Settings

REST framework所有的配置参数都放置在Django的settings文件中的 `REST_FRAMEWORK` 变量中。

看起来像下面的样子：

```
REST_FRAMEWORK = {
    'DEFAULT_RENDERER_CLASSES': (
        'rest_framework.renderers.JSONRenderer',
    ),
    'DEFAULT_PARSER_CLASSES': (
        'rest_framework.parsers.JSONParser',
    )
}
```

一、获取配置项的值

使用 `api_settings` 对象来访问你在settings中配置的参数值，例如：

```
from rest_framework.settings import api_settings

print(api_settings.DEFAULT_AUTHENTICATION_CLASSES)
```

`api_settings` 首先会检查用户是否在settings中设置了该参数，如果没有，就返回框架默认的参数值。

二、API运行策略配置项

下面的配置项用于控制基本的API策略，会应用于所有的基于APIView的类视图，以及使用 `@api_view` 装饰器的基于函数的视图。

- **DEFAULT_RENDERER_CLASSES**

一个渲染器类的列表或元组，用于确定在响应时使用的渲染器。

框架默认值:

```
(
    'rest_framework.renderers.JSONRenderer',
    'rest_framework.renderers.BrowsableAPIRenderer',
)
```

- **DEFAULT_PARSER_CLASSES**

一个解析器类的列表或元组，用于确定使用哪个解析器解析 `request.data` 数据。

默认配置:

```
(
    'rest_framework.parsers.JSONParser',
    'rest_framework.parsers.FormParser',
    'rest_framework.parsers.MultiPartParser'
)
```

- **DEFAULT_AUTHENTICATION_CLASSES**

一个认证类的列表或元组，指定使用何种认证方式验证 `request.user` 或 `request.auth` 属性。

默认配置:

```
(
    'rest_framework.authentication.SessionAuthentication',
    'rest_framework.authentication.BasicAuthentication'
)
```

- **DEFAULT_PERMISSION_CLASSES**

一个权限类的列表或元组，用于指定视图的权限。

默认值:

```
(
    'rest_framework.permissions.AllowAny',
)
```

- **DEFAULT_THROTTLE_CLASSES**

一个限流类的列表或元组，用于限制视图的访问次数。

默认值为空: `()`

- **DEFAULT_CONTENT_NEGOTIATION_CLASS**

指定用于内容协商的类。

默认值: `'rest_framework.negotiation.DefaultContentNegotiation'`

- **DEFAULT_SCHEMA_CLASS**

指定视图检查类，用于模式生成。

默认值: `'rest_framework.schemas.AutoSchema'`

三、通用视图配置项

下面的配置项用于控制通用的基于类的视图的行为。

- **DEFAULT_FILTER_BACKENDS**

指定过滤器。如果设置为 `None` ，将关闭该功能。

- **PAGE_SIZE**

指定分页时，每页显示的数量。如果设置为None，将关闭分页功能。

默认值: `None`

- **SEARCH_PARAM**

查询参数的名字，用于 `SearchFilter` .

默认值: `search`

- **ORDERING_PARAM**

排序参数的名字，用于 `OrderingFilter` .

默认值: `ordering`

四、版本相关配置项

- **DEFAULT_VERSION**

用于 `request.version` ，如果没有显式提供版本信息。

默认值: `None`

- **ALLOWED_VERSIONS**

允许使用的版本。如果提供的版本不再这个配置项的范围之内，会弹出错误。

默认值: `None`

- **VERSION_PARAM**

版本参数的名字。

默认值: `'version'`

五、认证相关的配置项

下面的配置项用于控制未认证的请求的行为：

- **UNAUTHENTICATED_USER**

用于初始化未认证请求中 `request.user` 属性的值。如果移除了认证框架，比如将 `django.contrib.auth` 从 `INSTALLED_APPS` 中删除，请将 `UNAUTHENTICATED_USER` 设置为 `None` 。

应该给该配置项提供一个可调用的方法或者None，比如：

```
1   “UNAUTHENTICATED_USER”:  lambda  :  "匿名用户"
```

默认值: `django.contrib.auth.models.AnonymousUser`

- **UNAUTHENTICATED_TOKEN**

用于初始化未认证请求中 `request.auth` 的值。应该给该配置项提供一个可调用的方法或者None，同上。

默认值: `None`

六、测试相关配置项

下面的配置项用于配置APIRequestFactory和APIClient

- **TEST_REQUEST_DEFAULT_FORMAT**

制造测试请求时默认的格式。它必须匹配 `TEST_REQUEST_RENDERER_CLASSES` 设置中的某个渲染器.

默认值: `'multipart'`

- **TEST_REQUEST_RENDERER_CLASSES**

当构造测试请求时支持的渲染器类。

例如: `client.post('/users', {'username': 'jamie'}, format='json')`

默认值:

```
(
    'rest_framework.renderers.MultiPartRenderer',
    'rest_framework.renderers.JSONRenderer'
)
```

七、模式生成控制配置项

- **SCHEMA_COERCE_PATH_PK**

如果设置，则 `'pk'` 在生成模式路径参数时，将URL conf中的标识符映射到实际字段名称。通常这将是 `'id'` 。这给出了更合适的表示，因为“主键”是实现细节，而“标识符”是更一般的概念。

默认值: `True`

- **SCHEMA_COERCE_METHOD_NAMES**

如果设置，则用于将内部视图集方法名称映射到模式生成中使用的外部操作名称。这允许我们生成更适合外部表示的名称，而不是代码库内部使用的名称。

默认值: `{'retrieve': 'read', 'destroy': 'delete'}`

八、内容类型控制

- **URL_FORMAT_OVERRIDE**

一个URL参数的名字，用于覆写默认的内容协商头部中 `Accept` 属性的值。比如 `http://example.com/organizations/?format=csv` ，这时将使用csv内容类型。

如果设置为 `None` ，那么URL格式覆写的功能将关闭。

默认参数名: `'format'`

- **FORMAT_SUFFIX_KWARG**

URL conf中的参数名称，可用于提供格式后缀。当使用 `format_suffix_patterns` 包含URL后缀模式时，将应用此设置。

例如： `http://example.com/organizations.csv/`

默认值: `'format'`

九、日期和时间格式配置

- **DATETIME_FORMAT**

指定 `DateTimeField` 字段在渲染的时候使用的字符串格式，如果设置为None，则返回一个Python的datatime对象。可以设置的值有 `None` ， `'iso-8601'` 或者一个Python的strftime格式字符串。

默认值: `'iso-8601'`

- **DATETIME_INPUT_FORMATS**

输入的日期时间格式。一个列表，包含类似 `'iso-8601'` 或者Python的strftime格式字符串。

默认值: `['iso-8601']`

- **DATE_FORMAT**

针对 `DateField` 日期字段，基本类似DATETIME_FORMAT配置项。

默认值: `'iso-8601'`

- **DATE_INPUT_FORMATS**

类似DATETIME_INPUT_FORMATS。

默认值: `['iso-8601']`

- **TIME_FORMAT**

针对 `TimeField` 时间字段，基本类似DATE_FORMAT配置项。

默认值: `'iso-8601'`

- **TIME_INPUT_FORMATS**

基本类似DATETIME_INPUT_FORMATS。

默认值: `['iso-8601']`

十、编码

- **UNICODE_JSON**

如果设置为True，那么将允许在JSON响应中出现unicode特殊字符，例如：

```
1   {"unicode black star":"★"}
```

如果设置为 `False` 将转义非ascii字符，例如：

```
1   {"unicode black star":"\u2605"}
```

默认: `True`

- **COMPACT_JSON**

当设置为 `True` ，将使用紧凑风格的json响应，多余的空格被去除。

```
1   {"is_admin":false,"email":"jane@example"}
```

设置为 `False` ，允许存在一定的空格，例如：

```
1   {"is_admin": false, "email": "jane@example"}
```

默认值: `True`

- **STRICT_JSON**

当设置为 `True`，只接收严格语法的json数据。

默认值: `True`

- **COERCE_DECIMAL_TO_STRING**

与 `DecimalField` 字段相关，很少使用。

默认值: `True`

十一、视图名和描述

- **VIEW_NAME_FUNCTION**

该配置项用于指示生成视图名的函数。

Default: `'rest_framework.views.get_view_name'`

- **VIEW_DESCRIPTION_FUNCTION**

该配置项用于指示生成视图描述的函数。

Default: `'rest_framework.views.get_view_description'`

十二、HTML下拉框截断

- **HTML_SELECT_CUTOFF**

设置全局的 `html_cutoff` 值，必须是一个正整数。

默认值: 1000

- **HTML_SELECT_CUTOFF_TEXT**

一个字符串，用来表示 `html_cutoff_text` 属性。

默认值: `"More than {count} items..."`

十三、其它配置项

- **EXCEPTION_HANDLER**

一个字符串。表示处理异常的函数。如果函数返回None，将弹出500异常。

Default: `'rest_framework.views.exception_handler'`

- **NON_FIELD_ERRORS_KEY**

一个字符串，用于指示非字段错误的键。

默认值: `'non_field_errors'`

- **URL_FIELD_NAME**

一个字符串，用于表示 `HyperlinkedModelSerializer` 生成的URL字段的键。也就是设置那个自动生成的url字段你想让它叫什么名字。

默认值: `'url'`

- **NUM_PROXIES**

一个大于等于0的整数，用于指定API后台运行的app代理的数量。

默认值: `None`