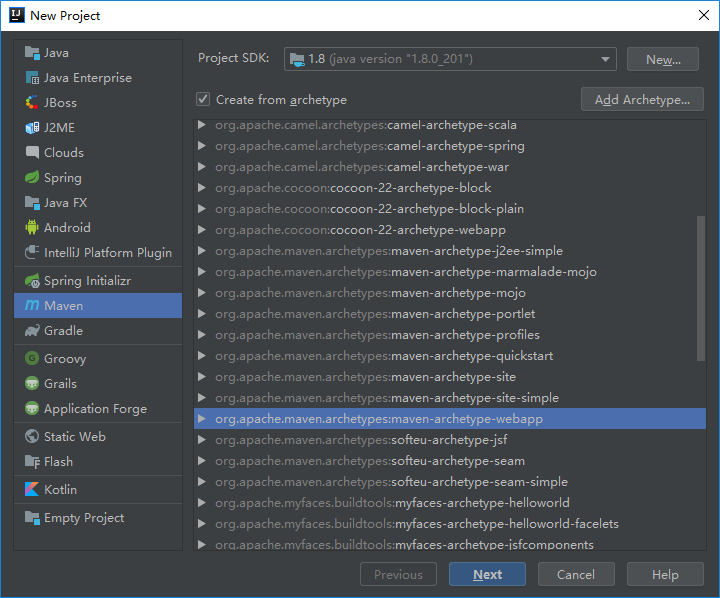
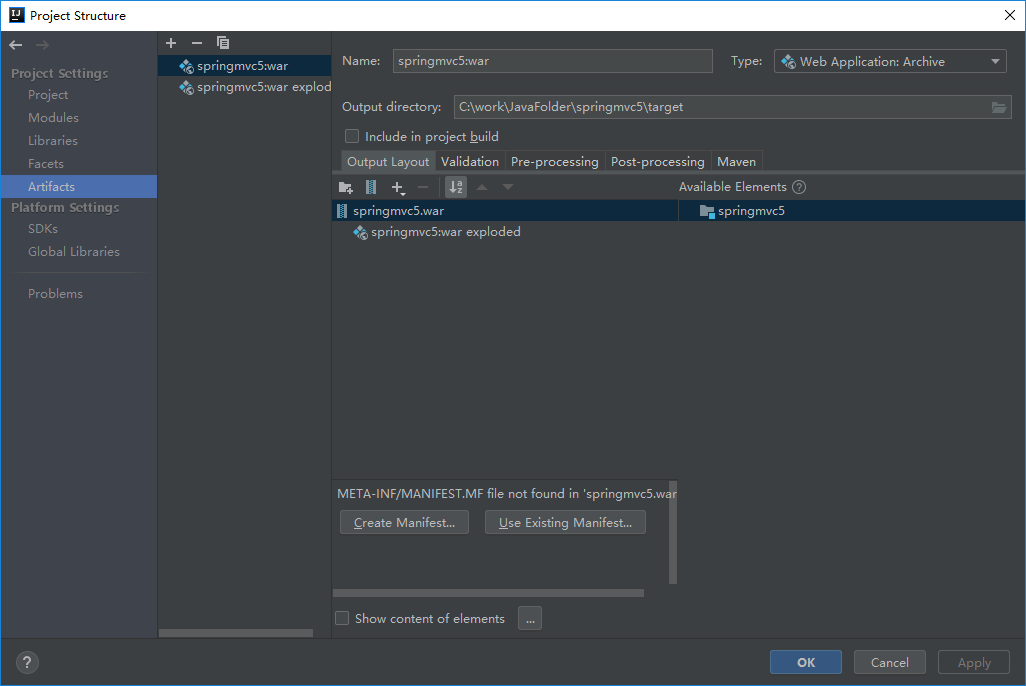
1. New project -> maven -> 选择特定的模板

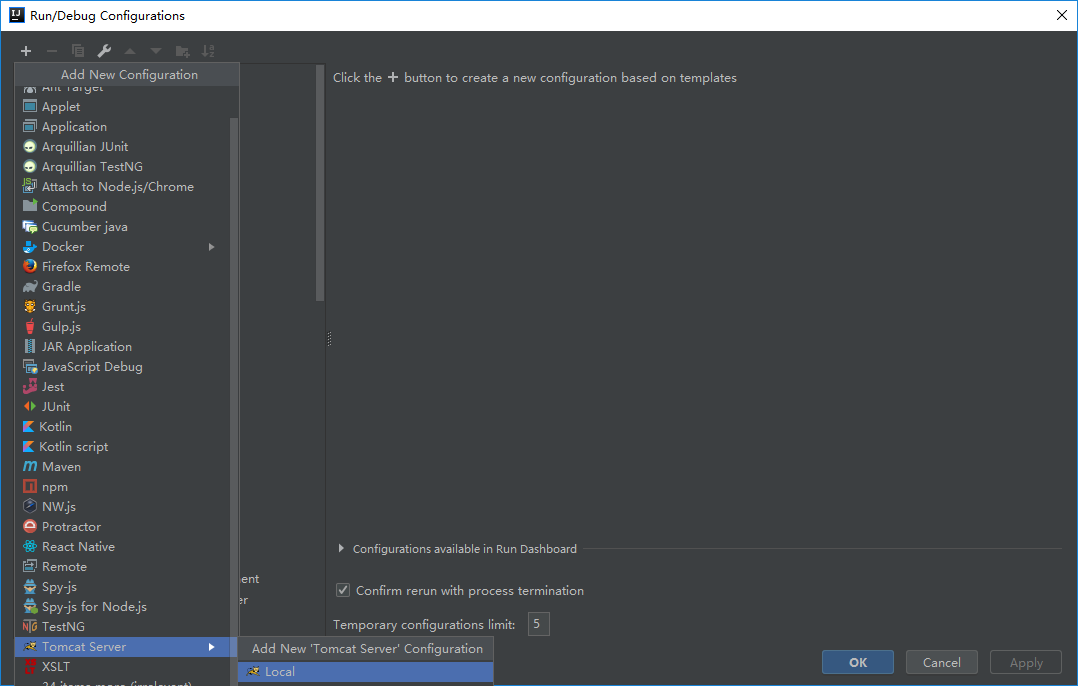


一直点下一步，然后 点击 右下角弹出框的 Auto Detect

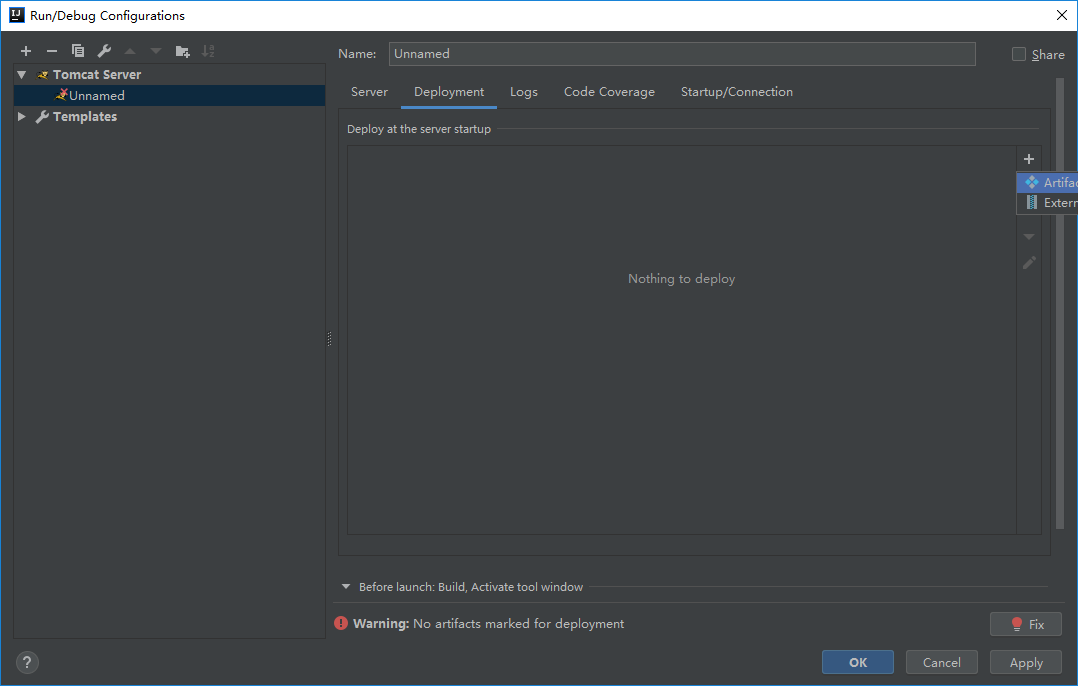
此时 Project Structure 对话框中（右键项目，选择 Open module Settings）的 Artifacts 已经默认有输出了



1. Run -> Edit Configuration 新建 配置模板 选择tomcat ->local (需先下载好tomcat)



Deployment tab中添加 web 的Artifacts 为部署项



1. run 来 启动 项目， 可以弹出浏览器 显示helloworld
2. Pom.xml 文件中添加 如下配置项

<dependency>  
 <groupId>org.springframework</groupId>  
 <artifactId>spring-webmvc</artifactId>  
 <version>4.2.6.RELEASE</version>  
</dependency>  
  
  
  
<!-- Tag libs support for view layer -->  
<dependency>  
 <groupId>javax.servlet</groupId>  
 <artifactId>jstl</artifactId>  
 <version>1.2</version>  
 <scope>runtime</scope>  
</dependency>  
<dependency>  
 <groupId>taglibs</groupId>  
 <artifactId>standard</artifactId>  
 <version>1.1.2</version>  
 <scope>runtime</scope>  
</dependency>

Main 目录下新建 java 目录，并设置其为 代码根目录，并在其下新建 com.xl.controller 的包。新建 StudentController.java 的类文件

import com.test.service.StudentService;  
import org.springframework.stereotype.Controller;  
import org.springframework.web.bind.annotation.RequestMapping;  
import org.springframework.web.bind.annotation.RequestMethod;  
import org.springframework.web.servlet.ModelAndView;  
import javax.annotation.Resource;  
  
@Controller  
@RequestMapping("student")  
public class StudentController {  
  
 @Resource(name = "studentService")  
 private StudentService studentService;  
  
 @RequestMapping(value = "/getAllStudent", method = RequestMethod.*GET*)  
 public ModelAndView getAllStudent() {  
 ModelAndView mav = new ModelAndView();  
 mav.setViewName("studentDisplay");  
 mav.addObject("students", studentService.getAllStudent());  
 return mav;  
 }  
}

1. 修改 web.xml文件 ，添加 spring servlet相关的配置

<!DOCTYPE web-app PUBLIC  
 "-//Sun Microsystems, Inc.//DTD Web Application 2.3//EN"  
 "http://java.sun.com/dtd/web-app\_2\_3.dtd" >  
  
<web-app id="WebApp\_ID" version="2.4"  
 xmlns="http://java.sun.com/xml/ns/j2ee"  
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"  
 xsi:schemaLocation="http://java.sun.com/xml/ns/j2ee  
 http://java.sun.com/xml/ns/j2ee/web-app\_2\_4.xsd">  
 <display-name>Archetype Created Web Application</display-name>  
  
  
 <servlet>  
 <servlet-name>spring</servlet-name>  
 <servlet-class>org.springframework.web.servlet.DispatcherServlet</servlet-class>  
 <init-param>  
 <param-name>contextConfigLocation</param-name>  
 <param-value>/WEB-INF/spring-servlet.xml</param-value>  
 </init-param>  
 <load-on-startup>1</load-on-startup>  
 </servlet>  
 <servlet-mapping>  
 <servlet-name>spring</servlet-name>  
 <url-pattern>/</url-pattern>  
 </servlet-mapping>  
  
</web-app>

1. WEB-INF 目录下 新建 spring-servlet.xml文件（web.xml中所配置）。

<beans xmlns="http://www.springframework.org/schema/beans"  
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"  
 xmlns:mvc="http://www.springframework.org/schema/mvc"  
 xmlns:context="http://www.springframework.org/schema/context"  
 xsi:schemaLocation="http://www.springframework.org/schema/beans http://www.springframework.org/schema/beans/spring-beans-3.0.xsd  
 http://www.springframework.org/schema/mvc http://www.springframework.org/schema/mvc/spring-mvc.xsd  
 http://www.springframework.org/schema/context http://www.springframework.org/schema/context/spring-context-3.0.xsd">  
  
 <!-- 开启spring的扫描注入，使用如下注解 -->  
 <!-- @Component,@Repository,@Service,@Controller-->  
 <context:component-scan base-package="com.xl"/>  
  
 <!-- 开启springMVC的注解驱动，使得url可以映射到对应的controller -->  
 <mvc:annotation-driven />  
  
 <!-- 视图解析 -->  
 <bean class="org.springframework.web.servlet.view.InternalResourceViewResolver">  
 <property name="prefix" value="/WEB-INF/views/"/>  
 <property name="suffix" value=".jsp"/>  
 </bean>  
  
</beans>

1. WEB-INF 目录下新建 views 目录，并 新建 studentDisplay.jsp 文件（文件名为StudentController.java 文件 gettAllStudent 方法 中 setViewName 函数的参数所决定）

<%@ **page** language="java" contentType="text/html; charset=UTF-8" pageEncoding="UTF-8"%>  
<%@ **taglib** prefix="c" uri="http://java.sun.com/jsp/jstl/core"%>  
<%@ **taglib** prefix="fmt" uri="http://java.sun.com/jsp/jstl/fmt"%>  
  
<html>  
<head>  
 <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />  
 <title>Spring MVC Hello World</title>  
</head>  
  
<body>  
<h2>All Students in System</h2>  
  
<table border="1">  
 <tr>  
 <th>Id</th>  
 <th>Name</th>  
 <th>Age</th>  
 <th>Sex</th>  
 <th>**${**students**}**</th>  
 </tr>  
  
</table>  
  
</body>  
</html>

1. 在浏览器中输入 相对路径为 /student/getAllStudent （此地址为 StudentController.java 文件 中的 @Request Mapping 所配置） 的url ，回车
2. 建立 返回json的方法，建立实体类

import java.io.Serializable;  
  
   
public class Student implements Serializable {  
 private String id;  
 private String name;  
 private int age;  
 private boolean sex;  
  
  
 public Student() {  
 }  
  
 public Student(String id, String name, int age, boolean sex) {  
 this.id = id;  
 this.name = name;  
 this.age = age;  
 this.sex = sex;  
 }  
  
 public String getId() {  
 return id;  
 }  
  
 public void setId(String id) {  
 this.id = id;  
 }  
  
 public String getName() {  
 return name;  
 }  
  
 public void setName(String name) {  
 this.name = name;  
 }  
  
 public int getAge() {  
 return age;  
 }  
  
 public void setAge(int age) {  
 this.age = age;  
 }  
  
 public boolean isSex() {  
 return sex;  
 }  
  
 public void setSex(boolean sex) {  
 this.sex = sex;  
 }  
  
   
}

1. pom.xml中添加引用

<dependency>  
 <groupId>com.fasterxml.jackson.core</groupId>  
 <artifactId>jackson-databind</artifactId>  
 <version>2.5.0</version>  
</dependency>

1. StudentController.java中添加 方法

@ResponseBody  
@RequestMapping(value = "/getStudent", method = RequestMethod.*GET*)  
  
public Student getStudent() {  
 Student aa = new Student();  
 aa.setAge(3);  
 aa.setName("的er得分");  
 aa.setSex(false);  
 return aa;  
}

1. 在浏览器中哦输入相对路径/student/getStudent