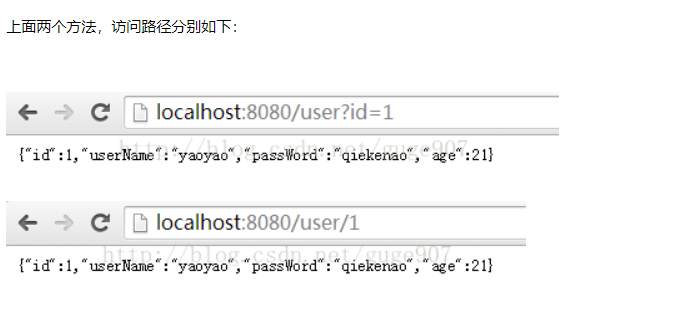
@RequestParam：获取的是请求参数，一般是url问号后面的参数值

@PathVariable：获取的是请求路径中参数的值

使用@RequestParam时，URL是这样的：http://host:port/path?参数名=参数值

使用@PathVariable时，URL是这样的：http://host:port/path/参数值



**3.2REST**

* REST：即 Representational State Transfer。（资源）表现层状态转化。是目前最流行的一种互联网软件架构。它结构清晰、符合标准、易于理解、扩展方便，所以正得到越来越多网站的采用
* 资源（**Resources**）：网络上的一个实体，或者说是网络上的一个具体信息。它可以是一段文本、一张图片、一首歌曲、一种服务，总之就是一个具体的存在。可以用一个URI（统一资源定位符）指向它，每种资源对应一个特定的 **URI**。要获取这个资源，访问它的URI就可以，因此 **URI**即为每一个资源的独一无二的识别符。
* 表现层（**Representation**）：把资源具体呈现出来的形式，叫做它的表现层（**Representation**）。比如，文本可以用 txt 格式表现，也可以用 HTML 格式、XML 格式、JSON 格式表现，甚至可以采用二进制格式。
* 状态转化（**State Transfer**）：每发出一个请求，就代表了客户端和服务器的一次交互过程。HTTP协议，是一个无状态协议，即所有的状态都保存在服务器端。因此，如果客户端想要操作服务器，必须通过某种手段，让服务器端发生**“**状态转化**”**（**State Transfer**）。而这种转化是建立在表现层之上的，所以就是 **“**表现层状态转化**”**。具体说，就是 **HTTP**协议里面，四个表示操作方式的动词：**GET**、**POST**、**PUT**、**DELETE**。它们分别对应四种基本操作：**GET**用来获取资源，**POST**用来新建资源，**PUT**用来更新资源，**DELETE**用来删除资源。

**示例：**   
- /order/1 HTTP **GET**：得到 id = 1 的 order   
- /order/1 HTTP **DELETE**：删除 id = 1的 order   
- /order/1 HTTP **PUT**：更新id = 1的 order   
- /order HTTP **POST**：新增 order

**HiddenHttpMethodFilter**：浏览器 form 表单只支持 GET与 POST 请求，而DELETE、PUT 等 method 并不支持，Spring3.0 添加了一个过滤器，可以将这些请求转换为标准的 http 方法，使得支持 GET、POST、PUT 与DELETE 请求。

**3.2.1在web.xml中配置HiddenHttpMethodFilter**

**web.xml**

1. *<!--*
2. *配置org.springframework.web.filter.HiddenHttpMethodFilter:可以把POST请求转换成DELETE或者POST请求*
3. *-->*
5. <filter>
6. <filter-name>HiddenHttpMethodFilter</filter-name>
7. <filter-class>org.springframework.web.filter.HiddenHttpMethodFilter</filter-class>
8. </filter>
10. <filter-mapping>
11. <filter-name>HiddenHttpMethodFilter</filter-name>
12. <url-pattern>/\*</url-pattern>
13. </filter-mapping>

**3.2.2在SpringMVCTest中测试相应的方法**

**SpringMVCTest.java**

1. */\**
2. *\* 如何发送PUT和DELETE请求*
3. *\* 1.需要配置HiddenHttpMethodFilter*
4. *\* 2.需要发送POST请求*
5. *\* 3.需要发送POST请求时携带一个name="\_method"的隐藏域，value值为DELETE或者PUT*
6. *\**
7. *\* 在springmvc框架中通过@PathVariable注解来获取id值*
8. *\* \*/*
9. *//get请求*
10. @RequestMapping(value="/testRest/{id}",method=RequestMethod.GET)
11. public String testRest(@PathVariable Integer id)
12. {
13. System.out.println("testRest Get"+id);
14. return SUCCESS;
15. }
17. *//post请求*
18. @RequestMapping(value="/testRest",method=RequestMethod.POST)
19. public String testRest()
20. {
21. System.out.println("testRest POST");
22. return SUCCESS;
23. }
25. *//delete请求*
26. @RequestMapping(value="/testRest/{id}",method=RequestMethod.DELETE)
27. public String testRestDelete(@PathVariable Integer id)
28. {
29. System.out.println("testRest DELETE"+id);
30. return SUCCESS;
31. }
33. *//修改请求*
34. @RequestMapping(value="/testRest/{id}",method=RequestMethod.PUT)
35. public String testRestPut(@PathVariable Integer id)
36. {
37. System.out.println("testRest PUT"+id);
38. return SUCCESS;
39. }

**3.2.3index.jsp中的写法**

**index.jsp**

1. <a href="springmvc/testRest/1">Test Rest Get</a><br/>
2. <form action="springmvc/testRest" method="post">
3. <input type="submit" value="TestRest Post"/>
4. </form><br/>
5. <form action="springmvc/testRest/1" method="post">
6. <input type="hidden" name="\_method" value="DELETE"/>
7. <input type="submit" value="TestRest DELETE"/>
8. </form><br/>
9. <form action="springmvc/testRest/1" method="post">
10. <input type="hidden" name="\_method" value="PUT"/>
11. <input type="submit" value="TestRest PUT"/>
12. </form><br/>