UNIVERSITÄT BERN

8. Managing a coding project

Athina Tzovara

University of Bern

Athina.Tzovara@unibe.ch

Today

1. Python functionalities (lambda functions *args / **kargs)
2. Working on a larger scale coding project
3. Designing a project roadmap, attracting contributors
4. Summary

Reminder: what we saw in Lecture 6: Transformations

With `.transform()` we can return a transformed version of our full dataset, to recombine

The output and input will have the same shape

Example: we can re-center the data to zero-mean:

```
df.groupby('key').transform(lambda x: x - x.mean())
```

	data1	data2
0	-1.5	1.0
1	-1.5	-3.5
2	-1.5	-3.0
3	1.5	-1.0
4	1.5	3.5
5	1.5	3.0

Imperative vs. Functional Programming

Imperative:

- Programming with statements
- Program flows step by step with detailed commands

- ”easy” to understand
- Debugging is straightforward

- Lengthy code
- Poor scalability
- Multi-threading is not straightforward

C ; Python

Functional:

- Mathematical logic
- lambda calculus philosophy
- Emphasis on abstraction composition and ‘purity’

- Less code
- Scales well
- Higher-order functions

- Slower for simple calculations
- Code readability is low
- Steep learning curve

Lisp, Haskell, Erlang

Ocaml;
Python with lambda functions

Example of lambda functions

Keyword `def`
`Argument: x`

```
def identity(x):
    return x
```

Identity function: returns its argument

```
lambda x: x
```

```
<function __main__.<lambda>(x)>
```

Equivalent with lambda function

Keyword `lambda`

`Bound variable: x` (argument to a lambda function)
`Body: x` (free variable)

Example of lambda functions

More elaborate lambda: a function that adds 1 to an argument:

```
lambda x: x + 1
```

```
<function __main__.<lambda>(x)>
```

```
(lambda x: x + 1)(2)
```

```
3
```

```
(lambda x: x)(1)
```

```
1
```

We apply the function to an argument

Example of lambda functions

More elaborate lambda: a function that adds 1 to an argument:

```
add_one = lambda x: x + 1
add_one(2)
```

```
3
```

We can name the lambda function
(it is an expression)

```
def add_one(x):
    return x + 1

add_one(2)
```

```
3
```

Equivalent function

Example of lambda functions

We can have multi-argument delta functions:

listing arguments, separating with a comma (,)

surrounding with parentheses:

```
course_name = lambda course, program: f'Course and Program: {course.title()+", "} {program.title()}'
```

```
course_name('advanced python', 'bioinformatics')
```

```
'Course and Program: Advanced Python,  Bioinformatics'
```

Input: two arguments

Output: one string

Multi-argument lambda functions

Another example of lambda functions:

```
(lambda x,y: x+y)(1,2)
```

```
3
```

```
adding = lambda x,y: x+y
adding(1,2)
```

```
3
```

Syntax for lambda functions

1. Only expressions, not statements in its body

```
(lambda x: assert x > 5)(2)
```

```
  File "<ipython-input-31-6103c644798f>", line 1
    (lambda x: assert x > 5)(2)
                     ^
SyntaxError: invalid syntax
```

Syntax for lambda functions

1. Only expressions, not statements in its body
2. Written as one single line of execution

```
(lambda x:
... (x % 2 and 'odd' or 'even'))(3)
```

```
'odd'
```

Syntax for lambda functions

1. Only expressions, not statements in its body
2. Written as one single line of execution
3. It does not support type annotations

```
lambda first: str, last: str: first.title() + " " + last.title()
```

```
  File "<ipython-input-34-0b78cd4a6455>", line 1
    lambda first: str, last: str: first.title() + " " + last.title()
                                ^
SyntaxError: invalid syntax
```

Syntax for lambda functions

1. Only expressions, not statements in its body
2. Written as one single line of execution
3. It does not support type annotations
4. It needs to be immediately invoked

```
(lambda x: x * x)(3)
```

```
9
```

Example 1 of lambda function

We want to sort items of a list of number pairs, according to the second element of each pair

```
mylist = [(1, 2), (2, 3), (1, 9), (31, 0)]
```

Our list

```
def sorting_factor(x):
    return x[1]

mylist.sort(key=sorting_factor)
mylist
```

```
[(31, 0), (1, 2), (2, 3), (1, 9)]
```

Solution with ‘traditional’ functions, imperative programming- like

```
mylist.sort(key=lambda x: x[1])
mylist
```

```
[(31, 0), (1, 2), (2, 3), (1, 9)]
```

Solution with lambda functions, functional programming- like

Example 2 of lambda function (last week)

With `.transform()` we can return a transformed version of our full dataset, to recombine

The output and input will have the same shape

Example: we can re-center the data to zero-mean:

```
df.groupby('key').transform(lambda x: x - x.mean())
```

	data1	data2
0	-1.5	1.0
1	-1.5	-3.5
2	-1.5	-3.0
3	1.5	-1.0
4	1.5	3.5
5	1.5	3.0

Functional features of python

Further reading:

http://python-history.blogspot.com/2009/04/origins-of-pythons-functional-features.html

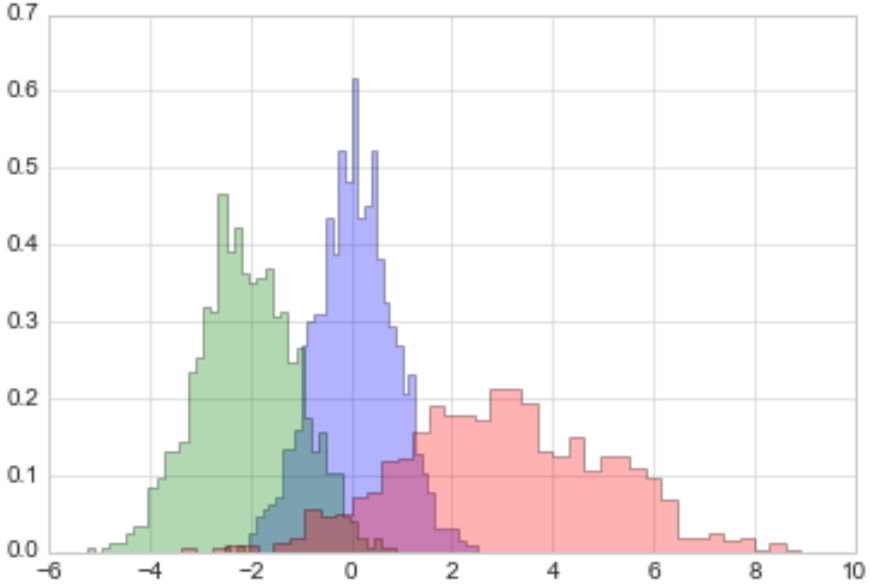
Reminder: what we encountered in Lecture 7: **kwargs

Visualization can be improved, e.g. with increasing transparency with `alpha`
Coding can also be improved, `**kwargs`

```
x1 = np.random.normal(0, 0.8, 1000)
x2 = np.random.normal(-2, 1, 1000)
x3 = np.random.normal(3, 2, 1000)

kwargs = dict(histtype='stepfilled', alpha=0.3, density=True, bins=40)

plt.hist(x1, **kwargs)
plt.hist(x2, **kwargs)
plt.hist(x3, **kwargs);
```

We use the same arguments for each histogram

Instead of repetition we can replace them with a dictionary and `**kwargs`



Extra: `*args` `**kargs`

The question:

How do we pass multiple arguments to a Python function?

The challenge:

Sometimes we need to pass multiple arguments in a function, where the number of arguments can only be determined at runtime

Extra: `*args` `**kargs`

Key question: How can we create a function that outputs a sentence from a list of words?

Option 1. We give as input a list that contains all our words:

```
def create_sentence(words_list):
    sentence = ""
    for word in words_list:
        sentence += word + " "
    return sentence

list_of_words = ["Hello", "world"]
print(create_sentence(list_of_words))
```

```
Hello world
```

Extra: `*args` `**kargs`

Key question: How can we create a function that outputs a sentence from a list of words?

Option 1 (intuitive). We give as input a list that contains all our words:
Drawback: we need to create a list every time we call the function.

```
def create_sentence(words_list):
    sentence = ""
    for word in words_list:
        sentence += word + " "
    return sentence

list_of_words = ["Hello", "world"]
print(create_sentence(list_of_words))
```

```
Hello world
```

Extra: `*args` `**kargs`

Key question: How can we create a function that outputs a sentence from a list of words?

Option 2 (the Python-way). We use `*args` which allows us to pass a varying number of positional arguments.
We are no longer passing a list in the function, but rather two separate arguments

```python
def create_sentence(*args):
    sentence = ""
    for word in args:
        sentence += word + " "
    return sentence

print(create_sentence("Hello", "world"))
```

*** : Unpacking operator**
It gives a tuple, not a list

```
Hello world
```

Extra: `*args` `**kargs`

Key question: How can we create a function that outputs a sentence from a list of words?

Option 2 (the Python-way). We use `*args` which allows us to pass a varying number of positional arguments.
We are no longer passing a list in the function, but rather two separate arguments
We can easily extend to as many arguments as we wish:

```
def create_sentence(*args):
    sentence = ""
    for word in args:
        sentence += word + " "
    return sentence

print(create_sentence("Hello", "world", "how", "are", "you"))
```

```
Hello world how are you
```

Extra: `*args` `**kargs`

Key question: How can we create a function that outputs a sentence from a list of words?

Option 3 (also the Python-way). We use `**kargs` which allows us to pass a keyword (named) arguments.

```
def create_sentence(**kargs):
    sentence = ""
    for word in kargs.values():
        sentence += word + " "
    return sentence

print(create_sentence(w1 = "Hello", w2 = "world"))
```

```
Hello world
```

If we iterate over the dictionary we need .values()

**** : Unpacking opera**
It gives a dictionar

Extra: `*args` `**kargs`

Key question: How can we create a function that outputs a sentence from a list of words?

Option 4 (also the Python-way). We can also mix positional and named arguments (and standard arguments):

```
def create_sentence(*args, **kargs):
    sentence = ""
    for word in kargs.values():
        sentence += word + " "
    return sentence

print(create_sentence(w1 = "Hello", w2 = "world"))
```

```
Hello world
```

Extra: `*args` `**kargs`

Key question: How can we create a function that outputs a sentence from a list of words?

Option 4 (also the Python-way). We can also mix positional and named arguments (and standard arguments):
However, `*args` need to precede `**kargs`:

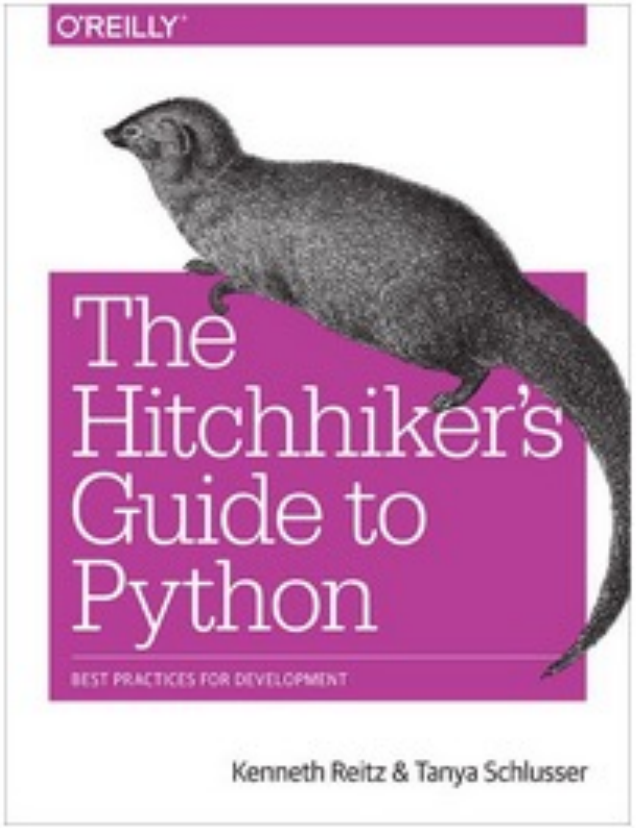
```
def create_sentence(**kargs, *args):
    sentence = ""
    for word in kargs.values():
        sentence += word + " "
    return sentence

print(create_sentence(w1 = "Hello", w2 = "world"))
```

```
  File "<ipython-input-23-66413c0f512f>", line 1
    def create_sentence(**kargs, *args):
                                 ^
SyntaxError: invalid syntax
```

Today

1. Python functionalities (lambda functions *args / **kargs)
2. **Working on a larger scale coding project**
3. Designing a project roadmap, attracting contributors
4. Summary

Main resources for today



https://mozilla.github.io/open-leadership-training-series/

Goals for coding project management

1. Acquire skills and background that can help you complement and improve coding of your Python project

2. Get some background on the open-source development of Python libraries or tools and ways to contribute

Managing a (coding) project

Source: www.shecodes.io/

Working on a larger scale project

1. How does our project best meet its objective?

2. Creating clean and effective code:
    1. clear logic and dependencies;
    2. clean organization of files and folders

3. Which functions should go into which modules?

4. How do our data flow through the project?

5. Which functions can be grouped together, or isolated?

Planning what the finished coding project will look like

Why do we need coding project management?

1. Structuring a project
2. Setting milestones
3. Breaking down the project to smaller tasks
4. Contributing and attracting contributors
5. Creating guidelines
6. Code of Conduct: what happens when things do not go as planned?

Very often: working on a repository

GitLab

Structure of a repository

Example repository for Python projects

Source: https://github.com/navdeep-G/samplemod

navdeep-G / samplemod Public template

Notificatic

Code | Pull requests 7 | Actions | Security | Insights

master | 1 branch | 0 tags | Go to file | Code

navdeep-G Update README.rst d469f2f on 20 Jul 2019 29 commits

docs	basics	10 years ago
sample	Lets allow the helpers to be helpfull	5 years ago
tests	Lets allow the helpers to be helpfull	5 years ago
.gitignore	add a Python gitignore	5 years ago
LICENSE	Update LICENSE	5 years ago
MANIFEST.in	need to include the LICENSE file, otherwise pypi installs ...	5 years ago
Makefile	Update Makefile	6 years ago
README.rst	Update README.rst	2 years ago
requirements.txt	basics	10 years ago
setup.py	Update setup.py	4 years ago

README.rst

Sample Module Repository

This simple project is an example repo for Python projects.

Learn more.

If you want to learn more about `setup.py` files, check out this repository.

Structure of a repository

Project name

Folders and files

Project description

navdeep-G / samplemod Public template

Notificatic

Code | Pull requests 7 | Actions | Security | Insights

master | 1 branch | 0 tags | Go to file | Code

navdeep-G Update README.rst d469f2f on 20 Jul 2019 29 commits

docs	basics	10 years ago
sample	Lets allow the helpers to be helpfull	5 years ago
tests	Lets allow the helpers to be helpfull	5 years ago
.gitignore	add a Python gitignore	5 years ago
LICENSE	Update LICENSE	5 years ago
MANIFEST.in	need to include the LICENSE file, otherwise pypi installs ...	5 years ago
Makefile	Update Makefile	6 years ago
README.rst	Update README.rst	2 years ago
requirements.txt	basics	10 years ago
setup.py	Update setup.py	4 years ago

README.rst

Sample Module Repository

This simple project is an example repo for Python projects.

Learn more.

If you want to learn more about `setup.py` files, check out this repository.

Structure of a repository

navdeep-G / samplemod Public template

Notificatic

Code | Pull requests 7 | Actions | Security | Insights

master | 1 branch | 0 tags | Go to file | Code

navdeep-G Update README.rst d469f2f on 20 Jul 2019 29 commits

docs	basics	10 years ago
sample	Lets allow the helpers to be helpfull	5 years ago
tests	Lets allow the helpers to be helpfull	5 years ago
.gitignore	add a Python gitignore	5 years ago
LICENSE	Update LICENSE	5 years ago
MANIFEST.in	need to include the LICENSE file, otherwise pypi installs ...	5 years ago
Makefile	Update Makefile	6 years ago
README.rst	Update README.rst	2 years ago
requirements.txt	basics	10 years ago
setup.py	Update setup.py	4 years ago

Documentation → docs

Testing → tests

License → LICENSE

README → README.rst

Requirements → requirements.txt

README.rst

Sample Module Repository

This simple project is an example repo for Python projects.

Learn more.

If you want to learn more about `setup.py` files, check out this repository.

Structuring open source projects

Some parts of a repository are particularly relevant for open source projects:

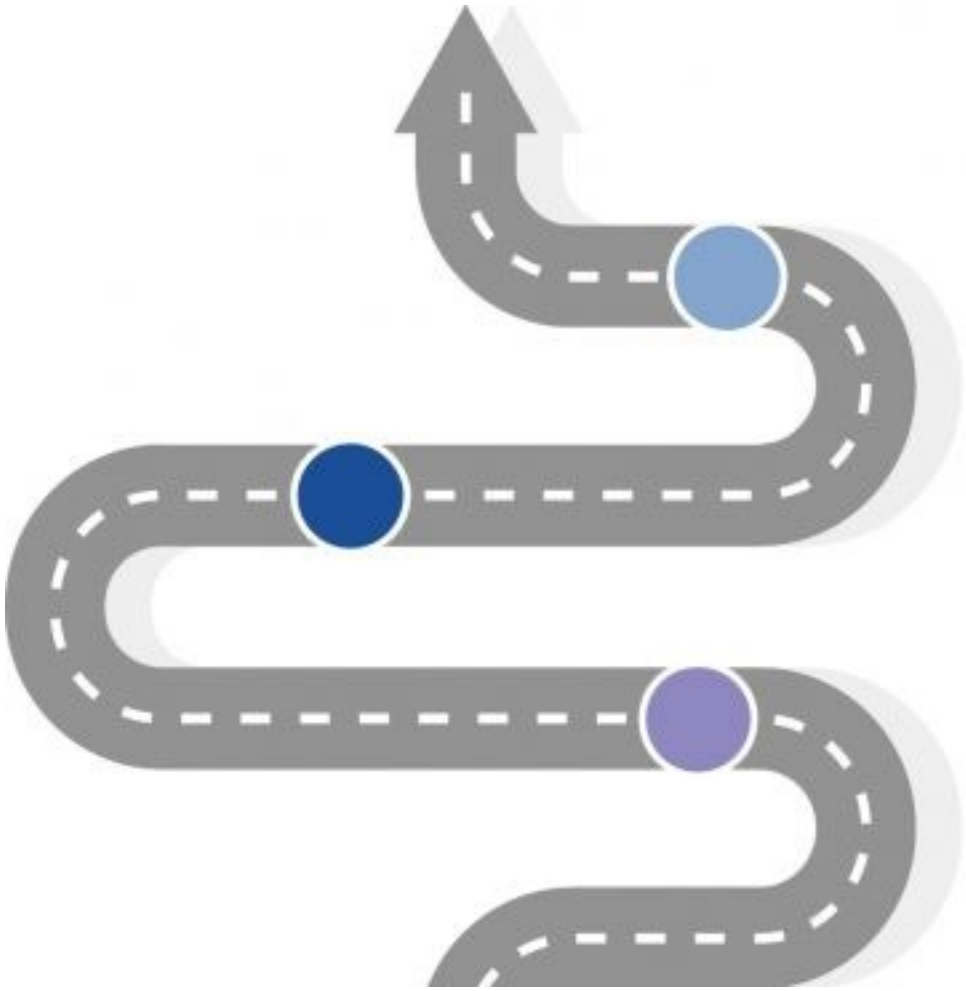
1. Roadmap
2. README
3. License
4. Contributing guidelines

Today

1. Python functionalities (lambda functions *args / **kargs)
2. Working on a larger scale coding project
3. **Designing a project roadmap, attracting contributors**
4. Summary

Roadmap: what goes in a roadmap

1. Project Summary & Welcome
2. How to Get Involved
3. Timeline



1. Project summary and welcome

Welcome and orient visitors to your project. Users may have been linked directly to the Roadmap, so it's important to help them understand where they are.

2. How to get involved

New contributors might want to jump in right away

A roadmap can point them to parts of the project they can immediately work on, and important documentation they should check out (COC, CONTRIBUTING.md).

3. Timeline

The start of the roadmap:

This section organizes **tasks** needed to complete your project around **milestones**, mapping out what you're working on now and where it's going next.

3. Timeline - Milestones

Project milestones are significant turning points or events that will move the project forward.

- Project status goals (feature release, minimum viable project)
- Dates / Events (Presentations, Release, Exams, etc)
- Timeframes (short, medium, long term)

3. Timeline - Tasks

List tasks to complete for each milestone. Info you can include with each task to make it easy for newcomers to get involved:

- What needs to be done
- What does success look like
- Pointers to get started
- Why this task is important – link to your project goals

Example: Github RoadMap

alexcnichols Update footnote in README 678c54a 19 days ago **44** commits

.github/ISSUE_TEMPLATE	Direct all feedback to public feedback discussions	25 days ago
CODE_OF_CONDUCT.md	Create CODE_OF_CONDUCT.md	2 years ago
LICENSE	Create LICENSE	2 years ago
README.md	Update footnote in README	19 days ago
SECURITY.md	Create SECURITY.md	2 years ago

README.md

GitHub public roadmap

View the official GitHub public product roadmap[1]

Our product roadmap is where you can learn about what features we're working on, what stage they're in, and when we expect to bring them to you. Have any questions or comments about items on the roadmap? Share your feedback via GitHub public feedback discussions.

The roadmap repository is for communicating GitHub's roadmap. Existing issues are currently read-only, and we are locking conversations, as we get started. Interaction limits are also in place to ensure issues originate from GitHub. We're planning to iterate on the format of the roadmap itself, and we see potential to engage more in discussions about the future of GitHub products and features. If you have feedback about this roadmap repository itself, such as how the issues are presented, let us know through general feedback in GitHub public feedback discussions.

Hands-on: Designing a ROADMAP for a coding project

Project: Program a software that will gather the grades for the Spring Semester 2024 exams, that can be used by Swiss Universities

- Write a **mission** and **summary** for your project: start with a **name** for your project
- Outline your **milestones**: **what** needs to be done for your project, and **when**?
- Provide a short list of **tasks for each milestone** that are required to successfully complete the project work on a given milestone.
- Note down any relevant **events** and **timeframes**

Why do READMEs matter?

What is a README?

Found in the root directory of your repository

In ALL CAPS, a request for all to “READ ME”

First stop for visitors

Could also be: website landing page, event description

In your README show:

- what you're doing, for whom, and why
- what makes your project special
- how to get started
- where to find key resources

A closer look at a Python README

NumPy

powered by NumFOCUS | Pypi downloads 93M/month | Conda downloads 26M | stackoverflow Ask questions | DOI 10.1038/s41592-019-0686-2

NumPy is the fundamental package needed 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

It provides:

- a powerful N-dimensional array object
- sophisticated (broadcasting) functions
- tools for integrating C/C++ and Fortran code
- useful linear algebra, Fourier transform, and random number capabilities

Testing:

NumPy requires `pytest`. Tests can then be run after installation with:

```
python -c 'import numpy; numpy.test()'
```

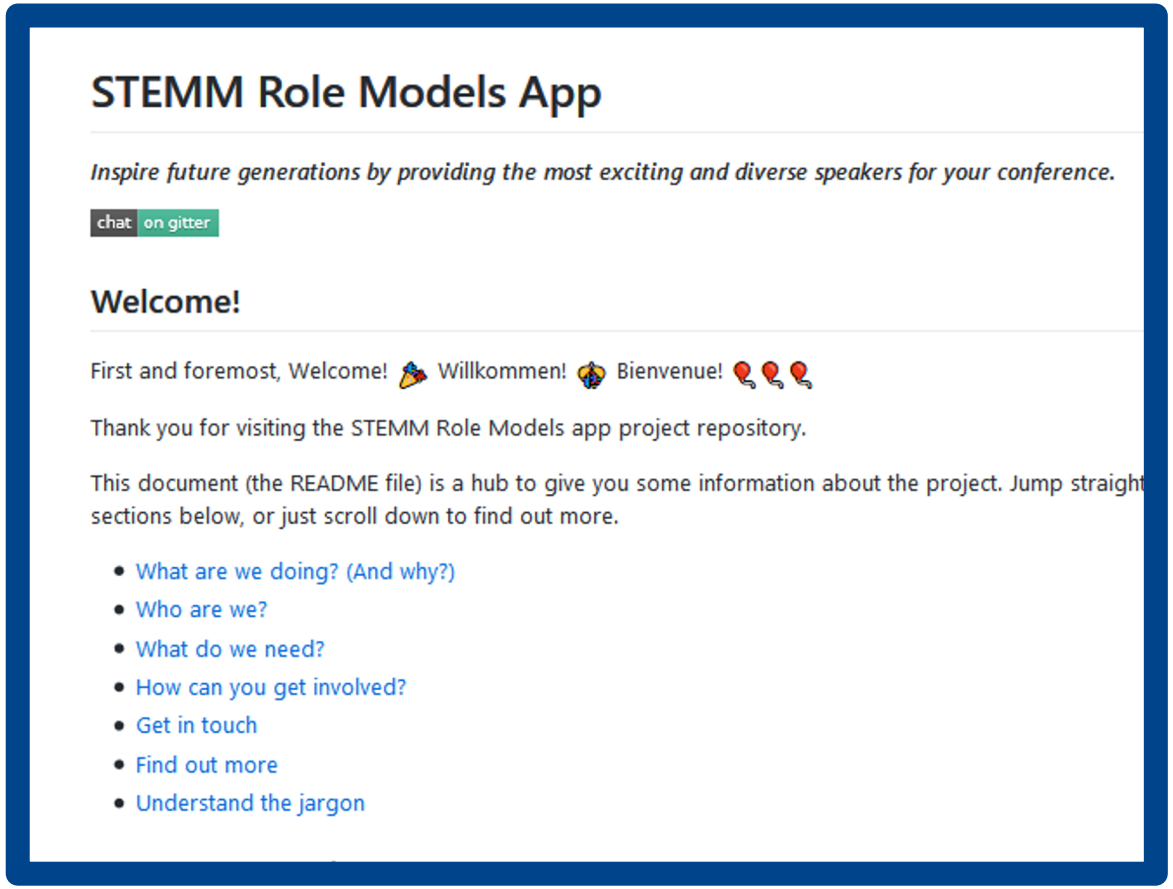
Project description & vision

Links to:

- How to contribute & get involved
- Bug reports
- Source code

README example: Numpy

A closer look at a second README



STEMM Role Models App

Inspire future generations by providing the most exciting and diverse speakers for your conference.

chat on gitter

Welcome!

First and foremost, Welcome! Willkommen! Bienvenue!

Thank you for visiting the STEMM Role Models app project repository.

This document (the README file) is a hub to give you some information about the project. Jump straight
sections below, or just scroll down to find out more.

- What are we doing? (And why?)
- Who are we?
- What do we need?
- How can you get involved?
- Get in touch
- Find out more
- Understand the jargon

Welcome message

Project description & vision

Links to:

- How to contribute & get involved
- License
- Code of Conduct, reporting info

README example: STEMM Role Models App

Hands-on: Designing a README for a coding project

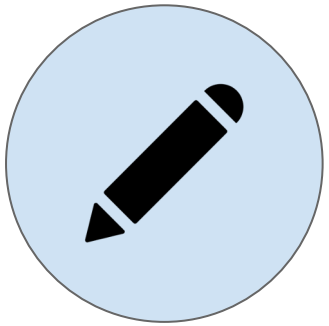
Project: Program a software that will gather the grades for the Spring Semester 2024 exams, that can be used by Swiss Universities

- Project description: WHAT are you doing; WHO are your collaborators; what are your goals with this coding project.
- Explain WHY your project is useful and for which audience.

Open Licenses: common elements

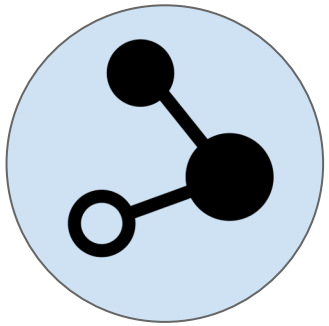
use

Anyone can use the work for any purpose



modify

Anyone can modify the work



share

Anyone can redistribute both the original and modified work to anyone else with the same license

"Open source software is software that can be freely used, modified, and shared (in both modified and unmodified form) by anyone. "

- GitHub Glossary, Open Source

Attribution

Most open licenses require others to credit the authors or copyright holders of the work.

Examples: CC BY (and almost all other licenses): retaining copyright granting others a license to use it

No attribution: CC0 (public domain, **no copyright holder**): waiving copyright in the work and placing it in the public domain; anybody can use it for any purpose, with or without attribution

Non-copyleft

(permissive, non-reciprocal)

An open license that **does not require** derivative works to shared with the same license.

Examples:
CC BY
MIT, BSD, APL-2.0

Copyleft

(reciprocal, viral)

An open license that **requires** all derivative works to be shared with the same license.

Examples:
CC BY-SA
GPLv3, MPL-2.0

Patent Clause

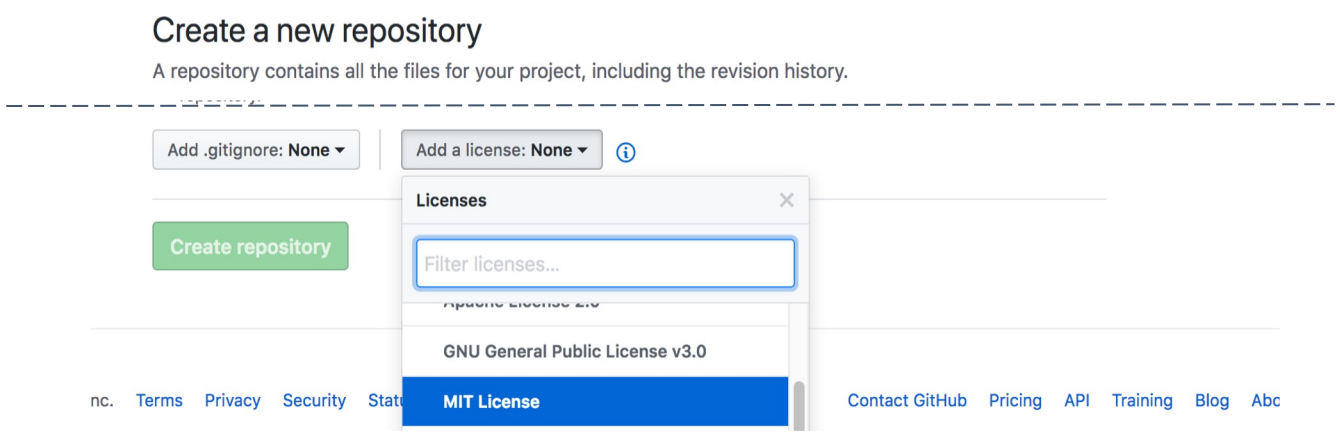
Most modern open source software licenses contain a clause designed to prevent people from using patent law to take away open source rights.

Examples: MPL-2.0, APL-2.0, GPLv3

Older licenses don’t have this clause: MIT, BSD

How to apply a license in your coding project

Place the full text of the license in a text file (usually named LICENSE) in the root directory. GitHub can generate certain software licenses.



Specific instructions on individual licenses: choosealicense.com, creativecommons.org.

You can include multiple licenses (ie one for software, one of content) as long as you are explicit about which license applies to which parts of your work. You can do this in the License section in your README.

License types

Software (choosealicense.com)

	non-copyleft	copyleft
No patent clause	BSD, MIT	
Patent snapback	APL- 2.0	GPLv3, MPL-2.0

Content (creativecommons.org): CC0, CC BY, CC BY-SA

	non-copyleft	copyleft
No attribution	CC0	
Attribution	CC BY	CC BY-SA

Data: CC0

Further Reading on Licenses

- The Open Source Definition (10 Criteria) | https://opensource.org/osd
- Legal Matters | producingoss.com

- Software: Choose an Open Source License | choosealicense.com
- Content: Choose a License | creativecommons.org

Hands-on: Choose a license

Project: Program a software that will gather the grades for the Spring Semester 2024 exams, that can be used by Swiss Universities

- Go to : https://choosealicense.com/ and pick a software license for your project!
- Which one did you choose and why?

Attracting coding contributors

- Document your project & process
- README, Contributing.md, CODE_OF_CONDUCT.md, LICENSE
- Labeling issues appropriately

Good first issue

keras-team / keras

★ 41k

Deep Learning for humans

deep-learning | tensorflow | neural-networks | machine-learning | data-science | python

Good first issues

- Convert the docstrings of examples in `examples/*` to use Markdown formatting — 11
  Good first issue | type:docs | stat:contributions welcome
  #12219 opened 3 months ago by gabrieldemarmiesse
- Implement K.arange for the cntk backend. Good first issue — 1
  stat:contributions welcome
  #12164 opened 3 months ago by gabrieldemarmiesse
- implement K.cumsum and K.cumprod for the cntk backend. Good first issue — 1
  stat:contributions welcome
  #12163 opened 3 months ago by gabrieldemarmiesse

Python Updated 3 days ago

CONTRIBUTING: Example

Welcoming contributors

Code of Conduct

Short version

Rationale of project

Different ways of contributing

Community style guidelines

Contributing to Atom

👍🎉 First off, thanks for taking the time to contribute! 🎉👍

The following is a set of guidelines for contributing to Atom and its packages, which are hosted in the Atom Organization on GitHub. These are mostly guidelines, not rules. Use your best judgment, and feel free to propose changes to this document in a pull request.

Table Of Contents

CONTRIBUTING file

WHY?

- structure contributions
- provide guidelines
- document style
- improve efficiency

WHO?

- project owners
- project contributors
- project consumers

CONTRIBUTING: Example

CODE_OF_CONDUCT.md	📝 Update Contributing.md (#20707)	10 months ago
CONTRIBUTING.md	Update Discuss links to Github Discussions links	2 months ago
Dockerfile	Update Dockerfile (#20845)	10 months ago
LICENSE.md	Update LICENSE.md (#21997)	9 months ago
PULL_REQUEST_TEMPLAT...	Apply suggestions from code review	2 years ago
README.md	Remove dependancy status badge.	last month
SUPPORT.md	Update Discuss links to Github Discussions links	2 months ago
atom.sh	Merge pull request #13414 from passeride/master	2 months ago
coffeelint.json	🔥 Remove newlines_after_classes rule	7 years ago
package-lock.json	⬆️ Bump language-css@0.45.1	2 months ago
package.json	1.61.0-dev	7 days ago
stylelint.config.js	Reformat all JS files using prettier	3 years ago

README.md

Atom

Azure Pipelines succeeded

Atom is a hackable text editor for the 21st century, built on Electron, and based on everything we love about our favorite editors. We designed it to be deeply customizable, but still approachable using the default configuration.

Source: https://github.com/atom/atom/

Branch: master atom / CONTRIBUTING.md Find file Copy path

zeke remove links to Contributor Covenant CoC d81e2a2 on Dec 3, 2015

26 contributors

524 lines (418 sloc) 45.8 KB Raw Blame History

Contributing to Atom

👍🎉 First off, thanks for taking the time to contribute! 🎉👍

The following is a set of guidelines for contributing to Atom and its packages, which are hosted in the Atom Organization on GitHub. These are just guidelines, not rules, use your best judgment and feel free to propose changes to this document in a pull request.

Table Of Contents

HOW?

- location
- cheer
- introduction
- background
- how-to
- style

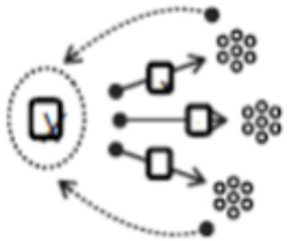
Hands-on: Designing a contributing file for a coding project

Project: Program a software that will gather the grades for the Spring Semester 2024 exams, that can be used by Swiss Universities

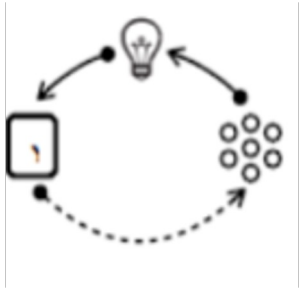
- List important resources for your project like your readme or roadmap files
- Explain how contributors can submit changes (relevant for the next homework once you have a repository)
- Describe good first tasks / bugs for new contributors
- Tell readers where they can find help, what is the process of getting in touch with you

Interactions with contributors / project community

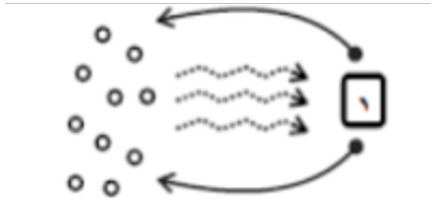
Gifting



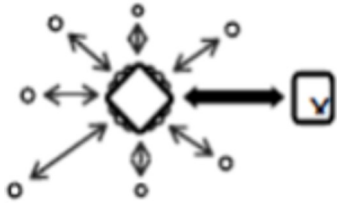
Soliciting Ideas



Learning Through Use



Creating Together



Networking Common Interests

Giving

Listening

Collaborating

A Framework of Open Practices
by Mozilla Open Innovation & the Copenhagen Institute for Interaction Design

What is a project community?

Project code of conduct

a set of rules outlining the social norms, rules, & responsibilities of an individual project, party or organization.

Project code of conduct

a set of rules outlining the social norms, rules, & responsibilities of an individual project, party or organization.

Do coding projects need a code of conduct?

- *Sets clear expectations for all contributors*
- *Sets guidelines for what to do in case of conflicts*

Example: Contributor Covenant

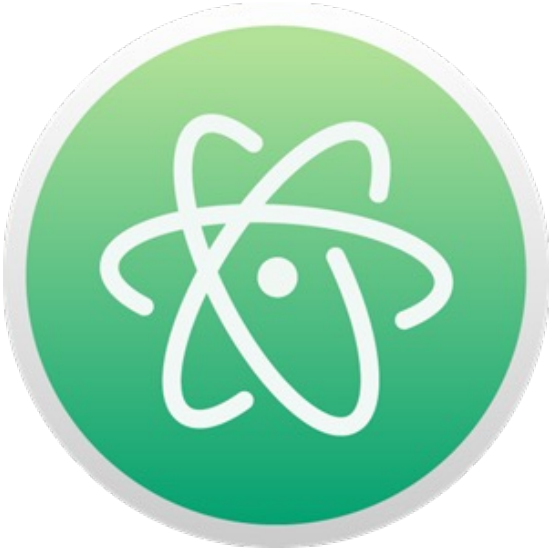
Contributor Covenant

Home Adopters Latest Version Translations FAQ

CONTRIBUTOR COVENANT CODE OF CONDUCT

Source: https://www.contributor-covenant.org/version/1/4/code-of-conduct/

Example: Contributor Covenant



Code of Conduct

This project and everyone participating in it is governed by the Atom Code of Conduct. By participating, you are expected to uphold this code. Please report unacceptable behavior to atom@github.com.

Source: https://github.com/atom/atom/blob/master/CONTRIBUTING.md#code-of-conduct

Further reading

The Hitchhiker's guide to Python
https://docs.python-guide.org/writing/structure/

Best practices in 'working open', Mozilla Open Leadership Training Series
https://mozilla.github.io/open-leadership-training-series/

Additional reading on Licenses:
Open Source Initiative: https://opensource.org/osd
Producing Open Source Software: https://producingoss.com/
Choose an open source license for software: https://choosealicense.com/
Open source licenses for content: https://creativecommons.org/

Today

1. Python functionalities (lambda functions)
2. Working on a larger scale coding project
3. Designing a project roadmap, attracting contributors
4. **Summary**
5. **Github: Next week!**