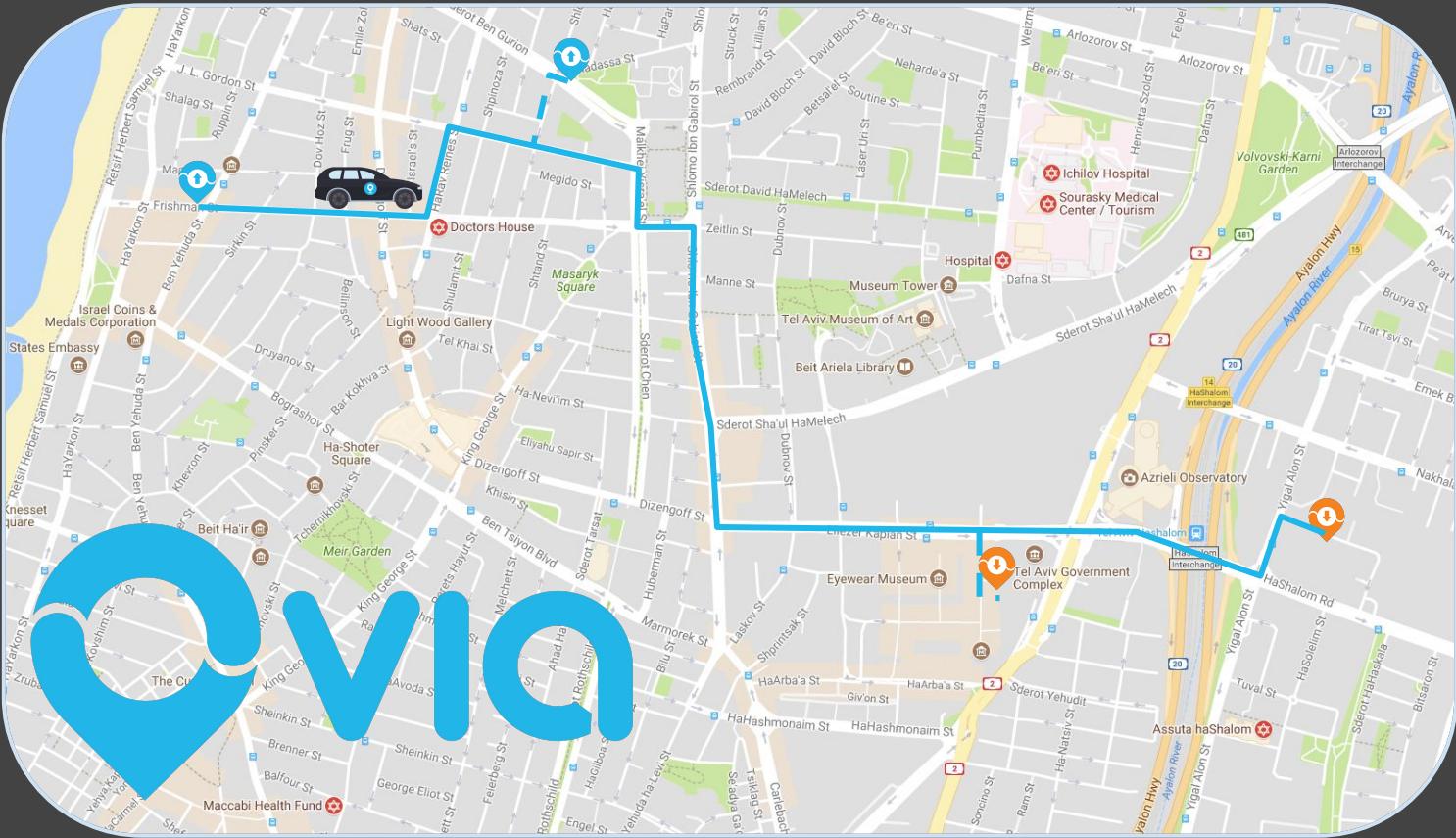


Help your colleagues help themselves - a Sphinx tutorial

Dalya Gartzman

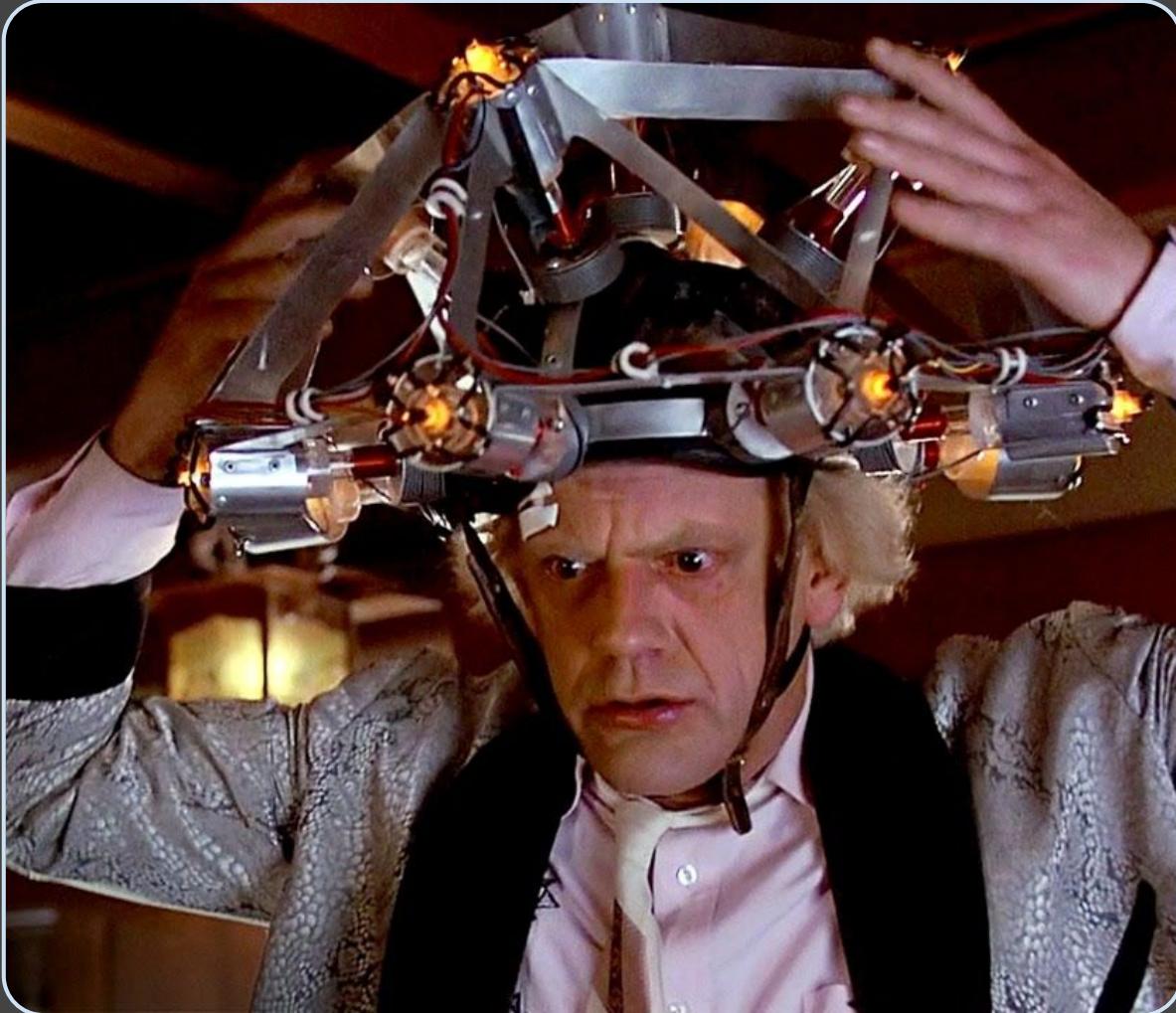
github.com/DalyaG/Sphinx185







L
HTOA - S
CRIA - C
MTIA - X1
MT2A - X2
CRIA - BI
CPIA - BO
CPJA - PU
TYIA - TY
CRIA - SI
LPIA - LC
MT2A - GO



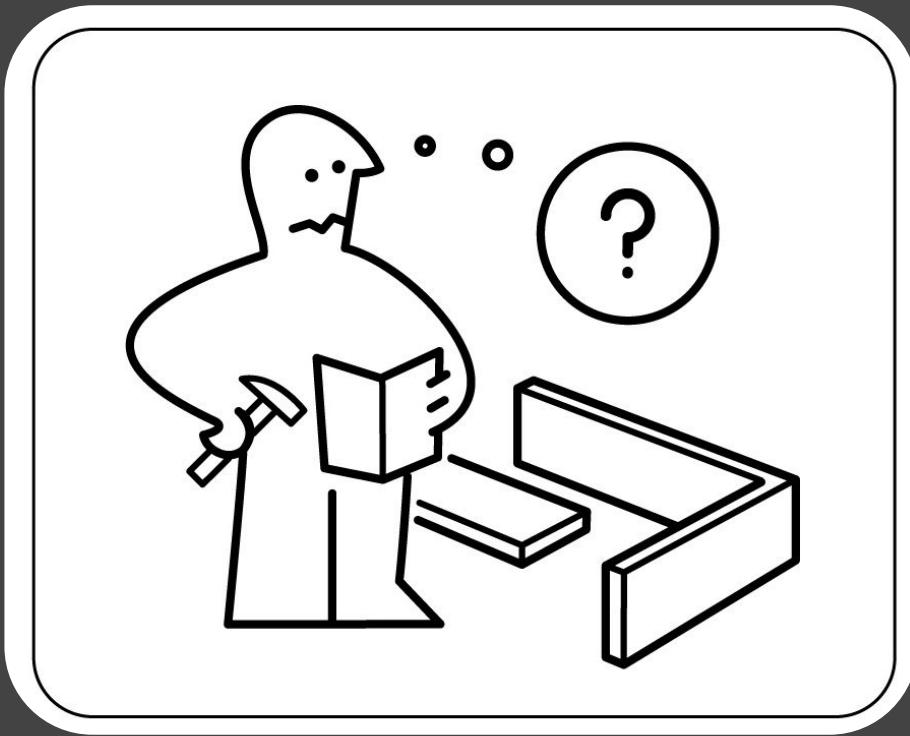




PUBLISH

PUBLISH EVERYTHING







NEWBIES

YOU HAD ONE JOB





READY PLAYER ONE

Press Start

READY PLAYER ONE

Press Start



Intro - What will we accomplish today?

Intro - What will we accomplish today?

Sphinx185 documentation » [next](#) | [modules](#) | [index](#)



Sphinx185

- [Euler's 185th Riddle](#)
- [Solver Runner](#)
- [Input Parser](#)
- [Naming Utils](#)
- [The ILP Manager class](#)
- [The main configurations file: conf.py](#)
- [How To Use This for YOUR Next Project :\)](#)

Quick search [Go](#)

Welcome to Sphinx185's documentation!

This documentation follows the solution of the 185th riddle from [Project Euler](#).
The goal of this project is to help make [Sphinx](#) accessible and easy to use.
To use this for YOUR next project, browse this documentation and follow the instructions [here](#).
Feel as free as you can to copy, paste and change anything you see here.

The source code is available [here](#).
[Hope you find this useful!](#)
[DalyaG](#)

OK Let's get to business:

- [Euler's 185th Riddle](#)
- [Solver Runner](#)
- [Input Parser](#)
- [Naming Utils](#)
- [The ILP Manager class](#)
- [The main configurations file: conf.py](#)
- [How To Use This for YOUR Next Project :\)](#)
- [Index](#)

Part I - Overview

Intro - What will we accomplish today?

Sphinx185 documentation »



Sphinx185

- Euler's 185th Riddle
- Solver Runner
- Input Parser
- Naming Utils
- The ILP Manager class
- The main configurations file: conf.py

Quick search

Go

The main configurations file: conf.py

Below is the code for `conf.py`, the main configuration file that is used for generating this documentation.

This file was originally made by running `sphinx-quickstart`.

It was modified and customized such that, hopefully, it has a clearer structure, and is easier to re-modify for the sake

```
"""
This is the main file in which the configuration for the documentation is made.

"""
# -*- coding: utf-8 -*-
#
# Sphinx185 documentation build configuration file, created by
# sphinx-quickstart on Sat May 12 19:34:31 2018.
#
# DalyaG: This file was heavily modified from its original build.
# Hope you find this useful :)

# -- Define here your working directory

import os
import sys
sys.path.append(os.path.abspath('/Users/dalya/Documents/Sphinx185'))

# -- Some general info about the project

project = u'Sphinx185'
copyright = u'2018, DalyaG'
author = u'DalyaG'

# -- A few basic configurations

# The documentation in this project will be mostly generated from .rst files
# In this project, every auto-documented module/class has its own .rst file, under the main documentat
# which is rendered into an .html page.
# Get more information here: http://www.sphinx-doc.org/en/master/usage/restructuredtext/basics.html
source_suffix = ['.rst']
```

Part II - Main configuration file - conf.py

Intro - What will we accomplish today?

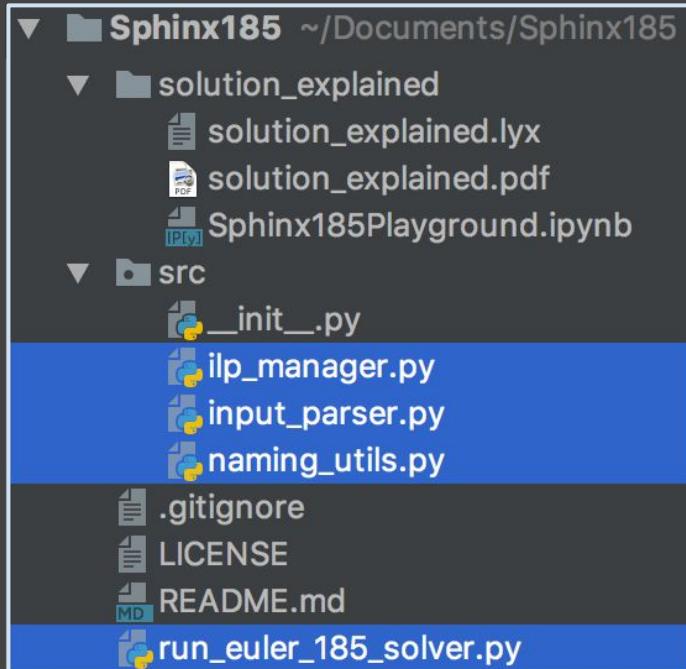
How To Use This for YOUR Next Project :)

1. `pip install sphinx`
2. Copy the `documentation_template_for_your_next_project` folder in this repository, make it a subdirectory in your local project and rename it `documentation`.
3. Edit `conf.py` by searching the pattern `#CHNAGEME#` inside the file and follow the instructions.
4. Edit `index.rst` by following the inline instructions.
5. To add a page to your documentation:
 - a. Create a `.rst` file for the function/module you wish to document (you can use the templates supplied here), and
 - b. Add the name of the `.rst` file to the `toctree` in `index.rst`.
6. In terminal, inside the `documentation` folder, run `make html`.
(TIP OF THE WEEK: actually, always run `make clean html` to clear sphinx cache and build from scratch)
7. To view locally - open the file `documentation/_build/html/index.html` in your browser, and enjoy the read :)
8. To share - you can use [GitHub Pages](#) to host your documentation:
 - a. Copy the content of `documentation/_build/html/` into a new `docs` folder, under the root of the project.
 - b. Create an empty file `.nojekyll` inside `docs` folder
(this tells GitHub Pages to bypass the default `jekyll` themes and use the `html` and `css` in your project)
 - c. Push your changes to master branch.
 - d. In your repository on GitHub, go to “Settings” -> “GitHub Pages” -> “Source” and select “master branch /docs folder”.
 - e. Share your beautiful documentation site at https://<your_git_usrname>.github.io/<project_name>/

Part III - Document YOUR next project

Part I - Overview

Part I - Overview



Part I - Overview

38 commits 1 branch 0 releases 1 contributor MIT

Branch: master ▾ New pull request Create new file Upload files Find file Clone or download ▾

 DalyaG	updated readme	Latest commit 67ca419 2 days ago
 data	maybe last update	2 days ago
 docs	updated docs	2 days ago
 documentation	updated ip	2 days ago
 documentation_template_for_your_next_project	updated css	2 days ago
 solution_explained	Documentation is ready	8 days ago
 src	Documentation is ready	8 days ago
 .gitignore	Documentation is ready	8 days ago
 HelpYourColleaguesHelpThemselves_aSphinxTutorial.pdf	maybe last update	2 days ago
 LICENSE	Initial commit	2 months ago
 README.md	updated readme	2 days ago
 run_euler_185_solver.py	Documentation is ready	8 days ago

Part I - Overview

38 commits 1 branch 0 releases 1 contributor MIT

Branch: master Sphinx185 / run_euler_185_solver.py Find file Copy path

Branch: master DalyaG Documentation is ready 10ae6b1 8 days ago

DalyaG 1 contributor

data 36 lines (22 sloc) | 1.02 KB Raw Blame History

```
1 from optparse import OptionParser
2
3 from src.ilp_manager import ILPManager
4 from src.input_parser import input_parser
5
6
7
8 def main():
9     """
10     Run this code to get the solution to 185th problem in Project Euler.
11
12     Terminal options:
13
14     .. option:: -m <sequence_length>, --sequence_length <sequence_length>
15
16         Length of sequence in the riddle. Assume file 'data/input_m.txt' exists.
17
18         """
19
```

Part I - Overview

38 commits 1 branch 0 releases 1 contributor MIT

Branch: master Sphinx185 / run_euler_185_solver.py Find file Copy path

Branch: master Sphinx185 / src / ilp_manager.py Find file Copy path

DalyaG Documentation is ready 10ae6b1 8 days ago

1 contributor

data 36 lines (22)

docs

documenter

solution_

src

.gitignore

HelpYour

LICENSE

README

run_euler_185_solver.py

1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21

```
from pulp import
from itertools import product
from naming_utils import s_star_index_node, index_color_node

class ILPManager(object):
    """Model the 185th problem in Project Euler as an ILP (Integer Linear Program)"""
    def __init__(self, m):
        """To instantiate this module, please specify the length on sequences."""
        self.m = m
        self.n_digits = 10
        self.optimal_status = 'Optimal'
        self.active_edge = 1.0

    def build_ilp_solver(self, sequences_list, n_correct_vertices_list):
        """
        Given input sequences, and number or correct vertices in each of them,
        build an ILP representation of the problem.
        """

Raw Blame History
```

Part I - Overview

Sphinx185 documentation >



Sphinx185

- Euler's 185th Riddle
- Solver Runner
- Input Parser
- Naming Utils
- The ILP Manager class
- The main configurations file: conf.py

Quick search

Go

Welcome to Sphinx185's documentation!

This documentation follows the solution of the 185th riddle from [Project Euler](#).

The goal of this piece is to help make [Sphinx](#) accessible and easy to use.

Feel as free as you can to copy, paste and change anything you see here.

The source code is available [here](#).

Hope you find this useful!

@DalyaG

OK Let's get to business:

- Euler's 185th Riddle
- Solver Runner
- Input Parser
- Naming Utils
- The ILP Manager class
- The main configurations file: conf.py
- [Index](#)

dalyag.github.io/Sphinx185

Part I - Overview

Sphinx185 documentation >



Sphinx185

- Euler's 185th Riddle
- Solver Runner
- Input Parser
- Naming Utils
- The ILP Manager
- The main config file: conf.py

Quick search

Solver Runner

`run_euler_185_solver.main()` [\[SOURCE\]](#)

Run this code to get the solution to 185th problem in Project Euler.

Terminal options:

`-m <sequence_length>, --sequence_length <sequence_length>`

Length of sequence in the riddle. Assume file ‘data/input_m.txt’ exists.

Part I - Overview

Sphinx185 documentation >



Sphinx185

- Euler's 185th R
- Solver Runner
- Input Parser
- Naming Utils
- The ILP Manager
- The main config file: conf.py

Quick search

Solver Runner

Class API

run

Run

Term

`class src.ilp_manager.ILPManager(m)` [\[SOURCE\]](#)

Model the 185th problem in Project Euler as an ILP (Integer Linear Program)

To instantiate this module, please specify the length on sequences.

`build_ilp_solver(sequences_list, n_correct_vertices_list)` [\[SOURCE\]](#)

Given input sequences, and number or correct vertices in each of them, build an ILP representation of the

Parameters:

- `sequences_list` – List of sequences, each of length self.m, of integers between 0 a
- `n_correct_vertices_list` – Number of correct vertices in each sequence in sequenc equal to the color of the corresponding vertex in the solution s_star.

Returns:

tuple, containing:

- `ilp_solver`: Pulp instance, holding all the information needed for the solution.
- `s_star_to_color_edges`: The edges (variables) in the `ilp_solver`.

Part I - Overview

Euler's 185th Riddle

In case you are interested, the riddle goes something like this:

Let s^* be a sequence of m **unknown** digits (colors from 0 to 9).

We are given n sequences of m **known** colors, $x_{i,j}$ marks the color of the j^{th} index in sequence i .

For each sequence we are given the number of **correct colors**, that is, indices in the sequence for which $x_{i,j} = s_j^*$.

Our goal is to discover the colors of s^* , which we are promised are unique, given the input.

Part I - Overview

Euler's 185th Riddle

In case you are interested, the riddle goes something like this:

Let s^* be a sequence of m **unknown** digits (colors from 0 to 9).

We are given n sequences of m **known** colors, $x_{i,j}$ marks the i -th

For each sequence we are given the number of **correct colors**, t_i .

Our goal is to discover the colors of s^* , which we are promised a

Solver Runner

`run_euler_185_solver.main()` [\[SOURCE\]](#)

Run this code to get the solution to 185th problem in Project Euler.

Terminal options:

`-m <sequence_length>, --sequence_length <sequence_length>`

Length of sequence in the riddle. Assume file 'data/input_m.txt' exists.

Part I - Overview

Euler's 185th Riddle

In case you are interested:

Let s^* be a sequence of length m .
We are given m .
For each sequence s of length m ,
Our goal is to determine if s is a solution to the riddle.

Solver Runner

Source code for run_euler_185_solver

```
from optparse import OptionParser
from src.ilp_manager import ILPManager
from src.input_parser import input_parser

[docs]def main():
    """
    Run this code to get the solution to 185th problem in Project Euler.

    Terminal options:
    .. option:: -m <sequence_length>, --sequence_length <sequence_length>
        Length of sequence in the riddle. Assume file 'data/input_m.txt' exists.

    """
    parser = OptionParser()
    parser.add_option("-m", "--sequence_length",
                      help="Length of sequence in the riddle. Assume file 'data/input_m.txt' exists.")
    options, _ = parser.parse_args()

    m = int(options.sequence_length)
    sequences_list, n_correct_vertices_list = input_parser(m)

    ilp_manager = ILPManager(m)
    ilp_solver, s_star_to_color_edges = ilp_manager.build_ilp_solver(sequences_list, n_correct_vertices_list)
    s_star = ilp_manager.solve_ilp(ilp_solver, s_star_to_color_edges)

    print "Solution is: {}".format('.join([str(i) for i in s_star]))

if __name__ == '__main__':
    main()
```

[solver.main\(\) \[SOURCE\]](#)

Get the solution to 185th problem in Project Euler.

`-m <sequence_length>, --sequence_length <sequence_length>`
Length of sequence in the riddle. Assume file 'data/input_m.txt' exists.

Part I - Overview

Euler's 185th Riddle

In case you are interested:

Let s^* be a sequence.

We are given 10 sequences.

For each sequence we are given a color.

Our goal is to find s^* .

Source code for run_euler_185_solver

```
from optparse import OptionParser
from src.ilp_manager import ILPManager
from src.input_parser import input_parser

[docs]def main():
    """
    Run this code to get the solution to 185th problem in Project Euler.

    Terminal options:
    .. option:: -m <sequence_length>, --sequence_length <sequence_length>
        Length of sequence in the riddle. Assume file 'data/input_m.txt' exists.

    ...
    parser = OptionParser()
    parser.add_option("-m", "--sequence_length",
                      help="Length of sequence in the riddle. Assume file 'data/input_m.txt' exists")
    options, _ = parser.parse_args()

    m = int(options.sequence_length)
    sequences_list, n_correct_vertices_list = input_parser(m)

    ilp_manager = ILPManager(m)
    ilp_solver, s_star_to_color_edges = ilp_manager.build_ilp_solver(sequences_list, n_correct_vertices_list)
    s_star = ilp_manager.solve_ilp(ilp_solver, s_star_to_color_edges)

    print "Solution is: {}".format(''.join([str(i) for i in s_star]))

if __name__ == '__main__':
    main()
```

Solver Runner

src.ilp_manager.ILPManager(m) [source]

Model the 185th problem in Project Euler as an ILP (Integer Linear Program)

To instantiate this module, please specify the length on sequences.

build_ilp_solver(sequences_list, n_correct_vertices_list) [source]

Given input sequences, and number or correct vertices in each of them, build an ILP representation of the problem.

Parameters:

- **sequences_list** – List of sequences, each of length self.m, of integers between 0 and 9.
- **n_correct_vertices_list** – Number of correct vertices in each sequence in sequences_list. A vertex is correct if its color is equal to the color of the corresponding vertex in the solution s_star.

Returns:

tuple, containing:

- **ilp_solver**: Pulp instance, holding all the information needed for the solution.
- **s_star_to_color_edges**: The edges (variables) in the ilp_solver.

solve_ilp(ilp_solver, s_star_to_color_edges) [source]

Given a solver with the needed information, solve the ILP and extract the solution to problem 185.

Parameters:

- **ilp_solver** – Pulp instance, holding all the information needed for the solution.
- **s_star_to_color_edges** – The edges (variables) in the ilp_solver.

Returns:

s_star: List of integers that is the solution to problem 185.

Part I - Overview

Euler's 185th Riddle

In case you are interested:

Let s^* be a sequence.

We are given n .

For each sequence s ,

Our goal is to determine if s is a solution of the problem.

Source code for run_euler_185_solver

```
from optparse import OptionParser
from src.ilp_manager import ILPManager
from src.input_parser import input_parser
[docs]def main():
    """
    Run this code in a terminal or Jupyter notebook.
    ... options:
        Length of the sequence.
        ...
    parser = OptionParser()
    parser.add_option("-l", "--length", type="int", dest="length",
                      help="Length of the sequence.", default=10)
    options, sequences = parser.parse_args()
    m = int(options.length)
    sequences_list = sequences.sequences
    ilp_manager = ILPManager(m)
    ilp_solver = ilp_manager.get_ilp_solver()
    s_star = ilp_solver.solve()
    print "Solution found: %s" % s_star
    if __name__ == "__main__":
        main()
```

Solver Runner

[class src.ilp_manager.ILPManager\(m\) \[SOURCE\]](https://dalyag.github.io/Sphinx185/index.html)



Welcome to Sphinx185's documentation!

This documentation follows the solution of the 185th riddle from [Project Euler](#).

The goal of this project is to help make [Sphinx](#) accessible and easy to use.

To use this for YOUR next project, browse this documentation and follow the instructions [here](#).

Feel as free as you can to copy, paste and change anything you see here.

The source code is available [here](#).

Hope you find this useful!

DalyaG

- Euler's 185th Riddle
- Solver Runner
- Input Parser
- Naming Utils
- The ILP Manager class
- The main configurations file: config.py
- How To Use This for YOUR Next Project :)

Part I - Overview

Sphinx185 documentation >

Welcome to Sphinx185's documentation!



This documentation follows the solution of the 185th riddle from [Project Euler](#).
The goal of this piece is to help make [Sphinx](#) accessible and easy to use.
Feel as free as you can to copy, paste and change anything you see here.

The source code is available [here](#).
Hope you find this useful!
@DalyaG

OK Let's get to business:

- [Euler's 185th Riddle](#)
- [Solver Runner](#)
- [Input Parser](#)
- [Naming Utils](#)
- [The ILP Manager class](#)
- [The main configurations file: conf.py](#)
- [Index](#)

index.rst -
the content
of the home
page

Part I - Overview

conf.py -
settings for
the entire
project

Sphinx185 documentation >

Welcome to Sphinx185's documentation!



Sphinx185

- Euler's 185th Riddle
- Solver Runner
- Input Parser
- Naming Utils
- The ILP Manager class
- The main configurations file: conf.py

Quick search Go

This documentation follows the solution of the 185th riddle from [Project Euler](#).
The goal of this piece is to help make [Sphinx](#) accessible and easy to use.
Feel as free as you can to copy, paste and change anything you see here.

The source code is available [here](#).
Hope you find this useful!
@DalyaG

OK Let's get to business:

- Euler's 185th Riddle
- Solver Runner
- Input Parser
- Naming Utils
- The ILP Manager class
- The main configurations file: conf.py
- Index

Part II - Main configuration file - conf.py

Part II - Main configuration file - conf.py

```
"""
This is the main file in which the configuration for the documentation is made.

"""

# -*- coding: utf-8 -*-

# Sphinx185 documentation build configuration file, created by
# sphinx-quickstart on Sat May 12 19:34:31 2018.

# DalyaG: This file was heavily modified from its original build.
# Hope you find this useful :)

# -- Define here your working directory --

import os
import sys
sys.path.append(os.path.abspath('/Users/dalya/Documents/Sphinx185'))

# -- Some general info about the project --

project = u'Sphinx185'
copyright = u'2018, DalyaG'
author = u'dalyaG'

# -- A few basic configurations --

# The documentation in this project will be mostly generated from .rst files
# In this project, every auto-documented module/class has its own .rst file, under the main documentation dir,
# which is rendered into an .html page.
# Get more information here: http://www.sphinx-doc.org/en/master/usage/restructuredtext/basics.html
source_suffix = ['.rst']

# This is the name of the main page of the project.
# It means that you need to have an `index.rst` file where you will design the landing page of your project.
# It will be rendered into an .html page that you can find at: `build/html/index.html`
# (this is a standard name. change it only if you know what you are doing)
master_doc = 'index'

# List of patterns, relative to source directory, that match files and
# directories to ignore when looking for source files.
# This patterns also effect to html_static_path and html_extra_path
exclude_patterns = ['_build']

# List here any paths that contain templates, relative to this directory.
# You can find some not-so-intuitive information here: http://www.sphinx-doc.org/en/master/template.html
# But the best way to learn is by example, right? :)
# So, for example, in this project, I wanted to change the title of the Table Of Contents in the sidebar.
# So I copied '<Sphinx install dir>/themes/basic/globaltoc.html' into the '_templates' folder,
# and replaced 'Table of Content' with 'Sphinx185'.
templates_path = ['_templates']
```

```
# -- Define and configure non-default extensions ----

# You can find a list of available extension here: http://www.sphinx-doc.org/en/master/extensions.html
extensions = ['sphinx.ext.todo', 'sphinx.ext.viewcode', 'sphinx.ext.autodoc', 'sphinx.ext.imgmath']

# Above extensions explanation and configurations:

# todo: When you use the syntax ``todo`` some todo command`` in your doctstring,
# it will appear in a highlighted box in the documentation
# In order for this extension to work, make sure you include the following:
todo_include_todos = True

# viewcode: Next to each function/module in the documentation, you will have an internal link to the source code.
# The source code will have colors defined by the Pygments (syntax highlighting) style.
# You can checkout the available pygments here: https://help.farbox.com/pygments.html
pygments_style = 'native'

# autodoc: The best thing about Sphinx INHO is autodoc.
# It will automatically generate documentation for the docstrings in your code.
# Get more info here: http://www.sphinx-doc.org/en/master/ext/autodoc.html
# Some useful configurations:
autoclass_content = 'both' # Include both the class's and the init's docstrings.
autodoc_member_order = 'bysource' # In the documentation, keep the same order of members as in the code.
autodoc_default_flags = ['members'] # Default: include the docstrings of all the class/module members.

# imgmath: Sphinx allows use of LaTeX in the html documentation, but not directly. It is first rendered to an image.
# You can add here whatever preamble you are used to add to your LaTeX document.
imgmath_latex_preamble = r'''
\usepackage{color}
\definecolor{offwhite}{rgb}{230,238,238}
\everymath{\color{offwhite}}
\everydisplay{\color{offwhite}}
...
'''

# -- Options for HTML output --

# The theme to use for HTML and HTML Help pages.
# You can find available themes here: http://www.sphinx-doc.org/en/master/theming.html
# In this project, I wanted to use a non-default theme, and so I downloaded the `graphite` template from here:
# https://github.com/sphinx-doc/sphinx-themes-graphite
# Some adjustments I made to graphite:
# - I did not use the pygment configuration, and so removed "pygments_style = graphite.GraphiteStyle" from theme.conf
# - In the static folder, I configured several classes both in graphite.css and in html4css1.css,
#   you can download the original and compare to find those changes.
html_theme = 'graphite'
# When using a non-built-in theme, define the path to your template code.
html_theme_path = ['']

# Add any paths that contain custom static files (such as style sheets) here,
# relative to this directory. They are copied after the builtin static files,
# so a file named 'default.css' will overwrite the builtin 'default.css'.
html_static_path = ['static']

# Defining the static path allows me to add my own logo for the project:
# (make sure the theme of your choice support the use of logo.
html_logo = '_static/sphinx_and_dalya.png'

# Custom sidebar templates, must be a dictionary that maps document names to template names.
# In this project I chose to include in the sidebar:
# - Table of Contents: I chose globaltoc as it is less refined,
#   and I configured the title by editing the globaltoc template (see explanation above, in the templates_path comment)
# - Search box: appears below the ToC, and can be configured by editing css attributes.
html_sidebars = {
    '**': [
        'globaltoc.html',
        'searchbox.html'
    ]
}
```

Part

DONT LET CONF.PY EAT YOU UP, KID

nf.py

```
"""
This is the main file in which t
"""

# -*- coding: utf-8 -*-
#
# Sphinx185 documentation build
# sphinx-quickstart on Sat May 1
#
# DalyaG: This file was heavily
# Hope you find this useful :)

# -- Define here your working di

import os
import sys
sys.path.append(os.path.abspath('

# -- Some general info about th
project = u'Sphinx185'
copyright = u'2018, DalyaG'
author = u'dalyaG'

# -- A few basic configurations
# The documentation in this proj
# In This project, every auto-do
# which is rendered into an .H
# Get more information here: ht
source_suffix = ['.rst']

# This is the name of the main p
# It means that you need to have
# It will be rendered into an .h
# (this is a standard name. chan
master_doc = 'index'

# List of patterns, relative to
# directories to ignore when loo
# This patterns also effect to h
exclude_patterns = ['_build']

# List here any paths that conta
# You can find some not-so-intui
# But the best way to learn is b
# So, for example, in this proje
# So I copied `<Sphinx install
# and replaced `Table of Co
templates_path = ['_templates']
```

GO ON, YOU CAN DO IT!

```
er/extensions.html
x.ext.imgmath']

ernal link to the source code,
nting) style.
gments.html

gs in your code.

mbers as in the code.
ass/module members.

t is first rendered to an image.

ml
phite` template from here;
phite.GraphiteStyle" from theme.conf
n html4css1.css,
```

Part II - Main configuration file - conf.py

```
"""
This is the main file in which the configuration for the documentation is made.
```

```
"""
# -*- coding: utf-8 -*-
#
# Sphinx185 documentation build
# sphinx-quickstart on Sat May
#
# DalyaG: This file was heavil
# Hope you find this useful :)
```

```
# -- Define here your working
```

```
import os
import sys
sys.path.append(os.path.abspath(
```

```
# -- Some general info about
```

```
project = u'Sphinx185'
copyright = u'2018, DalyaG'
author = u'dalyaG'
```

```
# -- A few basic configuration
```

```
# The documentation in this pr
# In This project, every auto-
# which is rendered into an
# Get more information here: h
source_suffix = ['.rst']
```

```
# This is the name of the main
# It means that you need to ha
# It will be rendered into an
# (this is a standard name. ch
master_doc = 'index'
```

```
# List of patterns, relative to t
# directories to ignore when looki
# This patterns also effect to html
exclude_patterns = ['_build']
```

```
# List here any paths that contain templates, relative to this directory.
# You can find some not-so-intuitive information here: http://www.sphinx-doc.org/en/master/template.html
# But the best way to learn is by example, right? :)
# So, for example, in this project, I wanted to change the title of the Table Of Contents in the sidebar.
# So I copied '<Sphinx install dir>/themes/basic/globaltoc.html' into the '_templates' folder,
# and replaced 'Table of Content' with 'Sphinx185'.
templates_path = ['_templates']
```

```
"""
# -- Define and configure non-default extensions -----
# You can find a list of available extension here: http://www.sphinx-doc.org/en/master/extensions.html
extensions = ['sphinx.ext.todo', 'sphinx.ext.viewcode', 'sphinx.ext.autodoc', 'sphinx.ext.imgmath']

# Above extensions explanation and configurations:
```

" in your docstring,
the following:
have an internal link to the source code.
writer highlighting) style.
[getbox.com/pygments.html](#)

the docstrings in your code.
[autodoc.html](#)
docstrings.
order of members as in the code.
of all the class/module members.
directly. It is first rendered to an image.
x document.

[xr/theming.html](#)
added the 'graphite' template from here:
[site.py](#)
ite.css and in html4css1.css,

here,
files,
ject:

```
# Custom sidebar templates, must be a dictionary that maps document names to template names.
# In This project I chose to include in the sidebar:
#   - Table of Contents: I chose globaltoc as it is less refined,
#     and configured the title by editing the globaltoc template (see explanation above, in the templates_path comment)
#   - Crossbox: appears below the TOC, and can be configured by editing css attributes.
html_sidebars = {
    '**': [
        'globaltoc.html',
        'searchbox.html'
    ]
}
```

Full details - EXTRA SLIDES

Part II - Main configuration file - conf.py

'sphinx.ext.autodoc'

```
#           It will appear in a highlighted box in the documentation.
#           In order for this extension to work, make sure you include the following:
todo_include_todos = True

# viewcode: Next to each function/module in the documentation, you will have an internal link to the source code.
#           The source code will have colors defined by the Pygments (syntax highlighting) style.
#           You can checkout the available pygments here: https://help.farbox.com/pygments.html
pygments_style = 'native'

# autodoc: The best thing about Sphinx IMO is autodoc.
#           It allows you to automatically generate documentation for the docstrings in your code.
#           Get more info here: http://www.sphinx-doc.org/en/master/ext/autodoc.html
# Some useful configurations:
autoclass_content = 'both' # Include both the class's and the init's docstrings.
autodoc_member_order = 'bysource' # In the documentation, keep the same order of members as in the code.
autodoc_default_flags = ['members'] # Default: include the docstrings of all the class/module members.

# imgmath: Sphinx allows use of LaTeX in the html documentation, but not directly. It is first rendered to an image.
# You can add here whatever preamble you are used to adding to your LaTeX document.
imgmath_latex_preamble = r'''
\usepackage{color}
\definecolor{offwhite}{rgb}{(230,238,238)}
\everymath{\color{offwhite}}
\everydisplay{\color{offwhite}}
...
\color{black}

# -- Options for HTML output --
# The theme to use for HTML and HTML Help pages.
# You can find available themes here: http://www.sphinx-doc.org/en/master/theming.html
# In this project, I wanted to use a non-default theme, and so I downloaded the 'graphite' template from here:
# https://github.com/Cehoy/Sphinx-theme-graphite
# Some adjustments I made to graphite:
#   - I did not use the pygment configuration, and so removed "pygments_style = graphite.GraphiteStyle" from theme.conf
#   - I copied the 'graphite' template and deleted graphite.py
#   - In the static folder, I configured several classes both in graphite.css and in html4css1.css,
#     you can compare the original and compare to find those changes.
html_theme = 'graphite'
# When using a non-built-in theme, define the path to your template code.
html_theme_path = ['.']

# Add any paths that contain custom static files (such as style sheets) here,
# relative to this directory. They are copied after the builtin static files,
# so a file named 'default.css' will overwrite the builtin 'default.css'.
html_static_path = ['_static']

# Defining the static path allows me to add my own logo for the project:
# (make sure the theme of your choice support the use of logo.
html_logo = '_static/sphinx_and_daiya.png'

# Custom sidebar templates, must be a dictionary that maps document names to template names.
# In This project I chose to include in the sidebar:
#   - Table of Contents: I chose globaltoc as it is less refined,
#     and configured the title by editing the globaltoc template (see explanation above, in the templates_path comment)
#     Content box appears below the ToC, and can be configured by editing css attributes.
html_sidebars = {
    '**': [
        'globaltoc.html',
        'searchbox.html'
    ]
}
```

Part II - Main configuration file - conf.py

'sphinx.ext.autodoc'

.. autoclass:: src.ilp_manager.ILPManager

```
#           It will appear in a highlighted box in the documentation.
#           In order for this extension to work, make sure you include the following:
todo_include_todos = True

# Sphinx TINHO is autodoc,
# automatically generate documentation for the docstrings in your code.
# p://www.sphinx-doc.org/en/master/ext/autodoc.html
# module in the documentation, you will have an internal link to the source code.
# ave colors defined by the Pygments (syntax highlighting) style.
# available pygments here: https://help.farbox.com/pygments.html

# Both the class's and the init's docstrings.
# In the documentation, keep the same order of members as in the code.
# # Default: include the docstrings of all the class/module members.

# imgmath: Sphinx allows use of LaTeX in the html documentation, but not directly. It is first rendered to an image.
# You can add here whatever preamble you are used to adding to your LaTeX document.
imgmath_latex_preamble = r'''
\usepackage{color}
\definecolor{offwhite}{rgb}{(230,238,238)}
\renewcommand{\color}[1]{\color{offwhite}}
\renewcommand{\textcolor}[2]{\color{offwhite}#2}
\renewcommand{\textcolor}[2]{\color{offwhite}#2}
...
'''

# -- Options for HTML output --
# The theme to use for HTML and HTML Help pages.
# You can find available themes here: http://www.sphinx-doc.org/en/master/theming.html
# In this project, I wanted to use a non-default theme, and so I downloaded the 'graphite' template from here;
# https://github.com/CaiqueSilveira/graphite
# Some adjustments I made to graphite:
# - I did not use the pygments configuration, and so removed "pygments_style = graphite.GraphiteStyle" from theme.conf
# - I copied the static folder, and deleted graphite.py
# - In the static folder, I configured several classes both in graphite.css and in html4css1.css,
#   you can compare the original and compare to find those changes.
html_theme = 'graphite'
# When using a non-built-in theme, define the path to your template code.
html_theme_path = ['.']

# Add any paths that contain custom static files (such as style sheets) here,
# relative to this directory. They are copied after the builtin static files,
# so a file named 'default.css' will overwrite the builtin 'default.css'.
html_static_path = ['_static']

# Defining the static path allows me to add my own logo for the project:
# (make sure the theme of your choice support the use of logo.
html_logo = '_static/sphinx_and_daiya.png'

# Custom sidebar templates, must be a dictionary that maps document names to template names.
# In This project I chose to include in the sidebar:
# - Table of Contents: I chose globaltoc as it is less refined,
#   and configured the title by editing the globaltoc template (see explanation above, in the templates_path comment)
# - Search box: appears below the ToC, and can be configured by editing css attributes.
html_sidebars = {
    '**': [
        'globaltoc.html',
        'searchbox.html'
    ]
}
```

Part II - Main configuration file - conf.py

```
'sphinx.ext.autodoc'
```

```
.. autoclass:: src.ilp_manager.ILPManager
```



```
class src.ilp_manager.ILPManager(m) [source]
```

Model the 185th problem in Project Euler as an ILP (Integer Linear Program)

To instantiate this module, please specify the length on sequences.

```
build_ilp_solver(sequences_list, n_correct_vertices_list) [source]
```

Given input sequences, and number or correct vertices in each of them, build an ILP representation of the problem.

Parameters:

- *sequences_list* – List of sequences, each of length self.m, of integers between 0 and 9.
- *n_correct_vertices_list* – Number of correct vertices in each sequence in *sequences_list*. A vertex is correct if its color is equal to the color of the corresponding vertex in the solution *s_star*.

Returns:

tuple, containing:

- *ilp_solver*: Pulp instance, holding all the information needed for the solution.
- *s_star_to_color_edges*: The edges (variables) in the *ilp_solver*.

```
solve_ilp(ilp_solver, s_star_to_color_edges) [source]
```

Given a solver with the needed information, solve the ILP and extract the solution to problem 185.

Parameters:

- *ilp_solver* – Pulp instance, holding all the information needed for the solution.
- *s_star_to_color_edges* – The edges (variables) in the *ilp_solver*.

Returns:

s_star: List of integers that is the solution to problem 185.

```
#           It will appear in a highlighted box in the documentation.
#           In order for this extension to work, make sure you include the following:
todo_include_todos = True
```

```
module in the documentation, you will have an internal link to the source code.
ave colors defined by the Pygments (syntax highlighting) style.
p://www.sphinx-doc.org/en/master/ext/autodoc.html
```

```
inx THO is autodoc,
matically generate documentation for the docstrings in your code.
p://www.sphinx-doc.org/en/master/ext/autodoc.html

ude both the class's and the init's docstrings.
# In the documentation, keep the same order of members as in the code.
| # default: include the docstrings of all the class/module members.
```

```
# imgmath: Sphinx allows use of LaTeX in the html documentation, but not directly. It is first rendered to an image.
# You can add here whatever preamble you are used to adding to your LaTeX document.
imgmath_latex_preamble = r'''
\usepackage{color}
\definecolor{offwhite}{rgb}{(230,238,238)}
\everymath{\color{offwhite}}
\everydisplay{\color{offwhite}}
'''
```

```
## Options for HTML output

# The theme to use for HTML and HTM Help pages.
# You can find available themes here: http://www.sphinx-doc.org/en/master/theming.html
# In this project, I wanted to use a non-default theme, and so I downloaded the 'graphite' template from here:
# https://github.com/rtomayko/sphinx-theme-graphite
# Some adjustments I made to graphite:
# - I did not use the pygment configuration, and so removed "pygments_style = graphite.GraphiteStyle" from theme.conf
# and deleted graphite.py
```

```
# - In the static folder, I configured several classes both in graphite.css and in html4css1.css,
# you can download the original and compare to find those changes.
html_theme = 'graphite'
# When using a non-built-in theme, define the path to your template code.
html_theme_path = ['..']

# Add any paths that contain custom static files (such as style sheets) here,
# relative to this directory. They are copied after the builtin static files,
# so a file named 'default.css' will overwrite the builtin 'default.css'.
html_static_path = ['_static']
```

```
# Defining the static path allows me to add my own logo for the project:
# (make sure the theme of your choice support the use of logo.
html_logo = '_static/sphinx_and_daiya.png'

# Custom sidebar templates, must be a dictionary that maps document names to template names.
# In This project I chose to include in the sidebar:
# - Table of Contents: I chose globaltoc as it is less refined,
# and configured the title by editing the globaltoc template (see explanation above, in the templates_path comment)
# Content box appears below the ToC, and can be configured by editing css attributes.
html_sidebars = {
    '**': [
        'globaltoc.html',
        'searchbox.html'
    ]
}
```

Part II - Main configuration file - conf.py

'sphinx.ext.autodoc'

```
#           It will appear in a highlighted box in the documentation.
#           In order for this extension to work, make sure you include the following:
todo_include_todos = True

# viewcode: Next to each function/module in the documentation, you will have an internal link to the source code.
#           The source code will have colors defined by the Pygments (syntax highlighting) style.
#           You can checkout the available pygments here: https://help.farbox.com/pygments.html
pygments_style = 'native'

# autodoc: The best thing about Sphinx IMO is autodoc.
#           It allows you to automatically generate documentation for the docstrings in your code.
#           Get more info here: http://www.sphinx-doc.org/en/master/ext/autodoc.html
# Some useful configurations:
autoclass_content = "both" # Include both the class's and the init's docstrings.
autodoc_member_order = "bysource" # In the documentation, keep the same order of members as in the code.
autodoc_default_flags = ['members'] # Default: include the docstrings of all the class/module members.

# imgmath: Sphinx allows use of LaTeX in the html documentation, but not directly. It is first rendered to an image.
# You can add here whatever preamble you are used to adding to your LaTeX document.
imgmath_latex_preamble = r"""
\usepackage{color}
\definecolor{offwhite}{rgb}{230,238,238}
\everymath{\color{offwhite}}
\everydisplay{\color{offwhite}}
...
"""

# -- Options for HTML output --
# The theme to use for HTML and HTML Help pages.
# You can find available themes here: http://www.sphinx-doc.org/en/master/theming.html
# In this project, I wanted to use a non-default theme, and so I downloaded the 'graphite' template from here:
# https://github.com/sphinx-doc/sphinx/tree/graphite
# Some adjustments I made to graphite:
#   - I did not use the pygment configuration, and so removed "pygments_style = graphite.GraphiteStyle" from theme.conf
#   - In the static folder, I configured several classes both in graphite.css and in html4css1.css,
#     you can download the original and compare to find those changes.
html_theme = 'graphite'
# When using a non-built-in theme, define the path to your template code.
html_theme_path = ['.']

# Add any paths that contain custom static files (such as style sheets) here,
# relative to this directory. They are copied after the builtin static files,
# so an file named 'default.css' will overwrite the builtin 'default.css'.
html_static_path = ['_static']

# Defining the static path allows me to add my own logo for the project:
# (make sure the theme of your choice support the use of logo.
html_logo = '_static/sphinx_and_daiya.png'

# Custom sidebar templates, must be a dictionary that maps document names to template names.
# In This project I chose to include in the sidebar:
#   - Table of Contents: I chose globaltoc as it is less refined,
#     and I configured the title by editing the globaltoc template (see explanation above, in the templates_path comment)
#     Content box appears below the ToC, and can be configured by editing css attributes.
html_sidebars = {
    '**': [
        'globaltoc.html',
        'searchbox.html'
    ]
}
```

Full details here:

<http://www.sphinx-doc.org/en/master/ext/autodoc.html>

Part III - Document YOUR next project

Part III - Document YOUR next project



I FORGET THINGS ALMOST INSTANTLY,
IT RUNS IN MY FAMINLY

Part III - Document YOUR next project

We will follow these instructions:

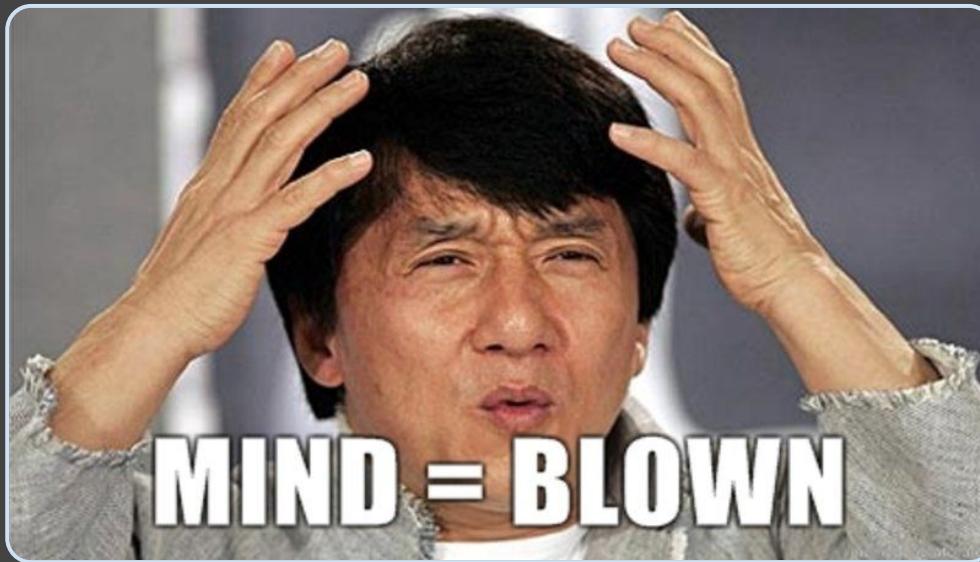
(found here https://dalyag.github.io/Sphinx185/how_to_use_this_for_your_next_project.html)

How To Use This for YOUR Next Project :)

1. `pip install sphinx`
2. Copy the `documentation_template_for_your_next_project` folder in this repository, make it a subdirectory in your local project and rename it `documentation`.
3. Edit `conf.py` by searching the pattern `#CHNAGEME#` inside the file and follow the instructions.
4. Edit `index.rst` by following the inline instructions.
5. To add a page to your documentation:
 - a. Create a `.rst` file for the function/module you wish to document (you can use the templates supplied here), and
 - b. Add the name of the `.rst` file to the `toctree` in `index.rst`.
6. In terminal, inside the `documentation` folder, run `make html`.
(TIP OF THE WEEK: actually, always run `make clean html` to clear sphinx cache and build from scratch)
7. To view locally - open the file `documentation/_build/html/index.html` in your browser, and enjoy the read :)
8. To share - you can use [GitHub Pages](#) to host your documentation:
 - a. Copy the content of `documentation/_build/html/` into a new `docs` folder, under the root of the project.
 - b. Create an empty file `_nojekyll` inside `docs` folder (this tells GitHub Pages to bypass the default `jekyll` themes and use the `html` and `css` in your project)
 - c. Push your changes to master branch.
 - d. In your repository on GitHub, go to "Settings" -> "GitHub Pages" -> "Source" and select "master branch /docs folder".
 - e. Share your beautiful documentation site at https://<your_git_usrname>.github.io/<project_name>/

Part III - Document YOUR next project

1. pip install sphinx

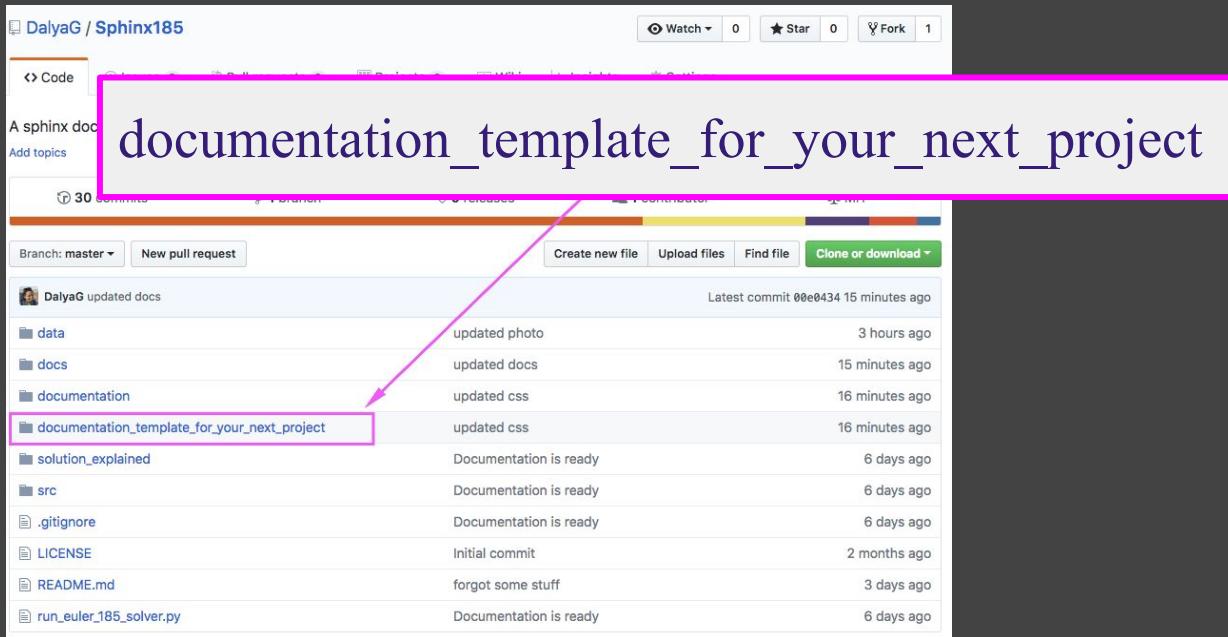


Part III - Document YOUR next project

2. Go to <https://github.com/DalyaG/Sphinx185>

Part III - Document YOUR next project

2. Go to <https://github.com/DalyaG/Sphinx185>



Part III - Document YOUR next project

2. Go to <https://github.com/DalyaG/Sphinx185>

- * Download this folder to your local project:
documentation_template_for_your_next_project
- * Rename it *documentation*

Part III - Document YOUR next project

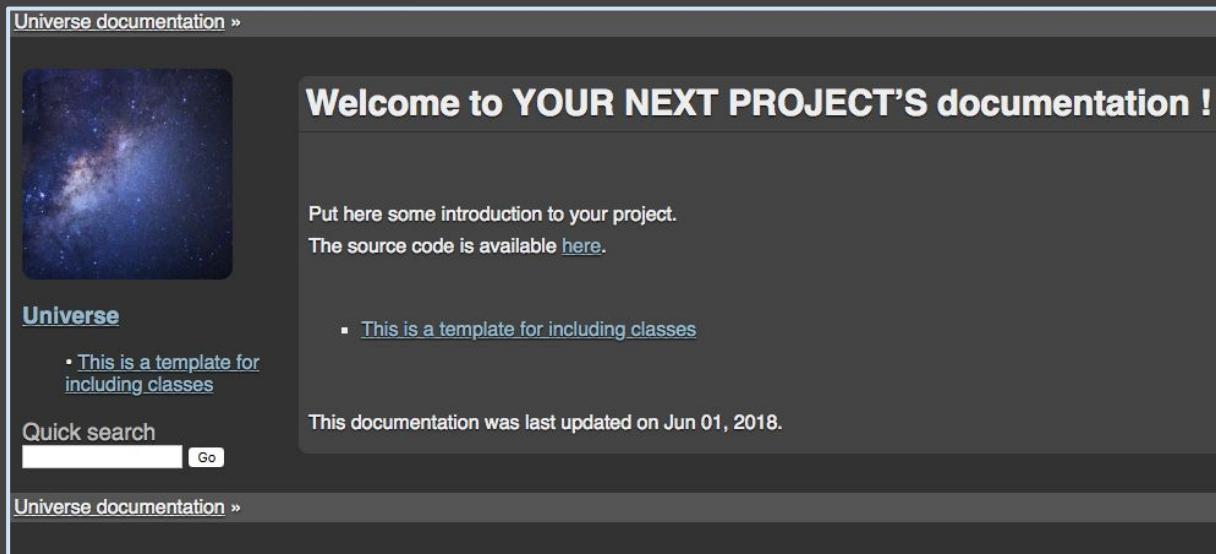
Bravo! You now have a documentation!

Part III - Document YOUR next project

Bravo! You now have a documentation!

Open this file in your browser to see:

<your_project>/documentation/_build/html/index.html



Part III - Document YOUR next project

3. Edit *conf.py*

Part III - Document YOUR next project

3. Edit *conf.py* by searching the pattern `#CHNAGEME#` inside the file and follow the instructions.

```
# --- Some general info about the project ---

#CHNAGEME# Change this to fit your project.
project = u'Universe'
copyright = u'2018, DalyaG'
author = u'DalyaG'
```

Part III - Document YOUR next project

4. Edit *index.rst*

Part III - Document YOUR next project

4. Edit *index.rst* by following the inline instructions.

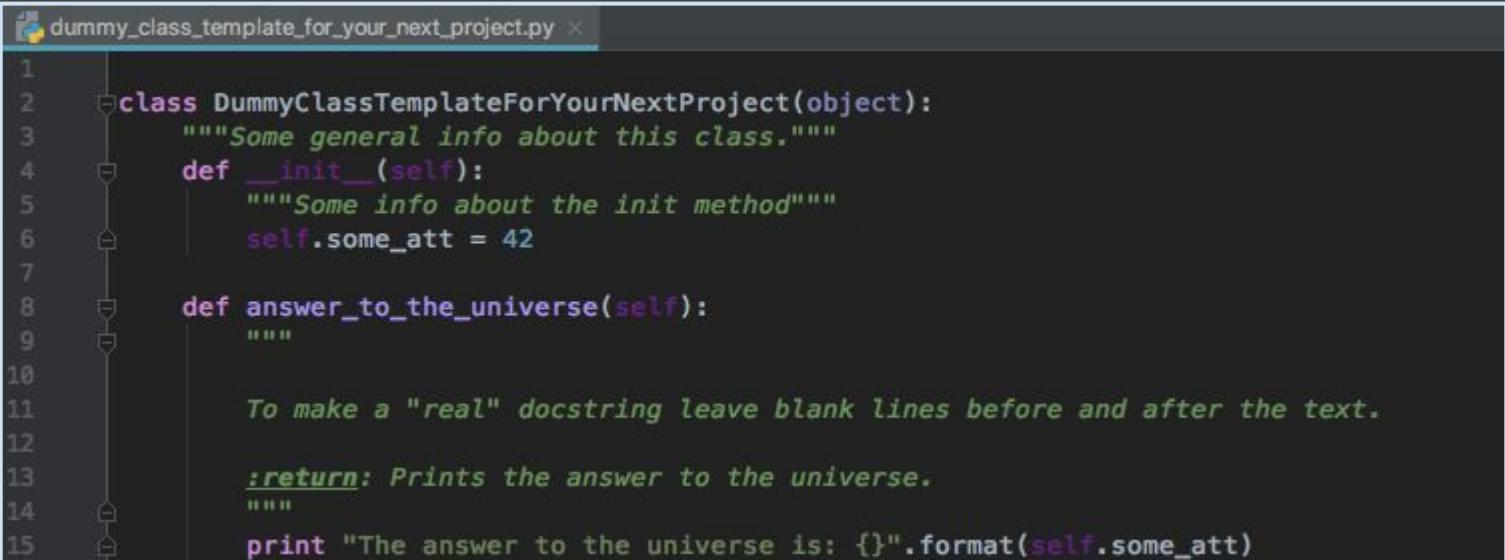
```
Welcome to YOUR NEXT PROJECT'S documentation !
=====
| Put here some introduction to your project.
| The source code is available `here <https://github.com/username/projectname>`_.
```

Part III - Document YOUR next project

5. To add a page to your documentation:

Part III - Document YOUR next project

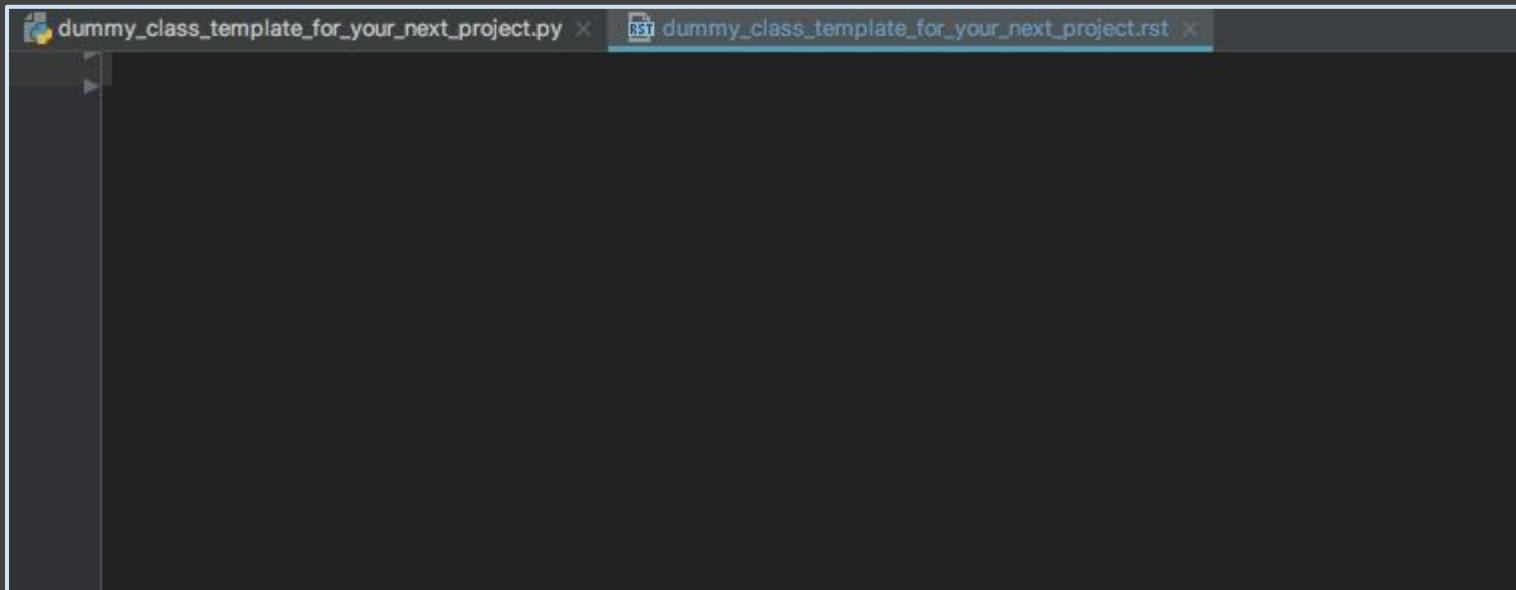
5. To add a page to your documentation:



```
1  dummy_class_template_for_your_next_project.py ×
2
3  class DummyClassTemplateForYourNextProject(object):
4      """Some general info about this class."""
5      def __init__(self):
6          """Some info about the init method"""
7          self.some_att = 42
8
8      def answer_to_the_universe(self):
9          """
10
11          To make a "real" docstring leave blank lines before and after the text.
12
13          :return: Prints the answer to the universe.
14          """
15          print "The answer to the universe is: {}".format(self.some_att)
```

Part III - Document YOUR next project

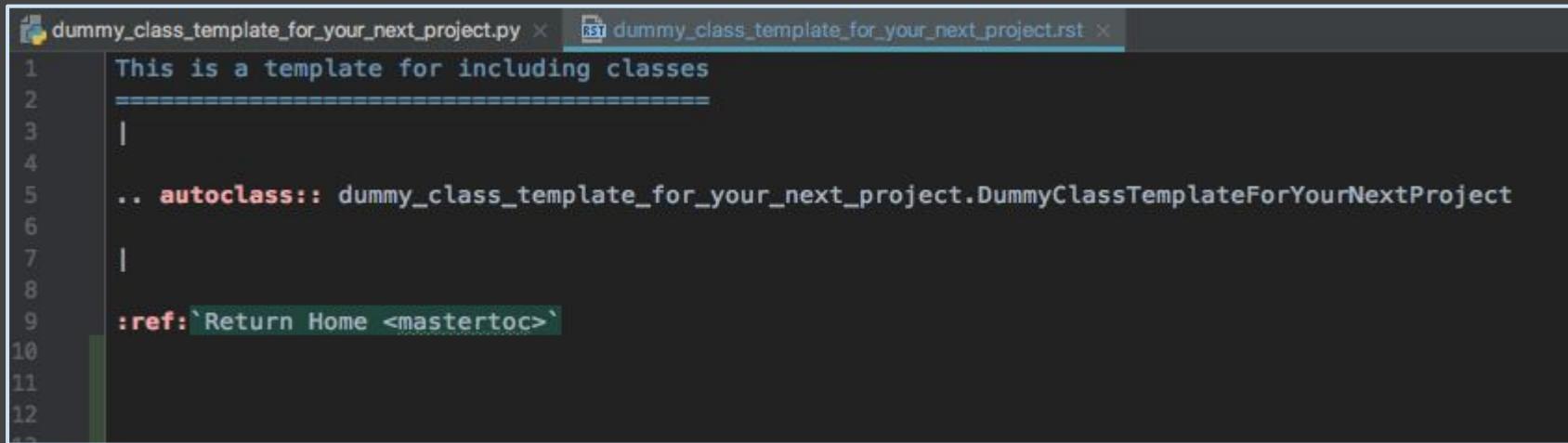
5. To add a page to your documentation:
Create a *.rst* file



Part III - Document YOUR next project

5. To add a page to your documentation:

Create a `.rst` file and edit is to include your module



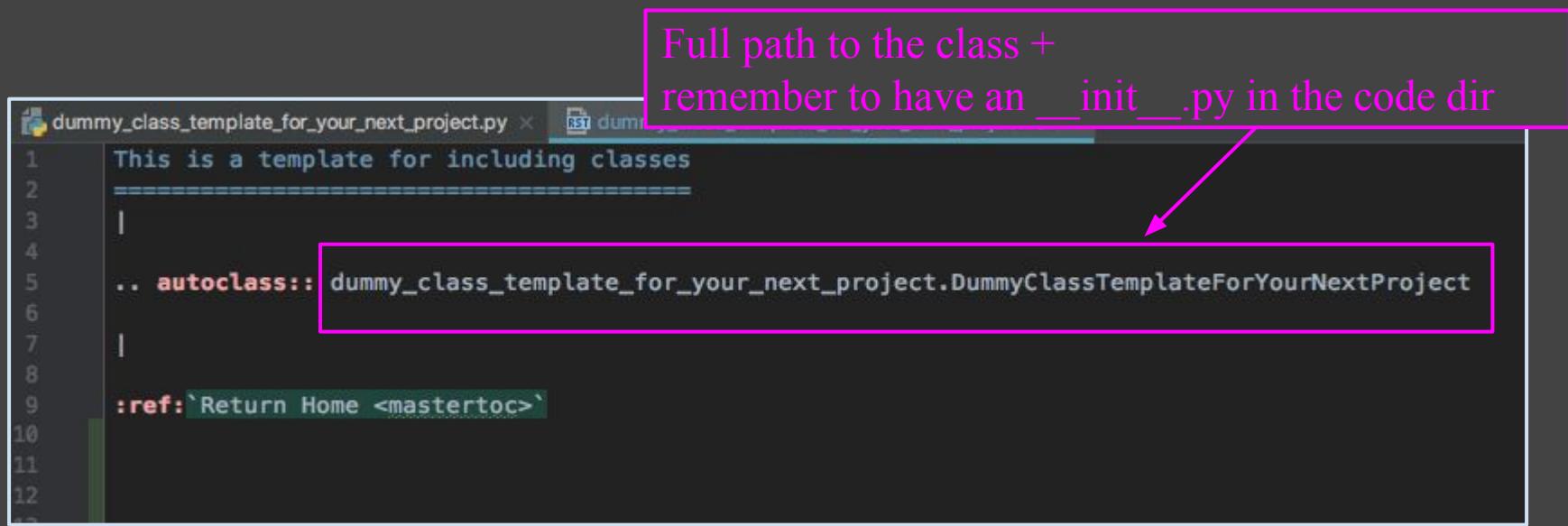
The screenshot shows a code editor with two tabs open. The left tab is named `dummy_class_template_for_your_next_project.py` and contains Python code. The right tab is named `RST dummy_class_template_for_your_next_project.rst` and contains reStructuredText (rst) code.

```
This is a template for including classes
=====
|
.. autoclass:: dummy_class_template_for_your_next_project.DummyClassTemplateForYourNextProject
|
:ref:`Return Home <mastertoc>`
```

Part III - Document YOUR next project

5. To add a page to your documentation:

Create a *.rst* file and edit it to include your module



The screenshot shows a code editor with two tabs: "dummy_class_template_for_your_next_project.py" and "rst.dummy". The "rst.dummy" tab contains the following RST code:

```
This is a template for including classes
=====
|
.. autoclass:: dummy_class_template_for_your_next_project.DummyClassTemplateForYourNextProject
|
:ref:`Return Home <mastertoc>`
```

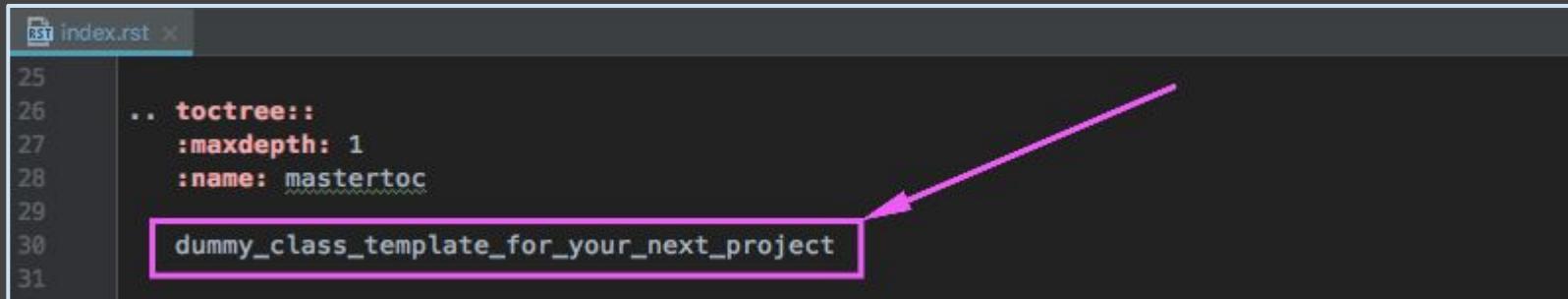
A callout box with a purple border and a matching arrow points from the text "remember to have an `__init__.py` in the code dir" to the class definition line in the RST code.

Full path to the class +
remember to have an `__init__.py` in the code dir

Part III - Document YOUR next project

5. To add a page to your documentation:

Create a `.rst` file and edit it to include your module
And add its name to the *TOC* in `index.rst`



```
index.rst
25
26 .. toctree::
27     :maxdepth: 1
28     :name: mastertoc
29
30     dummy_class_template_for_your_next_project
31
```

A screenshot of a code editor showing the file `index.rst`. The code contains reStructuredText directives. A pink rectangular box highlights the line `dummy_class_template_for_your_next_project`, and a pink arrow points from the bottom right towards this highlighted line.

Part III - Document YOUR next project

6. In terminal, inside the *documentation* folder, run *make clean html*:

Part III - Document YOUR next project

6. In terminal, inside the *documentation* folder, run *make clean html*:

```
~/Documents/Sphinx185/documentation (git::master) ] make clean html
```



Part III - Document YOUR next project

6. In terminal, inside the *documentation* folder, run *make clean html*:

```
[ dalya@Dalya-Gartzman-Mackbook ~/Documents/Sphinx185/documentation (git::master) ] make clean html
Removing everything under '_build'...
Running Sphinx v1.6.6
making output directory...
loading pickled environment... not yet created
building [mo]: targets for 0 po files that are out of date
building [html]: targets for 8 source files that are out of date
updating environment: 8 added, 0 changed, 0 removed
reading sources... [100%] run_euler_185_solver
looking for now-outdated files... none found
pickling environment... done
checking consistency... done
preparing documents... done
writing output... [100%] run_euler_185_solver
generating indices... genindex py-modindex
highlighting module code... [100%] src.naming_utils
writing additional pages... search
copying static files... done
copying extra files... done
dumping search index in English (code: en) ... done
dumping object inventory... done
build succeeded.

Build finished. The HTML pages are in _build/html.
[ dalya@Dalya-Gartzman-Mackbook ~/Documents/Sphinx185/documentation (git::master) ]
```

Part III - Document YOUR next project

6. In terminal, inside the *documentation* folder, run *make clean html*:

```
[ dalya@Dalya-Gartzman-Mackbook ~/Documents/Sphinx185/documentation (git::master) ] make clean html
Removing everything under '_build'...
Running Sphinx v1.6.6
making output directory...
loading pickled environment... not yet created
building [mo]: targets for 0 po files that are out of date
building [html]: targets for 8 source files that are out of date
updating environment: 8 added, 0 changed, 0 removed
reading sources... [100%] run_euler_185_solver
/Users/dalya/Documents/Sphinx185/documentation/index.rst:9: WARNING: Line block ends without a blank line.
looking for now-outdated files... none found
pickling environment... done
checking consistency... done
preparing documents... done
writing output... [100%] run_euler_185_solver
generating indices... genindex py-modindex
highlighting module code... [100%] src.naming_utils
writing additional pages... search
copying static files... done
copying extra files... done
dumping search index in English (code: en) ... done
dumping object inventory... done
build succeeded, 1 warning.

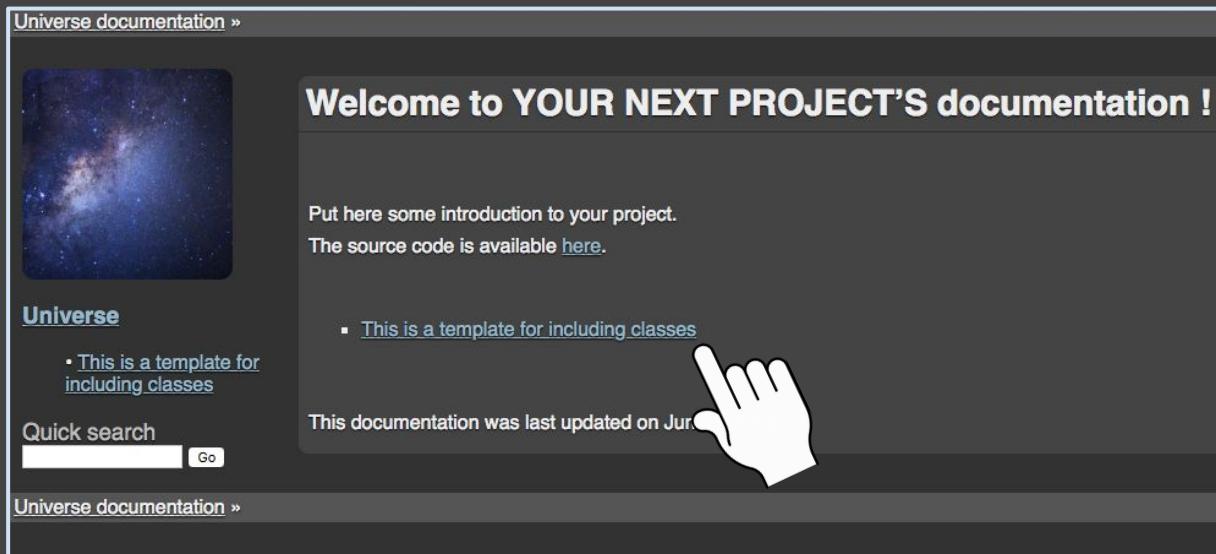
Build finished. The HTML pages are in _build/html.
[ dalya@Dalya-Gartzman-Mackbook ~/Documents/Sphinx185/documentation (git::master) ]
```

Part III - Document YOUR next project

7. To view locally

Part III - Document YOUR next project

7. To view locally, open this file in your browser
documentation/_build/html/index.html



The screenshot shows a web browser window displaying a documentation page. The title bar reads "Universe documentation »". The main content area features a large image of a galaxy on the left and a bold title "Welcome to YOUR NEXT PROJECT'S documentation !" on the right. Below the title, there is placeholder text: "Put here some introduction to your project." and "The source code is available [here](#)". A sidebar on the left is titled "Universe" and contains a bullet point: "• This is a template for including classes". At the bottom of the page, there is a "Quick search" input field and a "Go" button. A large white hand icon with a pointing finger is overlaid on the bottom right corner of the page content.

Universe documentation »

Welcome to YOUR NEXT PROJECT'S documentation !

Put here some introduction to your project.
The source code is available [here](#).

Universe

- This is a template for including classes

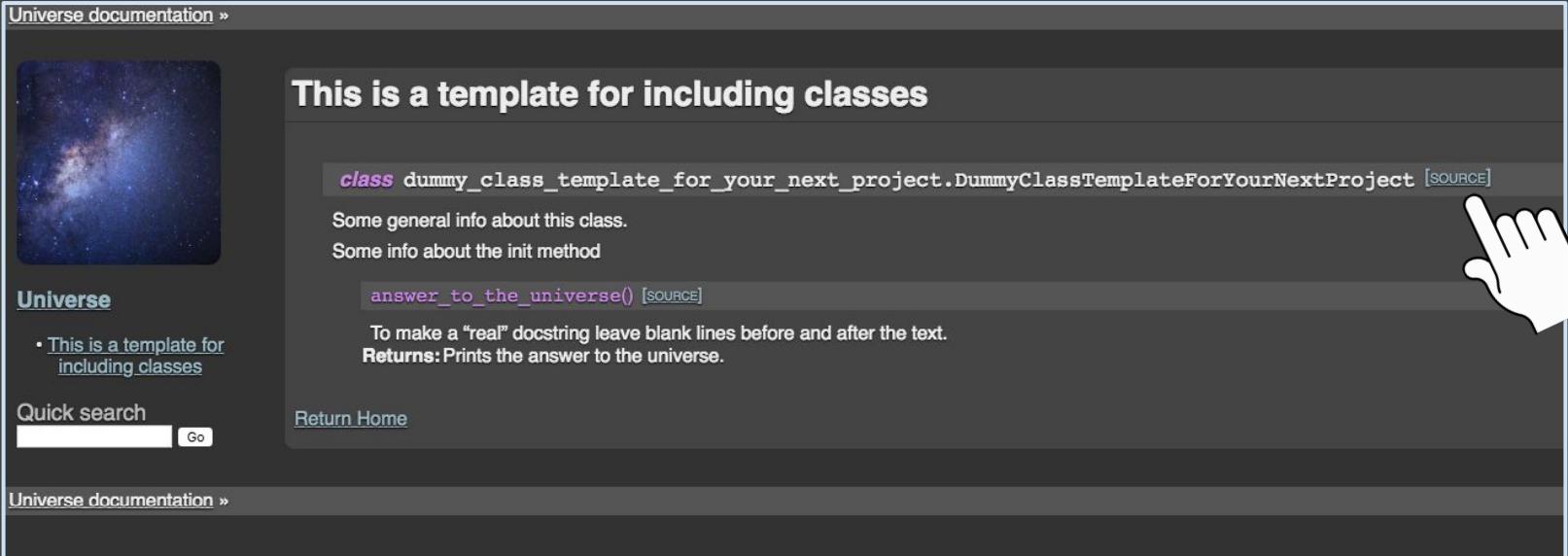
Quick search Go

This documentation was last updated on Jun 1, 2023.

Universe documentation »

Part III - Document YOUR next project

7. To view locally, open this file in your browser
documentation/_build/html/index.html



Universe documentation »

This is a template for including classes

`class dummy_class_template_for_your_next_project.DummyClassTemplateForYourNextProject [SOURCE]`

Some general info about this class.
Some info about the init method

`answer_to_the_universe() [SOURCE]`

To make a “real” docstring leave blank lines before and after the text.
Returns: Prints the answer to the universe.

Universe

- This is a template for including classes

Quick search Go

Return Home

Universe documentation »

Part III - Document YOUR next project

7. To view locally, open this file in your browser *documentation/_build/html/index.html*

Universe documentation » Module code »



Source code for dummy_class_template_for_your_next_project

```
[docs]class DummyClassTemplateForYourNextProject(object):
    """Some general info about this class."""
    def __init__(self):
        """Some info about the init method"""
        self.some_att = 42

[docs]    def answer_to_the_universe(self):
        """
        To make a "real" docstring leave blank lines before and after the text.

        :return: Prints the answer to the universe.
        """
        print "The answer to the universe is: {}".format(self.some_att)
```

Universe

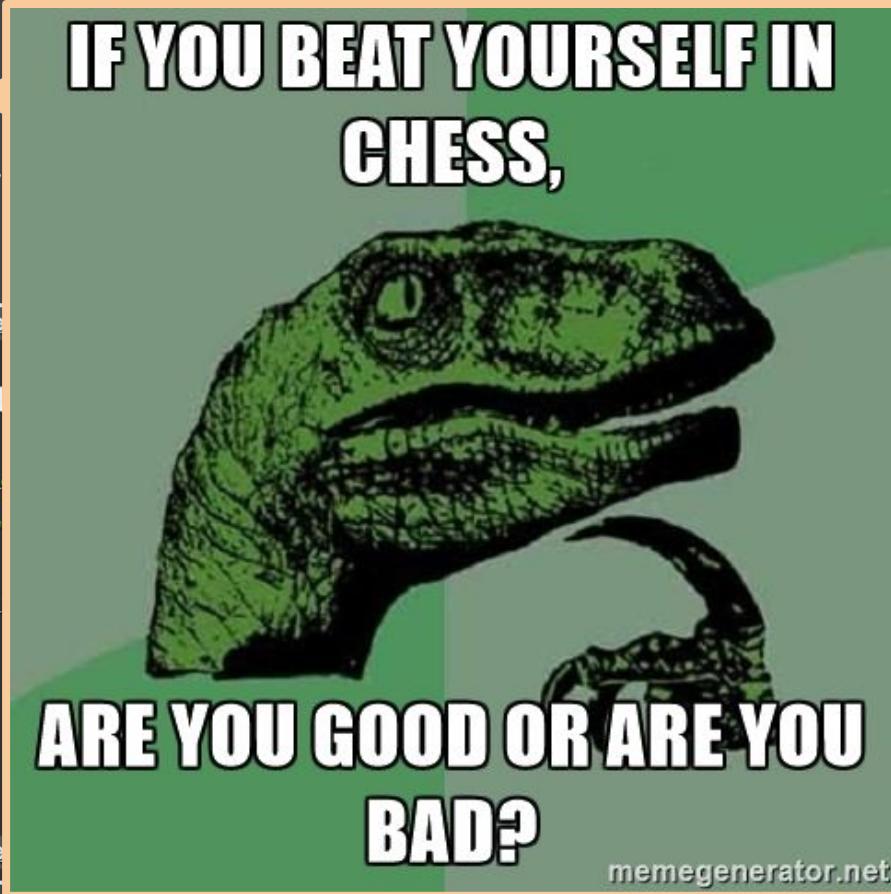
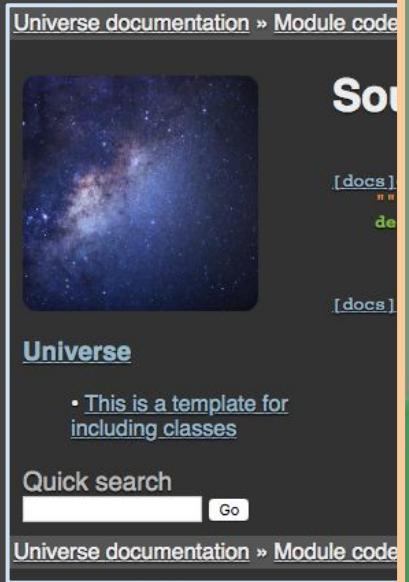
- This is a template for including classes

Quick search Go

Universe documentation » Module code »

Part III - Document YOUR next project

7. To view local documentation



browser

your_next_project

Part III - Document YOUR next project

8. SHARE your documentation!



Part III - Document YOUR next project

8. Hosting on GitHub Pages:



Part III - Document YOUR next project

8. Hosting on GitHub Pages:

Part III - Document YOUR next project

8. Hosting on GitHub Pages:

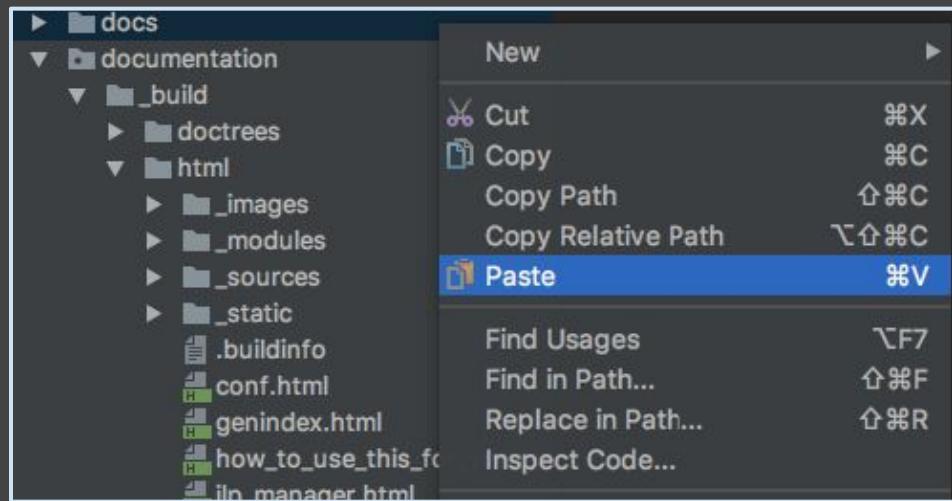
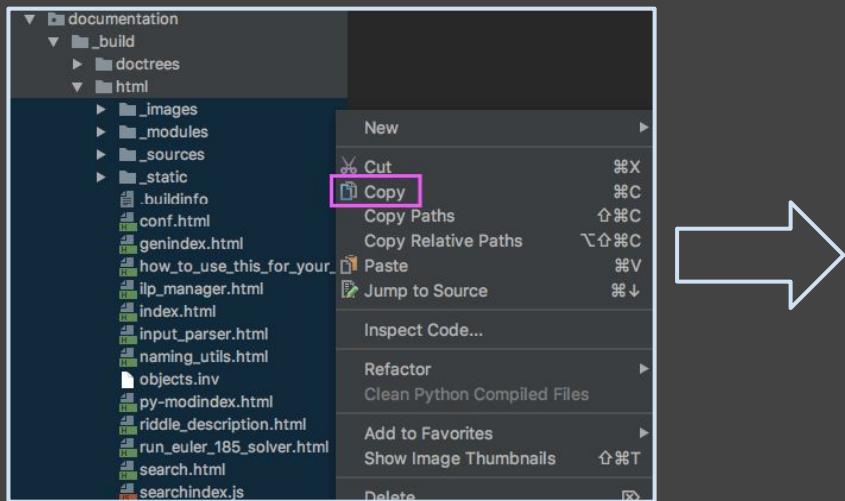
a. Create a folder *docs* under the root of the project



Part III - Document YOUR next project

8. Hosting on GitHub Pages:

a. And copy inside it the content of
documentation/_build/html/

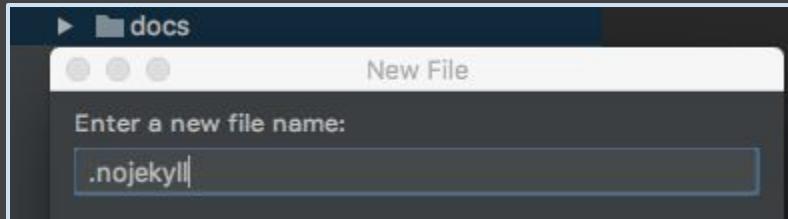


Part III - Document YOUR next project

8. Hosting on GitHub Pages:

b. Create an empty file `.nojekyll` inside `docs` folder

(this tells GitHub Pages to bypass the default jekyll themes and use the html and css in your project)



Part III - Document YOUR next project

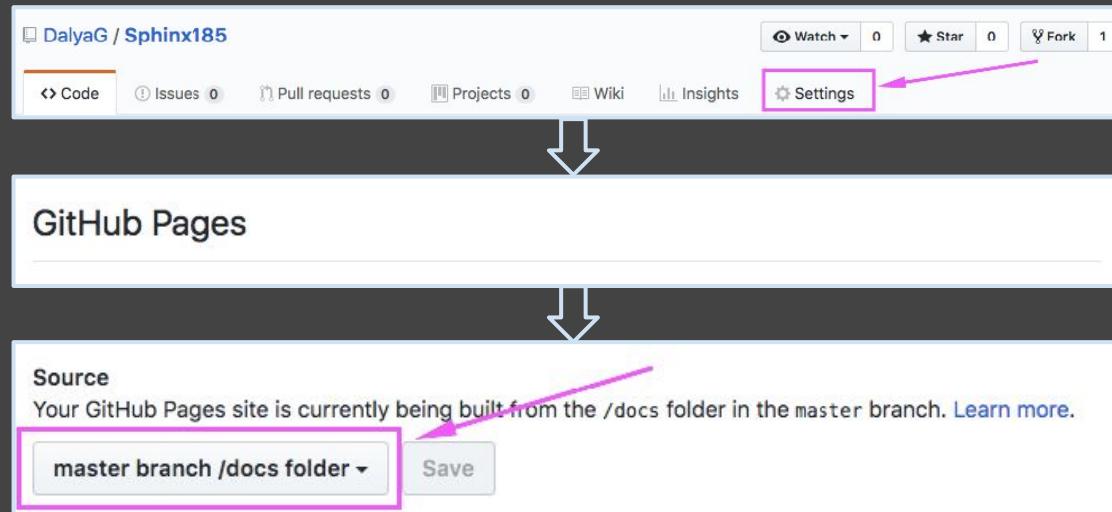
8. Hosting on GitHub Pages:

c. Push your changes to master branch.

Part III - Document YOUR next project

8. Hosting on GitHub Pages:

d. In your repository on GitHub, go to “Settings” -> “GitHub Pages” -> “Source”



Part III - Document YOUR next project

8. Hosting on GitHub Pages:

e. Share your beautiful documentation site at

https://<your_git_username>.github.io/<project_name>/

GitHub Pages

GitHub Pages is designed to host your personal, organization, or project pages from a GitHub repository.

✓ Your site is published at <https://dalyag.github.io/Sphinx185/>

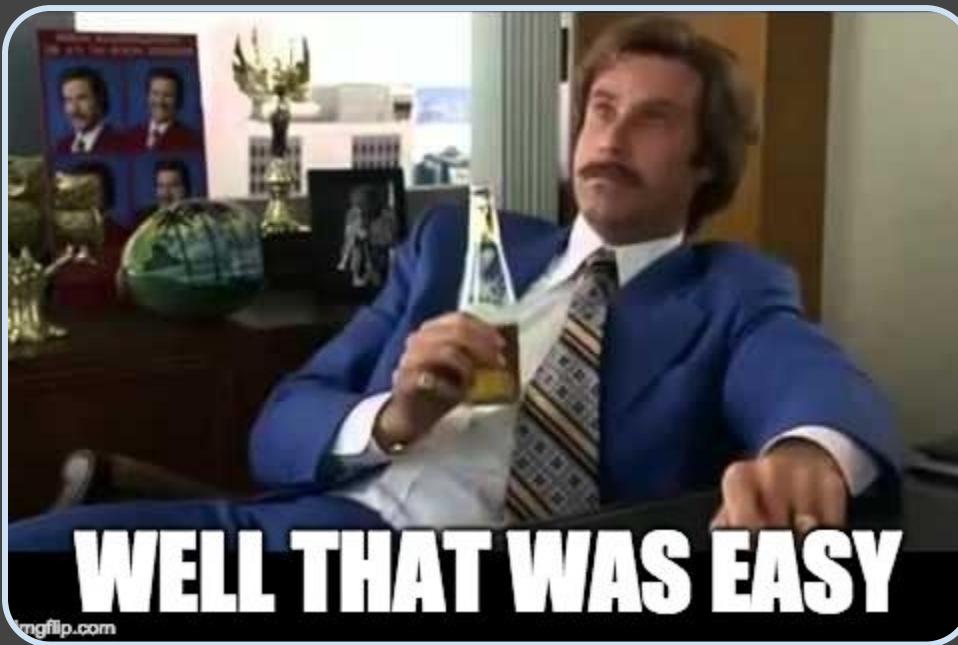
Part III - Document YOUR next project

8. Hosting on GitHub Pages.

e. S
ht

The screenshot shows a GitHub Pages documentation site for a project named "Universe". The main page features a dark header with the project name "Universe" and a "Universe documentation »" link. Below the header is a large image of a galaxy. The main content area has a dark background with white text. It starts with a bold heading "Welcome to YOUR NEXT PROJECT'S documentation !". Below this, there is a text input field with placeholder text: "Put here some introduction to your project." and "The source code is available [here](#)". A bulleted list follows, with the first item being "• [This is a template for including classes](#)". At the bottom of the content area, it says "This documentation was last updated on Jun 01, 2018." On the left side of the main content, there is a sidebar with the title "Universe" and a list item "• [This is a template for including classes](#)". Below the sidebar is a "Quick search" bar with a "Go" button. The footer of the page also contains a "Universe documentation »" link.

Part III - Document YOUR next project



Part III - Recap

WHO CAN DO IT?



YOU CAN DO IT!

Part IV - Surprise! Live Demo!

```
HANDLE*** Address 8016a950 has base at 80100000
.6.2 irq1:if  SYSVER 0xf0000565

Name          Dll Base DateStamp - Name
ntoskrnl.exe 80010000 33247f80  ntkrnlmp1.dll
atapi.sys    80007000 33248040  SIPORT.
Disk.sys     801db000 336015c0  ASS2.SY
Ntfs.sys     80237000 344eeb40  twvid.sy
NTice.sys    f1f48000 31ec6c80  floppy.SY
Cdrom.SYS   f228e000 31ec6c90  null.SYS
KSecDD.SYS  f2290000 335e0000  .SYS
win32k.sys   fe0e2000 34000000  .dll
Cdfs.SYS    fdca2000 335e0000  .sys
.            fdca35000 335e0000  .
```

github.com/DalyaG/Sphinx185

PALPOT.C / SPP
PALPOT.SYS f1dd0000

Take Home Message



Thank you :)

Questions?

DalyaG@gmail.com

