MOD300 Anvendt Python programmering og modellering

Enrico Riccardi[1]

Department of Mathematics and Physics, University of Stavanger (UiS).[1]

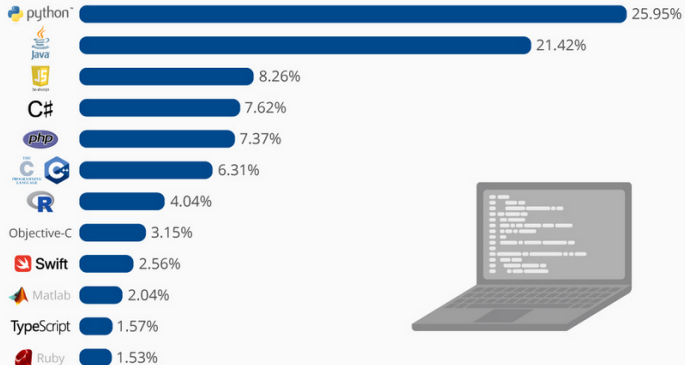Sep 29, 2025

University
of Stavanger

The Most Popular Programming Languages
Share of the most popular programming languages in the world*

Language	Share
python	25.95%
Java	21.42%
JS	8.26%
C#	7.62%
php	7.37%
C C++	6.31%
R	4.04%
Objective-C	3.15%
Swift	2.56%
Matlab	2.04%
TypeScript	1.57%
Ruby	1.53%

* Based on the PYPL-Index, an analysis of Google search trends
for programming language tutorials.
Source: PYPL

@StatistaCharts

statista

What makes python sexy?

- Community
- Training material for LLMs
- Environments
- Integration with other software
- Speed
- Readability
- Re-usability
- M-L libraries
- Community standards

Coding standards

🔘 Lifetime	Use
1-shot	🚫
Week+	Git + Github/GitLab
3 month+	+ Testing
6 month+	+ Documentation, automated testing

👹👹 Dev/Users	Use
1	Push to main
2+	+ Branches, merging
2+ (+students)	+ Code review
2+ (+external)	+ Release branch

Different code editors are available to interpret python language.

- jupyter notebooks are mostly dedicated to learning (Markdown)

- ipython is for interactive coding (similar to R, Matlab, etc)

- python packages (.py) developing suites (debug possibilities and git integration)

Different code editors are available to interpret python language.

- jupyter notebooks are mostly dedicated to learning (Markdown)
- ipython is for interactive coding (similar to R, Matlab, etc)
- python packages (.py) developing suites (debug possibilities and git integration)

Different code editors are available to interpret python language.

- jupyter notebooks are mostly dedicated to learning (Markdown)
- ipython is for interactive coding (similar to R, Matlab, etc)
- python packages (.py) developing suites (debug possibilities and git integration)

Different code editors are available to interpret python language.

- jupyter notebooks are mostly dedicated to learning (Markdown)
- ipython is for interactive coding (similar to R, Matlab, etc)
- python packages (.py) developing suites (debug possibilities and git integration)

When developing code, there are **guidelines** and best practices aimed at improving the quality, readability, and maintainability of a code.

There are different levels of coding quality, mostly depending on the code intended usage (and developer skills).

- Private codes can be whatever (Cpt. Obvious)
- Public packages shall use a 'Golden code standards' such to be used and eventually supported by communities.

When developing code, there are **guidelines** and best practices aimed at improving the quality, readability, and maintainability of a code.

There are different levels of coding quality, mostly depending on the code intended usage (and developer skills).

- Private codes can be whatever (Cpt. Obvious)
- Public packages shall use a 'Golden code standards' such to be used and eventually supported by communities.

Golden code standard

Principle of 'clean coding':

1. Readability and Clarity: A good code shall be possible to read as when reading a book

2. Structure and object oriented: A code shall be composed by objects, each of them connected in the less redundant way possible.

3. Consistency and Style: Variable naming, function naming and classes naming has to be consistent.

4. Documentation: Each file, each function and each class shall contain the relative description of its aim and its usage

5. Maintainability: Code dependencies have to be stated and consistently defined and updated, such that a suitable environment can be developed at any point in time

Golden code standard

Principle of 'clean coding':

1. Readability and Clarity: A good code shall be possible to read as when reading a book

2. Structure and object oriented: A code shall be composed by objects, each of them connected in the less redundant way possible.

3. Consistency and Style: Variable naming, function naming and classes naming has to be consistent.

4. Documentation: Each file, each function and each class shall contain the relative description of its aim and its usage

5. Maintainability: Code dependencies have to be stated and consistently defined and updated, such that a suitable environment can be developed at any point in time

Golden code standard

Principle of 'clean coding':

1. Readability and Clarity: A good code shall be possible to read as when reading a book

2. Structure and object oriented: A code shall be composed by objects, each of them connected in the less redundant way possible.

3. Consistency and Style: Variable naming, function naming and classes naming has to be consistent.

4. Documentation: Each file, each function and each class shall contain the relative description of its aim and its usage

5. Maintainability: Code dependencies have to be stated and consistently defined and updated, such that a suitable environment can be developed at any point in time.

Golden code standard

Principle of 'clean coding':

1. **Readability and Clarity**: A good code shall be possible to read as when reading a book

2. **Structure and object oriented**: A code shall be composed by objects, each of them connected in the less redundant way possible.

3. **Consistency and Style**: Variable naming, function naming and classes naming has to be consistent.

4. Documentation: Each file, each function and each class shall contain the relative description of its aim and its usage

5. Maintainability: Code dependencies have to be stated and consistently defined and updated, such that a suitable environment can be developed at any point in time.

Golden code standard

Principle of 'clean coding':

1. **Readability and Clarity**: A good code shall be possible to read as when reading a book

2. **Structure and object oriented**: A code shall be composed by objects, each of them connected in the less redundant way possible.

3. **Consistency and Style**: Variable naming, function naming and classes naming has to be consistent.

4. **Documentation**: Each file, each function and each class shall contain the relative description of its aim and its usage

5. Maintainability: Code dependencies have to be stated and consistently defined and updated, such that a suitable environment can be developed at any point in time.

Principle of 'clean coding':

1. Readability and Clarity: A good code shall be possible to read as when reading a book

2. Structure and object oriented: A code shall be composed by objects, each of them connected in the less redundant way possible.

3. Consistency and Style: Variable naming, function naming and classes naming has to be consistent.

4. Documentation: Each file, each function and each class shall contain the relative description of its aim and its usage

5. Maintainability: Code dependencies have to be stated and consistently defined and updated, such that a suitable environment can be developed at any point in time.

1. Testing: Unit testing shall cover the majority of the code
2. Error Handling: Each error shall be captured and properly identified.
3. Examples and benchmarks: Users shall be able to execute minimal examples of the code for computational checks.
4. Performance Optimization: Libraries shall be able to use the available computational power in the machine (e.g. GPU-CUDA)

1. Testing: Unit testing shall cover the majority of the code
2. Error Handling: Each error shall be captured and properly identified.
3. Examples and benchmarks: Users shall be able to execute minimal examples of the code for computational checks.
4. Performance Optimization: Libraries shall be able to use the available computational power in the machine (e.g. GPU-CUDA)

1. Testing: Unit testing shall cover the majority of the code
2. Error Handling: Each error shall be captured and properly identified.
3. Examples and benchmarks: Users shall be able to execute minimal examples of the code for computational checks.
4. Performance Optimization: Libraries shall be able to use the available computational power in the machine (e.g. GPU-CUDA)

1. Testing: Unit testing shall cover the majority of the code
2. Error Handling: Each error shall be captured and properly identified.
3. Examples and benchmarks: Users shall be able to execute minimal examples of the code for computational checks.
4. Performance Optimization: Libraries shall be able to use the available computational power in the machine (e.g. GPU-CUDA)

1. Testing: Unit testing shall cover the majority of the code
2. Error Handling: Each error shall be captured and properly identified.
3. Examples and benchmarks: Users shall be able to execute minimal examples of the code for computational checks.
4. Performance Optimization: Libraries shall be able to use the available computational power in the machine (e.g. GPU-CUDA)

Git is a distributed version control system that tracks changes in any set of computer files, usually used for coordinating work among programmers who are collaboratively developing source code during software development. Its goals include speed, data integrity, and support for distributed, non-linear workflows (thousands of parallel branches running on different computers). [Wiki]
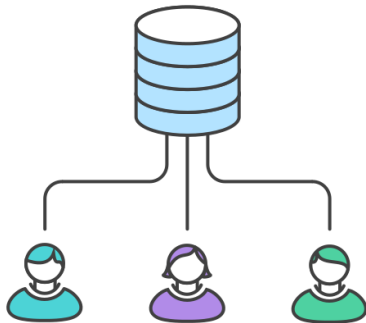
Let's try to be more accessible.

Git is a computer program/tool to save and download files on a hosting server (e.g. GitHub and GitLab).

Git is a distributed version control system that tracks changes in any set of computer files, usually used for coordinating work among programmers who are collaboratively developing source code during software development. Its goals include speed, data integrity, and support for distributed, non-linear workflows (thousands of parallel branches running on different computers). [Wiki]

Let's try to be more accessible.

Git is a computer program/tool to save and download files on a hosting server (e.g. GitHub and GitLab).

GIT

- Git facilitates users to track the various versions of files. It is not a necessary tool, but it can be very very helpful. Generally, the time spent to learn its syntax is well paid off

(do you remember to save some file like
manuscript_draft_v4.02_final_definitive_forreal_lastcomments_edite
Exactly! Imagine to do that for a repository of files...)

- It permits to save and share the intermediate stages of a work in progress (which software is complete and always up to date?) in an accessible, consistent and structured way, allowing an effective version tracking. It allows retrieval of previous working versions, limiting the risk to overwrite useful files.

GIT

- Git facilitates users to track the various versions of files. It is not a necessary tool, but it can be very very helpful. Generally, the time spent to learn its syntax is well paid off

(do you remember to save some file like *manuscript_draft_v4.02_final_definitive_forreal_lastcomments_edite* Exactly! Imagine to do that for a repository of files...)

- It permits to save and share the intermediate stages of a work in progress (which software is complete and always up to date?) in an accessible, consistent and structured way, allowing an effective version tracking. It allows retrieval of previous working versions, limiting the risk to overwrite useful files.

A distributed version control system

GIT

- Git facilitates users to track the various versions of files. It is not a necessary tool, but it can be very very helpful. Generally, the time spent to learn its syntax is well paid off

(do you remember to save some file like
manuscript_draft_v4.02_final_definitive_forreal_lastcomments_edited
Exactly! Imagine to do that for a repository of files...)

- It permits to save and share the intermediate stages of a work in progress (which software is complete and always up to date?) in an accessible, consistent and structured way, allowing an effective version tracking. It allows retrieval of previous working versions, limiting the risk to overwrite useful files.

The tool is particularly useful for programmers working in teams or in projects whose outcomes can be used by others.

- Git helps to co-develop a code, test its functions and the compatibility of the various code sections.

- A long list of further possibilities became possible by git.

- Different software integration on development platforms, based on git, will help you to develop and co-develop your code.

- The platform GitLab and GitHub have a large set of functionalities to further support code documentation and public releases.

- Files can be disclosed to the public, becoming a great integration of your CV, showing what you are able to do in an open and accessible way.

What is git actually for

The tool is particularly useful for programmers working in teams or in projects whose outcomes can be used by others.

- Git helps to co-develop a code, test its functions and the compatibility of the various code sections.
- A long list of further possibilities became possible by git.
- Different software integration on development platforms, based on git, will help you to develop and co-develop your code.
- The platform GitLab and GitHub have a large set of functionalities to further support code documentation and public releases.
- Files can be disclosed to the public, becoming a great integration of your CV, showing what you are able to do in an open and accessible way

The tool is particularly useful for programmers working in teams or in projects whose outcomes can be used by others.

- Git helps to co-develop a code, test its functions and the compatibility of the various code sections.
- A long list of further possibilities became possible by git.
- Different software integration on development platforms, based on git, will help you to develop and co-develop your code.
- The platform GitLab and GitHub have a large set of functionalities to further support code documentation and public releases.
- Files can be disclosed to the public, becoming a great integration of your CV, showing what you are able to do in an open and accessible way.

What is git actually for

The tool is particularly useful for programmers working in teams or in projects whose outcomes can be used by others.

- Git helps to co-develop a code, test its functions and the compatibility of the various code sections.
- A long list of further possibilities became possible by git.
- Different software integration on development platforms, based on git, will help you to develop and co-develop your code.
- The platform GitLab and GitHub have a large set of functionalities to further support code documentation and public releases.
- Files can be disclosed to the public, becoming a great integration of your CV, showing what you are able to do in an open and accessible way.

The tool is particularly useful for programmers working in teams or in projects whose outcomes can be used by others.
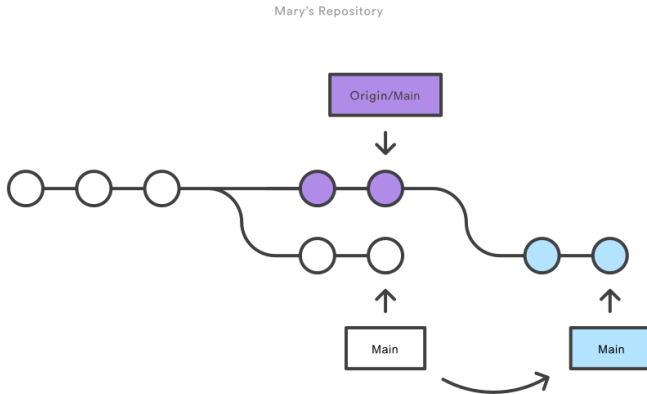
- Git helps to co-develop a code, test its functions and the compatibility of the various code sections.
- A long list of further possibilities became possible by git.
- Different software integration on development platforms, based on git, will help you to develop and co-develop your code.
- The platform GitLab and GitHub have a large set of functionalities to further support code documentation and public releases.
- Files can be disclosed to the public, becoming a great integration of your CV, showing what you are able to do in an open and accessible way.

The tool is particularly useful for programmers working in teams or in projects whose outcomes can be used by others.

- Git helps to co-develop a code, test its functions and the compatibility of the various code sections.
- A long list of further possibilities became possible by git.
- Different software integration on development platforms, based on git, will help you to develop and co-develop your code.
- The platform GitLab and GitHub have a large set of functionalities to further support code documentation and public releases.
- Files can be disclosed to the public, becoming a great integration of your CV, showing what you are able to do in an open and accessible way.

Mary's Repository

As the open libraries are exploding in numbers, you might need some criteria to assert the reliability of a project.

Unit test driven development!

That is taking full advantage of python object oriented structure.

Community

Good project are not only used by communities, but also **supported**

Git allows the development of projects without a clear lead.
Community engagement is generally a desirable target to help
develop to directly integrate feedbacks by users (and fix bugs).

As the open libraries are exploding in numbers, you might need some criteria to assert the reliability of a project.

Unit test driven development!

That is taking full advantage of python object oriented structure.

Community

Good project are not only used by communities, but also **supported**

Git allows the development of projects without a clear lead. Community engagement is generally a desirable target to help develop to directly integrate feedbacks by users (and fix bugs).