Librairies disponibles pour piloter un écran OLED SSD1306

Pour gérer l’affichage sur votre écran OLED, il y a plusieurs librairies Open Source à notre disposition. Pour cet article, je vais vous présenter les deux principales. La librairie développée par Adafruit et celle de Sparkfun.

Librairies **Adafruit\_GFX et** Adafruit\_SSD1306

Adafruit a développé une librairie très puissante qui va nous permettre de gérer l’affichage de notre mini écran mais aussi de tracer plein de chose très facilement grâce à la librairie dédiée, GFX Library. Vous pouvez récupérer la librairie Adafruit SSD1306 sur le github[https://github.com/adafruit/Adafruit\_SSD1306](https://github.com/adafruit/Adafruit_SSD1306" \t "_blank) et la librairie GFX ici <https://github.com/adafruit/Adafruit-GFX-Library/archive/master.zip>.

*Attention. Il est nécessaire d’inclure les 2 librairies dans votre projet. Pour gagner quelques précieux octets, vous pouvez utiliser la librairie Micro\_OLED de Sparkfun décrite dans le prochain paragraphe.*

Liste des fonctions de la librairie disponibles

|  |  |
| --- | --- |
| Fonctions de la librairie Adafruit\_SSD1306 |  |
| Adafruit\_SSD1306 display(OLED\_RESET) | Initialise l’objet display(Pin pour le reset) |
| display() | Actualise l’affichage |
| clearDisplay() | Efface l’écran et le buffer |
| invertDisplay(bool) | Inverse l’affichage (true ou false) |

|  |  |
| --- | --- |
| Fonctions Adafruit\_GFX |  |
| drawPixel(uint16\_t x, uint16\_t y, uint16\_t color) | Dessine un pixel en X,Y de la couleur color |
| drawLine(uint16\_t x0, uint16\_t y0, uint16\_t x1, uint16\_t y1, uint16\_t color) | Dessine une ligne de X1,Y1 à x2,Y2 de la couleur color |
| drawFastVLine(uint16\_t x0, uint16\_t y0, uint16\_t length, uint16\_t color)  drawFastHLine(uin86\_t x0, uin86\_t y0, uint8\_t length, uint16\_t color); | Tracé optimisé de lignes verticales et horizontales |
| drawRect(uint16\_t x0, uint16\_t y0, uint16\_t w, uint16\_t h, uint16\_t color)  fillRect(uint16\_t x0, uint16\_t y0, uint16\_t w, uint16\_t h, uint16\_t color) | Dessine un rectangle depuis X,Y de largeur w et hauteur h  Idem mais plein |

|  |  |
| --- | --- |
| drawCircle(uint16\_t x0, uint16\_t y0, uint16\_t r, uint16\_t color)  fillCircle(uint16\_t x0, uint16\_t y0, uint16\_t r, uint16\_t color) | Dessine un cercle de centre X,Y et de rayon r  Idem mais le cercle est plein |
| drawRoundRect(uint16\_t x0, uint16\_t y0, uint16\_t w, uint16\_t h, uint16\_t radius, uint16\_t color)  fillRoundRect(uint16\_t x0, uint16\_t y0, uint16\_t w, uint16\_t h, uint16\_t radius, uint16\_t color) | Idem rectangle mais avec un arrondi de rayon raduis aux angles  Idem mais le rectangle est plein |
| drawTriangle(uint16\_t x0, uint16\_t y0, uint16\_t x1, uint16\_t y1, uint16\_t x2, uint16\_t y2, uint16\_t color)  fillTriangle(uint16\_t x0, uint16\_t y0, uint16\_t x1, uint16\_t y1, uint16\_t x2, uint16\_t y2, uint16\_t color) | Dessine un triangle en spécifiant les coordonnées de chaque sommets (x0,y1), (x2,y2), (x3,y3)  Idem mais le triangle est plein |
| drawChar(uint16\_t x, uint16\_t y, char c, uint16\_t color, uint16\_t bg, uint8\_t size) | Dessine un caractère en x,y |
| drawBitmap(int16\_t x, int16\_t y, uint8\_t \*bitmap, int16\_t w, int16\_t h, uint16\_t color) | Affiche un bitmap en x,y de largeur w et hauteur t |
| fillScreen(uint16\_t color); | Colorie entièrement l’écran dans la couleur spécifiée |
| setRotation(uint8\_t rotation) | Rotation de l’affichage : 0 -> 0°, 1 -> 90°, 2 -> 180°, 3 -> 270° |

Le paramètre color permettant de définir la couleur d’affichage uniquement sur les écrans couleur est disponible dans toutes les fonctions graphiques de la librairie GFX. Pour plus d’informations, vous pouvez consulter [ce document de formation en anglais](https://learn.adafruit.com/downloads/pdf/adafruit-gfx-graphics-library.pdf).

La librairie **Adafruit\_GFX** est utilisée par d’autres librairies dédiées à chaque (microcontroleur) écran.

Afficher du texte

L’affichage d’un texte demande un peu plus de travail. Il est nécessaire de modifier les paramètres d’affichage paramètre par paramètre. Voici un petit exemple pour afficher Hello Word en 0,0:

display.setTextSize(1);                    // setTextSize applique un facteur d’échelle qui permet d’agrandir ou réduire la font

display.setTextColor(WHITE);             // Définir la couleur du texte

display.setCursor(0,0);                  // Positionner le curseur en x=0, y=0

display.println(« Hello, world! »);                        // un println comme pour écrire sur le port série

display.setTextColor(BLACK, WHITE);             // On inverse les couleurs, le fond devient noir

display.println(« Hello, world! »);

// Vous pouvez changer à la volée de Font (pour cela vous devez la déclarer comme une librairie en début de projet, par exemple #include <Fonts/FreeMono9pt7b.h>)

display.setFont(&FreeMono9pt7b);

display.setTextColor(WHITE);

display.println(« Hello, world! »);

display.setFont();                                             // Pour revenir à la Font par défaut

Par contre, vous pouvez oublier les accents, ils ne sont pas gérés dans les Fonts disponibles.