

IceCube Summer School 2023

Coding and Software Best Practices

Pierpaolo Savina
pierpaolo.savina@icecube.wisc.edu

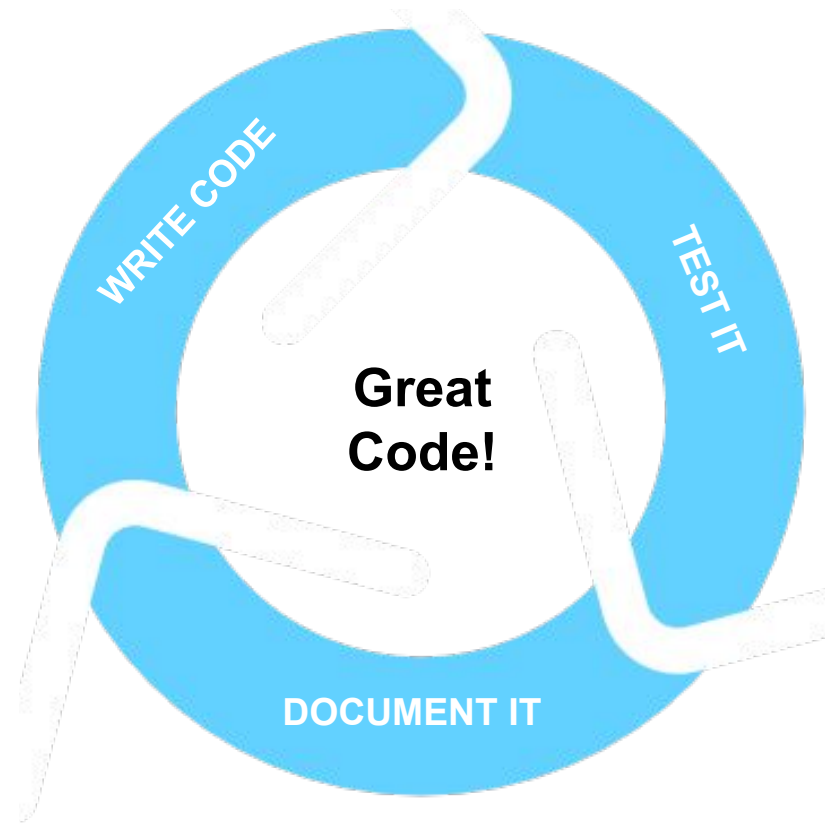
Outline

How to write good **code** in the most efficient way:

- coding standards and style
- version control

Tests: make sure your code works the way you say it does.

Documentation:
How to share your code



Introduction

Coding standards:

set of guidelines and best practices that are used to create **consistent**, high-quality code.

Rules, techniques, and best practices to develop **cleaner**, and **readable** code.



How to make a good code

Coding Standards:

- **Improve Code Quality:** Coding standards ensure code written consistently.
Easier to understand and use.
- **Increase Efficiency:** avoid common mistakes and implementing proven solutions.
- **Facilitate Collaboration:** easier to share.

O'REILLY®

Copyrighted Material



Becoming
a Better
Programmer

A HANDBOOK FOR PEOPLE WHO CARE ABOUT CODE

Which Language?

In IceCube, usually written in **python** or **C++**

Python:

- Easier
- Slower (for intensive calculations)

C++:

- Harder to read
- Big and complex calculations

```
C++
1 #include <iostream>
2
3 using namespace std;
4
5 int main() {
6     int secret_number, num_of_guesses, user_guess;
7
8     secret_number = 42;
9
10    num_of_guesses = 1;
11
12    // Read in user's guess as an integer
13    cout << "Please enter a number: ";
14    cin >> user_guess;
15
16    while ((user_guess != secret_number) && (num_of_guesses < 5)) {
17        cout << "Incorrect. " << "Please enter a number: ";
18        cin >> user_guess;
19        num_of_guesses++;
20    }
21
22    if (user_guess == secret_number) {
23        cout << "Correct" << endl;
24    } else {
25        cout << "Incorrect. Game over." << endl;
26    }
27
28    return 0;
29 }
```

```
Python
1 secret_number = 42
2
3 num_of_guesses = 1
4
5 # Read in user's guess as an integer
6 user_guess = int(input("Please enter a number: "))
7
8 while user_guess != secret_number and num_of_guesses < 5:
9     print("Incorrect. ")
10    user_guess = int(input("Please enter a number: "))
11    num_of_guesses = num_of_guesses + 1
12
13 if user_guess == secret_number:
14     print("Correct")
15 else:
16     print("Incorrect. Game over.")
```

How to write good code

Write modular code:

- A function should perform a single purpose with minimal operations.
- Easier and faster to fix smaller parts.
- Extend to big picture: Good software does **ONE** single task **REALLY WELL**.

Write your code with **consistency** -> **style guides**

Use version control: **git**

Write your code with an editor: **Emacs**, **Jupyter Notebooks**, **vscode**

```
fig, ax = plt.subplots(1,1, figsize=(8,8), dpi=100)

def plot_azimuth_diff(df, rec, ax, label, color):
    dTheta = np.degrees(df[rec]) - np.degrees(df['mc_azimuth'])
    median = np.median(dTheta)

    n_events = len(df)
    weights = pd.Series([1/n_events]*n_events)

    ax.hist(dTheta, histtype='step', linewidth=3, bins=np.linspace(-90,90,51),
            weights=weights, label=label, color=color)
    ax.axvline(median, color=color, linewidth=2, linestyle='--')

plot_azimuth_diff(data_df, 'dnn_azimuth', ax, 'Event Generator', 'blue')
plot_azimuth_diff(data_df, 'monopod_azimuth', ax, 'Monopod', 'red')
ax.set_xlabel(r'$\Delta \phi$ [deg]', fontsize=20)
ax.set_ylabel('Normalized distribution', fontsize=20)
ax.set_title('DNN Cascades - Monopod Comparison (Burnt Sample)')
ax.set_xlim(-45,45)

ax.legend(fontsize=20)
```

NOTE: personally recommend vscode: great support to both languages (autocomplete, easier debugging). Moreover Jupyter Notebooks can be executed inside vscode

Coding Standards and Style Guides

Choose a style guide and integrate a linter into your workflow:

- [Python PEP8](#)
- [Google Style Guides](#)
- [C++](#)

Linting: automated checking of your source code for programmatic and stylistic errors

Linters/Code Checkers

- C++
 - [cpplint](#)
 - [clang-analyzer](#)
- Python:
 - [flake8](#)
 - [black](#)

Structure of a project

Organize code:

```
README.rst
LICENSE
setup.py
requirements.txt
sample/__init__.py
sample/core.py
sample/helpers.py
docs/conf.py
docs/index.rst
tests/test_basic.py
tests/test_advanced.py
```

README: The document every new user will read before using new code.

- Describe the general purpose of the code.
- Explain how to install and (eventually) compile.
- Write clear README's with lots of description.

LICENSE: Whether you make a project on your own or contribute a feature to a project, code can be released for public use.

Keep your contact information updated with IceCube, you may be reached for licensing purposes.

Version Control with git

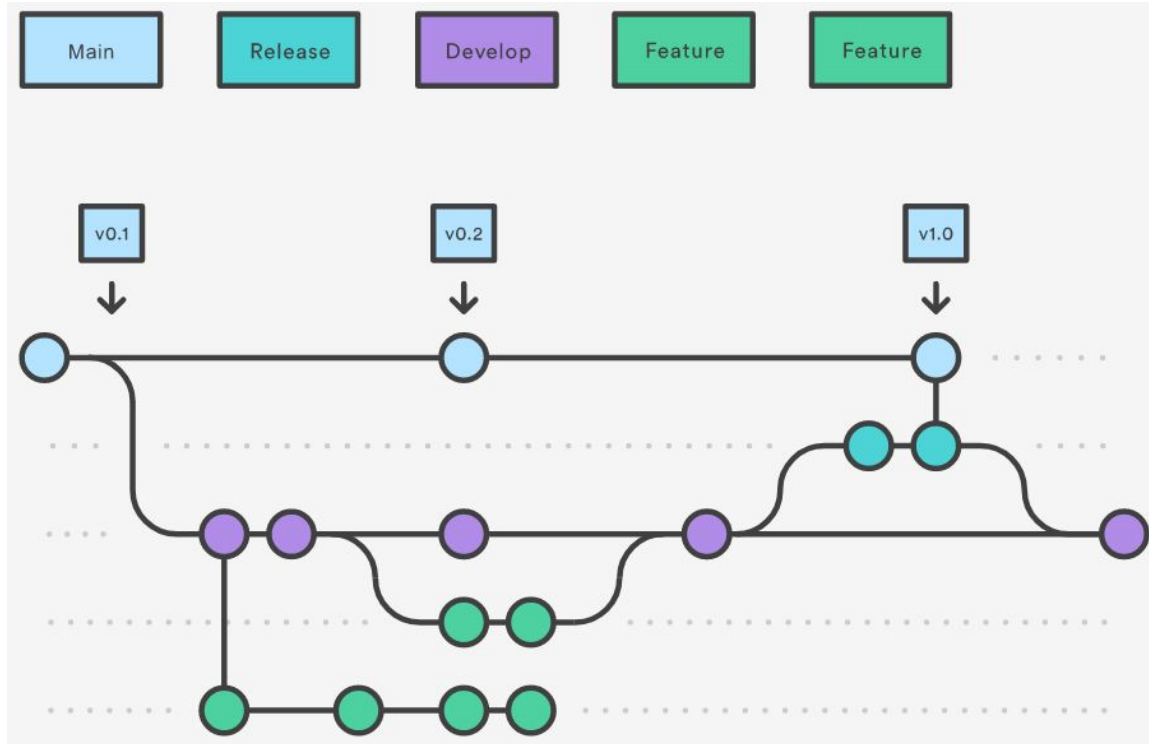
Git is a Version Control System (VCS): **multiple versions of a code**, shared across multiple developers.

See changes you make to your code and easily revert them.

Github.com is a website that hosts git repositories on a remote server ([Github guide](#))

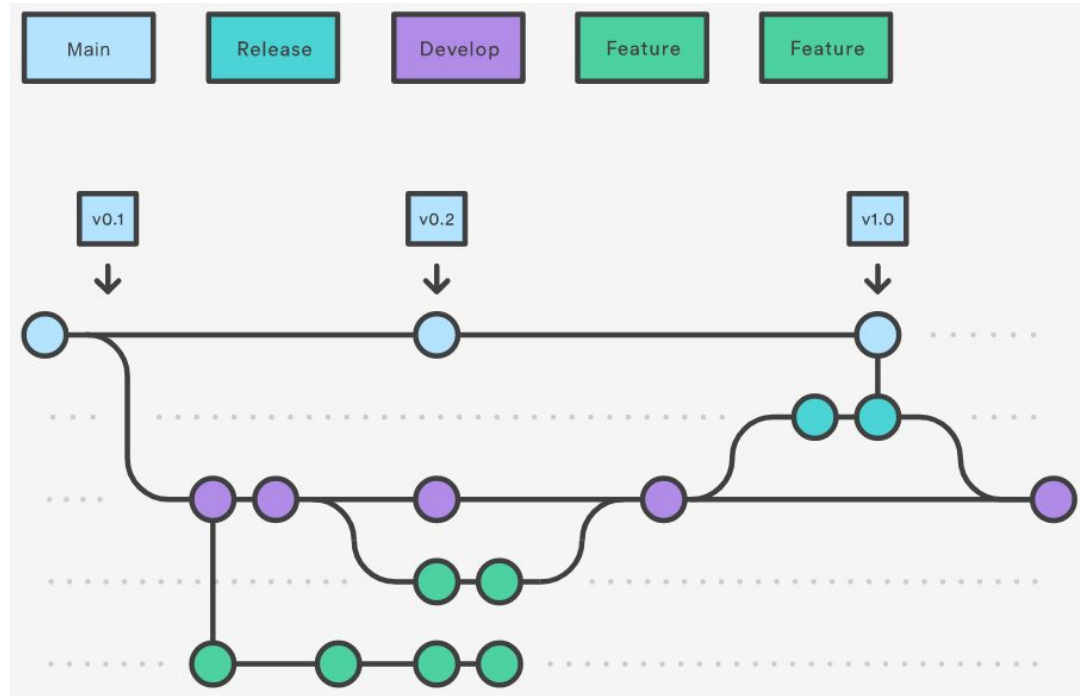
Workflow:

Code → Test → Commit → Merge



Branches

- Branches used to experiment and add features to software, integrated after testing.
- In IceCube: branches used for different analyses.
- Name your branch after your name and analysis.
- Unless your work is to improve/add features to widely-used software, analyzers do not attempt to merge with the main branch.
- your code should be independent of the software infrastructure.



How to make commits

```
git add .  
git commit -m 'commit message'  
git push origin <branch>
```

The “**commit message**” explains the update.

Each commit **narrowly-focused**:
one update at a time.
Small updates often.

1. Correctness, simplicity, and clarity come first.
2. Use tests and documentation as internal design checks.
3. Make small, atomic commits.
4. Keep style-related commits separate from functional changes.
5. Prefer short-lived branches.

Testing your code

- Tests verify that all code functions operate successfully and as designed.
- Tests should sample all minimal examples.
- Tests are described in your docs.
- **MAKE SURE TESTS WORK BEFORE RELEASING NEW VERSIONS OR MERGING (next slides).**

Documenting the code

Main ways to document: **README**, **comments** and **tech notes**.

README:

Used to share important informations about your analysis:

- Requirements
- How to install
- How to use it

Comments:

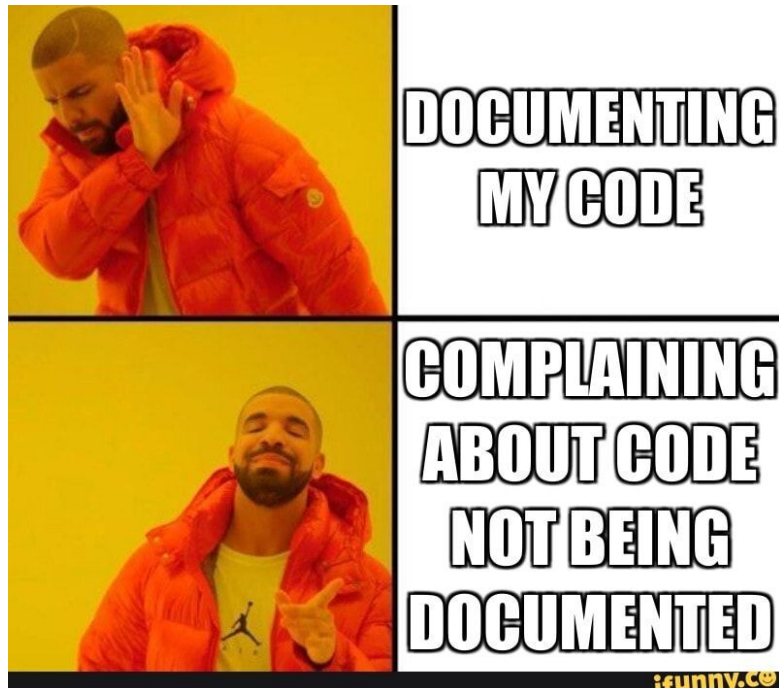
Lines in your code where you explain what's happening.

- Comment everywhere and a lot.
- Explain what the code does at each step.
- Docstrings: Comments inside functions about the purpose.

Tech note:

A paper that describes the full details of a software suite.

- Explain motivation, include relevant mathematics, plots.
- Show examples for all/common scenarios.
- List classes and functions with definitions.



ChatGPT is your friend

[ChatGPT](#)

Introducing ChatGPT

We've trained a model called ChatGPT which interacts in a conversational way. The dialogue format makes it possible for ChatGPT to answer followup questions, admit its mistakes, challenge incorrect premises, and reject inappropriate requests.

[Try ChatGPT ↗](#)

[Read about ChatGPT Plus](#)

Might help with writing code as well as documenting!!!

Exercise

P Dear ChatGPT, can you help me create a tutorial for using github?

I would like to create a new repository with a readme and a .gitignore. After that I would like to create a branch called developer where I add sentences to the readme
Finally I would like to do a merge to the main branch



Certainly! I'd be happy to help you create a tutorial for using GitHub. Here's a step-by-step guide for achieving the tasks you mentioned:

1. Sign up for GitHub:
 - Visit the GitHub website (<https://github.com>) and sign up for a new account if you haven't done so already.
 - Follow the instructions to complete the account creation process.
2. Create a new repository:
 - Once you're logged in, click on the "+" icon at the top right corner of the page.
 - Select "New repository" from the dropdown menu.
 - Enter a name for your repository, choose whether it should be public or private, and provide an optional description.
 - Check the "Initialize this repository with a README" option to create a README file automatically.
 - You can also choose to add a .gitignore file for a specific programming language by selecting the appropriate option from the dropdown menu.
 - Click on the "Create repository" button to create the new repository.

Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere? [Import a repository](#).

Owner *



psavina

Repository name *

IceCubeSummerSchool2023

✔ IceCubeSummerSchool2023 is available.

Great repository names are short and memorable. Need inspiration? How about [shiny-octo-giggle?](#)

Description (optional)

A simple exercise



Public

Anyone on the internet can see this repository. You choose who can commit.



Private

You choose who can see and commit to this repository.

Initialize this repository with:

Add a README file

This is where you can write a long description for your project. [Learn more about READMEs](#).

Add .gitignore

.gitignore template: Python

Choose which files not to track from a list of templates. [Learn more about ignoring files](#).

Choose a license

License: GNU General Public License v3.0

A license tells others what they can and can't do with your code. [Learn more about licenses](#).

This will set `main` as the default branch. Change the default name in your [settings](#).

ⓘ You are creating a public repository in your personal account.

Check out Aswathi's tutorial tomorrow, she's doing a more complete exercise!!!

Exercise

3. Clone the repository:

- On the repository page, click on the "Code" button.
- Copy the repository URL.
- Open a terminal or command prompt on your local machine.
- Navigate to the directory where you want to clone the repository.
- Run the following command, replacing `**<repository-url>**` with the URL you copied:

```
bash Copy code  
git clone <repository-url>
```

- This will create a local copy of the repository on your machine.

4. Create a new branch:

- Open a terminal or command prompt and navigate to the repository's directory.
- Run the following command to create a new branch called "developer":

```
Copy code  
git branch developer
```

- Switch to the new branch with the following command:

```
Copy code  
git checkout developer
```

5. Edit the README file:

- Open the README file in a text editor of your choice.
- Add the sentences or any other content you want to include.
- Save the file.

The screenshot shows the GitHub repository page for 'psavina/Initial commit'. At the top, there are navigation buttons for 'main', '1 branch', and '0 tags'. Below that, there are buttons for 'Go to file', 'Add file', and 'Code'. The repository name 'psavina Initial commit' is displayed, along with the commit hash '522124f' and the time 'now'. A table lists the files in the repository:

File	Commit	Time
.gitignore	Initial commit	now
LICENSE	Initial commit	now
README.md	Initial commit	now

Below the table, the 'README.md' file is shown. The content of the README is:

```
IceCubeSummerSchool2023  
  
A simple exercise
```

```
psavina@i3LaptopPierpaolo:~$ git clone https://github.com/psavina/IceCubeSummerSchool2023.git  
Cloning into 'IceCubeSummerSchool2023'...  
remote: Enumerating objects: 5, done.  
remote: Counting objects: 100% (5/5), done.  
remote: Compressing objects: 100% (4/4), done.  
remote: Total 5 (delta 0), reused 0 (delta 0), pack-reused 0  
Unpacking objects: 100% (5/5), 13.97 KiB | 3.49 MiB/s, done.  
psavina@i3LaptopPierpaolo:~$ cd IceCubeSummerSchool2023/  
psavina@i3LaptopPierpaolo:~/IceCubeSummerSchool2023$ git branch developer  
psavina@i3LaptopPierpaolo:~/IceCubeSummerSchool2023$ git checkout developer  
Switched to branch 'developer'  
psavina@i3LaptopPierpaolo:~/IceCubeSummerSchool2023$
```

```
# IceCubeSummerSchool2023  
A simple exercise  
  
I Like Pizza!
```

Exercise

6. Commit and push changes:

- Go back to the terminal or command prompt.
- Run the following command to stage the changes you made:

```
csharp Copy code  
  
git add README.md
```

- Commit the changes with a descriptive message:

```
sql Copy code  
  
git commit -m "Added sentences to README"
```

- Push the changes to the remote repository:

```
perl Copy code  
  
git push origin developer
```

7. Merge the branch to the main branch:

- Go to the repository's page on GitHub.
- Click on the "Pull requests" tab.
- Click on the "New pull request" button.
- Set the base branch to "main" and the compare branch to "developer".
- Review the changes and add a description for the pull request.
- Click on the "Create pull request" button.
- If there are no conflicts, click on the "Merge pull request" button to merge the changes into the main branch.

```
psavina@i3LaptopPierpaolo:~$ git clone https://github.com/psavina/IceCubeSummerSchool2023.git  
Cloning into 'IceCubeSummerSchool2023'...  
remote: Enumerating objects: 5, done.  
remote: Counting objects: 100% (5/5), done.  
remote: Compressing objects: 100% (4/4), done.  
remote: Total 5 (delta 0), reused 0 (delta 0), pack-reused 0  
Unpacking objects: 100% (5/5), 13.97 KiB | 3.49 MiB/s, done.  
psavina@i3LaptopPierpaolo:~$ cd IceCubeSummerSchool2023/  
psavina@i3LaptopPierpaolo:~/IceCubeSummerSchool2023$ git branch developer  
psavina@i3LaptopPierpaolo:~/IceCubeSummerSchool2023$ git checkout developer  
Switched to branch 'developer'  
psavina@i3LaptopPierpaolo:~/IceCubeSummerSchool2023$ nano README.md  
psavina@i3LaptopPierpaolo:~/IceCubeSummerSchool2023$ git add .  
psavina@i3LaptopPierpaolo:~/IceCubeSummerSchool2023$ git commit -m 'Included my food preferences inside README'  
[developer b5be608] Included my food preferences inside README  
1 file changed, 3 insertions(+)  
psavina@i3LaptopPierpaolo:~/IceCubeSummerSchool2023$ git push origin developer  
Username for 'https://github.com': psavina  
Password for 'https://psavina@github.com':  
Enumerating objects: 5, done.  
Counting objects: 100% (5/5), done.  
Delta compression using up to 8 threads  
Compressing objects: 100% (3/3), done.  
Writing objects: 100% (3/3), 347 bytes | 347.00 KiB/s, done.  
Total 3 (delta 1), reused 0 (delta 0)  
remote: Resolving deltas: 100% (1/1), completed with 1 local object.  
remote:  
remote: Create a pull request for 'developer' on GitHub by visiting:  
remote:   https://github.com/psavina/IceCubeSummerSchool2023/pull/new/developer  
remote:  
To https://github.com/psavina/IceCubeSummerSchool2023.git  
   * [new branch]   developer -> developer  
psavina@i3LaptopPierpaolo:~/IceCubeSummerSchool2023$
```

Comparing changes

Choose two branches to see what's changed or to start a new pull request. If you need to, you can also [compare across forks](#).

base: main ← compare: developer ✓ Able to merge. These branches can be automatically merged.

Discuss and review the changes in this comparison with others. [Learn about pull requests](#) Create pull request

1 commit 1 file changed 1 contributor

Commits on Jun 5, 2023


Included my food preferences inside README
Pierpaolo Savina committed 3 minutes ago

Showing 1 changed file with 3 additions and 0 deletions. Split Unified

```
▼ 3 README.md  
... .. 00 -1,2 +1,5 00  
1 1 # IceCubeSummerSchool2023  
2 2 A simple exercise  
3 +  
4 +  
5 + I Like Pizza!
```

Last advices

- #software in the IceCube Slack for any questions/errors with code.
- Also ask in your Working Group channels (do this first!)
- Go to #icecube-it for any hardware-related issues.
- When in doubt, ask!
- Learn and use GitHub now so you don't have to during a code review!

A close-up image of Morpheus from the movie The Matrix, wearing his signature sunglasses and a serious expression. The image is used as a background for a meme.

What if I told you

**That spaghetti should be served as
a part of a meal, not as a part of your code**