**API接口规范**

整体规范建议采用RESTful 方式来实施。

**1. 协议**

API与用户的通信协议，使用HTTPS协议[聊天采用socket]，确保交互数据的传输安全。

### ****2. 域名****

<https://example.org/api/>

### ****HTTP请求方式****

对于资源的具体操作类型，由HTTP动词表示。

常用的HTTP动词有下面四个（括号里是对应的SQL命令）。

GET（SELECT）：从服务器取出资源（一项或多项）。

POST（CREATE）：在服务器新建一个资源。

PUT（UPDATE）：在服务器更新资源（客户端提供改变后的完整资源）。

DELETE（DELETE）：从服务器删除资源。

下面是一些例子。

GET /room：列出所有房间

POST /room：新建一个房间

GET /room/ID：获取某个指定房间的信息

PUT /room/ID：更新某个指定房间的信息

DELETE /room/ID：删除某个房间

**4. 过滤信息**

如果记录数量很多，服务器不可能都将它们返回给用户。API应该提供参数，过滤返回结果。

下面是一些常见的参数。

?limit=10：指定返回记录的数量

?offset=10：指定返回记录的开始位置。

?page=2&per\_page=100：指定第几页，以及每页的记录数。

?sortby=name&order=asc：指定返回结果按照哪个属性排序，以及排序顺序。

?producy\_type=1：指定筛选条件

**5. API 传入参数**

传入参数分为4种类型：

地址栏参数

\* restful 地址栏参数 /api/room/122 122为房间编号，获取房间为122的信息

\* get方式的查询字串 见过滤信息小节

请求body数据

cookie

request header

cookie和header 一般都是用于OAuth认证的2种途径

**6. 返回数据**

只要api接口成功接到请求，就不能返回200以外的HTTP状态。

为了保障前后端的数据交互的顺畅，建议规范数据的返回，并采用固定的数据格式封装。

接口返回模板：

{

status:0,

data:{}||[],

msg:''

}

status: 接口的执行的状态

=0表示成功

<0 表示有异常=""

Data 接口的主数据

可以根据实际返回数组或JSON对象

Msg 信息

当status!=0 都应该有错误信息