Таллер де ботс

Йоутбис (Joutbis)- Viquitrobada 2017

Índex

- Instal·lació de les eines
- Permisos i autenticacions
- Funcions bàsiques de pywikibot
- Estructura bàsica d'un bot
- Més funcions de pywikibot
- Cerques amb API
- Expressions regulars
- Wikidata

Per què un bot?

- Un bot és un programa que fa edicions de forma automàtica.
- Permet fer tasques massives.
  - Arreglar ortografia
  - Posar un peu de pàgina comú
  - Canviar paràmetres de plantilla
  - ...
- No cal saber programar
  - Però si en saps pots fer més coses.

Instal·lació de les eines

- **python** – llenguatge de programació
- **pip** – llibreries addicionals per python
- **pywikibot** – llibreria per interactuar amb el món wiki

Instal·lació de les eines

- **python** i **pip**
  - Si tens Linux, sempre hi són.
  - Per Windows: https://www.python.org/downloads/
    - Recomano 2.7 (MSI installer)
    - **Important**: fer upgrade de **pip**
    - `python -m pip install -U pip`

Instal·lació de les eines

- **pywikibot**
  - Ajuda d'instal·lació
  - Versions:
    - compat (obsoleta)
    - core
      - estable: http://tools.wmflabs.org/pywikibot/core.zip
      - desenvolupament (master):

        Git: `git clone --recursive https://gerrit.wikimedia.org/r/pywikibot/core.git`

        - `git pull https://gerrit.wikimedia.org/r/pywikibot/core master`

        URL: `https://github.com/wikimedia/pywikibot-core/`

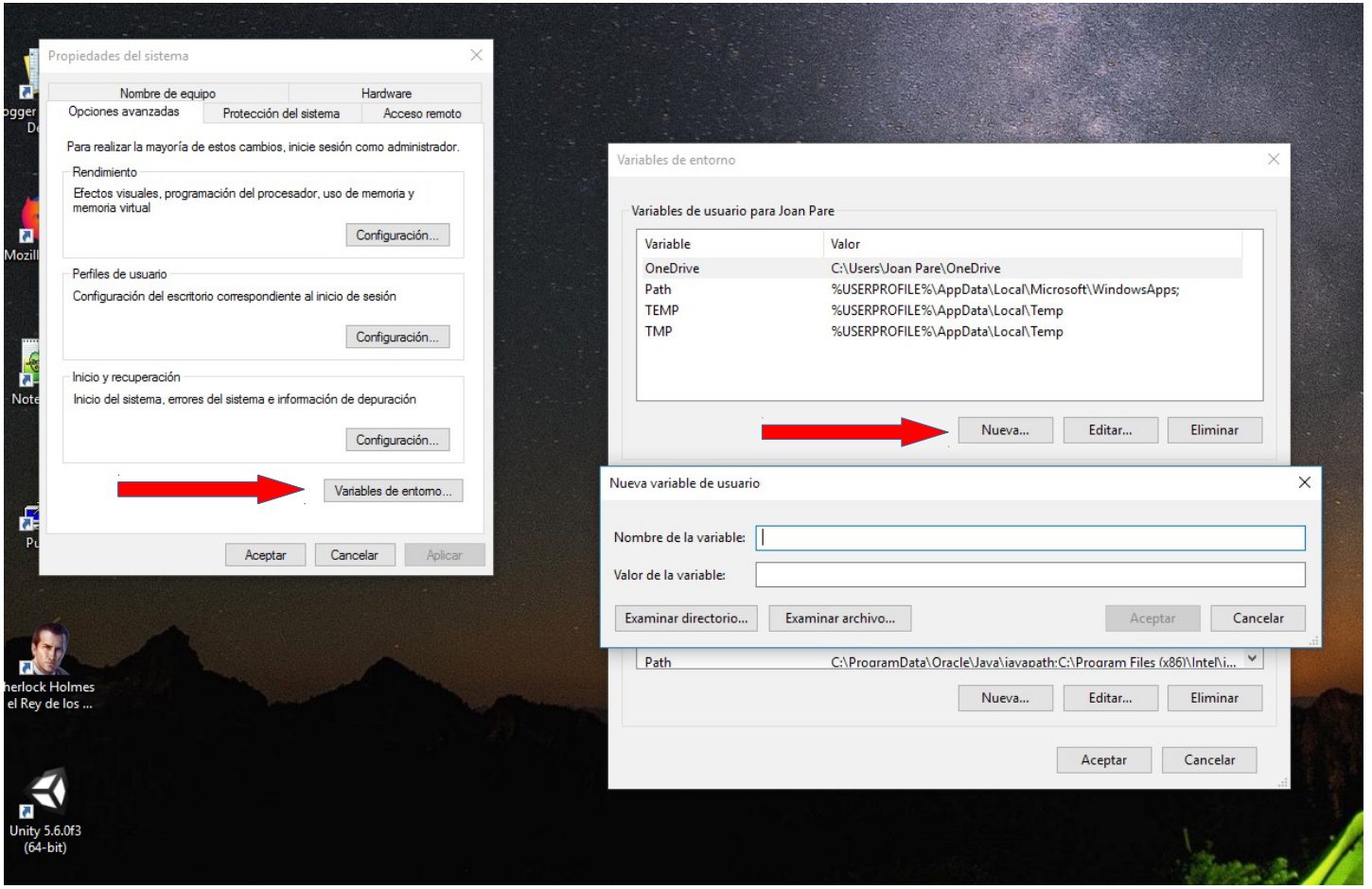
        - Ajuda Aquí

Instal·lació de les eines

- **Tasques posteriors**
  - Per Windows, actualitzar el PIP:
    - https://pip.pypa.io/en/stable/installing/
    - `python -m pip install -U pip`
  - Alguns mòduls necessaris:
    - `sudo pip install requests requests-oauthlib`
    - `sudo pip install httplib2`
    - d'altres, segons el que utilitzis.
  - PYTHONPATH
    - A .bashrc
    - `PYTHONPATH=$PYTHONPATH:~/.pywikibot_master/core_old`
    - `export PYTHONPATH`

Instal·lació de les eines

- **PYTHONPATH en Windows**
  - Explorador d'arxius, el meu ordinador, botó dret, Propietats



Permisos i autenticacions

- **Permisos**
  - https://www.mediawiki.org/wiki/Manual:Pywikibot/OAuth/Wikimedia
  - https://meta.wikimedia.org/wiki/Special:OAuthConsumerRegistration/propose
  - Entrar amb l'usuari del bot
  - Omplir el formulari
    - Retorna quatre tirallongues hexadecimals, que s'han d'omplir a user-config.py

Permisos i autenticacions

user-config.py

```
# -*- coding: utf-8  -*-

mylang = 'ca'
family = 'wikipedia'
usernames['wikipedia']['ca'] =          \
  usernames['meta']['*'] =          \
  usernames['commons']['*'] =           \
  usernames['wikidata']['*'] =          \
  usernames['wiktionary']['*'] =        \
  usernames['wikibooks']['*'] =         \
  usernames['wikinews']['*'] =          \
  usernames['wikiquote']['*'] =         \
  usernames['wikisource']['*'] =        \
  usernames['wikiversity']['*'] =       \
  usernames['wikivoyage']['*'] =        \
  'NomDelBot'
authenticate['*.wikipedia.org'] =       \
  authenticate['*.wikimedia.org'] =     \
  authenticate['*.wikidata.org'] =      \
  authenticate['*.wiktionary.org'] =        \
  authenticate['*.wikibooks.org'] =     \
  authenticate['*.wikinews.org'] =      \
  authenticate['*.wikiquote.org'] =     \
  authenticate['*.wikisource.org'] =        \
  authenticate['*.wikiversity.org'] =       \
  authenticate['*.wikivoyage.org'] =    \
  authenticate['*.mediawiki.org'] =     \
  ('0123456789abcdef0123456789abcdef','0123456789abcdef0123456789abcdef489d6feb',
'0123456789abcdef0123456789abcdef', '0123456789abcdef0123456789abcdefb64ecb78')
```

Funcions bàsiques

- **python pwb.py <funció.py>**

  Llista de scripts

  - Exemple:

    - `python pwb.py category.py listify`
    - `python pwb.py replace.py -ns:0 arrivar arribar -search:arrivar`
    - `python pwb.py solve_disambiguation.py  "Flamenc"`

Funcions bàsiques

- `siteca = pywikibot.Site('ca')`
  - **Seleccionar en quina wiki treballarem**
- `pag = pywikibot.Page(siteca,article)`
  - **Seleccionar una pàgina**
- `text = pag.get()`
  - **Llegir el wikitext**
- `pag.put(text,comentari,minor)`
  - **Gravar el text a la pàgina**

Estructura d'un bot

```
# -*- coding: utf-8 -*-
import pywikibot
import re
import sys
import time
import codecs

def main():

 fitxer = codecs.open('entrada','r',encoding='utf-8')
 siteca = pywikibot.Site('ca')

 for pagina in fitxer:
   article=pagina.rstrip()

   pag = pywikibot.Page(siteca,article)
   min = 1
   while True:
    try:
      txtorig = pag.get()                  # llegim wikitext
      break
    except pywikibot.IsRedirectPage:       # si és una redirecció
      pag     = pag.getRedirectTarget()    # mirem cap a quin article
      txtorig = pag.get()                  # llegim wikitext
      article = pag.title()                # actualitzem títol
      break
    except pywikibot.NoPage:
      exit(-1)
    except pywikibot.data.api.APIError:    # de vegades passa
      time.sleep(min*60)
      min = min+1
```

Estructura d'un bot

```
    txtnou = manipular_text(txtorig)

    if txtorig != txtnou:
      try:
         pywikibot.showDiff(txtorig,txtnou)
         comentari = u"Proves del taller de bots"
         pag.put(txtnou,comment=comentari,minorEdit=False)

      except KeyboardInterrupt:
         return

if __name__ == '__main__':
     main()
```

Provem-ho!

Estructura d'un bot

```
def preguntar():
  while 1:
   sys.stdout.write("Canviar? (s/n/t) ")
   a=sys.stdin.readline()
   a=a.rstrip()
   if a=='t':
       return 0
   if a=='s':
       return 1
   if a=='n':
       return -1


if txtorig != txtnou:
   try:
     pywikibot.showDiff(txtorig,txtnou)
     comentari = u"Proves del taller de bots"
     if preguntem:                        ######## NOU  ############
       quefem = preguntar()
       if quefem == 0:
          preguntem = False

     if quefem >= 0:
       pag.put(txtnou,comment=comentari,minorEdit=False)

   except KeyboardInterrupt:
     return
```

Què més es pot fer

- `type(variable)` – Diu el tipus de la variable
- `dir (variable)` – Llista els mètodes disponibles
- `PrettyPrinter` – Que quedi bonic!

```
pag = pywikibot.Page(siteca,article)
print type(pag)
print dir(pag)
print "----------------------------------------"
pp = pprint.PrettyPrinter(indent=4)
pp.pprint(dir(pag))
```

- https://doc.wikimedia.org/pywikibot/api_ref/pywikibot.html

Cerques amb API

https://www.mediawiki.org/wiki/API:Query
https://www.mediawiki.org/w/api.php?action=help&modules=query
https://www.mediawiki.org/w/api.php?action=help&modules=query%2Bsearch

```
def get_search(searchstr):
    params = {
         'action'          :'query',
         'list'            :'search',
         'srsearch'        :searchstr,
         'srwhat'          :'text',
         'srlimit'         : 40
        }

    sr=pywikibot.data.api.Request(**params).submit()
    return sr

def main():
    cerca = get_search(u"bacallà")
    for article_info in cerca[u'query'][u'search']:
        print u"Títol de l'article:",article_info[u'title']
```

Cerques amb API

https://www.mediawiki.org/wiki/API:Query
https://www.mediawiki.org/w/api.php?action=help&modules=query
https://www.mediawiki.org/w/api.php?action=help&modules=query%2Bembeddedin

```
def get_search(searchstr):    # Troba articles que tenen una plantilla
    params = {
        'action'          :'query',
        'list'            :'embeddedin',
        'eititle'         :'Plantilla:'+searchstr,
        'eifilterredir'   :'nonredirects',
        'eilimit'         : 500,
        'continue'        :''
        }

    l=[]
    lastContinue = {'continue': ''}
    while True:
       pr = params.copy()
       pr.update(lastContinue)
       sr=pywikibot.data.api.Request(**pr).submit()
       if 'error' in sr:
          raise Error(sr['error'])
       if 'warnings' in sr:
          raise Error(sr['warnings'])
       for i in sr['query']['embeddedin']:
           l.append(i['title'])
       if 'continue' not in sr:
          break
       lastContinue = sr['continue']
    return l
```

Cerques amb API

https://www.mediawiki.org/wiki/API:Query
https://www.mediawiki.org/w/api.php?action=help&modules=query
https://www.mediawiki.org/w/api.php?action=help&modules=query%2Bembeddedin

```
def main():
    if len(sys.argv)!=2:
       print u"Ús: python trobar_plantilla.py <nom-de-la-plantilla>".encode("utf-
8")
    else:
       l_masies = get_search(sys.argv[1])
       for p in l_masies:
          print p.encode("utf-8")
main()
```

Expressions regulars

- **Autòmata per trobar textos arbitraris**
- **Amb asteriscos, per entendre‘ns**
- **Un exemple:**

```
r"<[Rr][Ee][Ff]\s+[Nn][Aa][Mm][Ee](.*?)<\/[Rr][Ee][Ff]>"
```

- **http://regex101.com**

Expressions regulars

Expressió	Explicació	Exemple
`.`	Un punt equival a qualsevol caràcter	`pep`
`a+`	Un + indica una o més repeticions	`aaa` `abc`
`a?`	Un ? Indica 0 o 1 repeticions	`aa` `<res>`
`a*`	Un ? Indica 0 o més repeticions	`aaa` `<res>`
`[a-z]`	Els claudàtors indiquen un rang	`duh`
`[a-z]+`	Es poden combinar els anteriors	`hola u`
`[^>=]+`	^ indica la negació del conjunt	`abcdef>`
`a|b`	O l'un, o l'altre	`Hola` `Va bé`

Expressions regulars

Expressió	Explicació	Exemple
\s	Espai en blanc o tabulador	
\S	Qualsevol cosa que no sigui espai en blanc	;
\d	Un dígit	2
\D	Qualsevol cosa que no sigui un dígit	=
\w	Un caràcter que pugui ser en un paraula	è
\W	El contrari de l'anterior: separadors	;
()	Recordar alguna part per més endavant	abcdef>
^$	Inici (^) i final ($)	

Expressions regulars

- **? com a fre**
  - L‘asterisc agafa tot el que pot (greedy)
  - L‘interrogant el frena (lazy)
  - No té res a veure amb el significat anterior (0 o 1)
  - “Tot el que trobis fins el tancament d‘un claudàtor“
    - `.*\]` `# agafa fins l‘últim claudàtor`
    - `.*?\]` `# s‘atura al primer claudàtor`

Expressions regulars

- **Funcions python d'expressions regulars**
  - https://docs.python.org/2/library/re.html
  - re.search(), re.match() i re.sub()
  - Vigileu amb els flags
    - re.MULTILINE, re.DOTALL, re.UNICODE
  - MatchObject
    - group(n)
    - start(n)
    - end(n)

Expressions regulars

```
# -*- coding: utf-8 -*-
import re

exp_reg = r"<[Rr][Ee][Ff]\s+[Nn][Aa][Mm][Ee]\s*=\s*\"?(.*?)\"?\s*>(.*?)<\/[Rr][Ee]
[Ff]>"
text = u'ara ve la referència<Ref name = "diari">{{ref-web|
url=http://www.diari.com| títol=El diari}}</ref>'

mo = re.search(exp_reg,text)        # cerquem una referència
if mo != None:
    print u"No hem trobat res"
else:
    print u"Referència trobada!"
    print dir(mo)                           # es poden consultar moltes coses
    print mo.group(0)                       # tot el que ha concordat
    print mo.group(1)                       # el primer parèntesi
    print mo.group(2)                       # el segon parèntesi

print re.sub(exp_reg,r"\nnomref: \1;\ncontingut: \2",text)
print re.findall(exp_reg,text)
```

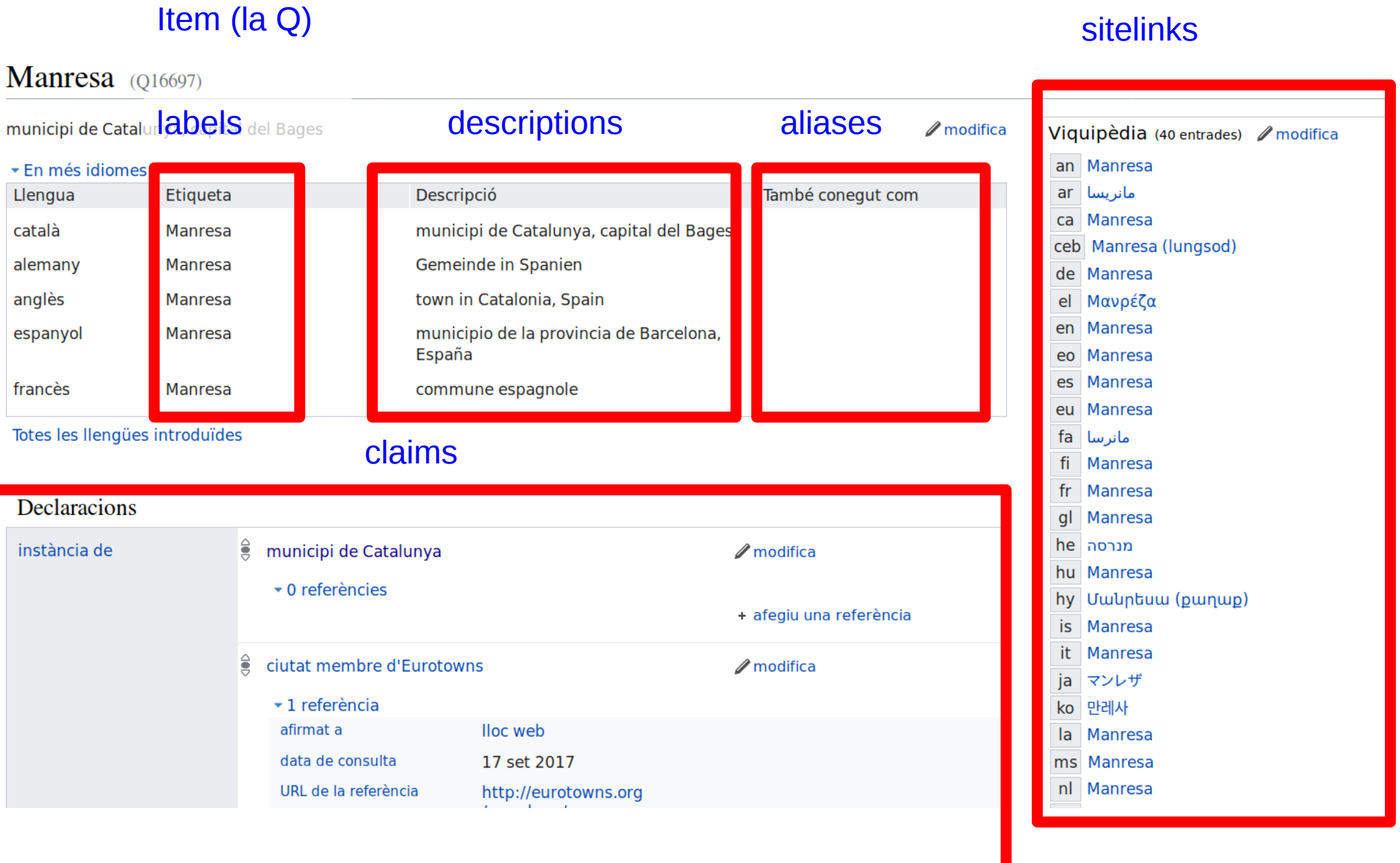
Wikidata

- **Contingut molt estructurat**
  - python ho veu com un diccionari
  - Un diccionari on els elements són llistes
  - Llistes que contenen altres diccionaris ...
  - Tipus de dades propis (data, quantitat, ...)
- **El tractament és molt diferent**
  - Orientat a bots, però té moltes variants.
- **Exploració per nivells, cal saber com moure‘s**

Wikidata

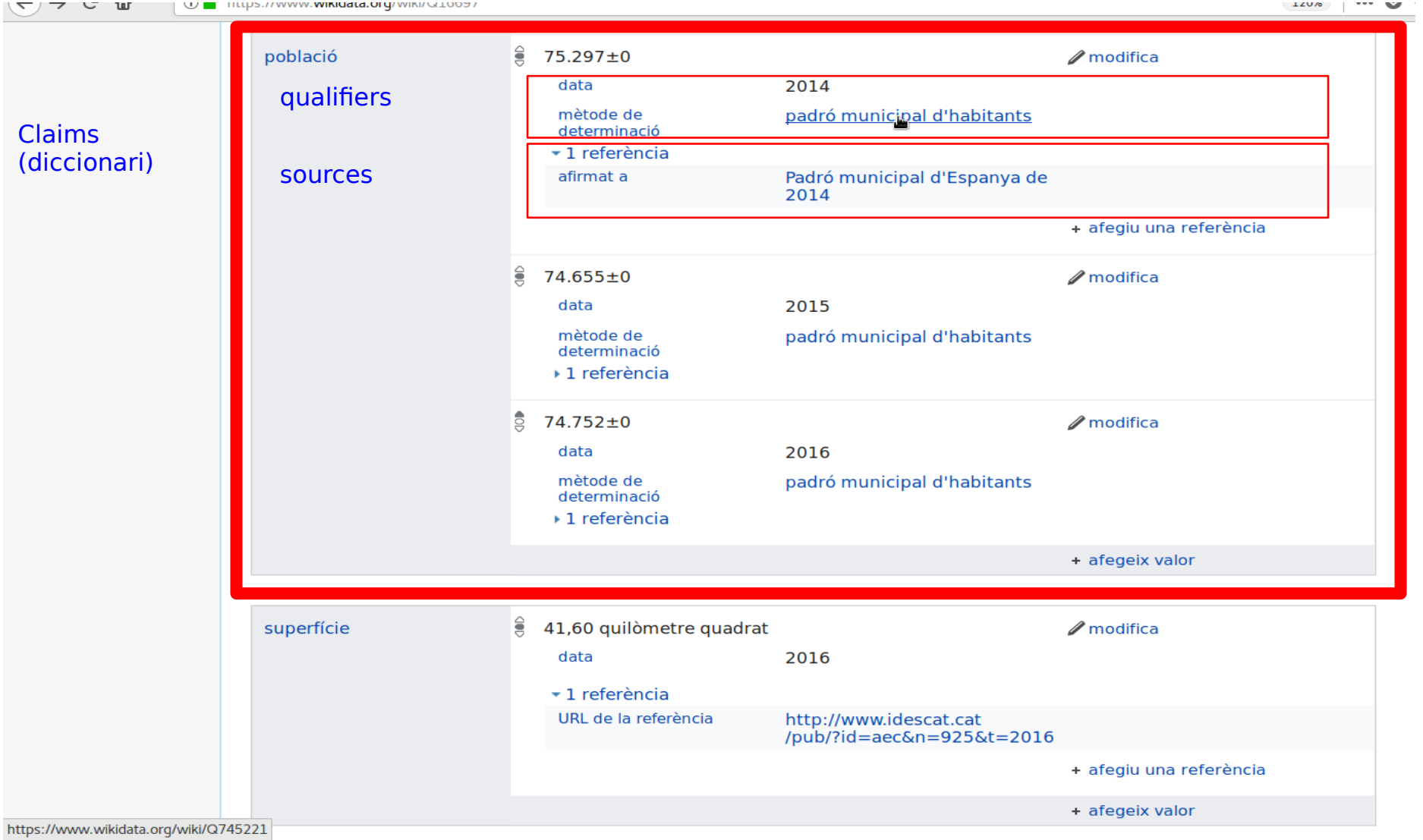
- **Sandbox per proves**
  - https://www.wikidata.org/wiki/Q4115189
- **Abans de fer massives, demanar permís**
  - https://www.wikidata.org/wiki/Wikidata:Requests_for_permissions/Bot
- **El centre i l'origen són les Q's (item)**
  - D'aquí s'accedeix a tota la resta

Wikidata- Nomenclatura



Wikidata - Nomenclatura

- Els atributs (qualifiers) i les referències (sources) també són claims



Wikidata – trobar la Q

- **Primer, el site wikidata**
  - `wdsite = pywikibot.Site("wikidata","wikidata")`
- **A partir del número de Q**
  - `item = pywikibot.ItemPage(wdsite,"Q16697")`
- **A partir d'una pàgina de la wiki**
  - `casite = pywikibot.Site("ca")`
  - `pagina = pywikibot.Page(casite,u"Manresa")`
  - `item = pywikibot.ItemPage.fromPage(pagina)`
- **O a partir de generadors i cerques**
  - **Ho veurem després**

Wikidata – obtenir info

- **Referència (sense dades ni unitats)**
- **Exemples (amb unitats)**
- **Exemple senzill**

```
item_dict = item.get()      # obligatori, fer-ho sempre
# llegeix el diccionari amb tota la info de l'item
# llavors ja podem accedir-hi

item.labels          equival a   item_dict['labels']
item.claims          equival a   item_dict['claims']
```

Wikidata – navegació

- **El claim és un diccionari**
  - **Hi busquem si hi ha la clau (P) que ens cal**
  - **Si hi és, a sota hi ha una llista (que potser és d'un sol membre!)**

```
if 'P1082' in item.claims:
    llista = item.claims['P1082']
    print type(llista)       # llista de Claims

    for pob in llista:
        print type(pob)       # és un Claim
        tg = pob.getTarget()  # obligatori
        print type(tg)        # és una WbQuantity
        print tg.amount        # la DADA
```

Wikidata – atributs

- **Els atributs (Qualifiers) també són claims**
  - **Per tant, són un altre diccionari**
  - **Caldrà tornar a fer getTarget() al nivell apropiat**

```
if 'P585' in pob.qualifiers:
    ldata = pob.qualifiers['P585']
    print type(ldata)        # llista de Claims

    for dates in ldata:
        print type(dates)                # és un Claim
        data_info = dates.getTarget() # obligatori
        print type(data_info)            # és una WbTime
        print data_info.year            # la DADA
```

Wikidata – fonts

- **Les fonts (Sources) són una llista de claims**
  - **No cal saber-s'ho. Vas aprofondint, de mica en mica**

```
for ref in pob.sources:    # sources és una llista de
                           # diccionaris
  try:
    afirmat = ref['P248']
  except KeyError:
    print u'No hi ha "afirmat a" a les referències'
    continue
  print type(afirmat)      # és una llista
  for elt in afirmat:
    print type(elt)        # ara ja és un Claim
    tg = elt.getTarget()
    print type(tg)          # a l'exemple, és un Item
                            # podem tornar al principi!
```

Wikidata – unitats

- **Ara mirem la superfície**

```
if 'P2046' in item.claims:
    llista = item.claims['P2046']
    for superf in llista:
        print type(superf)      # ja és un claim
        tg = superf.getTarget()
        print type(tg)           # això és una quantitat
        print dir(tg)            # a veure què té
        print tg.amount
        print tg.unit
        unit_item = tg.get_unit_item()
        unit_item.get()
        print unit_item.labels['ca']
```

Wikidata – Modificacions

- **Construir el Claim (amb Qualif. i sources)**
- **Penjar-lo del lloc on correspongui**

```
wdsite = pywikibot.Site("wikidata", "wikidata")
repowd = wdsite.data_repository()

item = pywikibot.ItemPage(wdsite,"Q4115189") # sandbox

item_dict = item.get()

if 'P2046' in item.claims:      # superfície
    clm_pop = item.claims['P2046']
    for clm in clm_pop:
      item.removeClaims(clm)     # esborrem primer
```

Wikidata – Modificacions

- **Claim i source**

```
superficie = 77
nouclm = pywikibot.Claim(repowd,'P2046')
      # crem un claim de superfície
unitat_item = pywikibot.ItemPage(repowd,"Q712226")
      # km2
nouclm.setTarget(pywikibot.WbQuantity(amount=superficie,site
=repowd,unit=unitat_item))

urldelaref=pywikibot.Claim(repowd,'P854')
  # URL de la referència
urldelaref.setTarget('http://www.idescat.cat/pub/?
id=aec&n=203')
lrefs = [urldelaref]    # creem una llista amb la ref
```

Wikidata – Modificacions

- **Qualifiers**

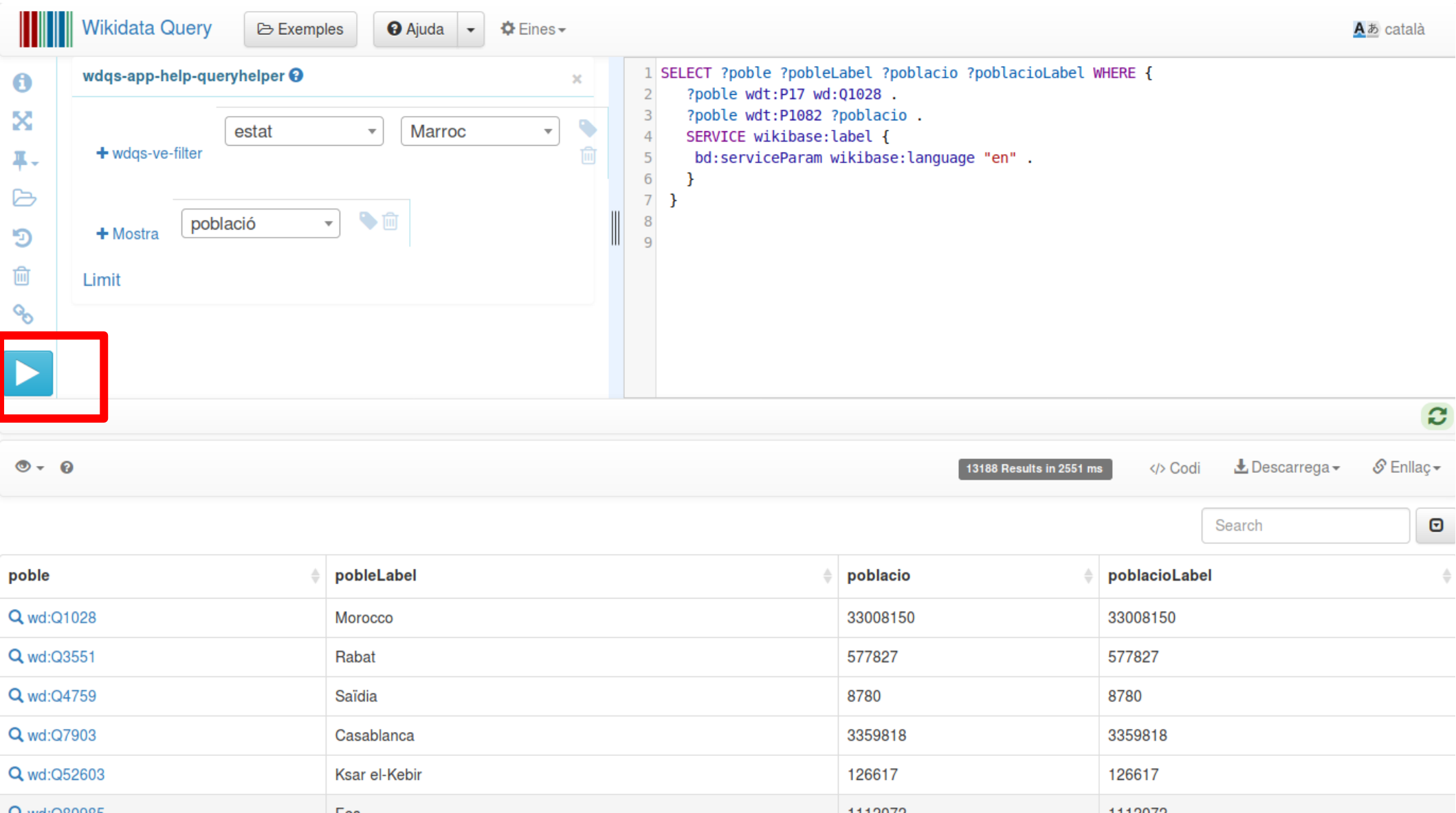
```
qualif = pywikibot.Claim(repowd,'P459')
qualif.setTarget(pywikibot.ItemPage(repowd, u"Q745221"))
            #padró


item.addClaim(nouclm)          # grava a Wikidata
nouclm.addQualifier(qualif)   # afegeix atribut
nouclm.addSources(lrefs)      # afegeix la referència
```

Wikidata Query Service

- **Interfície SPARQL amb Wikidata**
  - **https://query.wikidata.org**
- **Es pot preguntar de tot, però és complicat**
  - **https://www.wikidata.org/wiki/Wikidata:SPARQL_query_service/Wikidata_Query_Help**
  - **https://www.wikidata.org/wiki/Wikidata:SPARQL_tutorial**
- **Es pot integrar amb el bot**

Wikidata Query Service



poble	pobleLabel	poblacio	poblacioLabel
wd:Q1028	Morocco	33008150	33008150
wd:Q3551	Rabat	577827	577827
wd:Q4759	Saïdia	8780	8780
wd:Q7903	Casablanca	3359818	3359818
wd:Q52603	Ksar el-Kebir	126617	126617

Wikidata – Query amb bot

- **Municipis d'Àraba**

```
from pywikibot import pagegenerators as pg
 ...
QUERY='SELECT ?poble ?pobleLabel WHERE { ?poble wdt:P131
wd:Q81801 . ?poble wdt:P31 wd:Q2074737 .  SERVICE
wikibase:label { bd:serviceParam wikibase:language "ca" .  }
}'
# Q81801 Àraba
# Q2074737 municipi d'Espanya

generador =
pg.WikidataSPARQLPageGenerator(QUERY,item_name='poble',site=
wdsite)
if generador!=None:
  for item in generador:
       item.get()
```

Wikidata – Query amb bot

- **També, amb un fitxer separat per la query**
  - **Exemple, entitats del Marroc amb població informada**

Fitxer marroc.rq

```
SELECT ?poble ?pobleLabel ?poblacio ?poblacioLabel WHERE {
   ?poble wdt:P17 wd:Q1028 .
   ?poble wdt:P1082 ?poblacio .
   SERVICE wikibase:label {
    bd:serviceParam wikibase:language "en" .
   }
 }
# Items on estat és Marroc i tenen informada població
```

Wikidata – Query amb bot

Després, al programa principal ...

```
from pywikibot import pagegenerators as pg
…

with open('marroc.rq','r') as fit_query:
   QUERY = fit_query.read().replace('\n','')

generador =
pg.WikidataSPARQLPageGenerator(QUERY,item_name='poble',site=
wdsite)
if generador!=None:
  for item in generador:
       item.get()
```