

COMP20270

Python Object Oriented Programming

5 ECTS

Delivered over 6 weeks

Prof. Pádraig Cunningham (padraig.cunningham@ucd.ie)

OOP
in Python
and
Python
Notebooks

Employee Payroll Classes

Three simple classes showing, Encapsulation,

```
In [ ]: class Employee():
        def __init__(self, name):
            self.name = name

        class HourlyPaidEmployee(Employee):

            def __init__(self, name):
                Employee.__init__(self, name)
                self.hours = 0
                self.rate = 0

            def set_hours(self, hours):
                self.hours = hours
```

COMP20270 Structure



■ Moodle Page

- <https://csmoodle.ucd.ie/moodle/course/view.php?id=696>
- Student enrolment key 'poop2018'

■ Labs & Lectures

- Two 1 hour lectures and one 2 hour lecture
 - Hands-on, working on Python Notebooks
- 1 x 2 hour lab
 - Worksheets - not graded

■ Assessment

- In-class test Wk 9 (20%)
- 1st Assignment Due Monday of Wk 10 (15%) (12th Nov)
- 2nd Assignment Due Monday of Wk 12 (15%) (26th Nov)
- End of term exam (50%)
- Due Monday 5th March

Grading Scheme



CS Grading Scheme applies for this module - pass mark is 40%

Grade	Min	Max
A+	95	100
A	90	95
A-	85	90
B+	80	85
B	75	80
B-	70	75
C+	65	70
C	60	65
C-	55	60
D+	50	55
D	45	50
D-	40	45

Grade	Min	Max
E+	35	40
E	30	35
E-	25	30
F+	20	25
F	15	20
F-	10	15
G+	8	10
G	5	8
G-	2	5
NG	0	0

<https://www.cs.ucd.ie/Grading>

Plagiarism Policy

- Plagiarism is a **serious academic offence**.
- Our staff and demonstrators are proactive in looking for possible plagiarism in all submitted work.
- Suspected plagiarism is reported to the CS Plagiarism Subcommittee for investigation:
 - Usually includes an interview with student(s) involved
 - 1st offence: **usually** 0 or NG in the affected components
 - 2nd offence: referred to **University Disciplinary Committee**
- Students who enable plagiarism are equally responsible. See:
http://www.ucd.ie/registry/academicsecretariat/docs/plagiarism_po.pdf
http://www.ucd.ie/registry/academicsecretariat/docs/student_code.pdf
<http://libguides.ucd.ie/academicintegrity>

Installing Python

- In the module we will use **Python 3.7**
- Python 3.x is recommended for new code and fixes many of the issues and inconsistencies from Python 2.
- Be aware: Python 3.x code is not fully backwards compatible with Python 2.
- Install Python via the **Anaconda** distribution which provides a version of Python tailored for data analytics, with easy installation of many third party packages.

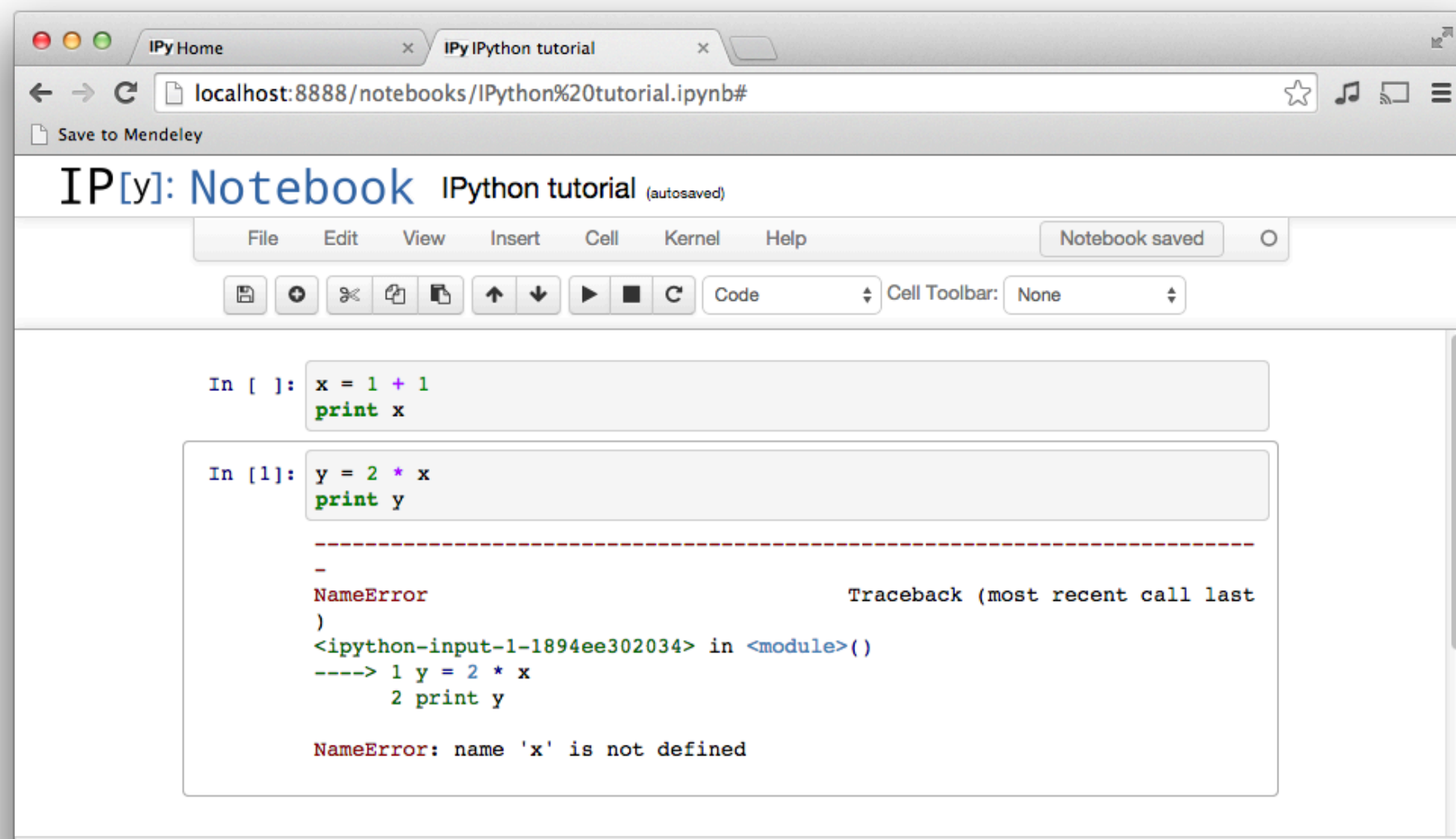
<https://www.anaconda.com/download/>



- ◆ Download and run Anaconda the graphical or terminal installer for Python 3.7 (not 2.7!) for your operating system
 - Windows, OSX or Linux.

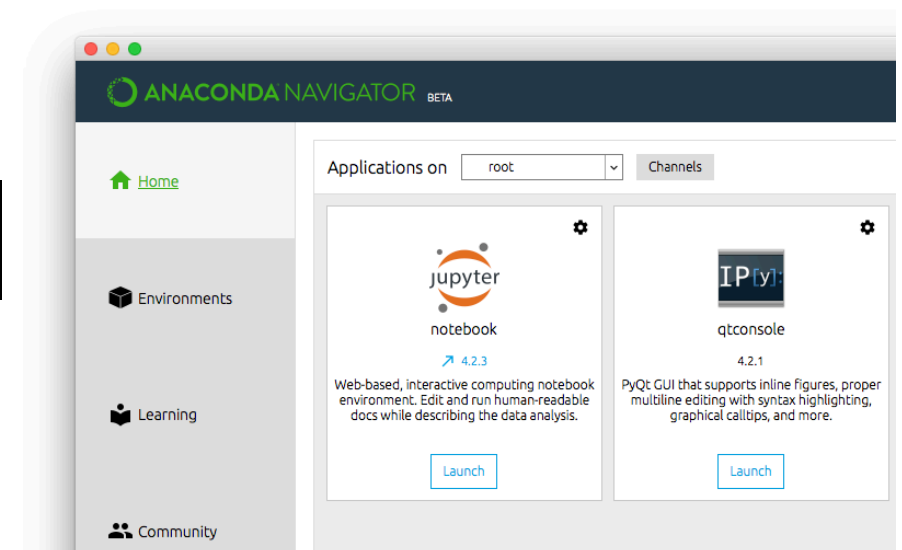
Jupyter and IPython Notebooks

- **Basic idea:** Rather than using an editor or development environment, use an interactive browser-based environment to learn programming.
- Online notebooks are now increasingly used in commercial settings (e.g. data science teams).



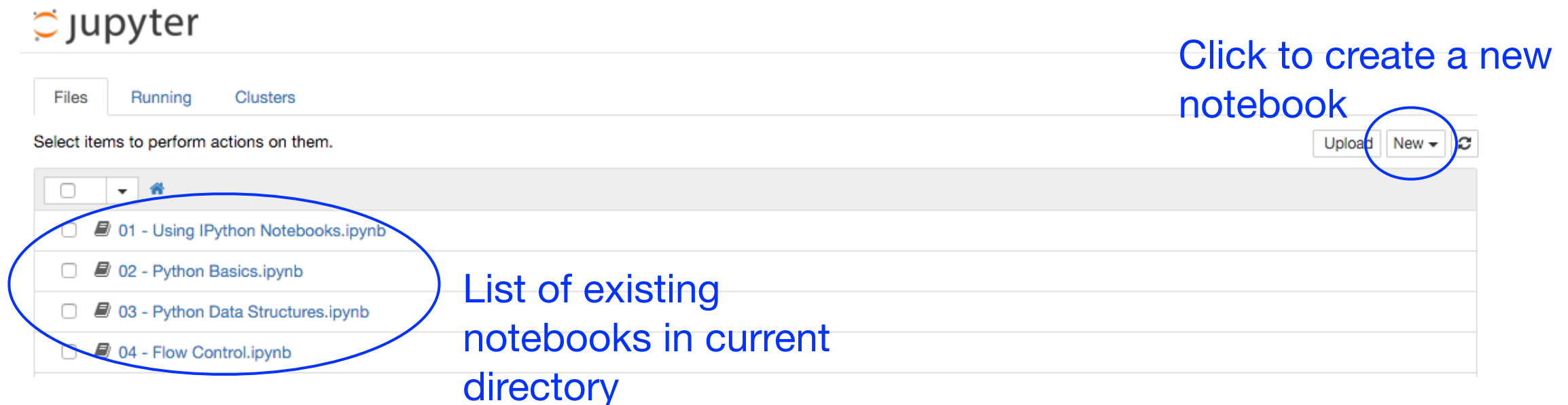
Jupyter and Python Notebooks

- **Jupyter project:** a web application for interactive data science and scientific computing.
- **IPython Notebooks:** an engine for running Python code under the Jupyter system.
- We will use IPython Notebooks for many of the labs and assignments.
- To start the Notebook server, either:
 1. In the terminal, type `jupyter notebook`
 2. Or click the Anaconda Navigator icon, then choose **jupyter-notebook** from the list of apps.
- This should load the IPython Notebook dashboard in your browser. Later you can also manually go to <http://localhost:8888>



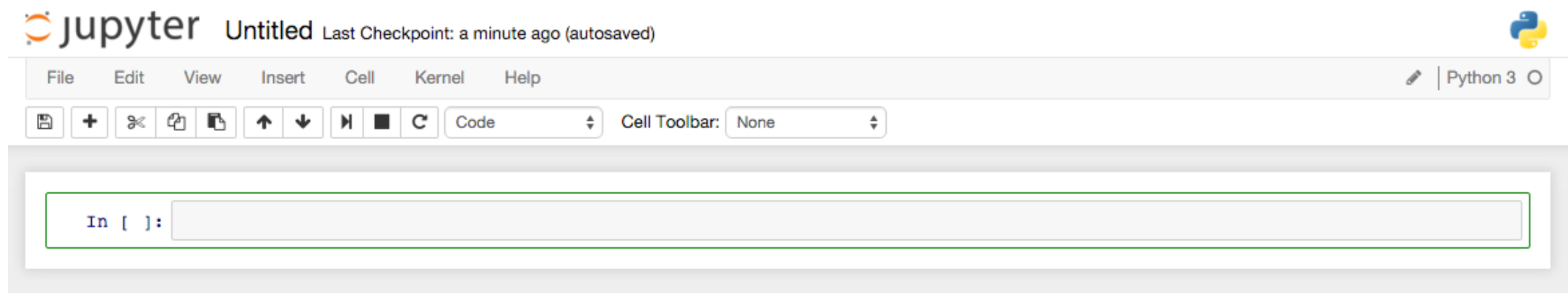
Notebook Dashboard

- The IPython dashboard provides a mini filesystem interface for creating and accessing notebooks.
- Note: The dashboard shows notebooks in the directory where you launched the notebook server.



The screenshot shows the Jupyter Notebook Dashboard. At the top, there's a 'jupyter' logo and tabs for 'Files', 'Running', and 'Clusters'. Below the tabs, it says 'Select items to perform actions on them.' On the right, there are buttons for 'Upload', 'New', and a refresh icon. A blue circle highlights the 'New' button with the text 'Click to create a new notebook'. Below the buttons, there's a list of notebooks: '01 - Using IPython Notebooks.ipynb', '02 - Python Basics.ipynb', '03 - Python Data Structures.ipynb', and '04 - Flow Control.ipynb'. A blue circle highlights this list with the text 'List of existing notebooks in current directory'.

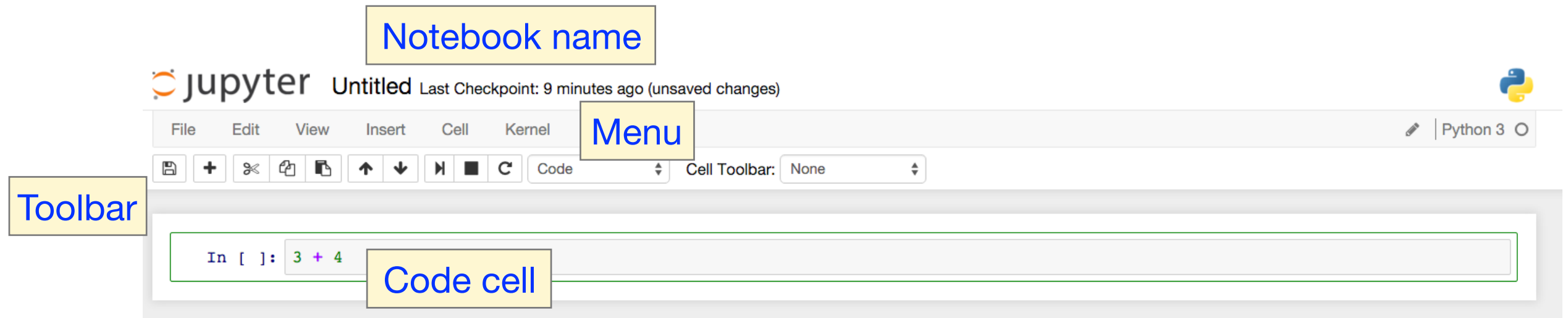
- To start writing code, create **New** → **Python 3 Notebook**



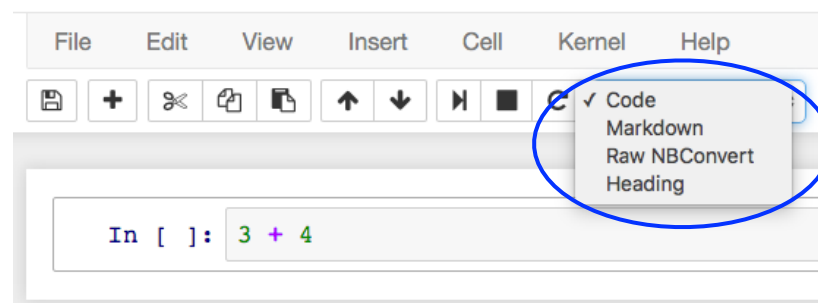
The screenshot shows the Jupyter Notebook Editor. At the top, there's a 'jupyter' logo and the text 'Untitled Last Checkpoint: a minute ago (autosaved)'. Below the logo, there's a menu bar with 'File', 'Edit', 'View', 'Insert', 'Cell', 'Kernel', and 'Help'. On the right, there's a Python logo and the text 'Python 3'. Below the menu bar, there's a toolbar with icons for saving, adding, deleting, and other actions. Below the toolbar, there's a text area with the prompt 'In []:' and a large input field for writing code.

Notebook Interface

- When you create a new notebook, you will be presented with the notebook name, a menu bar, a toolbar and an empty code cell.



- IPython notebooks have two fundamental types of cells:
 1. **Markdown cells:** Contain text content for explaining a notebook.
 2. **Code cells:** Allow you to type and run Python code.



Every new cell starts off being a code cell. But this can be changed by using the drop-down on the toolbar

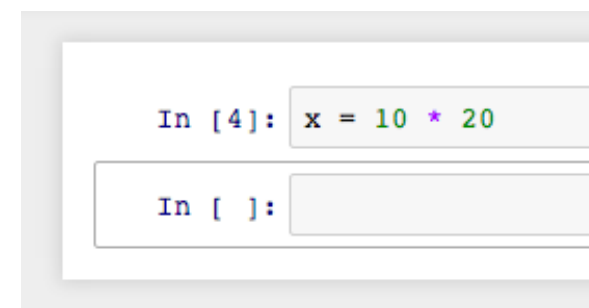
Code Cells

- In a code cell, you can enter one or more lines of Python code. Run the code by hitting Shift-Enter or by pressing the **Play** button in the toolbar.
- You can modify and re-run code cells multiple times in any order.
- When a code cell is executed, the code it contains is sent to the **kernel** associated with the notebook - i.e. the Python instance running in the background.
- The results returned from this computation are displayed as the cell's output. Note that some code will not have an output.

Change cell order



Start Stop



No visible output cell

- Restarting the kernel associated with a notebook clears all previous history (e.g. variable values).



Markdown Cells

- It can be helpful to provide explanatory text in notebooks.
- **Markdown** is a lightweight type of markup language with plain text formatting syntax which can be rendered as HTML.
- IPython supports a set of common Markdown commands. HTML tags and LaTeX formulae can also be included.
- When a Markdown cell is executed, the Markdown code is converted into the corresponding formatted rich text.

```
This is normal text.
```

This is normal text.

```
*This is italics*.
```

This is italics.

```
And **this is bold**.
```

And **this is bold.**

```
# Heading 1
## Heading 2
### Heading 3
```

Heading 1
Heading 2
Heading 3

```
Example <font color='red'>HTML use</font>
```

Example **HTML use**

```
Formula: $x=\frac{y}{z}$
```

Formula: $x = \frac{y}{z}$

The Zen of Python



```
import this
```

The Zen of Python, by Tim Peters

Beautiful is better than ugly.

Explicit is better than implicit.

Simple is better than complex.

Complex is better than complicated.

Flat is better than nested.

Sparse is better than dense.

Readability counts.

Special cases aren't special enough to break the rules.

Although practicality beats purity.

Errors should never pass silently.

Unless explicitly silenced.

In the face of ambiguity, refuse the temptation to guess.

There should be one-- and preferably only one --obvious way to do it.

Although that way may not be obvious at first unless you're Dutch.

Now is better than never.

Although never is often better than *right* now.

If the implementation is hard to explain, it's a bad idea.

If the implementation is easy to explain, it may be a good idea.

Namespaces are one honking great idea -- let's do more of those!

Online Resources

■ Python

- Official Python 3 documentation
- <https://docs.python.org/3/>

■ IPython Notebooks

- Official documentation
<http://ipython.readthedocs.org/en/stable/overview.html>

■ Markdown

- Github guide to Markdown
<https://help.github.com/articles/markdown-basics>
- Original Markdown syntax specification
<http://daringfireball.net/projects/markdown/syntax/>