Tamagotchi

Le tamagotchi est un animal virtuel originaire du Japon qui a eu un énorme succès auprès des enfants en 1997.

Prérequis

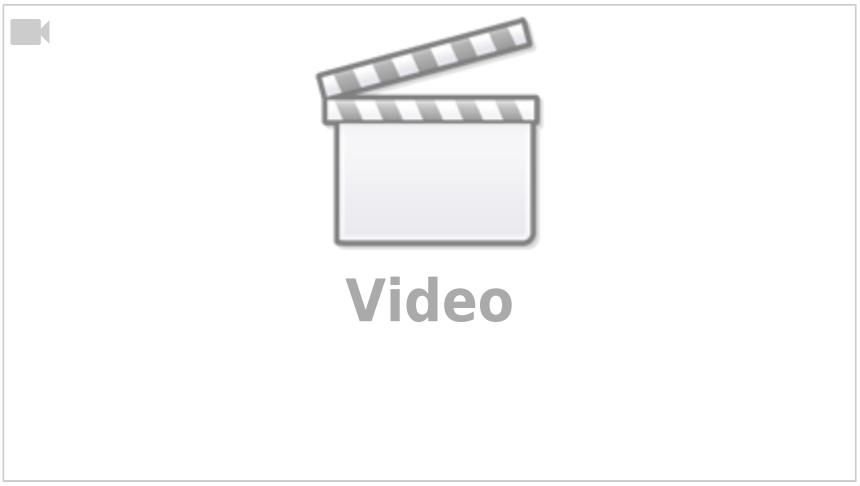
un éditeur python et un navigateur connecté

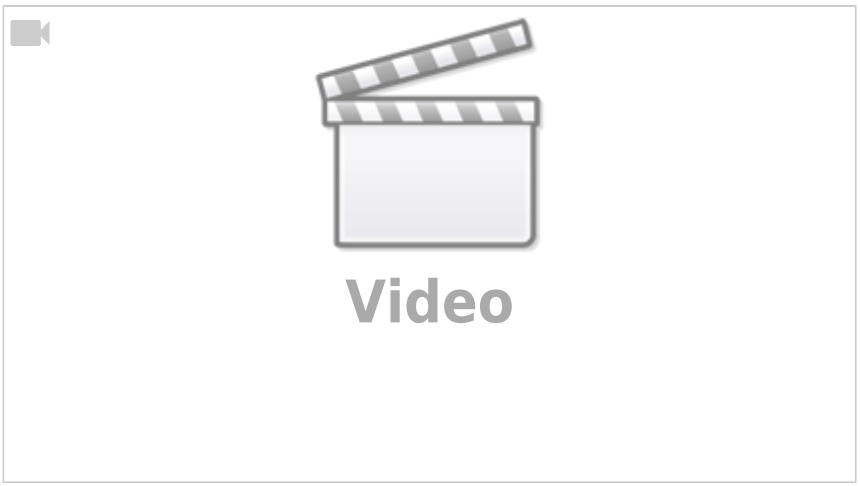
Objectifs

1. Découvrir l'environnement de développement python (IDE, git)
2. identifier les morceaux (commentaires, fonctions, variables etc.) dans un code python et comprendre le fonctionnement global du script
3. lire et modifier un code source

Étape 0 : introduction

Présentation du Tamagotchi



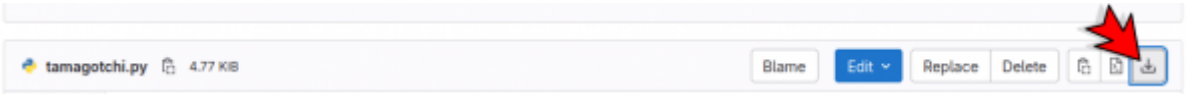


Objectifs de l'atelier

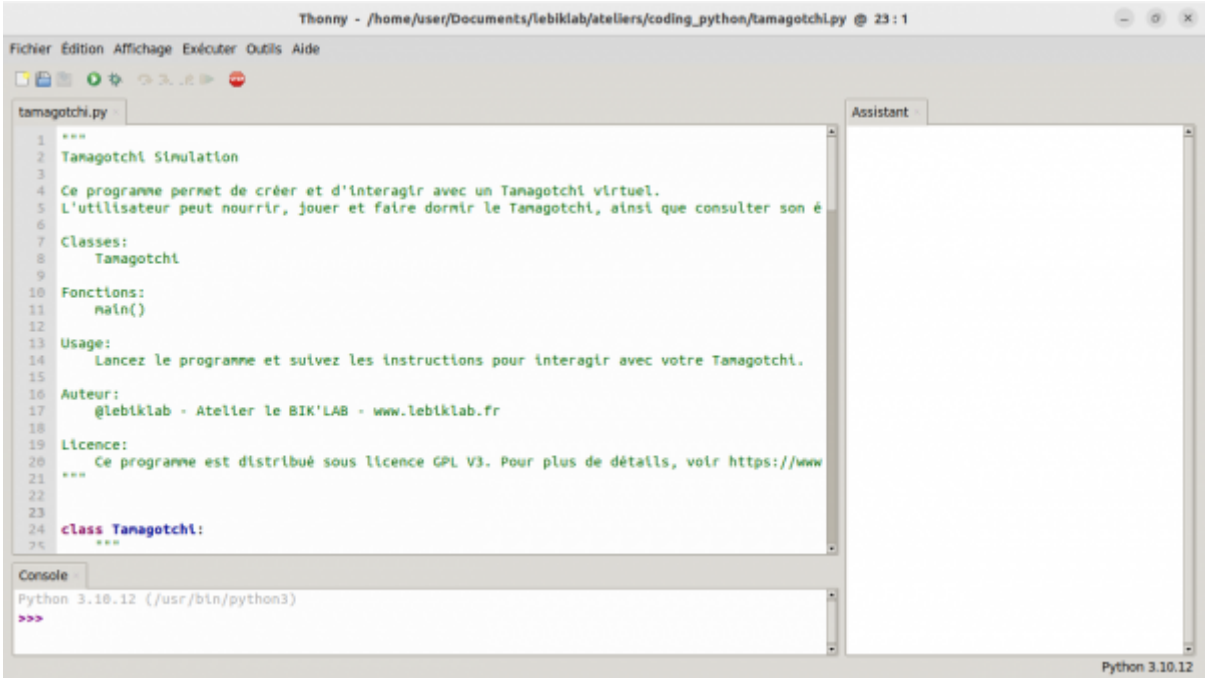
1. Découvrir l'environnement de programmation python
2. Créer un programme interactif

Étape 1 : récupérer le projet à modifier

1. avec le navigateur aller sur la page https://gitlab.com/lebiklab/atelier-python/-/blob/main/tamagotchi.py
2. cliquer sur le "Download"



1. enregistrer le fichier localement sous le nom `tamagotchi.py` (par convention on enregistre les fichiers python avec l'extension `.py`)
2. ouvrir le fichier dans Thonny



3. lire le code, identifier et distinguer des blocs en se repérant avec les couleurs

4. exécuter le code et interagir avec

Étape 2 : ajouter un signal sonore pour le tamagotchi

Le Tamagotchi traditionnel se manifeste par un bip. Nous allons ajouter un bip à notre tamagotchi. Pour cela, nous allons récupérer le son “Alerte” de notre système d’exploitation. Le son produit pourra donc être différent selon la machine sur laquelle nous exécuterons notre script python `tamagotchi.py`

Aller à la ligne n°124, `while True:`, créer une nouvelle ligne et ajouter le code suivant `print(“\a”)`

Attention à bien respecter l’indentation du code

En python, c’est en utilisant des tabulations que l’on délimite les blocs de code, à partir du bord gauche de la “page” (l’écran).

Ta ligne `print(“\a”)` doit donc être alignée avec les instructions qui suivent le `While True:`

```
    while True:
        # BIP !
        print("\a")
        print("\nChoisis une action: ")
        print("1. Nourrir")
        print("2. Jouer")
        print("3. Dormir")
        print("4. Statut")
        print("5. Quitter")
```

Relance le jeu

Étape 4 : Améliorations et fonctions à ajouter

''sleep'' : ajouter une pause

On va ajouter une pause entre chaque action. Pendant cette pause, le Tamagotchi pourra de lui-même soit dormir, soit se nourrir, soit jouer, soit faire ses besoins (et donc salir). Le fait de faire ces actions par lui-même fera varier les niveaux de bonheur, fatigue et faim du Tamogotchi.

des seuils

On va également ajouter des seuils. Ex : si le Tamagotchi a 10 de faim et 10 de fatigue, il meurt. Si le Tamagotchi à 0 de bonheur, il refuse de manger et veut jouer

la fonction caliner

pour faire un calin à son Tamagotchi la fonction caliner n'existe pas encore ...)

Étape 5 : ajout de la pause

1. Rechercher dans un moteur de recherches "Comment faire une pause dans un programme python avec sleep"
2. Créer un fichier `test_sleep.py` avec un exemple fonctionnel de l'utilisation de sleep, en faisant une pause de 30 secondes.
3. Tester le script `test_sleep.py`
4. Quand ce script fonctionne, ajouter les élements nécessaires (et aux bons endroits) dans le programme `tamagotchi.py`

test_sleep.py

exemple de code avec une pause de 30 secondes

```
import time
print("Cela commence ici : %s" % time.ctime())
time.sleep(30)
print("Le temps est maintenant écoulé : %s" % time.ctime())
```

Les lignes importantes sont

- `import time` qui permet d'importer la libraire `time`
- `time.sleep(30)` qui utilise la libraire `time` pour faire une pause de 30 secondes

Modification du fichier "tamagotchi.py"

import

À la ligne 24, avant la déclaration de la `class Tamagotchi`, ajouter la ligne d'import

```
"""
[...]
Licence:
    Ce programme est distribué sous licence GPL V3. Pour plus de détails,
voir https://www.gnu.org/licenses/gpl-3.0.html
"""


import time

class Tamagotchi:
    """
```

```
    Classe représentant un Tamagotchi virtuel.
    [...]
    """
```

sleep

À la ligne 127, avant l'action `print("\a")` (BIP), on va ajouter la pause

```
    while True:
        # pause
        time.sleep(30)
        # BIP !
        print("\a")
```

Test

1. Exécuter et tester le code
2. Avez-vous des idées d'amélioration ?

Actuellement, la pause s'execute systématiquement : après avoir nommé le tamagotchi et même quand on consulte son statut. On pourrait améliorer ça en faisant une pause uniquement lorsque le tamagotchi effectue une action comme dormir, manger ou jouer ?

Amélioration

Comment faire pour marquer une pause uniquement dans les actions dormir, manger ou jouer ?

on va déplacer l'instruction sleep(30) dans la définition d'une methode d'action (ex nourrir)

 attention à placer l'instruction au bon endroit et à respecter l'indentation !

```
    def dormir(self):
        """
        Fait dormir le Tamagotchi, augmentant son énergie.

        Si le niveau d'énergie est inférieur à 10, il est augmenté de 2. Si le niveau
        d'énergie est déjà à 10, un message indiquant que le Tamagotchi est déjà plein d'énergie
        est affiché.
        """
        if self.energie < 10:
            self.energie += 2
            sleep(30)
            print(f"{self.name} a dormi. Énergie: {self.energie}")
```

```
        else:
            print(f"{self.name} est déjà plein d'énergie!")
```

On va maintenant ajouter cette pause à chacune des méthodes dormir(), manger() et jouer().

On va également attribuer un temps de pause différent pour chacune de ces actions

}

Étape 6 : Ajout de la fonction ''caliner''

Si le tamagotchi a faim, il refusera le calin. Sinon, grâce au calin, ses niveaux d'energie et de bonheur augmenteront

Exercice : ajouter la méthode `caliner()` dans la `class Tamagotchi()` en s'inspirant des méthodes `nourrir()`, `dormir()` ou `manger()`.

- seuil de faim maximum pour un calin : 8
- un calin ajoute :
  - + 2 de bonheur
  - +1 en energie

Pour pouvoir utiliser ta fonction tu dois l'ajouter dans le menu de ton application (les instructions affichées à l'écran après le `While True:`) et prendre en compte le choix de l'utilisateur (`if choice ==`)

Bonus récupérer des pikachu en ASCII art pour illustrer le mood du Tamagotchi (print de docstring)