Title: Pywikibot : Compat to Core Migration
Student: Priyanka Jayaswal
Abstract:

One of the widely used tools of Mediawiki tools is Pywikibot, which may prove to be the most powerful developing device if implemented judiciously. It is based on python which makes it easier for contribution. The project I'm proposing is to resolve the issue of **Core to Compat Migration**. Tracked in Bugzilla - 55880. The projects goal is to:

- ➢ Fasten the migration process as much as possible. -
- ➢ Provide more functionalities in core -

Besides the goal discussed, I propose to make an easy to use documentation on pywikibot by:

- ❏ Completing incomplete documentation in Wikibooks.

- ➢ http://en.wikibooks.org/wiki/Pywikibot/Scripts
- ➢ http://en.wikibooks.org/wiki/Pywikibot/Scripts/Other_scripts

- ❏ Preparing a complete guide on pywikibot usage containing proper compilation of already existing resources and creating new references for unavailable data.

- Mentors: Amir Sarabadani, John Vandenberg (for porting patrol.py only)

User Information:

Name: Priyanka Jayaswal
Email Address: priyankajayaswaliitkgp@gmail.com / priyankajayaswal025@gmail.com Freenode IRC Nick: prijaya
Location (City, Country and/or Time Zone): Kharagpur (GMT +05:30 hours)
Proposal Title: Pywikibot: Compat to Core Migration

Motivation for Proposal / Goal:

Bots are a major workhorse of Mediawiki projects. They play an important role in performing thousands of edits across all projects, which would otherwise prove to be difficult and mundane. It is this set of powerful features that motivated me to try my hand at it. The aim of my project is to encourage use of pywikibot (which is more-user friendly, hence likely to attract more active contributors) for its diverse use and application at creating versatile bots. While working on pywikibot for applying in FOSS OPW Round 9 Project on Pywikibots, I realised that migration is an essential step needed to start further contribution towards development of new features. The potential of pywikibots can be realized fully after migrations are implemented. Hence, owing to my interest to work further towards contributing in this interesting subject, I was motivated to apply for it.

Going on an elaborate discussion on the objectives and goals of my project, I'll proceed further by giving a brief discussion on the versions of the present "pywikibot".

- **Core:** This is the newer version(2) which is referred as the *rewrite.*
- **Compat:** This is the older version(1) which is referred as the *trunk*.

The migration from Version 1 "wikibot" to Version 2 "pywikibot" has been done due to following points:

- ➔ Version 2 is an attempt to lessen the complexity in the existing version by avoiding the packaging issues. The newer version is contained as a standard package in itself, and other required/dependent modules are contained within it. This is easy to follow, and intuitive for creating Bots, while earlier had to import various other libraries (e.g., wikipedia), and customize them accordingly making the implementation complicated.
- ➔ Other tree folders have been reduced by correct categorization making the structure less tangible.
- ➔ "Version 2 framework *only* supports wikis using MediaWiki v.1.14 or higher software." - which are modified time to time making it bug free and more feature integrated with each release, hence more appealing for users." -- > In other words, Version 2 is not compatible with versions earlier of MediaWiki 1.14. This way we leverage the stable, less buggy, and frequently updated releases.

Project Goal 1:

- ❑ Speeding up the migration process as much as possible.

For Core to Compat migration, we first need to fix the bugs tracked in Bugzilla - 55880[1] first. This would help us cover the initial issues which involves the basic barrier for complete migration. This would then speed up the process of migration.

The basic approach needed to fix these dependencies is to start by replacing the earlier model by class implementation, keeping all the functionalities intact and fixing the existing issue that comes up with each step. With this, I need to parallely update the documentations as well.

This approach I have derived from my experiencing at trying to port us-scripts.py from pywikibot-compat/scripts to pywikibot-core. You may have a look at my progress[2]. But with the kind of file being ported the approach modifies accordingly. During the micro-task, I learnt certain important lessons as I overcame the issues.

- ➔ Initially, I started by getting a brief idea of the code dependency on other scripts. Since nothing as such was found, I went for direct migration and committed it on gerrit [2].
- ➔ From the reviews I received in the process, I was asked to implement object-oriented approach of programming (owing to the fact that Version 1 is shifting to it in Version 2 as already discussed above).

- ➔ Then I added class StatesRedirectBot, specific to the issue of the script on redirecting of pages. I studied the functions well so that I don't miss them and restructured the code with bot implementation.
- ➔ I even realised that sometimes seemingly correct code might not be sufficient for merging; there may be options of including more functionalities in it. With more experience, we may understand that. These should be covered in later part when we have tried much on the codes and become completely familiar with the entire structure. [As happened in the case, I worked on.]
- ➔ I became familiar with basic yet essential norms in coding in open-source programming.

So, I have a basic idea upon how the approach for solving the bugs should be through this and it is easy to overcome difficulties due to cooperative community actions.

[1] : https://bugzilla.wikimedia.org/showdependencytree.cgi?id=55880&hide_resolved=1
[2] : https://gerrit.wikimedia.org/r/#/c/167036/

Report on approach (bug specific) is given in timeline.

- ❏ Provide more functionalities in core.

➢ Extra : Since I have worked on Mediawiki and used its extensions like Translate, Visual Editor and others during my initial attempt at getting familiar with Wikimedia projects [3], I may even introduce new functionality by writing code for the translate extension of mediawiki to help translators or translation managers. As discussed with my mentors, I may even work on preparing a structural draft on few of the features where pywikibot has not been properly supported like Flow, Abuse Filter, Liquid Threads, Translate, New Pages Feed, Checkuser, ULS, VisualEditor/Parsoid, Flagged Revs, if the time permits.

[3] : https://gerrit.wikimedia.org/r/#/c/166990/

Project Goal 2:

- ❏ Completing documentation in Wikibooks.

As I was going through the documentation, I came across the following links:

- http://en.wikibooks.org/wiki/Pywikibot/Scripts
- http://en.wikibooks.org/wiki/Pywikibot/Scripts/Other_scripts

In these, the files presently there in pywikibot-core has been discussed properly. But in most of the cases, the pages are not existent. Since, this resource might be really helpful for newbies to start with pywikibot, I propose to update the pages and add in the most recent data for user reference.

For this purpose, I have prepared a sample page for the script I've been working on i.e., us-states.py. This is a sample documentation on how the missing pages would be updated. It is troublesome when one comes across an incomplete information, especially for newbie. As such I believe, updating these pages is important. I hope that this modification would be really beneficial for further development of pywikibot as well.
Additional Task: Beside this, I plan at preparing a better compilation of all the resources available on pywikibot. Presently, these are not arranged in a user-friendly format. I propose to prepare a Complete Manual Page, which would include all the important informations, links and recent advancements in pywikibot at one place. This guide on pywikibot usage would also include new references for unavailable data.

Implementation Details:

The execution of project involves these three steps:

I. Bug fixation
II. Repository contents Migration Analysis.
III. Documentation for proposed pages parallely while coding.

I. **Bug Fixation (Approach to follow)--**

Earlier attempts of total migration of compat to core shows a lot of bugs which need to be fixed with utmost priority. For the purpose, I have divided the bugs on common grounds as stated below to make the work easier:

- Bugs with status : PATCH_TO_REVIEW : For such bugs we need to add new functionalities (extra), if possible To resolve them we need some detailed study on the kind of bug, its possible dependencies, features, goals and objectives. These may be helpful in deriving conclusion about what functionality need to be added.
- Status - New: These bugs need to be solved with a testing approach in mind. [Since, I had worked on a similar kind of bug, I would try to tackle it by bot class implementation approach. But since nothing is known in advance, there is no defined approach.]
- The complicated ones: These bugs are the ones which are dependent on either fixing other bugs, or including necessary functionalities in an existing script without which it is not working.

In the timeline, I have discussed each such bug specifying the approach to resolving it according to the above categorization.

II. Repository contents Migration Analysis. --

Once the bugs are fixed, we will become familiar with the working of most of the scripts and many issues of migration would already be solved completely such as:

➢ logs folder
➢ copyright

- ➢ pywikibot (to great extent)
- ➢ tests
- ➢ watchlists

Other folders like wiktionary, userinterfaces, commondelimiters, family appear to be mere normal porting files with outdated information.
maintenance would take another good amount of time. All details how it would be carried is discussed in the timeline with perfectly marked outline.

III. **Documentation for proposed pages parallely while coding.**

Since, my work would involve a lot of documentation, I have planned to keep them distributed non-uniformly during the time. I plan to use this time as buffer time during which I will complete any incomplete work from past(if any parallely) and have distributed it nicely over the timeline. Normal documentation would most probably go parallely with bug fixing. I have planned to use this time to balance the amount of work according to my convenience.

Please go through the detailed description on how I would carry my work as under in the tentative timeline.

The Tentative Timeline:

The project demands from me a full time commitment, not intending to do anything else during the period(besides college activities). I'm highly interested in working efficiently on the project, in order to avoid as much as possible inaccuracies in the timeline. I have distributed the work non-uniformly, in order to relax for some periods, such as the exam time period at my college. I have distributed the proposed work as under:

As per my timeline, I would like to mention that I'll be working on documenting my progress with each bug fixed in the form of a progress report with simultaneous updation in the official documentation. I have tried to re-distribute my work such that besides the already alloted time for documentation particularly I may devote sufficient time for each issue as I understand the importance of a proper documentation.

Day 1 - Day 7: As said earlier, I plan to start with solving the bugs already reported.
I shall create a spreadsheet to be used for the 'Review Phase' which is later discussed and would make sure it is updated with time. Besides, I plan to give my initial days at solving bugs of similar nature parallely like:
Bug 64159 : Port tag_nowcommons.py to core
Bug 68503 : replacementfile for replace.py
These are the ones, which demand direct migration. I will make the necessary changes to employ bot implementation using OOPs approach and work on the bugs.
Bug 72206 : Port patrol.py to core -> I wish to solve this early for getting an idea upon how patrolling is to be done (This may give me an idea on how I may proceed with my work on the

additional task I have proposed. Accordingly, I may include it in my timeline as well. -> this is regarding translate work which I didn't commit to complete but just mentioned that I may work on, if time permits)
Bug 69980 : patrol.py depends on mwlib.uparser not available on wmflabs].
Bug 64878 : Port parserfunctioncount.py to core

Day 8 - Day 14 : I'll work on second kind of bugs i.e., those which arise due to lack of functionality. Such examples are work on pagegenerators
Bug 55007: missing some pagegenerators in core branch / 3 remaining are to be made),
Bug 55881 : Missing possibility to retrieve images from a page that were not included through templates → implementing linkedPage function
Bug 55882 : The order of parameters is lost when using extract_templates_and_params() and extract_templates_and_params_regex( ) -> which may solve the dependent
Bug 57097 : Port warnfile.py to core branch on warnfile.py -- replacement in core.
Bug 64880 : Port speedy_delete.py to core -> new. (This issue need not my involvement but if possible I may look into the procedure how it was fixed. It will be helpful to understand issues of such type).

Day 15 - Day 21 : I may devote some time to fix if any bug is not resolved till then. Besides, I may work on :
Bug 64877: Add cross-wiki Special:Import support. -> I guess it is easy to fix.
Bug 57995 :Add function for list=watchlistraw -> dependent bug
Bug 68988 : watchlist.py should use CachedRequest , (watchlist folder migration to be completed with this from compat to core).

As discussed with my mentor, I would include a 'review phase' after one month of work since it is important to prioritize the work based on the the basic knowledge gained by working in the first month. This may be done by compiling data about each script in a spreadsheet format with detailed information on:

- ➔ proper description on "all" of the scripts in compat on what it aims at achieving.
- ➔ details on any similarity with an existing script (if any). [this would be needed to decide if merging can also hold to be an option. Also, it will be needed so as to keep track such that no duplication may occur.
- ➔ present status if someone is working on the script keeping track of the most recent update. (this may be done by adding evidence of recent use --for this I've been suggested to use the git/svn history to identify who contributed the script, and email them to ask if they know where the script is currently being used.).
- ➔ it would include a column for 'porting priority' - scripts which are no longer used would have a low porting priority.

With these informations, I may make my work more organized. The mentors would review it providing extra information or suggesting someone to mail about a certain script which isn't

ported. The spreadsheet would continue to be improved over the period of the project, and in the end it will clearly describe what scripts are still high priority porting tasks.

This would be a continuous process.

Day 22 - Day 35 : Working on bugs with external bug fixing issues eg. :
Bug 64839 : Port maintainer.py et al to core and move to separate package -> I guess that it would take time. We also need to determine its possible use and then act accordingly.
As already discussed above, about how to treat a patch to review kind of bug. [Once, I get familiar on how to proceed in such cases I may swap this week's work with the week which needs more priority] Presently I assume that these bugs need minor amendments in functionality.
Bug 55882 : The order of parameters is lost when using extract_templates_and_params() and extract_templates_and_params_regex() as most probably we may have to prepare some "complicated structures" being referred in the bug report,

Day 36 - Day 42 : Due to my exams, I may focus more on documentation work as proposed earlier at spare time. Besides, I shall start my work on manual page preparation during this course.

Day 43 - Day 49 : I shall continue my work on bugs of kind PATCH_TO_REVIEW
Bug 64835 : Add block functionality in pywikibot (apispec.py) -> new),
Bug 64838 :(Port catimages.py to core -> needs much time as I need to find why it may not be converted into core and has come under other scripts).
Bug 64845 : Merge commons_category_redirect.py from compat into category_redirect.py -> needs time as once the bug was stated Won't fix once.
Then I shall start with fixing bugs of non-complicated history. like :
Bug 64876 : Port overcat_simple_filter.py to core

Day 50 - Day 56 : Keeping with work distribution, I plan to fix here minor issues so that I may get sufficient buffer time for any previous task that is left(if any). -> (new hence, I guess easy to fix)
Bug 64875 : Port ndashredir.py to core
Bug 64871 : Port match_images.py and move to scripts repository
Bug 64870 : Port inline_images.py to core
This might also act as the second 'Review Phase' where description about the scripts not covered during first time may be added.

Day 57 - Day 63 : I shall try to fix other bugs and would prefer going for detailed study on the other migrating folders. Since core/ tests is more advanced in compat/tests, here I need to figure out if any test is left for migration and possibly add other new test cases which may be needed.
Bug 64856 : Port various Commons upload bots to core & move to separate repository -> this may add many new scripts for manipulation of images as suggested.
Bug 64854 : Port followlive.py to core -> new, demands just normal porting.

Bug 64853 : Port djvutext.py to core -> Limitations on Proofread is mentioned with other details and an already proposed patch. Might need some time.
Bug 67724 : commands.log missing in core -> Working on creating log collector for any bot. With this I may complete log folder migration from pywikibot/compat to core.

Day 64 - Day 70 : Would devote this week for updating any incomplete task and work on documentation of scripts indicated in Project Goal 2, since, till then I'll be familiar with most of the scripts. Besides, I shall work on some other minor bugs as well.
Bug 64848 : Port and re-package copyright*.py -> Checking data about the working status of the scripts and then proceed further. Appears to be a minor porting, after this, I shall solve the issue with compat/copyright migration to core.
Bug 64849 : Port deledpimage.py to core -> By taking reference from the Chinese script this may be solved on those lines as inferred from bug report.

Day 71 - Day 77 : I wish to complete all tasks on bug fixing and start my work on individual folder specific migration since now I would be familiar with all the scripts and may work accordingly. With this, I shall start with my work on commondelimiters in compat.

Day 78 - Day 84 : I would work on migration of maintenance and pywikibot migration based on detailed verification of data in code to avoid duplication. The time needed might vary depending on bugs solved earlier.

Day 85 - Day 90 : All other things like updation of data in wiktionary, families and userinterfaces will be covered in this week. All tasks would be wrapped up with all documentation and Manual preparation.

Day 91 - Day 95 : By this period, I plan to wrap up all work with proper documentation.

After going through the project in detail, I came up with the above timeline after keeping in consideration all other activities in mind, therefore, I hope that I would be strictly abiding by the above schedule and come up with the best possible result.

Do you have other obligations from late December to March (school, work, vacation, etc.)?
My only obligations are college-related. As stated in my timeline about my exams, I intend to start coding early so that I completely justify my work if been selected.
As requested, I am detailing these:
mandatory laboratory activities (6 hours / week),
homework assignments - (2-3 assignments every 2 weeks) and
exams (5 exams in February that would not extend more than 1 and a half week - though there is plenty of time in between as well which I have alloted for documentation)

About Me (let us know who you are!):

I am a third year undergraduate student at IIT Kharagpur. I am pursuing my five year Integrated M.Sc. course in Mathematics and Computing. I am really interested in programming, and I hope to make my career in this field.

I believe in being versatile and ever-ready for any challenge which might appear, hence, I have tried my hand on various aspects of life, be it technical (I have been the mechanical team member at KRRSG (Kharagpur Robosoccer Students' Group which concentrates on robotics.) or managerial (I have been the organiser of our college's fest). Keeping it more specific,

Technical abilities: I generally code in Java, Python, C and C++. Even though, I have not worked on very big projects, I am very comfortable with these programming languages due to my curriculum (algorithms, data structures, object oriented programming).
I'm also thinking on keeping a blog strictly related to my project progress. I guess reporting every 1-2 weeks (depends) about my development status and getting feedback from the user's community would be really beneficial.
I am having in view a full commitment of 40 hours per week and once enrolled for the task; I'll keep upto the expectations.