List management bot

[[User:Strainu]]
We had a problem...

The secret flow of Wiki Loves Monuments infographic by Elke Wetzig (Elya) CC BY SA 3.0 via Wikimedia Commons
...I needed another solution

- no database

- **modular** - due to the large number of pages, I had to be able to parse different chunks independently;

- **allow caching** - parsing 60,000+ images takes about 2 days; the results need to be heavily reused

- **be aware of local content** - Infoboxes and other templates can contain a great deal of information

- **allow external data** imports with as little preparation as possible
No database?

- Yep, only text files: json and CSV
- OS-agnostic, no 3rd party software, no flame wars...
Modularity

- Due to the large number of pages (almost 100K), I had to be able to parse different namespaces and websites independently
- I have one script for each task, the main ones glued together with a shell script
- Secondary scripts (like the one used to create articles) are independent
- Further selection of the target can be done using the command line parameters and/or config files
Caching

- Parsing 60,000 images takes about 2 days; the results need to be reused as much as possible.

- I use three different parsing modes:
  - Quick: don't parse any pages already in the database.
  - Normal: parse only files changed since they were added to the database.
  - Full: parse all the pages.

- Output can be written incrementally (good for bad internet connections).
Local content

• Bad news: there are lots of different templates, each with their own fields.
• Good news: most are redirects ;)
• Images in articles must be good, right?
• Don't forget about coordinates!
• What else?
External data

- As simple as a CSV with headers, but!
- **THE HEADERS HAVE TO MATCH**

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Cod</strong></td>
<td><strong>Creatori</strong></td>
</tr>
</tbody>
</table>
Other lists? Really?

Nope, not really :)
Oh, come on...

- You need to configure each new database; this is not trivial, but takes way less than for erfgoedbot
  - I'm planning to move all the configuration in a single place
- Parsing is generic – you can build the database and perhaps extract some statistics
- Updating the lists is still WIP – a duck-taped version works for RAN (the list of archeological sites in Romania)
  - I don't have any other list and no real incentive to work on this :(
Show me the code!

https://github.com/rowiki/wikiro/tree/master/robots/python/pywikipedia/monumente

- **monumente.sh** - a shell script that can do most of the work by running the scripts below
- **update_database.py** - parse the monument lists and extract the data
- **parse_monument_article.py** - parse the articles and images of monuments and log them
- **corroborate_monument_data.py** - parse all the databases, log errors and warnings and updates the database where possible.

- **stats.py** - generates some statistics in wikitext format.
- **add_template_to_images.py** - adds the correct template to images used in the list but which don't have the template
- **error_remove.py** - removes a set of errors from the lists using regular expressions.
- **create_shortcuts.py** - create pages in the "Cod" namespace that redirect to the list entries
- **create_articles.py** - a highly customized script that creates articles about LMI monuments.
- **cleanup_code.py** - remove spaces and potentially other detectable errors from the codes
- **json2txt.py** - Convert the list of files parsed to a text file
Q&A?