# 设计指南

## 概述

**解决问题**

1，代码中充满硬编码，系统移植困难，影响各省版本的上线。

2，配置来源的多样性。

3，访问来源的多样性。

4，访问接口的不一致性。

5，配置的无约定，造成代码混乱，无法统一。

## 整体设计图



## 初始化流程图



## 请求配置信息流程图



## 设计图

### 组件图



### 配置核心组件



|  |  |
| --- | --- |
| **类名** | **描述** |
| Configuration | 统一的配置访问接口 |
| AbstractConfiguration | 抽象的配置访问接口，自定义的配置访问器一般实现它 |
| MapConfiguration | Map形式的配置访问接口，支持从System、Map、Properties中获取配置信息 |
| DataBaseConfiguration | 数据库形式的配置访问接口 |

### 配置适配器组件



|  |  |
| --- | --- |
| **类名** | **描述** |
| AppConfigAdapter | 配置访问适配器接口, 可扩展 |
| JSAppConfigActionAdapter | JS访问配置信息的适配器 |
| ELAppConfigServletAdapter | JSP页面访问配置信息的适配器 |
| SpringStaticAppConfigAdapter | Java代码访问配置信息的适配器 |

### 配置异常组件

|  |  |
| --- | --- |
| **类名** | **描述** |
| ConfigurationException | 基础配置异常 |
| ConversionException | 数据类型转换异常 |
| NoSuchElementException | 找不到目标元素异常 |

# 开发指南

## 组件配置

在Spring的配置文件中，配置应用程序配置信息集合。其它配置器例如数据库配置器类似

|  |
| --- |
| <util:map id=*"appConfigs"*>  <entry key=*"oscacheConfig"*>  <bean id=*"testConfig"* class=*"com.tydic.commons.configuration.MapConfiguration"*>  <constructor-arg index=*"0"* value=*"oscache.properties"*></constructor-arg>  </bean>  </entry>  </util:map>  <bean class=*"com.tydic.commons.configuration.adapter.SpringStaticAppConfigAdapter"* factory-method=*"newInstance"*>  <property name=*"appConfigs"* ref=*"appConfigs"*></property>  </bean> |

## 前端使用配置组件

配置Spring中定义Action的Bean

|  |
| --- |
| <bean id=*"appconfigAction"* class=*"com.tydic.commons.configuration.adapter.JSAppConfigActionAdapter"*>  <property name=*"appConfigs"* ref=*"appConfigs"*></property>  </bean> |

配置Struts中Action

|  |
| --- |
| <package name=*"test"* extends=*"struts-default"* namespace=*"/"*>  <action name=*"appconfig"* method=*"loadConfig"* class=*"appconfigAction"*>  </action>  </package> |

引入必须的javascript

|  |
| --- |
| <script type=*"text/javascript"* src=*"js/jquery-1.5.2.min.js"*></script>  <script type=*"text/javascript"* src=*"js/appconfig.js"*></script> |

异步访问

|  |
| --- |
| AppConfig.init("/shop/appconfig.action");  AppConfig.loadConfig({  configNames:["oscacheConfig"],  async:**true**,  callback:**function**(config){  alert(config.oscacheConfig["cache.capacity"]);  }  }); |

同步访问

|  |
| --- |
| AppConfig.loadConfig({  configNames:["oscacheConfig"],  async:**false**  });  alert(AppConfig.oscacheConfig["cache.capacity"]); |

## JSP中的使用配置组件

配置web.xml

|  |
| --- |
| <servlet>  <servlet-name>AppConfigServlet</servlet-name>  <servlet-class>com.tydic.commons.configuration.adapter.ELAppConfigServletAdapter</servlet-class>  <load-on-startup>1</load-on-startup>  </servlet> |

JSP中使用方式一

|  |
| --- |
| ${appConfigs.oscacheConfig.map["cache.capacity"]} |

JSP中使用方式二

|  |
| --- |
| <%=SpringStaticAppConfigAdapter .getConfiguration("oscacheConfig").getProperty("cache.capacity")%> |

## JAVA中的使用配置组件

|  |
| --- |
| SpringStaticAppConfigAdapter .getConfiguration("oscacheConfig").getProperty("cache.capacity") |

# 附录

## 附录一 设计类图



## 附录二 相关依赖包



## 附录三 API文档、JAR包及源码包

