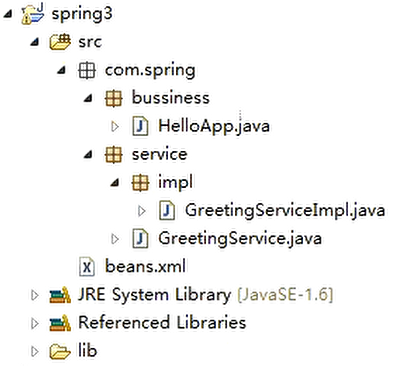
## Spring项目的目录结构



## Spring3 Annotation配置

<context:component-scan base-package="com.spring">

表示自动扫描com.sping下的包，去找到使用：

@Component（一般用途的组件的典型化注解）

@Repository（持久层中的一个Dao组件）

@Service（服务层中的一个服务组件）

@Controller（表现层中的一个控制组件）

* 了解Dorado7的设计诉求以及实现原理。
* 并借此掌握Dorado7中的基本概念和基本术语，例如：立体数据模型，DataSet，DataPath，DataType，DataProvider，DataResolver等。
* 提供一定范例，通过这些规范掌握Dorado中基本的页面开发技术和IDE操作技巧，包括AJAX技术交互机制，立体数据模型的应用和智能方法适配等。
* 增加对JavaScript技术的了解和加深对BS系统的理解。

## 1、产品定位

展现中间件，Dorado致力于解决展现层的中间件的开发，其核心价值：更好的前端页面、更高的开发效率。

适合基于BS架构的管理类应用，但是不太适合面向互联网的开放式站点。主要原因：

* 界面元素的风格-信息管理类的维护界面与面向互联网的界面元素风格存在较大的差异性。
* 对搜索引擎的友好度-对于这种注重AJAX操作的前端开发技术本身就与搜索引擎存在不适应的情况。

Dorado7最适合采用ORM技术实现数据持久层的开发模式，它在设计之初就是为了ORM考虑的，但是考虑到Dorado5那种JDBC类型的快速开发方式依然能满足不同类型客户的需求，在Dorado7中依然提供了基于JDBC的开发模式。

## 2、价值取向

### 更好用的前端界面

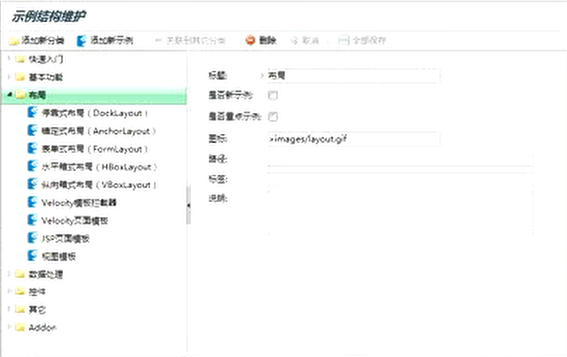
* 丰富的界面控件（70多个控件）
* 支持各种主流浏览器（IE、Chrome、Firefox、Safari、Opera等）
* 比Dorado5更加Ajax化，更多的采用异步处理，可以更好的支持异步服务端通讯，从而进一步提高界面操作的友好性。
* JS、CSS的按需装载和Server端自动合并。

### 更高的开发效率

* 基于Eclipse的IDE。
* Dorado产品一贯的特色——以XML配置简化前端界面的定义，并彻底以XML替代传统JSP。
* 以数据模型为核心的开发模式。如：数据敏感空间，立体数据模型遵循“约定优于配置”的设计思想，例如自动方法适配。

以XML替代传统JSP的原因：

1）基于管理类应用界面的特点，采用控件+布局管理器的界面定义模式最为高效。如Swing、SWT等，如下图的界面风格：



这种类型的界面风格有以下几个特征：

* 信息量大，对控件利用率的要求高；
* 需要很好的适应各种分辨率；
* 界面元素相对程式化。

2）有利于简化界面开发过程和降低后期维护成本；基于HTML的开发模式页面代码冗长且不易维护；基于HTML的开发模式对开发者要求较高，需时刻考虑HTML代码对各种浏览器的兼容性。

注：如有需要，开发者有多种方式回到基于HTML的开发模式下。

## 3、以数据模型为核心的开发模式

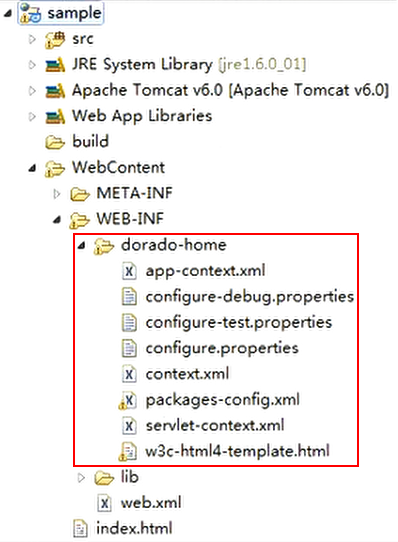
Dorado主张将数据与表现形式分离，实现这种分离，需要几个概念：

* 数据载体——DataSet
* 数据感知控件——DataGrid、TextEditor、…

数据模型驱动型控件集的优点：

* 开发者日常接触的API的数量大大减少
* 数据模型中提供了很多附加的功能，例如：状态管理、翻页、校验、提交等等。可以在很大程度上简化开发。
* 培养开发人员以更贴近业务逻辑的角度思考问题，始终把精力放在核心的业务逻辑的实现上，而不必因为各种零散的界面特效而思维发散。
* 页面代码更加集中，所有更加容易阅读和维护。大部分的页面代码会围绕数据模型展开。
* 当界面设计改变时，重构变得更加简单。因为数据模型几乎不需要改变，开发者需要的知识改动一些其他的数据感知控件。

## 4、Dorado项目基本结构和配置



web.xml

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

<web-app version="3.0" xmlns="http://java.sun.com/xml/ns/javaee"

xmlns:xsi=<http://www.w3.org/2001/XMLSchema-instance>

xsi:schemaLocation="http://java.sun.com/xml/ns/javaee

http://java.sun.com/xml/ns/javaee/web-app\_3\_0.xsd">

<listener>

<listener-class>com.bstek.dorado.web.servlet.SpringContextLoaderListener</listener-class>

</listener>

<servlet>

<servlet-name>doradoServlet</servlet-name>

<servlet-class>com.bstek.dorado.web.servlet.DoradoServlet</servlet-class>

<load-on-startup>1</load-on-startup>

</servlet>

<servlet-mapping>

<servlet-name>doradoServlet</servlet-name>

<url-pattern>\*.d</url-pattern>

</servlet-mapping>

<servlet-mapping>

<servlet-name>doradoServlet</servlet-name>

<url-pattern>\*.c</url-pattern>

</servlet-mapping>

<servlet-mapping>

<servlet-name>doradoServlet</servlet-name>

<url-pattern>\*.dpkg</url-pattern>

</servlet-mapping>

<servlet-mapping>

<servlet-name>doradoServlet</servlet-name>

<url-pattern>/dorado/\*</url-pattern>

</servlet-mapping>

</web-app>

### SpringContextLoaderListener

Dorado7将Spring作为自己比较底层的依赖包，因此在web.xml中可以看到由dorado提供但却以Spring命名的listener：SpringContextLoaderListener，该listener是继承Spring中标准的ContextLoaderListener的一个类，用于装载dorado的一些基础配置。

### DoradoServlet

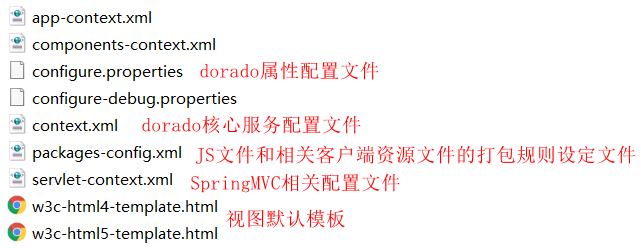
doradoServlet用于提供dorado引擎服务的Servlet，也可以用于Spring的MVC服务。

|  |  |
| --- | --- |
| 拦截字符串 | 说明 |
| \*.d | dorado最基本的服务，如视图请求，AJAX调用请求 |
| \*.dpkg | dorado的package资源请求，根据dorado资源打包方式，可包含的资源有js，css等，dorado中资源的按需装载和自动合并功能都是通过这个后缀实现的。 |
| /dorado/\* | dorado自身的服务：控制台服务，对外界提供的特殊服务：如为IDE提供的服务。 |

### doradohome

dorado工程在WEB-INF目录下会包含一个doradohome的文件夹，用于定义dorado的一些系统全局配置和关键的spring配置。

doradohome下包含的文件：

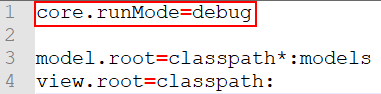


## 5、运行模式

### 设计说明

dorado提供了运行模式处理机制，它允许你根据不同的运行环境定义名称不同的配置文件，而在最终发布时，我们只要调整一个系统全局变量就可以进行切换。这样我们从一种运行环境切换到另一种环境时，系统发布人员只要修改一个变量的值。

dorado7通过configure.properties文件中runMode的值做运行模式的切换，如图：



默认提供3种运行机制：production（生产环境）、debug（开发和调试）、test（测试）。如果不指定运行模式，默认就是production运行模式，也可自定义运行模式。

## 6、配置装载规则

configure.properties是首先被装载的属性文件，如果发现core.runMode做了设定，则系统会自动查找configure-xxx.properties文件，并加载。

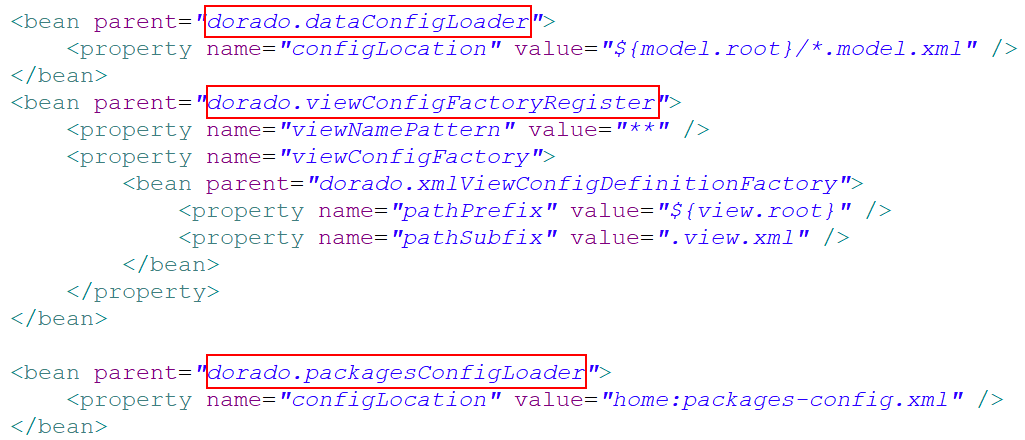
产品正式发布的时候建议将core.runMode的值清空，默认情况下如果dorado发现这个值为空，就自动的以production模式运行。

### dorado默认提供的常用配置属性

| **属性名** | **默认值** | **说明** |
| --- | --- | --- |
| core.runMode | production | 生产环境 |
| view.debugEnabled | false | 是否允许启用Dorado提供的Debugger工具，如果启用，则在浏览器中的视图中按F2快捷按钮就可以打开Debugger工具 |
| view.useMinifiedJavaScript | true | 是否使用压缩后的JavaScript，一般产品调试时采用非压缩状态，有利于调试，而正式发布后采用压缩状态，有利于降低网络的数据流量，提供性能 |
| view.useMinifiedStyleSheet | true | 是否使用压缩后的Style，原理同上。 |
| view.outputPrettyJson | false | 是否格式化输出JSON数据，原理同上。 |
| view.skin | default | 默认的皮肤 |
| view.mergeJavaScript | true | 是否合并JavaScript，正式发布后建议采用合并，原理同上 |
| view.mergeStyleSheet | true | 是否合并CSS文件，正式发布后建议采用合并，原理同上 |
| view.useRandomStringAlias | true | 为LoadRunner这类压力测试工具专门设计的参数  是否启用随机别名机制,例如作为服务的view的名字 |
| core.classPathResourceReloadable | false | 主要针对资源类文件的加载，如js,css等，一般用在deubg模式，避免重新发布和重启 |
| data.config.autoReloadEnabled | false | 针对doradohome文件的重新加载，也适用于debug模式 |
| data.config.autoRecalculatePaths | false | autoRecalculatePaths与autoReloadEnabled类似，处理pathPrefix |
| view.showExceptionStackTrace | false | 是否显示ExceptionStackTrace |

## 7、context.xml基本配置

核心服务配置文件



dorado.dataConfigLoader

设定数据模型文件的存放位置，默认该位置为src下的models这个package下，所有的数据模型文件都以.model.xml结尾。dorado7会自动装载这些模型文件。

dorado.viewConfigFactoryRegister

设定dorado视图文件与请求URL的匹配规则的配置文件，默认的视图文件都以.view.xml文件结尾。

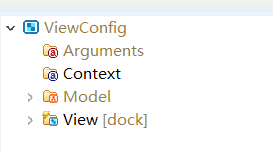
dorado.packageConfigLoader

设定dorado中js和css资源打包规则的配置文件的存放位置。

## 8、View配置文件基本结构

MainView.view.xml视图配置文件

一个空白的视图配置文件共有四个节点：



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

<ViewConfig>

<Arguments/>

<Context/>

<Model/>

<View/>

</ViewConfig>

### Arguments

* 定义参数，可在View的其他地方来引用这个参数，在特定情况下可以简化配置，如提取公用信息配置在arguments中。另外arguments的值也可以从外部传入。这种参数可在当前的View配置文件中通过EL表达式引用。引用方法：通过${arguments.foo}的形式引用。

### Context

* 对应为DoradoContext的VIEW范围上下文对象，它在View的整个生命周期中中无论在Browser端还是Server端，我们都可以很方便的访问和存储其中的数据。
* 代码编写方式与map类似，使用时我们可以再发送一个ajax请求，服务端就可以根据其key值取到刚才存入的字符串。
* 同时我们也可以在这次ajax请求的服务端java代码中将一个字符串存入Context，然后在这次ajax请求结束后，我们可以在Browser端对应这次ajax请求的回调函数或对应执行成功的事件中通过java端设定的key获取到返回的值。
* 使用方法：Server端通过context.getAttribute(DoradoContext.VIEW,”foo”)读取。Client端通过view.get(“context”).get(“foo”)或者view.get(“context.foo”)读取。

### Model

* Model是View的私有模型对象，View可以在此处定义自己私有的模型对象覆盖models中的某个全局的模型对象，并进行个性化设定。

### View

* View节点是经常使用的一个节点，dorado7中的控件都是添加到这个节点下的。