Laravel API 规范

Laravel/Lumen API 规范

1.路由规范及数据请求

API接口一律不使用表单提交，改为使用 XMLHttpRequest 对象进行异步请求。Jquery提供了$.ajax、$.get、$.post 等方法，Angular提供了Http对象。

1.1 GET

使用场景：获取指定数据，查数据接口，数据安全性较低的接口等

是否可带参数：可附带参数，参数附加在链接后

Laravel示例：

请求链接：http://api.exapmle.com/test?a=params-one&b=params-two

```
Route::get('test', function(){...});
```

Lumen示例：

请求链接：http://api.exapmle.com/test?a=params-one&b=params-two

```
$app->get('test', function() {...});
```

注意 请求链接不建议使用路由参数，一律使用普通传参方法，如下写法不再使用：

```
Route::get('user/{name?}', function($name = null)
{
    return $name;
});
```

1.2 POST

使用场景：登入类接口，大量数据提交，文件上传等

是否可带参数：可附带参数，可附带文件数据

Laravel示例：

请求链接：http://api.exapmle.com/test

1.3 PUT

使用场景：更新类接口，更新数据使用

是否可带参数：同POST

Laravel示例：

请求链接：http://api.exapmle.com/test

```
Route::put('test', function(){...});
```

Lumen示例：

请求链接：http://api.exapmle.com/test

```
$app->put('test', function() {...});
```

1.4 DELETE

使用场景：删除接口，删除指定数据

是否可带参数：同GET(注意)

Laravel示例：

请求链接：http://api.exapmle.com/test?a=params-one&b=params-two

```
Route::delete('test', function(){...});
```

Lumen示例：

请求链接：http://api.exapmle.com/test?a=params-one&b=params-two

```
$app->delete('test', function() {...});
```

1.5 Header

请求头内可以附带权限Token等校验字段，这些参数必须在中间件中进行处理

XMLRequest示例：

```
var request = new XMLHttpRequest();
var formData = new FormData();
request.setRequestHeader('key-one', 'value-one');
http.open('post', 'http://api.exapmle.com/test', true);
http.onload = function (data) {
    if(http.readyState == 4 || http.status == 200) {
        //响应成功处理(http.responseText)
```

```
    }
    else{
        //响应失败处理(http.responseText)
    }
}
http.onerror = function () {
    //请求发送失败处理
};

//发送请求
http.send(formData);
```

Angular示例：

```
let headers = new Headers();
headers.append('key-one', 'value-one');
headers.append('key-two', 'value-two');
let options: RequestOptions = new RequestOptions(headers);
this.http.post(
   "http://api.exapmle.com/test",
   {a:"params-one",b:"params-two"},
   options
).subscribe(res => { }, res => { });
```

Lumen示例：

```
//中间件获取头部信息
$value_one= $request->header('key-one');
$value_two= $request->header('key-two');


//设置允许跨域访问，并且允许接受此类头部参数,设置允许的请求类型(index.php)
header("Access-Control-Allow-Origin:*");
header('Access-Control-Allow-Headers:key-one,key-two');
header('Access-Control-Allow-Methods:PATCH,HEADER,POST,GET,OPTIONS,PUT, DELETE');

//Lumen不支持OPTONS请求，需要手动响应OPTIONS请求(index.php)
if ($_SERVER['REQUEST_METHOD'] == 'OPTIONS') return;
```

Laravel示例：

可参照Lumen示例，特别的Laravel支持了OPTIONS请求，无需单独对OPTIONS请求进行响应

1.6FileUpload

XMLRequest示例：

```
<input type="file" name="file-key" id="file-key" multiple="multiple"
accept="image/*,video/mp4" />

var request = new XMLHttpRequest();
var formData = new FormData();
var file=document.getElementById('file-key').files[0];
formData.append('file-key',file)
http.open('post', 'http://api.exapmle.com/test', true);
http.onload = function (data) {
    if(http.readyState == 4 || http.status == 200) {
        //响应成功处理(http.responseText)
```

```
    }
    else{
        //响应失败处理(http.responseText)
    }
}

request.upload.onprogress = function(evt) {
    //获取上传进度
    var level = Math.round(evt.loaded * 100 / evt.total);
}

http.onerror = function () {
    //请求发送失败处理
};

//发送请求
http.send(formData);
```

ACCEPT可选类型：

文件类型	accept值	
*.3gpp	audio/3gpp, video/3gpp	3GPP Audio/Video
*.ac3	audio/ac3	AC3 Audio
*.asf	allpication/vnd.ms-asf	Advanced Streaming Format
*.au	audio/basic	AU Audio
*.css	text/css	Cascading Style Sheets
*.csv	text/csv	Comma Separated Values
*.doc	application/msword	MS Word Document
*.dot	application/msword	MS Word Template
*.dtd	application/xml-dtd	Document Type Definition
*.dwg	image/vnd.dwg	AutoCAD Drawing Database
*.dxf	image/vnd.dxf	AutoCAD Drawing Interchange Format
*.gif	image/gif	Graphic Interchange Format
*.htm	text/html	HyperText Markup Language
*.html	text/html	HyperText Markup Language
*.jp2	image/jp2	JPEG-2000
*.jpe	image/jpeg	JPEG
*.jpeg	image/jpeg	JPEG
*.jpg	image/jpeg	JPEG
*.js	text/javascript, application/javascript	JavaScript
*.json	application/json	JavaScript Object Notation
*.mp2	audio/mpeg, video/mpeg	MPEG Audio/Video Stream, Layer Ⅱ
*.mp3	audio/mpeg	MPEG Audio Stream, Layer Ⅲ

*.mp4	audio/mp4, video/mp4	MPEG-4 Audio/Video
*.mpeg	video/mpeg	MPEG Video Stream, Layer Ⅱ
*.mpg	video/mpeg	MPEG Video Stream, Layer Ⅱ
*.mpp	application/vnd.ms-project	MS Project Project
*.ogg	application/ogg, audio/ogg	Ogg Vorbis
*.pdf	application/pdf	Portable Document Format
*.png	image/png	Portable Network Graphics
*.pot	application/vnd.ms-powerpoint	MS PowerPoint Template
*.pps	application/vnd.ms-powerpoint	MS PowerPoint Slideshow
*.ppt	application/vnd.ms-powerpoint	MS PowerPoint Presentation
*.rtf	application/rtf, text/rtf	Rich Text Format
*.svf	image/vnd.svf	Simple Vector Format
*.tif	image/tiff	Tagged Image Format File
*.tiff	image/tiff	Tagged Image Format File
*.txt	text/plain	Plain Text
*.wdb	application/vnd.ms-works	MS Works Database
*.wps	application/vnd.ms-works	Works Text Document
*.xhtml	application/xhtml+xml	Extensible HyperText Markup Language
*.xlc	application/vnd.ms-excel	MS Excel Chart
*.xlm	application/vnd.ms-excel	MS Excel Macro
*.xls	application/vnd.ms-excel	MS Excel Spreadsheet
*.xlt	application/vnd.ms-excel	MS Excel Template
*.xlw	application/vnd.ms-excel	MS Excel Workspace
*.xml	text/xml, application/xml	Extensible Markup Language
*.zip	aplication/zip	Compressed Archive

Lumen/Laravel示例：

```
if ($request->hasFile('file-key')&&$request->file('file-key')->isValid()) {

        //文件后缀名称
        $extension = $request->file('file-key')
             ->getClientOriginalExtension();

        //文件原名
        $name = $request->file('file-key')->getClientOriginalName();

        //文件大小
        $size = $request->file('file-key')->getSize();
```

```
        //MIME类型
        $mime = $request->file('file-key')->getMimeType();

        //保存文件到指定路径
        $request->file('file-key')->move('/var/www/upload', 'name.type');
}
```

注意：文件保存后一般提需要供一个文件的可访问URL，多平台开发时请把图片全部放在一个公共的存储区域，确保每个平台都能访问这些文件

2.参数过滤

注意：所有接口必须做参数校验，对参数缺少、参数多余，参数错误这些情况必须进行处理。不允许出现'裸奔'现象。

a: 必要参数校验 所有必须的参数全部判断是否存在，必要时进行类型校验

b:多余参数剔除 只有需要的参数才会被添加到参数队列中，多与参数需要进行清除，但是提供了多余参数不会影响接口的正常使用

c:错误参数全部作为多余参数处理

d:可选参数，如果参数存在，就添加进参数队列中

3.响应数据格式

a:成功消息

```
[
    'result'=>true,
    'message'=>'响应消息，可不设置',
]
```

b.失败消息

```
[
    'result'=>false,
    'message'=>'错误消息，提示错误原因',
]
```

c.数据获取成功

```
[
    'result'=>true,
    'message'=>'响应消息，可不设置',
    'datas'=>any(查询结果数据)
]
```

d:数据获取失败(同失败消息)

4.控制器&模型&路由规则

控制器只处理简单的数据封装逻辑，模型做提供查询方法，但不应该过度封装增加耦合度。

4.1控制器创建

app\Http\Controllers\...

4.2模型创建

app\Http\Model\...　　非复杂项目一般模型不分文件夹

app\Http\Model\Member\Member.php 复制项目分文件夹管理

app\Http\Model\Member\Favorite.php

4.3路由创建

routes\web.php　　默认路由

routes\member.php　用户模块路由（每个模块需要分文件）

路由分组：不同权限的路由需要分组

5.代码风格

模型注释

```
/*
 * exp:      方法描述
 * params:   接口参数
 * return:   返回参数
 * author:   作者
 * date:     修改日期
 */
```

```
接口参数:    array[account,password],number(id),string(name)
返回参数:    boolean(result),array[total,rows],array[result,message,datas],DB::GET
可变返回参:  boolean(result)|string(error_message)
```

模型方法示例

```
/*
 * exp:      用户登入方法
 * params:   array[account,password]
 * return:   boolean(result)
 * author:   zhangsan
 * date:     2017-06-08 20:47:25
 */
 function static function login($params){
 //方法名称小写字母开头，方法名称尽量简单易懂，花括号紧跟圆角括号(必要)

     if($params['account'] == "admin" && $params['password'] == "123456789"){
     //TAB缩进4个空格,数组键名使用单引号括住,字符串使用双引号（建议）

         $res=['result' => true, 'message' => "success message", 'datas' =>
['token'=>"a1b31c4dge543ffewr45g"]];
         //一行代码超过250个字符要换行，否则不要换行（建议）

     }
     else{

         foreach($params as $key => $value) $params[$key] = md5($value);
         //单行foreach代码（建议）

         foreach($params as $key => $value){

             $value = md5($value).time();

             $params[$key]= md5($value);
         }
                 //多行foreach代码（建议）
```

```
        }

        ...

        return true;
    }
```

控制器方法示例

```
//登入接口
function login(Request $request){

    $params = RequestApp::RequestToArray(['account', 'password']);

    $params['datas'] = RequestApp::AppendExp(['name'], $request, $params['datas']);

    if($params['result']){

        return RequestApp::AutoMessage(Member::login($params),"登入成功","账号或密码错
误");

    }
    else{

        return RequestApp::ErrorMessage("参数错误");

    }

}
```