# libcaca Reference Manual

0.3

Generated by Doxygen 1.3.4

Thu Dec 18 01:29:07 2003

# Contents



# 1 libcaca developer documentation

## 1.1 Introduction

*libcaca* is a graphics library that outputs text instead of pixels, so that it can work on older video cards or text terminals. It is not unlike the famous AAlib library. libcaca needs a terminal to work, thus it should work on all Unix systems (including Mac OS X) using either the slang library or the ncurses library, on DOS using the conio library, and on Windows systems using either slang or ncurses (through Cygwin emulation) or conio.

*libcaca* is free software, released under the GNU Lesser General Public License. This ensures that *libcaca* will always remain free software.

## 1.2 The libcaca API

The complete *libcaca* programming interface is available from the **caca.h**(p. 8) file.

## 1.3 Environment variables

Some environment variables can be used to change the behaviour of *libcaca* without having to modify the program which uses it. These variables are:

- **CACA_BACKGROUND**: set the background type.
  - `solid` uses solid coloured backgrounds for all characters. This feature does not work with all terminal emulators. This is the default choice.
  - `black` uses only black backgrounds to render characters.
- **CACA_ANTIALIASING**: set the antialiasing mode. Antialiasing smoothens the rendered image and avoids the commonly seen staircase effect.
  - `none` disables antialiasing.
  - `prefilter` uses a simple prefilter antialiasing method. This is the default choice.
- **CACA_DITHERING**: set the dithering mode. Dithering is necessary when rendering a picture that has more colours than the usually available palette.
  - `none` disables dithering.
  - `ordered2` uses a 2x2 bayer matrix for dithering.
  - `ordered4` uses a 4x4 bayer matrix for dithering. This is the default choice.
  - `ordered8` uses a 8x8 bayer matrix for dithering.
  - `random` uses random dithering.

# 2 libcaca File Index

## 2.1 libcaca File List

Here is a list of all documented files with brief descriptions:



# 3 libcaca File Documentation

## 3.1 bitmap.c File Reference

Bitmap functions.

### Data Structures

- struct **caca_bitmap**

### Defines

- #define **DENSITY_CHARS** 13
- #define **XRATIO** 5∗5
- #define **YRATIO** 3∗3
- #define **HRATIO** 2∗2
- #define **FUZZINESS** XRATIO ∗ 0x800

### Typedefs

- typedef unsigned char **uint8_t**
- typedef unsigned short **uint16_t**

- typedef unsigned int **uint32_t**

**Functions**

- caca_bitmap * **caca_create_bitmap** (unsigned int bpp, unsigned int w, unsigned int h, unsigned int pitch, unsigned int rmask, unsigned int gmask, unsigned int bmask, unsigned int amask)

  *Create an internal bitmap object.*

- void **caca_set_bitmap_palette** (struct caca_bitmap *bitmap, unsigned int red[], unsigned int green[], unsigned int blue[], unsigned int alpha[])

  *Set the palette of an 8bpp bitmap object.*

- void **caca_free_bitmap** (struct caca_bitmap *bitmap)

  *Free the memory associated with a bitmap.*

- void **caca_draw_bitmap** (int x1, int y1, int x2, int y2, const struct caca_bitmap *bitmap, void *pixels)

  *Draw a bitmap on the screen.*

**Variables**

- enum **caca_feature _caca_background**
- enum **caca_feature _caca_dithering**
- enum **caca_feature _caca_antialiasing**

### 3.1.1 Detailed Description

Bitmap functions.

**Version:**
$Id: **bitmap.c**(p. 2) 154 2003-12-18 00:11:52Z sam $

**Author:**
Sam Hocevar <sam@zoy.org>

This file contains bitmap blitting functions.

### 3.1.2 Function Documentation

#### 3.1.2.1 struct caca_bitmap* caca_create_bitmap (unsigned int *bpp*, unsigned int *w*, unsigned int *h*, unsigned int *pitch*, unsigned int *rmask*, unsigned int *gmask*, unsigned int *bmask*, unsigned int *amask*)

Create an internal bitmap object.

**Parameters:**
***bitmap*** The bitmap depth in bits per pixel.

***w*** The bitmap width in pixels.

***h*** The bitmap height in pixels.

***pitch*** The bitmap pitch in bytes.

***rmask*** The bitmask for red values.

***gmask*** The bitmask for green values.

***bmask*** The bitmask for blue values.

***amask*** The bitmask for alpha values.

**Returns:**
A bitmap object or NULL upon error.

#### 3.1.2.2 void caca_draw_bitmap (int *x1*, int *y1*, int *x2*, int *y2*, const struct caca_bitmap * *bitmap*, void * *pixels*)

Draw a bitmap on the screen.

**Parameters:**

***x1*** X coordinate of the upper-left corner of the drawing area.

***y1*** Y coordinate of the upper-left corner of the drawing area.

***x2*** X coordinate of the lower-right corner of the drawing area.

***y2*** Y coordinate of the lower-right corner of the drawing area.

***bitmap*** The bitmap object to be drawn.

***pixels*** A pointer to the bitmap's pixels.

**Returns:**
void

#### 3.1.2.3 void caca_free_bitmap (struct caca_bitmap * *bitmap*)

Free the memory associated with a bitmap.

**Parameters:**

***bitmap*** The bitmap object to be freed.

**Returns:**
void

#### 3.1.2.4 void caca_set_bitmap_palette (struct caca_bitmap * *bitmap*, unsigned int *red*[], unsigned int *green*[], unsigned int *blue*[], unsigned int *alpha*[])

Set the palette of an 8bpp bitmap object.

**Parameters:**

***bpp*** The bitmap object.

***red*** An array of 256 red values.

***green*** An array of 256 green values.

***blue*** An array of 256 blue values.

***alpha*** An array of 256 alpha values.

**Returns:**
void

## 3.2 box.c File Reference

Simple box drawing functions.

### Functions

- void **caca_draw_box** (int x1, int y1, int x2, int y2, char c)
  *Draw a box on the screen using the given character.*
- void **caca_draw_thin_box** (int x1, int y1, int x2, int y2)
  *Draw a thin box on the screen.*
- void **caca_fill_box** (int x1, int y1, int x2, int y2, char c)
  *Fill a box on the screen using the given character.*

### 3.2.1 Detailed Description

Simple box drawing functions.

**Version:**
$Id: **box.c**(p. 5) 154 2003-12-18 00:11:52Z sam $

**Author:**
Sam Hocevar <sam@zoy.org>

This file contains box drawing functions, both filled and outline.

### 3.2.2 Function Documentation

#### 3.2.2.1 void caca_draw_box (int *x1*, int *y1*, int *x2*, int *y2*, char *c*)

Draw a box on the screen using the given character.

**Parameters:**
***x1*** X coordinate of the upper-left corner of the box.
***y1*** Y coordinate of the upper-left corner of the box.
***x2*** X coordinate of the lower-right corner of the box.
***y2*** Y coordinate of the lower-right corner of the box.
***c*** Character to draw the box outline with.

**Returns:**
void

#### 3.2.2.2 void caca_draw_thin_box (int *x1*, int *y1*, int *x2*, int *y2*)

Draw a thin box on the screen.

**Parameters:**
***x1*** X coordinate of the upper-left corner of the box.

***y1*** Y coordinate of the upper-left corner of the box.
***x2*** X coordinate of the lower-right corner of the box.
***y2*** Y coordinate of the lower-right corner of the box.

**Returns:**
void

#### 3.2.2.3 void caca_fill_box (int *x1*, int *y1*, int *x2*, int *y2*, char *c*)

Fill a box on the screen using the given character.

**Parameters:**
***x1*** X coordinate of the upper-left corner of the box.
***y1*** Y coordinate of the upper-left corner of the box.
***x2*** X coordinate of the lower-right corner of the box.
***y2*** Y coordinate of the lower-right corner of the box.
***c*** Character to fill the box with.

**Returns:**
void

## 3.3 caca.c File Reference

Main *libcaca* functions.

### Functions

- int **caca_init** (void)
  *Initialise libcaca.*
- unsigned int **caca_get_width** (void)
  *Get the screen width.*
- unsigned int **caca_get_height** (void)
  *Get the screen height.*
- const char * **caca_get_color_name** (enum **caca_color** color)
  *Translate a colour value into its name.*
- enum **caca_feature caca_get_feature** (enum **caca_feature** feature)
  *Get the current value of a feature.*
- void **caca_set_feature** (enum **caca_feature** feature)
  *Set a feature.*
- const char * **caca_get_feature_name** (enum **caca_feature** feature)
  *Translate a feature value into its name.*
- void **caca_end** (void)
  *Uninitialise libcaca.*

### 3.3.1 Detailed Description

Main *libcaca* functions.

**Version:**
$Id: **caca.c**(p. 6) 154 2003-12-18 00:11:52Z sam $

**Author:**
Sam Hocevar <sam@zoy.org>

This file contains the main functions used by *libcaca* applications to initialise the library, get the screen properties, set the framerate and so on.

### 3.3.2 Function Documentation

#### 3.3.2.1 void caca_end (void)

Uninitialise libcaca.

**Returns:**
void

#### 3.3.2.2 const char* caca_get_color_name (enum caca_color *color*)

Translate a colour value into its name.

**Parameters:**
***color*** The colour value.

**Returns:**
A static string containing the colour's name.

#### 3.3.2.3 enum caca_feature caca_get_feature (enum caca_feature *feature*)

Get the current value of a feature.

**Parameters:**
***feature*** The requested feature.

**Returns:**
The current value of the feature.

#### 3.3.2.4 const char* caca_get_feature_name (enum caca_feature *feature*)

Translate a feature value into its name.

**Parameters:**
***feature*** The feature value.

**Returns:**
A static string containing the feature's name.

#### 3.3.2.5 unsigned int caca_get_height (void)

Get the screen height.

**Returns:**
The screen height, in character cells.

#### 3.3.2.6 unsigned int caca_get_width (void)

Get the screen width.

**Returns:**
The screen width, in character cells.

#### 3.3.2.7 int caca_init (void)

Initialise libcaca.

**Returns:**
0 upon success, a non-zero value if an error occurs.

#### 3.3.2.8 void caca_set_feature (enum caca_feature *feature*)

Set a feature.

**Parameters:**
***feature*** The wanted feature.

**Returns:**
void

## 3.4 caca.h File Reference

The *libcaca* public header.

**Defines**

- #define **CACA_BACKGROUND_MIN** 0x11
- #define **CACA_BACKGROUND_MAX** 0x12
- #define **CACA_ANTIALIASING_MIN** 0x21
- #define **CACA_ANTIALIASING_MAX** 0x22
- #define **CACA_DITHERING_MIN** 0x31
- #define **CACA_DITHERING_MAX** 0x35
- #define **caca_dithering caca_feature**
- #define **caca_set_dithering** caca_set_feature
- #define **caca_get_dithering_name** caca_get_feature_name
- #define **CACA_DITHER_NONE** CACA_DITHERING_NONE
- #define **CACA_DITHER_ORDERED** CACA_DITHERING_ORDERED8
- #define **CACA_DITHER_RANDOM** CACA_DITHERING_RANDOM

### Enumerations

- enum **caca_color** {

  **CACA_COLOR_BLACK** = 0, **CACA_COLOR_BLUE** = 1, **CACA_COLOR_GREEN** = 2, **CACA_COLOR_CYAN** = 3,

  **CACA_COLOR_RED** = 4, **CACA_COLOR_MAGENTA** = 5, **CACA_COLOR_BROWN** = 6, **CACA_COLOR_LIGHTGRAY** = 7,

  **CACA_COLOR_DARKGRAY** = 8, **CACA_COLOR_LIGHTBLUE** = 9, **CACA_COLOR_LIGHTGREEN** = 10, **CACA_COLOR_LIGHTCYAN** = 11,

  **CACA_COLOR_LIGHTRED** = 12, **CACA_COLOR_LIGHTMAGENTA** = 13, **CACA_COLOR_YELLOW** = 14, **CACA_COLOR_WHITE** = 15 }
- enum **caca_feature** {

  **CACA_BACKGROUND** = 0x10, **CACA_BACKGROUND_BLACK** = 0x11, **CACA_BACKGROUND_SOLID** = 0x12, **CACA_ANTIALIASING** = 0x20,

  **CACA_ANTIALIASING_NONE** = 0x21, **CACA_ANTIALIASING_PREFILTER** = 0x22, **CACA_DITHERING** = 0x30, **CACA_DITHERING_NONE** = 0x31,

  **CACA_DITHERING_ORDERED2** = 0x32, **CACA_DITHERING_ORDERED4** = 0x33, **CACA_DITHERING_ORDERED8** = 0x34, **CACA_DITHERING_RANDOM** = 0x35,

  **CACA_UNKNOWN_FEATURE** = 0xffff }
- enum **caca_event** { **CACA_EVENT_NONE** = 0x00000000, **CACA_EVENT_KEY_PRESS** = 0x01000000, **CACA_EVENT_KEY_RELEASE** = 0x02000000, **CACA_EVENT_MOUSE_CLICK** = 0x04000000 }
- enum **caca_key** {

  **CACA_KEY_UP** = 273, **CACA_KEY_DOWN** = 274, **CACA_KEY_LEFT** = 275, **CACA_KEY_RIGHT** = 276,

  **CACA_KEY_F1** = 282, **CACA_KEY_F2** = 283, **CACA_KEY_F3** = 284, **CACA_KEY_F4** = 285,

  **CACA_KEY_F5** = 286, **CACA_KEY_F6** = 287, **CACA_KEY_F7** = 288, **CACA_KEY_F8** = 289,

  **CACA_KEY_F9** = 290, **CACA_KEY_F10** = 291, **CACA_KEY_F11** = 292, **CACA_KEY_F12** = 293,

  **CACA_KEY_F13** = 294, **CACA_KEY_F14** = 295, **CACA_KEY_F15** = 296 }

### Functions

- const char * **caca_get_color_name** (enum **caca_color**)

  *Translate a colour value into its name.*

- const char * **caca_get_feature_name** (enum **caca_feature**)

  *Translate a feature value into its name.*

- int **caca_init** (void)

  *Initialise libcaca.*

- void **caca_set_delay** (unsigned int)

  *Set the refresh delay.*

- enum **caca_feature caca_get_feature** (enum **caca_feature**)
  *Get the current value of a feature.*
- void **caca_set_feature** (enum **caca_feature**)
  *Set a feature.*
- unsigned int **caca_get_rendertime** (void)
  *Get the average rendering time.*
- unsigned int **caca_get_width** (void)
  *Get the screen width.*
- unsigned int **caca_get_height** (void)
  *Get the screen height.*
- void **caca_refresh** (void)
  *Flush pending changes and redraw the screen.*
- void **caca_end** (void)
  *Uninitialise libcaca.*
- unsigned int **caca_get_event** (void)
  *Get the next mouse or keyboard input event.*
- void **caca_set_color** (enum **caca_color**, enum **caca_color**)
  *Set the default colour pair.*
- enum **caca_color caca_get_fg_color** (void)
  *Get the current foreground colour.*
- enum **caca_color caca_get_bg_color** (void)
  *Get the current background colour.*
- void **caca_putchar** (int, int, char)
  *Print a character at given coordinates.*
- void **caca_putstr** (int, int, const char *)
  *Print a string at given coordinates.*
- void **caca_printf** (int, int, const char *,...)
  *Format a string at given coordinates.*
- void **caca_clear** (void)
  *Clear the screen.*
- void **caca_draw_line** (int, int, int, int, char)
  *Draw a line on the screen using the given character.*
- void **caca_draw_polyline** (const int x[], const int y[], int, char)

*Draw a polyline on the screen using the given character and coordinate arrays. The first and last points are not connected, so in order to draw a polygon you need to specify the starting point at the end of the list as well.*

- void **caca_draw_thin_line** (int, int, int, int)

  *Draw a thin line on the screen, using ASCII art.*

- void **caca_draw_thin_polyline** (const int x[], const int y[], int)

  *Draw a thin polyline on the screen using the given coordinate arrays and with ASCII art. The first and last points are not connected, so in order to draw a polygon you need to specify the starting point at the end of the list as well.*

- void **caca_draw_circle** (int, int, int, char)

  *Draw a circle on the screen using the given character.*

- void **caca_draw_ellipse** (int, int, int, int, char)

  *Draw an ellipse on the screen using the given character.*

- void **caca_draw_thin_ellipse** (int, int, int, int)

  *Draw a thin ellipse on the screen.*

- void **caca_fill_ellipse** (int, int, int, int, char)

  *Fill an ellipse on the screen using the given character.*

- void **caca_draw_box** (int, int, int, int, char)

  *Draw a box on the screen using the given character.*

- void **caca_draw_thin_box** (int, int, int, int)

  *Draw a thin box on the screen.*

- void **caca_fill_box** (int, int, int, int, char)

  *Fill a box on the screen using the given character.*

- void **caca_draw_triangle** (int, int, int, int, int, int, char)

  *Draw a triangle on the screen using the given character.*

- void **caca_draw_thin_triangle** (int, int, int, int, int, int)

  *Draw a thin triangle on the screen.*

- void **caca_fill_triangle** (int, int, int, int, int, int, char)

  *Fill a triangle on the screen using the given character.*

- int **caca_rand** (int, int)

  *Generate a random integer within a range.*

- unsigned int **caca_sqrt** (unsigned int)

  *Approximate a square root, using Newton's method to avoid costly floating point calculations.*

- caca_sprite * **caca_load_sprite** (const char *)

  *Allocate a sprite loaded from a file.*

- int **caca_get_sprite_frames** (const struct caca_sprite *)
  *Return the number of frames in a sprite.*
- int **caca_get_sprite_width** (const struct caca_sprite *, int)
  *Return the width of a sprite.*
- int **caca_get_sprite_height** (const struct caca_sprite *, int)
  *Return the height of a sprite.*
- int **caca_get_sprite_dx** (const struct caca_sprite *, int)
  *Return the X coordinate of a sprite's handle.*
- int **caca_get_sprite_dy** (const struct caca_sprite *, int)
  *Return the Y coordinate of a sprite's handle.*
- void **caca_draw_sprite** (int, int, const struct caca_sprite *, int)
  *Draw a sprite's specific frame at the given coordinates. If the frame does not exist, nothing is displayed.*
- void **caca_free_sprite** (struct caca_sprite *)
  *Free the memory associated with a sprite.*
- caca_bitmap * **caca_create_bitmap** (unsigned int, unsigned int, unsigned int, unsigned int, unsigned int, unsigned int, unsigned int, unsigned int)
  *Create an internal bitmap object.*
- void **caca_set_bitmap_palette** (struct caca_bitmap *, unsigned int r[], unsigned int g[], unsigned int b[], unsigned int a[])
  *Set the palette of an 8bpp bitmap object.*
- void **caca_draw_bitmap** (int, int, int, int, const struct caca_bitmap *, void *)
  *Draw a bitmap on the screen.*
- void **caca_free_bitmap** (struct caca_bitmap *)
  *Free the memory associated with a bitmap.*

### 3.4.1 Detailed Description

The *libcaca* public header.

**Version:**
$Id: **caca.h**(p. 8) 154 2003-12-18 00:11:52Z sam $

**Author:**
Sam Hocevar <sam@zoy.org>

This header contains the public types and functions that applications using *libcaca* may use.

### 3.4.2 Enumeration Type Documentation

#### 3.4.2.1 enum caca_color

The colour definitions to be used with **caca_set_color()**(p. 23).

#### 3.4.2.2 enum caca_event

The event types returned by **caca_get_event()**(p. 19).

#### 3.4.2.3 enum caca_feature

The internal libcaca features.

#### 3.4.2.4 enum caca_key

The special key values returned by **caca_get_event()**(p. 19).

### 3.4.3 Function Documentation

#### 3.4.3.1 void caca_clear (void)

Clear the screen.

**Returns:**
void

#### 3.4.3.2 struct caca_bitmap* caca_create_bitmap (unsigned int *bpp*, unsigned int *w*, unsigned int *h*, unsigned int *pitch*, unsigned int *rmask*, unsigned int *gmask*, unsigned int *bmask*, unsigned int *amask*)

Create an internal bitmap object.

**Parameters:**
***bitmap*** The bitmap depth in bits per pixel.
***w*** The bitmap width in pixels.
***h*** The bitmap height in pixels.
***pitch*** The bitmap pitch in bytes.
***rmask*** The bitmask for red values.
***gmask*** The bitmask for green values.
***bmask*** The bitmask for blue values.
***amask*** The bitmask for alpha values.

**Returns:**
A bitmap object or NULL upon error.

#### 3.4.3.3 void caca_draw_bitmap (int *x1*, int *y1*, int *x2*, int *y2*, const struct caca_bitmap * *bitmap*, void * *pixels*)

Draw a bitmap on the screen.

**Parameters:**

***x1*** X coordinate of the upper-left corner of the drawing area.

***y1*** Y coordinate of the upper-left corner of the drawing area.

***x2*** X coordinate of the lower-right corner of the drawing area.

***y2*** Y coordinate of the lower-right corner of the drawing area.

***bitmap*** The bitmap object to be drawn.

***pixels*** A pointer to the bitmap's pixels.

**Returns:**

void

#### 3.4.3.4 void caca_draw_box (int *x1*, int *y1*, int *x2*, int *y2*, char *c*)

Draw a box on the screen using the given character.

**Parameters:**

***x1*** X coordinate of the upper-left corner of the box.

***y1*** Y coordinate of the upper-left corner of the box.

***x2*** X coordinate of the lower-right corner of the box.

***y2*** Y coordinate of the lower-right corner of the box.

***c*** Character to draw the box outline with.

**Returns:**

void

#### 3.4.3.5 void caca_draw_circle (int *x*, int *y*, int *r*, char *c*)

Draw a circle on the screen using the given character.

**Parameters:**

***x*** Center X coordinate.

***y*** Center Y coordinate.

***r*** Circle radius.

***c*** Character to draw the circle outline with.

**Returns:**

void

#### 3.4.3.6 void caca_draw_ellipse (int *xo*, int *yo*, int *a*, int *b*, char *c*)

Draw an ellipse on the screen using the given character.

**Parameters:**

***xo*** Center X coordinate.

***yo*** Center Y coordinate.

***a*** Ellipse X radius.

***b*** Ellipse Y radius.

***c*** Character to draw the ellipse outline with.

**Returns:**

void

#### 3.4.3.7 void caca_draw_line (int *x1*, int *y1*, int *x2*, int *y2*, char *c*)

Draw a line on the screen using the given character.

**Parameters:**

***x1*** X coordinate of the first point.

***y1*** Y coordinate of the first point.

***x2*** X coordinate of the second point.

***y2*** Y coordinate of the second point.

***c*** Character to draw the line with.

**Returns:**

void

#### 3.4.3.8 void caca_draw_polyline (const int *x*[], const int *y*[], int *n*, char *c*)

Draw a polyline on the screen using the given character and coordinate arrays. The first and last points are not connected, so in order to draw a polygon you need to specify the starting point at the end of the list as well.

**Parameters:**

***x*** Array of X coordinates. Must have `n` + 1 elements.

***y*** Array of Y coordinates. Must have `n` + 1 elements.

***n*** Number of lines to draw.

***c*** Character to draw the lines with.

**Returns:**

void

#### 3.4.3.9 void caca_draw_sprite (int *x*, int *y*, const struct caca_sprite * *sprite*, int *f*)

Draw a sprite's specific frame at the given coordinates. If the frame does not exist, nothing is displayed.

**Parameters:**

***x*** The X coordinate.

***y*** The Y coordinate.

***sprite*** The sprite.

***f*** The frame index.

**Returns:**

void

#### 3.4.3.10 void caca_draw_thin_box (int *x1*, int *y1*, int *x2*, int *y2*)

Draw a thin box on the screen.

**Parameters:**

***x1*** X coordinate of the upper-left corner of the box.

***y1*** Y coordinate of the upper-left corner of the box.

***x2*** X coordinate of the lower-right corner of the box.

***y2*** Y coordinate of the lower-right corner of the box.

**Returns:**

void

#### 3.4.3.11 void caca_draw_thin_ellipse (int *xo*, int *yo*, int *a*, int *b*)

Draw a thin ellipse on the screen.

**Parameters:**

***xo*** Center X coordinate.

***yo*** Center Y coordinate.

***a*** Ellipse X radius.

***b*** Ellipse Y radius.

**Returns:**

void

#### 3.4.3.12 void caca_draw_thin_line (int *x1*, int *y1*, int *x2*, int *y2*)

Draw a thin line on the screen, using ASCII art.

**Parameters:**

***x1*** X coordinate of the first point.

***y1*** Y coordinate of the first point.

***x2*** X coordinate of the second point.

***y2*** Y coordinate of the second point.

**Returns:**

void

#### 3.4.3.13 void caca_draw_thin_polyline (const int *x*[], const int *y*[], int *n*)

Draw a thin polyline on the screen using the given coordinate arrays and with ASCII art. The first and last points are not connected, so in order to draw a polygon you need to specify the starting point at the end of the list as well.

**Parameters:**

***x*** Array of X coordinates. Must have n + 1 elements.

***y*** Array of Y coordinates. Must have n + 1 elements.

***n*** Number of lines to draw.

**Returns:**

void

#### 3.4.3.14 void caca_draw_thin_triangle (int *x1*, int *y1*, int *x2*, int *y2*, int *x3*, int *y3*)

Draw a thin triangle on the screen.

**Parameters:**

***x1*** X coordinate of the first point.

***y1*** Y coordinate of the first point.

***x2*** X coordinate of the second point.

***y2*** Y coordinate of the second point.

***x3*** X coordinate of the third point.

***y3*** Y coordinate of the third point.

**Returns:**

void

#### 3.4.3.15 void caca_draw_triangle (int *x1*, int *y1*, int *x2*, int *y2*, int *x3*, int *y3*, char *c*)

Draw a triangle on the screen using the given character.

**Parameters:**

***x1*** X coordinate of the first point.

***y1*** Y coordinate of the first point.

***x2*** X coordinate of the second point.

***y2*** Y coordinate of the second point.

***x3*** X coordinate of the third point.

***y3*** Y coordinate of the third point.

***c*** Character to draw the triangle outline with.

**Returns:**

void

#### 3.4.3.16 void caca_end (void)

Uninitialise libcaca.

**Returns:**
void

#### 3.4.3.17 void caca_fill_box (int *x1*, int *y1*, int *x2*, int *y2*, char *c*)

Fill a box on the screen using the given character.

**Parameters:**
***x1*** X coordinate of the upper-left corner of the box.
***y1*** Y coordinate of the upper-left corner of the box.
***x2*** X coordinate of the lower-right corner of the box.
***y2*** Y coordinate of the lower-right corner of the box.
***c*** Character to fill the box with.

**Returns:**
void

#### 3.4.3.18 void caca_fill_ellipse (int *xo*, int *yo*, int *a*, int *b*, char *c*)

Fill an ellipse on the screen using the given character.

**Parameters:**
***xo*** Center X coordinate.
***yo*** Center Y coordinate.
***a*** Ellipse X radius.
***b*** Ellipse Y radius.
***c*** Character to fill the ellipse with.

**Returns:**
void

#### 3.4.3.19 void caca_fill_triangle (int *x1*, int *y1*, int *x2*, int *y2*, int *x3*, int *y3*, char *c*)

Fill a triangle on the screen using the given character.

**Parameters:**
***x1*** X coordinate of the first point.
***y1*** Y coordinate of the first point.
***x2*** X coordinate of the second point.
***y2*** Y coordinate of the second point.
***x3*** X coordinate of the third point.
***y3*** Y coordinate of the third point.
***c*** Character to fill the triangle with.

**Returns:**
void

#### 3.4.3.20 void caca_free_bitmap (struct caca_bitmap * *bitmap*)

Free the memory associated with a bitmap.

**Parameters:**
***bitmap*** The bitmap object to be freed.

**Returns:**
void

#### 3.4.3.21 void caca_free_sprite (struct caca_sprite * *sprite*)

Free the memory associated with a sprite.

**Parameters:**
***sprite*** The sprite to be freed.

**Returns:**
void

#### 3.4.3.22 enum caca_color caca_get_bg_color (void)

Get the current background colour.

**Returns:**
The current background colour.

#### 3.4.3.23 const char* caca_get_color_name (enum caca_color *color*)

Translate a colour value into its name.

**Parameters:**
***color*** The colour value.

**Returns:**
A static string containing the colour's name.

#### 3.4.3.24 unsigned int caca_get_event (void)

Get the next mouse or keyboard input event.

**Returns:**
The next event in the queue, or 0 if no event is pending.

#### 3.4.3.25 enum caca_feature caca_get_feature (enum caca_feature *feature*)

Get the current value of a feature.

**Parameters:**
***feature*** The requested feature.

**Returns:**
The current value of the feature.

#### 3.4.3.26 const char* caca_get_feature_name (enum caca_feature *feature*)

Translate a feature value into its name.

**Parameters:**
***feature*** The feature value.

**Returns:**
A static string containing the feature's name.

#### 3.4.3.27 enum caca_color caca_get_fg_color (void)

Get the current foreground colour.

**Returns:**
The current foreground colour.

#### 3.4.3.28 unsigned int caca_get_height (void)

Get the screen height.

**Returns:**
The screen height, in character cells.

#### 3.4.3.29 unsigned int caca_get_rendertime (void)

Get the average rendering time.

**Returns:**
The render time in microseconds.

#### 3.4.3.30 int caca_get_sprite_dx (const struct caca_sprite * *sprite*, int *f*)

Return the X coordinate of a sprite's handle.

**Parameters:**
***sprite*** The sprite.
***f*** The frame index.

**Returns:**
The X coordinate of the given frame's handle.

#### 3.4.3.31 int caca_get_sprite_dy (const struct caca_sprite * *sprite*, int *f*)

Return the Y coordinate of a sprite's handle.

**Parameters:**
***sprite*** The sprite.
***f*** The frame index.

**Returns:**
The Y coordinate of the given frame's handle.

#### 3.4.3.32 int caca_get_sprite_frames (const struct caca_sprite * *sprite*)

Return the number of frames in a sprite.

**Parameters:**
***sprite*** The sprite.

**Returns:**
The number of frames.

#### 3.4.3.33 int caca_get_sprite_height (const struct caca_sprite * *sprite*, int *f*)

Return the height of a sprite.

**Parameters:**
***sprite*** The sprite.
***f*** The frame index.

**Returns:**
The height of the given frame of the sprite.

#### 3.4.3.34 int caca_get_sprite_width (const struct caca_sprite * *sprite*, int *f*)

Return the width of a sprite.

**Parameters:**
***sprite*** The sprite.
***f*** The frame index.

**Returns:**
The width of the given frame of the sprite.

#### 3.4.3.35 unsigned int caca_get_width (void)

Get the screen width.

**Returns:**
The screen width, in character cells.

#### 3.4.3.36 int caca_init (void)

Initialise libcaca.

**Returns:**
0 upon success, a non-zero value if an error occurs.

#### 3.4.3.37 struct caca_sprite* caca_load_sprite (const char * *file*)

Allocate a sprite loaded from a file.

**Parameters:**
*file* The filename.

**Returns:**
The sprite, or NULL if an error occured.

#### 3.4.3.38 void caca_printf (int *x*, int *y*, const char * *format*, ...)

Format a string at given coordinates.

**Parameters:**
*x* The X coordinate of the string.
*y* The Y coordinate of the string.
*format* The format string to print.
... Arguments to the format string.

**Returns:**
void

#### 3.4.3.39 void caca_putchar (int *x*, int *y*, char *c*)

Print a character at given coordinates.

**Parameters:**
*x* The X coordinate of the character.
*y* The Y coordinate of the character.
*c* The character to print.

**Returns:**
void

#### 3.4.3.40 void caca_putstr (int *x*, int *y*, const char * *s*)

Print a string at given coordinates.

**Parameters:**
*x* The X coordinate of the string.
*y* The Y coordinate of the string.
*s* The string to print.

**Returns:**
void

#### 3.4.3.41 int caca_rand (int *min*, int *max*)

Generate a random integer within a range.

**Parameters:**

***min*** The lower bound of the integer range.

***max*** The upper bound of the integer range.

**Returns:**

A random integer comprised between `min` and `max`, inclusive.

#### 3.4.3.42 void caca_refresh (void)

Flush pending changes and redraw the screen.

**Returns:**

void

#### 3.4.3.43 void caca_set_bitmap_palette (struct caca_bitmap * *bitmap*, unsigned int *red*[ ], unsigned int *green*[ ], unsigned int *blue*[ ], unsigned int *alpha*[ ])

Set the palette of an 8bpp bitmap object.

**Parameters:**

***bpp*** The bitmap object.

***red*** An array of 256 red values.

***green*** An array of 256 green values.

***blue*** An array of 256 blue values.

***alpha*** An array of 256 alpha values.

**Returns:**

void

#### 3.4.3.44 void caca_set_color (enum caca_color *fgcolor*, enum caca_color *bgcolor*)

Set the default colour pair.

**Parameters:**

***fgcolor*** The desired foreground colour.

***bgcolor*** The desired background colour.

**Returns:**

void

#### 3.4.3.45 void caca_set_delay (unsigned int *usec*)

Set the refresh delay.

**Parameters:**
***usec*** The refresh delay in microseconds.

**Returns:**
void

#### 3.4.3.46 void caca_set_feature (enum caca_feature *feature*)

Set a feature.

**Parameters:**
***feature*** The wanted feature.

**Returns:**
void

#### 3.4.3.47 unsigned int caca_sqrt (unsigned int *a*)

Approximate a square root, using Newton's method to avoid costly floating point calculations.

**Parameters:**
***a*** A positive integer.

**Returns:**
The approximate square root of `a`.

## 3.5 caca_internals.h File Reference

The *libcaca* private header.

### Functions

- int **_caca_init_graphics** (void)
- int **_caca_end_graphics** (void)

### Variables

- unsigned int **_caca_width**
- unsigned int **_caca_height**
- enum **caca_feature** **_caca_background**
- enum **caca_feature** **_caca_dithering**
- enum **caca_feature** **_caca_antialiasing**

### 3.5.1 Detailed Description

The *libcaca* private header.

**Version:**
$Id: **caca_internals.h**(p. 24) 148 2003-12-15 10:38:03Z sam $

**Author:**
Sam Hocevar <sam@zoy.org>

This header contains the private types and functions used by *libcaca*.

## 3.6 conic.c File Reference

Ellipse and circle drawing functions.

### Typedefs

- typedef unsigned char **uint8_t**

### Functions

- void **caca_draw_circle** (int x, int y, int r, char c)
  *Draw a circle on the screen using the given character.*
- void **caca_fill_ellipse** (int xo, int yo, int a, int b, char c)
  *Fill an ellipse on the screen using the given character.*
- void **caca_draw_ellipse** (int xo, int yo, int a, int b, char c)
  *Draw an ellipse on the screen using the given character.*
- void **caca_draw_thin_ellipse** (int xo, int yo, int a, int b)
  *Draw a thin ellipse on the screen.*

### 3.6.1 Detailed Description

Ellipse and circle drawing functions.

**Version:**
$Id: **conic.c**(p. 25) 154 2003-12-18 00:11:52Z sam $

**Author:**
Sam Hocevar <sam@zoy.org>

This file contains ellipse and circle drawing functions, both filled and outline.

### 3.6.2 Function Documentation

#### 3.6.2.1 void caca_draw_circle (int *x*, int *y*, int *r*, char *c*)

Draw a circle on the screen using the given character.

**Parameters:**

***x*** Center X coordinate.

***y*** Center Y coordinate.

***r*** Circle radius.

***c*** Character to draw the circle outline with.

**Returns:**

void

#### 3.6.2.2 void caca_draw_ellipse (int *xo*, int *yo*, int *a*, int *b*, char *c*)

Draw an ellipse on the screen using the given character.

**Parameters:**

***xo*** Center X coordinate.

***yo*** Center Y coordinate.

***a*** Ellipse X radius.

***b*** Ellipse Y radius.

***c*** Character to draw the ellipse outline with.

**Returns:**

void

#### 3.6.2.3 void caca_draw_thin_ellipse (int *xo*, int *yo*, int *a*, int *b*)

Draw a thin ellipse on the screen.

**Parameters:**

***xo*** Center X coordinate.

***yo*** Center Y coordinate.

***a*** Ellipse X radius.

***b*** Ellipse Y radius.

**Returns:**

void

#### 3.6.2.4 void caca_fill_ellipse (int *xo*, int *yo*, int *a*, int *b*, char *c*)

Fill an ellipse on the screen using the given character.

**Parameters:**

***xo*** Center X coordinate.

***yo*** Center Y coordinate.

***a*** Ellipse X radius.

***b*** Ellipse Y radius.

***c*** Character to fill the ellipse with.

**Returns:**
void

## 3.7 graphics.c File Reference

Character drawing functions.

### Defines

- #define **IDLE_USEC** 10000

### Functions

- void **caca_set_color** (enum **caca_color** fgcolor, enum **caca_color** bgcolor)
  *Set the default colour pair.*
- enum **caca_color caca_get_fg_color** (void)
  *Get the current foreground colour.*
- enum **caca_color caca_get_bg_color** (void)
  *Get the current background colour.*
- void **caca_putchar** (int x, int y, char c)
  *Print a character at given coordinates.*
- void **caca_putstr** (int x, int y, const char *s)
  *Print a string at given coordinates.*
- void **caca_printf** (int x, int y, const char *format,...)
  *Format a string at given coordinates.*
- void **caca_clear** (void)
  *Clear the screen.*
- int **_caca_init_graphics** (void)
- int **_caca_end_graphics** (void)
- void **caca_set_delay** (unsigned int usec)
  *Set the refresh delay.*
- unsigned int **caca_get_rendertime** (void)
  *Get the average rendering time.*
- void **caca_refresh** (void)
  *Flush pending changes and redraw the screen.*

**Variables**

- unsigned int **_caca_width**
- unsigned int **_caca_height**

### 3.7.1 Detailed Description

Character drawing functions.

**Version:**
$Id: **graphics.c**(p. 27) 154 2003-12-18 00:11:52Z sam $

**Author:**
Sam Hocevar <sam@zoy.org>

This file contains character and string drawing functions.

### 3.7.2 Function Documentation

#### 3.7.2.1 void caca_clear (void)

Clear the screen.

**Returns:**
void

#### 3.7.2.2 enum caca_color caca_get_bg_color (void)

Get the current background colour.

**Returns:**
The current background colour.

#### 3.7.2.3 enum caca_color caca_get_fg_color (void)

Get the current foreground colour.

**Returns:**
The current foreground colour.

#### 3.7.2.4 unsigned int caca_get_rendertime (void)

Get the average rendering time.

**Returns:**
The render time in microseconds.

#### 3.7.2.5 void caca_printf (int *x*, int *y*, const char * *format*, ...)

Format a string at given coordinates.

**Parameters:**

***x*** The X coordinate of the string.

***y*** The Y coordinate of the string.

***format*** The format string to print.

*...* Arguments to the format string.

**Returns:**

void

#### 3.7.2.6 void caca_putchar (int *x*, int *y*, char *c*)

Print a character at given coordinates.

**Parameters:**

***x*** The X coordinate of the character.

***y*** The Y coordinate of the character.

***c*** The character to print.

**Returns:**

void

#### 3.7.2.7 void caca_putstr (int *x*, int *y*, const char * *s*)

Print a string at given coordinates.

**Parameters:**

***x*** The X coordinate of the string.

***y*** The Y coordinate of the string.

***s*** The string to print.

**Returns:**

void

#### 3.7.2.8 void caca_refresh (void)

Flush pending changes and redraw the screen.

**Returns:**

void

#### 3.7.2.9 void caca_set_color (enum caca_color *fgcolor*, enum caca_color *bgcolor*)

Set the default colour pair.

**Parameters:**
***fgcolor*** The desired foreground colour.

***bgcolor*** The desired background colour.

**Returns:**
void

#### 3.7.2.10 void caca_set_delay (unsigned int *usec*)

Set the refresh delay.

**Parameters:**
***usec*** The refresh delay in microseconds.

**Returns:**
void

## 3.8 io.c File Reference

Event handling functions.

### Defines

- #define **KEY_BUFLEN** 10

### Functions

- unsigned int **caca_get_event** (void)
  *Get the next mouse or keyboard input event.*

### 3.8.1 Detailed Description

Event handling functions.

**Version:**
$Id: **io.c**(p. 30) 154 2003-12-18 00:11:52Z sam $

**Author:**
Sam Hocevar <sam@zoy.org>

This file contains event handling functions for keyboard and mouse input.

### 3.8.2 Function Documentation

#### 3.8.2.1 unsigned int caca_get_event (void)

Get the next mouse or keyboard input event.

**Returns:**
The next event in the queue, or 0 if no event is pending.

## 3.9 line.c File Reference

Line drawing functions.

### Data Structures

- struct **line**

### Typedefs

- typedef unsigned char **uint8_t**

### Functions

- void **caca_draw_line** (int x1, int y1, int x2, int y2, char c)

  *Draw a line on the screen using the given character.*

- void **caca_draw_polyline** (const int x[], const int y[], int n, char c)

  *Draw a polyline on the screen using the given character and coordinate arrays. The first and last points are not connected, so in order to draw a polygon you need to specify the starting point at the end of the list as well.*

- void **caca_draw_thin_line** (int x1, int y1, int x2, int y2)

  *Draw a thin line on the screen, using ASCII art.*

- void **caca_draw_thin_polyline** (const int x[], const int y[], int n)

  *Draw a thin polyline on the screen using the given coordinate arrays and with ASCII art. The first and last points are not connected, so in order to draw a polygon you need to specify the starting point at the end of the list as well.*

### 3.9.1 Detailed Description

Line drawing functions.

**Version:**
$Id: **line.c**(p. 31) 154 2003-12-18 00:11:52Z sam $

**Author:**
Sam Hocevar <sam@zoy.org>

This file contains line and polyline drawing functions, with both thin and thick styles.

### 3.9.2 Function Documentation

#### 3.9.2.1 void caca_draw_line (int *x1*, int *y1*, int *x2*, int *y2*, char *c*)

Draw a line on the screen using the given character.

**Parameters:**

***x1*** X coordinate of the first point.

***y1*** Y coordinate of the first point.

***x2*** X coordinate of the second point.

***y2*** Y coordinate of the second point.

***c*** Character to draw the line with.

**Returns:**

void

#### 3.9.2.2 void caca_draw_polyline (const int *x*[], const int *y*[], int *n*, char *c*)

Draw a polyline on the screen using the given character and coordinate arrays. The first and last points are not connected, so in order to draw a polygon you need to specify the starting point at the end of the list as well.

**Parameters:**

***x*** Array of X coordinates. Must have `n` + 1 elements.

***y*** Array of Y coordinates. Must have `n` + 1 elements.

***n*** Number of lines to draw.

***c*** Character to draw the lines with.

**Returns:**

void

#### 3.9.2.3 void caca_draw_thin_line (int *x1*, int *y1*, int *x2*, int *y2*)

Draw a thin line on the screen, using ASCII art.

**Parameters:**

***x1*** X coordinate of the first point.

***y1*** Y coordinate of the first point.

***x2*** X coordinate of the second point.

***y2*** Y coordinate of the second point.

**Returns:**

void

#### 3.9.2.4 void caca_draw_thin_polyline (const int *x*[], const int *y*[], int *n*)

Draw a thin polyline on the screen using the given coordinate arrays and with ASCII art. The first and last points are not connected, so in order to draw a polygon you need to specify the starting point at the end of the list as well.

**Parameters:**

***x*** Array of X coordinates. Must have `n` + 1 elements.

***y*** Array of Y coordinates. Must have `n` + 1 elements.

***n*** Number of lines to draw.

**Returns:**

void

## 3.10 math.c File Reference

Math functions.

### Functions

- int **caca_rand** (int min, int max)
  *Generate a random integer within a range.*
- unsigned int **caca_sqrt** (unsigned int a)
  *Approximate a square root, using Newton's method to avoid costly floating point calculations.*

### 3.10.1 Detailed Description

Math functions.

**Version:**

$Id: **math.c**(p. 33) 154 2003-12-18 00:11:52Z sam $

**Author:**

Sam Hocevar <`sam@zoy.org`>

This file contains simple mathematical routines.

### 3.10.2 Function Documentation

#### 3.10.2.1 int caca_rand (int *min*, int *max*)

Generate a random integer within a range.

**Parameters:**

***min*** The lower bound of the integer range.

***max*** The upper bound of the integer range.

**Returns:**

A random integer comprised between `min` and `max`, inclusive.

#### 3.10.2.2 unsigned int caca_sqrt (unsigned int *a*)

Approximate a square root, using Newton's method to avoid costly floating point calculations.

**Parameters:**
*a* A positive integer.

**Returns:**
The approximate square root of `a`.

## 3.11 sprite.c File Reference

Sprite loading and blitting.

### Data Structures

- struct **caca_frame**
- struct **caca_sprite**

### Functions

- caca_sprite * **caca_load_sprite** (const char *file)
  *Allocate a sprite loaded from a file.*
- int **caca_get_sprite_frames** (const struct caca_sprite *sprite)
  *Return the number of frames in a sprite.*
- int **caca_get_sprite_width** (const struct caca_sprite *sprite, int f)
  *Return the width of a sprite.*
- int **caca_get_sprite_height** (const struct caca_sprite *sprite, int f)
  *Return the height of a sprite.*
- int **caca_get_sprite_dx** (const struct caca_sprite *sprite, int f)
  *Return the X coordinate of a sprite's handle.*
- int **caca_get_sprite_dy** (const struct caca_sprite *sprite, int f)
  *Return the Y coordinate of a sprite's handle.*
- void **caca_draw_sprite** (int x, int y, const struct caca_sprite *sprite, int f)
  *Draw a sprite's specific frame at the given coordinates. If the frame does not exist, nothing is displayed.*
- void **caca_free_sprite** (struct caca_sprite *sprite)
  *Free the memory associated with a sprite.*

### 3.11.1 Detailed Description

Sprite loading and blitting.

**Version:**
$Id: **sprite.c**(p. 34) 154 2003-12-18 00:11:52Z sam $

**Author:**
Sam Hocevar <sam@zoy.org>

This file contains a small framework for sprite loading and blitting.

### 3.11.2 Function Documentation

#### 3.11.2.1 void caca_draw_sprite (int *x*, int *y*, const struct caca_sprite * *sprite*, int *f*)

Draw a sprite's specific frame at the given coordinates. If the frame does not exist, nothing is displayed.

**Parameters:**
***x*** The X coordinate.
***y*** The Y coordinate.
***sprite*** The sprite.
***f*** The frame index.

**Returns:**
void

#### 3.11.2.2 void caca_free_sprite (struct caca_sprite * *sprite*)

Free the memory associated with a sprite.

**Parameters:**
***sprite*** The sprite to be freed.

**Returns:**
void

#### 3.11.2.3 int caca_get_sprite_dx (const struct caca_sprite * *sprite*, int *f*)

Return the X coordinate of a sprite's handle.

**Parameters:**
***sprite*** The sprite.
***f*** The frame index.

**Returns:**
The X coordinate of the given frame's handle.

#### 3.11.2.4 int caca_get_sprite_dy (const struct caca_sprite * *sprite*, int *f*)

Return the Y coordinate of a sprite's handle.

**Parameters:**
***sprite*** The sprite.
***f*** The frame index.

**Returns:**
The Y coordinate of the given frame's handle.

#### 3.11.2.5 int caca_get_sprite_frames (const struct caca_sprite * *sprite*)

Return the number of frames in a sprite.

**Parameters:**
***sprite*** The sprite.

**Returns:**
The number of frames.

#### 3.11.2.6 int caca_get_sprite_height (const struct caca_sprite * *sprite*, int *f*)

Return the height of a sprite.

**Parameters:**
***sprite*** The sprite.
***f*** The frame index.

**Returns:**
The height of the given frame of the sprite.

#### 3.11.2.7 int caca_get_sprite_width (const struct caca_sprite * *sprite*, int *f*)

Return the width of a sprite.

**Parameters:**
***sprite*** The sprite.
***f*** The frame index.

**Returns:**
The width of the given frame of the sprite.

#### 3.11.2.8 struct caca_sprite* caca_load_sprite (const char * *file*)

Allocate a sprite loaded from a file.

**Parameters:**
***file*** The filename.

**Returns:**
The sprite, or NULL if an error occured.

## 3.12 triangle.c File Reference

Triangle drawing functions.

### Functions

- void **caca_draw_triangle** (int x1, int y1, int x2, int y2, int x3, int y3, char c)
  *Draw a triangle on the screen using the given character.*
- void **caca_draw_thin_triangle** (int x1, int y1, int x2, int y2, int x3, int y3)
  *Draw a thin triangle on the screen.*
- void **caca_fill_triangle** (int x1, int y1, int x2, int y2, int x3, int y3, char c)
  *Fill a triangle on the screen using the given character.*

### 3.12.1 Detailed Description

Triangle drawing functions.

**Version:**
$Id: **triangle.c**(p. 37) 154 2003-12-18 00:11:52Z sam $

**Author:**
Sam Hocevar <sam@zoy.org>

This file contains triangle drawing functions, both filled and outline.

### 3.12.2 Function Documentation

#### 3.12.2.1 void caca_draw_thin_triangle (int *x1*, int *y1*, int *x2*, int *y2*, int *x3*, int *y3*)

Draw a thin triangle on the screen.

**Parameters:**
***x1*** X coordinate of the first point.
***y1*** Y coordinate of the first point.
***x2*** X coordinate of the second point.
***y2*** Y coordinate of the second point.
***x3*** X coordinate of the third point.
***y3*** Y coordinate of the third point.

**Returns:**
void

#### 3.12.2.2 void caca_draw_triangle (int *x1*, int *y1*, int *x2*, int *y2*, int *x3*, int *y3*, char *c*)

Draw a triangle on the screen using the given character.

**Parameters:**

***x1*** X coordinate of the first point.

***y1*** Y coordinate of the first point.

***x2*** X coordinate of the second point.

***y2*** Y coordinate of the second point.

***x3*** X coordinate of the third point.

***y3*** Y coordinate of the third point.

***c*** Character to draw the triangle outline with.

**Returns:**

void

#### 3.12.2.3 void caca_fill_triangle (int *x1*, int *y1*, int *x2*, int *y2*, int *x3*, int *y3*, char *c*)

Fill a triangle on the screen using the given character.

**Parameters:**

***x1*** X coordinate of the first point.

***y1*** Y coordinate of the first point.

***x2*** X coordinate of the second point.

***y2*** Y coordinate of the second point.

***x3*** X coordinate of the third point.

***y3*** Y coordinate of the third point.

***c*** Character to fill the triangle with.

**Returns:**

void

# Index