Cerridwen Documentation

Release 1.2.0

Leslie P. Polzer

Jan 07, 2019

Contents

Cerridwen provides solar system data that is suitable for both astronomical and astrological purposes. It comes with a simple command-line utility and a JSON server, but is also designed to serve as a basis for your own application.

The motivation for this package is to have a reliable open-source library and API that provides comprehensive data on various planetary bodies and factors at a certain point in time.

CHAPTER 1

Demo server

Take a good look first! :-)

You can check out a demo of the JSON API at this address:

> http://cerridwen.bluemagician.vc/api/v1/moon

This should work with your browser as well.

The current implementation of this API endpoint caches data for 10 seconds. In any case please let me know if you intend to use this for more than testing.

Starting with version 1.1.0 there's also another endpoint with sun data:

> http://cerridwen.bluemagician.vc/api/v1/sun

CHAPTER 2

Quickstart

Are you hooked by now? ;-)

Installation via pip is very simple. Here are some command lines to get you started:

```
pip install cerridwen
```

This will install Cerridwen and its dependencies. Flask will be installed when you start `cerridwen-server` for the first time.

To test Cerridwen's data on the console, invoke:

```
cerridwen
```

If everything is to your satisfaction you can then start the API server if you wish:

```
cerridwen-server
```

It will start up in the foreground and listen on port 2828, serving moon data via HTTP in JSON format at the URI `/v1/moon`.

You can test it as follows:

```
curl http://localhost:2828/v1/moon
```

This should give you a proper JSON response with the current moon data.

Change the listen port by passing the `-p` switch to `cerridwen-server`, followed by the desired port.

CHAPTER 3

Setup from source

Install dependencies

This can be different for every Unix distribution/OS.

This approach should work most of the time:

```
pip3 install cerridwen
```

This will install Cerridwen's release version and its dependencies.

Alternatively, run the following from the toplevel source dir:

```
pip3 -r requirements.txt
```

Afterwards, try to run the CLI applications directly from source:

```
python3 cerridwen/cli.py
```

This should print basic information.

CHAPTER 4

FAQ

4.1 What zodiac is used for the longitudes?

All longitudes whether absolute or relative are based on the tropical zodiac. In this system of reference zero degrees refers to zero degrees tropical Aries, which in turn corresponds to the sun's position at the vernal equinox of the year in question.

4.2 What about other planetary bodies?

Cerridwen's source code is designed to be easily extensible to other planets and points. The goal is to add more planets in the future, probably starting with Mercury.

4.3 Will you add more moon data?

Yes! For example equatorial latitude and lunation numbers.

4.4 What's the precision of the generated data?

Please see the documentation on *Precision*.

4.5 Hey, some of this stuff is slow!

You're right! At the moment the new and full moons are computed anew everytime, which is hard on CPU power. This will change radically with the next version of the module which will have a separate lookup table generation stage for these and other events. This will also pave the way for certain new features like the lunation number.

4.6 How can I help?

First and foremost: use it! Also: tell your friends and fellow astronomers/astrologers!

You can also help write docs, contribute source code and tell me what you'd like to see in the project.

Donations are also welcome, they help me eat and pay my rent! :-) Even 1$ helps.

CHAPTER 5

Contributing

Cerridwen's codebase is on GitHub, at skypher/cerridwen.

Feel free to browse, fork and submit patches and bug reports.

Feature requests are also welcome!

If you need help, you can also write to me at <leslie.polzer@gmx.net>.

CHAPTER 6

Requirements

Cerridwen depends on Python 3. You might be able to make it work with Python 2 as well. Patches welcome! Please let me know if there's a version of Python 3 that does not run Cerridwen properly.

It also depends on these packages:

- pyswisseph, the Python interface to the Swiss Ephemeris library
- numpy, which Cerridwen uses for its ephemeris calculations
- Flask, if you wish to run Cerridwen's API server

These dependencies will be installed automatically as needed.

CHAPTER 7

Precision

There are two main data sources in Cerridwen with slightly different precision characteristics.

Most data, like planetary position and rise/set times, is pulled directly from the Swiss Ephemeris library, whose authors claim a precision of 0.001 arc seconds, or less than 2.8×10^{-7} degrees.

Other data like new and full moon events are generated by Cerridwen's algorithms. The results of these algorithms are guaranteed to be exact within $2 * 10^{-6}$ degrees, or 0.0072 arc seconds. This is in fact ensured by an assertion in the code.

Detection of the next sign change is accurate within $4 * 10^{-6}$ degrees, or 0.0144 arc seconds.

It follows that Cerridwen's calculations are precise enough to get event times down to the correct second.

Warning: Please note that the current implementation of the API server uses memoization for the moon data, generating a new response every 10 seconds only due to efficiency considerations. You can easily turn this off or modify this if you run your own API server, or just wait for the next version of Cerridwen that will be able to calculate new and full moons in a more efficient manner.

CHAPTER 8

Changelog

Only major releases are documented here.

8.1 1.4.1

- Recover from bitrot.
- Amend documentation

8.2 1.4.0

- Rework package structure, picking apart the mudball that was __init__.py.
- Add planets Uranus, Neptune and Pluto. Very basic interface only.
- Add planets Mercury, Venus, Mars, Jupiter, Saturn. Their interface is yet incomplete though.
- Add new Planet methods max_speed, mean_orbital_period, relative_orbital_velocity, average_motion_per_year, aspect_lookahead, default_sample_interval.
- Add sign change detection via Planet.next_sign_change().
- Add Moon.last_new_or_full_moon()
- Precision lowered to 0.0072 arc seconds (was 0.0036), it was needed for proper ingress calculation.
- Update code for astropy 0.4 (rewrote one test case).
- Various bugfixes.

8.3 1.3.0

- Add arc seconds to relative position

- Add right ascension, declination and ecliptical latitude
- Refurbish cli.py

8.4 1.2.0

- Use astropy for time conversions
- Vast documentation update
- Extend test suite
- Remove sun data from moon endpoint response

8.5 1.1.0

- Swiss Ephemeris data files are now included in the package
- Use nose instead of doctest for quick sanity tests
- Add a lot of functions (e.g. rise/set times)
- cerridwen-server: new switch –test/-t for quick testing
- Various minor amendments and changes
- New sun data computation function and API endpoint

8.6 1.0.0

Initial release.

CHAPTER 9

Licensing

Cerridwen is distributed under the MIT license:

```
License for Cerridwen

Copyright (c) 2014 Leslie P. Polzer <leslie.polzer@gmx.net>

Permission is hereby granted, free of charge, to any person obtaining a copy
of this software and associated documentation files (the "Software"), to deal
in the Software without restriction, including without limitation the rights
to use, copy, modify, merge, publish, distribute, sublicense, and/or sell
copies of the Software, and to permit persons to whom the Software is
furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all
copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR
IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,
FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER
LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,
OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE
SOFTWARE.
```

CHAPTER 10

Module API

Much of this is still missing docstrings, so this is just a rough overview.

If you need help, just shoot me a quick mail to <leslie.polzer@gmx.net> with questions.

10.1 Data computation

These functions take an optional Julian day (defaulting to the current point in time) and optional longitudes and latitudes, the latter of which are used for rise/set calculations. You only need to pass latitude and longitude if you want the function's result to include rise/set data. If you do so, you must pass both latitude and longitude.

The return value of these functions is an OrderedDict.

cerridwen.**compute_sun_data**(*jd=None*, *observer=None*)

Collect data for the sun.

Parameters

- **jd** (*float or None*) – reference date as Julian day, defaults to *jd_now()*
- **observer** (*LatLong or None*) – pass the observer position to have the output include rise and set times.

Returns a collection of sun data

Return type OrderedDict

cerridwen.**compute_moon_data**(*jd=None*, *observer=None*)

Collect data for the moon.

Parameters

- **jd** (*float or None*) – reference date as Julian day, defaults to *jd_now()*
- **observer** (*LatLong or None*) – pass the observer position to have the output include rise and set times.

Returns a collection of sun data

Return type OrderedDict

10.2 Date utilities

These functions provide Julian day conversions and printable output.

cerridwen.**jd_now**()

cerridwen.**jd2iso**(*jd*)
Convert a Julian date into an ISO 8601 date string representation

CHAPTER 11

HTTP API

CHAPTER 12

Indices and tables

- genindex
- modindex
- search

Python Module Index

C

Index

C

J