

M5Canvas Class

About M5Canvas

M5Canvas is a derived class of LGFX_Sprite, and its usage is basically the same as the Sprite of TFT_eSPI. We call it M5Canvas because it is appropriate to call it "Canvas" in the sense of drawing on memory, not "Sprite" as defined within M5GFX. (However, methods such as pushSprite remain for compatibility with TFT_eSprite.)

For more detailed API information not included below, refer to [here](#).

Benefits of Canvas

- It can be drawn in memory in advance and displayed on the panel at once for quick display.
- Useful for displaying small characters, etc.
- Transparent colors can be used to provide overlap.
- When drawing on the display, scaling, rotation, and anti-aliasing can be performed.

Constructor

Syntax:

1. M5Canvas()

- No arguments
- Note that if the argument is not specified, it must be specified at push or drawing time, or problems will occur.

2. M5Canvas(M5GFX&)

- Specify M5GFX.
- If M5Canvas is specified as an argument, it will draw to the M5GFX by default.

It is recommended to use the **M5Canvas(M5GFX&)** constructor to ensure that the correct drawing target is set.

LGFX_Sprite Class

- [createSprite](#)
- [fillSprite](#)
- [pushSprite](#)
- [deleteSprite](#)
- [createFromBmp](#)
- [createFromBmpFile](#)
- [setBitmapColor](#)
- [setColorDepth](#)
- [createPalette](#)
- [getPalette](#)
- [setPaletteGrayscale](#)
- [setPaletteColor](#)
- [getPaletteIndex](#)

- [deletePalette](#)
- [readPixelValue](#)
- [setPsrn](#)
- [setBuffer](#)
- [pushRotated](#)
- [pushRotatedWithAA](#)
- [pushRotateZoom](#)
- [pushRotateZoomWithAA](#)
- [pushAffine](#)
- [pushAffineWithAA](#)

Vital Note:

If a color palette is used, all functions related to color settings can only achieve the purpose of setting colors by using the color index number of the palette. The obtained color information is the color index number rather than the color value.

| createSprite

Syntax:

```
void* createSprite(int32_t w, int32_t h)
```

Description:

- create sprite

Parameters:

- w: sprite width
- h: sprite height

Return:

- null

| fillSprite

Syntax:

```
void fillSprite (const T& color)
```

Description:

- Fill the sprite

Parameters:

- color: fill color

Return:

- null

| pushSprite

Syntax1:

```
void pushSprite(int32_t x, int32_t y, const T& transp)
```

Syntax2:

```
void pushSprite(LovyanGFX* dst, int32_t x, int32_t y, const T& transp)
```

Syntax3:

```
void pushSprite(int32_t x, int32_t y)
```

Syntax4:

```
void pushSprite(LovyanGFX* dst, int32_t x, int32_t y)
```

Description:

- Push the canvas based on the specified coordinates, with the reference point located at the top left corner of the canvas

Parameters:

- dst: target LovyanGFX object
- x: canvas reference point x coordinate
- y: canvas reference point y coordinate
- transp: transparent color

Return:

- null

| deleteSprite

Syntax:

```
void deleteSprite(void)
```

Description:

- delete sprite

Parameters:

- null

Return:

- null

Example

1. Simple Usage

```
1  #include <M5GFX.h>
2  #include <M5Unified.h>
3
4  static int32_t Disw;
5  static int32_t Dish;
6
7  void setup() {
8      M5.begin();
9      Disw = M5.Lcd.width();
10     Dish = M5.Lcd.height();
11
12     M5.Lcd.fillScreen(TFT_WHITE);
13
14     M5Canvas canvas(&M5.Lcd);
15     canvas.createSprite(100, 100); //set sprite size
16     canvas.fillSprite(TFT_PINK); //fill sprite with color XXX
17     delay(1000);
18     canvas.fillSprite(TFT_BLACK); //fill sprite with color XXX
19     canvas.println("M5Canvas");
20     canvas.pushSprite(&M5.Lcd, Disw / 2 - 50, Dish / 2 - 50, TFT_PINK); //"&M5.Lcd" is not necessary
21     canvas.deleteSprite();
22     //In this example, PINK will not be displayed
23 }
24
25 void loop() {
26 }
```

The example program runs as follows:



2. Merge Multiple Canvases for Drawing


```
1  #include <M5GFX.h>
2  #include <M5Unified.h>
3
4  static int32_t Disw;
5  static int32_t Dish;
6
7  void setup() {
8      M5.begin();
9      Disw = M5.Lcd.width();
10     Dish = M5.Lcd.height();
11
12     M5.Lcd.fillScreen(TFT_WHITE);
13
14     M5Canvas canvas(&M5.Lcd);
15     M5Canvas can1(&M5.Lcd);
16     M5Canvas can2(&M5.Lcd);
17
18     canvas.createSprite(100, 100); //set sprite size
19     canvas.fillSprite(TFT_PINK); //fill sprite with color XXX
20     canvas.println("M5Canvas");
21
22     can1.createSprite(30, 30);
23     can1.fillSprite(TFT_BLUE);
24     can1.println("Can1");
25     can1.fillCircle(15, 15, 5, TFT_YELLOW);
26
27     can2.createSprite(30, 30);
28     can2.fillSprite(TFT_GREEN);
29     can2.println("Can2");
30     can2.fillTriangle(15, 10, 0, 30, 30, 30, TFT_BLUE);
31
32     canvas.pushSprite(Disw / 2 - 50, Dish / 2 - 50);
33     can1.pushSprite(Disw / 2 - 30, Dish / 2 - 30);
34     can2.pushSprite(Disw / 2, Dish / 2);
35
36     canvas.deleteSprite();
37     can1.deleteSprite();
38     can2.deleteSprite();
39 }
40
41 void loop() {
42 }
```

The example program runs as follows:



createFromBmp

Syntax1:

```
bool createFromBmp(const uint8_t *bmp_data, uint32_t bmp_len = ~0u)
```

Description:

- Create a canvas from BMP data

Parameters:

- bmp_data: BMP data pointer
- bmp_len: BMP data length (default is ~0u)

Syntax2:

```
bool createFromBmp(T &fs, const char *path)
```

Description:

- Create a canvas from a BMP file

Parameters:

- fs: file system object
- SPIFFS
- SD
- etc
- path: BMP file path

Return:

- bool
- true: creation successful
- false: creation failed

createFromBmpFile

Syntax1:

```
bool createFromBmpFile(const char *path)
```

Syntax2:

```
bool createFromBmpFile(T &fs, const char *path)
```

Description:

- Create a canvas from a BMP file

Parameters:

- path: BMP file path
- fs: file system object
- SPIFFS
- SD
- etc

Return:

- bool
- true: creation successful
- false: creation failed

This example program requires a microSD card formatted with FAT32, with two **PNG** images placed in the root directory and named **LGFX_Canavs_Test01.bmp** and **LGFX_Canavs_Test02.bmp**. The example program targets the **M5Fire** device, and the images have a resolution of **320*240**. You can directly download [Example Image 1](#) and [Example Image 2](#). If the image resolution is not **320*240**, the program will decide how to display it based on presets, which may lead to display anomalies.

Note:

The header file **SD.h** must be placed before **<M5Unified.h>** in the code below, otherwise compilation will fail.

Example

```
1  #include <Arduino.h>
2  #include <SD.h> // This header file must be before M5Unified.h
3  #include <SPI.h>
4  #include <M5GFX.h>
5  #include <M5Unified.h>
6
7  #define SD_SPI_CS_PIN 4
8  #define SD_SPI_SCK_PIN 18
9  #define SD_SPI_MOSI_PIN 23
10 #define SD_SPI_MISO_PIN 19
11
12 static int32_t Disw;
13 static int32_t Dish;
14
15 void setup() {
16     M5.begin();
17     Disw = M5.Lcd.width();
18     Dish = M5.Lcd.height();
19
20     M5.Lcd.fillScreen(TFT_BLACK);
21
22     M5.Display.setTextFont(&fonts::Orbitron_Light_24);
23     M5.Display.setTextSize(1);
24     // SD Card Initialization
25     SPI.begin(SD_SPI_SCK_PIN, SD_SPI_MISO_PIN, SD_SPI_MOSI_PIN, SD_SPI_CS_PIN);
26     if (!SD.begin(SD_SPI_CS_PIN, SPI, 25000000)) {
27         // Print a message if SD card initialization failed or if the SD card does not exist.
28         M5.Display.print("\n SD card not detected\n");
29         while (1); ;
30     } else {
31         M5.Display.print("\n SD card detected\n");
32     }
33     delay(1000);
34
35     M5.Display.print("\n SD card read test...\n");
36     if (SD.open("/LGFX_Canavs_Test01.bmp", FILE_READ, false)) {
37         M5.Display.print(" BMP file 01 detected\n");
38     } else {
39         M5.Display.print(" BMP file 01 not detected\n");
40     }
41     if (SD.open("/LGFX_Canavs_Test02.bmp", FILE_READ, false)) {
42         M5.Display.print(" BMP file 01 detected\n");
43     } else {
44         M5.Display.print(" BMP file 01 not detected\n");
45     }
46     delay(2000);
47 }
48
49 void loop() {
50     M5Canvas canvas(&M5.Lcd);
```

```

51     if (canvas.createFromBmp(SD, "/LGFX_Canavs_Test01.bmp")) {
52         canvas.pushSprite(0, 0);
53     } else {
54         M5.Display.print("\ncreateFromBmp failed\n");
55     }
56     delay(1000);
57     if (canvas.createFromBmpFile(SD, "/LGFX_Canavs_Test02.bmp")) {
58         canvas.pushSprite(0, 0);
59     } else {
60         M5.Display.print("\ncreateFromBmpFile failed\n");
61     }
62     delay(1000);
63     canvas.deleteSprite();
64 }

```

The function of the example is to alternate the display of two images. The first image is `LGFX_Canavs_Test01.bmp` , and the second image is `LGFX_Canavs_Test02.bmp` .

[Fire_M5GFX_bmp.mp4](#)

| setBitmapColor

Syntax:

```
void setBitmapColor(uint16_t fgcolor, uint16_t bgcolor)
```

Description:

- Set the bitmap color, only applicable to 1-bit.

Parameters:

- fgcolor: foreground color
- bgcolor: background color

Return:

- null

| setColorDepth

Syntax1:

```
void setColorDepth(uint8_t bpp)
```

Description:

- Set the color depth

Parameters:

- bpp: bits per pixel
- 1: 1 bit
- 2: 2 bits
- 4: 4 bits
- 8: 8 bits

- 16: 16 bits
- 24: 24 bits
- 32: 32 bits

Syntax2:

```
void* setColorDepth(color_depth_t depth)
```

Description:

- Set the color depth

Parameters:

- depth: color depth

Return:

- null

Note:

1. If a color palette is used, the `bpp` or `depth` can only be 1, 2, 4, or 8 bits.
2. Before calling `createPalette`, `setColorDepth` must be called first to set the color depth; otherwise, the creation of the palette will fail.
3. If the `setColorDepth` is used to set the color depth to 1 bit, then only `createPalette()` can be used to create the palette, and `createPalette(const uint16_t* colors, uint32_t count)` or `createPalette(const uint32_t* colors, uint32_t count)` cannot be used.

| createPalette

Syntax1:

```
bool createPalette(void)
```

Syntax2:

```
bool createPalette(const uint16_t* colors, uint32_t count)
```

Syntax3:

```
bool createPalette(const uint32_t* colors, uint32_t count)
```

Description:

- Create a color palette

Parameters:

- colors: color array pointer
- `uint16_t`: RGB565 16-bit color
- `uint32_t`: RGB888/ARGB8888 24-bit color/32-bit color
- count: number of colors

Return:

- bool
- true: creation successful
- false: creation failed

| getPalette

Function Prototype:

```
RGBColor* getPalette(void)
```

Description:

- Get the current palette information

Parameters:

- null

Return:

- RGBColor*: pointer to the bgr888_t struct, pointing to the first color in the current palette

| setPaletteGrayscale

Syntax:

```
void setPaletteGrayscale(void)
```

Description:

- Set the palette to grayscale

Parameters:

- null

Return:

- null

| setPaletteColor

Syntax1:

```
void setPaletteColor(uint32_t index, uint32_t color)
```

Description:

- Set the palette color

Parameters:

- index: color palette index
- color: color

Syntax2:

```
void setPaletteColor(size_t index, const bgr888_t& rgb)
```

Description:

- Set the palette color

Parameters:

- index: color palette index
- rgb: RGB color

Syntax3:

```
void setPaletteColor(size_t index, uint8_t r, uint8_t g, uint8_t b)
```

Description:

- Set the palette color

Parameters:

- index: color palette index
- r: red component
- g: green component
- b: blue component

Return:

- null

| getPaletteIndex

Syntax:

```
int32_t getPaletteIndex(const T& color)
```

Description:

- Obtain the palette index

Parameters:

- color: palette color

Return:

- res: color index
- -1: function execution failed

| deletePalette

Syntax:

```
void deletePalette(void)
```

Description:

- Delete the color palette

Parameters:

- null

Return:

- null

Example

```
1  #include <M5Unified.h>
2  #include <M5GFX.h>
3
4  #define ROYAL_BLUE 0X435C
5  #define LAVENDER_PURPLE 0xE73F
6  #define SADDLE_BROWN 0x8A22
7  #define INDIA_RED 0xCAEB
8  #define FOREST_GREEN 0x2444
9  #define SALMON_PINK 0xFC0E
10
11 static int32_t Disw;
12 static int32_t Dish;
13 static uint16_t pale[] = {WHITE, ROYAL_BLUE, LAVENDER_PURPLE, SADDLE_BROWN, BLUE, INDIA_RED, FOREST_GREEN,
14 // static uint16_t pale[256];
15
16 M5Canvas canvas(&M5.Lcd);
17
18 void setup() {
19     M5.begin();
20
21     Disw = M5.Lcd.width();
22     Dish = M5.Lcd.height();
23
24     canvas.createSprite(Disw, Dish);
25
26     canvas.setColorDepth(lgfx::v1::palette_8bit); // This must be cited before createPalette
27     canvas.setTextDatum(top_center);
28     canvas.drawString("M5Canvas Palette", Disw / 2, 0, &fonts::FreeMonoBold24pt7b);
29     canvas.drawString("Palette Color 0", Disw / 2, 100, &fonts::FreeMonoBold24pt7b);
30     canvas.drawString("is background color", Disw / 2, 150, &fonts::FreeMonoBold24pt7b);
31
32     canvas.createPalette(pale, 256);
33
34     // If you choose "static uint16_t pale[256];", following code must be used
35
36     // canvas.setPaletteColor(0, WHITE);
37     // canvas.setPaletteColor(1, ROYAL_BLUE);
38     // canvas.setPaletteColor(2, LAVENDER_PURPLE);
39     // canvas.setPaletteColor(3, SADDLE_BROWN);
40     // canvas.setPaletteColor(4, INDIA_RED);
41     // canvas.setPaletteColor(5, FOREST_GREEN);
42     // canvas.setPaletteColor(6, SALMON_PINK);
43
44     canvas.setTextColor(canvas.getPaletteIndex(ROYAL_BLUE));
45     canvas.drawString("Palette Color 1", Disw / 2, 300, &fonts::FreeMonoBold24pt7b);
46     canvas.setTextColor(canvas.getPaletteIndex(LAVENDER_PURPLE));
47     canvas.drawString("Palette Color 2", Disw / 2, 350, &fonts::FreeMonoBold24pt7b);
48     canvas.setTextColor(canvas.getPaletteIndex(SADDLE_BROWN));
49     canvas.drawString("Palette Color 3", Disw / 2, 400, &fonts::FreeMonoBold24pt7b);
50     canvas.setTextColor(canvas.getPaletteIndex(INDIA_RED));
```

```

51 canvas.drawString("Palette Color 4", Disw / 2, 450, &fonts::FreeMonoBold24pt7b);
52 canvas.setTextColor(canvas.getPaletteIndex(FOREST_GREEN));
53 canvas.drawString("Palette Color 5", Disw / 2, 500, &fonts::FreeMonoBold24pt7b);
54 canvas.setTextColor(canvas.getPaletteIndex(SALMON_PINK));
55 canvas.drawString("Palette Color 6", Disw / 2, 550, &fonts::FreeMonoBold24pt7b);
56
57 canvas.pushSprite(0,0);
58 canvas.deletePalette();//must behind pushSprite()
59 canvas.deleteSprite();
60 }
61
62 void loop() {
63 }

```

The example program runs as follows:



| readPixelValue

Syntax:

```
uint32_t readPixelValue(int32_t x, int32_t y)
```

Description:

- Read the pixel value at the specified coordinates

Parameters:

- x: pixel x coordinate
- y: pixel y coordinate

Return:

- uint32_t: pixel value
- 0-1 (1 bpp)
- 0-255 (8 bpp)
- 0-0xFFFF (RGB565-16 bpp)
- 0-0xFFFFFFFF (RGB888-24 bpp)

| setPsrAm

Syntax:

```
void setPsrAm( bool enabled )
```

Description:

- Set whether to use PSRAM.

Parameters:

- enabled: PSRAM usage flag
- true: use PSRAM
- false: do not use PSRAM

Return:

- null

| setBuffer

Syntax:

```
void setBuffer(void* buffer, int32_t w, int32_t h, uint8_t bpp = 0)
```

Description:

- Set the buffer

Parameters:

- buffer: buffer pointer
- w: width
- h: height
- bpp: bits per pixel (default is 0)
 - 0: use default
 - 1: 1 bit
 - 2: 2 bits
 - 4: 4 bits
 - 8: 8 bits
 - 16: 16 bits

Return:

- null

| pushRotated

Syntax1:

```
void pushRotated(float angle, const T& transp)
```

Syntax2:

```
void pushRotated(LovyanGFX* dst, float angle, const T& transp)
```

Syntax3:

```
void pushRotated(float angle)
```

Syntax4:

```
void pushRotated(LovyanGFX* dst, float angle)
```

Description:

- Rotate the canvas by the specified angle

Parameters:

- dst: target LovyanGFX object
- angle: rotation angle
- transp: transparent color

Return:

- null

| pushRotatedWithAA

Syntax1:

```
void pushRotatedWithAA(float angle, const T& transp)
```

Syntax2:

```
void pushRotatedWithAA(LovyanGFX* dst, float angle, const T& transp)
```

Syntax3:

```
void pushRotatedWithAA(float angle)
```

Syntax4:

```
void pushRotatedWithAA(LovyanGFX* dst, float angle)
```

Description:

- Rotate the canvas by the specified angle and use anti-aliasing

Parameters:

- dst: target LovyanGFX object
- angle: rotation angle

- transp: transparent color

Return:

- null

| pushRotateZoom

Syntax1:

```
void pushRotateZoom(float angle, float zoom_x, float zoom_y, const T& transp)
```

Syntax2:

```
void pushRotateZoom(LovyanGFX* dst, float angle, float zoom_x, float zoom_y, const T& transp)
```

Syntax3:

```
void pushRotateZoom(float dst_x, float dst_y, float angle, float zoom_x, float zoom_y, const T& transp)
```

Syntax4:

```
void pushRotateZoom(LovyanGFX* dst, float dst_x, float dst_y, float angle, float zoom_x, float zoom_y,
```

Syntax5:

```
void pushRotateZoom(float angle, float zoom_x, float zoom_y)
```

Syntax6:

```
void pushRotateZoom(LovyanGFX* dst, float angle, float zoom_x, float zoom_y)
```

Syntax7:

```
void pushRotateZoom(float dst_x, float dst_y, float angle, float zoom_x, float zoom_y)
```

Syntax8:

```
void pushRotateZoom(LovyanGFX* dst, float dst_x, float dst_y, float angle, float zoom_x, float zoom_y)
```

Description:

- Rotate the canvas by the specified angle and zoom factors

Parameters:

- dst: target LovyanGFX object
- dst_x: target x coordinate
- dst_y: target y coordinate
- angle: rotation angle
- zoom_x: x axis zoom factor
- zoom_y: y axis zoom factor
- transp: transparent color

Return:

- null

| pushRotateZoomWithAA

Syntax1:

```
void pushRotateZoomWithAA(float angle, float zoom_x, float zoom_y, const T& transp)
```

Syntax2:

```
void pushRotateZoomWithAA(LovyanGFX* dst , float angle, float zoom_x, float zoom_y, const T& transp)
```

Syntax3:

```
void pushRotateZoomWithAA(float dst_x, float dst_y, float angle, float zoom_x, float zoom_y, const T& t
```

Syntax4:

```
void pushRotateZoomWithAA(LovyanGFX* dst, float dst_x, float dst_y, float angle, float zoom_x, float zo
```

Syntax5:

```
void pushRotateZoomWithAA(float angle, float zoom_x, float zoom_y)
```

Syntax6:

```
void pushRotateZoomWithAA(LovyanGFX* dst, float angle, float zoom_x, float zoom_y)
```

Syntax7:

```
void pushRotateZoomWithAA(float dst_x, float dst_y, float angle, float zoom_x, float zoom_y)
```

Syntax8:

```
void pushRotateZoomWithAA(LovyanGFX* dst, float dst_x, float dst_y, float angle, float zoom_x, float zo
```

Description:

- Rotate the canvas by the specified angle and zoom factors, and use anti-aliasing

Parameters:

- dst: target LovyanGFX object
- dst_x: target x coordinate
- dst_y: target y coordinate
- angle: rotation angle
- zoom_x: x axis zoom factor
- zoom_y: y axis zoom factor
- transp: transparent color

Return:

- null

| pushAffine

Syntax1:

```
void pushAffine(const float matrix[6], const T& transp)
```

Syntax2:

```
void pushAffine(LovyanGFX* dst, const float matrix[6], const T& transp)
```

Syntax3:

```
void pushAffine(const float matrix[6])
```

Syntax4:

```
void pushAffine(LovyanGFX* dst, const float matrix[6])
```

Description:

- Perform an affine transformation using the specified matrix

Parameters:

- dst: target LovyanGFX object
- matrix: transformation matrix
- [a, c, e]
- [b, d, f]
- [0, 0, 1]
- transp: transparent color

Return:

- null

| pushAffineWithAA

Syntax1:

```
void pushAffineWithAA(const float matrix[6], const T& transp)
```

Syntax2:

```
void pushAffineWithAA(LovyanGFX* dst, const float matrix[6], const T& transp)
```

Syntax3:

```
void pushAffineWithAA(const float matrix[6])
```

Syntax4:


```
void pushAffineWithAA(LovyanGFX* dst, const float matrix[6])
```

Description:

- Perform an affine transformation using the specified matrix and use anti-aliasing

Parameters:

- dst: target LovyanGFX object
- matrix: transformation matrix
- [a, c, e]
- [b, d, f]
- [0, 0, 1]
- transp: transparent color

Return:

- null

Example

```
1  #include <M5GFX.h>
2  #include <M5Unified.h>
3
4  static int32_t Disw;
5  static int32_t Dish;
6  static float Affine_mat[9] = {1, 0, 60,
7                                0, 1, 20,
8                                0, 0, 1  };
9
10 void setup() {
11     M5.begin();
12     Disw = M5.Lcd.width();
13     Dish = M5.Lcd.height();
14
15     M5.Lcd.fillScreen(TFT_WHITE);
16
17     M5Canvas canvas(&M5.Lcd);
18     canvas.createSprite(100, 100); //set sprite size
19     canvas.fillSprite(TFT_BLACK); //fill sprite with color XXX
20     canvas.pushSprite(Disw / 2 - 50, Dish / 2 - 50);
21     delay(1000);
22     canvas.fillSprite(TFT_PINK);
23
24     // Rotate 45°
25     canvas.pushRotated(45);
26     // canvas.pushRotatedWithAA(45);
27
28     // Rotate 90° Reduce to 80%
29     // canvas.pushRotateZoom(90, 0.8, 0.8);
30     // canvas.pushRotateZoomWithAA(90, 0.8, 0.8);
31
32     // Shift 10 units from the origin using the affine matrix
33     // canvas.pushAffine(Affine_mat);
34     // canvas.pushAffineWithAA(Affine_mat);
35
36     canvas.deleteSprite();
37 }
38
39 void loop() {
40 }
```

The example program runs as follows:

[Fire_M5GFX_push.mp4](#)