# Ask Django

여러분의 파이썬/장고 페이스메이커가 되겠습니다.

**EP 13. Token 인증**

# DRF에서 지원하는 인증

- `rest_framework.authentication.SessionAuthentication`
  - 외부 서비스/앱에서 세션인증을 못 쓰죠.
- `rest_framework.authentication.BasicAuthentication`
  - 외부 서비스/앱에서 매번 username/password를 넘기는 것은 보안상 위험하고, 못할 짓.
- `rest_framework.authentication.TokenAuthentication` #doc
  - 초기에 username/password으로 Token을 발급받고,
  - 이 Token을 매 API요청에 담아서 보내어 인증을 처리

# 시작하기 전에 - 기본 코드

```
# 앱/models.py
class Post(models.Model):
    author = models.ForeignKey(settings.AUTH_USER_MODEL, related_name='+')
    message = models.TextField(blank=True)
    photo = models.ImageField(blank=True)
    created_at = models.DateTimeField(auto_now_add=True)


# 앱/serializers.py
from rest_framework.fields import ReadOnlyField
from rest_framework.serializers import ModelSerializer

class PostSerializer(ModelSerializer):
    author_username = ReadOnlyField(source='author.username')

    class Meta:
        model = Post
        fields = ('id', 'author_username', 'message', 'photo')
```

```python
# 앱/views.py
from rest_framework.authentication import TokenAuthentication
from rest_framework.permissions import IsAuthenticated
from rest_framework.viewsets import ModelViewSet

class PostModelViewSet(ModelViewSet):
    queryset = Post.objects.all()
    serializer_class = PostSerializer
    authentication_classes = [TokenAuthentication]
    permission_classes = [IsAuthenticated]

    def perform_create(self, serializer):
        serializer.save(author=self.request.user)

# 앱/urls.py
router = DefaultRouter()
router.register('post', PostModelViewSet)

urlpatterns = [
    url(r'', include(router.urls)),
]
```

# Token 인증 설정하기

```
# settings

INSTALLED_APPS = (
    'rest_framework.authtoken',
)
```

migrate가 필요합니다.

`rest_framework.authtoken.Token`모델에 대한 테이블을 생성해야 합니다.

# Token 모델 #src

- 각 User에 대해 1:1 Relation => OneToOneField
- 각 User별 Token 인스턴스가 자동생성되지 않습니다.
- Token은 유저 별로 Unique합니다. Token만으로 인증을 수행합니다.

```
# rest_framework/authotken/models.py

class Token(models.Model):  # 모델 코드에서 주요부분만 발췌
    key = models.CharField(max_length=40, primary_key=True)
    user = models.OneToOneField(settings.AUTH_USER_MODEL,
                                related_name='auth_token')
```

# Token 생성

### 방법1) ObtatinAuthToken뷰를 통한 획득 or 생성 #src

token 획득 API 뷰에서 Token 획득/생성을 아래 코드와 같이 처리하므로,
**Token 재생성**이 아니라면 특별히 처리해줄 필요는 없습니다.

```
# rest_framework/authtoken/views.py

class ObtainAuthToken(APIView):
    def post(self, request, *args, **kwargs):
        # 중략
        token, created = Token.objects.get_or_create(user=user)
        return Response({'token': token.key})
```

# Token 생성

**방법2) Signal을 통한 자동 생성 (필수는 아님)**

```
from django.conf import settings
from django.db.models.signals import post_save
from django.dispatch import receiver
from rest_framework.authtoken.models import Token


@receiver(post_save, sender=settings.AUTH_USER_MODEL)
def create_auth_token(sender, instance=None, created=False, **kwargs):
    if created:
        Token.objects.create(user=instance)
```

# Token 생성

### 방법3) Management 명령을 통한 생성

```
# 본 명령은 생성된 Token을 변경하지 않습니다. 필수는 아님.
python3 manage.py drf_create_token <username>

# 강제로 Token 재생성하기
python3 manage.py drf_create_token -r <username>
```

# Token 획득을 API ENDPOINT로 노출하기

```
from rest_framework.authtoken.views import obtain_auth_token

urlpatterns += [
    url(r'^api-token-auth/', obtain_auth_token),
]
```

# HTTPie를 통한 Token 획득

```
쉘> http POST http://주소/api-token-auth/ username="유저명" password="암호"

HTTP/1.0 200 OK
Allow: POST, OPTIONS
Content-Type: application/json
Date: Fri, 01 Dec 2017 06:49:35 GMT
Server: WSGIServer/0.2 CPython/3.6.1

{
    "token": "9cdd705c0a0e5adb8671d22bd6c7a99bbacab227"
}
```

# HTTPie 예제 (맥/리눅스)

```
export HOST="http://localhost:8000"
export TOKEN="9cdd705c0a0e5adb8671d22bd6c7a99bbacab227"

# Post List
http GET $HOST/api/post/ "Authorization: Token $TOKEN"

# Post Create
http POST $HOST/api/post/ "Authorization: Token $TOKEN" message="hello"

# Post Create with Photo
http --form POST $HOST/api/post/ "Authorization: Token $TOKEN" message="hello" photo@"f1.jpg"

# Post#16 Detail
http GET $HOST/api/post/16/ "Authorization: Token $TOKEN"

# Post#16 Update
http PATCH $HOST/api/post/16/ "Authorization: Token $TOKEN" message="patched"
http PUT $HOST/api/post/16/ "Authorization: Token $TOKEN" message="updated"

# Post#16 Delete
http DELETE $HOST/api/post/16/ "Authorization: Token $TOKEN"
```

# 파이썬 코드를 통한 Token 획득

```
import requests

HOST = 'http://localhost:8000'

res = requests.post(HOST + '/api-token-auth/', {
    'username': '유저명',    # FIXME: 기입해주세요.
    'password': '암호',      # FIXME: 기입해주세요.
})
res.raise_for_status()

token = res.json()['token']

print(token)
```

이제 모든 요청에는 다음 인증헤더를 붙여주셔야 합니다.

```
headers = {
    'Authorization': 'Token ' + token,  # 필히 띄워쓰기
}
```

```
# Post List
res = requests.get(HOST + '/api/post/', headers=headers)
res.raise_for_status()
print(res.json())

# Post Create
data = {'message': 'hello requests'}
res = requests.post(HOST + '/api/post/', data=data, headers=headers)
print(res)
print(res.json())

# Post Create with Photo
files = {'photo': open('f1.jpg', 'rb')}
data = {'message': 'hello requests'}
res = requests.post(HOST + '/api/post/', files=files, data=data, headers=headers)
print(res)
print(res.json())
```

```
# Post#16 Detail
res = requests.get(HOST + '/api/post/', headers=headers)
res = requests.get(HOST + '/api/post/16/', headers=headers)
res.raise_for_status()
print(res.json())

# Post#16 Patch
res = requests.patch(HOST + '/api/post/16/', headers=headers, data={'message': 'hello'})
res.raise_for_status()
print(res.json())

# Post#16 Update
res = requests.put(HOST + '/api/post/16/', headers=headers, data={'message': 'hello'})
res.raise_for_status()
print(res.json())

# Post#16 Delete
res = requests.delete(HOST + '/api/post/16/', headers=headers, data={'message': 'hello'})
res.raise_for_status()
print(res.ok)
```