Jupyter Notebook Basico

August 22, 2016

1 Jupyter Notebooks: Básico

Los Notebooks de Jupyter (evolución de *ipython notebook*) son una alternativa para la creación de documentos interactivos con visualización y ejecución de código python (al igual que de otros lenguajes), redacción de texto en markdown y html, y soporte de ecuaciones a traves de *mathjax*. La unidad básica de ejecución de Jupyter son las celdas, las cuales pueden ser de tipo `markdown` para el texto o `code` para la ejecución de código.

1.1 Instalación

Para la instalación de notebook puede procederse de las siguientes formas:

1.1.1 Distribución Anaconda

La forma más simple de todas y que permite la instalación sin dificultades en linux, windows y mac es por medio de la distribución anaconda, una distribución de paquetes python de uso esencial en la computación científica. Los paquetes básicos para este fin se preinstalan (numpy, scipy, matplotlib, pandas, spyder y jupyter por ejemplo).

Descargue el instalador correspondiente (python 2 o 3, 32 o 64 bits, windows, linux o mac) en la página oficial de Continuum. A continuación siga las instrucciones básicas para cada sistema operativo (en linux invoque bash para la ejecución del instalador).

```
bash Anaconda3-4.1.1-Linux-x86_64.sh
```

1.1.2 Distribución Miniconda

El gestor *conda* es la utilidad de gestión de paquetes asociada a la distribución anaconda, la cual permite la instalación de paquetes python realizando la adecuada resolución de dependencias incluso cuando estas no son paquetes python. Si se requiere un mayor control sobre los paquetes instalados (computador con poca capacidad de almacenamiento, limitaciones en el consumo de red o baja memoria RAM), la opción adecuada es el uso de esta distribución, la cual solo viene por defecto con *conda* y *python*.

Descargue el instalador corrrespondiente (python 2 o 3, 32 o 64 bits, windows, linux o mac) de la página oficial de pydata, y siga las instrucciones básicas de instalación para cada sistema operativo. Para el caso de linux,

```
bash Miniconda3-latest-Linux-x86_64.sh
```

A continuación, usando `conda` se realizará la instalación de *jupyter*.

```
conda install jupyter
```

1.1.3 Gestor PIP

Si ya se posee una instalación previa de python, puede usar el gestor de paquetes PIP para su instalación. Este método sería el método avanzado de instalación, ya que requiere tambien una gestión extra para las dependencias que no sean python puro, como por ejemplo si se usan *bindings* a otro lenguaje o si se requiere compilar con base a CPython API.

Partiendo de una distribución linux, se requiere:

```
sudo python-dev python-pip
sudo pip install ipython[all] jupyter
```

Si desea instalar los equivalentes para python3, realice el cambio acorde:

```
sudo python3-dev python3-pip
sudo pip3 install ipython[all] jupyter
```

1.2 Markdown

Markdown es un lenguaje de marcado diseñado con el fin de facilitar la conversión texto a web, en un formato plano que indique de forma familiar su equivalente web.

Markdown se ha popularizado y existen multiples extensiones al lenguaje, dos de las más populares corresponden a *github flavored markdown (GFM)* y *Pandoc Markdown*. Debe prestar atención a las caracteristicas soportadas.

1.2.1 Titulos

La elaboración de titulos se logra de una manera simple, asociado un número igual de # al nivel de titulo deseado (en correspondencia a los titulos H1 a H6 de html).

```
# Jupyter Notebooks: Básico
## Instalación
### Distribución Anaconda
```

1.2.2 Tipo de letra y parrafos

Es posible cambiar el tipo de letra usando decoradores para el texto. Es posible generar una **negrilla** así `__negrilla__` o una *cursiva* como `_cursiva_`. El guión bajo _ puede ser sustituido con los mismos efectos por `*`. Si se desea resaltar un texto como código, de la forma que se realizo en este parrafo, debe usar los *backticks* para este fin, así `` `_cursiva_` ``.

Markdown carece en su forma original de otras formas de formato de texto, sin embargo si se posee conocimiento html, se puede usar este directamente en markdown para generar subrayado usando la etiqueta asociada `<u>subrayado</u>`.

Para separar un parrafo de otro, debe dejarse una linea vacia, y para forzar el cambio de linea se debe dejar doble espacio al final de la linea.

1.2.3 Listas

Las listas en markdown son posibles tanto ordenadas como no ordenadas. Para la generación de listas ordenadas basta inicial las lineas con un número arbitrario seguido inmediatamente de punto, y separar con un espacio el texto a listar. Las listas deben iniciar tras una linea vacia, y los elementos deben terminar con forzado de linea nueva.

En el caso de listas no ordenadas, puede iniciar indistintamente con los simbolos `+`, `-` y `*`, y cumplir las condiciones de linea vacia al inicio, espacio tras la viñeta y forzado de nueva linea al final.

Código:

```
1. Elemento 1
1. Elemento 2
35. Elemento 3
0. Elemento 4
    - Subelemento 1
    + Subelemento 2
    * Subelemento 3
        + Subsubelemento
```

Resultado:

1. Elemento 1
2. Elemento 2
3. Elemento 3
4. Elemento 4

- Subelemento 1
- Subelemento 2
- Subelemento 3
  - Subsubelemento

En el ejemplo se muestra la independencia del valor numerico y del simbolo de viñeta usado para la generación. Tambien es posible como se muestra, la generación de sublistas, las cuales de forma automatica cambian la viñeta.

1.2.4 Enlaces e imagenes

Markdown permite generar enlaces mediante indicación explicita de la url de manera inmediata o mediante el uso de un identificador para su reutilización. La primera forma usa la primera agrupación del texto en corchetes y la segunda en parentesis, así `[Markdown](https://daringfireball.net/projects/markdown/)` generando Markdown. Si se desea usar el identificador para reutilización, la segunda agrupación es en corchetes

a indica una etiqueta, la cual más adelante empezando linea se indica la segunda agrupación, seguido de dos puntos y espacio para la url.

De esta forma podemos indicar en markdown un enlace que reutilizará, como se muestra en la página oficial.

```
De esta forma podemos indicar en [markdown][markdown] un enlace que reutilizará, co

[markdown]: https://daringfireball.net/projects/markdown/syntax#link "Enlaces en ma
```

En ambos casos despues de la url, se puede indicar entre comillas el atributo de titulo dejando un espacio.

Las imagenes son básicamente enlaces a un archivo de imagen. De esta forma su estructura es igual salvo por la indicación del signo de cierre de admiración antes de la primera agrupación.

logo Jupyter

La imagen anterior se genero de la siguiente forma:

```
![logo Jupyter](jupyter.png "Logo Jupyter")
```

O si se desea la imagen almacenada remotamente (la visualización de imagenes depende del soporte del navegador en el notebook, pero depende del controlador adecuado del formato que

desee exportar, por ejemplo en el caso de pdf depende de los controladores asociados a los paquetes de LaTeX).

```
![logo Jupyter](http://jupyter.org/assets/main-logo.svg "Logo Jupyter")
```

En este caso la primera agrupación corresponde al texto alternativo que se muestra cuando la imagen no se puede visualizar.

Los enlaces a secciones del propio documento requieren del uso de la etiqueta html `href`, y dichas secciones deben estar en celdas separadas y al inicio de la misma.

1.2.5 Bloque de citación y código

Para la citación de texto, se usa el estilo de correo con lineas iniciando en >. Ejemplo:

> Esto es un bloque de citación.

y se produce así:

```
> Esto es un bloque de citación.
```

Las ejemplificaciones de código realizadas no solo se han hecho con *backticks* sino con bloques de código. Estos se logran dejando una linea vacia y luego tabulando cada linea que se quiera en este estilo.

```
    ### Bloque de citación y código

    Para la citación de texto, se usa el estilo de correo con lineas iniciando en `

    > Esto es un bloque de citación.
```

1.2.6 Caracterres especiales

Finalmente, si se desea producir de forma textual algun caracter o secuencia de caracteres que markdown transforma, debe generarse una cadena de escape con \ antes del caracter especial.

```
1\. Esto no es una lista.
\*Esto no es una cursiva\*
\![Esto no es una imagen](https://daringfireball.net/projects/markdown/syntax#backs
```

1\. Esto no es una lista.
Esto no es una cursiva
\![Esto no es una imagen

1.2.7 Extras

Los notebooks dan soporte a dos caracteristicas extras no contenidas en el markdown oficial, que son el soporte de ecuaciones y de tablas. Cualquier otra caracteristica que pueda encontrarse en una página web puede ser soportada incluyendo directamente código html, javascript y css.

Ecuaciones El soporte de las ecuaciones en Jupyter se logra renderizando con MathJax, lo que permite el uso de código LaTeX para las ecuaciones. Las ecuaciones en linea, como esta $\int \vec{F} \cdot d\vec{x}$, son generadas usando un solo signo pesos a cada lado de la expresión LaTeX y sin dejar espacio a los signos pesos, así `$\int \vec{F}\cdot d\vec{x}$`. Para ecuaciones aisladas y centradas, se usa doble signo pesos `$$\int \vec{F}\cdot d\vec{x}$$`,

$$\int \vec{F} \cdot d\vec{x}.$$

El uso del *backslash* con parentesis y corchetes no viene habilitado por defecto en mathjax, pero usando los archivos de configuración puede cambiarse este comportamiento. Aún asi, para uso en notebooks es recomendado el $ tanto en markdown como en el *magic* de LaTeX (si se desea exportar a pdf tambien será más comodo).

Columna 1	Columna 2	Columna 3
Izquierda	Derecha	Centro

Tablas

```
| Columna 1 | Columna 2 | Columna 3 |
|---|---:|:---:|
|Izquierda|Derecha|Centro|
```

1.3 Código

Jupyter Notebook soporta celdas de código con coloreado de sintaxis y ejecución de algunos lenguajes, mediante *magics* o soporte directo del kernel del lenguaje.

Para usar este tipo de celdas, usamos desde el menu `code` (por defecto es el tipo de celda al inicio) en lugar de `markdown`. Tambien se habilita con `Esc+y`.

1.3.1 Python

Al inicializar nuestro notebook con el kernel de python, directamente las celdas de código pueden ejecutar código python sin ninguna adición extra.

```
In [12]: from __future__ import print_function, division # Compatibilidad entre pyt
         def imprime_division(a, b):
             print(a/b)
```

El bloque anterior se ejecuta en python y mantiene una conexión a una instancia de interprete interactivo de python, de manera que podemos usar celdas de markdown y otras celdas de código, y luego hacer uso de las definiciones de estas celdas. Como a continuación:

```
In [13]: imprime_division(4, 5)
```

```
0.8
```

1.3.2 Magics

Existen una gran cantidad de *magics* disponibles. Particularmente nos interesarán dos casos, **bash** y **LaTeX**.
La ejecución de celdas LaTeX es solo para los ambientes matematicos (con soporte por MathJax) y no un soporte completo de una distribución LaTeX.

```
In [4]: %%latex
        $\left(\frac{1}{\Gamma}\right)^{2}$
```

$\left(\frac{1}{\Gamma}\right)^{2}$

Para la ejecución de bash se puede anteceder un signo ! antes del comando a usar (de una sola linea) o usar el *magic* correspondiente para múltiples lineas. Esto solo es posible en sistemas *UNIX*.

```
In [21]: !ls -al
```

```
total 204
drwxrwxr-x 3 cosmoscalibur cosmoscalibur   4096 ago  7 18:02 .
drwxrwxr-x 5 cosmoscalibur cosmoscalibur   4096 ago  2 21:18 ..
drwxr-xr-x 2 cosmoscalibur cosmoscalibur   4096 ago  7 11:45 .ipynb_checkpoints
-rw-rw-r-- 1 cosmoscalibur cosmoscalibur  14736 ago  7 18:02 Jupyter_Notebook_Basic
-rw-rw-r-- 1 cosmoscalibur cosmoscalibur  11375 ago  2 21:24 presentacion_herramier
-rw-rw-r-- 1 cosmoscalibur cosmoscalibur 139557 ago  2 21:24 presentacion_herramier
```

```
In [23]: %%bash
         touch archivoprueba.txt
         ls *.txt
         echo "Ya se verifico creación, ahora se eliminará."
         rm archivoprueba.txt
         ls archivoprueba.txt
```

```
archivoprueba.txt
Ya se verifico creación, ahora se eliminará.


ls: cannot access 'archivoprueba.txt': No such file or directory
```

En caso de usar los comandos bash con ! es posible asociar sus salidas a una variable python, y obtener una integración natural entre python y bash.

```
In [24]: listaarchivos = !ls
         print(listaarchivos)
```

```
['Jupyter_Notebook_Basico.ipynb', 'presentacion_herramientas.md', 'presentacion_her
```

Otros *magics* disponibles son:

```
In [25]: %lsmagic

Out[25]: Available line magics:
         %alias  %alias_magic  %autocall  %automagic  %autosave  %bookmark  %cat  %

         Available cell magics:
         %%!  %%HTML  %%SVG  %%bash  %%capture  %%debug  %%file  %%html  %%javascri

         Automagic is ON, % prefix IS NOT needed for line magics.
```

2 Referencias

1. Jupyter Notebook: Examples Index.
2. Jupyter Site.