RestFul 编程风格

一种软件架构风格，设计风格而不是标准，只是提供了一组设计原则和约束条件。它主要用于客户端和服务器交互类的软件。基于这个风格设计的软件可以更简洁，更有层次，更易于实现缓存等机制。

REST（英文：Representational State Transfer，简称 REST）描述了一个架构样式的网络系统，比如 web 应用程序。它首次出现在 2000 年 Roy Fielding 的博士论文中，他是 HTTP 规范的主要编写者之一。在目前主流的三种 Web 服务交互方案中，REST 相比于 SOAP（Simple Object Access protocol，简单对象访问协议）以及 XML-RPC 更加简单明了，无论是对 URL 的处理还是对 Payload 的编码，REST 都倾向于用更加简单轻量的方法设计和实现。值得注意的是 REST 并没有一个明确的标准，而更像是一种设计的风格。

Representational State Transfer （资源的）表现层状态转化

所谓"资源"，就是网络上的一个实体，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实在。你可以用一个 URI 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 就成了每一个资源的地址或独一无二的识别符。所谓"上网"，就是与互联网上一系列的"资源"互动，调用它的 URI。

表现层，我们把"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；图片可以用 JPG 格式表现，也可以用 PNG 格式表现。

Restful 风格： http://bbs.csdn.net/topics/390908212

非 resfull 风格：http://bbs.csdn.net/topics?tid=390908212

严格地说，有些网址最后的".html"后缀名是不必要的，因为这个后缀名表示格式，属于"表现层"范畴，而 URI 应该只代表"资源"的位置。它的具体表现形式，应该在 HTTP 请求的头信息中用 Accept 和 Content-Type 字段指定，这两个字段才是对"表现层"的描述。

状态转化（State Transfer）

访问一个网站，就代表了客户端和服务器的一个互动过程。在这个过程中，势必涉及到数据和状态的变化。

互联网通信协议 HTTP 协议，是一个无状态协议。这意味着，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"。

客户端用到的手段，只能是 HTTP 协议。

具体来说，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。

访问服务器资源，通过不同的 http 请求方式，服务器就知道对 CRUD 的哪个操作！
JAX-RS 发布服务就是使用 RESTFUL 风格。

综述
综合上面的解释，我们总结一下什么是 RESTful 架构：
（1）每一个 URI 代表一种资源；
（2）客户端和服务器之间，传递这种资源的某种表现层；
（3）客户端通过四个 HTTP 动词，对服务器端资源进行操作，实现"表现层状态转化"。

domain
@XmlRootElement(name = "User") 指定序列化名称，即转换 xml、json 名称。
service
@Path("/userService") 服务器访问资源路径
@Consumes 能否处理的请求的数据格式类型
@Produces 生成哪种格式的数据返回给客户端

@GET 查
@POST 增
@DELETE 删
@PUT 改
发布服务：
JAXRSServerFactoryBean
调用服务：
WebClient.create("").type(MediaType.APPLICATION_XML).post(user);