
wx-icons-humanity

Release 0.1.3

Humanity and Humanity Dark icon themes for wxPython

Dominic Davis-Foster

Jul 26, 2023

Documentation

1	Installation	3
1.1	from PyPI	3
1.2	from GitHub	3
2	Documentation	5
2.1	Usage	5
2.2	API Reference	5
2.3	Overview	5
2.4	Coding style	5
2.5	Automated tests	6
2.6	Type Annotations	6
2.7	Build documentation locally	6
2.8	Downloading source code	6
2.8.1	Building from source	7
	Index	9

This package provides wxPython wxArtProvider classes with icons from the Humanity and Humanity Dark Icon Themes.

Installation

1.1 from PyPI

```
$ python3 -m pip install wx_icons_humanity --user
```

1.2 from GitHub

```
$ python3 -m pip install git+https://github.com/domdfcoding/custom_wx_icons_humanity@master --user
```


Documentation

2.1 Usage

To use `wx_icons_humanity` in your application:

```
from wx_icons_humanity import wxHumanityIconTheme

class MyApp(wx.App):
    def OnInit(self):
        wx.ArtProvider.Push(wxHumanityIconTheme())
        self.frame = TestFrame(None, wx.ID_ANY)
        self.SetTopWindow(self.frame)
        self.frame.Show()
        return True
```

And then the icons can be accessed through `wx.ArtProvider`:

```
wx.ArtProvider.GetBitmap('document-new', wx.ART_OTHER, wx.Size(48, 48))
```

Any `FreeDesktop Icon Theme Specification` name can be used.

Currently the *Client ID* is not used, so just pass `wx.ART_OTHER`.

2.2 API Reference

2.3 Overview

`wx_icons_humanity` uses `tox` to automate testing and packaging, and `pre-commit` to maintain code quality.

Install `pre-commit` with `pip` and install the git hook:

```
$ python -m pip install pre-commit
$ pre-commit install
```

2.4 Coding style

`formate` is used for code formatting.

It can be run manually via `pre-commit`:

```
$ pre-commit run formate -a
```

Or, to run the complete autoformatting suite:

```
$ pre-commit run -a
```

2.5 Automated tests

Tests are run with `tox` and `pytest`. To run tests for a specific Python version, such as Python 3.6:

```
$ tox -e py36
```

To run tests for all Python versions, simply run:

```
$ tox
```

2.6 Type Annotations

Type annotations are checked using `mypy`. Run `mypy` using `tox`:

```
$ tox -e mypy
```

2.7 Build documentation locally

The documentation is powered by Sphinx. A local copy of the documentation can be built with `tox`:

```
$ tox -e docs
```

2.8 Downloading source code

The `wx_icons_humanity` source code is available on GitHub, and can be accessed from the following URL:
https://github.com/domdfcoding/custom_wx_icons_humanity

If you have `git` installed, you can clone the repository with the following command:

```
$ git clone https://github.com/domdfcoding/custom_wx_icons_humanity
```

```
Cloning into 'custom_wx_icons_humanity'...
remote: Enumerating objects: 47, done.
remote: Counting objects: 100% (47/47), done.
remote: Compressing objects: 100% (41/41), done.
remote: Total 173 (delta 16), reused 17 (delta 6), pack-reused 126
Receiving objects: 100% (173/173), 126.56 KiB | 678.00 KiB/s, done.
Resolving deltas: 100% (66/66), done.
```

Alternatively, the code can be downloaded in a ‘zip’ file by clicking:

Clone or download → Download Zip

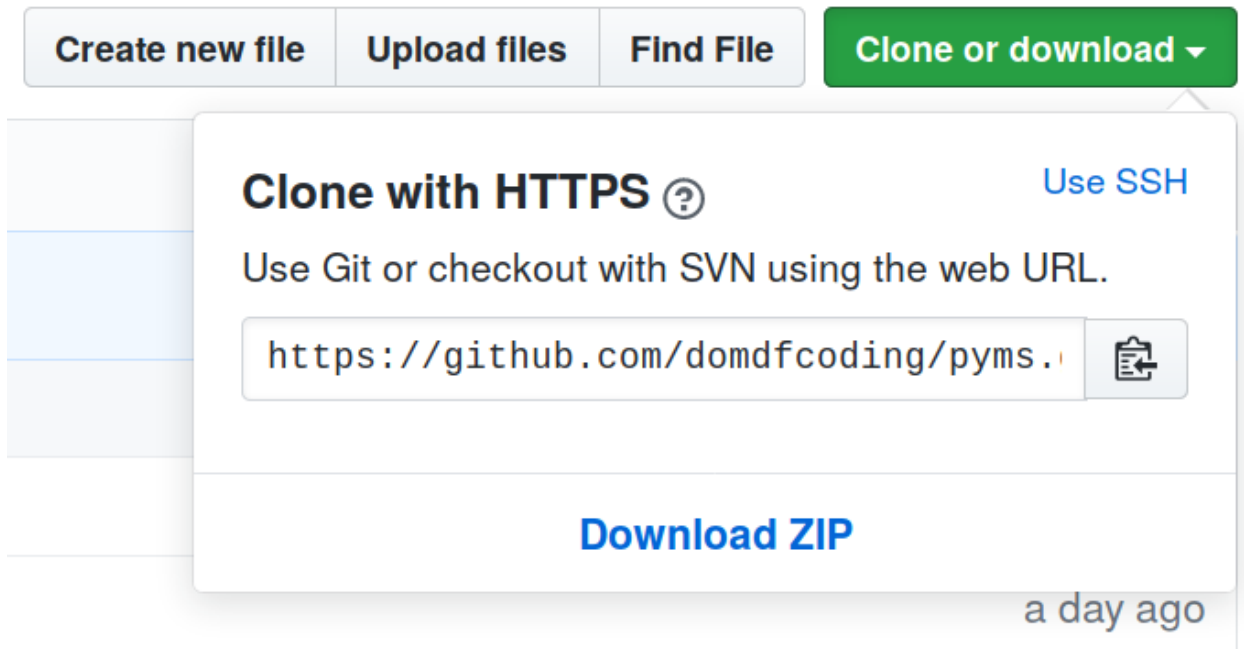


Fig. 1: Downloading a 'zip' file of the source code

2.8.1 Building from source

The recommended way to build `wx_icons_humanity` is to use `tox`:

```
$ tox -e build
```

The source and wheel distributions will be in the directory `dist`.

If you wish, you may also use `pep517.build` or another **PEP 517**-compatible build tool.

P

Python Enhancement Proposals
PEP 517,[7](#)