

Apêndice 1 - Notebook webscraping BDTD

Script para buscar resumos das teses e dissertações na BDTD

```
[27]: import requests
from bs4 import BeautifulSoup as bs
import pandas as pd
import json

[28]: # Definindo configurações globais de proxy para realizar a extração dentro da
      ↳ rede Petrobras
chave = 'XXXX'
pwd = 'XXXXXXXXX'
proxy_url = 'http://' + chave + ':' + pwd + '@inet-sys.gnet.petrobras.com.br:804/'
proxies = {
    'http' : proxy_url ,
    'https' : proxy_url ,
}

[29]: areas = pd.read_csv('Areas_CNPQ.csv', sep=';')

[30]: #função para coletar link para cada tese de uma determinada áreas do
      ↳ conhecimento do CNPQ

def link_area(area, page):
    #Separar nome e sobrenome do autor
    nivel_1 = area[0]
    nivel_2 = area[1]

    #Preparar nome e sobrenome para a url
    nivel_1 = '+' .join(nivel_1.split())

    if str(nivel_2) != 'nan':
        nivel_2 = '+' .join(nivel_2.split())

    #preparar a url
    url = ('http://bdtb.ibict.br/vufind/Search/Results?filter%5B%5D=dc.
    ↳subject.cnpq.fl_str_mv%3A"CNPQ%3A%3A' +
          nivel_1 +
          '%3A%3A' +
          nivel_2 +
          '"&page=' +
```

```

        str(page))
    else:
        url = ('http://bdtd.ibict.br/vufind/Search/Results?filter%5B%5D=dc.
→subject.cnpq.fl_str_mv%3A"CNPQ%3A%3A' +
            nivel_1 +
            '&page=' +
            str(page))

    #Fazer requisição e parsear o arquivo html
    f = requests.get(url, proxies = proxies).text
    soup = bs(f, "html.parser")

    #Coletando link para as teses
    links = []
    for doc in soup.find_all('a', href=True):
        if 'title' in doc.get('class', []):
            links.append(doc['href'])
    return links

```

[31]: *#Coletar o link com as teses das áreas não relacionadas à Petrobras*
 area_oposta = areas[areas['PETROBRAS'] == 0]

```

n_pages = 10 # Cada página retorna 20 teses
links_area_oposta = []

for area in area_oposta.iterrows():
    for p in range(n_pages):
        link = link_area(area[1], p)
        if link != []:
            links_area_oposta = links_area_oposta + link
        else:
            break

```

[32]: *#Coletar o link com as teses das áreas não relacionadas à Petrobras*
 mesma_area = areas[areas['PETROBRAS'] == 1]

```

n_pages = 10 # Cada página retorna 20 teses
links_mesma_area = []

for area in mesma_area.iterrows():
    for p in range(n_pages):
        link = link_area(area[1], p)
        if link != []:
            links_mesma_area = links_mesma_area + link
        else:
            break

```

```

[33]: # Função para coletar os metadados de uma tese ou dissertação da BDTD dado uma URL
def tese_link(url):
    #definir url
    url = 'http://bdtd.ibict.br' + link

    #Requisitar html e fazer o parser
    f = requests.get(url, proxies = proxies).text
    soup = bs(f, "html.parser")

    #Dicionário para armazenar as informações da tese
    tese = {}

    #Adicionar título
    tese['Title'] = soup.find('h3').get_text()
    for doc in soup.find_all('tr'):

        #Identificar atributo
        try:
            atributo = doc.find('th').get_text()
        except:
            pass

        #Verificar se o atributo possui mais de um dado
        for row in doc.find_all('td'):


            #Adicionar o atributo no dicionário
            if row.find('div') == None:
                try:
                    tese[atributo] = doc.find('td').get_text()
                except:
                    pass
            else:
                element = []

                #No dicionário, adicionar todos os dados ao seu respectivo atributo
                for e in doc.find_all('div'):
                    try:
                        element.append(e.find('a').get_text())
                    except:
                        pass
                tese[atributo] = element


    return(tese)

```

```
[38]: teses = {}
n = 0
for link in links_area_oposta:
    url = 'http://bdtd.ibict.br'+link
    teses[url] = tese_link(link)
    n = n + 1
    if n % 500 == 0:
        print(n)
        with open('teses_areas_opostas_Large.json', 'w') as fp:
            json.dump(teses, fp)
```

500 1000 1500 2000 2500 3000 3500 4000 4500 5000 5500 6000 
↪6500

```
[39]: teses = {}
n = 0
for link in links_mesma_area:
    url = 'http://bdtd.ibict.br'+link
    teses[url] = tese_link(link)
    n = n + 1
    if n % 500 == 0:
        print(n)
        with open('teses_mesmas_areas_Large.json', 'w') as fp:
            json.dump(teses, fp)
```

500 1000 1500 2000 2500 3000 3500 4000 4500 5000 5500 6000 
↪6500