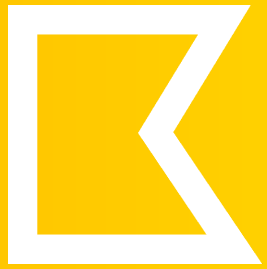


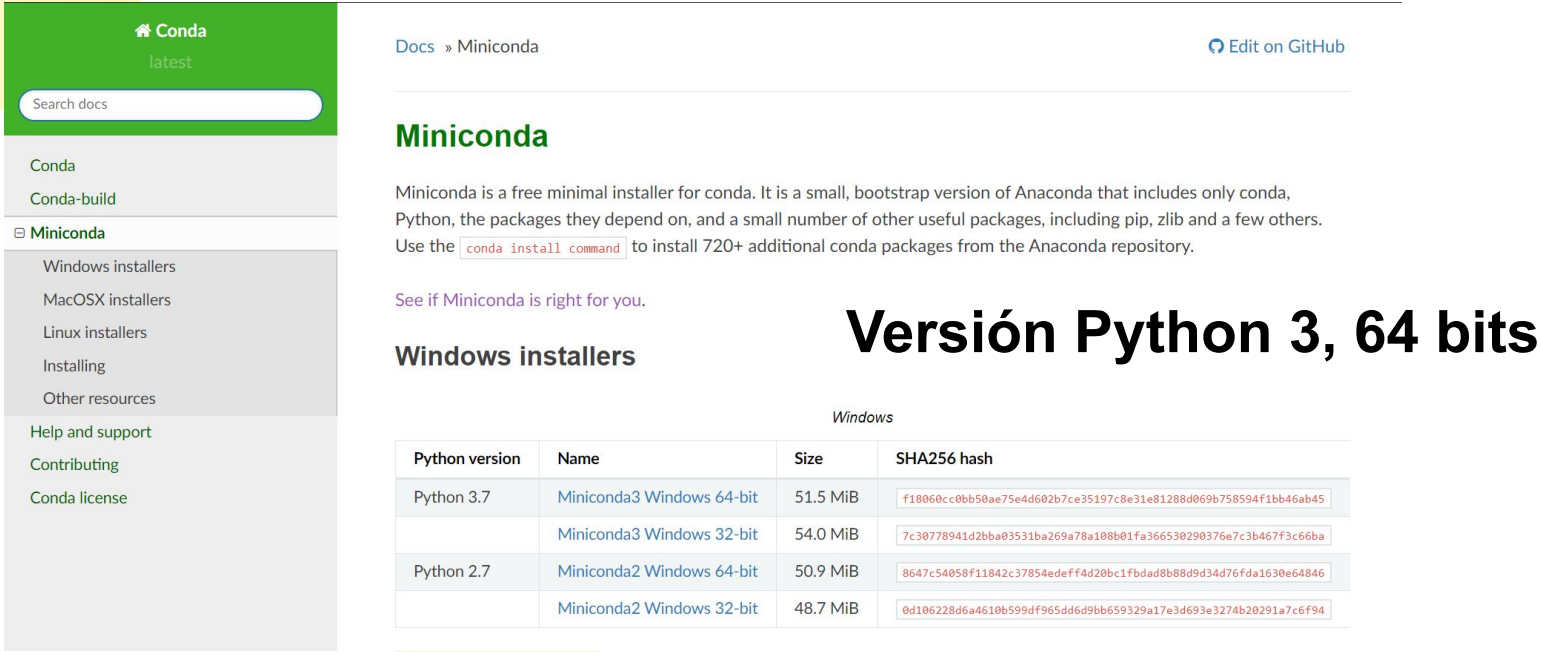
KSCHOOL

Python

Máster en Data Science

Franco D. Albareti

Descargar Miniconda (50 Mb)



Conda
latest

Search docs

- Conda
- Conda-build
- Miniconda
  - Windows installers
  - MacOSX installers
  - Linux installers
  - Installing
  - Other resources
- Help and support
- Contributing
- Conda license

Docs » Miniconda

Edit on GitHub

Miniconda

Miniconda is a free minimal installer for conda. It is a small, bootstrap version of Anaconda that includes only conda, Python, the packages they depend on, and a small number of other useful packages, including pip, zlib and a few others. Use the `conda install command` to install 720+ additional conda packages from the Anaconda repository.

See if Miniconda is right for you.

Versión Python 3, 64 bits

Windows installers

Windows

Python version	Name	Size	SHA256 hash
Python 3.7	Miniconda3 Windows 64-bit	51.5 MiB	f18060cc0bb50ae75e4d602b7ce35197c8e31e81288d069b758594f1bb46ab45
	Miniconda3 Windows 32-bit	54.0 MiB	7c30778941d2bba03531ba269a78a108b01fa366530290376e7c3b467f3c66ba
Python 2.7	Miniconda2 Windows 64-bit	50.9 MiB	8647c54058f11842c37854edeff4d20bc1fbdad8b88d9d34d76fda1630e64846
	Miniconda2 Windows 32-bit	48.7 MiB	0d106228d6a4610b599df965dd6d9bb659329a17e3d693e3274b20291a7c6f94

Syllabus Ia

- Introducción, instalación y primeros pasos
- IPython, Notebooks y Jupyter(-Lab)
  - Celdas
  - Markdown
  - Magic functions
- Sintaxis básica de Python
  - Variables escalares
  - Operadores
  - Memoria

Syllabus Ib

- Estructuras en Python:
  - Strings
  - Listas
  - Tuplas y Sets
  - Diccionarios
- Control Flow
  - Condicionales, Bucles, …
  - map, filter, reduce
  - Excepciones, Debugger

Syllabus IIa

- Funciones
  - Scope local y global
  - Lambda functions
- Virtual Environments
  - Gestor de paquetes y entornos virtuales (conda)
- Módulos y Scripts
- Ficheros
  - Context managers
  - Pickle, json

Syllabus IIb

- Clases
  - Instancia
  - Atributos y métodos
  - Herencia
  - Atributos/métodos protegidos y mágicos
  - Métodos de clase y métodos estáticos
- Miscellaneous
  - Decoradores
  - Editores: Spyder, VS Code
  - Módulos extras: Interact, SymPy

Python

- Creado en los 90 (Guido van Rossum)
- Lenguaje de alto nivel
- Programación orientada a objetos (OOP)
- Tipado dinámico y fuerte
- Interpretado
- Open Source -> Gran catálogo de librerías
- Propósito general -> Popular

TIOBE Index for November 2019

The TIOBE Programming Community index is an indicator of the popularity of programming languages. The index is updated once a month. The ratings are based on the number of skilled engineers world-wide, courses and third party vendors. Popular search engines such as Google, Bing, Yahoo!, Wikipedia, Amazon, YouTube and Baidu are used to calculate the ratings. It is important to note that the TIOBE index is not about the *best* programming language or the language in which *most lines of code* have been written.

www.tiobe.com

Nov 2019	Nov 2018	Change	Programming Language	Ratings	Change
1	1		Java	16.246%	-0.50%
2	2		C	16.037%	+1.64%
3	4	^	Python	9.842%	+2.16%
4	3	v	C++	5.605%	-2.68%
5	6	^	C#	4.316%	+0.36%
6	5	v	Visual Basic .NET	4.229%	-2.26%
7	7		JavaScript	1.929%	-0.73%
8	8		PHP	1.720%	-0.66%
9	9		SQL	1.690%	-0.15%
10	12	^	Swift	1.653%	+0.20%
11	16	^^	Ruby	1.261%	+0.17%
12	11	v	Objective-C	1.195%	-0.28%
13	13		Delphi/Object Pascal	1.142%	-0.28%
14	25	^^	Groovy	1.099%	+0.50%
15	15		Assembly language	1.022%	-0.09%

Nov 2019	Nov 2018	Change	Programming Language	Ratings	Change
1	1		Java	16.246%	-0.50%
2	2		C	16.037%	+1.64%
3	4	↑	Python	9.842%	+2.16%
4	3	↓	C++	5.605%	-2.68%
5	6	↑	C#	4.316%	+0.36%
6	5	↓	Visual Basic .NET	4.229%	-2.26%
7	7		JavaScript	1.929%	-0.73%
8	8		PHP	1.720%	-0.66%
9	9		SQL	1.690%	-0.15%
10	12	↑	Swift	1.653%	+0.20%
11	16	⇈	Ruby	1.261%	+0.17%
12	11	↓	Objective-C	1.195%	-0.28%
13	13		Delphi/Object Pascal	1.142%	-0.28%
14	25	⇈	Groovy	1.099%	+0.50%
15	15		Assembly language	1.022%	-0.09%

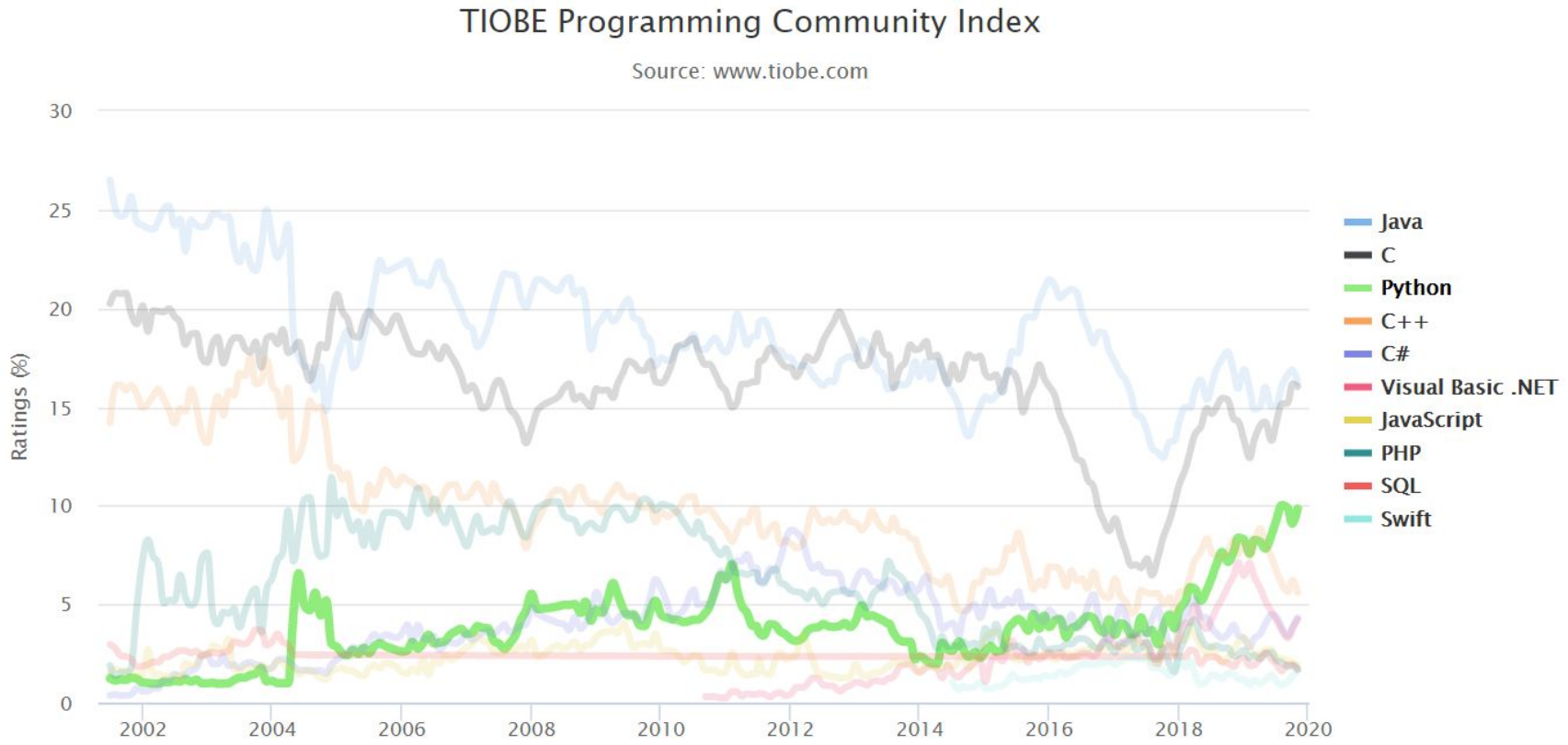
Python 3º

Nov 2019	Nov 2018	Change	Programming Language	Ratings	Change
1	1		Java	16.246%	-0.50%
2	2		C	16.037%	+1.64%
3	4	↑	Python	9.842%	+2.16%
4	3	↓	C++	5.605%	-2.68%
5	6	↑	C#	4.316%	+0.36%
6	5	↓	Visual Basic .NET	4.229%	-2.26%
7	7		JavaScript	1.929%	-0.73%
8	8		PHP	1.720%	-0.66%
9	9		SQL	1.690%	-0.15%
10	12	↑	Swift	1.653%	+0.20%
11	16	⇈	Ruby	1.261%	+0.17%
12	11	↓	Objective-C	1.195%	-0.28%
13	13		Delphi/Object Pascal	1.142%	-0.28%
14	25	⇈	Groovy	1.099%	+0.50%
15	15		Assembly language	1.022%	-0.09%
16	14	↓	R	0.980%	-0.43%
17	20	↑	Visual Basic	0.957%	+0.10%
18	23	⇈	D	0.927%	+0.25%
19	17	↓	MATLAB	0.890%	-0.14%
20	10	⇊	Go	0.853%	-0.64%

KSCHOOL

Python 3º
R 16º

- Cada lenguaje se adapta mejor a ciertas funciones
- Python es de propósito general, favorece su popularidad


TIOBE Programming Community Index
Source: www.tiobe.com
Ratings (%)
30
25
20
15
10
5
0
2002
2004
2006
2008
2010
2012
2014
2016
2018
2020
Java
C
Python
C++
C#
Visual Basic .NET
JavaScript
PHP
SQL
Swift

Instalación

- Anaconda (450 Mb) o Miniconda (50 Mb) Distributions
- Anaconda trae por defecto muchas librerías preinstaladas
- Miniconda trae lo básico
- Python/R
- Conda -> Gestor de paquetes y de entornos virtuales (lo veremos)
- Usad las opciones recomendadas para la instalación!

Primeros pasos

- Abrimos PowerShell/Terminal (Conda)
- Escribimos python (o python3/py/py3)
- Para salir -> exit() || ctrl + D || ctrl + Z + enter

Python Shell

```
(base) PS C:\Users\franc> python
Python 3.7.4 (default, Aug  9 2019, 18:34:13) [MSC v.1915 64 bit (AMD64)] :: Anaconda, Inc. on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> 2+2
4
>>> a=1
>>> a
1
>>> exit()
(base) PS C:\Users\franc>
```

Primeros pasos

- Abrimos PowerShell/Terminal (Conda)
- Escribimos python (o python3/py/py3)
- Para salir -> exit() || ctrl + D || ctrl + Z + enter

Python Shell

```
(base) PS C:\Users\franc> python
Python 3.7.4 (default, Aug  9 2019, 18:34:13) [MSC v.1915 64 bit (AMD64)] :: Anaconda, Inc. on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> 2+2
4
>>> a=1
>>> a
1
>>> exit()
(base) PS C:\Users\franc>
```

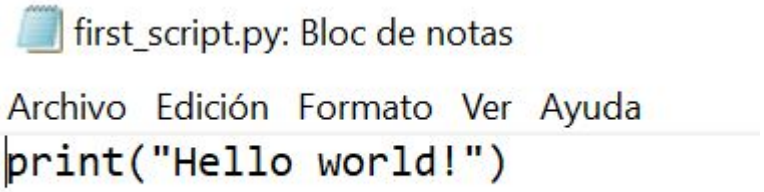
Primeros pasos

- Pasamos script por línea de comando
- No suele usarse

```
(base) PS C:\Users\franc> python -c "a=2+2; print('Hello world'); print(a)"
Hello world
4
(base) PS C:\Users\franc>
```

Primeros pasos

- “Source a file”, archivo tipo texto con extensión “.py”




```
(base) PS C:\Users\franc> python .\first_script.py
Hello world!
(base) PS C:\Users\franc>
```

Primeros pasos

- ¿Qué más podemos hacer con el comando python?

```
(base) PS C:\Users\franc> python -h
usage: C:\Users\franc\Miniconda3\python.exe [option] ... [-c cmd | -m mod | file | -] [arg] ...
Options and arguments (and corresponding environment variables):
-b     : issue warnings about str(bytes_instance), str(bytearray_instance)
         and comparing bytes/bytearray with str. (-bb: issue errors)
-B     : don't write .pyc files on import; also PYTHONDONTWRITEBYTECODE=x
-c cmd : program passed in as string (terminates option list)
-d     : debug output from parser; also PYTHONDEBUG=x
-E     : ignore PYTHON* environment variables (such as PYTHONPATH)
-h     : print this help message and exit (also --help)
-i     : inspect interactively after running script; forces a prompt even
         if stdin does not appear to be a terminal; also PYTHONINSPECT=x
-I     : isolate Python from the user's environment (implies -E and -s)
-m mod : run library module as a script (terminates option list)
-O     : remove assert and __debug__-dependent statements; add .opt-1 before
         .pyc extension; also PYTHONOPTIMIZE=x
```

iPython

- Shell mejorada/interactiva

```
(base) PS C:\Users\franc> conda install ipython
Collecting package metadata (current_repodata.json): done
Solving environment: done
```

```
(base) PS C:\Users\franc> ipython
Python 3.7.4 (default, Aug  9 2019, 18:34:13) [MSC v.1915 64 bit (AMD64)]
Type 'copyright', 'credits' or 'license' for more information
IPython 7.9.0 -- An enhanced Interactive Python. Type '?' for help.

In [1]:
```

iPython

- Shell mejorada/interactiva

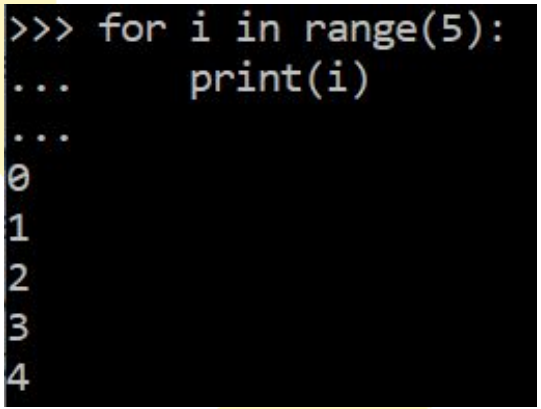
```
(base) PS C:\Users\franc> conda install ipython
Collecting package metadata (current_repodata.json): done
Solving environment: done
```

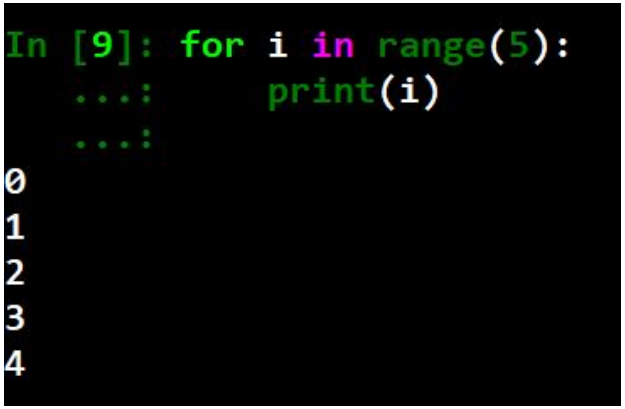
```
(base) PS C:\Users\franc> ipython
Python 3.7.4 (default, Aug  9 2019, 18:34:13) [MSC v.1915 64 bit (AMD64)]
Type 'copyright', 'credits' or 'license' for more information
IPython 7.9.0 -- An enhanced Interactive Python. Type '?' for help.

In [1]:
```

iPython

- Shell mejorada/interactiva
- Colores, autocompletado, resaltado de errores
- Historial extendido

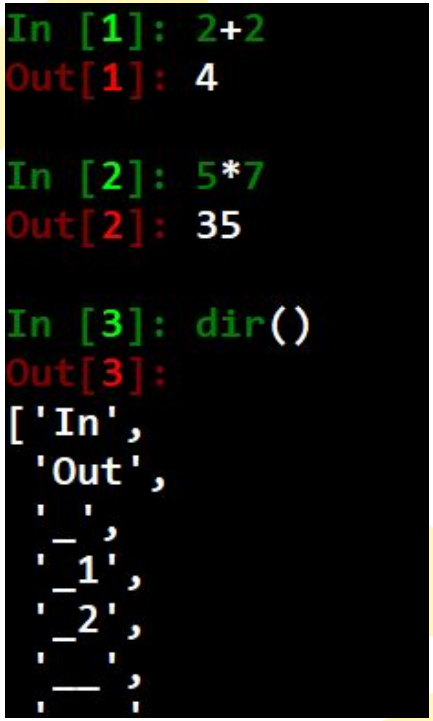
Python Shell

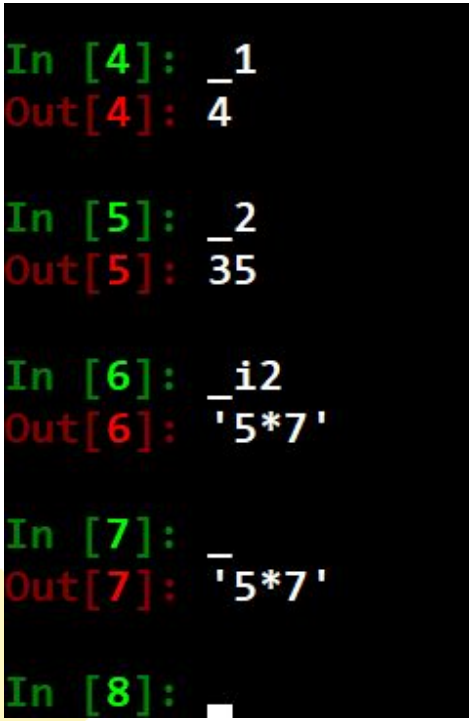


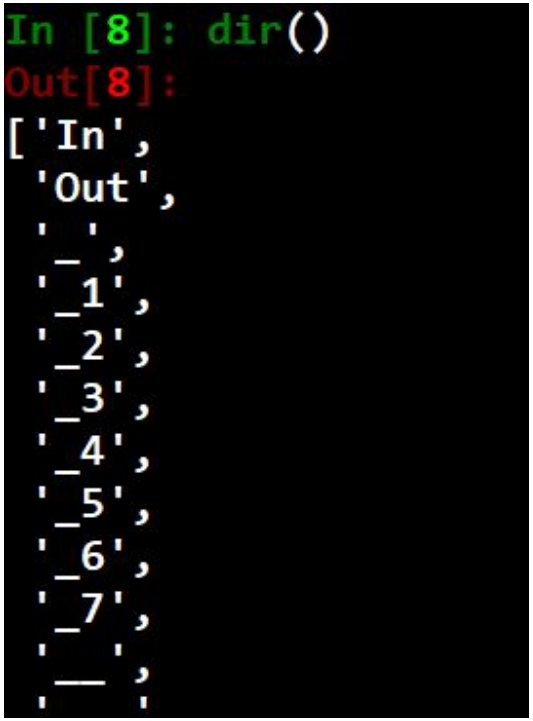


iPython Shell

iPython


In [1]: 2+2
Out[1]: 4
In [2]: 5*7
Out[2]: 35
In [3]: dir()
Out[3]:
['In',
'Out',
'_',
'_1',
'_2',
'__',


In [4]: _1
Out[4]: 4
In [5]: _2
Out[5]: 35
In [6]: _i2
Out[6]: '5*7'
In [7]: _
Out[7]: '5*7'
In [8]:


In [8]: dir()
Out[8]:
['In',
'Out',
'_',
'_1',
'_2',
'_3',
'_4',
'_5',
'_6',
'_7',
'__',

iPython

- Creado por Fernando Pérez (2001), mientras escribía su tesis en Física Teórica.
- Inspirado por Mathematica (1980).
- Celdas (Cells): Unidades básicas de código/texto
- Notebooks (“nb”): Documento de tipo json.
  - Ejecución de código interactivo -> Mucha flexibilidad
  - Ideal para Data Science
- Python Notebooks -> Archivos “.ipynb”/“.pynb”

Notebooks

- iPython pasó a llamarse Jupyter en 2014 (Galileo)
- iPython Notebooks = Jupyter Notebooks
- Jupyter no sólo abarca Python, es políglota.
- **Jupyter Lab** (2018), más potente.
- Notebooks:
  - Herramienta de trabajo estándar en Data Science
  - Cloud computing: Colab (Google), Amazon, Azure.
  - Investigación/Sector privado (Netflix, Amazon, ...)

Jupyter

```
(base) PS C:\Users\franc> conda install jupyter
Collecting package metadata (current_repodata.json): done
Solving environment: done
```

```
(base) PS C:\Users\franc> jupyter -h
usage: jupyter [-h] [--version] [--config-dir] [--data-dir] [--runtime-dir]
               [--paths] [--json]
               [subcommand]

Jupyter: Interactive Computing

positional arguments:
  subcommand     the subcommand to launch

optional arguments:
  -h, --help     show this help message and exit
  --version      show the jupyter command's version and exit
  --config-dir   show Jupyter config dir
  --data-dir     show Jupyter data dir
  --runtime-dir  show Jupyter runtime dir
  --paths        show all Jupyter paths. Add --json for machine-readable
                 format.
  --json         output paths as machine-readable json

Available subcommands: bundlerextension console kernel kernelspec migrate
nbconvert nbextension notebook qtconsole run script serverextension
troubleshoot trust
(base) PS C:\Users\franc>
```

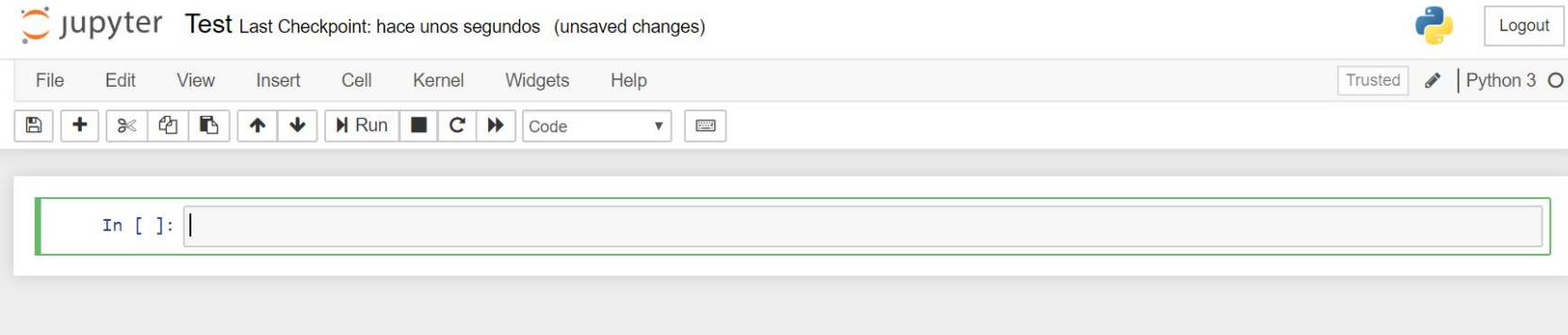
```
(base) PS C:\Users\franc> jupyter -h
usage: jupyter [-h] [--version] [--config-dir] [--data-dir] [--runtime-dir]
               [--paths] [--json]
               [subcommand]

Jupyter: Interactive Computing

positional arguments:
  subcommand     the subcommand to launch

optional arguments:
  -h, --help     show this help message and exit
  --version      show the jupyter command's version and exit
  --config-dir   show Jupyter config dir
  --data-dir     show Jupyter data dir
  --runtime-dir  show Jupyter runtime dir
  --paths        show all Jupyter paths. Add --json for machine-readable
                 format.
  --json         output paths as machine-readable json

Available subcommands: bundlerextension console kernel kernelspec migrate
nbconvert nbextension notebook qtconsole run script serverextension
troubleshoot trust
(base) PS C:\Users\franc>
```

```
(base) PS C:\Users\franc> jupyter notebook -h
The Jupyter HTML Notebook.

This launches a Tornado based HTML Notebook Server that serves up an
HTML5/Javascript Notebook client.

Subcommands
-----------

Subcommands are launched as `jupyter-notebook cmd [args]`. For information on
using subcommand 'cmd', do: `jupyter-notebook cmd -h`.

list
    List currently running notebook servers.
stop
    Stop currently running notebook server for a given port
password
    Set a password for the notebook server.

Options
-------

Arguments that take values are actually convenience aliases to full
Configurables, whose aliases are listed on the help line. For more information
on full configurables, see '--help-all'.
```

```
(base) PS C:\Users\franc> jupyter notebook
[I 14:19:31.871 NotebookApp] Serving notebooks from local directory: C:\Users\franc
[I 14:19:31.871 NotebookApp] The Jupyter Notebook is running at:
[I 14:19:31.871 NotebookApp] http://localhost:8888/?token=f009aa42ea9636e533cf8e501c26dd5b37958c6da482ae42
[I 14:19:31.872 NotebookApp]  or http://127.0.0.1:8888/?token=f009aa42ea9636e533cf8e501c26dd5b37958c6da482ae42
[I 14:19:31.872 NotebookApp] Use Control-C to stop this server and shut down all kernels (twice to skip confirmation).
[C 14:19:31.968 NotebookApp]

    To access the notebook, open this file in a browser:
        file:///C:/Users/franc/AppData/Roaming/jupyter/runtime/nbserver-1380-open.html
    Or copy and paste one of these URLs:
        http://localhost:8888/?token=f009aa42ea9636e533cf8e501c26dd5b37958c6da482ae42
     or http://127.0.0.1:8888/?token=f009aa42ea9636e533cf8e501c26dd5b37958c6da482ae42
```

- Attached to terminal
- Corre como un servidor (http://localhost:8888)
- Accedemos a través de un web browser (Chrome, Edge, Mozilla).


jupyter
Quit
Logout
Files
Running
Clusters
Select items to perform actions on them.
Upload
New
0
/
Name
Last Modified
File size
3D Objects
hace 23 días
blockchain
hace 2 meses
Contacts
hace 23 días
Desktop
hace 23 días


jupyter
Test
Last Checkpoint: hace unos segundos (unsaved changes)
Logout
File
Edit
View
Insert
Cell
Kernel
Widgets
Help
Trusted
Python 3
Run
Code
In [ ]:

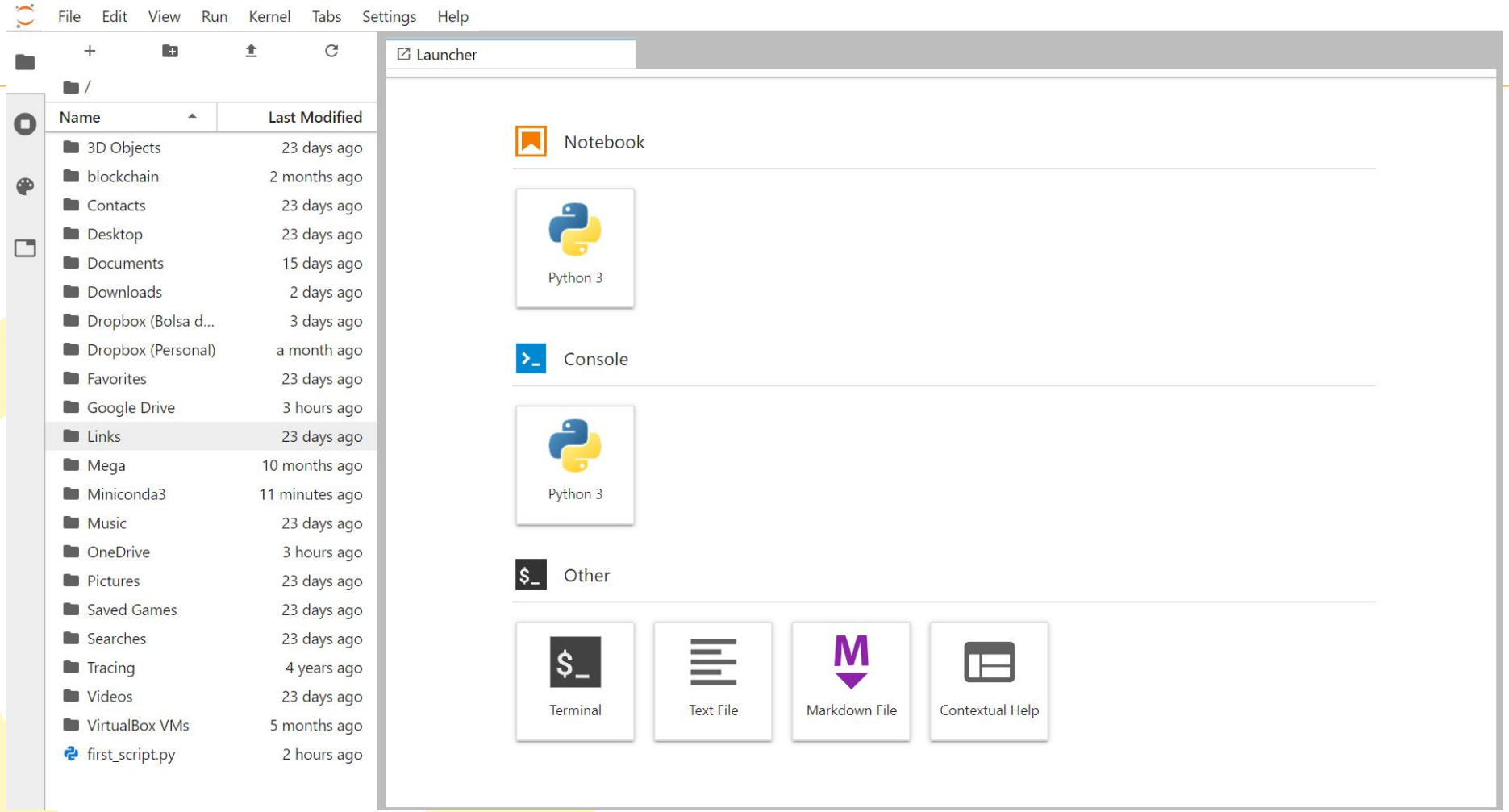
Jupyter Lab

```
(base) PS C:\Users\franc> conda install jupyterlab
Collecting package metadata (current_repodata.json): done
Solving environment: done
```

```
(base) PS C:\Users\franc> jupyter-lab -h
JupyterLab - An extensible computational environment for Jupyter.

This launches a Tornado based HTML Server that serves up an HTML5/Javascript
JupyterLab client.

JupyterLab has three different modes of running:

* Core mode (`--core-mode`): in this mode JupyterLab will run using the JavaScript
  assets contained in the installed `jupyterlab` Python package. In core mode, no
  extensions are enabled. This is the default in a stable JupyterLab release if you
  have no extensions installed.
* Dev mode (`--dev-mode`): uses the unpublished local JavaScript packages in the
  `dev_mode` folder.  In this case JupyterLab will show a red stripe at the top of
  the page.  It can only be used if JupyterLab is installed as `pip install -e .`.
* App mode: JupyterLab allows multiple JupyterLab "applications" to be
  created by the user with different combinations of extensions. The `--app-dir` can
  be used to set a directory for different applications. The default application
  path can be found using `jupyter lab path`.
```

```
(base) PS C:\Users\franc> jupyter-lab
[I 14:42:24.903 LabApp] JupyterLab extension loaded from C:\Users\franc\Miniconda3\lib\site-packages\jupyterlab
[I 14:42:24.903 LabApp] JupyterLab application directory is C:\Users\franc\Miniconda3\share\jupyter\lab
[I 14:42:24.909 LabApp] Serving notebooks from local directory: C:\Users\franc
[I 14:42:24.911 LabApp] The Jupyter Notebook is running at:
[I 14:42:24.912 LabApp] http://localhost:8888/?token=da7c2dc1a56a895c81c5933e2d963f749ca1556450a14a96
[I 14:42:24.913 LabApp]  or http://127.0.0.1:8888/?token=da7c2dc1a56a895c81c5933e2d963f749ca1556450a14a96
[I 14:42:24.913 LabApp] Use Control-C to stop this server and shut down all kernels (twice to skip confirmation).
[C 14:42:24.979 LabApp]

    To access the notebook, open this file in a browser:
        file:///C:/Users/franc/AppData/Roaming/jupyter/runtime/nbserver-12000-open.html
    Or copy and paste one of these URLs:
        http://localhost:8888/?token=da7c2dc1a56a895c81c5933e2d963f749ca1556450a14a96
     or http://127.0.0.1:8888/?token=da7c2dc1a56a895c81c5933e2d963f749ca1556450a14a96
```

- Attached to terminal
- Corre como un servidor (http://localhost:8888)
- Accedemos a través de un web browser (Chrome, Edge, Mozilla).


File Edit View Run Kernel Tabs Settings Help
/
Name
Last Modified
3D Objects 23 days ago
blockchain 2 months ago
Contacts 23 days ago
Desktop 23 days ago
Documents 15 days ago
Downloads 2 days ago
Dropbox (Bolsa d... 3 days ago
Dropbox (Personal) a month ago
Favorites 23 days ago
Google Drive 3 hours ago
Links 23 days ago
Mega 10 months ago
Miniconda3 11 minutes ago
Music 23 days ago
OneDrive 3 hours ago
Pictures 23 days ago
Saved Games 23 days ago
Searches 23 days ago
Tracing 4 years ago
Videos 23 days ago
VirtualBox VMs 5 months ago
first_script.py 2 hours ago
Launcher
Notebook
Python 3
Console
Python 3
Other
Terminal
Text File
Markdown File
Contextual Help

Jupyter

- Kernels -> “Unidad pensante” y donde se almacenan las variables
- A cada notebook se le asocia un Kernel independiente
- Los kernels pueden interrumpirse, reiniciarse, pararse, …
- Podemos tener varios kernels ejecutándose en paralelo
- Hay kernels para varios lenguajes de programación
- Ahora a probar!

Estructuras en Python

- Strings
- Listas
- Tuplas
- Sets
- Diccionarios

Control flow en Python

- If, elif, else
- For
- While
- Break, Continue
- Exceptions (try, else, finally)
- pdb -> Python debugger

Módulos

- Funcionalidad extra a parte de Python básico
  - Módulos, paquetes, librerías, …
- Open Source
- pip -> Gestor de paquetes
- virtualenvwrapper -> Gestor de entornos virtuales
- **conda** -> Gestor de paquetes y entornos virtuales

Conda

- Gestor de paquetes y entornos virtuales

```
(base) PS C:\Users\franc> conda -h
usage: conda-script.py [-h] [-V] command ...

conda is a tool for managing and deploying applications, environments and packages.

Options:

positional arguments:
  command
    clean        Remove unused packages and caches.
    config       Modify configuration values in .condarc. This is modeled
                 after the git config command. Writes to the user .condarc
                 file (C:\Users\franc\.condarc) by default.
    create       Create a new conda environment from a list of specified
```

Virtual Environments

- Entorno aislado para diferentes proyectos
- Cada proyecto puede tener sus propias dependencias
  - Módulos
  - Versiones de Python
- El código se ejecute de la forma deseada
- Entorno de trabajo limpio y específico para cada proyecto

```
(base) PS C:\Users\franc> conda create -n kschool
Collecting package metadata (current_repodata.json): done
Solving environment: done

## Package Plan ##

  environment location: C:\Users\franc\Miniconda3\envs\kschool



Proceed ([y]/n)? y

Preparing transaction: done
Verifying transaction: done
Executing transaction: done
#
# To activate this environment, use
#
#     $ conda activate kschool
#
# To deactivate an active environment, use
#
#     $ conda deactivate

(base) PS C:\Users\franc>
```

(1)

1) Creamos entorno virtual
2) Listamos entornos virtuales disponibles
3) Activamos uno de ellos
4) Listamos paquetes en dicho entorno

```
(base) PS C:\Users\franc> conda env list
# conda environments:
#
base                  *  C:\Users\franc\Miniconda3
kschool                  C:\Users\franc\Miniconda3\envs\kschool
nlp                      C:\Users\franc\Miniconda3\envs\nlp

(base) PS C:\Users\franc>
```

(2)

```
(base) PS C:\Users\franc> conda activate kschool
(kschool) PS C:\Users\franc>
```

(3)

```
(kschool) PS C:\Users\franc> conda list
# packages in environment at C:\Users\franc\Miniconda3\envs\kschool:
#
# Name                    Version                   Build  Channel
(kschool) PS C:\Users\franc>
```

(4)

Virtual Environments

- Podemos crear un virtual environment que ejecute una versión determinada de Python

```
(base) PS C:\Users\franc> conda create -n py2 python=2.7
Collecting package metadata (current_repodata.json): done
Solving environment: done
```

```
(base) PS C:\Users\franc> conda activate py2
(py2) PS C:\Users\franc> python
Python 2.7.17 (default, Oct 28 2019, 21:35:07) [MSC v.1500 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> _
```

- Comandos más usados
  - **conda list** -> Listamos módulos instalados
  - **conda install** -> Instalamos módulos
  - **conda update/upgrade** -> Actualizamos módulos
  - **conda remove/uninstall** -> Desinstalar módulos
  - **conda search** -> Busca información sobre paquetes
  - **conda info** -> Muestra información local
  - **conda create** -> Creamos entornos virtuales
  - **conda activate/deactivate**
  - **conda env** -> Interactuar con entornos virtuales

- **conda list** [*modulo*]-> Lista módulos instalados
  - **-h** -> Ayuda
  - **-e** -> Exporta fichero con módulos y versiones instaladas
  - **-r** -> Lista las modificaciones del entorno actual
  - **-n** -> Podemos especificar el nombre del entorno virtual

[*modulo*] -> numpy

numpy<1.15

numpy>=1.16

numpy=1.17.3

- **conda install** [*modulo*] -> Instala el módulo especificado
  - **-h** -> Ayuda
  - **--file** -> Instala los módulos especificados en el fichero
  - **--revision** -> Revierte el entorno a la versión especificada
  - **-c** -> Canal adicional a usar para descargar módulos
  - **-n** -> Podemos especificar el nombre del entorno virtual
  - **--no-deps** -> No modifica ni instala dependencias, no usar!
  - **--only-deps** -> Sólo instala dependencias

- **conda update/upgrade** [*modulo*]-> Actualiza módulos
  - **-h** -> Ayuda
  - **--file** -> Actualiza los módulos especificados en el fichero
  - **--all** -> Actualiza todos los módulos instalados
  - **-c** -> Canal adicional a usar para descargar módulos
  - **-n** -> Podemos especificar el nombre del entorno virtual
  - **--no-deps** -> No modifica ni instala dependencias, no usar!
  - **--only-deps** -> Sólo instala dependencias

- **conda remove/uninstall** [*modulo*] -> Desinstala módulos
  - **-h** -> Ayuda
  - **--all** -> Borra todos los paquetes del entorno
  - **-n** -> Podemos especificar el nombre del entorno virtual

- **conda search** [*modulo*] -> Busca información sobre módulos
  - **-h** -> Ayuda
  - **--envs** -> Busca todos los entornos locales
  - **-i** -> Muestra información detallada

- **conda info** [*modulo*] -> Muestra información sobre módulos
  - **-h** -> Ayuda
  - **-a** -> Muestra toda la información
  - **-e** -> Lista todos los entornos locales
  - **-s** -> Muestra variables de entorno
  - **--base** -> Muestra el directoria del entorno base

- **conda create** [*modulo*] -> Crea un entorno virtual con módulos
  - **-h** -> Ayuda
  - **-n** -> Nombre del entorno que queremos crear
  - **--file** -> Instala los módulos del fichero especificado
  - **--clone** -> Clona el entorno especificado

- **conda activate** [*env*] -> Activa el entorno virtual *env*
  - **--stack** -> Activa el entorno sobre el entorno ya activo

- **conda deactivate** -> Desactiva el entorno activo

La activación de entornos es a nivel de terminal.
Se pueden abrir varias terminales con diferentes entornos activos.

- **conda env** [*command*] -> Interaccionar con entornos virtuales
  - **-h** -> Ayuda
  - **create** -> Crear entorno a partir de un archivo
  - **export** -> Exportar entorno a un archivo
  - **list** -> Listar los entornos virtuales
  - **remove** -> Borrar entorno virtual
  - **update** -> Actualizar entorno
  - **config** -> Configurar entorno

- **conda clean** -> Elimina paquetes y caches que no se usan
  - **-h** -> Ayuda
  - **-a** -> Elimina todo
  - **-i** -> Elimina el índice
  - **-p** -> Elimina paquetes
  - **-t** -> Elimina tarballs

Ejercicios Virtual Environments

- Crea un entorno virtual llamado **“test”**
- Lista los entornos disponibles
- Activa el entorno **“test”**
- Lista los módulos instalados
- Instala la penúltima versión de **numpy**
- Actualiza numpy a la última versión
- Perdón, vuelve a la penúltima versión
- Lista los módulos instalados
- Exporta los módulos instalados del entorno a un fichero **“requirements.txt”**
- Crea otro entorno virtual llamado **“test_bis”**
- Instala los módulos del fichero “**requirements.txt**” creado anteriormente

Editores

- **Spyder**
  - Muy usado en su momento
  - Similar a RStudio (**R**)
  - conda install spyder
- **Visual Studio Code** (**VS Code**)
  - Moderno
  - Muchas herramientas para desarrolladores
  - Extensiones