baseUri就表示从一个相对的web应用的资源目录的根路径。其中“>”符号表示从web应用的根目录开始计算。package中资源的加载也支持其他的方式classpath。

common

如果希望所在View中都加载的js资源，可以通过名称为common的package定义，如：



这样View中的packages即使不定义，dorado的资源管理机制也会自动的配置中的js资源加载到所有的view中。

min压缩

如果希望支持资源包压缩功能，从而减轻网络资源下载的数据量，则需要将相关的css和js利用第三方的工具包（例如：YUI Compressor）做好min.css或min.js文件，并放在相应的资源文件夹下。这样一旦采用production运行模式，资源下载管理器会自动寻找对应的min类型的文件下载，只有在找不到的情况下才选用未做min处理的资源文件下载。

### 3）方法3：$load()

$load("http://map.qq.com/api/js?v=2.exp",{

type:"js",

callback:function(){

console.log("加载成功");

}

})

### 4）方法4：用JSP或Velocity等页面模板

可以用JSP或Velocity等页面模板技术，这样引用资源在页面模板中定义就可以。