1、创建web项目

2、导入SpringMVC的jar包

com.springsource.org.apache.log4j-1.2.15.jar

com.springsource.org.aspectj.weaver-1.6.8.RELEASE.jar

jstl-1.2.jar

spring-aop-4.2.4.RELEASE.jar

spring-aspects-4.2.4.RELEASE.jar

spring-beans-4.2.4.RELEASE.jar

spring-context-4.2.4.RELEASE.jar

spring-context-support-4.2.4.RELEASE.jar

spring-core-4.2.4.RELEASE.jar

spring-expression-4.2.4.RELEASE.jar

spring-jdbc-4.2.4.RELEASE.jar

spring-orm-4.2.4.RELEASE.jar

spring-tx-4.2.4.RELEASE.jar

spring-web-4.2.4.RELEASE.jar

spring-webmvc-4.2.4.RELEASE.jar

com.springsource.org.aopalliance-1.0.0.jar

com.springsource.org.apache.commons.logging-1.1.1.jar

3、在**web.xml**里配置

中央控制器DispactherServlet

```
<servlet>
  <servlet-name>springmvc</servlet-name>
  <servlet-class>org.springframework.web.servlet.DispatcherServlet</servlet-class>
</servlet>
```

配置拦截器

```
 <servlet-mapping>
   <servlet-name>springmvc</servlet-name>
   <url-pattern>*.do</url-pattern>
```

这地方可以是*.do、*.html 、*.action等任意一种

</servlet-mapping>

4、创建**springmvc**配置文件，然后在**web.xml**里配置**springmvc**

<!-- 自定义加载springmvc配置文件 -->

<init-param>

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

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

</init-param>

springmvc默认加载**springmvc**配置文件：

默认加载springmvc配置文件必须满足约定：

文件名称：servlet-name-servlet.xml springmvc-servlet.xml

文件路径：必须在WEB-INF下面

```
<!--
  前端控制器
  web项目：
  入口配置文件web.xml
  1、加载spring配置文件
  2、加载Springmvc配置文件
  -->
<servlet>
  <servlet-name>springmvc</servlet-name>
  <servlet-class>org.springframework.web.servlet.DispatcherServlet</servlet-class>
  <!--
  springmvc默认加载springmvc配置文件：
  默认加载springmvc配置文件必须满足约定：  springmvc配置文件--文件名
  文件名称：servlet-name-servlet.xml  springmvc-servlet.xml
  文件路径：必须在WEB-INF下面
   -->
  <!-- 自定义加载springmvc配置文件 -->
  <init-param>
      <param-name>contextConfigLocation</param-name>
      <param-value>classpath:springmvc.xml</param-value>
  </init-param>
</servlet>
```

5、在**springmvc**配置文件里配置处理器映射器、处理器适配器、视图解析器、管理**Controller**

<!-- 管理Controller-->

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

<!--

配置注解驱动：这东西就不需要再配置 映射器和适配器

默认创建：处理器映射器，处理器适配器，自动提供springmvc对json格式数据支持

记得要加mvc的约束

```
xmlns:mvc="http://www.springframework.org/schema/mvc"
http://www.springframework.org/schema/mvc
 http://www.springframework.org/schema/mvc/spring-mvc-4.0.xsd
 -->
 <mvc:annotation-driven/>

<!-- 处理器映射器：寻找Controller执行类 -->
 <bean
class="org.springframework.web.servlet.mvc.method.annotation.RequestMappingHandlerMapping">
</bean>
 <!-- 处理器适配器:执行Controller方法 -->
 <bean
class="org.springframework.web.servlet.mvc.method.annotation.RequestMappingHandlerAdapter">
</bean>
 <!--
  配置视图解析器
 功能：解析真正的物理视图
 解析方案：前缀+逻辑视图+后缀 组合成物理视图 /WEB-INF/jsps/hello.jsp
  -->
 <bean class="org.springframework.web.servlet.view.InternalResourceViewResolver">
  <property name="prefix" value="/WEB-INF/jsps/"></property>
  <property name="suffix" value=".jsp"></property>
 </bean>
```

6、创建**Controller**类

在类上声明该类为Controller，在类中创建相应的方法返回视图

```
@Controller     //注解声明该类为Controller
public class UserController {
```

7、在**Controller**使用返回视图分两种

首先要在方法上声明访问地址

```
@RequestMapping("hello"):  这个很重要  通过访问这个名字就可以访问到配置的页面
  localhost:8080/项目名/hello.do        //配置拦截器为*.do
```

7.1、返回物理视图

```
/**
 * 需求：入门测试
 *   返回：物理视图
 *  @RequestMapping("hello"):  这个很重要  通过访问这个名字就可以访问到配置的页面
 *    localhost:8080/项目名/hello.do        //配置拦截器为*.do
 */

@RequestMapping("hello")
public ModelAndView showPage(){
    //创建一个模式视图对象
    ModelAndView mv = new ModelAndView();
    //设置数据页面要接收的数据
    mv.addObject("hello", "凤姐在传智播客学习UI");
    //设置视图：物理视图
    mv.setViewName("/WEB-INF/jsps/hello.jsp");
    //返回物理视图,就不用配置视图解析器
    return mv;
}
```

7.1、返回逻辑视图（返回逻辑视图就必须配置视图解析器）

必须要配置视图解析器

```
@RequestMapping("hello2")
public ModelAndView showPage1(){
    //创建一个模式视图对象
    ModelAndView mv = new ModelAndView();
    //设置数据页面要接收的数据
    mv.addObject("hello", "凤姐在传智播客学习UI");
    //设置视图:返回逻辑视图，必须配置视图解析器  就给一个jsp页面的名称
    mv.setViewName("hello");
    //返回物理视图,就不用配置视图解析器
    return mv;
}
```

7.3返回逻辑视图第二种方法

```java
/**
 * 需求：入门案例
 * 返回值：String 返回逻辑视图
 * 参数：Map，ModelMap,Model
 */
@RequestMapping("hello1")
public String showPageHello(Model model){
    //放入数据
    model.addAttribute("hello", "黄晓明在传智播客学习!");
    //返回逻辑视图
    return "hello";
}
```

设置数据通过方法参Model数来设置

返回物理视图与返回逻辑视图的不同

1、当在springmvc配置中没有配置视图解析器时返回的是ModelAndView,需要设置物理视图。并返回一个物理视图MV

2、有配置视图解析器时返回就是逻辑视图,返回值可以是String,或者ModelAndView,返回的就是一个逻辑视图