A JUPYTER ENHANCEMENT PROPOSALS

Raniere Silva and Tania Sanchez Monroy (apologies)

EuroPython 2018, Edinburgh, 25 July, 2018

RANIERE SILVA

- rgaiacs
- rgaiacs
- rgaiacs.com

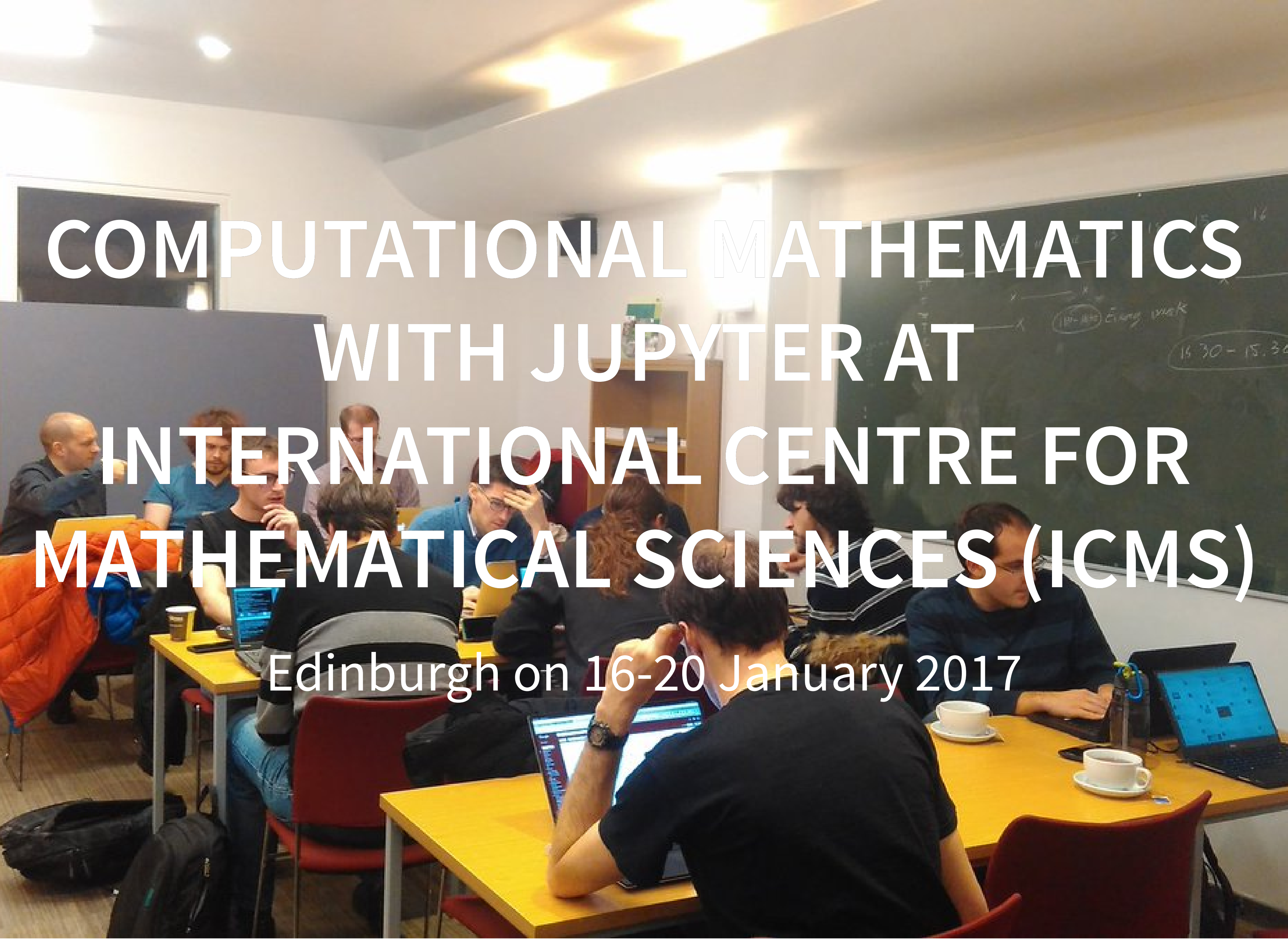
TANIA SANCHEZ MONROY

- ixek
- trallard
- bitsandchips.me

OPENDREAMKIT

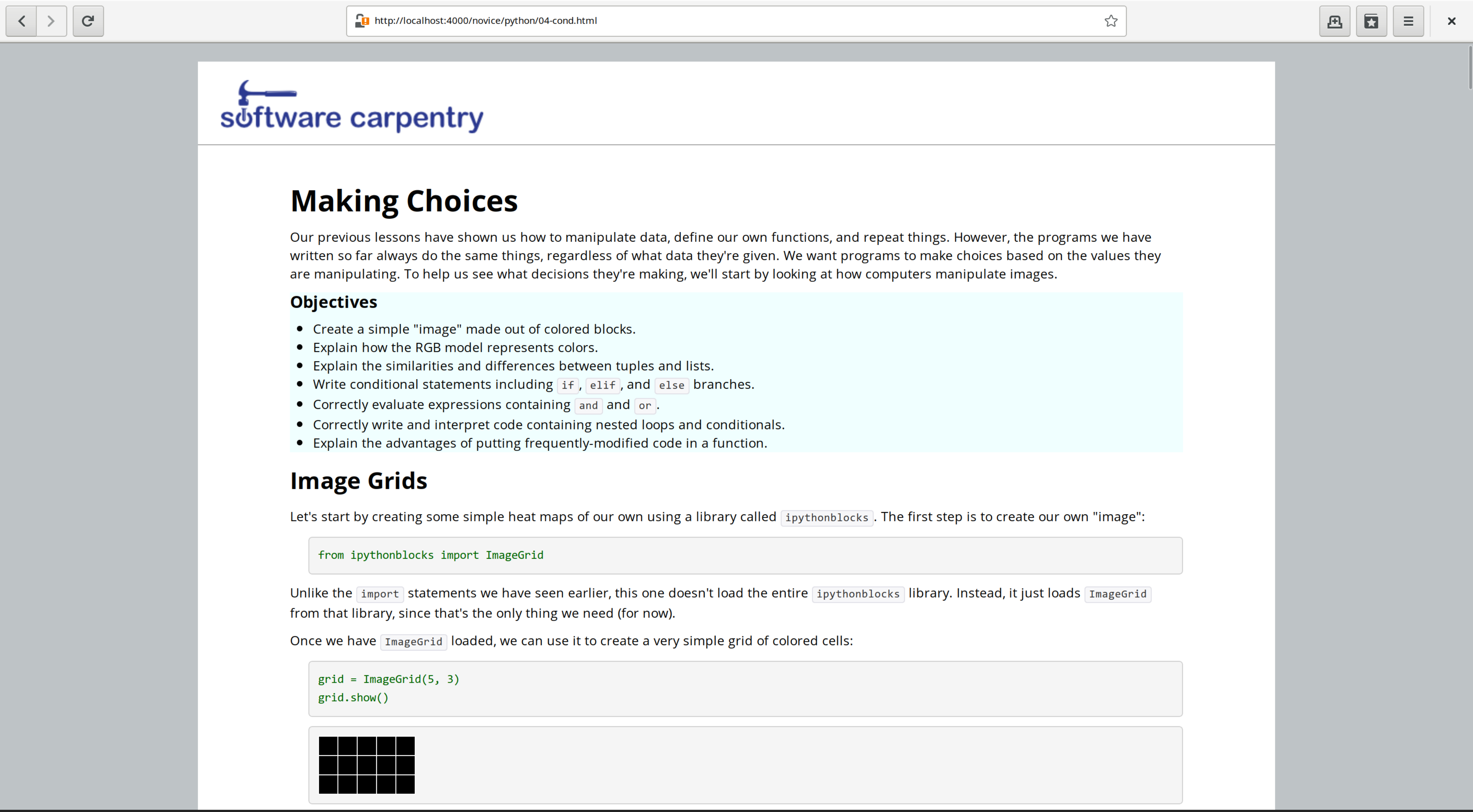
1. "is a Horizon 2020 European Research Infrastructure project (#676541) that will run from Sept. 2015 to August 2019."
2. "[the goal is] to create and strengthen virtual research environments."
3. "advanced the Jupyter Notebook Ecosystem"

More information at https://opendreamkit.org/.



COMPUTATIONAL MATHEMATICS WITH JUPYTER AT INTERNATIONAL CENTRE FOR MATHEMATICAL SCIENCES (ICMS)

Edinburgh on 16-20 January 2017



Making Choices

Our previous lessons have shown us how to manipulate data, define our own functions, and repeat things. However, the programs we have written so far always do the same things, regardless of what data they're given. We want programs to make choices based on the values they are manipulating. To help us see what decisions they're making, we'll start by looking at how computers manipulate images.

Objectives

- Create a simple "image" made out of colored blocks.
- Explain how the RGB model represents colors.
- Explain the similarities and differences between tuples and lists.
- Write conditional statements including `if`, `elif`, and `else` branches.
- Correctly evaluate expressions containing `and` and `or`.
- Correctly write and interpret code containing nested loops and conditionals.
- Explain the advantages of putting frequently-modified code in a function.

Image Grids

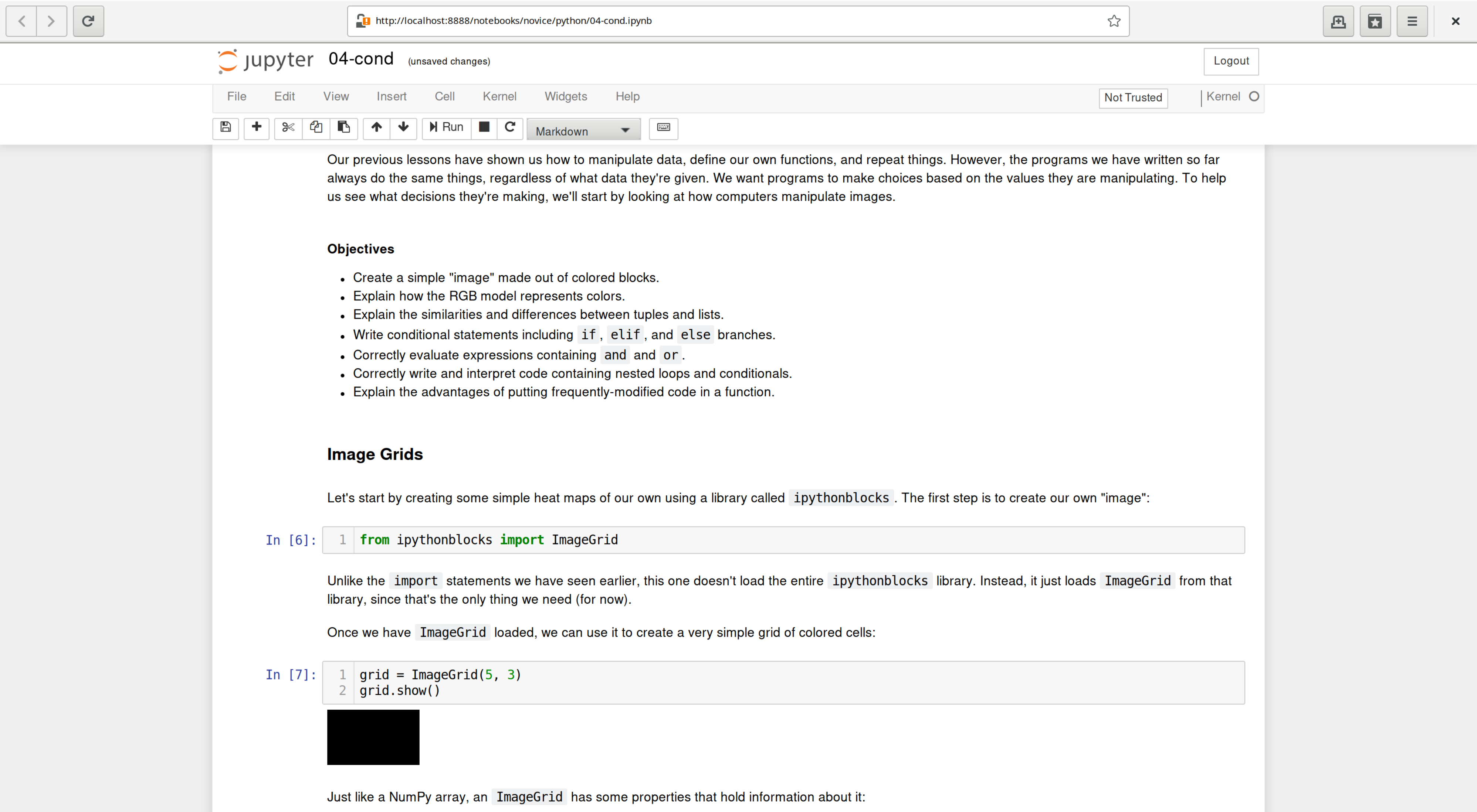
Let's start by creating some simple heat maps of our own using a library called `ipythonblocks`. The first step is to create our own "image":

```
from ipythonblocks import ImageGrid
```

Unlike the `import` statements we have seen earlier, this one doesn't load the entire `ipythonblocks` library. Instead, it just loads `ImageGrid` from that library, since that's the only thing we need (for now).

Once we have `ImageGrid` loaded, we can use it to create a very simple grid of colored cells:

```
grid = ImageGrid(5, 3)
grid.show()
```


http://localhost:8888/notebooks/novice/python/04-cond.ipynb
jupyter 04-cond (unsaved changes)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Kernel
Run
Markdown

Our previous lessons have shown us how to manipulate data, define our own functions, and repeat things. However, the programs we have written so far always do the same things, regardless of what data they're given. We want programs to make choices based on the values they are manipulating. To help us see what decisions they're making, we'll start by looking at how computers manipulate images.

Objectives

- Create a simple "image" made out of colored blocks.
- Explain how the RGB model represents colors.
- Explain the similarities and differences between tuples and lists.
- Write conditional statements including `if`, `elif`, and `else` branches.
- Correctly evaluate expressions containing `and` and `or`.
- Correctly write and interpret code containing nested loops and conditionals.
- Explain the advantages of putting frequently-modified code in a function.

Image Grids

Let's start by creating some simple heat maps of our own using a library called `ipythonblocks`. The first step is to create our own "image":

In [6]:

```
from ipythonblocks import ImageGrid
```

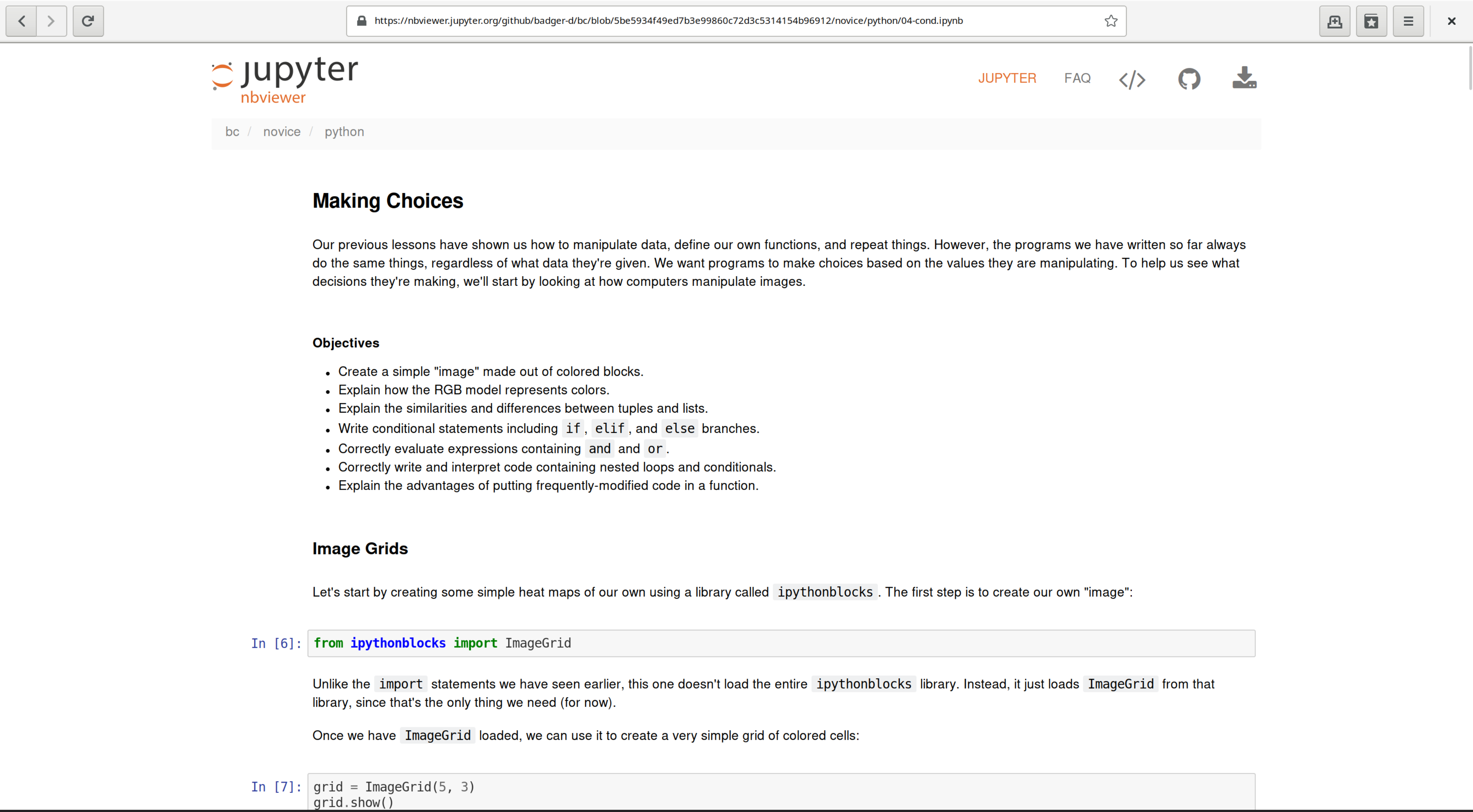
Unlike the `import` statements we have seen earlier, this one doesn't load the entire `ipythonblocks` library. Instead, it just loads `ImageGrid` from that library, since that's the only thing we need (for now).

Once we have `ImageGrid` loaded, we can use it to create a very simple grid of colored cells:

In [7]:

```
grid = ImageGrid(5, 3)
grid.show()
```

Just like a NumPy array, an `ImageGrid` has some properties that hold information about it:


https://nbviewer.jupyter.org/github/badger-d/bc/blob/5be5934f49ed7b3e99860c72d3c5314154b96912/novice/python/04-cond.ipynb
jupyter
nbviewer
JUPYTER
FAQ
bc / novice / python

Making Choices

Our previous lessons have shown us how to manipulate data, define our own functions, and repeat things. However, the programs we have written so far always do the same things, regardless of what data they're given. We want programs to make choices based on the values they are manipulating. To help us see what decisions they're making, we'll start by looking at how computers manipulate images.

Objectives

- Create a simple "image" made out of colored blocks.
- Explain how the RGB model represents colors.
- Explain the similarities and differences between tuples and lists.
- Write conditional statements including `if`, `elif`, and `else` branches.
- Correctly evaluate expressions containing `and` and `or`.
- Correctly write and interpret code containing nested loops and conditionals.
- Explain the advantages of putting frequently-modified code in a function.

Image Grids

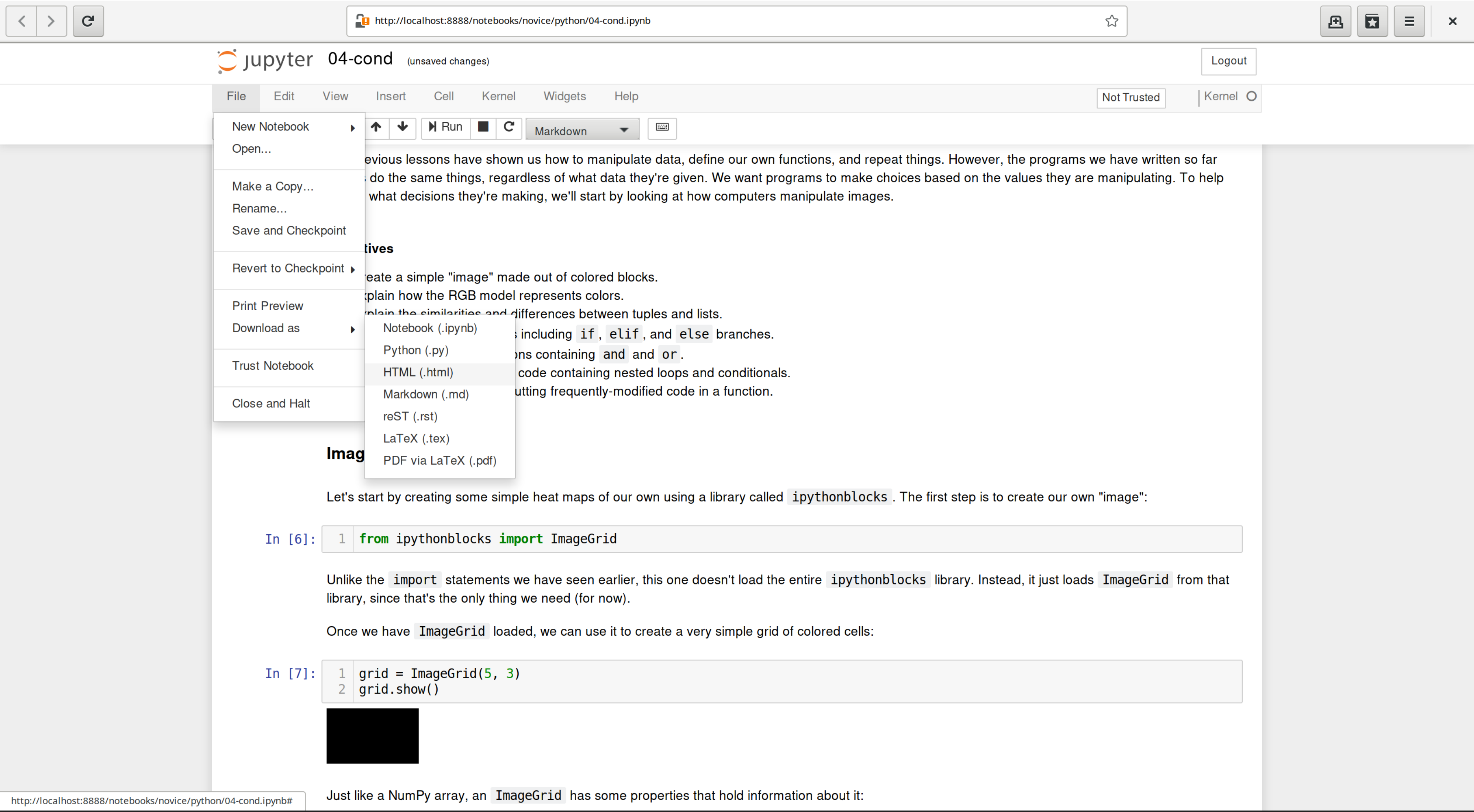
Let's start by creating some simple heat maps of our own using a library called `ipythonblocks`. The first step is to create our own "image":

In [6]:
```
from ipythonblocks import ImageGrid
```

Unlike the `import` statements we have seen earlier, this one doesn't load the entire `ipythonblocks` library. Instead, it just loads `ImageGrid` from that library, since that's the only thing we need (for now).

Once we have `ImageGrid` loaded, we can use it to create a very simple grid of colored cells:

In [7]:
```
grid = ImageGrid(5, 3)
grid.show()
```



evious lessons have shown us how to manipulate data, define our own functions, and repeat things. However, the programs we have written so far do the same things, regardless of what data they're given. We want programs to make choices based on the values they are manipulating. To help what decisions they're making, we'll start by looking at how computers manipulate images.

tives

- eate a simple "image" made out of colored blocks.
- plain how the RGB model represents colors.
- plain the similarities and differences between tuples and lists.
- including `if`, `elif`, and `else` branches.
- ons containing `and` and `or`.
- code containing nested loops and conditionals.
- utting frequently-modified code in a function.

Imag

Let's start by creating some simple heat maps of our own using a library called `ipythonblocks`. The first step is to create our own "image":

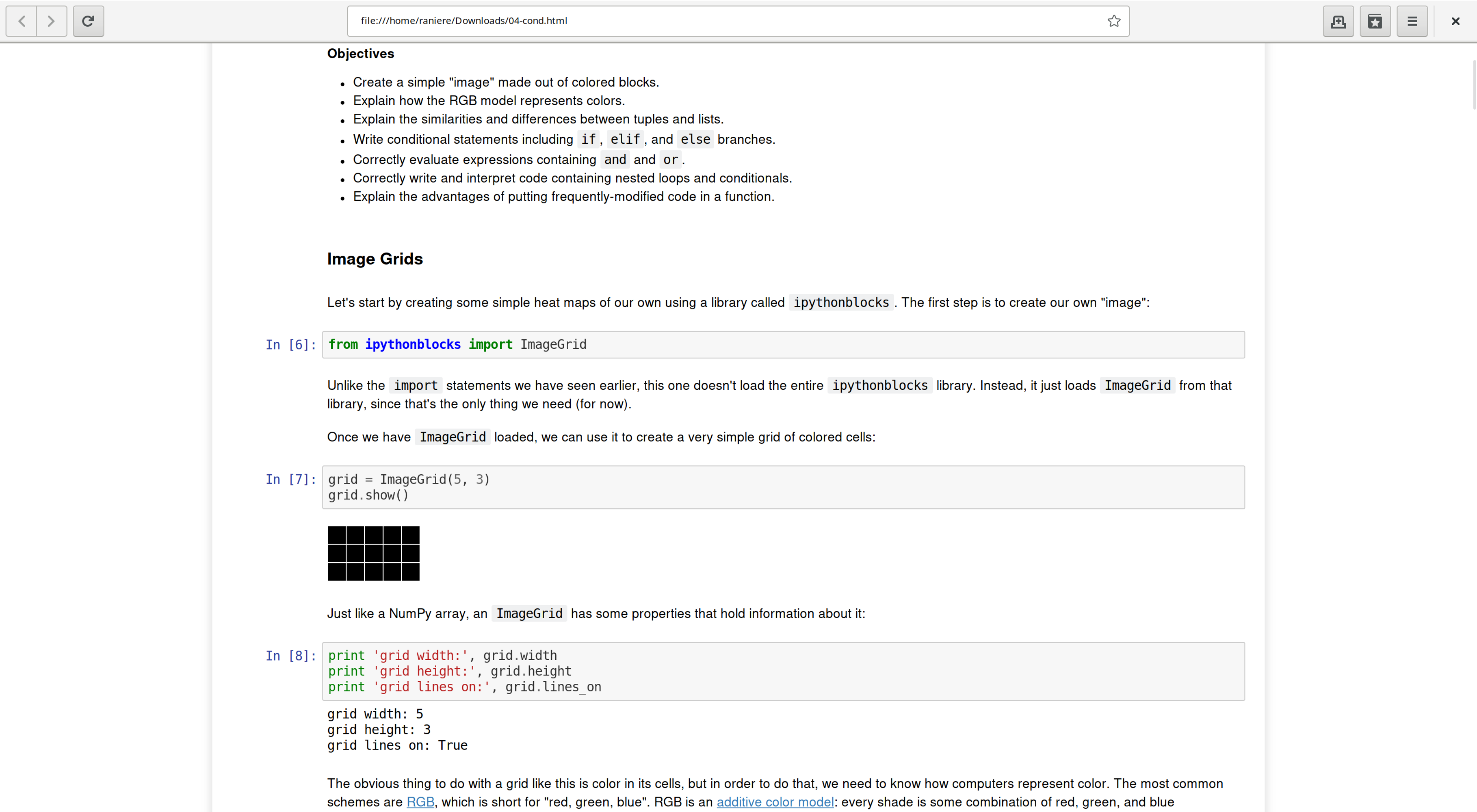
In [6]:
```
from ipythonblocks import ImageGrid
```

Unlike the `import` statements we have seen earlier, this one doesn't load the entire `ipythonblocks` library. Instead, it just loads `ImageGrid` from that library, since that's the only thing we need (for now).

Once we have `ImageGrid` loaded, we can use it to create a very simple grid of colored cells:

In [7]:
```
grid = ImageGrid(5, 3)
grid.show()
```

Just like a NumPy array, an `ImageGrid` has some properties that hold information about it:



Objectives

- Create a simple "image" made out of colored blocks.
- Explain how the RGB model represents colors.
- Explain the similarities and differences between tuples and lists.
- Write conditional statements including `if`, `elif`, and `else` branches.
- Correctly evaluate expressions containing `and` and `or`.
- Correctly write and interpret code containing nested loops and conditionals.
- Explain the advantages of putting frequently-modified code in a function.

Image Grids

Let's start by creating some simple heat maps of our own using a library called `ipythonblocks`. The first step is to create our own "image":

In [6]:
```
from ipythonblocks import ImageGrid
```

Unlike the `import` statements we have seen earlier, this one doesn't load the entire `ipythonblocks` library. Instead, it just loads `ImageGrid` from that library, since that's the only thing we need (for now).

Once we have `ImageGrid` loaded, we can use it to create a very simple grid of colored cells:

In [7]:
```
grid = ImageGrid(5, 3)
grid.show()
```

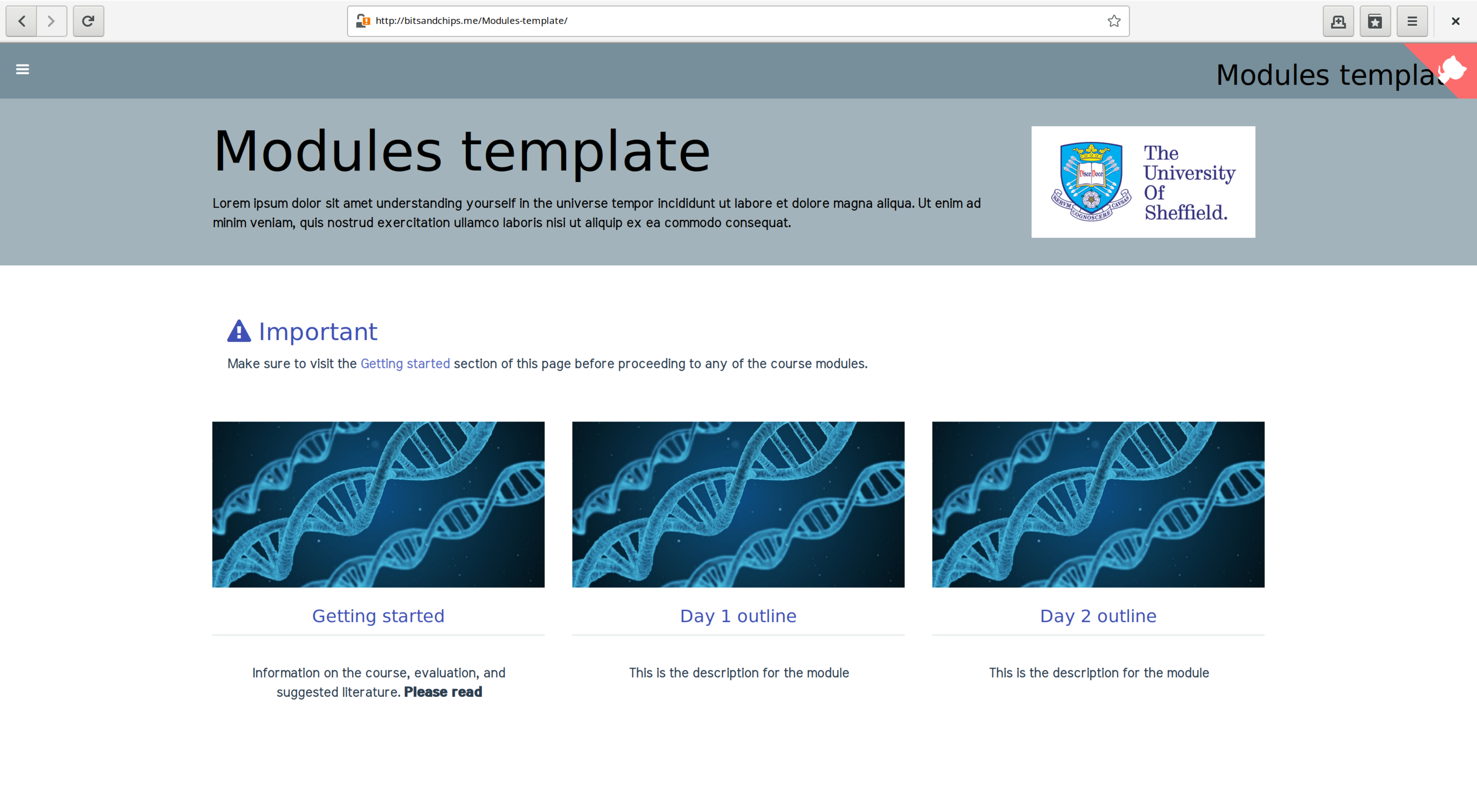
■	■	■	■	■
■	■	■	■	■
■	■	■	■	■

Just like a NumPy array, an `ImageGrid` has some properties that hold information about it:

In [8]:
```
print 'grid width:', grid.width
print 'grid height:', grid.height
print 'grid lines on:', grid.lines_on
```

```
grid width: 5
grid height: 3
grid lines on: True
```

The obvious thing to do with a grid like this is color in its cells, but in order to do that, we need to know how computers represent color. The most common schemes are RGB, which is short for "red, green, blue". RGB is an additive color model: every shade is some combination of red, green, and blue



Modules template

Lorem ipsum dolor sit amet understanding yourself in the universe tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat.

⚠ Important

Make sure to visit the Getting started section of this page before proceeding to any of the course modules.

Getting started

Information on the course, evaluation, and suggested literature. **Please read**

Day 1 outline

This is the description for the module

Day 2 outline

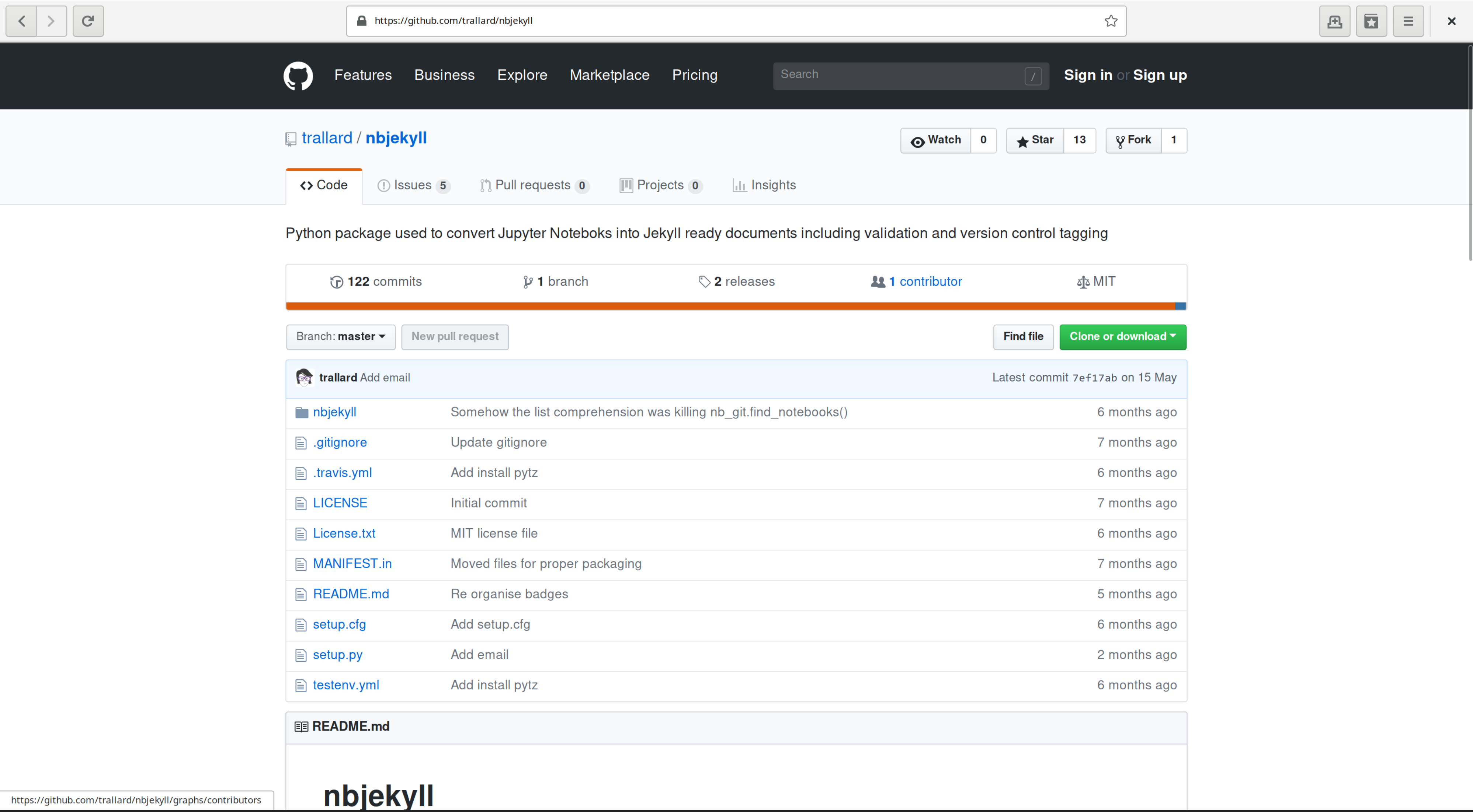
This is the description for the module

WHY JUPYTER NOTEBOOKS

- Environment familiar to authors
- Support for over 40 programming languages (by the time of this talk)

PIECES OF THE PIPELINE

- nbconvert
- static site generators, e.g. Sphinx, Hugo, Jekyll, …



trallard / nbjekyll

Python package used to convert Jupyter Noteboks into Jekyll ready documents including validation and version control tagging

122 commits | 1 branch | 2 releases | 1 contributor | MIT

trallard Add email — Latest commit 7ef17ab on 15 May

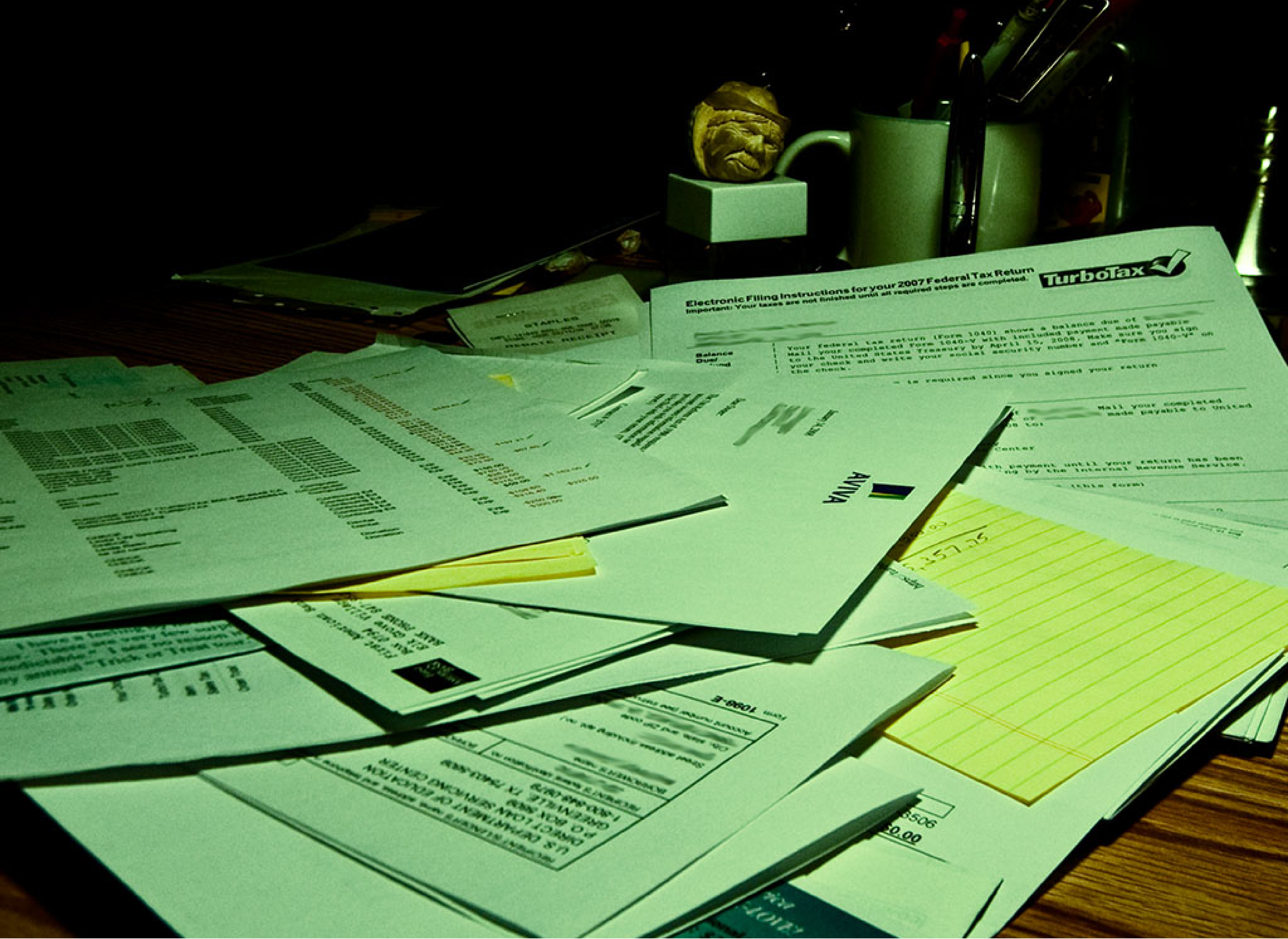
nbjekyll	Somehow the list comprehension was killing nb_git.find_notebooks()	6 months ago
.gitignore	Update gitignore	7 months ago
.travis.yml	Add install pytz	6 months ago
LICENSE	Initial commit	7 months ago
License.txt	MIT license file	6 months ago
MANIFEST.in	Moved files for proper packaging	7 months ago
README.md	Re organise badges	5 months ago
setup.cfg	Add setup.cfg	6 months ago
setup.py	Add email	2 months ago
testenv.yml	Add install pytz	6 months ago

README.md

nbjekyll

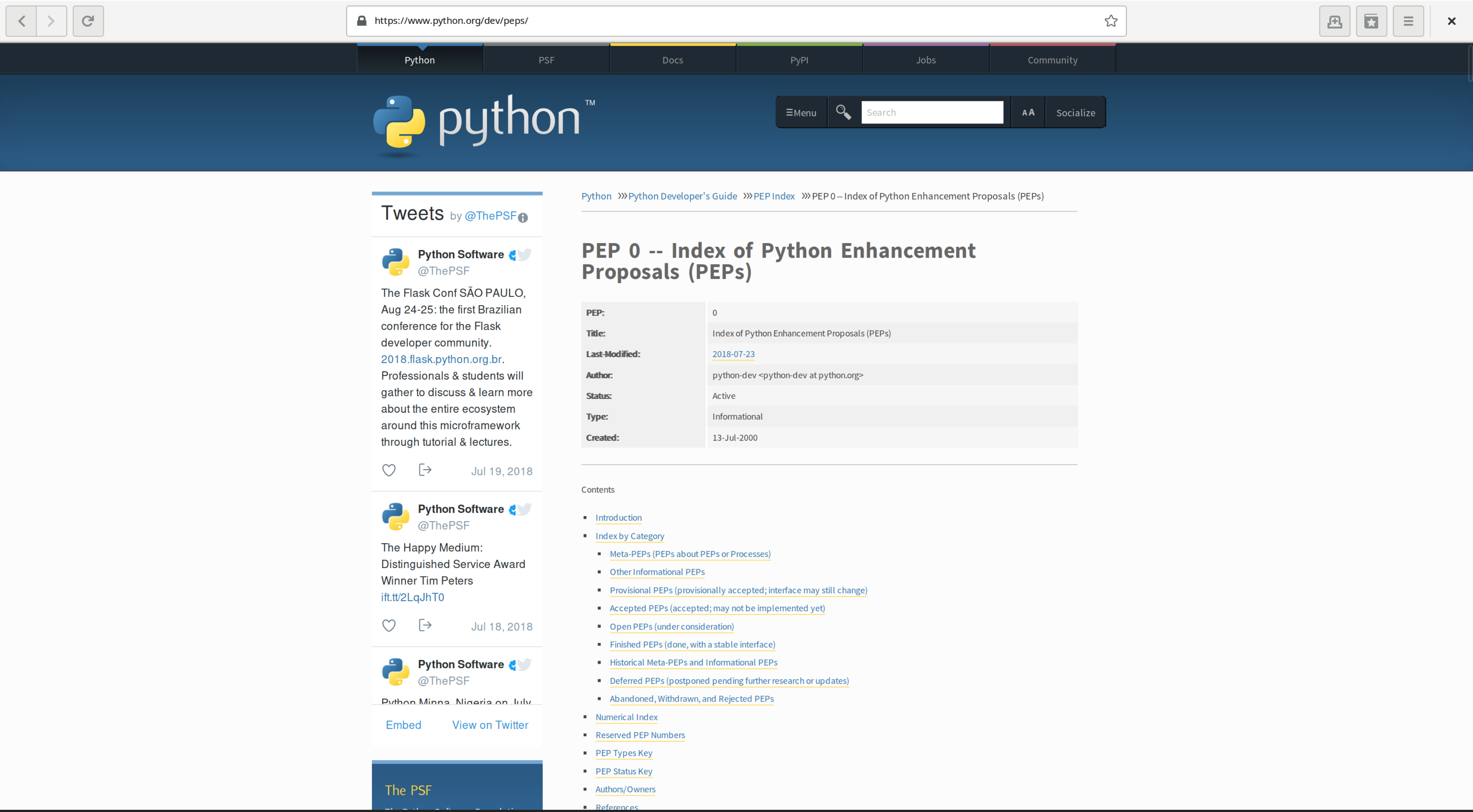
POLICE PUBLIC CALL BOX
POLICE PUBLIC CALL BOX
POLICE TELEPHONE
FREE
FOR USE OF
PUBLIC
ADVICE & ASSISTANCE
OBTAINABLE IMMEDIATELY
OFFICERS & CARS
RESPOND TO ALL CALLS
PULL TO OPEN
ST JOHN AMBULANCE

2 SALA DELLA PIAZZA HALL
TOILET
3 SALA DELLA PIAZZA HALL
Tony Andersen
ThinkPad


TurboTax
Electronic Filing Instructions for your 2007 Federal Tax Return
Important: Your taxes are not finished until all required steps are completed.
Balance Due/
Your federal tax return (Form 1040) shows a balance due of [redacted]
Mail your completed Form 1040-V with included payment made payable
to the United States Treasury by April 15, 2008. Make sure you sign
your check and write your social security number and "Form 1040-V" on
the check.
is required since you signed your return
Mail your completed
made payable to United
Center
payment until your return has been
by the Internal Revenue Service.
(this form)
STAPLES
REBATE RECEIPT
AVIVA
Form 1098-E
U.S. DEPARTMENT OF EDUCATION
DIRECT LOAN SERVICING CENTER
P.O. BOX 5609
GREENVILLE, TX 75403-5609

DEMO TIME

```
jupyter nbconvert --
template=custob.tpl notebook.ipynb
```



Python »» Python Developer's Guide »» PEP Index »» PEP 0 -- Index of Python Enhancement Proposals (PEPs)

PEP 0 -- Index of Python Enhancement Proposals (PEPs)

PEP:	0
Title:	Index of Python Enhancement Proposals (PEPs)
Last-Modified:	2018-07-23
Author:	python-dev <python-dev at python.org>
Status:	Active
Type:	Informational
Created:	13-Jul-2000

Contents

Tweets by @ThePSF

Python Software @ThePSF

The Flask Conf SÃO PAULO, Aug 24-25: the first Brazilian conference for the Flask developer community. 2018.flask.python.org.br. Professionals & students will gather to discuss & learn more about the entire ecosystem around this microframework through tutorial & lectures.

Jul 19, 2018

Python Software @ThePSF

The Happy Medium: Distinguished Service Award Winner Tim Peters ift.tt/2LqJhT0

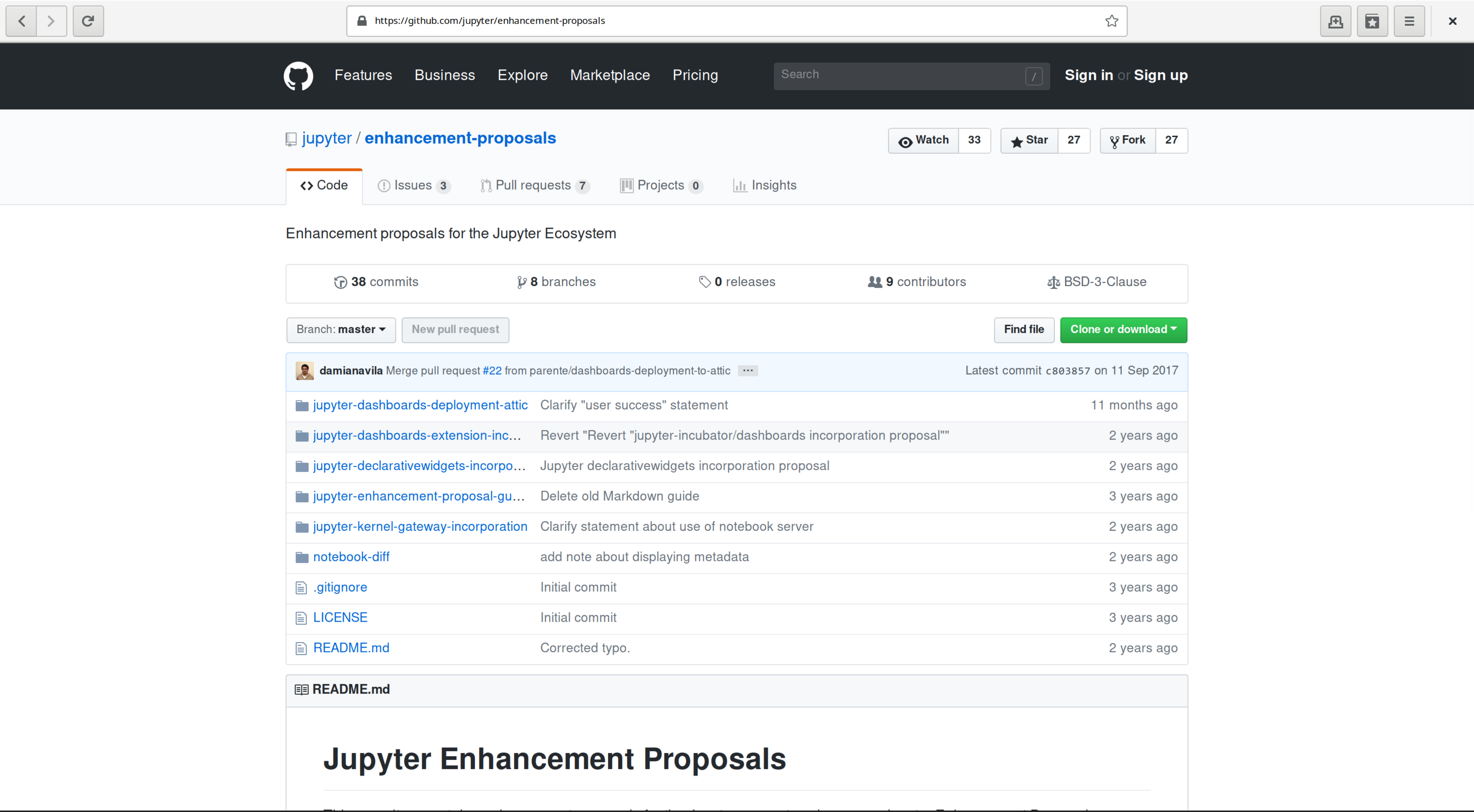
Jul 18, 2018

Python Software @ThePSF

Python Minna, Nigeria on July

Embed View on Twitter

The PSF



jupyter / **enhancement-proposals**

Enhancement proposals for the Jupyter Ecosystem

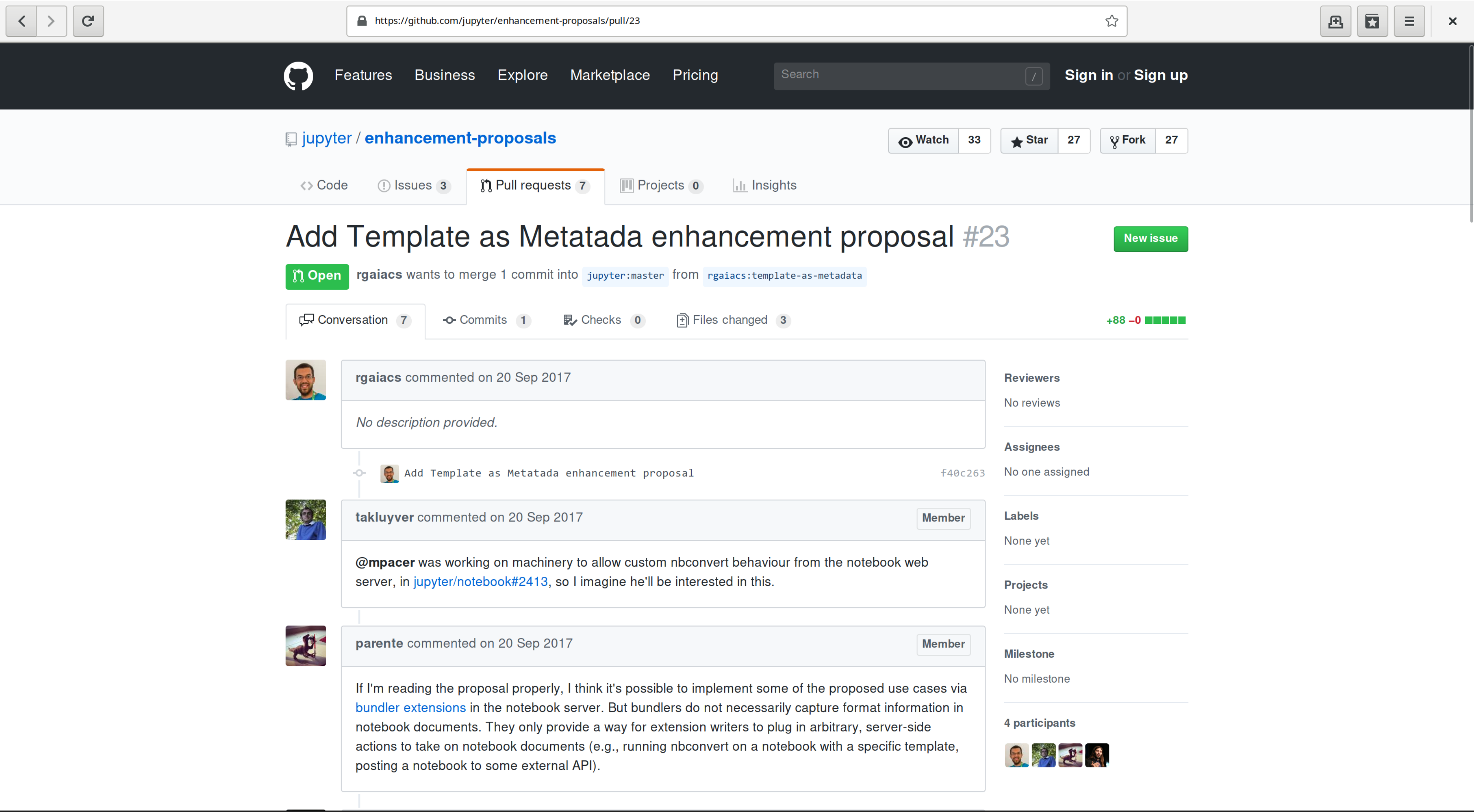
38 commits | **8** branches | **0** releases | **9** contributors | BSD-3-Clause

damianavila Merge pull request #22 from parente/dashboards-deployment-to-attic … Latest commit `c803857` on 11 Sep 2017

jupyter-dashboards-deployment-attic	Clarify "user success" statement	11 months ago
jupyter-dashboards-extension-inc…	Revert "Revert "jupyter-incubator/dashboards incorporation proposal""	2 years ago
jupyter-declarativewidgets-incorpo…	Jupyter declarativewidgets incorporation proposal	2 years ago
jupyter-enhancement-proposal-gu…	Delete old Markdown guide	3 years ago
jupyter-kernel-gateway-incorporation	Clarify statement about use of notebook server	2 years ago
notebook-diff	add note about displaying metadata	2 years ago
.gitignore	Initial commit	3 years ago
LICENSE	Initial commit	3 years ago
README.md	Corrected typo.	2 years ago

README.md

Jupyter Enhancement Proposals



Add Template as Metatada enhancement proposal #23

Open **rgaiacs** wants to merge 1 commit into `jupyter:master` from `rgaiacs:template-as-metadata`

rgaiacs commented on 20 Sep 2017

No description provided.

`Add Template as Metatada enhancement proposal` `f40c263`

takluyver commented on 20 Sep 2017 Member

@mpacer was working on machinery to allow custom nbconvert behaviour from the notebook web server, in jupyter/notebook#2413, so I imagine he'll be interested in this.

parente commented on 20 Sep 2017 Member

If I'm reading the proposal properly, I think it's possible to implement some of the proposed use cases via bundler extensions in the notebook server. But bundlers do not necessarily capture format information in notebook documents. They only provide a way for extension writers to plug in arbitrary, server-side actions to take on notebook documents (e.g., running nbconvert on a notebook with a specific template, posting a notebook to some external API).

Reviewers
No reviews

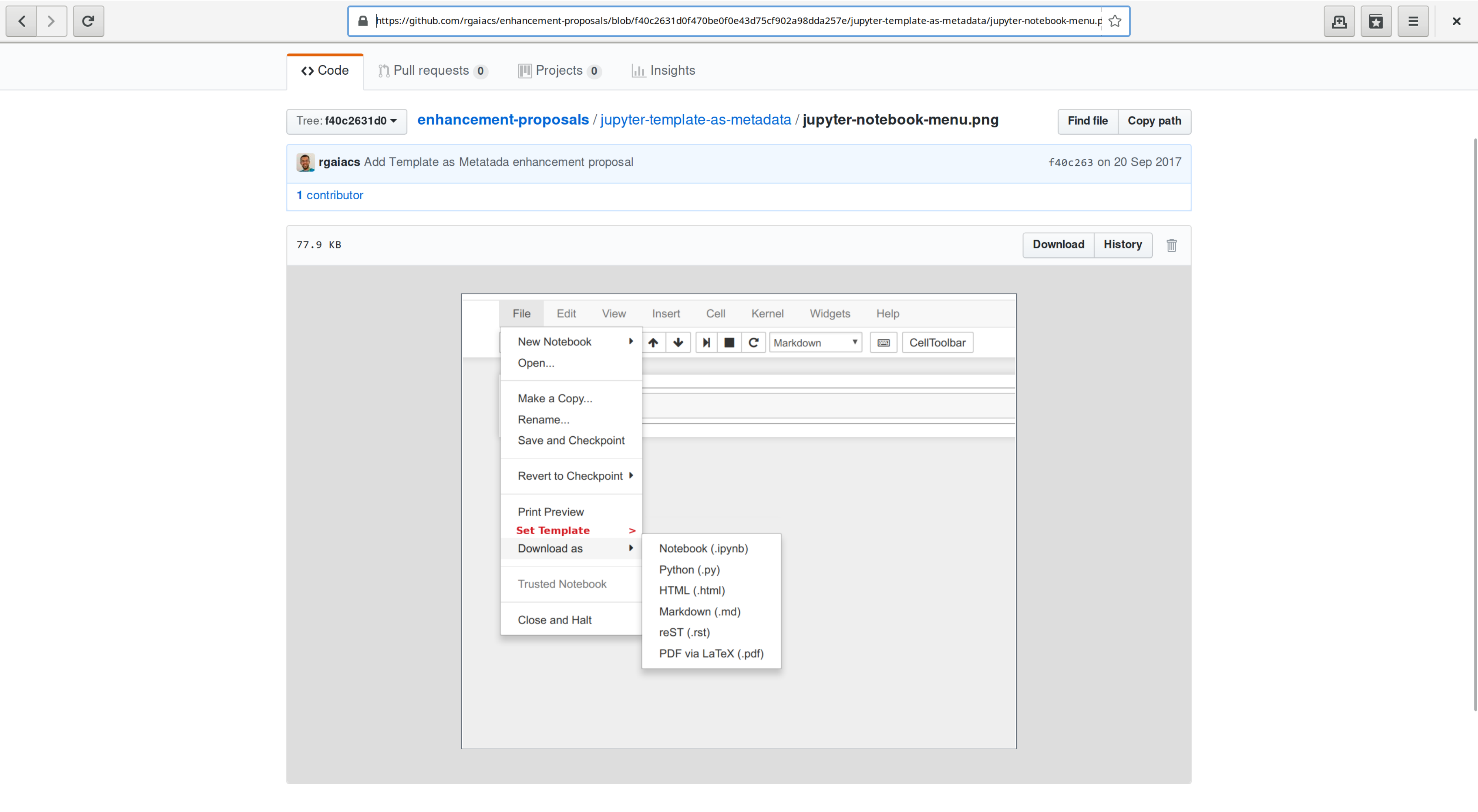
Assignees
No one assigned

Labels
None yet

Projects
None yet

Milestone
No milestone

4 participants



https://github.com/rgaiacs/enhancement-proposals/blob/f40c2631d0f470be0f0e43d75cf902a98dda257e/jupyter-template-as-metadata/jupyter-notebook-menu.p

Tree: f40c2631d0 enhancement-proposals / jupyter-template-as-metadata / jupyter-notebook-menu.png

rgaiacs Add Template as Metatada enhancement proposal f40c263 on 20 Sep 2017

1 contributor

77.9 KB

THANKS!!! AND ASK ME QUESTIONS.

- Raniere: 🐦 rgaiacs
- Tania: 🐦 ixek