链滴

何为 REST？与如何构建 REST？

作者：baozebing
原文链接：https://ld246.com/article/1588403427020
来源网站：链滴
许可协议：署名-相同方式共享 4.0 国际 (CC BY-SA 4.0)

表现层状态转换REST是Representational State Transfer的缩写，是Roy Thomas Fielding博士于000年在他的博士论文中提出来的一种万维网软件架构风格，目的是便于不同软件/程序在网络（例如联网）中互相传递信息。表现层状态转换是根基于超文本传输协议（HTTP）之上而确定的一组约束属性，是一种设计提供万维网络服务的软件构建风格。符合或兼容于这种架构风格（简称为 REST 或 ESTful）的网络服务，允许客户端发出以统一资源标识符访问和操作网络资源的请求，而与预先定义的无状态操作集一致化。因此表现层状态转换提供了在互联网络的计算系统之间，彼此资源可交互使的协作性质（interoperability）。相对于其它种类的网络服务，例如SOAP服务，则是以本身所定义操作集，来访问网络上的资源。

要点及标准

REST是设计风格而不是标准。REST通常基于使用HTTP、URI、XML以及HTML这些现有的广泛流行协议和标准。

- 资源是由URI来指定。
- 对资源的操作包括获取、创建、修改和删除，这些操作正好对应HTTP协议提供的GET、POST、PU和DELETE方法。
- 通过操作资源的表现形式来操作资源。
- 资源的表现形式则是XML或者HTML，取决于读者是机器还是人、是消费Web服务的客户软件还是eb浏览器。当然也可以是任何其他的格式，例如JSON。
- 统一接口(Uniform Interface)

REST架构的限制条件

- 客户端-服务器（Client-Server）

客户端-服务器结构限制的目的是将客户端和服务器端的关注点分离. 将用户界面所关注的逻辑和数据储所关注的逻辑分离开来有助于提高用户界面的跨平台的可移植性.通过简化服务器模块也有助于服务模块的可扩展性.

- 无状态（Stateless）

服务器不能保存客户端的信息, 每一次从客户端发送的请求中, 要包含所有的必须的状态信息, 会话信由客户端保存, 服务器端根据这些状态信息来处理请求. 服务器可以将会话状态信息传递给其他服务, 如数据库服务, 这样可以保持一段时间的状态信息, 从而实现认证功能. 当客户端可以切换到一个新状的时候发送请求信息. 当一个或者多个请求被发送之后, 客户端就处于一个状态变迁过程中. 每一个应的状态描述可以被客户端用来初始化下一次的状态变迁.

- 缓存（Cacheability）

如同万维网一样, 客户端和中间的通讯传递者可以将回复缓存起来. 回复必须明确的或者间接的表明本是否可以进行缓存, 这可以预防客户端在将来进行请求的时候得到陈旧的或者不恰当的数据. 管理良好缓存机制可以减少客户端-服务器之间的交互, 甚至完全避免客户端-服务器交互, 这进一步提了高性能可扩展性。

- 统一接口（Uniform Interface）

这是 RESTful 系统设计的基本出发点. 它简化了系统架构, 减少了耦合性, 可以让所有模块各自独立的行改进. 包括下列四个限制:

请求中包含资源的 ID (Resource identification in requests)	请求中包含了各种独立资源的标识, 例如, 在 Web 服务中的 URIs. 资源本身和发送给客户端的标识是独立. 例如, 服务器可以将自身的数据库信息以 HTML XML 或者 JSON 的方式发送给客户端, 但是这些可能都不是服务器的内部记录方式.
资源通过标识来操作 (Resource manipulation through representations)	当客户端拥有一个资源的标识, 包括附带的元数据, 则它就有足够的信息来删除这个资源.
消息的自我描述性 (Self-descriptive messages)	每一个消息都包含足够的信息来描述如何来处理这个信息. 例如, 媒体类型 (midia-type) 就可以确定需要什么样的分析器来分析媒体数据.
用超媒体驱动应用状态 (Hypermedia as the engine of application state (HATEOAS))	同用户访问 Web 服务器的 Home 页面相似,当一个 REST 客户端访问了最初的 REST 应用的 URI 之后, REST 客户端应该可以使用服务器端提供的链接,动态的发现所有的可用的资源和可执行的操作.随着访问的进行, 服务器在响应中提供文字超链接, 以便客户端可以得到当前可用的操作. 客户端无需用确定的编码的方式记录下服务器端所提供的动态应用的结构信息.

● 客户端一般不知道是否直接连接到了最终的服务器, 或者是路径上的中间服务器. 中间服务器可以通负载　均衡和共享缓存的机制提高系统的可扩展性,这样可也便于安全策略的部署.分层系统（Layered ystem）

关于状态

应该注意区别应用的状态和连接协议的状态。HTTP连接是无状态的（也就是不记录每个连接的信息，而REST传输会包含应用的所有状态信息，因此可以大幅降低对HTTP连接的重复请求资源消耗。

将REST应用于WEB服务中

符合REST设计风格的Web API称为RESTful API。

RESTful API的定义标准

直观简短的资源地址	URI，比如：http://example.com/resources
传输的资源	Web服务接受与返回的媒体类型，比如：JSON，XML，YAML等
对资源的操作	Web服务在该资源上所支持的一系列请求方法（比如：POST，GET，PUT或DELETE）

下表列出实现RESTful API时HTTP请求方法的典型例子

资源	GET	PUT	POST	DELETE
一组资源的URI，比如 https://example.com/resources	列出URI，以及该资源组中每个资源的详细信息	使用给定的一组资源替换当前整组资源。	在本组资源中创建/追加一个新的资源。该操作往往返回新资源的URL。	删除整组资源
单个资源的URI，比如 https://example.com/resources/142	获取指定的资源的详细信息，格式可以自选一个合适的媒体类型（比如：XML、JSON等）	替换/创建指定的资源。并将其追加到相应的资源组中	把指定的资源当做一个资源组，并在其下创建/追加一个新的元素，使其隶属于当前资源	删除指定的元素

PUT和DELETE方法是幂等方法。

GET方法是安全方法（不会对服务器端有修改，因此当然也是幂等的）。

不像基于SOAP的Web服务，RESTful Web服务并没有“正式”的标准。这是因为REST是一种架构而SOAP只是一个协议。

原文链接：何为 REST？与如何构建 REST？

虽然REST不是一个标准，但大部分RESTful Web服务实现会使用HTTP、URI、JSON和XML等各种准。

实现举例

例如，在一个简单的网店中列举所有商品。

```
GET http://www.xxxstore.com/products
```

呈现某一件商品

```
GET http://www.store.com/products/12345
```

下单购买商品

```
POST http://www.store.com/orders
<purchase-order>
  <item> ... </item>
</purchase-order>
```

REST的优点

- 可更高效利用缓存来提高响应速度
- 通讯本身的无状态性可以让不同的服务器的处理一系列请求中的不同请求，提高服务器的扩展性
- 浏览器即可作为客户端，简化软件需求
- 相对于其他叠加在HTTP协议之上的机制，REST的软件依赖性更小
- 不需要额外的资源发现机制
- 在软件技术演进中的长期的兼容性更好

原文链接：何为 REST？与如何构建 REST？