

# 一、内容协商Content negotiation

内容协商：客户端与服务器协商使用哪种数据格式返回给客户端的过程。

DRF框架的内容协商比较简单，根据可用的渲染器、每个渲染器的优先级以及客户端的 `Accept` 头部属性，确定应将哪种媒体类型返回给客户端。

比如，对于下面的 `Accept` 属性：

```
1 application/json; indent=4, application/json, application/yaml, text/html,
  */*
```

那么优先级顺序如下：

1. `application/json; indent=4`
2. `application/json` , `application/yaml` and `text/html`
3. `/*/*`

如果请求的视图只配置了“YAML”和“HTML”的渲染器，那么REST框架将选择在 `renderer_classes` 列表或 `DEFAULT_RENDERER_CLASSES` 设置中最先列出的渲染器。

## 二、自定义内容协商

我们一般不建议自定义内容协商的方法，如果非要如此：

1. 继承 `BaseContentNegotiation` 类。
2. 实现 `.select_parser(request, parsers)` 和 `.select_renderer(request, renderers, format_suffix)` 方法。

`select_parser()` 方法应该返回可用解析器列表中的一个解析器实例。如果没有解析器可以解析请求，则返回None。

`select_renderer()` 方法应该返回一个二元元组(渲染器实例, 媒体类型)，或者弹出一个 `NotAcceptable` 异常。

下面是一个自定义内容协商类，它会选择适当的分析器或渲染器，并忽略客户端的要求。

```
1 from rest_framework.negotiation import BaseContentNegotiation
2
3 class IgnoreClientContentNegotiation(BaseContentNegotiation):
```

```

4     def select_parser(self, request, parsers):
5         """
6         Select the first parser in the `parser_classes` list.
7         """
8         return parsers[0]
9
10    def select_renderer(self, request, renderers, format_suffix):
11        """
12        Select the first renderer in the `renderer_classes` list.
13        """
14        return (renderers[0], renderers[0].media_type)

```

## 三、配置内容协商

可以使用 `DEFAULT_CONTENT_NEGOTIATION_CLASS` 进行全局的内容协商配置。比如下面的例子，使用的是我们上面自定义的 `IgnoreClientContentNegotiation` 类：

```

1  REST_FRAMEWORK = {
2      'DEFAULT_CONTENT_NEGOTIATION_CLASS':
3      'myapp.negotiation.IgnoreClientContentNegotiation',
4  }

```

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

也可以在视图或视图集中使用 `content_negotiation_class` 类属性，指定视图使用的内容协商方法：

```

1  from myapp.negotiation import IgnoreClientContentNegotiation
2  from rest_framework.response import Response
3  from rest_framework.views import APIView
4
5  class NoNegotiationView(APIView):
6      """
7      An example view that does not perform content negotiation.
8      """
9      content_negotiation_class = IgnoreClientContentNegotiation
10
11     def get(self, request, format=None):
12         return Response({
13             'accepted media type': request.accepted_renderer.media_type
14         })

```