AFFICHER DU TEXTE

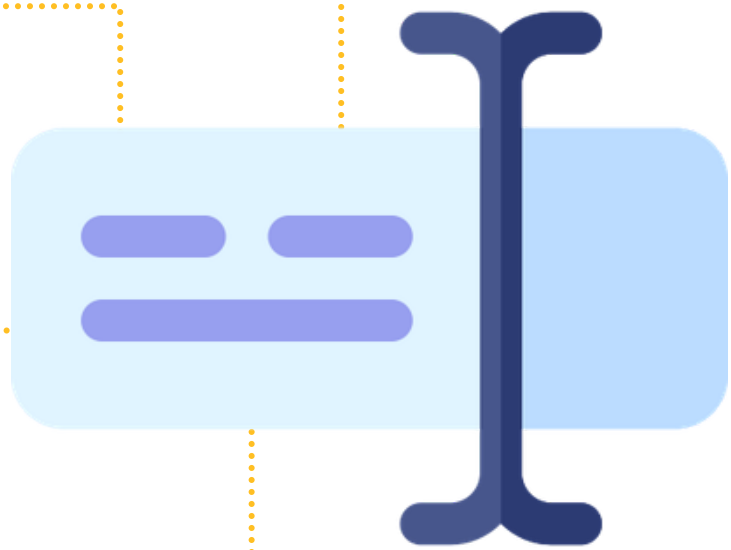
PROGRAMMATION EN MICROPYTHON

De quoi parle-t-on ?

L'écran OLED de la STeaMi permet d'afficher des informations issues du programme : un message d'accueil, l'état d'un capteur, la valeur d'une variable, pour rendre visible le comportement de la carte.

Disponible sur

MicroPython



Durée

60 min

Niveau de difficulté

Intermédiaire

Matériel

- 1 carte **STeaMi**
- 1 câble USB de données (micro-USB pour la STeaMi V1, USB-C pour la STeaMi V2)
- 1 ordinateur sous Windows, macOS ou Linux

Un éditeur MicroPython installé et configuré pour la STeaMi : Thonny (voir fiche de prise en main) ou tout autre éditeur compatible MicroPython (Mu, VS Code, Vittascience, mpremote...)

STeaMi

OBJECTIFS D'APPRENTISSAGE

- Comprendre comment l'écran s'actualise
- Composer un affichage à l'écran en plaçant texte et symboles aux points cardinaux ou à des coordonnées précises
- Hiérarchiser l'information affichée avec les raccourcis screen.title() et screen.subtitle()
- Faire vivre l'écran au rythme du programme pour suivre l'état d'une variable

Cette fiche d'activité a été produite par le Laboratoire d'Aix-périmentation et de Bidouille dans le cadre de l'action EXAO du projet I-Novmicro 2, une opération soutenue par l'État dans le cadre de l'AMI Compétences et Métiers d'Avenir du Programme France 2030, opéré par la Caisse des Dépôts, avec le soutien de la Région Sud.

AFFICHER DU TEXTE

ÉTAPE 1 - CONSTRUIRE

Programmer une carte électronique n'est pas toujours simple car son fonctionnement reste invisible. Contrairement à un programme exécuté sur ordinateur, les variables de votre code restent souvent inaccessibles à l'œil. L'écran intégré à la carte STeaMi permet cependant d'afficher les informations utiles pour suivre l'état de votre programme, qu'il s'agisse de la valeur d'un capteur, du résultat d'un calcul ou de l'évolution d'une variable.

Connecter la carte à l'ordinateur

Branchez la carte STeaMi à l'ordinateur grâce au câble USB et ouvrez votre éditeur. Il faudra s'assurer que l'éditeur a été configuré au préalable (par exemple pour Thonny, voir la fiche de prise en main).

Si votre éditeur est déjà configuré, une fois lancé, la console MicroPython doit afficher le prompt '>>>'. C'est **l'invite** (parfois appelée « prompt » en anglais) : un signe qui apparaît en début de ligne pour vous dire que la console est prête à recevoir une commande.

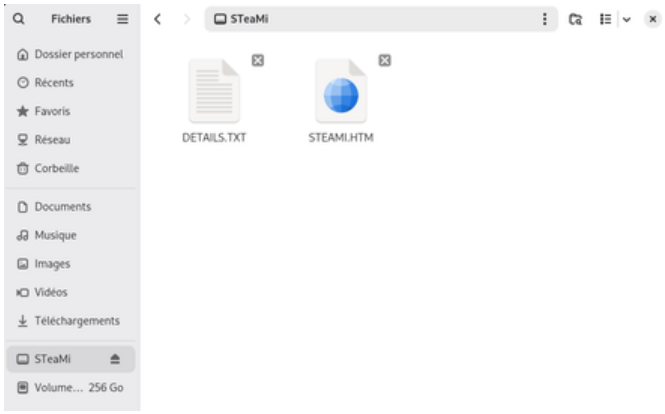
Les bibliothèques **ssd1327** et **steami_screen** sont déjà incluses dans MicroPython sur la STeaMi : aucune installation supplémentaire n'est nécessaire.

Repérer la géométrie de l'écran

L'écran de la STeaMi mesure 128 × 128 pixels. L'origine (0, 0) se situe en haut à gauche, l'axe horizontal va vers la droite, l'axe vertical vers le bas. Tout pixel peut donc être désigné par un couple de coordonnées comprises entre (0, 0) et (127, 127).

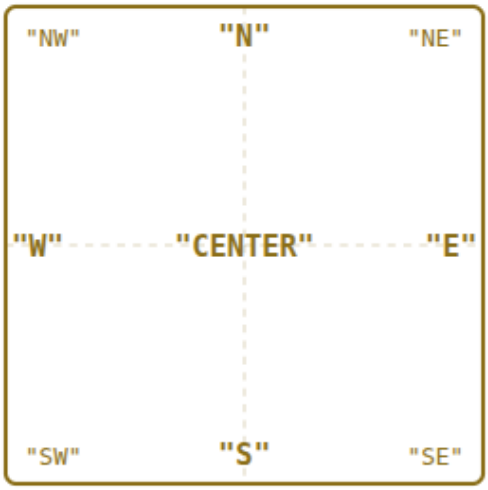
Pour éviter de calculer chaque coordonnée à la main, la bibliothèque **steami_screen** introduit neuf ancres réparties sur l'écran : les **quatre points cardinaux** (N, S, E, W), les **quatre diagonales** (NE, NW, SE, SW) et le **centre** (CENTER). Chaque ancre sert de point de référence autour duquel le texte est aligné.

1



Une fois la carte branchée, le lecteur STEAMI apparaît comme un disque amovible.

2



Les neuf positions cardinales utilisables avec screen.text(texte, at=...)

AFFICHER DU TEXTE

ÉTAPE 1 - CONSTRUIRE

Avec **at="N"**, le texte est aligné en haut et centré horizontalement ; avec **at="E"**, il est aligné à droite et centré verticalement ; avec **at="CENTER"**, il est centré dans les deux directions. Cette logique évite de mesurer la largeur d'une chaîne pour la placer correctement.

Programmer, exécuter, jouer

Afin de programmer votre carte, vous pouvez utiliser le programme fourni dans la section “Programmer”.

Une fois le code en place, deux manières de le lancer :

Test rapide. Lancez le programme depuis l’éditeur (typiquement bouton Run ▶️ ou F5). L'écran de la STeaMi affiche « **Hello World** » en haut, un cœur au centre et une ligne de tirets en bas.

Programme persistant. Enregistrez le fichier sous le nom main.py sur la carte, puis redémarrez (bouton RESET, ou Ctrl+D dans la console MicroPython qui réinitialise l'interpréteur). Le programme sera relancé à chaque démarrage de la carte.

Une fois le résultat à l'écran, vous pouvez ensuite faire évoluer le code : modifier le texte affiché, changer la position d'ancrage du cœur, jouer sur les couleurs ou afficher la valeur d'une variable qui évolue dans le temps.

3

Le programme tourne sur la carte

Le fichier main.py prêt à être téléversé sur la carte

AFFICHER DU TEXTE

ÉTAPE 2 - PROGRAMMER

```
# Testée avec firmware STeaMi 0.23.1
# Affichage de texte sur l'écran OLED 128x128 (SSD1327) intégré à la STeaMi.
# Hello World en haut, cœur au centre, ligne décorative en bas.

import ssd1327
from machine import SPI, Pin
from steami_screen import Screen, SSD1327Display

# Initialisation de l'écran intégré (SSD1327, 128x128)
spi = SPI(1)
dc  = Pin("DATA_COMMAND_DISPLAY")
res = Pin("RST_DISPLAY")
cs  = Pin("CS_DISPLAY")
raw     = ssd1327.WS_OLED_128X128_SPI(spi, dc, res, cs)
display = SSD1327Display(raw)
screen  = Screen(display)

# Affichage
screen.clear()                          # efface l'écran (noir)
screen.title("Hello World")             # texte en haut, centré
screen.text("<3", at="CENTER")          # cœur au centre de l'écran
screen.text("------------", at="S")     # ligne de tirets en bas
screen.show()                           # affiche sur l'écran
```

Fonctionnement du programme

- La fonction **screen.text(texte, at=..., color=..., scale=...)** permet d'afficher du texte sans calculer de coordonnées en pixels : il suffit d'indiquer une **ancre cardinale** parmi celles décrites à l'étape 1, section « **Repérer la géométrie de l'écran** », ou un couple de coordonnées explicites.
- Deux raccourcis pratiques structurent l'affichage : **screen.title()** place un texte en haut en **GRAY**, screen.subtitle() un texte en bas en **DARK**. Il n'est pas nécessaire de préciser position ni couleur.
- Les couleurs disponibles dans **steami_screen** sont les suivantes : BLACK, DARK, GRAY, LIGHT, WHITE (5 niveaux de gris du SSD1327), plus RED, GREEN, BLUE, YELLOW (qui dégradent automatiquement en niveaux de gris sur l'écran monochrome).
- Rien n'apparaît tant que **screen.show()** n'a pas été appelée. On peut imaginer l'écran de la STeaMi comme un tableau noir caché derrière un voile. Les fonctions **text(), clear(), pixel(),** etc. dessinent sur le tableau, mais le voile reste en place tant qu'on n'a pas appelé screen.show(). À ce moment-là, le voile tombe et tout ce qu'on a dessiné apparaît d'un coup. Techniquement, ce « tableau caché » s'appelle un **framebuffer** : une zone de mémoire dans laquelle on prépare l'image, avant de la transférer vers l'écran.
- Si vous avez besoin de coordonnées exactes (par exemple pour une animation pixel par pixel), **steami_screen** expose aussi **screen.pixel(), screen.line(), screen.rect() et screen.circle()**.

AFFICHER DU TEXTE

ÉTAPE 3 - AMÉLIORER

Afficher le cœur en grand

screen.text() accepte un facteur d'échelle qui agrandit le texte. Essayez **screen.text("<3", at="CENTER", scale=3)**. Quelle est la limite avant que le texte ne sorte de l'écran ?

Animation à La Linea

Avec une boucle **while**, créez une animation simple qui fait avancer un personnage minimaliste basé sur les symboles | et _. Utilisez **sleep_ms(100)** pour ralentir le mouvement et n'oubliez pas d'appeler **screen.clear()** au début de chaque image pour effacer l'écran.

État du bouton USER

Affichez en temps réel si le bouton A de la STeaMi est pressé ou non, par exemple avec **screen.title("A: ON")** ou **screen.title("A: OFF")**. Que se passe-t-il avec un **sleep_ms(1000)** long dans la boucle ? Comment améliorer la réactivité ? (Indice : raccourcir le délai. Pour aller plus loin, vous pouvez explorer la programmation asynchrone avec uasyncio.)

Tableau de bord capteurs

Affichez simultanément plusieurs valeurs en utilisant les positions cardinales pour répartir l'information sur l'écran. Une fois la mise en page maîtrisée, branchez de vraies valeurs en lisant les capteurs intégrés (**HTS221** pour la température/humidité, **ISM330DL** pour l'accélération).

ALLER PLUS LOIN

Survival Series de Jenny Holzer (Centre Pompidou) - Comprenez comment l'artiste a fait du panneau LED un médium à part entière, depuis Times Square en 1982. *https://www.centrepompidou.fr/fr/ressources/oeuvre/c888Bex*

Pixel Revival : quand l'esthétique 8-bit redéfinit le design graphique - Découvrez comment la contrainte du pixel est devenue une esthétique revendiquée dans le design contemporain. *https://www.etapes.com/2025/04/18/pixel-revival-quand-lesthetique-8-bit-redefinit-les-codes-du-design-graphique/*