

---

# **pynotify**

***Release 1.2.4***

**Gene C**

**Mar 29, 2024**



**CONTENTS:**

<b>1</b>	<b>pynotify</b>	<b>1</b>
1.1	Overview . . . . .	1
1.2	Key features . . . . .	1
1.3	New / Interesting . . . . .	1
<b>2</b>	<b>Getting Started</b>	<b>3</b>
2.1	Mask Flags . . . . .	4
<b>3</b>	<b>Appendix</b>	<b>5</b>
3.1	Installation . . . . .	5
3.2	Dependencies . . . . .	5
3.3	Philosophy . . . . .	6
3.4	License . . . . .	6
<b>4</b>	<b>Changelog</b>	<b>7</b>
<b>5</b>	<b>MIT License</b>	<b>9</b>
<b>6</b>	<b>How to help with this project</b>	<b>11</b>
6.1	Important resources . . . . .	11
6.2	Reporting Bugs or feature requests . . . . .	11
6.3	Code Changes . . . . .	11
<b>7</b>	<b>Contributor Covenant Code of Conduct</b>	<b>13</b>
7.1	Our Pledge . . . . .	13
7.2	Our Standards . . . . .	13
7.3	Our Responsibilities . . . . .	13
7.4	Scope . . . . .	14
7.5	Enforcement . . . . .	14
7.6	Attribution . . . . .	14
7.7	Interpretation . . . . .	14
<b>8</b>	<b>Indices and tables</b>	<b>15</b>



**PYNOTIFY**

## **1.1 Overview**

pynotify : Python inotify implementation built atop standard C-library.

## **1.2 Key features**

- Provides a python class to monitor 1 or more file paths for inotify events.

## **1.3 New / Interesting**

Updates and new info goes here



## GETTING STARTED

A simple test program is available in the examples directory. The program monitors */tmp/xxx* - this can be a file or a directory. To get a quick idea in one terminal do:

```
mkdir /tmp/xxx
./examples/test_inotify.py
```

In another terminal do something like: .. code-block:: bash

```
touch /tmp/xxx/A touch /tmp/xxx/B rm /tmp/xxx/* rmdir /tmp/xxx
```

What the code does is essentially:

```
from pynotify import Inotify
inotify = Inotify()
inotify.add_watch('/tmp/xxx')

for events in inotify.get_events():
    for event in events:
        if event:
            print(..)
```

The first loop uses `get_events()` method which is an iterator that returns a list of events. Each even in the list provides for: .. code-block:: bash

```
event.wd # the watch descriptor event.mask # the event mask event.event_types # list of event type enums
from the mask event.path # the path being watched (/tmp/xxx) event.file # Possible subordinate file
(/tmp/xxx/A)
```

Thats it in a nutshell. `add_watch` takes one optional argument:

```
Inotify.add_watch(*path*, mask=xxx)
```

`add_watch()` returns the watched descriptor, *wd*. If successful *wd* is  $\geq 0$ . If  $< 0$  then it means the path is non-existent.

If not provided then the watch is for all possible event types. Where *mask* are event types from the enum *Inotify-Mask.IN\_XXX*. These event elements are the same as provided in */usr/include/sys/inotify.h*.

Before calling `get_events()` there is one additional attribute that can be set on an instance of *Inotify* which is a select timeout which defaults to 5 seconds. .. code-block:: bash

```
inotify.timeout
```

The timeout is passed down to `select()` which waits on the inotify file descriptor for events to provided. If its negative then the select will wait forever, if no events occur. Otherwise the select loop will break after the timeout. A value of zero causes select to return immediately. The default value should provide reasonable behaviour.

## 2.1 Mask Flags

You can get the full list of possible mask flags reading code, which has comments, or using:

```
from pynotify import InotifyMask, Inotify
for item in InotifyMask.mask_to_events(0xFFFFFFFF):
    item
```



## 3.1 Installation

### Available on

- [Github](#)
- [Archlinux AUR](#)

On Arch you can build using the provided PKGBUILD in the packaging directory or from the AUR. To build manually, clone the repo and :

```
rm -f dist/*  
/usr/bin/python -m build --wheel --no-isolation  
root_dest="/"   
./scripts/do-install $root_dest
```

When running as non-root then set root\_dest a user writable directory

## 3.2 Dependencies

- Run Time :
  - python (3.11 or later)
- Building Package:
  - git
  - hatch (aka python-hatch)
  - wheel (aka python-wheel)
  - build (aka python-build)
  - installer (aka python-installer)
  - rsync
- Optional for building docs:
  - sphinx
  - texlive-latexextra (archlinux packaguing of texlive tools)

## 3.3 Philosophy

We follow the *live at head commit* philosophy. This means we recommend using the latest commit on git master branch. We also provide git tags.

This approach is also taken by Google<sup>12</sup>.

## 3.4 License

Created by Gene C. and licensed under the terms of the MIT license.

- SPDX-License-Identifier: MIT
- Copyright (c) 2023 Gene C

---

<sup>1</sup> <https://github.com/google/googletest>

<sup>2</sup> <https://abseil.io/about/philosophy#upgrade-support>

## CHANGELOG

### [1.2.4] — 2024-03-29

Installer takes optional package name - use `in` PKGBUILD  
update Docs/Changelog.rst Docs/pynotify.pdf

### [1.2.3] — 2024-03-29

typo `in` readme.rst  
update Docs/Changelog.rst Docs/pynotify.pdf

### [1.2.2] — 2024-03-29

Fix README.rst title level  
update Docs/Changelog.rst Docs/pynotify.pdf

### [1.2.1] — 2024-03-29

Public release  
Initial commit - python Inotify `class`



## MIT LICENSE

Copyright © 2023 Gene C

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.



## HOW TO HELP WITH THIS PROJECT

Thank you for your interest in improving this project. This project is open-source under the MIT license.

### 6.1 Important resources

- [Git Repo](#)

### 6.2 Reporting Bugs or feature requests

Please report bugs on the issue tracker in the git repo. To make the report as useful as possible, please include

- operating system used
- version of python
- explanation of the problem or enhancement request.

### 6.3 Code Changes

If you make code changes, please update the documentation if it's appropriate.





## **CONTRIBUTOR COVENANT CODE OF CONDUCT**

### **7.1 Our Pledge**

In the interest of fostering an open and welcoming environment, we as contributors and maintainers pledge to making participation in our project and our community a harassment-free experience for everyone, regardless of age, body size, disability, ethnicity, sex characteristics, gender identity and expression, level of experience, education, socio-economic status, nationality, personal appearance, race, religion, or sexual identity and orientation.

### **7.2 Our Standards**

Examples of behavior that contributes to creating a positive environment include:

- Using welcoming and inclusive language
- Being respectful of differing viewpoints and experiences
- Gracefully accepting constructive criticism
- Focusing on what is best for the community
- Showing empathy towards other community members

Examples of unacceptable behavior by participants include:

- The use of sexualized language or imagery and unwelcome sexual attention or advances
- Trolling, insulting/derogatory comments, and personal or political attacks
- Public or private harassment
- Publishing others' private information, such as a physical or electronic address, without explicit permission
- Other conduct which could reasonably be considered inappropriate in a professional setting

### **7.3 Our Responsibilities**

Maintainers are responsible for clarifying the standards of acceptable behavior and are expected to take appropriate and fair corrective action in response to any instances of unacceptable behavior.

Maintainers have the right and responsibility to remove, edit, or reject comments, commits, code, wiki edits, issues, and other contributions that are not aligned to this Code of Conduct, or to ban temporarily or permanently any contributor for other behaviors that they deem inappropriate, threatening, offensive, or harmful.

## 7.4 Scope

This Code of Conduct applies both within project spaces and in public spaces when an individual is representing the project or its community. Examples of representing a project or community include using an official project e-mail address, posting via an official social media account, or acting as an appointed representative at an online or offline event. Representation of a project may be further defined and clarified by project maintainers.

## 7.5 Enforcement

Instances of abusive, harassing, or otherwise unacceptable behavior may be reported by contacting the project team at [<arch@sapience.com>](mailto:arch@sapience.com). All complaints will be reviewed and investigated and will result in a response that is deemed necessary and appropriate to the circumstances. The Code of Conduct Committee is obligated to maintain confidentiality with regard to the reporter of an incident. Further details of specific enforcement policies may be posted separately.

## 7.6 Attribution

This Code of Conduct is adapted from the Contributor Covenant, version 1.4, available at <https://www.contributor-covenant.org/version/1/4/code-of-conduct.html>

## 7.7 Interpretation

The interpretation of this document is at the discretion of the project team.

## INDICES AND TABLES

- `genindex`
- `modindex`
- `search`