

Primeros pasos con Python:

Manipulando imágenes

Rodolfo Ferro

@FERRORODOLFO

Abril 29, 2017

Universidad de Guanajuato
CIMAT A.C.



Tabla de contenido

1. Sobre Python
2. Instalación
3. Primeros pasos
4. Manipulando imágenes
5. Creando imágenes

Sobre Python

"Python es un lenguaje interpretado, sencillo, versátil y poderoso. La belleza del lenguaje radica en su sintaxis."



Un poco de historia (I)

- Python fue creado a inicios de los 90's por Guido van Rossum en el *Stichting Mathematisch Centrum* (CWI), en los Países Bajos, como un sucesor del lenguaje ABC.
- Guido sigue siendo el principal autor, aunque incluye muchas contribuciones por parte de otros.

Un poco de historia (II)

- En mayo del 2000, Guido y el equipo core de desarrollo de Python se mudan a *BeOpen.com* para formar el equipo BeOpen PythonLabs.
- En octubre del mismo año, los PythonLabs se mudan a Digital Creations (hoy Zope Corporation) y en 2001 se crea la *Python Software Foundation* (PSF), una organización sin fines de lucro creada específicamente para poseer todo lo relacionado con propiedad intelectual sobre Python.

Python en el mundo del SL

- Todos los releases de Python son Open Source.
- De acuerdo a la *Free Software Foundation*:

The screenshot shows a web browser displaying the Free Software Foundation (FSF) website at fsf.org. The page is titled "Python". The FSF logo is visible in the top left, and a "JOIN FSF NOW" button is in the top right. The main navigation menu includes links for about, campaigns, licensing, membership, resources, community, donate, and shop. Below the menu, a sub-navigation bar includes "Meet the free software gang >" and a search bar with a "Search" button. The main content area features a red background with a large black cartoon snake logo. Text on the page describes Python as a multi-purpose programming language and its popularity among developers. A sidebar on the right contains a "Sign up" form for newsletter subscribers, with fields for email address and a "Subscribe me" button. At the bottom, there's a "News" section with a link to a SecureDrop and Alexandre Oliva award article from March 25, 2017.

Meet the free software gang >

Python

by sdubois Contributions — Published on Jun 24, 2010 12:53 PM

Python is a multi-purpose programming language that packs a punch. A popular choice for developers, Python makes writing complex programs less difficult through its clear syntax. Python can be used for writing near anything, from a small desktop application to a powerful web application.

While there are many programming languages for one to choose from, Python is the favorite of many programmers in the free software community.

- Find more programming languages in the Free Software Directory



SecureDrop and Alexandre Oliva are 2016 Free Software Awards winners

Mar 25, 2017

Recordando un poco...

Un **Software Libre** es aquel que respeta las cuatro libertades que la *FSF* establece:

- La libertad de usar el programa con cualquier propósito.
- La libertad de estudiar cómo funciona el programa y modificarlo, adaptándolo a tus necesidades.
- La libertad de distribuir copias del programa, con lo cual puedes ayudar a tu próximo.
- La libertad de mejorar el programa y hacer públicas esas mejoras a los demás, de modo que toda la comunidad se beneficie.

Nota: El Software Libre no es necesariamente gratuito.

Instalación

Instalando Python

- Ir al sitio oficial: <https://www.python.org/downloads/>
- Descargar la versión 3.6.x.

The screenshot shows the Python Software Foundation website's main page. At the top, there is a navigation bar with links for Python, PSF, Docs, PyPI, Jobs, and Community. Below the navigation bar is the Python logo and a search bar with a magnifying glass icon and a 'GO' button. To the right of the search bar are 'Socialize' and 'Sign In' buttons. The main content area has a dark blue background with a large yellow and white striped parachute graphic on the right side. A prominent heading says 'Download the latest version for Mac OS X'. Below it are two buttons: 'Download Python 3.6.1' and 'Download Python 2.7.13'. A link to 'Wondering which version to use? Here's more about the difference between Python 2 and 3.' is provided. Another link to 'Looking for Python with a different OS? Python for Windows, Linux/UNIX, Mac OS X, Other' is also present. A link to 'Want to help test development versions of Python? Pre-releases' is at the bottom. At the very bottom of the main content area, there is a section titled 'Looking for a specific release?' followed by a table showing Python releases by version number. The table has columns for 'Release version', 'Release date', and 'Click for more'. It lists two rows: one for Python 3.6.1 (released 2017-03-21) and one for Python 3.4.6 (released 2017-01-17). Each row includes a download link and a 'Release Notes' link.

Release version	Release date	Click for more
Python 3.6.1	2017-03-21	Download Release Notes
Python 3.4.6	2017-01-17	Download Release Notes

Instalando Python

En Windows:

- Preferentemente instalarlo en el disco local c://.
- Dar check a la opción "Add Python to PATH".

En Linux/Unix:

- No hace falta algo adicional, todo es hermoso.
- Es más, por default ya viene instalada la versión 2.7 de Python.

Instalando paquetes

Al instalar Python, se instala **pip** (*Python Package Index*), que es un gestor de paquetes de Python. Usando **pip** instalaremos:

- **NumPy**, que incluye herramientas de métodos numéricos.
- **Matplotlib**, que incluye herramientas de visualización.

Nota: Python se instala con su librería estándar. Lo recomendable es instalar una distribución (como *Anaconda*) para una mejor gestión de paquetes, además de instalarse por default los más usados.

Paquetes de manipulación de imágenes

Existen diversos paquetes para manipular imágenes:

- **PIL** (*Python Imaging Library*)
- **OpenCV** (*OpenSource Computer Vision*)
- **scikit-image** (*Scientific Kit - Image*)
- Etc.

Cada uno trata de manera distinta las imágenes y se enfocan en cosas diferentes. Como es un taller de primeros pasos, no usaremos alguno de estos paquetes, sino que trabajaremos desde cero, sólo con **numpy** y **matplotlib**.

Instalando numpy y matplotlib

Abriremos una consola y ejecutaremos los siguientes comandos.

Para instalar numpy:

```
$ sudo pip install numpy
```

Para instalar matplotlib:

```
$ sudo pip install matplotlib
```

Nota: En Windows se omite el comando `sudo`.

Primeros pasos

Lo básico (I)

La belleza de Python radica en la sintaxis, los bloques de código deben tener la misma indentación, pues las llaves {} se utilizan para una estructura de datos (diccionarios).

Un ejemplo:

```
while True:  
    # El bloque de código dentro del ciclo debe  
    # estar en el mismo nivel de indentación  
  
    print("Infinite loop.")
```

Lo básico (II)

- No hace falta terminar cada línea con punto y coma.
- Los scripts de Python pueden ejecutarse desde consola, usando un IDE o con Jupyter notebooks.
- Todos los scripts de Python deben ser de extensión .py (excepto por los Jupyter notebooks).
- Para fines prácticos, nosotros correremos scripts desde consola y escribiremos código desde un bloc de notas, por lo que es recomendable instalar uno tipo Atom o Sublime Text.

Variables y operaciones (I)

- Las variables no se declaran, sólo se definen y Python sabe el tipo de variable que se está utilizando.
- Python cuenta con 5 tipos de datos estándar: números, cadenas de texto, listas, tuplas y diccionarios.
- Puedes realizar asignación múltiple.
- Las operaciones básicas con números son: +, -, *, /, //, **.
- En las cadenas de texto también puede usarse + (concatenación) y * (repeticIÓN).

Variables y operaciones (II)

Los tipos de variables que existen en Python son:

- **Enteros**, como 3.
- **Longs**, como 51924361L.
- **Flotantes**, como 3.14e4.
- **Complejos**, como 4.53-7j.
- **Strings**, como "Hola mundo".
- **Listas**, como [1., 3, 5.3, "perrito"].
- **Tuplas**, como (1., 3, 5.3, "perrito").
- **Diccionarios**, como {"Nombre":"Rodolfo", "Apellido":"Ferro"}

Condicionales

La sintaxis correspondiente es:

```
if/elif/else
    if condition:
        # Block content
    elif condition:
        # Block content
    else:
        # Block content
```

Ciclos

La sintaxis correspondiente es:

```
for
    for idx in iterable:
        # Block content
```

```
while
    while condition:
        # Block content
```

Funciones

La sintaxis correspondiente es:

Funciones

```
def function_name(parameters):  
    # Block content  
    return values
```

No hay limitación en los parámetros y valores de retorno, en el sentido de puede pasársele cualquier variable o función como argumento y cualesquiera variables pueden ser retornadas en una función.

Para concluir la sección...

Para poder concluir la sección y comenzar a tirar nuestras primeras líneas de código en Python, tengamos claro lo siguiente:

- ¿Qué es slicing?
- ¿Por qué conviene utilizar arreglos de NumPy en vez de listas de Python?
- Computacionalmente, ¿qué es una imagen?
- ¿Cuántas dimensiones tiene una imagen?
- ¿Hay alguna otra pregunta?



Repositorio del taller:

<https://github.com/RodolfoFerro/FLISoL17>

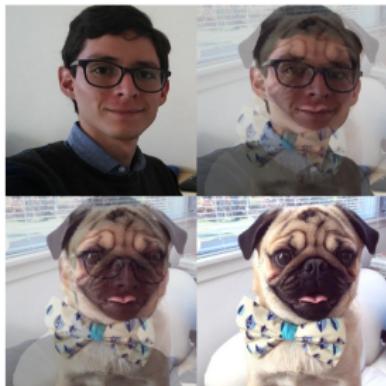
En este punto, descargar solamente la carpeta de imágenes.

Manipulando imágenes

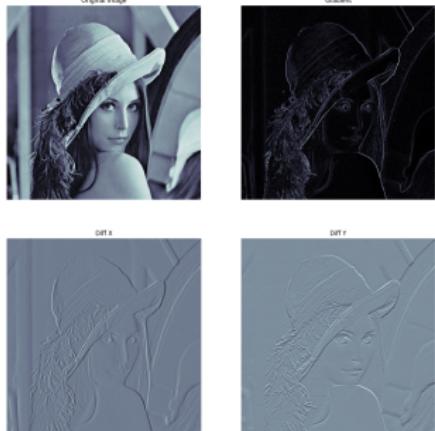
Objetivos

Queremos ser capaces de recrear las siguientes imágenes:

When sabes programar y te programas una transición de opacidad $(1 - \alpha)A + \alpha B$ para dos imágenes A y B , con $\alpha \in [0, 1]$ y te conviertes en un meme de pug:



This plot was created with the magic of Python.



Importando numpy y matplotlib

Se pueden importar librerías simplemente haciendo `import librería`, sin embargo, se pueden importar módulos específicos (ej. `from matplotlib import pyplot`) o importar módulos/librerías con un alias.

Por simplicidad nosotros importaremos `numpy` y `matplotlib` como sigue:

Importamos usando un alias:

```
import numpy as np  
import matplotlib.pyplot as plt
```

Mostrando una imagen

Matplotlib tiene una función llamada `imread`, cuyo argumento es una dirección y sirve para leer una imagen y guardarla como un arreglo de NumPy.

Podemos imprimir la matriz usando la función `print` y mostrarla en una ventana haciendo lo que sigue:

Mostrando una imagen:

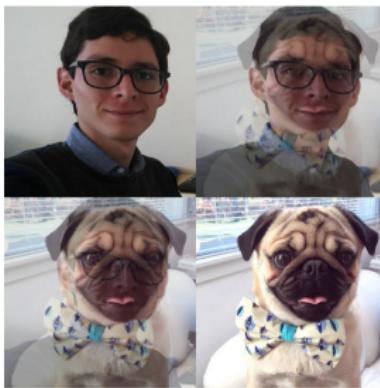
```
img = plt.imread("../imgs/Lenna.png")
plt.imshow(img)
plt.show()
```

Reto para llevar

Reto 1:

Dar formato a la transición de opacidad para que quede como sigue:

When sabes programar y te programas una transición de opacidad $(1 - \alpha)A + \alpha B$ para dos imágenes A y B , con $\alpha \in [0, 1]$ y te conviertes en un meme de pug:



This plot was created with the magic of Python.

Reto para llevar

Reto 2:

Nuestras funciones de derivadas de imagen funcionan sólo con imágenes que son cuadradas, ¿por qué?

Corregir esto para que sirva con la imagen leaf.png:



Para concluir la sección...

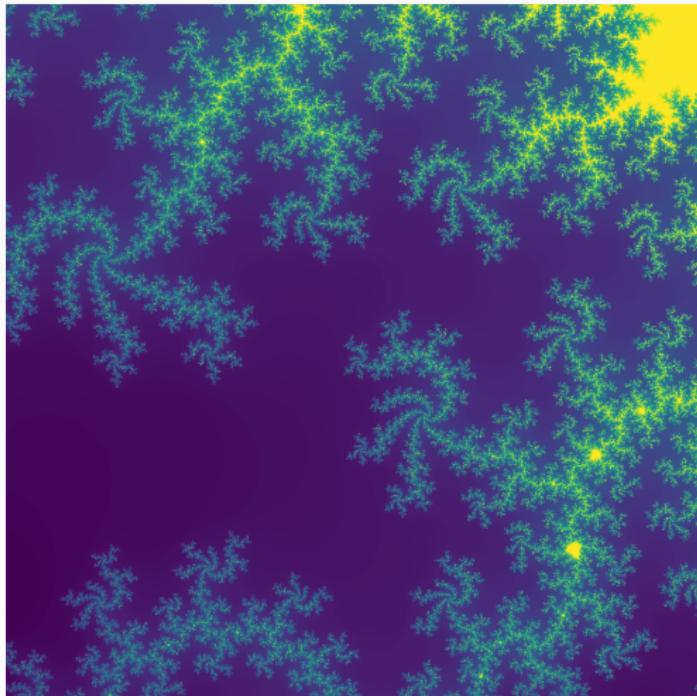
Para poder concluir la sección y continuar con la siguiente, respondamos:

- ¿Resulta sencillo manipular imágenes (arreglos de NumPy, en general)?
- ¿Por qué las derivadas de las imágenes ayudan a encontrar bordes?
- ¿Hay alguna otra pregunta?

Creando imágenes

Objetivos

Queremos ser capaces de recrear la siguiente imagen:



¿Preguntas?

Referencias

- [1] Python Software Foundation.
History of the software.
History and Licence, disponible en
<https://docs.python.org/3/license.html>, 2017.
- [2] Hipertextual.
¿Qué es Software Libre?
Diferencias entre Software Libre y Open Source, 2014.
- [3] Hipertextual.
¿Qué es Open Source?
Diferencias entre Software Libre y Open Source, 2014.

Referencias

[4] The Hitchhiker's Guide to Python.

Python Imaging Library.

Image Manipulation, 2016.

[5] The Hitchhiker's Guide to Python.

OpenSource Computer Vision.

Image Manipulation, 2016.

[6] Python Variable Types.

Tutorials Point.

Python Basic Tutorial.

Referencias

- [7] The Hitchhiker's Guide to Python.
Python Imaging Library.
Image Manipulation, 2016.
- [8] The Hitchhiker's Guide to Python.
OpenSource Computer Vision.
Image Manipulation, 2016.
- [9] Python Variable Types.
Tutorials Point.
Python Basic Tutorial.