Whois API

좋은징조 - 김형훈

목차

1. Whois API란?
2. Whois Module
3. Whois 결과
4. Whois API Module 사용법 (views.py)

1. Whois API란?

- 국가 인터넷 주소(도메인, IP 주소) 관리기관인 한국 인터넷진흥원이 제공하는 인터넷 주소의 등록, 할당 정보 검색 서비스

2. Whois Module

- python의 whois 모듈을 사용하면 whois API 웹 검색과 동일한 결과를 얻을 수 있다.

```
(env)> pip install python-whois
```

```
(env) C:\Users\kim\django>pip install python-whois
Collecting python-whois
  Using cached
https://files.pythonhosted.org/packages/3e/32/3a7fa80b485f4a1c41492c062396441f2abf28865bb65
d069c228af165f7/python-whois-0.7.1.tar.gz
Requirement already satisfied: future in
c:\users\kim\appdata\local\programs\python\python37-32\lib\site-packages (from python-
whois) (0.17.1)
Installing collected packages: python-whois
  Running setup.py install for python-whois ... done
Successfully installed python-whois-0.7.1
```

3. Whois 결과

```
>>> import whois
>>> print(whois.whois('www.naver.com'))
{
  "domain_name": [
    "NAVER.COM",
    "naver.com"
  ],
  "registrar": "Gabia, Inc.",
  "whois_server": "whois.gabia.com",
  "referral_url": null,
  "updated_date": [
    "2016-08-05 06:37:57",
    "2018-02-28 11:27:15"
  ],
  "creation_date": [
    "1997-09-12 04:00:00",
    "1997-09-12 00:00:00"
  ],
  "expiration_date": [
    "2023-09-11 04:00:00",
    "2023-09-11 00:00:00"
  ],
  "name_servers": [
    "NS1.NAVER.COM",
    "NS2.NAVER.COM",
    "ns1.naver.com",
    "ns2.naver.com"
  ],
  "status": [
    "clientDeleteProhibited https://icann.org/epp#clientDeleteProhibited",
    "clientTransferProhibited https://icann.org/epp#clientTransferProhibited",
    "clientUpdateProhibited https://icann.org/epp#clientUpdateProhibited",
    "ok https://icann.org/epp#ok"
  ],
  "emails": [
    "white.4818@navercorp.com",
    "dl_ssl@navercorp.com",
    "abuse@gabia.com"
  ],
  "dnssec": "unsigned",
  "name": "NAVER Corp.",
  "org": "NAVER Corp.",
  "address": "6 Buljung-ro, Bundang-gu, Seongnam-si, Gyeonggi-do, 463-867, Korea",
  "city": "Gyeonggi",
  "state": null,
  "zipcode": "463463",
  "country": "KR"
}
```

4. Whois API Module 사용법 (views.py)

models.py

```
class WhoisInfo(models.Model):
    qrInfo = models.OneToOneField(QRInfo, related_name='whoisInfo',
on_delete=models.CASCADE, primary_key=True)
    registrar = models.TextField(null=True)
    org = models.TextField(null=True)
    address = models.TextField()
    city = models.TextField()
    country = models.TextField(null=True)

class DomainName(models.Model):
    name = models.CharField(max_length=100, null=True)
    wInfo = models.ForeignKey(WhoisInfo, related_name='domainNames',
on_delete=models.CASCADE)

class EmailInfo(models.Model):
    email = models.EmailField()
    wInfo = models.ForeignKey(WhoisInfo, related_name='emails', on_delete=models.CASCADE)
```

serializers.py

```
class DomainNameSerializer(serializers.ModelSerializer):
    class Meta:
        model = DomainName
        fields = ('name',)

class EmailInfoSerializer(serializers.ModelSerializer):
    class Meta:
        model = EmailInfo
        fields = ('email', )

class WhoisInfoSerializer(serializers.ModelSerializer):
    domainNames = DomainNameSerializer(many=True, read_only=True)
    emails = EmailInfoSerializer(many=True, read_only=True)

    class Meta:
        model = WhoisInfo
        fields = ('registrar', 'org', 'address', 'city', 'country', 'domainNames',
'emails',)
```

views.py

```
from .models import *
from .serializers import *
```

```
from rest_framework import status
from rest_framework.response import Response
from rest_framework.decorators import api_view, renderer_classes
from rest_framework.renderers import JSONRenderer
from tutorial.screenshoter.screenshoters import Screenshoter

import whois

# Create your views here.

@api_view(['GET'])
@renderer_classes((JSONRenderer,))
def QRInformation(request, url):
    url_dict = {'url': url}

    if QRInfo.objects.filter(url=url).exists():
       print("exist!!!")
       qrinfo = QRInfo.objects.get(url=url)
       serial = TestSerializer(qrinfo)
       return Response(serial.data)
    else:
        qrseri = QRInfoSerializer(data=url_dict)
        try:
            qrseri.is_valid(raise_exception=True)
            qrinfo = QRInfo()
            qrinfo = qrseri.save()

            ...

            whoisRAWInfo = whois.whois(qrinfo.url)

            wInfo = WhoisInfo(qrInfo=qrinfo, registrar=whoisRAWInfo['registrar'],
org=whoisRAWInfo['org'], address=whoisRAWInfo['address'], city=whoisRAWInfo['city'],
country=whoisRAWInfo['country'])
            wInfo.save()

            if type(whoisRAWInfo['domain_name']) == list:
                for domainName in whoisRAWInfo['domain_name']:
                    dn = DomainName(name=domainName, wInfo=wInfo)
                    dn.save()
            else:
                dn = DomainName(name=whoisRAWInfo['domain_name'], wInfo=wInfo)
                dn.save()

            if type(whoisRAWInfo['emails']) == list:
                for email in whoisRAWInfo['emails']:
                    em = EmailInfo(email=email, wInfo=wInfo)
                    em.save()
            else:
                em = EmailInfo(name=whoisRAWInfo['emails'], wInfo=wInfo)
                em.save()

            ...
```

```
            result = TestSerializer(qrinfo)
            return Response(result.data)
        except Exception as e:
            print("[DEBUG] validate error : ", e)
            return Response(qrseri.errors, status=status.HTTP_500_INTERNAL_SERVER_ERROR)
```

```
from .models import *
from .serializers import *
from rest_framework import status
from rest_framework.response import Response
from rest_framework.decorators import api_view, renderer_classes
from rest_framework.renderers import JSONRenderer
from tutorial.screenshoter.screenshoters import Screenshoter

import whois # whois 모듈 가져오기

@api_view(['GET'])
@renderer_classes((JSONRenderer,))
def QRInformation(request, url):
            ...

            whoisRAWInfo = whois.whois(qrinfo.url)
            # 요청받은 URL의 whois 결과 값을 저장하는 변수 생성

            wInfo = WhoisInfo(qrInfo=qrinfo, registrar=whoisRAWInfo['registrar'],
org=whoisRAWInfo['org'], address=whoisRAWInfo['address'], city=whoisRAWInfo['city'],
country=whoisRAWInfo['country'])
            # whois 결과 값 중 registrar, org, address, city, country의 값을 추출하여
WhoisInfo 모델에 저장하는 객체 생성

            wInfo.save() # 객체 저장

            if type(whoisRAWInfo['domain_name']) == list:
            # whois 결과 값 중 domain_name 값이 리스트 형식일 경우
                for domainName in whoisRAWInfo['domain_name']: # for 문
                    dn = DomainName(name=domainName, wInfo=wInfo)
                    # whois 결과 값 중 모든 domain_name 값과 객체 저장
                    dn.save() # 도메인 저장
            else: # 아닐 경우
                dn = DomainName(name=whoisRAWInfo['domain_name'], wInfo=wInfo)
                # whois 결과 값 중 domain_name 값과 객체 저장
                dn.save() # 도메인 저장

            if type(whoisRAWInfo['emails']) == list:
            # whois 결과 값 중 email 값이 리스트 형식일 경우
                for email in whoisRAWInfo['emails']: # for 문
                    em = EmailInfo(email=email, wInfo=wInfo)
                    # whois 결과 값 중 모든 email 값과 객체 저장
                    em.save() # 이메일 저장
            else: # 아닐 경우
```

```
                em = EmailInfo(name=whoisRAWInfo['emails'], wInfo=wInfo)
                # whois 결과 값 중 email 값과 객체 저장
                em.save() # 이메일 저장

            ...
```