

Two-Day MDS-Rely Workshop on Code Packaging and Release

Jungtaek Kim

jungtaek.kim@pitt.edu

University of Pittsburgh

<https://jungtaekkim.github.io>

February 6, 2024 / February 13, 2024

Table of Contents

Overview

Codebase

GitHub Repositories

Git

Open-Source Licenses

Code Refactoring

Code Formatter

Lint

Takeaways





lets see who made this
idiotic program 1 year ago

Overview

Overview

- ▶ These sessions are all example-based.
 - ▶ We will talk about code management, code refactoring, code packaging, and code release.
 - ▶ Many alternatives are available, so you can choose any other frameworks.
 - ▶ There is no right answer for code management and packaging.
 - ▶ You can make your way to manage your project.
-
- ▶ If you are willing to review the code of your project, please let us know.

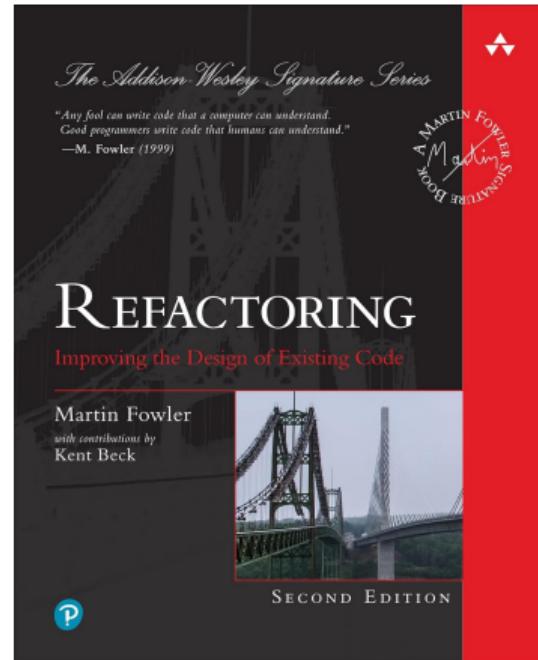
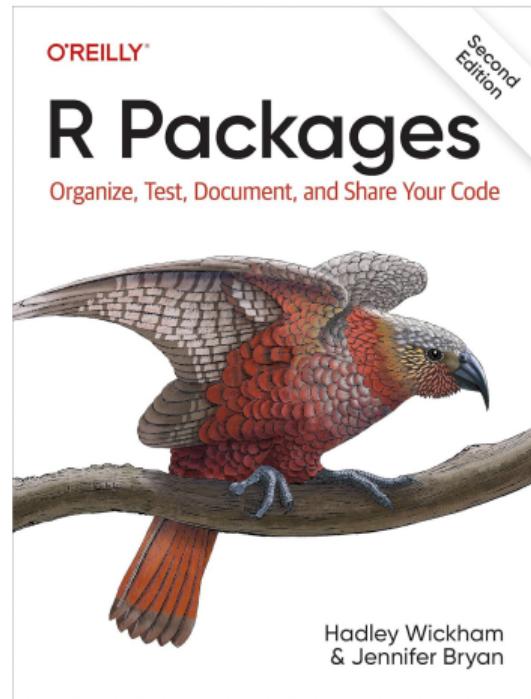
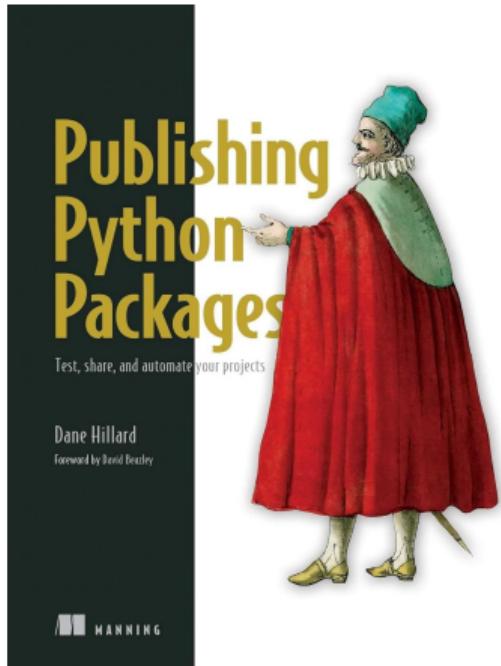
Overview

- ▶ We will look into several popular Python projects:
 - ▶ NumPy,
 - ▶ SciPy,
 - ▶ and scikit-learn.
- ▶ My Python project, BayesO will be explored.
- ▶ Your projects?

Overview

- ▶ We will cover many components for code packaging and release:
 - ▶ GitHub, Git, open-source licenses, code refactoring, code formatter, lint, unit tests, continuous integration, GitHub Actions, documentation, and package repositories.
- ▶ Journal of Open Source Software will be introduced.
- ▶ The goals of most of the code management and deployment tools are to
 - ▶ improve readability,
 - ▶ automate processes,
 - ▶ increase productivity,
 - ▶ speed up software development,
 - ▶ and ensure open access.

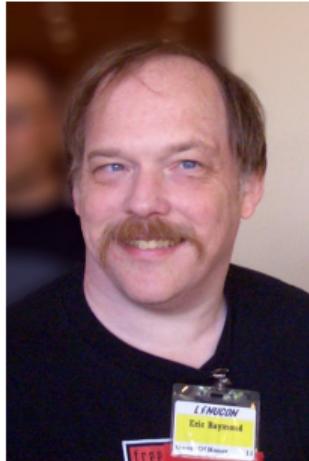
Useful Materials



Key People in Free and Open Source Software



Bruce Perens



Eric S. Raymond



Mark Shuttleworth



Richard Stallman



Linus Torvalds

Figure 1: Alphabetically ordered.

You need to do practice!

Codebase

Codebase

- ▶ Codebase typically contains
 - ▶ main source files,
 - ▶ README,
 - ▶ LICENSE,
 - ▶ CONTRIBUTING,
 - ▶ doc,
 - ▶ unit tests,
 - ▶ lint configuration files,
 - ▶ continuous integration files,
 - ▶ and others.

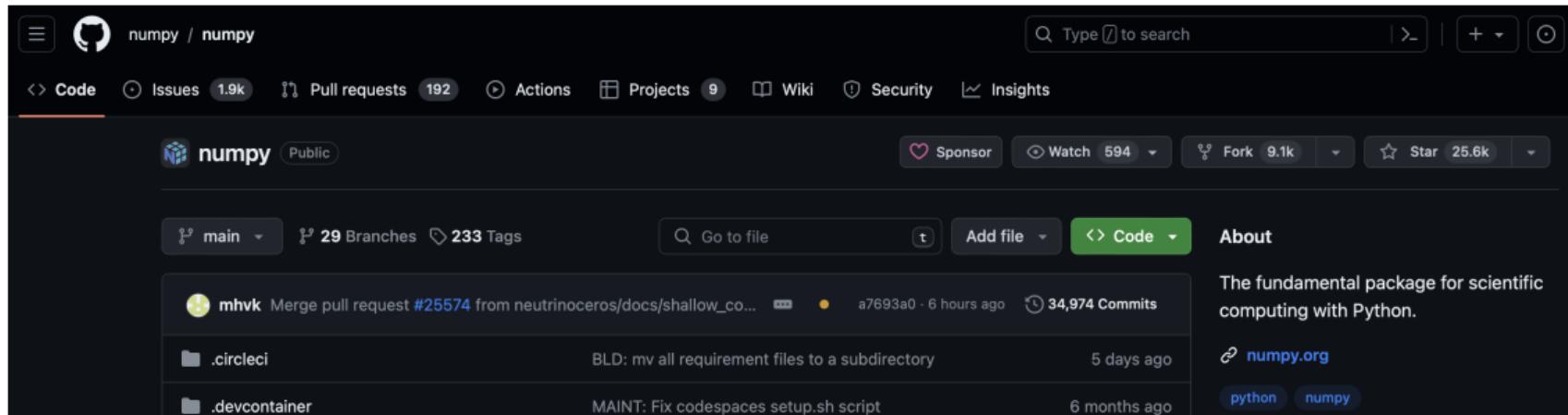
GitHub Repositories

NumPy GitHub Repository

- ▶ NumPy is a Python package for scientific computing.

```
A_M = np.log(2) / 2
B_M = np.log(2250) - A_M * 1971
Moore's_law = lambda year: np.exp(B_M) * np.exp(A_M * year)
```

NumPy GitHub Repository



A screenshot of the NumPy GitHub repository page. The page has a dark theme. At the top, there's a navigation bar with links for Code, Issues (1.9k), Pull requests (192), Actions, Projects (9), Wiki, Security, and Insights. Below the navigation bar, the repository name "numpy / numpy" is displayed along with a GitHub icon. To the right is a search bar with placeholder text "Type ⌂ to search". Further right are buttons for "Sponsor", "Watch 594", "Fork 9.1k", and "Star 25.6k". The main content area shows the "main" branch (29 branches, 233 tags), a "Go to file" search bar, and an "Add file" button. A green "Code" button is highlighted. Below this, a list of recent commits is shown:

- mhv Merge pull request #25574 from neutrinoce... · a7693a0 · 6 hours ago · 34,974 Commits
- .circleci BLD: mv all requirement files to a subdirectory · 5 days ago
- .devcontainer MAINT: Fix codespaces setup.sh script · 6 months ago

On the right side, there's an "About" section with the text: "The fundamental package for scientific computing with Python." Below it are links to "numpy.org" and "python numpy".

NumPy GitHub Repository

- [!\[\]\(dc8effb0c464520bd1295be2a13fc2a1_img.jpg\) Readme](#)
- [!\[\]\(54f0e489c736445a95388ad81636e44f_img.jpg\) View license](#)
- [!\[\]\(32fc3b18e4196bf9b984ab1f48cc0fa3_img.jpg\) Code of conduct](#)
- [!\[\]\(672575146cb8f7a1bb60712f46e7e239_img.jpg\) Security policy](#)
- [!\[\]\(636ba1fb840f541cf5b79833a87a68e4_img.jpg\) Cite this repository ▾](#)
- [!\[\]\(37b71d4c009cee74ec323a0d4e34e145_img.jpg\) Activity](#)
- [!\[\]\(6b14a866ddd9f42f3728aafea606ad81_img.jpg\) Custom properties](#)
- [!\[\]\(bdb580b102a9d1f8b6b0ff755ef155e8_img.jpg\) 25.6k stars](#)
- [!\[\]\(362ed188e1a2445fa1598a88b81c3162_img.jpg\) 594 watching](#)
- [!\[\]\(0a39fa25962a6ed36f96436f969dbfad_img.jpg\) 9.1k forks](#)
- [Report repository](#)

NumPy GitHub Repository

main	29 Branches	233 Tags	Go to file	Add file	Code
 mattip Merge pull request #25761 from seberg/finalize-shape	...	✓	d3345bb · 4 hours ago	⌚ 34,976 Commits	
 .circleci	BLD: mv all requirement files to a subdirectory		5 days ago		
 .devcontainer	MAINT: Fix codespaces setup.sh script		6 months ago		
 .github	Merge pull request #25735 from numpy/dependabot/gith...		4 days ago		
 .spin	CI: use version 0.3.26.0.2 of scipy-openblas wheels (#257...		3 days ago		
 benchmarks	Move from np.bool_ to np.bool		3 months ago		
 branding/logo	DOC: correct Logo SVG files rendered in dark by Figma (#...		4 months ago		
 doc	DEP: Finalize future warning for shape=1 descriptor drop...		7 hours ago		
 meson_cpu	BUG: Fix AVX512 build flags on Intel Classic Compiler		2 weeks ago		
 numpy	DEP: Finalize future warning for shape=1 descriptor drop...		7 hours ago		

NumPy GitHub Repository

README Code of conduct License Security



powered by NumFOCUS PyPI downloads 218M/month Conda downloads 70M stackoverflow Ask questions
DOI 10.1038/s41586-020-2649-2 openssf scorecard 8.7

NumPy is the fundamental package for scientific computing with Python.

- Website: <https://www.numpy.org>
- Documentation: <https://numpy.org/doc>
- Mailing list: <https://mail.python.org/mailman/listinfo/numpy-discussion>
- Source code: <https://github.com/numpy/numpy>
- Contributing: <https://www.numpy.org/devdocs/dev/index.html>
- Bug reports: <https://github.com/numpy/numpy/issues>
- Report a security vulnerability: <https://tidelift.com/docs/security>

NumPy GitHub Repository

[numpy / numpy / _core / multiarray.py](#) ↗



mdhaber and rossbar DOC: empty: standardize notes about uninitialized values (#25695) ⚙ ✓

Code

Blame

1749 lines (1421 loc) · 55.7 KB

Raw



```
1 """
2 Create the numpy._core.multiarray namespace for backward compatibility.
3 In v1.16 the multiarray and umath c-extension modules were merged into
4 a single _multiarray_umath extension module. So we replicate the old
5 namespace by importing from the extension module.
6
7 """
8
9 import functools
10 from . import overrides
11 from . import _multiarray_umath
12 from ._multiarray_umath import * # noqa: F403
```

Other GitHub Repositories

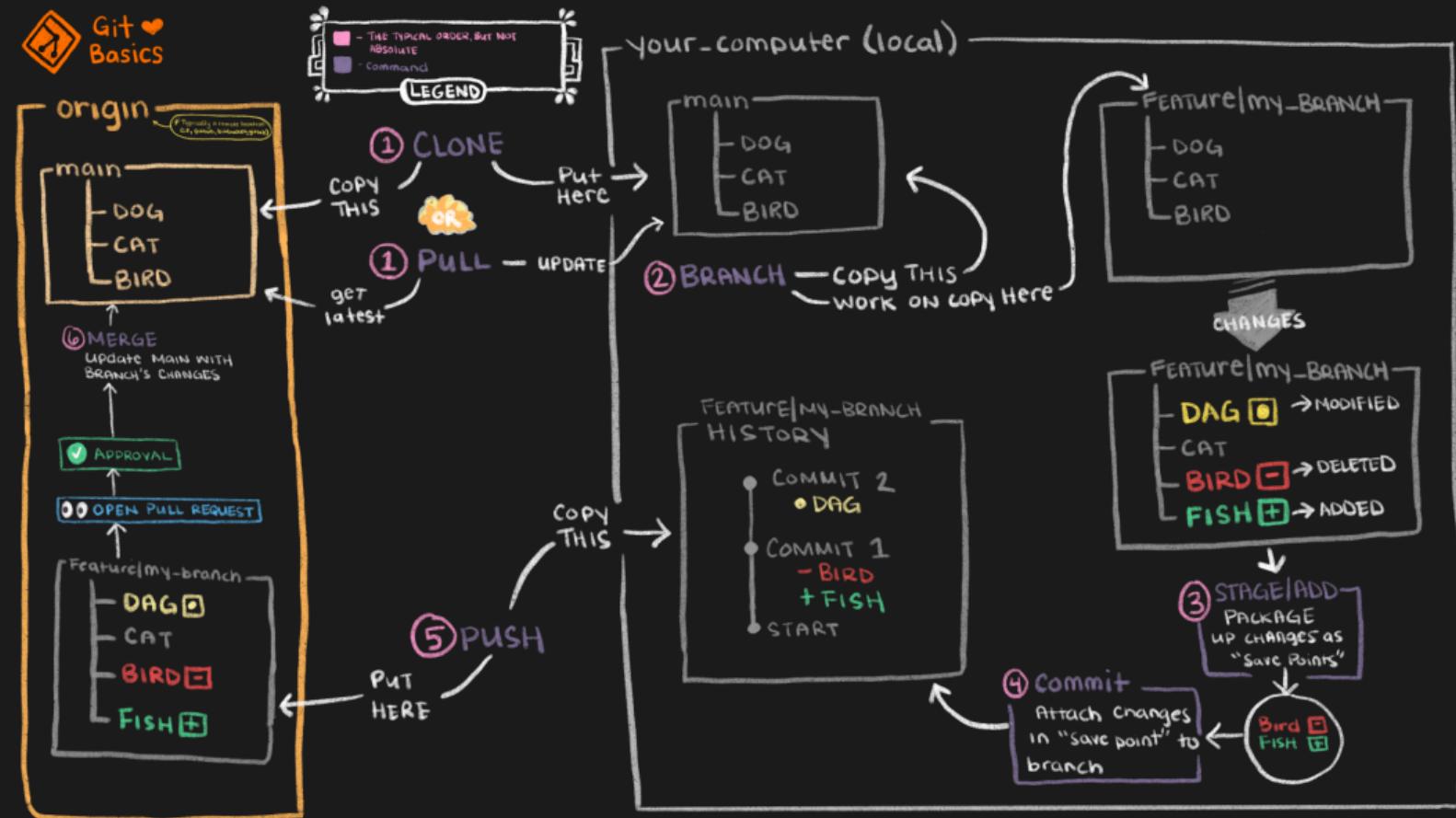
- ▶ SciPy
- ▶ scikit-learn
- ▶ BayesO

Git

- ▶ Git is a distributed version control system.
- ▶ It tracks changes in any set of computer files, usually used for coordinating work among programmers.
- ▶ Git was originally authored by Linus Torvalds in 2005 for development of the Linux kernel.
- ▶ Git has become the most popular distributed version control system, with nearly 95% of developers reporting it as their primary version control system as of 2022.
- ▶ Git repositories as a service are offered by many service providers such as GitHub and BitBucket.



Git Basics

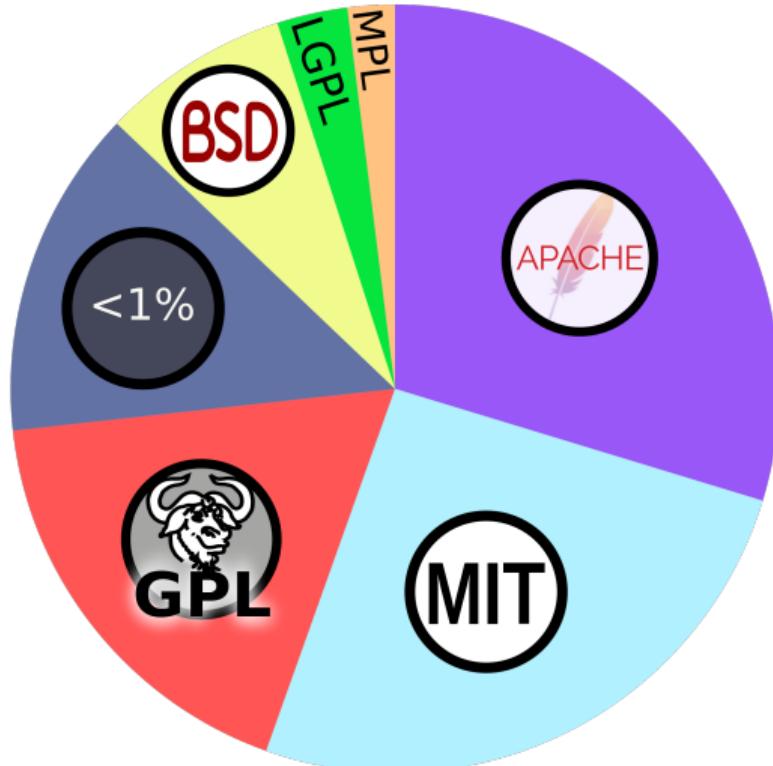


Open-Source Licenses

Open-Source Licenses

- ▶ Open-source licenses are software licenses that allow content to be used, modified, and shared.
- ▶ Free and open-source licenses use these existing legal structures for an inverse purpose.
- ▶ They grant the recipient the rights to use the software, examine the source code, modify it, and distribute the modifications.
- ▶ The two main categories of open-source licenses are permissive and copyleft.
- ▶ We often use the Open Source Initiative Approved Licenses such as the Apache 2.0 License, the GNU General Public License v3, and the MIT License.

Open-Source Licenses



Open-Source Licenses

GNU AGPLv3

Permissions of this strongest copyleft license are conditioned on making available complete source code of licensed works and modifications, which include larger works using a licensed work, under the same license. Copyright and license notices must be preserved. Contributors provide an express grant of patent rights. When a modified version is used to provide a service over a network, the complete source code of the modified version must be made available.

Permissions

- Commercial use
- Distribution
- Modification
- Patent use
- Private use

Conditions

- Disclose source
- License and copyright notice
- Network use is distribution
- Same license
- State changes

Limitations

- Liability
- Warranty

[View full GNU Affero General Public License v3.0 »](#)

Open-Source Licenses

GNU GPLv3

Permissions of this strong copyleft license are conditioned on making available complete source code of licensed works and modifications, which include larger works using a licensed work, under the same license. Copyright and license notices must be preserved. Contributors provide an express grant of patent rights.

Permissions

- Commercial use
- Distribution
- Modification
- Patent use
- Private use

Conditions

- Disclose source
- License and copyright notice
- Same license
- State changes

Limitations

- Liability
- Warranty

[View full GNU General Public License v3.0 »](#)

Open-Source Licenses

GNU GPLv3

Permissions of this copyleft license are conditioned on making available complete source code of licensed works and modifications under the same license or the GNU GPLv3. Copyright and license notices must be preserved. Contributors provide an express grant of patent rights. However, a larger work using the licensed work through interfaces provided by the licensed work may be distributed under different terms and without source code for the larger work.

Permissions

- Commercial use
- Distribution
- Modification
- Patent use
- Private use

Conditions

- Disclose source
- License and copyright notice
- Same license (library)
- State changes

Limitations

- Liability
- Warranty

[View full GNU Lesser General Public License v3.0 »](#)

Open-Source Licenses

Apache License 2.0

A permissive license whose main conditions require preservation of copyright and license notices.

Contributors provide an express grant of patent rights. Licensed works, modifications, and larger works may be distributed under different terms and without source code.

Permissions

- Commercial use
- Distribution
- Modification
- Patent use
- Private use

Conditions

- License and copyright notice
- State changes

Limitations

- Liability
- Trademark use
- Warranty

[View full Apache License 2.0 »](#)

Open-Source Licenses

MIT License

A short and simple permissive license with conditions only requiring preservation of copyright and license notices. Licensed works, modifications, and larger works may be distributed under different terms and without source code.

Permissions

- Commercial use
- Distribution
- Modification
- Private use

Conditions

- License and copyright notice

Limitations

- Liability
- Warranty

[View full MIT License »](#)

NumPy License: Modified BSD License

[README](#) [Code of conduct](#) [License](#) [Security](#)

Copyright (c) 2005–2024, NumPy Developers.
All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

- * Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
- * Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
- * Neither the name of the NumPy Developers nor the names of any contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

SciPy License: BSD License

[README](#) [Code of conduct](#) [BSD-3-Clause license](#) [Security](#)

Copyright (c) 2001-2002 Enthought, Inc. 2003-2024, SciPy Developers.
All rights reserved.

Redistribution and use in source and binary forms, with or without
modification, are permitted provided that the following conditions
are met:

1. Redistributions of source code must retain the above copyright
notice, this list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above
copyright notice, this list of conditions and the following
disclaimer in the documentation and/or other materials provided
with the distribution.
3. Neither the name of the copyright holder nor the names of its
contributors may be used to endorse or promote products derived
from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
"AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR
A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT
OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,
DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
(INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

scikit-learn License: BSD License

[README](#) [Code of conduct](#) [BSD-3-Clause license](#) [Security](#)

BSD 3-Clause License

Copyright (c) 2007-2023 The scikit-learn developers.
All rights reserved.

Redistribution and use in source and binary forms, with or without
modification, are permitted provided that the following conditions are met:

- * Redistributions of source code must retain the above copyright notice, this
list of conditions and the following disclaimer.
- * Redistributions in binary form must reproduce the above copyright notice,
this list of conditions and the following disclaimer in the documentation
and/or other materials provided with the distribution.
- * Neither the name of the copyright holder nor the names of its
contributors may be used to endorse or promote products derived from
this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS"
AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE
DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR CONTRIBUTORS BE LIABLE
FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL
DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR
SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER
CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY,
OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

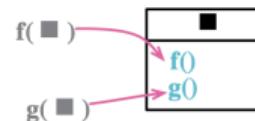
Code Refactoring

Code Refactoring

- ▶ Code refactoring is the process of restructuring existing computer code without changing its external behavior.
- ▶ It is intended to improve the design, structure, and/or implementation of the software, while preserving its functionality.
- ▶ Potential advantages of code refactoring may include improved code readability and reduced complexity.
- ▶ Another potential goal for refactoring is improved performance by writing programs that perform faster or use less memory.
- ▶ If done well, it may help developers discover and fix hidden or dormant bugs in the system by simplifying the underlying logic and eliminating unnecessary complexity levels.
- ▶ If done poorly, it may fail the requirement that external functionality not be changed, and may thus introduce new bugs.

Code Refactoring

Combine Functions into Class



```
function base(aReading) {...}  
function taxableCharge(aReading) {...}  
function calculateBaseCharge(aReading) {...}
```



```
class Reading {  
    base() {...}  
    taxableCharge() {...}  
    calculateBaseCharge() {...}  
}
```

Code Refactoring

Rename Variable

name
nm

```
let a = height * width;
```



```
let area = height * width;
```

Code Refactoring

Replace Magic Literal

$2 * 3.14 * \text{radius}$

π

```
function potentialEnergy(mass, height) {  
    return mass * 9.81 * height;  
}
```



```
const STANDARD_GRAVITY = 9.81;  
function potentialEnergy(mass, height) {  
    return mass * STANDARD_GRAVITY * height;  
}
```

Code Refactoring

Inline Variable



```
let basePrice = anOrder.basePrice;  
return (basePrice > 1000);
```



```
return anOrder.basePrice > 1000;
```

Introduce Assertion

assert (assumption)
=====

```
if (this.discountRate)
    base = base - (this.discountRate * base);
```



```
assert(this.discountRate >= 0);
if (this.discountRate)
    base = base - (this.discountRate * base);
```

Code Formatter

Code Formatter

- ▶ Black is one of the most popular code formatter in Python.
- ▶ It allows us to automatically reformat code in order to satisfy code convention.
- ▶ It can save much time to format code and increase code readability.



Black

```
# in:
```

```
j = [1,  
     2,  
     3  
]
```

```
# out:
```

```
j = [1, 2, 3]
```



Black

```
# in:  
  
ImportantClass.important_method(exc, limit, lookup_lines, capture_locals, extra_argument)  
  
# out:  
  
ImportantClass.important_method(  
    exc, limit, lookup_lines, capture_locals, extra_argument  
)
```

Black

```
# in:

def very_important_function(template: str, *variables, file: os.PathLike, engine: str, header: bool = True, debug: bool = False):
    """Applies `variables` to the `template` and writes to `file`."""
    with open(file, 'w') as f:
        ...

# out:

def very_important_function(
    template: str,
    *variables,
    file: os.PathLike,
    engine: str,
    header: bool = True,
    debug: bool = False,
):
    """Applies `variables` to the `template` and writes to `file`."""
    with open(file, "w") as f:
        ...
```

Black Playground

Black v24.1.0 - The uncompromising Python code formatter.

Playground built by José Padilla

```
1 from seven_dwarfs import Grumpy, Happy, Sleepy, Bashful, Sneezy, Dopey, Doc
2 x = { 'a':37,'b':42,
3
4 'c':927}
5
6 x = 123456789.123456789e123456789
7
8 if very_long_variable_name is not None and \
9 very_long_variable_name.field > 0 or \
10 very_long_variable_name.is_debug:
11 z = 'hello '+world'
12 else:
13 world = 'world'
14 a = 'hello {}'.format(world)
15 f = rf'hello {world}'
16 if (this
17 and that): y = 'hello "world'#FIXME: https://github.com/psf/black/issues/26
18 class Foo ( object ):
19     def f (self ):
20         return 37*-2
21     def g(self, x,y=42):
22         return y
23     def f ( a: List[ int ] ):
24         return 37-a[42-u : y**3]
25     def very_important_function(template: str,*variables,file: os.PathLike,debug:bool=False,):
26         """Applies `variables` to the `template` and writes to `file`."""
27         with open(file, "w") as f:
28             ...
29 # fmt: off
30 custom_formatting = [
31     0, 1, 2,
32     3, 4, 5,
33     6, 7, 8,
34 ]
35 # fmt: on
36 regular_formatting = [
37     0, 1, 2,
38     3, 4, 5,
39     6, 7, 8,
40 ]
```

```
1 from seven_dwarfs import Grumpy, Happy, Sleepy, Bashful, Sneezy, Dopey, Doc
2
3 x = {"a": 37, "b": 42, "c": 927}
4
5 x = 123456789.123456789e123456789
6
7 if (
8     very_long_variable_name is not None
9     and very_long_variable_name.field > 0
10    or very_long_variable_name.is_debug
11    ):
12     z = "hello " + "world"
13 else:
14     world = "world"
15     a = "hello {}".format(world)
16     f = rf"hello {world}"
17 if this and that:
18     y = "hello " "world" # FIXME: https://github.com/psf/black/issues/26
19
20 class Foo(object):
21     def f(self):
22         return 37 * -2
23     def g(self, x, y=42):
24         return y
25
26     def f(a: List[int]):
27         return 37 - a[42 - u : y**3]
28
29     def very_important_function(
30         template: str,
31         *variables,
32         file: os.PathLike,
33         debug: bool = False,
34     ):
35         """Applies `variables` to the `template` and writes to `file`."""
36         with open(file, "w") as f:
37             ...
38
39         # fmt: off
40         custom_formatting = [
41             0, 1, 2,
42             3, 4, 5,
43             6, 7, 8,
44         ]
45         # fmt: on
46         regular_formatting = [
47             0, 1, 2,
48             3, 4, 5,
49             6, 7, 8,
50         ]
```

Lint

Lint

- ▶ Lint is a static code analysis tool used to flag programming errors, bugs, stylistic errors and suspicious constructs.
- ▶ The term originates from a Unix utility that examined C language source code.
- ▶ The analysis performed by lint-like tools can also be performed by an optimizing compiler, which aims to generate faster code.
- ▶ Lint-like tools are especially useful for dynamically typed languages like JavaScript and Python.
- ▶ Because the compilers of such languages typically do not enforce as many and as strict rules prior to execution, linter tools can also be used as simple debuggers for finding common errors (e.g. syntactic discrepancies) as well as hard-to-find errors such as heisenbugs (drawing attention to suspicious code as “possible errors”).

--argument-naming-style

Naming style matching correct argument names.

Default: `snake_case`

--bad-names

Bad variable names which should always be refused, separated by a comma.

Default: ('foo', 'bar', 'baz', 'toto', 'tutu', 'tata')

--class-const-naming-style

Naming style matching correct class constant names.

Default: UPPER_CASE

--max-attributes

Maximum number of attributes for a class (see R0902).

Default:

--max-returns

Maximum number of return / yield for function / method body.

Default:

--max-line-length

Maximum number of characters on a single line.

Default: `100`

--max-nested-blocks

Maximum number of nested blocks for function / method body

Default:

Go to the example of Pylint.

Takeaways

Takeaways

- ▶ Do refactoring frequently.
- ▶ Do practice in your project.
- ▶ Open your project.
- ▶ Update your project consistently.
- ▶ Contribute to other open-source projects.

Thank you!