jueves, 18 de diciembre de 2014

Fundamentos para procesar imágenes con Pillow (y III)

Después de ver, en la primera parte de esta entrega, algunas funciones básicas de edición de imágenes del módulo **Image** de Pillow y, en la segunda parte, otras para convertir las imágenes y aplicar diferentes filtros, pasamos ahora a mostrar funciones del módulo **ImageOps** que se utilizan para añadir bordes a las imágenes y para aplicar efectos que crean imágenes inversas, espejadas, ecualizadas; o que permiten modificar sus colores, el contraste, etc.

Para finalizar se muestran algunos ejemplos que usan funciones del módulo **ImageDraw** que se utilizan para realizar dibujos vectoriales y escribir textos sobre una imagen.

Añadir bordes a una imagen

Para agregar un borde de color negro alrededor de la imagen de 5 píxeles de grosor:

```
from PIL import Image, ImageOps
imagen = Image.open("pantera.jpg")
imagen.size   # 800x600
imagen.mode   # RGB
imagenconborde = ImageOps.expand(imagen, border=5, fill=(0,0,255))
imagenconborde.show()
imagenconborde.save("panteraborde.jpg")
```

Lógicamente, si la imagen original tenía un tamaño de 800x600 ahora la obtenida será de 810x610, incrementándose en 5 píxeles cada margen de la imagen.

Buscar

Buscar

Python para impacientes

Python
IPython
EasyGUI
Tkinter
JupyterLab
Numpy

Anexos

Guía urgente de MySQL
Guía rápida de SQLite3

Entradas + populares

Dar color a las salidas en la consola
En Python para dar color a las salidas en la consola (o en la terminal de texto) existen varias posibilidades. Hay un método basado ...

Instalación de Python, paso a paso
Instalación de Python 3.6 A finales de 2016 se produjo el lanzamiento de Python 3.6 . El propósito de esta entrada es mostrar, pas...

Añadir, consultar, modificar y suprimir elementos en Numpy
Acceder a los elementos de un array. [], [,], ... Acceder a un elemento de un array. Para acceder a un elemento se utiliz...

Variables de control en Tkinter
Variables de control Las variables de control son objetos especiales que se asocian a los widgets para almacenar sus valore...

Cálculo con arrays Numpy
Numpy ofrece todo lo necesario para obtener un buen rendimiento cuando se trata de hacer cálculos con arrays. Por como está concebid...

Tkinter: interfaces gráficas en Python
Introducción Con Python hay muchas posibilidades para programar una interfaz gráfica de usuario (GUI) pero Tkinter es fácil d...

Operaciones con fechas y horas. Calendarios
Los módulos datetime y calendar amplían las posibilidades del módulo time que provee funciones para manipular expresiones de ti...

Convertir, copiar, ordenar, unir y dividir arrays Numpy
Esta entrada trata sobre algunos métodos que se utilizan en Numpy para convertir listas en arrays y viceversa; para copiar arrays d...

Tkinter: Tipos de ventanas
Ventanas de aplicación y de diálogos En la entrada anterior tratamos los distintos gestores de geometría que se utilizan para di...

Threading: programación con hilos (I)
En programación, la técnica que permite que una aplicación ejecute

```
imagenconborde.size    # 810x610
imagenconborde = ImageOps.expand(imagen, border=5, fill="red")
```

La forma de asignar colores depende del tipo de imagen:

Tipo de imagen	Formato color
1, L, I	número entero
RGB	tupla con 3 números enteros (Rojo, Verdez y Azul) o nombres de color (en inglés)
F	número entero o número de coma flotante
P	número entero, tupla con 3 números enteros o nombres de color

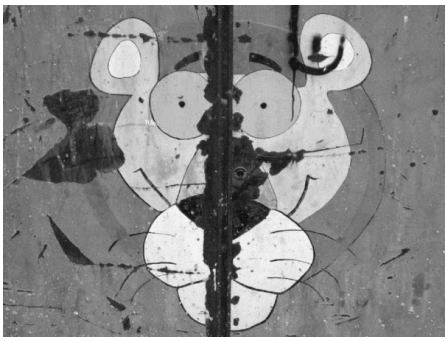
Recortar los bordes de una imagen

La función ***crop()*** se utiliza para recortar las imágenes por todos sus márgenes un mismo número de píxeles:

```
recortar = ImageOps.crop(imagen, border=50)
recortar.show()
```

Convertir la imagen a escala de grises

```
grises = ImageOps.grayscale(imagen)
grises.show()
```



Obtener la imagen inversa

```
inversa = ImageOps.invert(imagen)
inversa.show()
```

simultáneamente varias operaciones en el mismo espacio de proceso se...

Archivo

diciembre 2014 (3)

python.org

pypi.org

Sitios

- ActivePython
- Anaconda
- Bpython
- Django
- Flask
- Ipython
- IronPython
- Matplotlib
- MicroPython
- Numpy
- Pandas
- Pillow
- PortablePython
- PyBrain
- PyCharm
- PyDev
- PyGame
- Pypi
- PyPy
- Pyramid
- Python.org
- PyTorch
- SciPy.org
- Spyder
- Tensorflow
- TurboGears

Espejar una imagen (de izquierda a derecha)

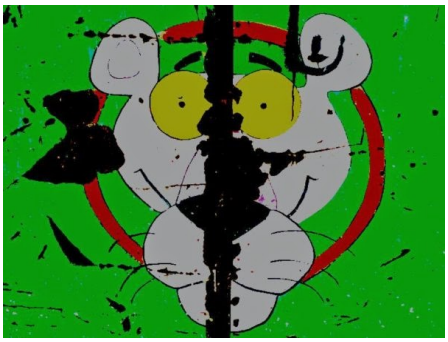
```
espejo = ImageOps.mirror(imagen)
espejo.show()
```

Voltear una imagen verticalmente (de arriba a abajo)

```
flipeada = ImageOps.flip(imagen)
flipeada.show()
```

Reducir el número de bits para cada canal de color

```
posterizada = ImageOps.posterize(imagen, 1)
posterizada.show()
```



Invertir los valores de los píxeles por encima de un umbral

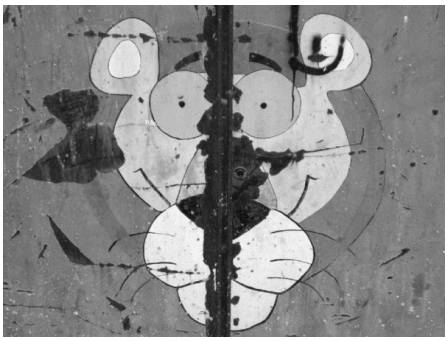
```
solarizada = ImageOps.solarize(imagen, threshold=64)
solarizada.show()
```

Maximizar contraste de una imagen (Autocontraste)

```
imagen = Image.open("nevando.jpg")
autocontraste = ImageOps.autocontrast(imagen, cutoff=0)
autocontraste.show()
```

La función calcula un histograma de la imagen de entrada y borra un porcentaje de los píxeles más claros y oscuros. Los píxeles más oscuros se convierten en negros (0) y los más claros en blancos (255).

Dar color a una imagen en blanco y negro



A los píxeles negros de la imagen se les asigna el primer color indicado en la función y a los píxeles blancos al segundo color.

```
imagen = Image.open("grises.jpg")
imagen.show()
coloriza = ImageOps.colorize(imagen, "black", "cyan")
coloriza.show()
coloriza.save("coloriza.jpg")
```

También, en este caso, es posible asignar un color RGB dando los valores en una tupla

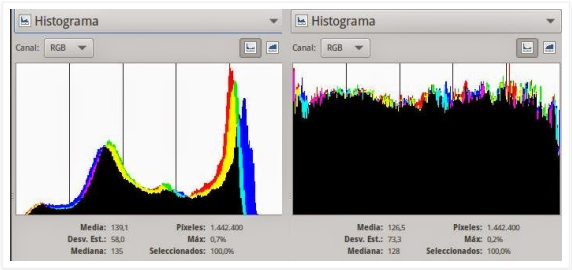
```
coloriza = ImageOps.colorize(imagenbn, (0,0,0), (0,100,200))
```

Ecualizar una imagen

Ecualizar es sinónimo de igualar. En este caso la función iguala el histograma de la imagen de entrada aplicando un mapeo no lineal con el fin de crear una distribución uniforme de los valores en la escala de grises, en la imagen de salida.

```
imagen = Image.open("nevando.jpg")
ecualizada = ImageOps.equalize(imagen)
ecualizada.show()
ecualizada.save("ecualizada.jpg")
```

A continuación, los histogramas de la imagen sin ecualizar y de la ecualizada:



Otras funciones disponibles: deform(), fit(), flip()

Dibujar sobre una imagen

```
from PIL import Image, ImageDraw
imagen = Image.open("nevando.jpg")
dibujo = ImageDraw.Draw(imagen)
dibujo.line((0, 90) + (800, 90), fill="white")
imagen.show()
imagen.save("linea.jpg")
```

Otras funciones:

Dibujar punto:

ImageDraw.Draw.point(xy, fill=None)

Dibujar arco:

ImageDraw.Draw.arc(xy, start, end, fill=None)

Dibujar polígono:

ImageDraw.Draw.polygon([(x1, y1),(x2, y2),...], fill=None, outline=None)

Dibujar rectángulo:

Draw.rectangle([(x0, y0),(x1, y1)], fill=None, outline=None)

Dibujar bitmap

ImageDraw.Draw.bitmap(xy, bitmap, fill=None)

Dibujar elipse:

ImageDraw.Draw.ellipse(xy, fill=None, outline=None)

Escribir texto sobre una imagen

```
from PIL import Image, ImageDraw, ImageFont
imagen = Image.open("linea.jpg")
imagen = imagen.convert("RGBA")
texto = Image.new('RGBA', imagen.size, (255,255,255,0))
fuente = ImageFont.truetype("Arabic Magic.ttf", 40)
```

```
dibujo = ImageDraw.Draw(texto)
dibujo.text((10, 40), "Pillow", font=fuente, fill=(255,255,255,128))
dibujo.text((110,40), "Python", font=fuente, fill=(255,255,255,255))
final = Image.alpha_composite(imagen, texto)
final.show()
final.save("texto.jpg")
```

Cambiar tamaño de fuentes:

```
ImageDraw.Draw.textsize(text, font=None)
```

Hay fuentes para descargar en Fontriver.

Ir al índice del tutorial de Python

Publicado por Pherkad en 10:27

Etiquetas: Pillow

Entrada más reciente Inicio Entrada antigua