



ES



Empleos

Mis Cursos

Blog

Agenda



Buscar en Platzi



Curso de Big Data y Ciencia de Datos

Artículo

Guía de Jupyter



Yeison Daza 23 PlatziRank ⌚ Oct. 17, 2016

¿Qué es Jupyter?

Es un entorno de desarrollo interactivo agnóstico del lenguaje para ciencias de la computación y ciencia de datos.

Jupyter notebook tiene tres componentes,

- Aplicación web, para correr código de forma interactiva desde un navegador web.
- Kernels, el proceso que corre el código en el lenguaje específico y regresa la salida del proceso a la aplicación web, jupyter soporta múltiples lenguajes como Python, Julia, R, etc.
- Documentos, código, anotaciones, imágenes, vídeo que compone el notebook, son almacenados en formato JSON.

Primeros pasos

Gestor de paquetes

Para poder instalar Jupyter y todas las dependencias que necesitamos en nuestro proyecto necesitamos tener instalado [PIP](#), el gestor de paquetes de Python.

Si tienes instalado Python 2.7 o 3.0> muy probablemente ya se encuentre instalado en tu sistema, para comprobarlo ejecuta el comando `pip --version` en la terminal, si arroja un error puedes seguir el [proceso de instalación](#).

Para instalar un paquete (en este caso Jupyter) lo único que debes hacer es ejecutar el comando

```
pip3 install jupyter
```

Si tienes instalado python 2.x

SIGUE CON:

[Continuando con nuestro ejemplo](#)

> **Nota:** Para realizar la instalación global requiere ejecutarse con sudo o iniciar la terminal con permisos de administrador.

Primer Notebook

Ya que sabemos como instalar las dependencias que necesitamos, vamos a crear nuestro primer proyecto.

Entorno virtual

Para poder tener organizados los proyectos que creemos y sus dependencias vamos a usar [entornos virtuales](#).

Primero instalamos virtualenv de manera global.

```
sudo pip install virtualenv
```

Nota: en Windows debes ejecutar el comando sin sudo, pero iniciando la terminal con modo administrador.

Ahora creamos nuestro entorno virtual

```
virtualenv primer_proyecto
```

Ahora cada vez que desees trabajar en el proyecto debes entrar a la carpeta y iniciar el entorno virtual.

```
source /bin/activate
```

En windows

```
\Scripts\activate
```

Nota: Recuerda estar dentro de la carpeta del proyecto para ejecutar este comando.

Una vez iniciado nuestro entorno virtual podemos instalar las dependencias que necesitamos localmente sin necesidad de instalarlas en el sistema operativo.



> Una vez iniciado nuestro entorno virtual debemos instalar Jupyter.

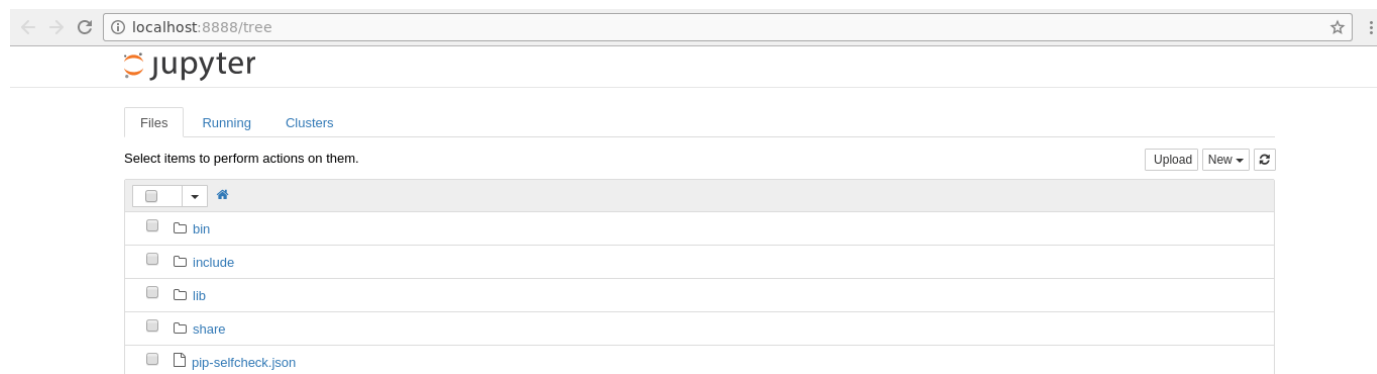
```
pip install jupyter
```

Una vez finalice la instalación iniciamos nuestro notebook

```
jupyter notebook
```

Se abrirá una ventana en nuestro navegador con nuestro **notebook**

Jupyter Dashboard



Jupyter trae una interfaz bastante sencilla, en ella encontramos tres pestañas

- *Files*: archivos del proyecto
- *Running*: procesos que se encuentran corriendo
- *Cluster*: administrador de los procesos en paralelo

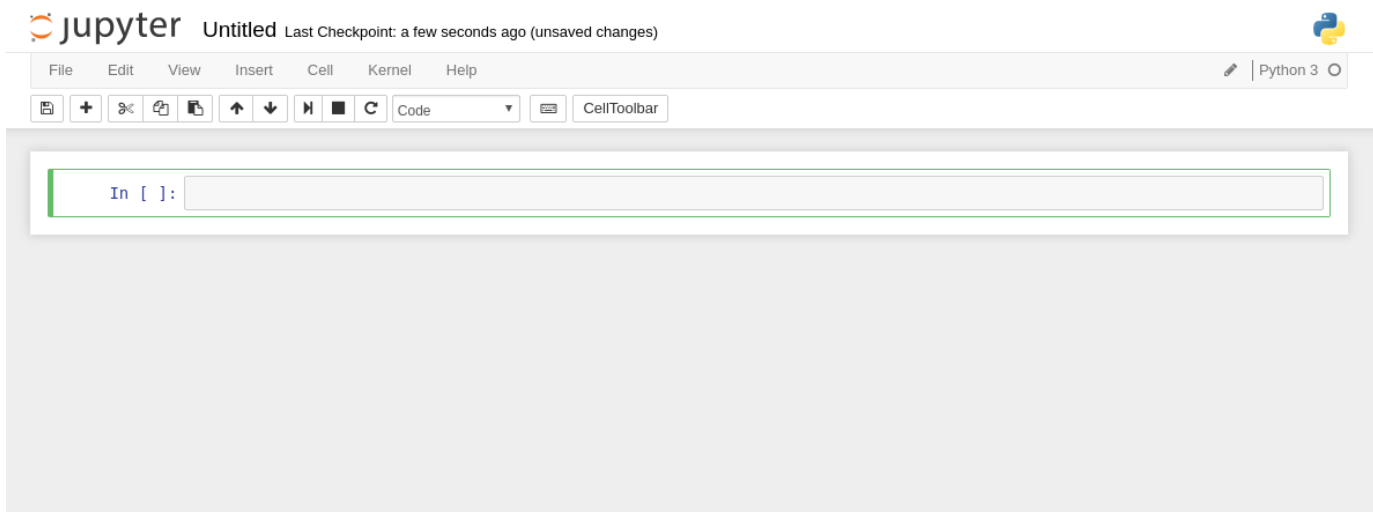
También encontramos dos botones:

- *Upload*: cargar archivos del computador
- *New*: crear nuevo archivo de texto, folder, terminal o notebook, en este ultimo nos lista los lenguajes con los cuales podemos crearlo.

Interfaz

Primero debemos crear nuestro primer notebook, para esto en el botón new, selecciona un notebook con python.





- **Header:** Consiste de un menú donde encuentras la opciones de edición y ejecución, este siempre estará fijo
- **Body:** Este es nuestro lugar de trabajo, este se compone de celdas las cuales pueden ser de tres tipos.

Markdown: Para crear textos con formato que sirvan como guía en el notebook

Código: Definimos el código que va a ejecutar.

Celdas sin formato: cuando necesitamos incluir texto sin formato.

Nota: en en celdas tipo código para ejecutar podemos usar ctrl + espacio, y si queremos ejecutar y crear una nueva celda shift + espacio.

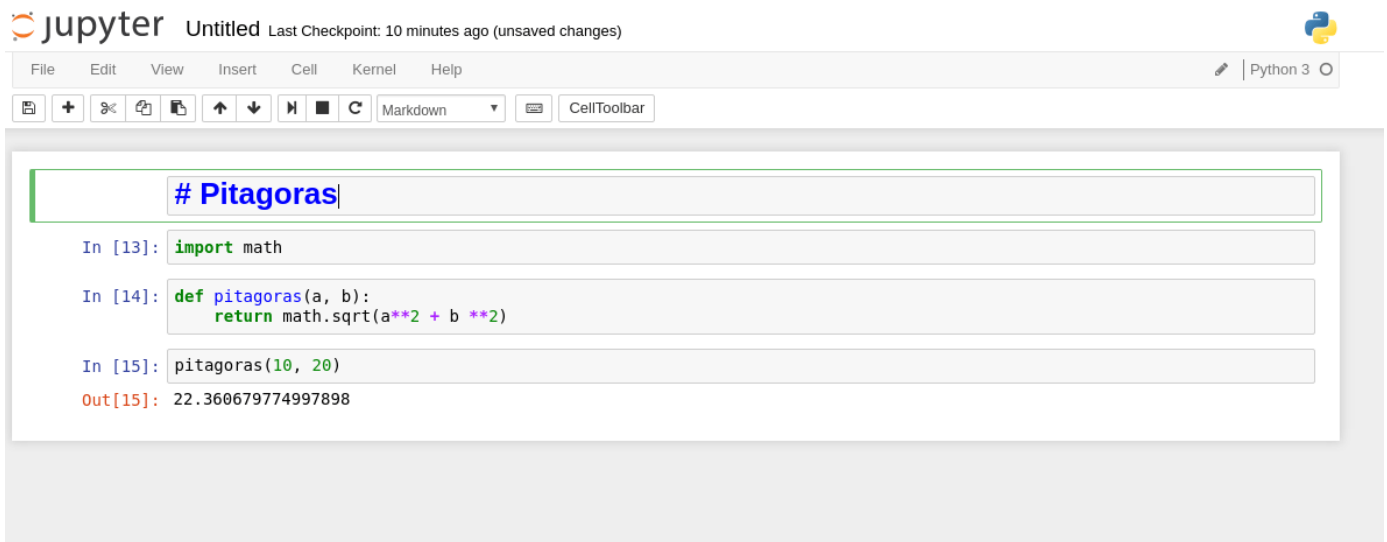
Usando Jupyter

Crea tu primer función en Jupyter y comparte tu resultado.

14

15

16



The screenshot shows a Jupyter Notebook titled "Untitled" with a last checkpoint 10 minutes ago. The interface includes a menu bar (File, Edit, View, Insert, Cell, Kernel, Help) and a toolbar with icons for file operations, execution, and cell management. The code cell contains the following Python code:

```
# Pitagoras

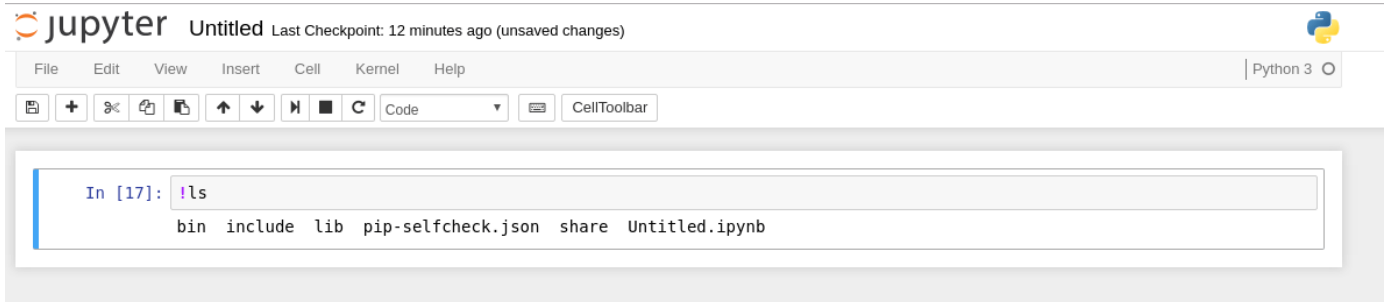
In [13]: import math

In [14]: def pitagoras(a, b):
          return math.sqrt(a**2 + b **2)

In [15]: pitagoras(10, 20)
Out[15]: 22.360679774997898
```

Tips

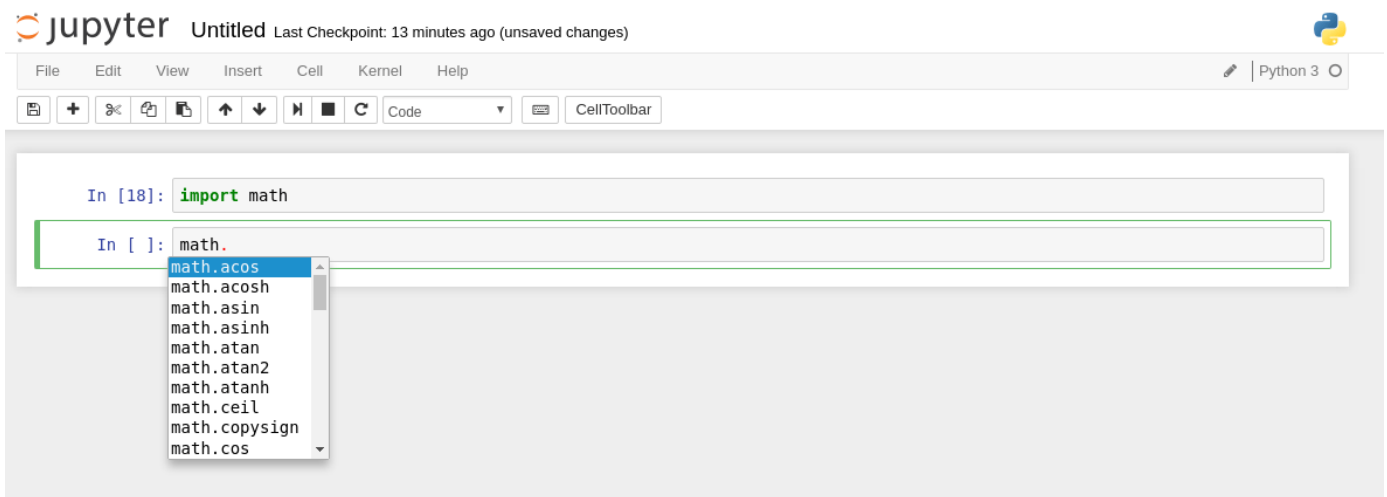
- Desde Jupyter podemos ejecutar comandos de consola usando el !



The screenshot shows a Jupyter Notebook titled "Untitled" with a last checkpoint 12 minutes ago. The code cell contains a shell command:

```
In [17]: !ls
bin include lib pip-selfcheck.json share Untitled.ipynb
```

- Usando la tecla tab podemos usar la función de auto completar



The screenshot shows a Jupyter Notebook titled "Untitled" with a last checkpoint 13 minutes ago. The code cell contains the following Python code:

```
In [18]: import math

In [ ]: math.
```

A dropdown menu is visible below the code cell, showing the following auto-completion suggestions:

- math.acos
- math.acosh
- math.asin
- math.asinh
- math.atan
- math.atan2
- math.atanh
- math.ceil
- math.copysign
- math.cos

