The DNS Protocol

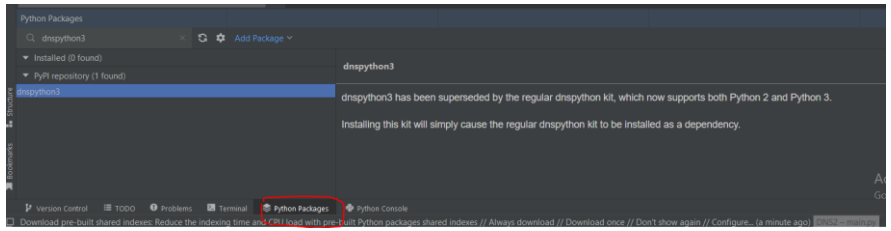
The DNS Protocol

- **The *Domain Name System (DNS)*** is the scheme by which millions of Internet hosts cooperate to answer the question of what hostnames resolve to which IP addresses.
-

```
C:\Users\Shereen>nslookup google.com
Server:  192.168.1.1
Address:  192.168.1.1

Non-authoritative answer:
Name:    google.com
Addresses:  2a00:1450:4006:804::200e
          172.217.171.238


C:\Users\Shereen>
```

Making a DNS Query from Python

- **Installing dnspython3**
  - **From CMD use:** pip install dnspython3
  - From Pycharm: search for dnspyhtons in python package then click install



DNS Record types

- **Commonly used record types**
  - **A** (IPV4 host address): This is the most popular type. A records create a DNS record that points to an IPv4 address. It allows you to use memonic names, such as **www.example.com**, in place of IP addresses like 127.0.0.1.
  - **AAAA** (IPv6 host address)
  - **CNAME** (Canonical name for an alias) is a type of record in the Domain Name System (DNS) used to map a domain name as an alias for another domain. CNAME records always point to another domain name and never directly to an IP address.

DNS Record types

- **MX Record**
  - A MX record also called mail exchanger record is a resource record in the Domain Name System that specifies a mail server responsible for accepting email messages on behalf of a recipient's domain. It also sets the **preference** value used to prioritizing mail delivery if multiple mail servers are available.
- **NS** (Name Server):Directs to name servers, where to ask if you want know about a subdomain
- **TXT** (Descriptive text):used to put miscellaneous info, used to verify ownership of domain

CNAME Records

- **CNAME records** can be used to alias one name to another. CNAME stands for Canonical Name.
- the CNAME record is represented by the following customizable elements:

Element	Description
Name	The host name for the record, without the domain name. This is generally referred to as “subdomain”. We automatically append the domain name.
TTL	The time-to-live in seconds. This is the amount of time the record is allowed to be cached by a resolver.
Content	The domain-name the CNAME maps to.

NS Record

- An NS record delegates a subdomain to a set of name servers. Whenever you delegate a domain to DNSimple.
- For example, there are the following entries delegating dnsimple.com to our name servers in the .com name servers:

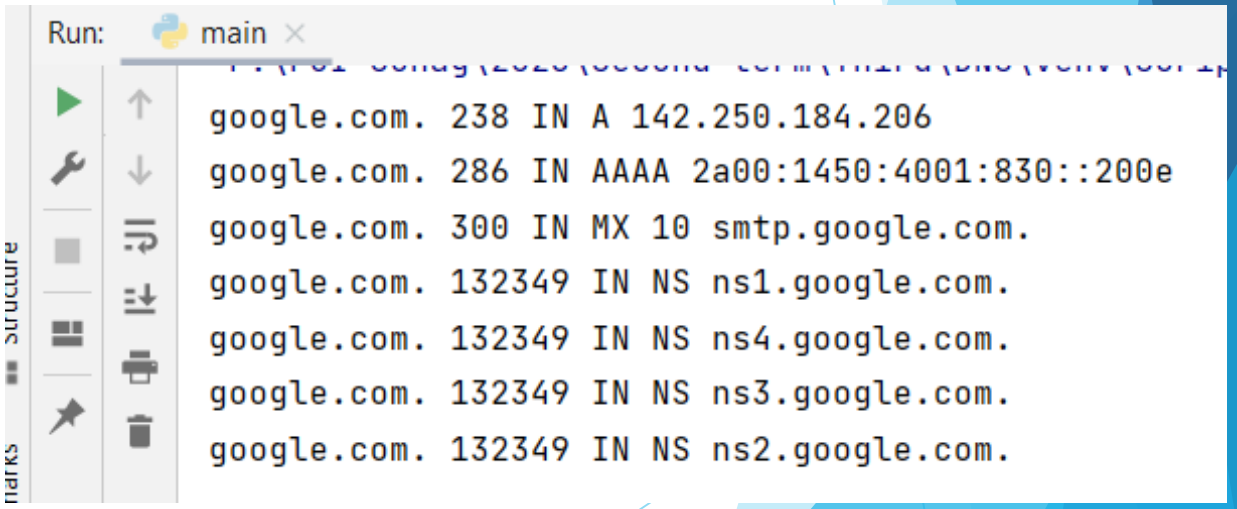
```
dnsimple.com. 172800 IN NS ns1.dnsimple.com.
dnsimple.com. 172800 IN NS ns2.dnsimple-edge.net.
dnsimple.com. 172800 IN NS ns3.dnsimple.com.
dnsimple.com. 172800 IN NS ns4.dnsimple-edge.org.
```

TXT Records

- A TXT record is a resource record used to provide the ability to associate text with a zone.
- This record allows domain administrators to insert any text content into DNS records.
- These records are used for various purposes. One example is ownership validation: To prove you own the domain, a provider may require you to add a TXT record with a particular value to your domain.
- https://support.dnsimple.com/articles/txt-record/

Making a DNS Query from Python

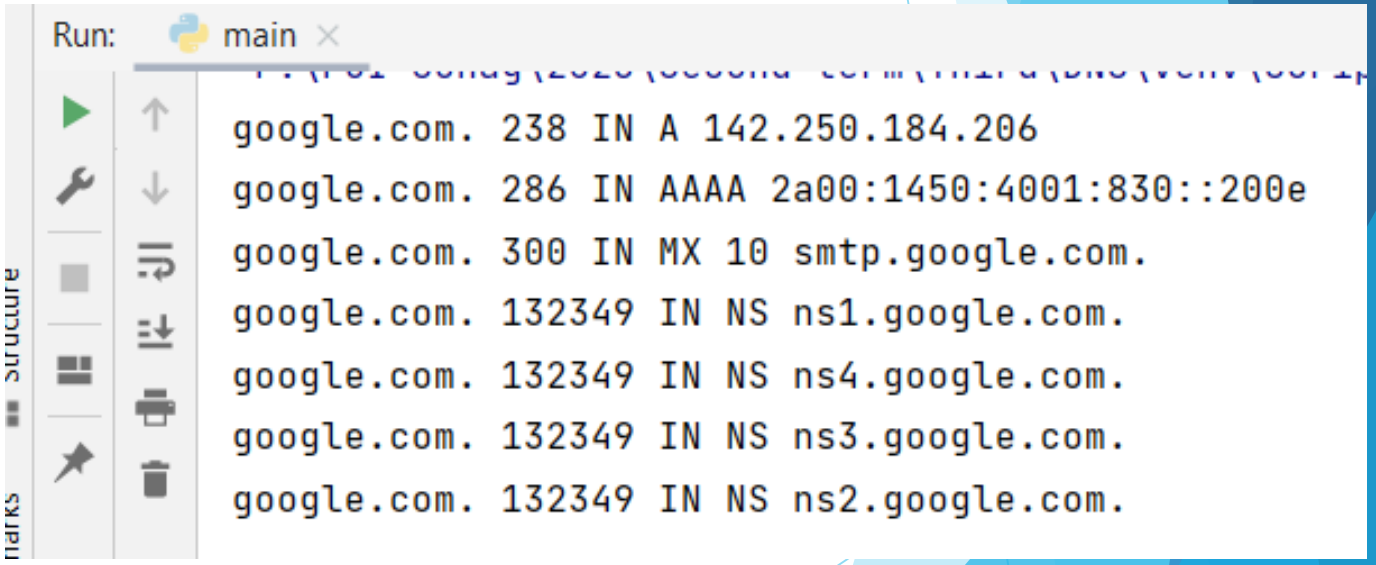
```
import dns.resolver

def lookup(name):

    for RecType in 'A', 'AAAA', 'CNAME', 'MX',
'NS':
        answer = dns.resolver.query(name,
RecType, raise_on_no_answer=False)

        if answer.rrset is not None:
            print(answer.rrset)


lookup('google.com')
```



- raise_on_no_answer: when no answer not return error

The Answer of DNS Query

- **In order, the keys that get printed on each line are as follows:**
  - The name looked up.
  - The time in seconds that you are allowed to cache the name before it expires.
  - The “class” like IN, which indicates that you are being returned Internet address responses.
  - The “type” of record. A, AAAA, NS, and MX for
  - Finally, the “data” provides the information you need to connect to or contact a service



Resolving Mail Domains

- **"For an email address `name@domain` find its mail server IP addresses."**
- **record.preference:** priority (10)
- **record.exchange:** name (smtp.google.com.)
- **record.exchange.to_text(omit_final_dot=True)**: delete dot from the end of the name

```
google.com. 58 IN MX 10 smtp.google.com.
```

Resolving Mail Domains

- **resolve_email_domain(domain) Method**
- Make query for MX record type
  - If exits :
    1. Sort records based on priority from lowest to highest
    2. For each record
       - Get record server name:
         - **name= record.exchange.to_text(omit_final_dot=True)**
       - call **resolve_hostname** (name)
  - else call **resolve_hostname** (domain name)

resolve_hostname method

- **def resolve_hostname(hostname):**

1. Make query using dns.resolver.query with hostname and rType=A
2. If exits :
   - For each record print address
   - return
3. Else : Make query using dns.resolver.query with hostname and rType= AAAA
4. If exits :
   - For each record print address
   - return

resolve_hostname method con’t

5. Else: Make query using dns.resolver.query with hostname and rType= CNAME
6. If exits :
   - For each record call resolve_hostname(cname)
   - Return
7. Else print(indent, 'ERROR: no A, AAAA, or CNAME records for', hostname)

resolve_mail method

```
import dns.resolver

def resolve_mail(domain):
    answer=dns.resolver.query(domain,'MX')

    if answer.rrset is not None:
        records = sorted(answer)
        print('record after sorting')
        print(records)
        for record in records:
           # print(record.preference) # print priority
            #print(record.exchange)  # print name
            name = record.exchange.to_text(omit_final_dot=True)
            resolve_host(name)
    else:
        print('No exchange name')
        resolve_host(domain)
        return
```

resolve_host Method

```
def resolve_host(hostname):
    answer=dns.resolver.query(hostname,'A')
    if answer.rrset is not None:
        for record in answer:
            print('hosname ',hostname,'has address',record.address)
        return
    else:
        answer=dns.resolver.query(hostname,'AAAA')
        if answer.rrset is not None:
            for record in answer:
                print(record.address)
            return
        else:
            answer = dns.resolver.query(hostname, 'CNAME')
            if answer.rrset is not None:
                record = answer[0]
                cname = record.address
                resolve_host(record.address)
            else:
                print('error no a,AAAA or CNAME for host name ')
```

Thank You