1. **配置Tomcat**
2. **导lib包**
3. **配置web.xml**

spring mvc 的配置 3x 1p 简称

<!--第一xml 开始-->

**<!--设置web应用的上下文参数 应用的 位置 -->**

<context-param>

<param-name>contextConfigLocation</param-name>

<param-value>classpath:applicationContext.xml</param-value>

</context-param>

<!--使用spring提供的监听器加载上下文的配置文件 -->

<listener>

<listener-class>org.springframework.web.context.ContextLoaderListener</listener-class>

</listener>

<!--第一xml 结束-->

<!--第二个xml 开始-->

**<!--配置spring mvc 的 前端控制器,拦截所有的 请求-->**

<servlet>

<servlet-name>dispatcherServlet</servlet-name>

<servlet-class>org.springframework.web.servlet.DispatcherServlet</servlet-class>

<init-param>

<param-name>contextConfigLocation</param-name>

<param-value>classpath:spring-mvc.xml</param-value>

</init-param>

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

</servlet>

<servlet-mapping>

<servlet-name>dispatcherServlet</servlet-name>

<url-pattern>/</url-pattern>

</servlet-mapping>

<!--第二个xml 结束-->

**<!--配置编码过滤器-->**

<filter>

<filter-name>CharacterEncodingFilter</filter-name>

<filter-class>org.springframework.web.filter.CharacterEncodingFilter</filter-class>

<init-param>

<param-name>encoding</param-name>

<param-value>utf-8</param-value>

</init-param>

<init-param>

<param-name>forceRequestEncoding</param-name>

<param-value>true</param-value>

</init-param>

<init-param>

<param-name>forceResponseEncoding</param-name>

<param-value>true</param-value>

</init-param>

</filter>

<filter-mapping>

<filter-name>CharacterEncodingFilter</filter-name>

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

</filter-mapping>

1. **配置src\applicationContext.xml**

**<!--配置包的扫描器-->**

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

1. **配置src\spring-mvc.xml**

**<!--配置包的扫描器-->**

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

**<!--开启springmvc的注解-->**

<mvc:annotation-driven/>

**<!--视图解析器** 他用来 告知dispatchServlet test02.jsp 的位置，以及后缀-->

<bean class="org.springframework.web.servlet.view.InternalResourceViewResolver">

<property name="prefix" value="/WEB-INF/pages/"></property>

<property name="suffix" value=".jsp"></property>

</bean>

**<!-- 静态资源配置 -->**

<mvc:resources mapping="/res/\*\*" location="/WEB-INF/res/" />

<!-- **aop的自动代理**proxy-target-class属性值决定是基于接口的还是基于类的代理被创建。

如果proxy-target-class 属性值被设置为true，那么基于类的代理将起作用（这时需要cglib库）。

如果proxy-target-class属值被设置为false或者这个属性被省略，那么标准的JDK 基于接口的代理将起作用 -->

<aop:aspectj-autoproxy proxy-target-class="false"/>

**6、controller 进行测试**





