

# **Script para crear Frames**

**Como usarlo**

**Fco. Javier Pérez Pacheco**

**`javi.pacheco@terra.es`**

**Script para crear Frames: Como usuario**

por Fco. Javier Pérez Pacheco

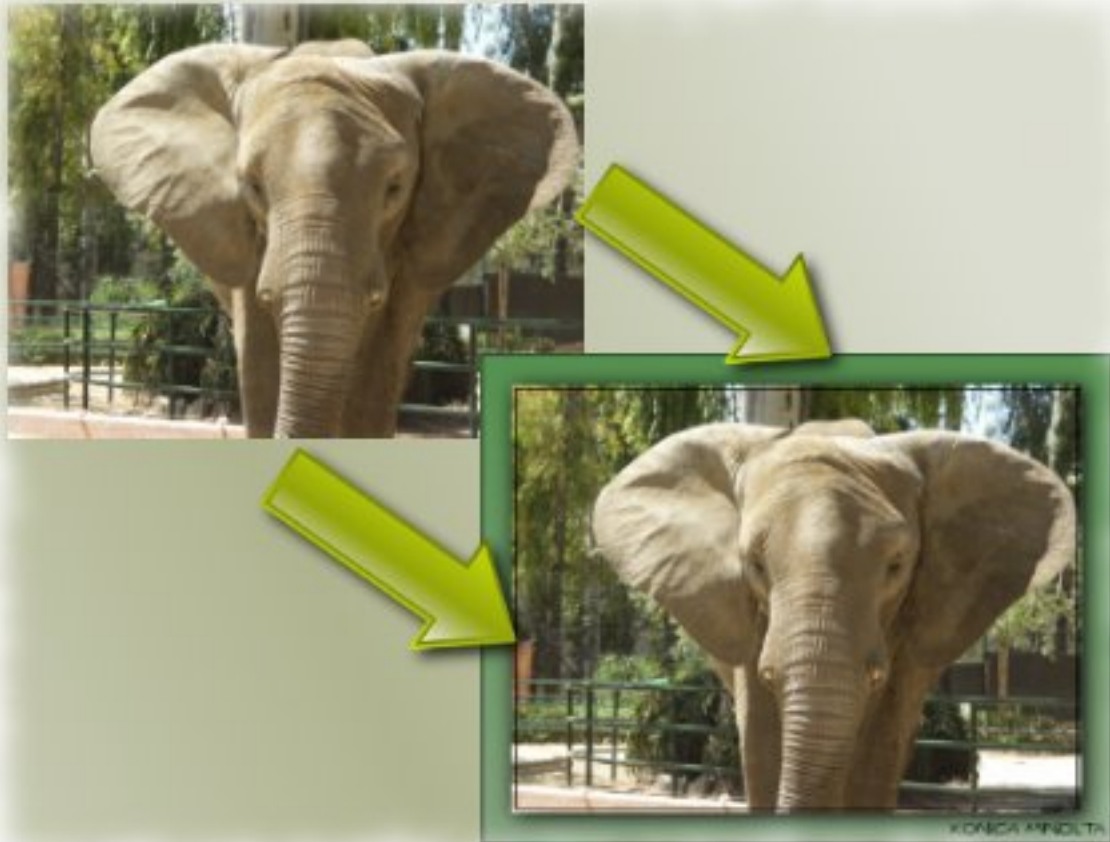
Copyright © 2005 Script Frames

# Tabla de contenidos

<b>1. Introducción .....</b>	<b>1</b>
<b>2. Instalación.....</b>	<b>2</b>
2.1. Prerequisitos .....	2
2.2. Como instalar el script.....	2
2.3. Información por defecto .....	3
<b>3. Empezamos a trabajar .....</b>	<b>5</b>
3.1. Conceptos generales .....	5
3.2. Como trabajar con fotografías.....	5
3.3. Como trabajar con directorios .....	8
3.4. Trabajar con configuraciones guardadas .....	10

# Capítulo 1. Introducción

Frames es un script realizado en python para Gimp que crea marcos para fotografías. En fotografía es muy común ponerle marcos a las fotos y este script te ayuda a ello. Además de poder hacer marcos a una fotografía, podemos crear marcos a un grupo de imágenes que tengamos en un directorio y podemos guardar nuestras configuraciones preferidas para poder utilizarlas en el futuro. A continuación dejo una foto de lo que podemos hacer con este script.



En las próximas secciones se explica todo lo necesario para poder utilizar este script correctamente.

# Capítulo 2. Instalación

A continuación se describe lo que necesitamos para poder instalar el script.

## 2.1. Prerequisitos

Antes de instalar el script tenemos que tener instalado varias cosas. Lo primero sería instalar la última versión de python (<http://www.python.org>), por supuesto el programa Gimp (<http://www.gimp.org>) e instalar el módulo de gimp para trabajar con python (<http://www.jamesh.id.au/software/pygimp/>). Con esto ya podemos empezar a trabajar.

## 2.2. Como instalar el script

Para instalar este script lo primero que tenemos que hacer es bajarnos el archivo frames.tar.gz. Este contiene 3 archivos:

- frames.py: archivo python con el script
- frames.xml: donde se guardan las configuraciones de usuario
- EXIF.py: archivo python para poder trabajar con datos EXIF. EXIF son datos que las cámaras guardan dentro de los JPG con información de la cámara en el momento de hacer la foto. Este script os lo podéis bajar de [http://home.cfl.rr.com/genecash/digital\\_camera.html](http://home.cfl.rr.com/genecash/digital_camera.html).

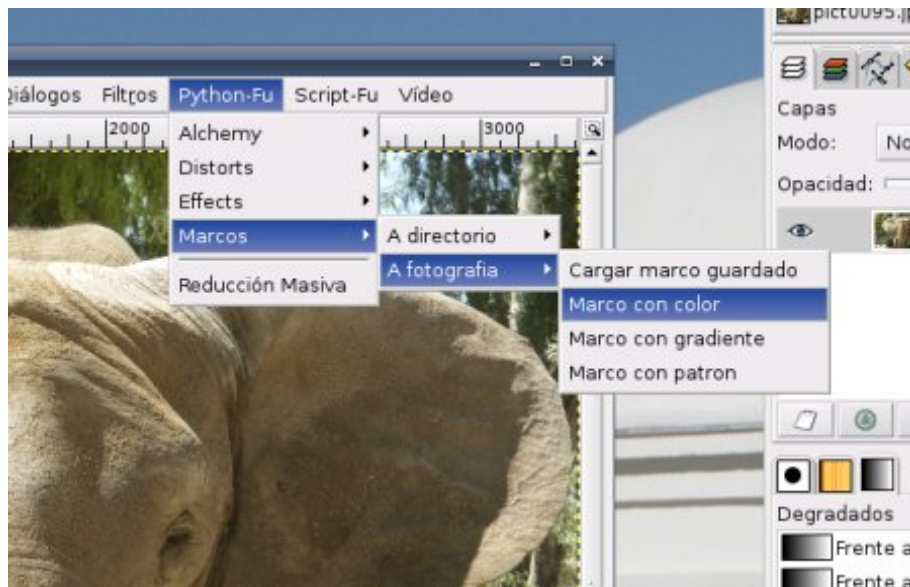
Estos 3 archivos tenemos que dejarlos juntos (muy importante) en el directorio plug-ins para gimp del usuario. También podríamos dejarlo en el directorio plug-ins de gimp general, pero posiblemente tengamos problemas ya que en ese directorio normalmente no tenemos permiso de ejecución y no podremos guardar nuestras configuraciones en el archivo XML. El directorio correcto sería:

```
/home/usuario/.gimp2.2/plug-ins
```

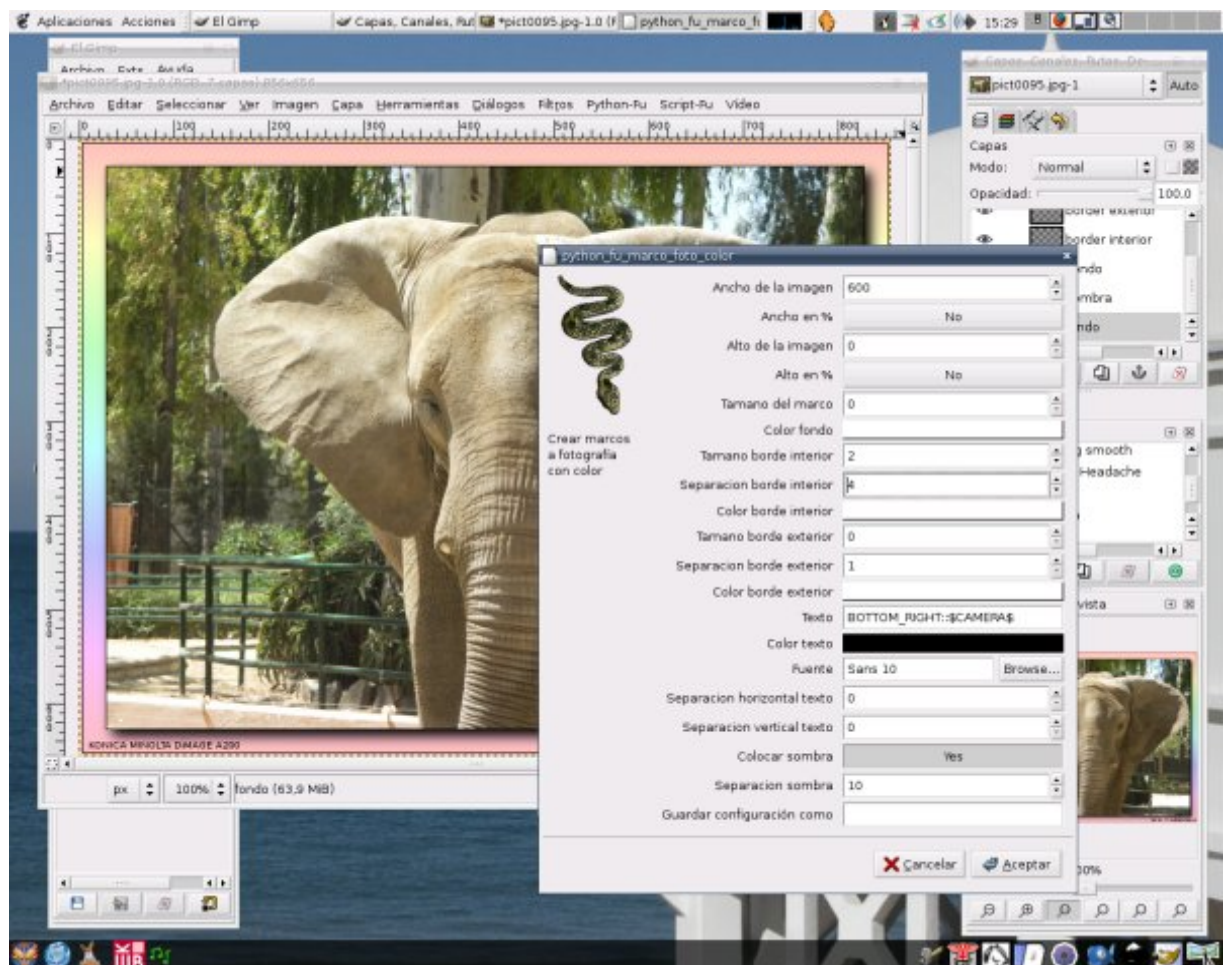
Por último, una vez copiados los archivos, tenemos que asegurarnos que el archivo frames.py tenga permiso de ejecución, si no es así hacemos lo siguiente:

```
# chmod +x frames.py
```

Una vez hecho todo esto ya tiene que funcionar nuestro script. Arrancamos gimp y abrimos una imagen cualquier de nuestro equipo, nos vamos al menú y vamos a /Python-Fu/Marcos y allí nos encontraremos con todos los script para empezar a trabajar con ellos. Esta es su apariencia:



Este es un pantallazo del script en acción.



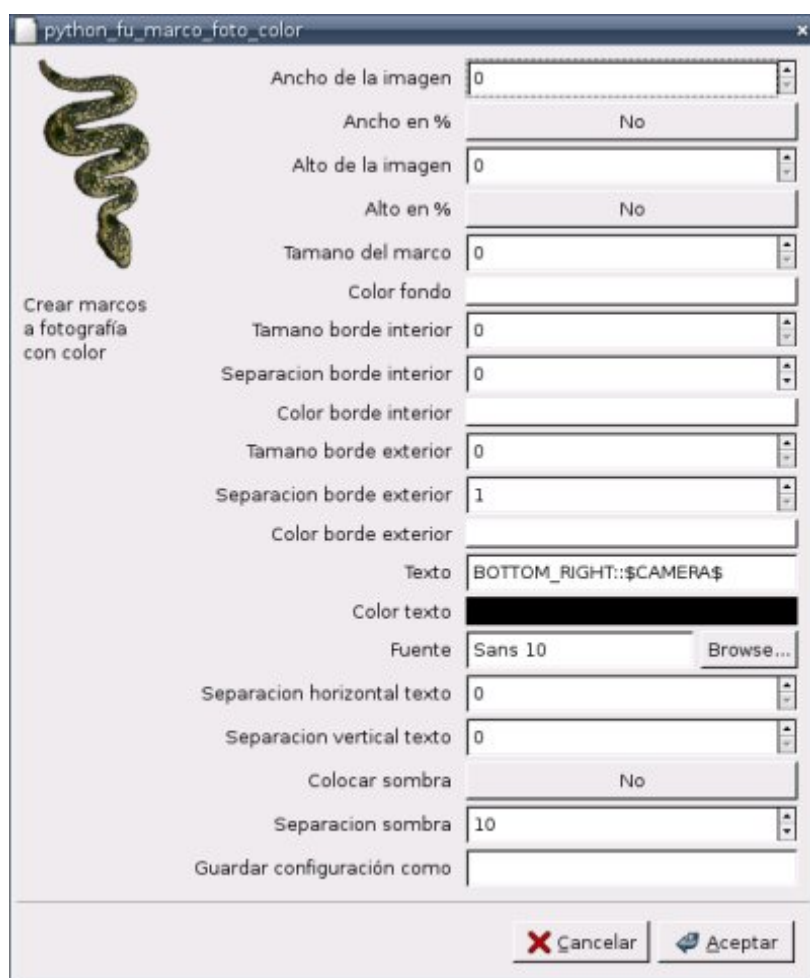
## 2.3. Información por defecto

Sólo si queremos podemos cambiar la información por defecto que aparece al arrancar los script. Si por ejemplo queremos que siempre por defecto aparezca como margen 20 pixels podemos hacerlo cambiando la información por defecto del script. Para ello abrimos el script con el editor que queramos y al principio existen muchas variables que podemos modificar. Durante las próximas secciones, cuando aprendemos a manejar el script, veremos más claro para que sirve esa información.

# Capítulo 3. Empezamos a trabajar

## 3.1. Conceptos generales

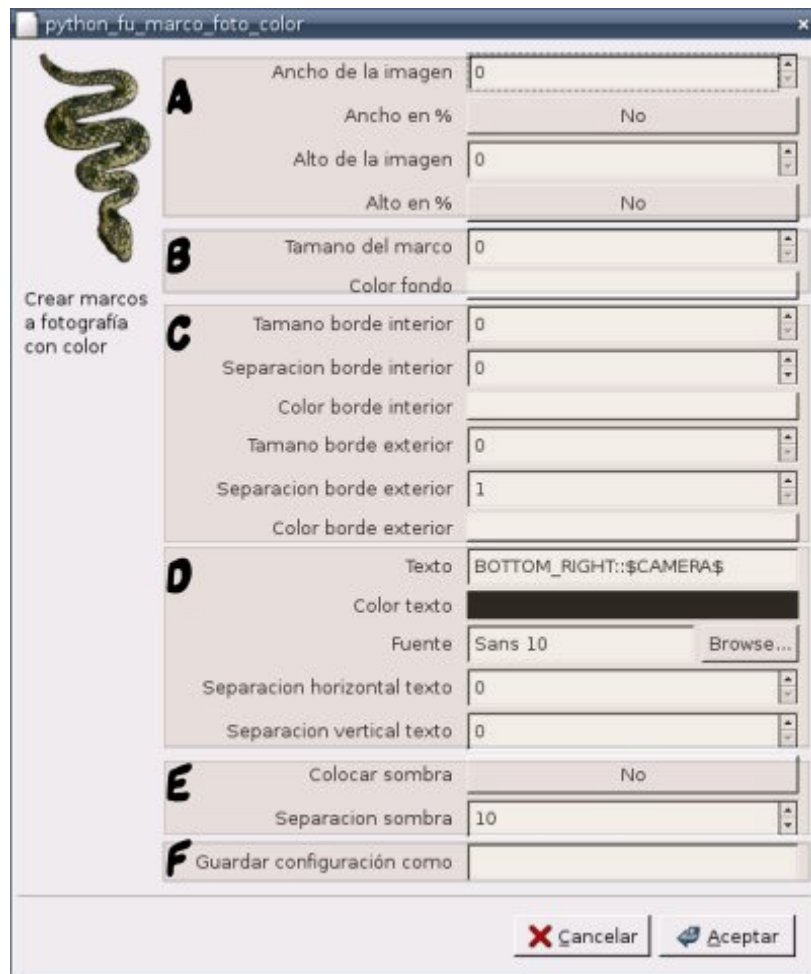
Nuestro script se divide en dos partes. Una parte para trabajar con fotografía (una sólo) y otra para trabajar con un grupo de estas (fotos de un directorio). Al irnos al menú de marcos de python-fu vemos directamente esta distinción. Dentro de cada uno de ellos vemos 4 script. El primero sería para trabajar con configuraciones guardadas previamente (el script viene ya con algunas configuraciones de inicio y tu puedes crear las tuyas propias) y luego tres script para crear marcos de color, gradiente y patrón. Todo tienen la misma estructura, pero según el que elijamos podemos crear (como marco) un color liso, un gradiente o un patrón. Esta es la forma la ventana de color, muy parecido a las otras dos:



A continuación describimos para que sirve cada cosa.

## 3.2. Como trabajar con fotografías

He dividido el script en diferentes partes para poder explicar mejor para que sirve. En esta imagen vemos las divisiones y como trabajarmos con ellas:



Ahora empieza lo bueno.

### 3.2.1. A: Redimensionado

Aquí podemos definir el alto y el ancho de la imagen. Podemos hacerlo con valores absolutos (tamaños en píxeles) o con tanto por ciertos de la imagen (pusando los boleanos). Si no queremos que la imagen se redimensione y se quede tal como está dejamos los valores de ancho y alto en 0. Si queremos que el script calcule el valor del ancho o alto en relación a uno dado, dejamos ese valor a 0. Por ejemplo, si nuestra imagen de de 800x400 y queremos dejar un ancho de 400 píxeles y el script calcule el valor relativo del alto, dejamos el alto a 0 y el ancho a 400 y el script creará la imagen a 400x200.

### 3.2.2. B: Marco de la imagen

Aquí definimos el tamaño de nuestro marco. Esta es la parte que cambiará de los 3 script según su color, gradiente o patrón. Así en cada uno podemos definir primero el tamaño y luego (según el script) definir el color, gradiente o patrón que queremos colocar a la imagen.

### 3.2.3. C: Bordes exteriores e interiores

Aquí podremos crear un borde exterior e interior a nuestra imagen. El borde interior sería el borde que crearía a

partir de la imagen y donde comienza el marco y el exterior empezará desde donde termina el marco y el final de la imagen. Podemos definir la separación de cada uno de los border, el color y el tamaño de estos. Lo mejor es hacer pruebas e ir viendo los resultados.

### 3.2.4. D: Textos

Aquí podremos colocar textos a nuestras imágenes. Podemos definir el color y la fuente de nuestro texto, también la separación del texto en vertical y en horizontal (este dependerá de la esquina en la que coloquemos nuestro texto) y por último lo más importante, el texto en sí. Podemos colocar el texto en seis partes de nuestra imagen: superior izquierda, derecha y centro e inferior izquierda, derecha y centro. La idea es formar una cadena que nos de toda la información. Nuestra cadena la dividiremos en dobles ampersand que nos darán los textos y las posiciones y dentro las dividiremos en dobles dos puntos que nos darán la posición en sí y el texto. Parece complicado pero no lo es. De principio las posiciones son:

- TOP\_LEFT: esquina superior izquierda
- TOP\_RIGHT: esquina superior derecha
- TOP\_CENTER: centro superior
- BOTTOM\_LEFT: esquina inferior izquierda
- BOTTOM\_RIGHT: esquina inferior derecha
- BOTTOM\_CENTER: centro inferior

Ponemos un caso práctico para verlo más claro. Por ejemplo, queremos poner un texto en la esquina inferior derecha con nuestro nombre y otro en la esquina superior izquierda con el nombre de la fotografía. Sería así:

```
BOTTOM_RIGHT::creado por javielinux&&TOP_LEFT::Nombre de la fotografía
```

Podemos ir separando por && los textos que queramos en las diferentes zonas de nuestra imagen. No es tan difícil ¿no?. Por último, tenemos la posibilidad de trabajar con datos EXIF que ha creado nuestra cámara digital y lo podemos introducir como textos en nuestra imagen. Así podemos colocar variables en nuestros textos que insertarán estos datos exif. Las variables se colocan con símbolos \$ a cada lado de la variable y las posibles variables son:

- \$CAMERA\$: nombre de la marca de la cámara con la que hicimos la imagen
- \$MODEL\$: nombre del modelo de la cámara con la que hicimos la imagen
- \$APERTURE\$: valor de apertura
- \$DATE\$: día y hora en la que se hizo la fotografía
- \$EXPOSURETIME\$: tiempo de exposición
- \$FOCALLENGTH\$: focal
- \$ISO\$: valor iso de la imagen
- \$SHUTTERSPEED\$: velocidad de obturación
- \$MAXAPERTURE\$: máxima apertura
- \$FNUMBER\$: número "f"
- \$RESOLUTIONX\$: resolución x de la imagen
- \$RESOLUTIONY\$: resolución y de la imagen
- \$RESOLUTIONUNIT\$: unidad de resolución

Para verlo mejor un caso práctico. Queremos colocar en el centro superior el nombre y el modelo de nuestra cámara y en la esquina inferior izquierda varios datos EXIF.

```
BOTTOM_LEFT::ISO: $ISO$ Resolución: $RESOLUTIONX$ -  
$RESOLUTIONY$&&TOP_CENTER::$CAMERA$ $MODEL$
```

Cuando se crean varios se ve más sencillo.

### **3.2.5. E: Sombra**

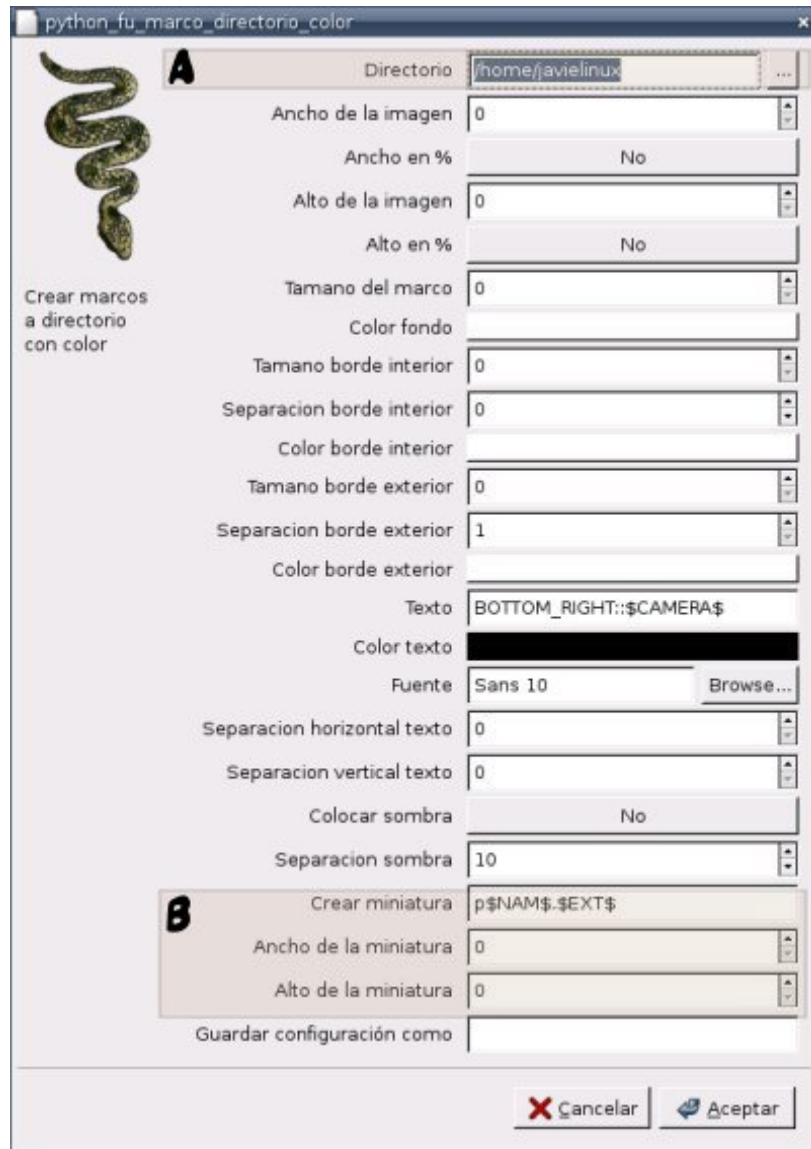
Aquí definimos si queremos una sombra a la imagen. Cuando decimos imagen, es a la imagen principal no a la imagen con el marco. También podemos definir la separación de la sombra con la imagen.

### **3.2.6. F: Guardar configuración**

Si nos a gustado nuestro script y queremos guardarlo para poder utilizarlo en un futuro le podemos un nombre aquí. En el caso de no querer guardarlo, lo dejamos en blanco.

## **3.3. Como trabajar con directorios**

Ya hemos visto como trabajar con fotografías independientemente, ahora explicamos como trabajar con un directorio completo. El trabajo con directorios es muy parecido al de fotografía pero tiene algunas cosas más. Aquí dejamos una imagen las zonas nuevas que vamos a explicar.



### 3.3.1. A: Directorio

Aquí establecemos el directorio que contiene las imágenes con las que vamos a trabajar. Debería de tener sólo imágenes JPG para no tener sorpresas.

### 3.3.2. B: Miniaturas

Podemos decir a nuestro script que nos cree una miniatura (thumbnails) de nuestra imagen a parte de las modificaciones que va hacer a la imagen. Si el texto "Crear miniatura" está en blanco no creará la imagen si está relleno, creará la imagen con un formato que escribiremos ahí. El formato tiene dos variables \$NAM\$ para el nombre y \$EXT\$ para la extensión. Imaginemos que queremos que nuestra miniatura tenga la palabra "mini\_" delante del nombre para distinguirla de la imagen real. Sería así:

mini\_ \$NAM\$. \$EXT\$

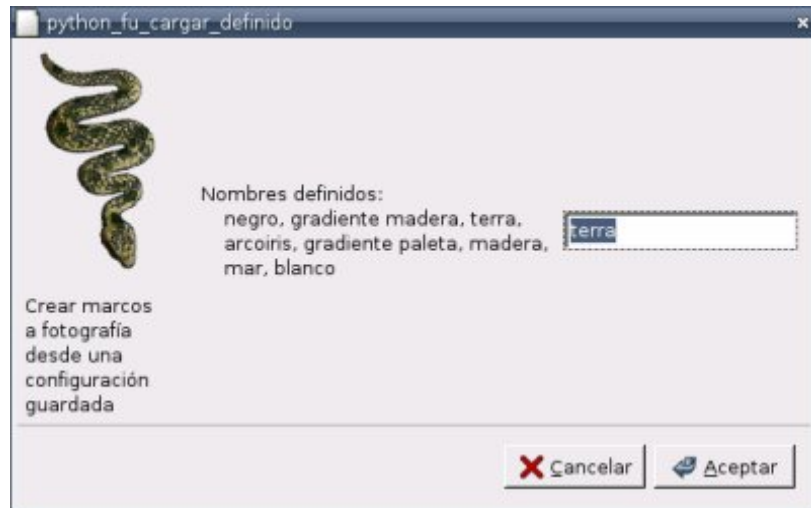
Las variables las reemplazará por su contenido en el nombre de la imagen original, si por ejemplo nuestra imagen se llamaba "foto.jpg", su miniatura se llamará "mini\_foto.jpg". También podemos definir el ancho y algo

de la miniatra. Si dejamos uno a 0, calcula la relación de la imagen mediante el otro dato como a la hora de redimensionar la imagen.

## 3.4. Trabajar con configuraciones guardadas

### 3.4.1. Configuraciones en imágenes

Para trabajar con configuraciones nada más que tenemos que saber el nombre. Como vemos en la siguiente imagen, a la izquierda aparecen separados por comas los nombres de las configuraciones que tenemos guardadas, y a la derecha tenemos un campo de texto donde tenemos que escribir el nombre.



Ahora slo me queda excusarme de lo cutre que está hecho. No he encontrado mejor forma de hacerlo. Se que no es la mejor forma posible, pero después de mucho buscar no he encontrado forma de crear combos con python-fu. La única forma de hacerlo era mediante radio-button, pero en python-fu los radios se generan a partir de tuplas y estas con constantes, por lo que no podía crear una tupla dinámicamente a partir de los datos del XML. Si existe una forma mejor, por favor comuniquemelo.

### 3.4.2. Configuraciones en directorios

Es igual que la de fotografía, pero tenemos que dar un directorio donde se encuentran las imágenes y podemos crear las miniatras de estas. Todo esto se explicó en la sección anterior.

