Ask Company

리액트와 함께 장고 시작하기 / 장고 DRF

Renderer를 통한 다양한 응답포맷 지원

여러분의 파이썬/장고 페이스메이커가 되겠습니다.

Renderer

같은 Endpoint에서 요청받은 타입에 맞춰, 다양한 응답포맷을 지원

Content-Type, URL의 방법을 통해 Renderer 지정 가능

기본 지원되는 Renderer

JSONRenderer (디폴트 지정) : json.dumps를 통한 JSON 직렬화

media_type → application/json, format → json

BrowsableAPIRenderer (디폴트 지정) : self-document HTML 렌더링

media_type → text/html, format → api

TemplateHTMLRenderer : 지정 템플릿을 통한 렌더링

media_type → text/html, format → api

Response에서 template_name 인자 지정 필요

API 서버라고 해서 모든 응답을 JSON으로 받지 않아도. 경우에 따라서 HTML 응답을 받을 수도. (Server Side Rendering?)

TemplateHTMLRenderer

템플릿을 통해 Render를 수행하기에 별도의 Serializer가 불필요

```
class PostDetail(RetrieveAPIView):
    queryset = Post.objects.all()
    renderer_classes = [TemplateHTMLRenderer]
    template_name = 'blog/post_detail.html'

    def get(self, request, *args, **kwargs):
        return Response({
            'post': self.get_object(),
        })
```

https://www.django-rest-framework.org/topics/html-and-forms/

이 외에도 다양한 Renderer

StaticHTMLRenderer

미리 렌더링된 HTML을 반환. Response 객체 생성 시에 HTML문자열을 직접 지정

```
@api_view(["GET"])
@renderer_classes([StaticHTMLRenderer])
def static_view(request):
    html = """<html><body>example</body></html>"""
    return Response(html)
```

그리고 AdminRenderer, HTMLFormRenderer, MultiPartRenderer 등이 지원됩니다.

써드파티 Renderer

drf-renderer-xlsx (내부적으로 openpyxl 활용, application/xlsx, 포맷: xlsx)

djangorestframework-yaml

djangorestframework-xml

djangorestframework-jsonp

djangorestframework-msgpack

djangorestframework-csv

rest-framework-latex

https://github.com/wharton/drf-renderer-xlsx

Renderer 선택

Renderer 클래스 리스트 지정하기

전역 지정

settings → REST_FRAMEWORK → DEFAULT_RENDERER_CLASSES 리스트에 문자열로 지정

APIView 마다 지정

queryset, serializer_class와 더불어, renderer_classes 리스트

@api_view 마다 지정

renderer_classes 장식자

Renderer 이외의 다른 속성들도 마찬가지 방법으로 설정

Response의 기본 2가지 유형의 응답 포맷

rest_framework.response.Response

- api : API Endpoint에 브라우저를 통해 접근할 때, 웹 UI로 조회 가능
- json : 보통의 API 접근

응답 포맷은 어떻게 결정되는 가?

Accept 헤더

- Accept: application/json
- Accept: text/html

GET인자 format

- ?format=json
- ?format=api

URL Captured Values에서의 format 인자

- .json
- .api

URL Captured Values에서의 format 인자

1. rest_framework.urlpatterns.format_suffix_patterns를 통해, 기존 urlpatterns 끝에 format 인자 지원을 추가
2. DefaultRouter에서는 기본 지원

router.urls에서의 URL Patterns

```
[
    <URLPattern '^post/$' [name='post-list']>,
    <URLPattern '^post\.(?P<format>[a-z0-9]+)/?$' [name='post-list']>,

    <URLPattern '^post/(?P<pk>[^/.]+)/$' [name='post-detail']>,
    <URLPattern '^post/(?P<pk>[^/.]+)\.(?P<format>[a-z0-9]+)/?$' [name='post-detail']>,

    <URLPattern '^$' [name='api-root']>,
    <URLPattern '^\.(?P<format>[a-z0-9]+)/?$' [name='api-root']>
]
```

@api_view에서의 format 인자

함수 기반 뷰에서는 URL Captured Values는 Keyword Arguments를 통해 전달됩니다. 즉, forma인자를 받도록 설정했다면 format인자를 추가해주세요.

views.py

```python
from rest_framework.decorators import api_view

@api_view(['GET'])
def hello(request, format=None):
    return Response([])
```

urls.py

```python
from rest_framework.urlpatterns import format_suffix_patterns

urlpatterns = format_suffix_patterns([
    path('hello/', views.hello),
])
```

생성된 URL Patterns

```
[
    <URLPattern 'hello/'>,
    <URLPattern 'hello<drf_format_suffix:format>'>,
]
```

예시) HTTPie 통한 format 지정

- http :8000/ Accept:application/json
- http :8000/?format=json
- http :8000/.json
- http :8000/ Accept:text/html
- http :8000/?format=api
- http :8000/.api

인생은 짧습니다.
파이썬/장고를 쓰세요.

여러분의 파이썬/장고 페이스메이커가 되겠습니다.

-

Ask Company