Manual de Referencia Rápida de PNGwriter

Versión 0.3.7 (12 / X / 2003)

http://pngwriter.sourceforge.net/

Introducción

Este es el Manual de Referencia Rápida de PNGwriter. Es un resumen de las funciones de PNGwriter. Para una descripción más detallada, ver el archivo header pngwriter.h

Notas Generales

*Es importante recordar que todas las funciones que aceptan un argumento del tipo "const char *" tambien aceptaran "char *". Tambien es importante recordar que cuando una funcion tiene un coeficiente de color como argumento, ese argumento puede ser un into de 0 a 65535 o un double de 0.0 a 1.0.*

Constructor

El constructor requiere el ancho y la altura de la imagen, el color de fondo para la imagen, y el nombre del archivo (un puntero o simplemente "miarchivo.png").

```
pngwriter();
pngwriter(const pngwriter &rhs);
pngwriter(int ancho, int altura, int backgroundcolour, char * filename);
pngwriter(int ancho, int altura, double backgroundcolour, char * filename);
pngwriter(int ancho, int altura, int backgroundcolour, const char * filename);
pngwriter(int ancho, int altura, double backgroundcolour, const char *
filename);
```

Operador de Asignación

PNGwriter sobrecarga el operador de asignación =.

```
pngwriter & operator = (const pngwriter & rhs);
```

Plotear

Los pixeles se enumeran partiendo desde (1, 1) y van hasta (ancho, altura). Al igual como sucede con muchas funciones en PNGwriter, esta funcion ha sido sobrecargada para aceptar argumentos tipo int o double para los coeficientes de color. Si son de tipo int, van desde 0 a 65535. Si son de tipo double, van desde 0.0 hasta 1.0.

```
void plot(int x, int y, int rojo, int verde, int azul);
void plot(int x, int y, double rojo, double verde, double azul);
```

Plotear HSV

Igual que el anterior, pero con esta funcion el color de un pixel en la posicion (x, y) puede ser cambiado en el espacio de colores de Hue, Saturation, Value. Los coeficioentes de color van desde 0 a 65535 si son de tipo int, o desde 0.0 hasta 1.0 si

son de tipo double.

```
void plotHSV(int x, int y, double hue, double saturation, double value);
void plotHSV(int x, int y, int hue, int saturation, int value);
```

Read

Con esta funcion averiguamos el color del pixel (x, y). Si "color" es 1, devolvera el coeficiente de color rojo, si es 2, el verde, si es 3, el azul. El valor devuelto sera de tipo int y estara entre 0 y 65535.

```
int read(int x, int y, int color);
```

Read, Promedio

Lo mismo que el anterior, pero el promedio de los tres coeficientes de color es devuelto.

```
int read(int x, int y);
```

dRead

Igual que Read, pero el valor devuelto sera de tipo double y estara entre 0.0 y 1.0.

```
double dread(int x, int y, int color);
```

dRead, Promedio

Lo mismo que el anterior, pero el promedio de los tres coeficientes de color es devuelto.

```
double dread(int x, int y);
```

Read HSV

Con esta funcion averiguamos el color del pixel (x, y), pero en el espacio de colores de HSV. Si "color" es 1, devolvera el coeficiente de color Hue, si es 2, el Saturation, si es 3, el Value. El valor devuelto sera de tipo int y estara entre 0 y 65535.

```
int readHSV(int x, int y, int color);
```

dRead HSV

Igual que el anterior, pero el valor devuelto sera de tipo double y estara entre 0.0 y 1.0.

```
double dreadHSV(int x, int y, int color);
```

Borrar todo

Toda la imagen se pinta de negro.

```
void clear(void);
```

Cerrar

Cerrar la instancia de la clase, y escribir la imagen al disco.

```
void close(void);
```

Cambiar el nombre

Para cambiar el nombre del archivo una vez que la instancia del PNGwriter ha sido creada. Si el argumento es un long unsigned int, por ejemplo 77, el nombre del archivo será 000000077.png.

```
void pngwriter_rename(char * nuevonombre);
void pngwriter_rename(const char * nuevonombre);
void pngwriter_rename(long unsigned int index);
```

Figuras

Disponibles en version int y double

```
void line(int xdesde, int ydesde, int xhasta, int yhasta, int rojo, int
verde,int azul);
void line(int xdesde, int ydesde, int xhasta, int yhasta, double rojo, double
verde,double azul);
void square(int xdesde, int ydesde, int xhasta, int yhasta, int rojo, int
verde,int azul);
void square(int xdesde, int ydesde, int xhasta, int yhasta, double rojo,
double verde,double azul);
void filledsquare(int xdesde, int ydesde, int xhasta, int yhasta, int rojo,
int verde,int azul);
void filledsquare(int xdesde, int ydesde, int xhasta, int yhasta, double rojo,
double verde,double azul);
void circle(int xcentro, int ycentro, int radio, int rojo, int verde, int
azul);
void circle(int xcentro, int ycentro, int radio, double rojo, double verde,
double azul);
void filledcircle(int xcentro, int ycentro, int radio, int rojo, int verde,
int azul);
void filledcircle(int xcentro, int ycentro, int radio, double rojo, double
verde, double azul);
```

Curva Bezier

(llamado así en honor del frances Pierre Bézier de Regie Renault) Una colección de fórmulas para describir líneas y superficies curvas, usados por primera vez en 1972 para modelar las lineas curvas de automóviles. *(de The Free On-line Dictionary of Computing)*
Ver http://www.moshplant.com/direct-or/bezier/ *para una de las muchas descripciones disponibles acerca de las curvas de bezier. Hay cuatro puntos usados para definir la curva: los dos extremos se llaman los puntos de anclaje, mientras que los otros puntos, que describen la curvatura, se llaman puntos de control. Moviendo los puntos de control podemos modificar la forma de la curva.*

```
void bezier(  int startPtX, int startPtY,
              int startControlX, int startControlY,
              int endPtX, int endPtY,
              int endControlX, int endControlY,
              double red, double green, double blue);
```

```
void bezier(  int startPtX, int startPtY,
              int startControlX, int startControlY,
              int endPtX, int endPtY,
              int endControlX, int endControlY,
              int red, int green, int blue);
```

Leer Desde Archivo

Abre el archivo PNG ya existente y lo incorpora a su almacenamiento.

```
void readfromfile(char * name);
void readfromfile(const char * name);
```

Averiguar Altura

Cuando abres un PNG ya existente, puedes averiguar su altura con esta funcion.

```
int getheight(void);
```

Averiguar Ancho

Cuando abres un PNG ya existente, puedes averiguar su ancho con esta funcion.

```
int getwidth(void);
```

Excribir PNG

Escribe el PNG al disco. Después de esto, todavía puedes seguir alterando la instancia de PNGwriter.

```
void write_png(void);
```

Elegir Nivel de Compresion

Elegir el nivel de compresion que sera usada para la imagen. -1 selecciona la compresion por defecto, 0 es ninguna, 9 es la mejor compresion.

```
void setcompressionlevel(int nivel);
```

Averiguar Bit Depth

Cuando abres un PNG ya existente, puedes averiguar su profundidad de bits con esta funcion.

```
int getbitdepth(void);
```

Averiguar Colour Type

Cuando abres un PNG ya existente, puedes averiguar su color type con esta funcion.

```
int getcolortype(void);
```

Cambiar Gamma

Cambiar el gamma de la imagen (filegamma). El valor prefijado de 0.5 deberia estar bien.

```
void setgamma(double gamma);
```

Averiguar Gamma

Averigua el coeficiente de gamma de la imagen. Esto es experimental.□

```
double getgamma(void);
```

Cambiar Texto

Cambia el texto contenido en el header del archivo PNG. Si esta funcion no se llama, se usara el texto por defecto.

```
void settext(char * title, char * author, char * description, char *
software);
void settext(const char * title, const char * author, const char *
description, const char * software);
```

Versión

Da la versión de PNGwriter.

```
double version(void);
```