**1.@Controller,@RequestMapping**

用@Controller註釋的類別才會被SpringMVC解析器搜索到，這個註釋是必須的。@RequestMapping註釋用於指定controller方法的URL索引。這兩個注解的具體用法請看以下代碼：

controller层LearnMVCController.java代码：

[复制代码](javascript:void(0);)

package www.asuan.com.controller;

import org.springframework.stereotype.Controller;

import org.springframework.web.bind.annotation.RequestMapping;

@Controller

@RequestMapping("/learnMVC")//父索引 可省略

public class LearnMVCController {

@RequestMapping("/ex") //子索引

public String learnMVC() {

return "learnMVC.ftl";

}

}

[复制代码](javascript:void(0);)

视图learnMVC.ftl代码：

[复制代码](javascript:void(0);)

<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">

<html>

<head>

<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">

<title>Insert title here</title>

</head>

<body>

<h2>Hello World!</h2>

</body>

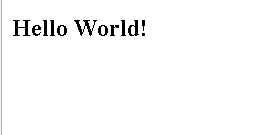
</html>

[复制代码](javascript:void(0);)

在有父索引的请况下，访问：http://localhost:8080/你的工程名/learnMVC/ex.htm

在省略父索引的请况下，访问：http://localhost:8080/你的工程名/ex.htm 就可将请求发送到所需controller方法。

运行结果：



**2.@RequestParam**

該註釋用於controller方法註釋中，用於綁定由View傳過來的參數，代碼如下：

controller代碼

[复制代码](javascript:void(0);)

package www.asuan.com.controller;

import org.springframework.stereotype.Controller;

import org.springframework.ui.Model;

import org.springframework.web.bind.annotation.RequestMapping;

import org.springframework.web.bind.annotation.RequestParam;

@Controller

@RequestMapping("/learnMVC")

public class LearnMVCController {

@RequestMapping("/ex")

public String learnMVC(@RequestParam(value = "userId", required = false) Integer userIdGot, Model model) {

String str = "成功得到綁定數據：" + userIdGot;

model.addAttribute("userIdGot", str);

return "learnMVC.ftl";

}

}

[复制代码](javascript:void(0);)

@RequestParam註釋參數中，Controller會掃描View傳遞過來的、參數名與value值相同的參數，此處為userId，掃描到參數後將參數的值綁定到註釋後面宣告的變數中，此處為userIdGot.

例如我們訪問地址：http://localhost:8080/你的工程名/learnMVC/ex.htm？userId=10001

那麼首先Controller掃描到URL鏈接後面帶的變數userId與value的值相同，所以Controller把變數userId的值10001賦給了@RequestParam後面宣告的變數userIdGot，這樣我們就得到了前臺傳過來的參數。

required參數為true的時候，如果你沒傳所需的參數,，程序將報錯。required參數可省略，省略時，其值預設為false.

注意：這個例子當中，如果宣告變數Integer userIdGot改成int userIdGot，而你又沒有傳相應的參數，訪問頁面將報錯。因為當Controller找不到匹配的變數時，會把null賦給這個變數，而null值裝化為int的時候會報錯，所以此處應用包裝類別。

編寫View文件如下：

[复制代码](javascript:void(0);)

<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">

<html>

<head>

<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">

<title>Insert title here</title>

</head>

<body>

<h2>${userIdGot}</h2>

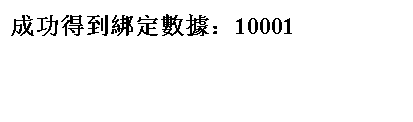
</body>

</html>

[复制代码](javascript:void(0);)

在瀏覽器訪問：http://localhost:8080/你的工程名/learnMVC/ex.htm？userId=10001

運行結果如下：



**3.@PathVariable**

@PathVariable用於綁定URL路徑上添加的參數，例如本文的連接：<http://www.cnblogs.com/sunang/p/3421707.html> 如果[http://www.cnblogs.com/sunang](http://www.cnblogs.com/sunang/p/3421707.html)是我們訪問控制層的路徑的話，那麼/p/3421707就是我們添加的參數，通過@PathVariable就可以綁定獲取這些參數在服務端進行相應操作。詳細使用方法情況以下代碼：

首先編寫learnMVC.ftl，代碼如下：

[复制代码](javascript:void(0);)

<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">

<html>

<head>

<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">

<title>Insert title here</title>

</head>

<body>

<h2>${msgGot}</h2>

</body>

</html>

[复制代码](javascript:void(0);)

編寫controller層，代碼如下：

[复制代码](javascript:void(0);)

package www.asuan.com.controller;

import org.springframework.stereotype.Controller;

import org.springframework.ui.Model;

import org.springframework.web.bind.annotation.PathVariable;

import org.springframework.web.bind.annotation.RequestMapping;

@Controller

@RequestMapping("/learnMVC")

public class LearnMVCController {

//子索引由 ex+參數+參數 組成

@RequestMapping("/ex/{type}/{articleId}")

public String learnMVC(@PathVariable(value = "type") String typeGot,

@PathVariable(value = "articleId") Integer articleIdGot, Model model) {

String str = "URL後面所傳輸的參數為：" + typeGot + "/" + articleIdGot;

model.addAttribute("msgGot", str);

return "learnMVC.ftl";

}

}

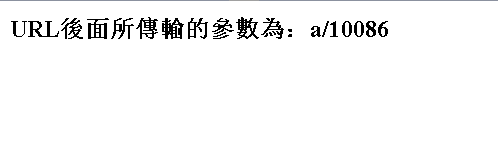
[复制代码](javascript:void(0);)

@PathVariable的實現原理和@RequestParam相似，Controller掃描@RequestMapping路徑中、{}號裏面的參數名，把參數值綁定到與@PathVariable的value值相等的變數中，比如上例中，{type}的值被綁定到value="type"的註釋後面宣告的變數typeGot中，參數的個數可以自由設定，這裡為兩個參數。

注意：在訪問URL時，必須用ex/參數值/參數值 的格式才能訪問到上例中的controller方法，另外還要注意參數的類別型。

現在讓我們運行工程，在瀏覽器訪問：http://localhost:8080/你的工程名/learnMVC/ex/p/10086.htm    注：此處p對應{type},10086對應{articleId}

運行結果：



@ModelAttribute用于綁定Controller接收的實體對象，本文以一個form submit的例子來演示下這個註釋的用法：

首先，寫一個 form 頁面register.ftl，代碼如下：

[复制代码](javascript:void(0);)

<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">

<html>

<head>

<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">

<title>Insert title here</title>

</head>

<body>

<table>

<form name="register" action="register.htm" method="post">

<tr>

<td>姓名</td>

<td><input name="userName" type="text"></input></td>

</tr>

<tr>

<td>年齡</td>

<td><input name="age" type="text"></input></td>

</tr>

<tr>

<td><input type="submit" value="submit"></input></td>

</tr>

</form>

<table>

</body>

</html>

[复制代码](javascript:void(0);)

還有一個註冊成功的頁面registerSuccess.ftl：

[复制代码](javascript:void(0);)

<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">

<html>

<head>

<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">

<title>Insert title here</title>

</head>

<body>

<h2>註冊成功！用戶名：${userNameGot}&nbsp年齡：${ageGot}</h2>

</body>

</html>

[复制代码](javascript:void(0);)

接下來，我們要新建一個VO(value object)文件用來把 form 信息綁定到vo對象,在src/main/java目錄下新建包www.asuan.com.vo,在包里新建UserInfo.java，代碼如下：

[复制代码](javascript:void(0);)

package www.asuan.www.vo;

public class UserInfo {

private String userName;

private int age;

public String getUserName() {

return userName;

}

public void setUserName(String userName) {

this.userName = userName;

}

public int getAge() {

return age;

}

public void setAge(int age) {

this.age = age;

}

}

[复制代码](javascript:void(0);)

接下是controller類別，代碼如下：

[复制代码](javascript:void(0);)

package www.asuan.com.controller;

import org.springframework.stereotype.Controller;

import org.springframework.ui.Model;

import org.springframework.web.bind.annotation.ModelAttribute;

import org.springframework.web.bind.annotation.RequestMapping;

import www.asuan.www.vo.UserInfo;

@Controller

@RequestMapping("/learnMVC")

public class LearnMVCController {

//使用該索引跳轉到註冊頁面

@RequestMapping("/goRegisterPage")

public String goRegisterPage() {

return "register.ftl";

}

//使用該索引進行註冊操作并返回註冊成功頁面

@RequestMapping("/register")

public String register(@ModelAttribute(value = "register") UserInfo userInfo, Model model) {

//通過@ModelAttribute，我們把 form 信息綁定到了userInfo對象裏面，可以直接使用

String userName = userInfo.getUserName();

int age = userInfo.getAge();

model.addAttribute("userNameGot", userName);

model.addAttribute("ageGot", age);

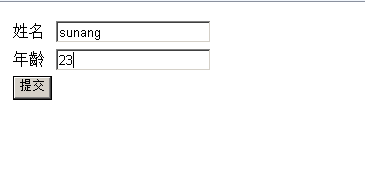
return "registerSuccess.ftl";

}

}

[复制代码](javascript:void(0);)

部署運行工程，在瀏覽器訪問：http://localhost:8080/你的工程名/learnMVC/goRegisterPage.htm 訪問 form 填寫頁面并填寫測試信息，運行結果如下：



點擊submit， form 的action配置了要訪問的controller路徑register.htm,所以Controller會執行register()方法，我們把該方法單獨列出分析：

[复制代码](javascript:void(0);)

@RequestMapping("/register")

public String register(@ModelAttribute(value = "register") UserInfo userInfo, Model model) {

//通過@ModelAttribute，我們把 form 信息綁定到了userInfo對象裏面，可以直接使用

String userName = userInfo.getUserName();

int age = userInfo.getAge();

model.addAttribute("userNameGot", userName);

model.addAttribute("ageGot", age);

return "registerSuccess.ftl";

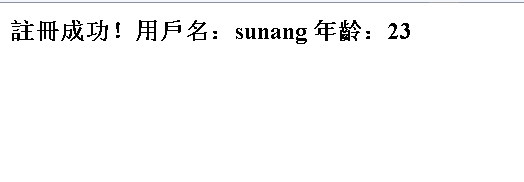
}

[复制代码](javascript:void(0);)

Controller會掃描與@ModelAttribute的value值相同的實體對象，這裡value = "register"，而submit的 form 的name="register"兩者相同，所以submit的 form 信息被綁定到了@ModelAttribute後面宣告的UserInfo對象中。

form 中的各個字段和實體類別對象的各個字段會按照字段名一一匹配，比如 form userName匹配UserInfo對象中的userName。在上例中 form 的兩個字段被綁定到了userInfo對象的兩個字段中，我們在進行業務處理的時候可以直接使用，通過這種方法我們就得到了 form 的信息。

運行結果：



@ResponseBody用于在controller方法中直接返回一個數據對象，常用于Ajax交互中，本文用Ajax交互的例子來演示下該註釋的用法。

step1.由於Ajax傳輸數據用到JSON，所以要先添加JSON依賴如下：

http://images.cnitblog.com/blog/581445/201311/19102505-f0ccefecea1847719007eeab202d13bf.png

Maven代碼如下：

[复制代码](javascript:void(0);)

<!-- JSON -->

<dependency>

<groupId>org.codehaus.jackson</groupId>

<artifactId>jackson-core-asl</artifactId>

<version>1.8.4</version>

</dependency>

<dependency>

<groupId>org.codehaus.jackson</groupId>

<artifactId>jackson-mapper-asl</artifactId>

<version>1.8.4</version>

</dependency>

[复制代码](javascript:void(0);)

在spring配置文件中加入JSON所需配置，此處以spring-servlet.xml為例，代碼如下：

[复制代码](javascript:void(0);)

<!-- JSON所需配置 -->

<bean class="org.springframework.web.servlet.mvc.annotation.AnnotationMethodHandlerAdapter">

<property name="messageConverters">

<list>

<ref bean="mappingJacksonHttpMessageConverter" />

</list>

</property>

</bean>

<bean id="mappingJacksonHttpMessageConverter" class="org.springframework.http.converter.json.MappingJacksonHttpMessageConverter">

<property name="supportedMediaTypes">

<list>

<value>application/json;charset=UTF-8</value>

</list>

</property>

</bean>

[复制代码](javascript:void(0);)

 step2.編寫頁面ajaxGetMsg.ftl，代碼如下：

[复制代码](javascript:void(0);)

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml">

<html>

<head>

<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">

<!-- 引入jquery文件-->

<script type="text/javascript" src="../js/jquery.js"></script>

<title>Insert title here</title>

</head>

<body>

<span id="content">

<!-- 點擊按鈕後調用getMsg()方法-->

<button type="button" onclick="javascript:getMsg();">@ResponseBody結合Ajax例子演示</button>

</span>

</body>

</html>

<script type="text/javascript">

function getMsg(){

var content = "";

//ajax訪問controller方法，利用@ResponseBody返回數據對象

$.ajax({

async:false,

cache : false,

type : 'POST',

dataType : "json",

url:"ajaxGetMsg.htm",

success:function(data){

$.each(data, function(i,obj){

content=content+obj;

});

$("#content").html(content);

},

error:function(){

alert("加载失败");

return;

}

});

}

</script>

[复制代码](javascript:void(0);)

step3.編寫controller方法，代碼如下：

[复制代码](javascript:void(0);)

package www.asuan.com.controller;

import java.util.ArrayList;

import java.util.List;

import org.springframework.stereotype.Controller;

import org.springframework.web.bind.annotation.RequestMapping;

import org.springframework.web.bind.annotation.ResponseBody;

@Controller

@RequestMapping("/learnMVC")

public class LearnMVCController {

//前往初始頁面

@RequestMapping("/indexPage")

public String indexPage(){

return "ajaxGetMsg.ftl";

}

//Ajax交互方法

@RequestMapping("/ajaxGetMsg")

@ResponseBody

public List<String> ajaxGetMsg() {

List<String> strList = new ArrayList<String>();

strList.add("學");

strList.add("習");

strList.add("Spring");

strList.add("M");

strList.add("V");

strList.add("C");

return strList;

}

}

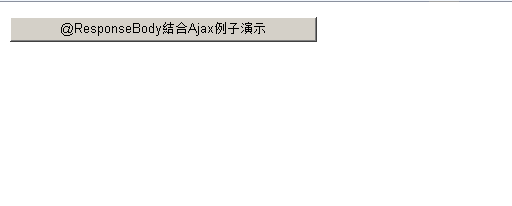
[复制代码](javascript:void(0);)

indexPage()方法用於訪問初始頁面，ajaxGetMsg()方法上加了@ResponseBody註釋，所以該方法可以直接向頁面返回數據對象，該方法的返回數據類別型為List<String>.

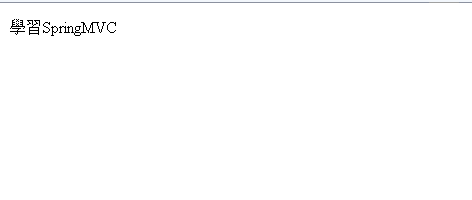
step4.運行調試

部署運行項目，瀏覽器訪問：http://localhost:8080/你的工程名/learnMVC/indexPage.htm

運行結果如下：



點擊按鈕，ajax加載數據得到如下結果：



**@Scope**

* **Spring container 要回傳哪個 bean instance。**
* singleton(default), prototype, request, session, globalSession。
  + singleton: container 中只存在一個 instance。
  + **prototype: 每個取得都是新的 instance**。