prieš pradedant requests naudojimą reikia instaliuotis pačią biblioteką(vieną kartą)

In [ ]:

```
!pip install requests
```

Requests

Requests yra Python biblioteka darbui su HTTP užklausomis. Norint pradėti darbą, reikia importuoti requests:

In [2]:

```
import requests
```

Dabar susikurkime objektą, kuris bus atsakas į mūsų užklausą:

In [3]:

```
r = requests.get("http://google.lt")
```

In [4]:

```
print(r)
print(type(r))
```

```
<Response [200]>
<class 'requests.models.Response'>
```

Pabandžius printinti, atprintina 200 http statuso kodą, šis kodas reiškia, kad mūsų užklausa GET buvo nusiųsta http serveriui ir iš serverio atėjo atsakymas 200 - viskas OK.

Kokius dar merodus ir atributus turi Response?

In [5]:

```
print(dir(r))
```

```
['__attrs__', '__bool__', '__class__', '__delattr__', '__dict__', '__dir__',
'__doc__', '__enter__', '__eq__', '__exit__', '__format__', '__ge__', '__get
attribute__', '__getstate__', '__gt__', '__hash__', '__init__', '__init_subc
lass__', '__iter__', '__le__', '__lt__', '__module__', '__ne__', '__new__',
'__nonzero__', '__reduce__', '__reduce_ex__', '__repr__', '__setattr__', '__
setstate__', '__sizeof__', '__str__', '__subclasshook__', '__weakref__', '_c
ontent', '_content_consumed', '_next', 'apparent_encoding', 'close', 'connec
tion', 'content', 'cookies', 'elapsed', 'encoding', 'headers', 'history', 'i
s_permanent_redirect', 'is_redirect', 'iter_content', 'iter_lines', 'json',
'links', 'next', 'ok', 'raise_for_status', 'raw', 'reason', 'request', 'stat
us_code', 'text', 'url']
```

statuso kodas

In [7]:

```
1 r.status_code
```

Out[7]:

```
200
```

.text - jame talpinamas tekstas gautas iš severio, jo atsakyme

In [8]:

```
1 r.text
```

Out[8]:

```
'<!doctype html><html itemscope="" itemtype="http://schema.org/WebPage" l
ang="lt"><head><meta content="text/html; charset=UTF-8" http-equiv="Conte
nt-Type"><meta content="/images/branding/googleg/1x/googleg_standard_colo
r_128dp.png" itemprop="image"><title>Google</title><script nonce="d-8fLAo
WngOurlGW9MYzAw">(function(){window.google={kEI:\'36b6YsLoMqPTmAWx14-oAw
\',kEXPI:\'0,1302536,56873,6059,206,4804,2316,383,246,5,5367,1123753,1578
490,16115,17444,1953,9287,17572,4859,1361,283,9007,3022,17587,4998,13228,
3847,10622,22741,5081,1593,1279,2742,149,561,542,840,1983,4314,3514,606,2
023,1777,520,14670,3227,2845,7,4811,28959,1851,2614,13142,3,346,230,6459,
149,13975,4,1528,2304,7039,25073,2658,7357,11442,6653,16786,5809,2548,409
4,4052,3,3541,1,42154,2,28138,11623,5679,1020,2381,28742,4567,6256,23421,
1252,5835,14967,4333,6089,1395,27082,8047,108,6582,98,701,14680,2162,517
9,4620,9835,7,1922,9779,23,6865,5527,9364,2,2641,10272,1460,5764,189,198
1,3667,3879,552,983,123,700,4,2,2,2,2,2,1,1,3716,3356,7,1,770,4822,1242,1
207,5328,1647,2206,641,75,1131,374,278,275,1408,398,326,329,2,73,688,1,45
5,891,433,161,612,180,490,82,404,339,139,1106,641,99,26,294,983,878,1659,
335,655,534,114,393,172,581,127,78,310,1254,621,441,24,259,1107,831,2,536
2741,1872,2983,612,8799655,3311,141,795,19735,1,303,44,1194,1,9,3,7,4,239
```

pabandome kitą užklausą, į paveikslėlį

In [19]:

```
1 r = requests.get("https://www.python.org/static/img/python-logo@2x.png")
```

In [21]:

```
1 r.status_code
```

Out[21]:

```
200
```

įvykdžius sekantį kodą pamatysim skirtumą kaip atrodo gaunamas tekstas ir paveikslėlis.

In [ ]:

```
1 r.text
```

In [ ]:

```
r.content
```

pats paveikslėlis yra binary - skaitmeninio, ne tekstinio formato informacija. Jį išsaugoti galime prisiminę failų išsaugojimo mode 'wb'

In [25]:

```
with open('logo.png', mode='wb') as f:
    f.write(r.content)
```

Status kodai

200 - OK 4xx, 5xx ne OK

In [26]:

```
r = requests.get("http://www.google.lt")
print(r.status_code)
```

```
200
```

pabandome užklausą į neegzistuojantį puslapį:

In [27]:

```
r = requests.get("http://www.google.lt/qqq.html")
print(r.status_code)
```

```
404
```

gauname serverio atsakymą - status code 404, reiškiantį kad kreipiamės į neesamą puslapį.

200 yra OK, 404 - Not Found, 500 - Internal server error ir pan. daugiau apie šiuos kodus pasiskaitykite čia - https://www.webfx.com/web-development/glossary/http-status-codes/ (https://www.webfx.com/web-development/glossary/http-status-codes/)

Šie kodai mums naudingi, jeigu tikriname ar pavyko prisijungimas, Tarkime:

In [32]:

```
r = requests.get("http://www.google.lt/blablanla")

if r.status_code not in range(400, 600):
    print("Pavyko prisijungti!")
else:
    print(f"Kažkas negerai.. Statuso kodas {r.status_code}")
```

```
Kažkas negerai.. Statuso kodas 404
```

prieš tai nagrinėtas pavyzdys gali būti supaprastintas .ok atributo pagalba. Jeigu mus tenkina visi kodai, mažesni už 400, galime tiesiog:

In [38]:

```
r = requests.get("http://www.google.lt/asdsad")

if r.ok:
    print("Pavyko prisijungti!")
else:
    print(f"Kažkas negerai.. Statuso kodas {r.status_code}")
```

```
Kažkas negerai.. Statuso kodas 404
```

ok grąžina True arba False reikšmę, todėl gali būti lengvai panaudotas su if.

headers

.headers - atsakymo į užklausą antraštės, jose būna įvairūs serverio, užklausos parametrai ir pan. Jeigu įdomu, kokie headers duomenys ką reiškia, galite pasiskaityti čia [https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers (https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers)](https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers)

In [39]:

```
r = requests.get("http://www.google.lt")
print(r.headers)
```

```
{'Date': 'Thu, 11 Aug 2022 08:45:02 GMT', 'Expires': '-1', 'Cache-Control':
'private, max-age=0', 'Content-Type': 'text/html; charset=windows-1257', 'Co
ntent-Encoding': 'gzip', 'Server': 'gws', 'Content-Length': '6390', 'X-XSS-P
rotection': '0', 'X-Frame-Options': 'SAMEORIGIN', 'Set-Cookie': 'AEC=AakniGO
8NFIfTeuwnCYDOqZpVrVgYYLxtoqZVmuDLcl9VgCsK0H_4nnasA; expires=Tue, 07-Feb-202
3 08:45:02 GMT; path=/; domain=.google.lt; Secure; HttpOnly; SameSite=lax'}
```

In [35]:

```
print(r.headers['Date'])
```

```
Thu, 11 Aug 2022 08:40:38 GMT
```

In [43]:

```
print(r.headers['Content-Type'])
```

```
text/html; charset=windows-1257
```

In [41]:

```
for key, val in r.headers.items():
    print(f"{key} =====> {val}")
```

```
Date =====> Thu, 11 Aug 2022 08:45:02 GMT
Expires =====> -1
Cache-Control =====> private, max-age=0
Content-Type =====> text/html; charset=windows-1257
Content-Encoding =====> gzip
Server =====> gws
Content-Length =====> 6390
X-XSS-Protection =====> 0
X-Frame-Options =====> SAMEORIGIN
Set-Cookie =====> AEC=AakniGO8NFIfTeuwnCYDOqZpVrVgYYLxtoqZVmuDLcl9VgCsK0H_4n
nasA; expires=Tue, 07-Feb-2023 08:45:02 GMT; path=/; domain=.google.lt; Secu
re; HttpOnly; SameSite=lax
```

url

url vadinamas adresas kokiu mes kreipiamės į resursą internete.

In [12]:

```
print(r.url)
```

```
http://www.google.lt/ (http://www.google.lt/)
```

url parametrai

patikslinti kreipimasi į serverį galima panaudojus url parametrus. Pabandžius google paiešką, pamatysime, kad google puslapis suformuoja url su visa eile parametrų, kuriuos atstovauja poros raktas=reikšmė, jungiamos &.

In [14]:

```
"""https://www.google.lt/search?
q=IESKAU2
&
source=hp
&
ei=zcL0&iflsig
&
ved=0ahUKEwji4s6L-xAgQQ4dUDCAc
&
uact=5&oq=IESKAU
&
gs_lcp=Cgdnd3
&
sclient=gws-wiz"""
```

Out[14]:

```
'https://www.google.lt/search?\nq=IESKAU2\n&\nsource=hp\n&\nei=zcL0&iflsig\n
&\nved=0ahUKEwji4s6L-xAgQQ4dUDCAc\n&\nuact=5&oq=IESKAU\n&\ngs_lcp=Cgdnd3\n&
\nsclient=gws-wiz'
```

requests leidžia mums patiems formuoti url parametrus žodyne ir paduoti į url.

In [15]:

```
payload = {'q': 'Perlas'}

r = requests.get("http://www.google.lt/search", params=payload)
print(r.url)
```

http://www.google.lt/search?q=Perlas (http://www.google.lt/search?q=Perlas)

In [ ]:

```
r.text
```

bandom paiešką python.org puslapyje, išsiaiškiname, kokie parametrai formuojami serverio, kaip paieškos žodis ir paieškos rezultatų puslapis, bei pabandome juos nusikopijuoti į savo parametrų žodyną.

In [ ]:

```
r = requests.get("https://www.python.org/search/?q=pep")
print(r.text)
```

In [ ]:

```
"""https://www.python.org/search/?q=pep&page=2"""
```

nukopijuojam rastus parametrus į savo žodyną payload, ir paduodam į užklausos params. Taip patogiau ir tiksliau.

In [50]:

```
payload = {'q': 'pep', 'page': '2'}
r = requests.get("https://www.python.org/search/", params=payload)
print(r.url)
```

```
https://www.python.org/search/?q=pep&page=2 (https://www.python.org/search/?q=pep&page=2)
```

In [ ]:

```
print(r.text)
```

Kiti HTTP metodai

iki šiol dirbome su .get() metodu, kuris yra bene dažniausiai naudojamas. Tačiau yra ir kiti, tokie kaip post, put, delete, patch. Labai geras resursas jų visų ištestavimui yra www.httpbin.org (http://www.httpbin.org) . pvz.:

į tokią užklausą serveris atsakys informacija su mūsų IP adresu JSON formate.

In [ ]:

```
r = requests.get("http://httpbin.org/ip")
print(r.text)
```

In [ ]:

```
Panagrinėkime, kaip dirbti su post metodu:
```

In [56]:

```
data = {'name': 'Jonas', 'lastname': 'Jonaitis', 'birthday': '1999.09.09' }
r = requests.post('http://httpbin.org/post', data=data)
print(r.text)
```


```
{
  "args": {},
  "data": "",
  "files": {},
  "form": {
    "birthday": "1999.09.09",
    "lastname": "Jonaitis",
    "name": "Jonas"
  },
  "headers": {
    "Accept": "*/*",
    "Accept-Encoding": "gzip, deflate",
    "Content-Length": "48",
    "Content-Type": "application/x-www-form-urlencoded",
    "Host": "httpbin.org",
    "User-Agent": "python-requests/2.28.1",
    "X-Amzn-Trace-Id": "Root=1-62f4d7ca-02426a6821f76c5a19882e7b"
  },
  "json": null,
```


google į tokią užklausą atsakytų The request method `POST` is inappropriate for the URL. Serveris turi palaikyti tokias užklausas, jos naudojamos siųsti savo informaciją. `GET` naudojama gauti informaciją.

In [ ]:

```
data = {'name': 'Jonas', 'lastname': 'Jonaitis', 'birthday': '1999.09.09' }
r = requests.post('http://google.lt/', data=data)
print(r.text)
```

In [ ]:

```

```