设计优美的 WEB API

概述

WEB API 的应用场景:

- 将已有系统的功能或数据开放给合作伙伴或生态圈
- 对外发布可嵌入到其他网页的微件
- 构建前后端分离的 WEB 应用
- 开发跨不同终端的移动应用
- 集成公司内部不同系统

评判标准

- 易于使用
- 便于更改
- 健壮稳定

设计规范

URI

正例：http://api.example.com/users

反例：http://api.example.com/service/api/users

反例：http://api.example.com/sv/u

反例：http://api.example.com/Users/12345

反例：http://example.com/API/getUserName

正例：http://api.example.com/v1/items/123456

反例：http://api.example.com/cgi-bin/get_user.php?user=100

保持统一

反例：获取好友信息，http://api.example.com/friends?id=100

反例：发送消息，http://api.example.com/friend/100/messages

正例：获取好友信息，http://api.example.com/friends/100

正例：发送消息，http://api.example.com/friends/100/messages

例子

```
http://api.example.com/v1/users/12345/profile-image

http://api.example.com/v1/users/12345/profile_image

http://api.example.com/v1/users/12345/profileImage
```

查询参数

业界有两种常用的参数设计(per-page 与 page、limit 与 offset)，用于标识每次获取的数据量和起始位置

风格1：http://api.example.com/friends?per-page=50&page=3

风格2：http://api.example.com/friends?limit=50&offset=100

在设计过滤的参数时，业界也有一些事实标准可供参考。如果我们期望查询结果的特定属性取值跟过滤参数的取值完全相同，那过滤参数的名称通常为属性名；如果我们期望查询结果任意属性部分包含过滤参数的取值，那过滤参数的名称通常为 “q”

完全符合：http://api.example.com/v1/users?name=ken

全文搜索：http://api.example.com/v1/users?q=ken

模糊搜索：http://yboss.yahooapis.com/ysearch/web?q=ipad

路径元素：http://api.example.com/v1/users/{id}

查询参数：http://api.example.com/v1/users?name=ken

HTTP 方法

HTTP 协议设计的本意，URI 用于标识被操作的目标对象（资源），而 HTTP 方法则是表示操作方法。

```
GET /v1/users/123 HTTP/1.1

Host: api.example.com
```

- GET，获取资源
- POST，新增资源
- PUT，更新已有资源
- DELETE，删除资源
- PATCH，更新部分资源
- HEAD，获取资源的元信息

资源的设计粒度

- LEVEL 0：使用 HTTP
- LEVEL 1：引入资源的概念
- LEVEL 2：引入 HTTP 动词（GET/POST/PUT/DELETE 等）
- LEVEL 3：引入 HATEOAS 概念

响应数据

常用的数据格式有：HTML、XML、JSON、YAML 等

如果服务在响应时支持不同类型的数据格式，那应用在调用服务时如何获得期望格式的响应数据呢？

- 使用查询参数的方法：

`示例：https://api.example.com/v1/users?format=xml`

- 使用扩展名的方法：

`示例：https://api.example.com/v1/users.json`

- 使用在请求首部指定媒体类型的方法，优先推荐此种方法：

```
GET /v1/users
Host: api.example.com
Accept: application/json
```

出错信息

- 1XX：消息
- 2XX：成功
- 3XX：重定向
- 4XX：客户端原因引起的错误
- 5XX：服务器端原因引起的错误

传递详细的出错信息：

- 1：定义私有的首部，将其填入响应消息的首部。
- 2：将详细的出错信息放入消息体。

版本管理

业界有三种常见的标注 WEB API 版本的方法：

- 在 URI 中嵌入版本编号：

`示例：http://api.linkedin.com/v1/people`

- 在查询字符串里加入版本信息：

`示例：http://api.example.com/users/123?v=2`

- 通过媒体类型来指定版本信息

```
Accept: application/vnd.github.v3+json
```

```
Content-Type: application/vnd.github.v3+json
```

版本管理一些业界规范

版本编号由点号连接的 3 个数字组成，例如：1.2.3，分别表示主版本编号、次版本编号、补丁版本编号，版本编号的增加遵循下述规则：

- 在对软件进行不向下兼容的变更时，增加主版本编号；
- 在对软件进行向下兼容的变更或废除某些特定的功能时，增加次版本编号；
- 如果软件的 API 没有发生变更，只是修正了部分 bug，则增加补丁版本编号。

Reference

https://github.com/godruoyi/restful-api-specification