

共享平台 RESTful api规范

版本	内容	作者	时间
V0.1	CRUD 的API定义及基本的RESTful规范	胡淮波	2014-9-29
V0.1.1	补充一些细节的内容	胡淮波	2014-9-30
V0.1.2	补充 域名规范 HTTP VERBS 的使用 URL的命名规范 API的文档描述方法	胡淮波	2014-10-10
V0.1.3	域名变更 调整错误输出	胡淮波	2014-10-11
V0.1.4	域名补充 调整输出的命名规范	胡淮波	2014-10-16
V0.1.5	命名格式规范 完善 query规范	胡淮波	2014-10-24

系统域名

根域为 sdp(shared platform)

dev.sdp.nd 门户 WEB 站点（开发者使用）
api.sdp.nd 门户 RESTful API（开发者使用）
各个子服务采用以下域名进行访问 *.service.sdp.nd（这些域名仅用于门户与服务之间或服务与服务之间的访问，用户无法直接使用） 注：api.sdp.nd 起 proxy的作用，在之上可以做安全、监控、日志等
弹性web框架 web.service.sdp.nd
CI&CD cc.service.sdp.nd //CI&CD服务 svn.sdp.nd //svn服务(开发者需要访问) nexus.sdp.nd // maven(开发者需要访问) jenkins.cc.service.sdp.nd //jenkins服务
关系数据库 rds.service.sdp.nd
mongodb mongo.service.sdp.nd
IM im.service.sdp.nd
push push.service.sdp.nd
分布式存储 store.service.sdp.nd
全文检索 search.service.sdp.nd
101帐号

```
account.service.sdp.nd
org.service.sdp.nd
api.account.service.sdp.nd
```

基本规范

REST

按照 [Richardson Maturity Mode](#) 对于REST评价的模型， 本规范基于 level2 来设计

几个关键问题的方案

- 版本，使用 HTTP Header 中的 Accept 来实现，如 `Accept: application/sdp.nd+json; version=1`
- 非标准 HTTP Verb，采用 `POST /{resources}/{resource_id}/actions/{action}` 的方式，参见 [Thoughts on RESTful API Design](#)
- Authentication，采用 OAuth2 支持 `bearer_token` 及 `mac_token`。详情见 [制作中](#)
- API 上的资源都统一采用 `uuid` 做为唯一标识
- CRUDL 中 L 的方案参考 [OData](#)。详情见下文：查询(L)方案
- 对于超长的 GET URL，可以采用 POST 来代替

Resource URL

- Resource URL 仅用来表示资源的路径，及一些特殊的 actions
- 以**复数（名词）**进行命名，不管返回单个或多个资源
- 使用**小写、数字及下划线**（下划线用来区分多个单词，如 `access_token`，为了与 json 对象及属性的命名方案保存一致不使用连字符“-”）
- 资源的路径从根到子依次如下 `/ {resources}/{resource_id}/{sub_resources}/{sub_resource_id}/{sub_resource_property}`
如 `GET /databases/{id}/name`

DTO(JSON)

- 资源都建议使用 `uuid` 作为唯一的标识，使用8-4-4-12的格式小写uuid，命名为 `id`
- JONS 中的 json-name 采用**小写、数字及下划线**进行命名，如 `access_token`
- 日期，建议采用 [ISO-8601](#) 的规范，如(1994-11-05T13:15:30Z)，在一些场合下也可以采用 [Unix time](#)

HTTP VERBS(CRUD)

POST(C)

创建资源；执行资源的actions；超长的 GET

GET(R)

获取或查询资源

PUT/PATCH(U)

更新资源（可以使用 PATCH 来标识局部更新,如果不支持，可以使用 X-HTTP-Method-Override: PATCH），PUT 与 PATCH 区别参考 [PUT/PATCH](#)

DELETE(D)

删除资源

错误

- 对于错误的响应，首先应遵循 HTTP 中关于 [status code](#) 的规范
- 总是返回以下结构化的数据

```
{
  "code":500001,
  "message":"{error message}",
  "request_id":"01234567-89ab-cdef-0123-456789abcdef",
  "host_id":"{server identity}",
  "server time":"2014-01-01T12:00:00Z"
```

```
}
```

其中

code 为6位的整数，高3位与http status code一致，低3位为自定义错误号，默认为 000

message 为错误的摘要信息，并且应该包含对用户处理该错误有指导意义的信息

request_id 为错误的uuid，用于帮助技术人员在日志系统中获得错误的详细

host_id 为发生错误的服务器

server_time 为发生错误时的服务器时间

状态码的使用

为每个请求返回适当的状态码 参见 [status code](#)

- 200 OK，当GET请求成功完成，DELETE或者PATCH请求同步完成。
- 201 Created，对于那些要服务器创建对象的请求来说，资源已创建完毕。
- 202 Accepted，POST，DELETE或者PATCH请求提交成功，稍后将异步的进行处理。
- 204 No Content，Response中包含一些Header和一个状态行，但不包括实体的主题内容（没有response body）
- 304 Not Modified，客户的缓存资源是最新的，要客户端使用缓存
- 400 Bad Request
- 401 Unauthorized: 请求失败，因为用户没有进行认证
- 403 Forbidden: 请求失败，因为用户被认定没有访问特定资源的权限
- 405 Method Not Allowed: 不支持该 Request 的方法
- 409 Conflict: 发出的请求在资源上造成了一些冲突
- 429 Too Many Requests: 请求频率超配，稍后再试。
- 500 Internal Server Error: 服务器遇到一个错误，使其无法为请求提供服务
- 501 Not Implemented: 客户端发起的请求超出服务器的能力范围(比如，使用了服务器不支持的请求方法)时，使用此状态码。
- 502 Bad Gateway: 代理使用的服务器遇到了上游的无效响应
- 503 Service Unavailable: 服务器目前无法为请求提供服务，但过一段时间就可以恢复服务

查询(L)方案

本方案参考 [Odata](#) 协议

查询有包含3个部分：过滤(filtering)、排序(sorting)和投影(projection)

URI的结构

```
http://rds.services.sdp.nd/databases?$filter=create_at gt 2014-12-01T12:00:00Z&$orderby=name&$offset=5&$limit=2&$select=id,name
_____ / _____ /
|                                     |
resource path                       query options
```

query options

为了区分与业务的参数进行区分，所有的 query options 使用 \$ 开头

\$filter

相当于 SQL 中的 where，语法为 `$filter={field} {operate} {value}`，如 `name eq 'myname'`

基于性能及实现的考虑：

- 仅支持单个字段的条件查询
- 仅支持有限的运算符

以下是对语法的解析：

`{field}` 为查询资源的字段名称

`{operate}` 为二元运算符，支持如下

eq	等于
gt	大于
ge	大于等于
lt	小于
le	小于等于

不支持 ne 不等于，主要基于性能的考虑

{value} 为运算比较的值，支持以下类型

1000	数值
'string'	字符串, 头尾使用单引号, 如果内容有单引号, 使用二个单引号进行转义
true/false	布尔型
2014-01-01T12:00:00Z	日期时间类型, 也可以如 2014-01-01

\$orderby

排序, 语法为 `$orderby={field} [asc|desc]`

- 仅支持单个字符的排序

以下是对语法的解析:

{field} 为查询资源的字段名称

[asc|desc] 排序的方向, 默认为 asc

\$offset

表示返回数据的偏移量, 如 `$offset=10`

一般结合 \$limit 来做分页的查询

\$limit

表示返回数据的最大笔数, 如 `$limit=10`

一般结合 \$offset 来做分页的查询

\$select

表示返回资源哪些列的数据, 语法为 `$select={field}[,{field}]*`

这个是一个投影操作

\$count

- 表示返回数据是否包含总条数, 语法为 `$count=[false|true]`, 默认值为 false
- 表示仅返回查询的总条数, 语法为 `/ {resourcepath} / $count [? $filter = {filter}]`

返回结果

根据不同的 query options 会有不同的返回类型

仅返回数据

```
GET /{resourcepath}?filter={filter}

// RESPONSE
{
  "value": [
    {
      "id": "01234567-89ab-cdef-0123-456789abcdef",
      "name": "example",
      ...
    },
    ...
  ]
}
```

返回数据与总条数

```
GET /{resourcepath}?filter={filter}&$count=true

// RESPONSE
{
  "count":100,
  "value":[
    {
      "id":"01234567-89ab-cdef-0123-456789abcdef",
      "name": "example",
      ...
    },
    ...
  ]
}
```

仅返回总条数

```
GET /{resourcepath}/$count=?filter={filter}

// RESPONSE
100
```

复杂的查询

在上述方法无法实现的查询，可以使用命名查询的方法，如

```
GET /databases/query_myquery?namelike=mydatabase
```

建议采用一个 query 作为前缀

接口文档的编写规范

应该采用与 HTTP 协议内容来描述接口的请求及响应，如

```
请求
POST /databases HTTP/1.1
HOST: rds.services.sdp.nd
Accept: application/sdp.nd+json; version=1
Content-Type: application/json

{
  "name": "example",
  ...
}

响应
HTTP/1.1 201 Created
Location: http://rds.services.sdp.nd/databases/01234567-89ab-cdef-0123-456789abcdef
ETag: "0123456789abcdef0123456789abcdef"
Last-Modified: Sun, 01 Jan 2012 12:00:00 GMT
Content-Type: application/json

{
  "id":"01234567-89ab-cdef-0123-456789abcdef",
  "name": "example",
  "user": "user",
  "password": "password"
  "created_at":"2014-01-01T12:00:00Z",
  "updated_at":"2014-01-01T12:00:00Z",
  ...
}
```

CRUDL 示例

创建（POST）

使用 `POST /{resources}`

请求示例

```
POST /databases
HOST: rds.services.sdp.nd
Accept: application/sdp.nd+json; version=1
Content-Type: application/json

{
  "name": "example",
  ...
}
```

响应示例

```
HTTP/1.1 201 Created
Location: http://rds.services.sdp.nd/database/01234567-89ab-cdef-0123-456789abcdef
ETag: "0123456789abcdef0123456789abcdef"
Last-Modified: Sun, 01 Jan 2012 12:00:00 GMT
Content-Type: application/json

{
  "id": "01234567-89ab-cdef-0123-456789abcdef",
  "name": "example",
  "user": "user",
  "password": "password"
  "created_at": "2014-01-01T12:00:00Z",
  "updated_at": "2014-01-01T12:00:00Z",
  ...
}
```

注：
如果create为异步的过程，则返回的状态码为 202

获取（GET）

单个资源的获取：使用 `GET /{resources}/{resource_id}`

请求示例

```
GET /databases/01234567-89ab-cdef-0123-456789abcdef
HOST: rds.services.sdp.nd
Accept: application/sdp.nd+json; version=1
```

响应示例

```
HTTP/1.1 200 OK
ETag: "0123456789abcdef0123456789abcdef"
Last-Modified: Sun, 01 Jan 2014 12:00:00 GMT
Content-Type: application/json

{
  "id": "01234567-89ab-cdef-0123-456789abcdef",
  "name": "example",
  "user": "user",
  "password": "password"
  "created_at": "2014-01-01T12:00:00Z",
  "updated_at": "2014-01-01T12:00:00Z",
  ...
}
```

```
}
```

更新（PUT/PATCH）

使用 PUT `/resources/{resource_id}`

请求示例

```
PUT /databases/01234567-89ab-cdef-0123-456789abcdef
HOST: rds.services.sdp.nd
Accept: application/sdp.nd+json; version=1
Content-Type: application/json

{
  "name": "new name",
  ...
}
```

响应示例

```
HTTP/1.1 200 OK
ETag: "0123456789abcdef0123456789abcdef"
Last-Modified: Sun, 01 Jan 2014 12:00:00 GMT
Content-Type: application/json

{
  "id": "01234567-89ab-cdef-0123-456789abcdef",
  "name": "example",
  "user": "user",
  "password": "password"
  "created_at": "2014-01-01T12:00:00Z",
  "updated_at": "2014-01-01T12:00:00Z",
  ...
}
```

删除（DELETE）

使用 DELETE `/resources/{resource_id}`

请求示例

```
DELETE /databases/01234567-89ab-cdef-0123-456789abcdef
HOST: rds.services.sdp.nd
Accept: application/sdp.nd+json; version=1
```

响应示例

```
HTTP/1.1 200 OK
ETag: "0123456789abcdef0123456789abcdef"
Last-Modified: Sun, 01 Jan 2014 12:00:00 GMT
Content-Type: application/json

{
  "id": "01234567-89ab-cdef-0123-456789abcdef",
  "name": "example",
  "user": "user",
  "password": "password"
  "created_at": "2014-01-01T12:00:00Z",
  "updated_at": "2014-01-01T12:00:00Z",
  ...
}
```

// 如果不响应内容

```
HTTP/1.1 204 No Content
ETag: "0123456789abcdef0123456789abcdef"
```

Last-Modified: Sun, 01 Jan 2014 12:00:00 GMT

查询（GET/POST）

使用 GET `/resources?${filter}={filter_exp}&${orderby}={orderby_exp}`

请求示例

```
GET /databases?${filter}=create_at gt 2014-01-01T12:00:00Z&${orderby}=name
HOST: rds.services.sdp.nd
Accept: application/sdp.nd+json; version=1
```

响应示例

```
HTTP/1.1 200 OK
ETag: "0123456789abcdef0123456789abcdef"
Last-Modified: Sun, 01 Jan 2014 12:00:00 GMT
Content-Type: application/json

{
  "value": [
    {
      "id": "01234567-89ab-cdef-0123-456789abcdef",
      "name": "example",
      "user": "user",
      "password": "password"
      "created_at": "2014-01-01T12:00:00Z",
      "updated_at": "2014-01-01T12:00:00Z",
      ...
    },
    ...
  ]
}
```

请求示例

```
GET /databases/${count}
HOST: rds.services.sdp.nd
Accept: application/sdp.nd+json; version=1
```

响应示例

```
HTTP/1.1 200 OK
ETag: "0123456789abcdef0123456789abcdef"
Last-Modified: Sun, 01 Jan 2014 12:00:00 GMT
Content-Type: application/json

20
```