How to interact with Wikidata via Pywikibot

[[User:Mike Peel]]

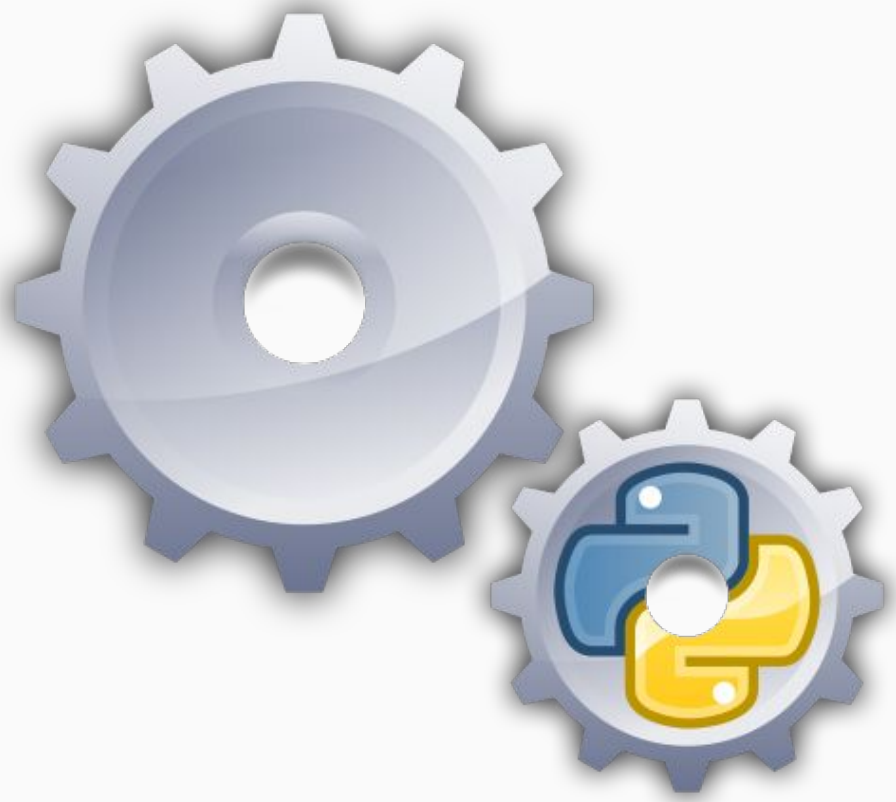
7 October 2022

Figuring out what to edit

Writing bot code

Bot approval

Running the bot



https://www.mediawiki.org/wiki/Manual:Pywikibot

Pywikibot

Python framework to edit MediaWiki content

‘pip install pywikibot’

or git pull https://github.com/wikimedia/pywikibot.git

Works with python2 & 3

Sorry, examples here are python2! change "print var" to "print(var)".

What can be done?

Create new items

Add new property statements

Edit existing property statement

Edit qualifiers, references

Delete/undelete a page

Search, query

Scrape websites, compare to databases

... and lots more!

Automated edits need to be 100% accurate (or >99.9%)

Semi-automated can be less accurate - but check changes before saving!

Focus on routine tasks that are easily described

Keep things modular/simple as much as possible

Look at code for existing bots - e.g., https://meta.wikimedia.org/wiki/User:Pi_bot

Code	Description
wir_newpages.py	Add sitelinks, or create new items, for humans with new Wikipedia entries
wir_newpages_duplicity.py wir_newpages_category.py wir_newpages_special.py	Add sitelinks, or create new items, for humans with new Wikipedia entries (backlog)
wikidata_p301_inverse.py wikidata_p910_inverse.py wikidata_p1753_inverse.py wikidata_p1754_inverse.py	Add reciprocal values for items that have inverse statements
wikidata_newshipname.py wikidata_viewof.py	Create new category items to link via category items
wikidata_newitem.py	Create new Wikidata entries for people with Commons categories
wikidata_new_from_wikipedia_query_article.py wikidata_new_from_wikipedia_query_category.py	Create new items from Wikipedias
wikidata_import_labels_from_commons.py	Add English labels from commons category names
wikidata_import_infobox_qid.py commons_wikidata_infobox_tidy.py	Wikidata Infobox tidying
wikidata_en_biography_names.py	Copy labels for humans from other languages to English
wikidata_bot_requests.py	Archive Wikidata bot requests
wikidata_bad_p373.py	Remove bad P373 values
whs_infobox.py	Migration of WHS Wikidata infoboxes
simplewp_commonscat_fix.py	Commons link maintenance
commonscat_p910_tidy.py	Items with P910 and a sitelink in both the topic and category items, where one redirects to the other
commonscat_move_from_P1754.py	Move commons sitelinks from list to category items
commonscat_move_from_P910.py	Move commons sitelinks from topic to category items
commonscat_copy_from_P373.py	Copy commons sitelinks from P373
commonscat_check.py	Checking for sitelinks to Commons category redirects. Mismatch between P373 and sitelink.
Commons, enwp	Import coordinates from Commons and Wikipedias
commons_wikidata_search.py	Search for new Commons sitelinks to add to Wikidata
commons_wikidata_infobox.py	Wikidata Infobox deployment
commons_qi_by_user.py commons_qi_by_user.py commons_qi_by_user.py	List of Commons QI, FI and VI by user
commons_migrate_ids_to_wikidata.py	Trim duplicate external IDs that are provided by Wikidata
commons_import_ids_to_wikidata.py	Import IDs from Commons to Wikidata
commons_defaultsort_conflicts.py	Maintain commons infoboxes by checking for default sort conflicts
commons_date_find.py	Regularly add Commons category links for date categories/items
commons_check_id.py	Commons sitelinks from ID matching
cochrane.py	Cochrane updates
cochrane_fr.py	Cochrane updates
check_tgwiki.py	Remove broken sitelinks to deleted pages
Bitbucket	Copy human names between languages and dialects
Bitbucket	Remove brackets from human name labels

Set up the pywikibot user configuration file

Need a file with this name in your code directory!

Initial definitions

user-config.py

```
# -*- coding: utf-8  -*-
from __future__ import unicode_literals

# Default mediawiki family
family = 'wikipedia'

# Default mediawiki language
mylang = 'pt'

# Username for each wiki
usernames['wikipedia']['pt'] = u'Mike Peel'
usernames['wikidata']['wikidata'] = u'Mike Peel'
```

MediaWiki family

Language

Usernames

Starting the code and connecting to ptwiki and Wikidata

Initial definitions

Import modules

Connect!

example.py

```
1  #!/usr/bin/env python
2  # -*- coding: utf-8 -*-
3  # Example pywikibot code description here
4  # Mike Peel     17-Sep-2018      v1 - start
5
6  # Import modules
7  import pywikibot
8  from pywikibot import pagegenerators
9  from pywikibot.data import api
10 import numpy as np
11 import requests
12
13 # You may need to enforce the use of utf-8
14 import sys
15 reload(sys)
16 sys.setdefaultencoding('UTF8')
17
18 # Connect to ptwiki
19 ptwiki = pywikibot.Site('pt', 'wikipedia')
20 # and then to wikidata
21 ptwiki_repo = ptwiki.data_repository()
```

Documentation!

Python2: use utf-8

Python3: remove lines 14-16!

Printing the QID, label, sitelink and a claim from Wikidata

Defines the function

Fetch the Wikidata item

Print a label

Print a sitelink

Print a claim

Print the QID

```
def printwikidata(wd_item):
    qid = wd_item.title()
    print qid

    item_dict = wd_item.get()

    try:
        print 'Name: ' + item_dict['labels']['en']
    except:
        print 'No English label!'
    try:
        print 'ptwiki article: ' + item_dict['sitelinks']['ptwiki']
    except:
        print 'No Portuguese article!'
    try:
        print item_dict['claims']['P31']
    except:
        print 'No P31'
```

The output from the previous slide

```
python example.py
```

```
Q511405
Name: Sky Polarization Observatory
No Portuguese article!
[Claim.fromJSON(DataSite("wikidata", "wikidata"), {u'type': u'statement', u'mainsnak': {u'datatype
': u'wikibase-item', u'datavalue': {u'type': u'wikibase-entityid', u'value': {u'entity-type': u'it
em', u'numeric-id': 184356}}, u'property': u'P31', u'snaktype': u'value'}, u'id': u'Q511405$2228cb
b0-4fab-4319-9890-4b2d5764dc27', u'rank': u'normal'}), Claim.fromJSON(DataSite("wikidata", "wikida
ta"), {u'type': u'statement', u'mainsnak': {u'datatype': u'wikibase-item', u'datavalue': {u'type':
 u'wikibase-entityid', u'value': {u'entity-type': u'item', u'numeric-id': 33093130}}, u'property':
 u'P31', u'snaktype': u'value'}, u'id': u'Q511405$b40c9ccf-4e9f-fe0b-cfe1-d983199084f2', u'rank':
u'normal'})]
```

Printing the property values correctly

Loop over claims

Get the claim target

```
43      try:
44          for claim in item_dict['claims']['P31']:
45              p31_value = claim.getTarget()
46              p31_item_dict = p31_value.get()
47              print 'P31 value: ' + p31_value.title()
48              print 'P31 label: ' + p31_item_dict['labels']['en']
49      except:
50          print "That didn't work!"
51      return 0
```

Get the target contents

The output from the previous slides

```
Q511405
Name: Sky Polarization Observatory
No Portuguese article!
[Claim.fromJSON(DataSite("wikidata", "wikidata"), {u'type': u'statement', u'mainsnak': {u'datatype
': u'wikibase-item', u'datavalue': {u'type': u'wikibase-entityid', u'value': {u'entity-type': u'it
em', u'numeric-id': 184356}}, u'property': u'P31', u'snaktype': u'value'}, u'id': u'Q511405$2228cb
b0-4fab-4319-9890-4b2d5764dc27', u'rank': u'normal'}), Claim.fromJSON(DataSite("wikidata", "wikida
ta"), {u'type': u'statement', u'mainsnak': {u'datatype': u'wikibase-item', u'datavalue': {u'type':
 u'wikibase-entityid', u'value': {u'entity-type': u'item', u'numeric-id': 33093130}}, u'property':
 u'P31', u'snaktype': u'value'}, u'id': u'Q511405$b40c9ccf-4e9f-fe0b-cfe1-d983199084f2', u'rank':
u'normal'})]
P31 value: Q184356
P31 label: radio telescope
P31 value: Q33093130
P31 label: cosmic microwave background experiment
```

How to generate lists of pages to edit - using a sparql query

Sparql query

```
69  sparql = "SELECT ?item WHERE { ?item wdt:P31 wd:Q184356 } LIMIT 10"
70  generator = pagegenerators.WikidataSPARQLPageGenerator(sparql, site=ptwiki_repo)
71  for page in generator:
72      printwikidata(page)
```

Loop over results

Run the query

BTW: can also get Wikidata item from Wikipedia page:

```
wd_item = pywikibot.ItemPage.fromPage(category)
item_dict = wd_item.get()
```

How to generate lists of pages to edit - using categories and template uses

```
 99  targetcat = 'Categoria:Telescópios'
100  cat = pywikibot.Category(ptwiki, targetcat)
101  subcats = pagegenerators.SubCategoriesPageGenerator(cat, recurse=False);
102  for subcat in subcats:
103      print subcat.title()
104
105  pages = pagegenerators.CategorizedPageGenerator(cat, recurse=False);
106  for page in pages:
107      print page.title()
108
109  template = pywikibot.Page(ptwiki, 'Predefinição:Info/Telescópio')
110  targets = template.embeddedin()
111  for target in targets:
112      print target.title()
113
114  targets = pagegenerators.RandomPageGenerator(total=10, site=ptwiki, namespaces='14')
115  for target in targets:
116      print target.title()
```

Search for existing Wikidata items

```
109 # From https://gist.github.com/ettorerizza/7eaebbd731781b6007d9bdd9ddd22713
110 def search_entities(site, itemtitle):
111     params = { 'action' :'wbsearchentities',
112                'format' : 'json',
113                'language' : 'en',
114                'type' : 'item',
115                'search': itemtitle}
116     request = api.Request(site=site, parameters=params)
117     return request.submit()
118
119 wikidataEntries = search_entities(ptwiki_repo, "Neuromat")
120 if wikidataEntries['search'] != []:
121     results = wikidataEntries['search']
122     numresults = len(results)
123     for i in range(0,numresults):
124         qid = results[i]['id']
125         label = results[i]['label']
126         print qid + " - " + label
```

Doing a test edit to Wikidata - adding ‘instance of’ = ‘sandbox’

Get the item

Get the target item

QID and property ID to edit

```
21  def editwikidata(wd_item, propertyid, value):
22      qid = wd_item.title()
23      print qid
24      item_dict = wd_item.get()
25
26      claim_target = pywikibot.ItemPage(ptwiki_repo, value)
27      newclaim = pywikibot.Claim(ptwiki_repo, propertyid)
28      newclaim.setTarget(claim_target)
29      print newclaim
30      wd_item.addClaim(newclaim, summary=u'Adding test claim')
31
32      return 0
33
34  testqid = 'Q4115189' # Wikidata sandbox
35  testproperty = 'P31' # instance of
36  testvalue = 'Q3938'  # Sandbox
37  wd_item = pywikibot.ItemPage(ptwiki_repo, testqid)
38  print editwikidata(wd_item, testproperty, testvalue)
```

Create a claim

Set claim target

Save the claim

Manually checking the edit before saving it

```
print newclaim
text = raw_input("Save? ")
if text == 'y':
    wd_item.addClaim(newclaim, summary=u'Adding test claim')
```

```
Q4115189
Claim.fromJSON(DataSite("wikidata", "wikidata"), {u'type': u'statement', u'mainsnak': {u'datatype'
: u'wikibase-item', u'datavalue': {u'type': u'wikibase-entityid', u'value': {u'entity-type': u'ite
m', u'numeric-id': 3938}}, u'property': 'P31', u'snaktype': u'value'}, u'rank': u'normal'})
Save?
```

Creating a new Wikidata entry

wikidata_newitem.py

```python
#!/usr/bin/python
# -*- coding: utf-8  -*-
# Create new Wikidata items
# Started 25 August 2018 by Mike Peel
from __future__ import unicode_literals

import pywikibot
import numpy as np
import time
import string
from pywikibot import pagegenerators
import urllib

commons = pywikibot.Site('commons', 'commons')
repo = commons.data_repository()  # this is a DataSite object

def newitem(category, items):
    new_item = pywikibot.ItemPage(repo)
    new_item.editLabels(labels={"en":category}, summary="Creating item")
    candidate_item = pywikibot.ItemPage(repo, new_item.getID())
    candidate_item = pywikibot.ItemPage(repo, new_item)
    print candidate_item

    data = {'sitelinks': [{'site': 'commonswiki', 'title': category}]}
    candidate_item.editEntity(data, summary=u'Add commons sitelink')

    for item in items:
        claim = pywikibot.Claim(repo, item[0])
        claim.setTarget(pywikibot.ItemPage(repo, item[1]))
        try:
            candidate_item.addClaim(claim, summary=u'Setting '+item[0]+' value')
        except:
            print "That didn't work"
    return

category = 'Category:1546 in Finland'
items = [['P31','Q4167836'],['P971','Q6570'],['P971','Q33']]
test = newitem(category, items)
```

Scrape a website

```
89  def parsesite(url):
90      try:
91          r = requests.get(url)
92          websitetext = r.text
93      except:
94          print 'Problem fetching page!'
95          return 0
96      # print websitetext
97      split = websitetext.split("<h1 style='display:none'>")
98      i = 0
99      for item in split:
100         i+=1
101         # Skip the top part
102         if i > 2:
103             # print item
104             print 'Title: ' + item.split('</h1>')[0].strip() + '\n'
105             print 'Museum: ' + item.split("strong>Museu:</strong><span itemprop='publisher'>")[1].split("</span>"
                    )[0].strip() + "\n"
106     return 0
107
108 parsesite('http://www.museusdoestado.rj.gov.br/sisgam/index.php?pagina=1&operador=or&busca=a%20b%20c%20d%20e%20f
        %20g%20h%20i%20j%20k%20l%20m%20n%20o%20p%20q%20r%20s%20t%20u%20v%20w%20x%20y%20z&museu=todos&qresultados=40')
```

Submitting a bot request

Wikidata:Requests for permissions/Bot/Pi bot

< Wikidata:Requests for permissions | Bot

*The following discussion is closed. **Please do not modify it.** Subsequent comments should be made in a new section. A summary of the conclusions reached follows.*

Approved--Ymblanter (talk) 18:53, 1 March 2018 (UTC)

Pi bot [edit]

Pi bot (talk • contribs • new items • SUL • Block log • User rights log • User rights)
Operator: Mike Peel (talk • contribs • logs)

Task/s: Fix mismatches between Commons category (P373) and the commons sitelink that are due to one linking to a category redirect.

Code: https://bitbucket.org/mikepeel/wikicode/src/master/commonscat_check.py

Function details: Query for cases where Commons category (P373) is not the same as the commons sitelink (but both are present). If one of them points to a commons category that has commons:Template:Category redirect in it, and that redirect points to a category with the same name as the other value on Wikidata, then update the value to avoid the redirect. Run daily to catch new cases caused by category moves on Commons. Example edits: [1] [2] (for the two different cases). --Mike Peel (talk) 14:36, 22 February 2018 (UTC)

Looks good, pls run 50 test edits.--Ymblanter (talk) 08:37, 25 February 2018 (UTC)

@Ymblanter: Done, see [3]. The test run showed up a few issues, so I've modified the code to cope with them. If there are multiple P373 values present, then the code skips that entry. If one of the two commons links leads to a missing page, then it counts them and doesn't make an edit at the moment (although I've drafted some code that will also handle this case, which I'll enable once I find some test cases). And if Wikidata reports an interwiki conflict, then the code logs that and moves on (I don't know if there's a better way to check for this in pywikibot?). Thanks. Mike Peel (talk) 13:36, 25 February 2018 (UTC)

Thanks. I will approve the bot in a couple of days provided there have been no objections raised.--Ymblanter (talk) 14:26, 25 February 2018 (UTC)

*The above discussion is preserved as an archive. **Please do not modify it.** Subsequent comments should be made in a new section.*

Category: Archived requests for permissions

Separate bot account

Code online somewhere

Do a test run

(normally lots more discussion here!)

Short task description

Full details

Approval

Cron jobs to automatically run the bot on a set schedule

```
pi@raspberrypi3:~ $ crontab -e
```

```
  GNU nano 2.7.4                        File: /tmp/crontab.JaMTUg/crontab

SHELL=/bin/bash

TERM=xterm
PYTHONIOENCODING=UTF-8
LANG=en_US.UTF-8
LC_ALL=en_US.UTF-8
# /etc/crontab: system-wide crontab

# m h dom mon dow user  command
0 8 * * * /home/pi/Documents/wikicode/pibot_daily.sh
0 9 1 * * /home/pi/Documents/wikicode/pibot_monthly.sh
```

pibot_daily.sh

```
1  #!/bin/bash
2  source /home/pi/.profile
3  cd /home/pi/Documents/wikicode/
4  /usr/bin/python permissions.py
5  /usr/bin/python guardian_obit.py
6  /usr/bin/python nyt_obit.py
7  /usr/bin/python commons_defaultsort_conflicts.py
```

Useful links

- https://github.com/mpeel/wikicode/ - pi bot + semi-automated scripts
- https://meta.wikimedia.org/wiki/User:Pi_bot - list of scripts + descriptions
- https://github.com/mpeel/wikicode/blob/master/example.py

- https://www.wikidata.org/wiki/Wikidata:Bot_requests - requests for bots to do things

- https://doc.wikimedia.org/pywikibot/master/api_ref/pywikibot.html
- https://phabricator.wikimedia.org/T276329 - see 'Tasks' for some intro tasks
- (WMF participates in Outreachy + Google Summer of Code each year!)

- https://www.google.com/ - good place to look for example code ;-)
- https://stackoverflow.com/ - where you'll actually find example code

Questions?

- Feel free to ask me on my talk page later!

  https://www.wikidata.org/wiki/User_talk:Mike_Peel
- Feel free to copy-paste-modify from Pi bot code!
- What kind of tasks are you thinking about doing?