Programmer Badger avec Thonny

Table of Contents

Une fois MicroPython installé, Badger n'apparaîtra plus comme un lecteur. Pour le programmer, il faut lui parler via un interprète - **Thonny**, qui est disponible pour Windows, Mac ou Linux.

- Installer la dernière version de **Thonny**. Il est recommandé de le télécharger à partir du site Web de **Thonny**, car les gestionnaires de packages ne disposent pas toujours de la version la plus récente.
- Ouvrit Thonny, s'assurer que l'interpréteur (affiché dans la case en bas à droite) est défini sur “MicroPython (Raspberry Pi Pico)”.
- Brancher le Badger sur l'ordinateur, s'il n'est pas déjà branché. Étant donné que le lanceur sera déjà en cours d'exécution, il faut l'arrêter avec le bouton d'arrêt de **Thonny** avant de lui envoyer des instructions.
- Après avoir appuyé sur stop, on doit obtenir une invite MicroPython. Le curseur clignotant à côté de >>> dans la case 'Shell' indique que Badger 2040 parle à l'ordinateur et est prêt à accepter des instructions ! (si on ne reçois pas d'invite ou si on obtiens une erreur, on a peut-être interrompu Badger à un moment inadéquat du programme, essayer d'appuyer sur reset sur Badger et appuyer à nouveau sur stop).

Installation de MicroPython pour Badger

Badger est préchargé avec MicroPython et une suite d'exemples, mais pour toutes les dernières fonctionnalités, ajustements et correctifs, il faut télécharger la version la plus récente.

Ces instructions supposeront qu'on utilise la version 1.18.5 ou une version ultérieure (publiée le 25/03/2022) - il y a eu pas mal de changements, donc si quelque chose ne fonctionne pas comme il se doit, essayer de mettre à jour la version de MicroPython !

- Télécharger le fichier MicroPython `.uf2` le plus récent à partir de la page Releases du dépôt Github pimoroni-pico (la version spéciale pour Badger 2040, qui intègre le lanceur BadgerOS et des démos).

- Maintenir enfoncé le bouton BOOT du Badger 2040, puis appuyer et relâcher le bouton RESET. Cela le mettra en mode bootloader, et il devrait apparaître comme un lecteur sur votre ordinateur appelé RPI-RP2.
- Copier le fichier `.uf2` sur le lecteur RPI-RP2 - une fois fait cela, Badger 2040 redémarrera en exécutant BadgerOS !

Actions de base

Allumer la LED intégrée

Pour vérifier que la carte fonctionne correctement et qu'on a installé la bonne version de MicroPython, allumer la LED blanche intégrée avec le code suivant - il faut l'entrer ligne par ligne dans le REPL (c'est la case 'Shell' en bas dans Thonny).

```
import badger2040
badger = badger2040.Badger2040()

badger.led(255)
```

(Ces deux premières lignes sont du code passe-partout dont on a besoin au début de tout code que l'on veut exécuter sur Badger).

Eteindre à nouveau la LED, pour être bien rangé !

```
badger.led(0)
```

Écrire du texte à l'écran

Voici comment afficher du texte à l'écran, dans sa forme la plus élémentaire. Vous devrer entrer le passe-partout, comme avant (sauf si vous utiliser toujours la même session Thonny).

```
import badger2040
badger = badger2040.Badger2040()

badger.pen(0)
badger.text("Hello Badger", 20, 20)
badger.update()
```

Le reste du code fait ceci :

- Choisit une couleur de stylo avec laquelle écrire - 0 est noir et 15 est blanc. Cet écran n'affichera que du noir ou du blanc, mais si vous choisisser un nombre compris entre 0 et 15, il le modifiera pour se rapprocher d'une nuance de gris.
- Écrit du texte dans le tampon d'écran - le '20, 20' est les coordonnées x/y du milieu gauche du texte.

- Met à jour l'écran E Ink (on peut dessiner plusieurs éléments dans le tampon d'écran avant de déclencher une mise à jour).

Si on essaye d'écrire plus de texte à l'écran, on constatera que les éléments qu'on a dessinés dans la mémoire tampon de l'écran resteront jusqu'à ce qu'on les écrase. Pour effacer l'écran et repartir à zéro :

```
badger.pen(15)
badger.clear()
```

Il y a plus de fonctions dans la bibliothèque pour changer la police, la taille et l'épaisseur du texte, ainsi que des fonctions pour dessiner des pixels, des lignes et des rectangles individuels.

Personnalisation des exemples BadgerOS

Lorsqu'on les exécute, certains des exemples fournis créent leurs propres fichiers de données qu'on peut modifier.

Pour que la fenêtre Fichiers soit visible activer **View > Files.**

La case supérieure peut être utilisée pour parcourir les fichiers locaux sur l'ordinateur, et la case inférieure affiche les fichiers sur Badger. On peut copier des fichiers de l'ordinateur vers le Badger en cliquant dessus avec le bouton droit de la souris dans la fenêtre du haut et en choisissant “Télécharger vers/” (ou ouvrir un fichier en cliquant dessus dans la fenêtre du bas, puis appuyer sur “enregistrer” lorsque c'est fait). Lorsqu'on a terminé de modifier/télécharger les fichiers de données, appuyer sur le bouton de réinitialisation du Badger pour redémarrer le lanceur.

Les fichiers de données sont créés la première fois qu'on exécute les exemples pertinents sur le Badger, donc si on brancher un nouveau Badger et commence à parcourir les fichiers, on ne les verra pas encore.

e-reader.py

Cet exemple est accompagné d'un extrait de Wind in the Willows - remplacer le contenu de 'book.txt' par un autre fichier texte pour qu'il s'affiche dans la liseuse.

Lors de l'exécution de l'exemple, appuyer sur A et B pour modifier le style et la taille de la police.

liste.py

Pour échanger la liste de contrôle de démonstration avec l'une des vôtres, modifier le contenu de checklist.txt. Chaque élément de la liste doit figurer sur une ligne distincte.

```
cake
biscuits
bees
mashed potato
```

image.py et ajout d'images

Pour obtenir des images sur Badger à l'aide de MicroPython, il faut d'abord les convertir en un tableau d'octets de 1 bit qui représente l'image monochrome. Un script Python fournit peut prendre en charge la conversion. Le script convertira une image bmp, jpg ou png en un fichier .bin formaté en noir et blanc de 296 x 128 que Badger pourra lire.

Voici quelques façons différentes de l'exécuter !

Conversion d'images à partir d'un Raspberry Pi

(ou autre ligne de commande Linux ; Ubuntu, Mac, WSL, etc.)Ouvrer un terminal.

- Télécharger le repo pimoroni-pico sur l'ordinateur : `git clone https://github.com/pimoroni/pimoroni-pico`
- Accéder à l'outil de conversion d'image : `cd pimoroni-pico/examples/badger2040/image_converter`
- Ensuite, en supposant que le fichier image de départ se trouve dans le même répertoire que 'convert.py', on peut le convertir comme ceci (indiquer le nom du fichier) : `python3 convert.py --binary --resize image_file_1.png`
- Ou convertir plusieurs images comme ceci : `python3 convert.py --binary --resize image_file_1.png image_file_2.png image_file_3.png`
- Si on a une erreur “Aucun module nommé PIL”, installer pip3 pillow, puis réessayer.

Conversion d'images depuis Thonny

(cela peut être une option plus facile lorsqu'on utilise Windows)

- Basculer l'interpréteur sur 'Le même interpréteur qui exécute Thonny (par défaut)' en utilisant la case en bas à droite. Cela lui indique d'utiliser la version complète de Thonny de Python, au lieu de la version de Badger de MicroPython.
- Installer PIL (Python Image Library) dans **Thonny**, avec **Outils > Gérer** les packages. Rechercher Pillow, cliquer dessus, puis sur “Installer”.
- Cliquer sur “nouveau” et copier et coller le contenu de **convert.py** dans l'onglet vide. Enregistrer quelque part sur l'ordinateur, nommé **convert.py** (au même endroit que les images que l'on veut convertir).

- Dans la zone inférieure du shell à l'invite Python, on peut ensuite exécuter **convert.py** depuis **Thonny** comme ceci (en modifiant les noms de fichiers) : `%Run convert.py --resize --binary image_file_1.png`

- Appuyer sur Entrée après avoir tapé cela - si on appui sur le bouton d'exécution vert, seule la première partie de la commande sera exécutée.
- Rebasculer l'interpréteur vers MicroPython (Raspberry Pi Pico) lorsque vous avez terminé la conversion !

Une fois que l'on a généré le fichier .bin, on peut le copier dans Badger en utilisant le menu Fichiers de Thonny :

- Accéder au fichier dans la fenêtre supérieure “Cet ordinateur”.
- Dans la case du bas, double-cliquer sur le répertoire d'images du Badger pour l'ouvrir.
- Faire un clic droit sur le fichier .bin dans la fenêtre du haut et sélectionner “Télécharger vers/images”

Les nouvelles images devraient apparaître dans l'exemple d'image après une réinitialisation de la carte.

Pour plus de contrôle sur le tramage, etc. (et un aperçu de ce à quoi elles vont ressembler), essayer de convertir les images en 1 bit noir et blanc avec un programme graphique tel que GIMP avant de les exécuter dans le script.

badge.py

Pour modifier le texte de l'exemple de badge nominatif, modifier 'badge.txt'. Double-cliquer sur badge.txt dans le menu Fichiers pour l'ouvrir dans **Thonny** (“sauvegarder” le fichier une fois terminé).

Pour changer l'image, il faut convertir l'image comme ci-dessus, avec les modifications suivantes :

- L'image du badge nominatif doit être nommée 'badge-image.bin' et dans le répertoire racine du Badger à côté de badge.txt (pas dans le répertoire /images)
- Cette image doit avoir exactement 104 pixels de largeur et 128 pixels de hauteur, il faut donc la redimensionner avec un programme graphique tel que GIMP avant de l'exécuter dans le script de conversion. Lorsqu'on viens de le convertir, n'utiliser pas l'indicateur **–resize** car cela l'étirera à 296 x 128, utiliser plutôt `python3 convert.py --binary badge-image.png`
- ou depuis **Thonny** : `%Run convert.py --binary badge-image.png`

Exécution de code

Si on souhaite modifier davantage les exemples (ou en écrire à partir de zéro), on peu trouver le code Python pour tous les exemples intégrés (plus quelques extras) sur Github.

Les exemples intégrés n'apparaîtront pas comme des fichiers Python qu'on peut modifier sur Badger - il faut copier et coller le code dans un nouveau fichier dans **Thonny** (cliquer d'abord sur le bouton raw facilite la copie) et appuyer sur le bouton vert 'exécuter' pour l'exécuter sur le Badger.

On sera invité à entrer un nom de fichier pour l'enregistrer. Si on lui donne le même nom de fichier que l'un des exemples intégrés (badge.py, clock.py, etc.), le lanceur exécutera la version de l'exemple au lieu de celle intégrée. Si on souhaite revenir à l'original, supprimer simplement la copie de Badger.

Dépannage

Thonny ne peut pas voir le Badger / L'appareil est occupé

Vérifier que l'interpréteur est réglé sur “Raspberry Pi Pico” et appuyer sur “stop” pour arrêter le lancement du lanceur ! Il faut que l'invite >>> soit visible dans le 'sh pour pouvoir transférer des fichiers ou envoyer des instructions Python à Badger.

Il suffit de maintenir enfoncé le bouton BOOT tout en mettant Badger en mode bootloader pour pouvoir y copier MicroPython, on n'a pas besoin qu'il soit en mode bootloader après cela (sauf si on souhaite télécharger un nouveau firmware).

L'indicateur de batterie dans le lanceur ne fonctionne pas

L'icône de la batterie en haut à gauche du lanceur utilise des valeurs de tension qui sont raisonnables pour une batterie LiPo, donc si on utiliser des piles AA/AAA, elles peuvent apparaître comme vides, même s'il reste beaucoup de jus. Pour que l'icône de la batterie représente mieux la situation de la batterie, enregistrer une propre copie de **launcher.py** sur le Badger et modifier les valeurs min/max à quelque chose de plus approprié - quelque chose comme ça devrait mieux fonctionner pour 2 x AAA (non rechargeables) :

```
MAX_BATTERY_VOLTAGE = 3,2
MIN_BATTERY_VOLTAGE = 2,0
```

Comment réinitialiser les paramètres d'usine ?

Si on a besoin de supprimer tous les programmes de la mémoire flash de Badger et de recommencer à zéro, on peut le faire en téléchargeant ce fichier .uf2 spécial et en le copiant sur le Badger pendant qu'il est en mode bootloader. Une fois qu'on a fait cela, copier à nouveau l'image Badger MicroPython.

Il peut également être utile d'effacer le flash si on rencontre des problèmes après la mise à niveau vers la dernière version de MicroPython

Penser à enregistrer d'abord tout code sur lequel on a travaillé sur l'ordinateur

Branchement d'une batterie sur le connecteur JST

Le moyen le plus simple d'alimenter Badger en électricité consiste à utiliser des piles double ou triple A. Le mieux est d'utiliser une configuration 2 x AAA et un support de batterie avec un connecteur JST-PH (Galleon 400mAh Hard Case LiPo Battery) pour pouvoir le brancher, car la batterie se colle à l'arrière du Badger sans être trop lourde et encombrante.

2 piles AAA rechargeables (NiMH) ne délivrent que 2,4 V, ce qui, à proprement parler, n'est pas suffisant pour Badger. Cependant, lors des tests, il continue de tourner jusqu'à une tension d'entrée de 2,05 V (sans la LED), donc si on souhaite utiliser des piles rechargeables, cela devrait aller.

Il est également possible d'alimenter Badger à partir d'une batterie LiPo/LiIon (pimoroni LiPo battery packs 5000mAh)

Lorsqu'on utilise la version 1.18.5 ou une version ultérieure, le lanceur et la plupart des exemples s'éteignent désormais automatiquement pour économiser la batterie. Lorsqu'il fonctionne sur batterie, le voyant s'allume lorsque Badger est éveillé et consomme de l'énergie, et s'éteint pour faire savoir quand la carte est endormie. Appuyer sur n'importe quel bouton avant pour le réactiver, ou appuyer simultanément sur A et C pour revenir au lanceur.