Unidad 10

Imagen: Visualización y Tinta

Elementos que se introducen en esta Unidad:

```
PImage, loadImage(), image()
tint(), noTint()
```

Las fotografías digitales son muy diferentes si las comparamos con las fotografías analógicas capturadas en film. Al igual que las pantallas de ordenador, las imágenes digitales son rejillas rectangulares de color. Las dimensiones de las imágenes digitales se miden en unidades de píxeles. Si una imagen es de 320 píxeles de ancho y 240 píxeles de alto, tiene 76.800 píxeles totales. Si una imagen es de 1280 píxeles de ancho y 1024 píxeles de ancho, el número total de píxeles es de 1310720 (1,3 megapíxeles). Cada imagen digital tiene una profundidad de color. La profundidad de color se refiere al número de bits utilizado para almacenar cada píxel. Si la profundidad de color de una imagen es 1, cada píxel puede ser uno de dos valores, por ejemplo, negro o blanco. Si la profundidad de color es de 4, cada píxel puede ser uno de 16 valores. Si la profundidad de color de una imagen es de 8, cada píxel puede ser uno de 256 valores. En cuanto a la imagen, se muestra con diferentes profundidades de color. Esto afecta a la apariencia:

1-bit (2 colores) 2-bit (4 colores) 4-bit (16 colores) 8-bit (256 colores)

Hoy en día, la mayoría de las computadoras trabajan con una profundidad de 24 bits, lo que quiere decir uno de 16777216 colores. Se lo llama comúnmente como *millones de colores*.
Las imágenes digitales se **componen de números** que representan los colores. El formato de archivo de una imagen determina como los números se ordenan en el archivo. Algunos formatos de archivo de almacenamiento de los datos de color, utilizan ecuaciones de matemáticas complejas para comprimir los datos y reducir el tamaño resultante del archivo final. Un programa que se carga un archivo de imagen debe conocer el formato de la misma para que pueda traducir los datos de archivo en la imagen esperada. Existen diferentes tipos de formatos de imágenes digitales que sirven para necesidades específicas. Processing pueden cargar archivos GIF, JPEG, y PNG, junto con algunos otros formatos como se describe en la referencia. Si no tiene su imagen en uno de estos formatos, se puede convertir a otros tipos de formatos con programas como GIMP o Adobe Photoshop.

Format	Extension	Profundidad de color	Transparencia
GIF	.gif	1-bit a 8-bit	1-bit
JPEG	.jpg	24-bit	No Posee
PNG	.png	1-bit a 24-bit	8-bit

-Visualización

Processing puede cargar imágenes, mostrarlas en pantalla, cambiar su posición, tamaño, color, tinta y transparencia. El tipo de dato que se utiliza para almacenar imágenes es conocido por el nombre de `PImage`. Al igual que los números enteros son almacenados en variables del tipo `int`, y los valores `true` y `false` en variables del tipo `boolean`, las imágenes son almacenadas en variables tipo `PImage`. Antes de mostrarla en al ventana, es necesario cargar la imagen con la función `loadImage()`. Se debe revisar con cuidado que además de ingresar el nombre correcto también este la extensión de formato correcta (por ejemplo, *kant.jpg*, *woah.gif*, *hola.png*). Las imágenes a cargar deben encontrarse en la carpeta *data*, dentro de la carpeta del *sketch*. Existe una forma muy sencilla de añadir una imagen a esa carpeta sin cometer ningún error. En la

barra de menú, dentro de la opción “Skecth”, se encuentra el comando “Add File”. Se podrá entonces navegar por su ordenador y buscar el archivo deseado. Una vez que se encuentra, se le da click a “Open” para añadir el archivo a la carpeta *data* de su skecth. Con la imagen dentro de la carpeta, habiéndose declarado un nuevo `PImage` y, posteriormente, cargado en ese `PImage` un archivo de imagen, este puede ser mostrado en pantalla con la función `image()`:

```
image(nombre, x, y)
image(nombre, x, y, ancho, alto)
```

Los parámetros de `image()` determinan la posición y el tamaño de la imagen (similar a otras funciones vistas previamente). El *nombre* es tan solo el nombre de la variable `PImage` donde se encuentra almacenada nuestra imagen. Si no se utilizan los parámetros *ancho* y *alto*, la imagen será mostrada en su tamaño original.

```
PImage img;
//La imagen debe estar guardada en la carpeta “data”
img = loadImage("arch.jpg");
image(img, 0, 0);
```

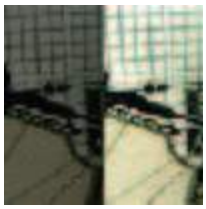
```
PImage img;
//La imagen debe estar guardada en la carpeta “data”
img = loadImage("arch.jpg");
image(img, 20, 20, 60, 60);
```

-Color de la Imagen y Transparencia

Las imágenes pueden ser coloreadas con la función `tint()`. Esta actúa del mismo modo que `fill()` y `stroke()`, solo que afecta únicamente a las imágenes:

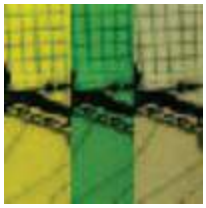
```
tint(gris)
tint(gris, alfa)
tint(valor1, valor2, valor3)
tint(valor1, valor2, valor3, alfa)
tint(color)
```

Todas las imágenes que se dibujan luego de declarada la función `tint()`, será pintada por ella. Si en algún momento se desea que no se pinte, se puede utilizar la función `noTint()`.



```
PImage img;
img = loadImage("arch.jpg");
tint(102);              //Pinta de gris
image(img, 0, 0);
noTint();               //Desactiva tint()
image(img, 50, 0);
```

```
PImage img;
img = loadImage("arch.jpg");
tint(0, 153, 204);      //Pinta de azul
image(img, 0, 0);
noTint();               //Desactiva tint()
image(img, 50, 0);
```



```
color amarillo = color(220, 214, 41);
color verde = color(110, 164, 32);
color tan = color(180, 177, 132);
PImage img;
img = loadImage("arch.jpg");
tint(amarillo);
image(img, 0, 0);
tint(verde);
image(img, 33, 0);
tint(tan);
image(img, 66, 0);
```

Los parámetros de `tint()` están ligados a lo establecido por `colorMode()`. Hay que tener cuidado de no cambiar el modo de color o el rango del mismo, ya que eso afectará los parámetros de `tint()`.
Para generar transparencia simplemente utilizaremos la misma función `tint()` pero con un valor de blanco (255 en un modo RGB).

```
PImage img;
img = loadImage("arch.jpg");
background(255);
tint(255, 102);  //Alfa de 102 sin cambiar el color de pintura
image(img, 0, 0, 100, 100);
tint(255, 204, 0, 153);       //Pinta amarillo con una
image(img, 20, 20, 100, 100); //transparencia de 153
```

```
PImage img;
img = loadImage("arch.jpg");
background(255);
tint(255, 51);
//Dibuja las imágenes 10 veces moviendo las mismas
//a la derecha
for (int i = 0; i < 10; i++) {
      image(img, i*10, 0);
}
```

Las imágenes de formato GIF y PNG pueden almacenar la transparencia de la imagen. Mientras que las GIF solo pueden convertir un color (ya que solo posee 1-bit de transparencia). En cambio, las imágenes PNG tienen una profundidad de transparencia de 8-bit, por lo tanto admiten 256 colores de transparencia.

```
//Carga una imagen GIF con 1-bit de transparencia
PImage img;
img = loadImage("archTrans.gif");
background(255);
image(img, 0, 0);
image(img, -20, 0);
```

```
//Carga una imagen PNG con 8-bit de transparencia
PImage img;
img = loadImage("arch.png");
background(255);
image(img, 0, 0);
image(img, -20, 0);
```