|  |
| --- |
| ItTrending |
| EFFC 3.5.1说明 |
| EFFC框架3.5.1版本调整说明 |

|  |
| --- |
| Chuan.yin  2017-5-8 |

目录

[前言 2](#_Toc482027473)

[修改说明 3](#_Toc482027474)

[出现的问题 6](#_Toc482027475)

[引用文献 7](#_Toc482027476)

[历史 8](#_Toc482027477)

# 前言

本文档用于说明EFFC 3.5.1版本的设计及改动说明，用于今后作为EFFC框架下开发指导之基础。

本文档的文件版本与EFFC框架版本相同，从本版本开始，所有的EFFC说明文档都采用Word作为记录载体。

# 修改说明

1. 更清晰的定义Modular，Data，Resource，Settings，梳理Proxy，Unit，Module三者之间的关系，定义如下图所示：
   1. Module



* 1. Module Proxy



* 1. Unit



* 1. Unit Proxy



1. 新增FrameExposed系列公共方法工具，该系列工具主要特性为通过动态对象来快速访问对象或类中的方法或提供便利访问方式，
   1. FrameExposedClass：可以快速访问类中定义的静态访问，忽视访问权限
   2. FrameExposedObject：可以快速访问对象中对应的各种方法和属性，忽视访问权限
   3. FrameExposedArray：可以快速访问数组中的各个对象，通过对象类型来获取数组中的对象
2. 新增扩展模块EWRA-EFFC Web RestFUL API开发接口定义，包含以下特性
   1. 访问加密算法-RSA256
   2. URI构成

每个网址代表一种资源（resource），所以网址中不能有动词，只能有名词，格式如下：

[http://api.xxx.com/{verison}/{entrypoint}/](http://api.xxx.com/%7bverison%7d/%7bentrypoint%7d/){id}/{subpoint}/{id}?{filter}

其中：

Version：api版本号，需要添加默认版本号的设定，version为目录定义格式为V{0.0}或v{0.0}

entrypoint：资源集合入口名称

id：资源单个实例标识符，可忽略

subpoint：单体实例的子资源集合入口

filter：过滤信息条件，如

* ?limit=10：每页笔数
* ?page=2：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。
* ?animal\_type\_id=1：指定筛选条件
  1. Http状态含义
     1. 200 OK - [GET]：服务器成功返回用户请求的数据，该操作是幂等的（Idempotent）。
     2. 201 CREATED - [POST/PUT/PATCH]：用户新建或修改数据成功。
     3. 202 Accepted - [\*]：表示一个请求已经进入后台排队（异步任务）
     4. 204 NO CONTENT - [DELETE]：用户删除数据成功。
     5. 400 INVALID REQUEST - [\*]：用户发出的请求有错误，服务器没有进行新建或修改数据的操作，该操作是幂等的。
     6. 401 Unauthorized - [\*]：表示用户没有权限（令牌、用户名、密码错误）。
     7. 403 Forbidden - [\*] 表示用户得到授权（与401错误相对），但是访问是被禁止的。
     8. 404 NOT FOUND - [\*]：用户发出的请求针对的是不存在的记录或者操作失败，服务器没有进行操作，该操作是幂等的。
     9. 406 Not Acceptable - [GET]：用户请求的格式不可得（比如用户请求JSON格式，但是只有XML格式）。
     10. 410 Gone -[GET]：用户请求的资源被永久删除，且不会再得到的。
     11. 422 Unprocesable entity - [POST/PUT/PATCH] 当创建一个对象时，发生一个验证错误。
     12. 500 INTERNAL SERVER ERROR - [\*]：服务器发生错误，用户将无法判断发出的请求是否成功。
  2. 安全校验机制-AuthenrizeLogic
     1. 框架提供一个基本的校验处理logic- AuthorizationLogic，该logic提供CreateToken(string id) 和IsValid（token）的方法（这两个方法为virsual方法，可以重载，每个business Assembly中只能有一个AuthenrizeLogic，如果有多个，只有第一个起作用），框架提供的默认校验方案为JWT
     2. 校验流程如下：
        1. 用户登录，client端访问post: /auth/{id}或/authorize/{id}接口，进行用户校验，通过后获取token，如果返回值为空则表明登录校验失败
        2. Client拿到token后，每次访问api都需要在header中添加'Authorization': 'Bearer ' + token，如果校验成功则api返回正确的资源，否则返回401
        3. 当校验不通过时，client需要判断是否为超时导致，返回的response body中的error开头为”TimeOut:”即为超时
     3. CreateToken为对外接口，而IsValid只在框架内作用，不对外
     4. EWRAAuthAttribute用于标识哪些method不需要进行校验处理，框架默认除了/auth或/authorize不需要进行校验外，其他的都需要进行校验。标记[EWRAAuth(false)]的method不会参与校验，但如果该method为树形调用方式的情况，则整个调用链上的method都需要添加[EWRAAuth(false)]，否则也会参与校验作业
  3. Rest返回的结果集格式规定
     1. StatusCode为2xx时：返回的内容格式为：{code:处理结果定义代码,result:数据结果对象}
     2. StatusCode为4xx时：返回的内容格式为{error:错误信息}
  4. 返回结果
     1. GET /collection：返回资源对象的列表（数组）
     2. GET /collection/resource：返回单个资源对象
     3. POST /collection：返回新生成的资源对象
     4. PUT /collection/resource：返回完整的资源对象
     5. PATCH /collection/resource：返回完整的资源对象
     6. DELETE /collection/resource：返回一个空文档
  5. HyperMedia API

RESTful API最好做到Hypermedia，即返回结果中提供链接，连向其他API方法，使得用户不查文档，也知道下一步应该做什么。

比如，当用户向api.example.com的根目录发出请求，会得到这样一个文档。

{"link": {

"rel": "collection https://www.example.com/zoos",

"href": "https://api.example.com/zoos",

"title": "List of zoos",

"type": "application/vnd.yourformat+json"

}}

* 1. 专用RestFUL的Logic定义
     1. 提供get，put，post，delete的基本方法识别操作，取消原来gologic中的logicdata做统一参数的方式，采用自定义参数识别方式
     2. RestLogic中只提供GET，HEAD，POST，PUT，DELETE，LOCK，MKLOC，COPY，MOVE，PATCH这些方法，常用方法为GET，POST（Create），PUT（Update），PATCH（deferent Update），DELETE
     3. 通过URL获取资源请求关系表，如Get: /users/zhangsan/salary，表示获取用户张三的工资信息，则框架处理结构为访问Users这个logic，张三作为第一个参数id，
  2. 添加访问时间过长等无响应状况的处理
  3. 计算结果缓存机制
     1. 根据Rest的method和Logic中获取的参数和登录校验的账号密码做key进行缓存作业，只对处理结果做缓存，中间数据层的缓存不在此列
        1. 只对get做缓存操作
        2. Put/patch/post来执行缓存的同步更新操作（在RestLogic中提供SetRefreshCacheRoute来保存要缓存的get方法的列表）
     2. 利用系统的缓存接口做缓存实现介质，未来可以使用redis做缓存解决方案
  4. 定义入口基类RestLoigc
     1. 实现get,post,put,patch,delete这5个方法，所有的方法名称只能为这5个
     2. 路由必须为唯一的，不可重复
     3. 通过RestLoigc和其继承关系建立树形路由访问路径，子类方法中的最后一个参数名称以“parent\_”开头（该参数只能有一个，且放在方法参数的最后位置），表示子路径，如:/zoo/{name}/animal/{name}，/animal/{name}就表示名为animal的logic下有个get(string name,object parent\_info)的方法
     4. 所有的方法的参数类型都只能是string类型，但标记树结构关系的参数（parent\_开头的参数）可以为任意数据类型，但该类型必须与父路径中对应方法的返回值类型是相同的，否则不会计入API路由接口
     5. parent\_开头的参数为上级get方法计算出来的结果，不可为null，否则直接返回404错误（安全考量）
     6. 提供PostData的访问方式，但不提供QueryString的访问方式，QueryString只提供以下固定名称的参数
        1. limit：每页笔数，必须为正整数
        2. page:页数，必须为正整数
  5. Attribute定义
     1. EWRARouteAttribute-自定义独立API入口，使用该属性后，则标识的method被定义为独立路由入口，其路由路径由EWRARouteAttribute.Route标识，对应的方法由EWRARouteAttribute.MethodVerb定义，因此方法名可以任意自定义
        1. Route：格式为:/xxx/yyy/xxx，可以含参数名称，如：/zoo/{name}/animal/，如果未定义参数，则会自动在最后添加参数定义以完善路由。
        2. MethodVerb：只能为get,post,put,patch,delete这5个方法，否则不计入API入口
     2. EWRARouteDesc-API接口的描述，会显示在API接口列表画面中
     3. EWRAAuth-设定API接口是否参与校验的属性，系统默认是都会参与校验，设定EWRAAuth为false后则改接口不参与校验
     4. EWRAIsOpen-设定API接口是否公开，系统默认public的method为公开API，如果EWRAIsOpen设定为false则改接口不公开，并且不可使用
     5. EWRAValid-API执行之前的数据校验接口，为一个大类提供校验的方法
        1. EWRAEmpty-针对post数据进行是否为空的校验，参数为post数据名称列表，用逗号分隔
        2. EWRAIntValid-针对post数据进行是否为int的校验,如果校验成功则将对应的数据转化成int类型，参数为post数据名称列表，用逗号分隔
        3. EWRADoubleValid-针对post数据进行是否为double的校验,如果校验成功则将对应的数据转化成double类型，参数为post数据名称列表，用逗号分隔
        4. EWRADateTimeValid-针对post数据进行是否为datetime的校验,如果校验成功则将对应的数据转化成datetime类型，参数为post数据名称列表，用逗号分隔

# 出现的问题

# 引用文献

1. RestFUL架构参考：http://www.ruanyifeng.com/blog/2014/05/restful\_api.html

# 历史