Python sur Trinket – Leçon 2

Révision

- Axes X/Y & leurs coordonnées
- Crayon levé, Crayon descendu
- Commencer « remplir » , terminer « remplir »

Discussion

- **Importer la tortue**
  - Importer la tortue nous permet d'utiliser les commandes associées à la tortue comme la création de formes.

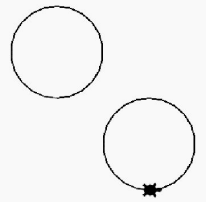
- **Fonctions**
  - Les fonctions sont des catégories de codes réutilisables organisées en blocs qui permettent d'exécuter des actions.
  - Des fonctions sont intégrées à la bibliothèque de code Python
    - Certaines fonctions n'ont pas besoin de précision
      - c.-à-d. turtle.penup( )
    - Certaines fonctions nécessitent que le programmeur écrive des informations supplémentaires,
      - c.-à-d. turtle.color("color")
  - D'autres fonctions ne font pas partie de la bibliothèque Python. Elles doivent être définies par l'utilisateur (programmeur).

- **Commentaire**
  - `# Ceci est un commentaire d'une ligne`
  - ```
    ''' Ceci est un
    commentaire de
    plusieurs lignes '''
    ```
  - Les commentaires sont des mots qui ne seront pas exécutés dans le programme
  - Utilisez les commentaires selon vos préférences ou votre style de programmation. Les commentaires ne sont pas vraiment pour l'utilisateur, mais plutôt afin de fournir des informations utiles à celui qui crée le code. Par exemple :
    - Les commentaires peuvent être écrits au-dessus d'un bloc de code afin d'aider le programmeur à se rappeler ce que le code entraîne
    ```

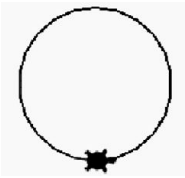
comme actions. Ainsi, lorsqu'on révise notre programme, nous n'avons pas nécessairement besoin de réviser toutes les lignes pour nous souvenir de ce que fait le code.

```
# Cette fonction crée un bonhomme de neige
Ligne de code pour créer un bonhomme de neige
Ligne de code pour créer un bonhomme de neige
```

- Si vous voulez exclure une ligne de code de votre programme sans nécessairement l'effacer, vous pouvez ajouter un # pour qu'elle soit considérée comme un commentaire. Pour l'intégrer à nouveau au code, effacer le #.

`turtle.circle(50)`
`turtle.penup()`
`turtle.goto(100, -100)`
`turtle.pendown()`
`turtle.circle(50)`
	

Voyons ce qu'il se passe si nous ne changeons pas nos coordonnées X&Y	

`turtle.circle(50)`
`turtle.penup()`
`#turtle.goto(100, -100)`
`turtle.pendown()`
`turtle.circle(50)`
	

- **Erreurs**
  - Python vous permet d'identifier facilement les erreurs de votre code en affichant un message d'erreur
  - C'est le travail du programmeur de comprendre la nature de l'erreur et de modifier celle-ci
  - c.-à-d.

```
11    noexist.penup()
```

```
NameError: name 'noexist' is not defined on line 11 in main.py
```

Commandes générales

`import turtle`	`turtle.circle(size)`
`x = turtle.Turtle() / import turtle as x`	`turtle.right(degrees) / turtle.left(degrees)`
`turtle.shape("shape")`	`turtle.forward(length) / turtle.backward(length)`
`turtle.penup() / turtle.pendown()`	`turtle.speed(speed)`
`turtle.pencolor("color") / turtle.fillcolor("color") / turtle.pencolor(r,g,b)`	`turtle.width(width)`
`turtle.goto(x,y)`	`turtle.begin_fill() / turtle.end_fill()`

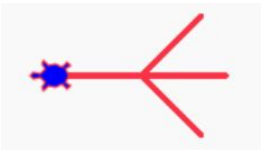
Initialiser un programme

```
import turtle
import random
t = turtle.Turtle()
t.shape("turtle")
t.speed(speed)
```

Pratiquer

- Créer le bras de bonhomme de neige

Snowflake Arm	

`turtle.shape("turtle")`
`turtle.pendown()`
`turtle.forward(50)`
`turtle.left(45)`
`turtle.forward(50)`
`turtle.backward(50)`
`turtle.right(45)`
`turtle.forward(50)`
`turtle.backward(50)`
`turtle.right(45)`
`turtle.forward(50)`
`turtle.backward(50)`
`turtle.right(135)`
`turtle.forward(50)`
	


kids code
jeunesse