Initiation à la programmation – TP3 et TP4

Un jeu à cliquer avec plusieurs versions

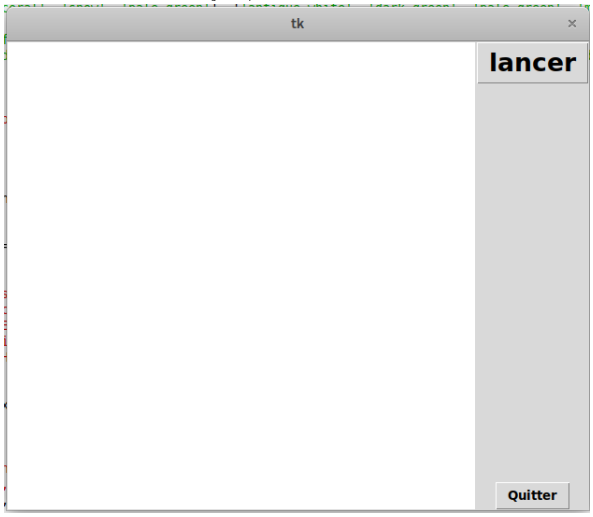
Il s'agit de mettre en place des outils qui permettront ensuite de réaliser plusieurs jeux où on joue avec des cartes cachées que l'on "retourne" en cliquant dessus. On travaillera avec 16 cartes carrées. En fait on ne prend pas 16 objets cartes mais une grille de 16 cases qui seront les 16 couleurs "cachées".

Dans un premier temps, on dispose d'une grille de 16 couleurs .

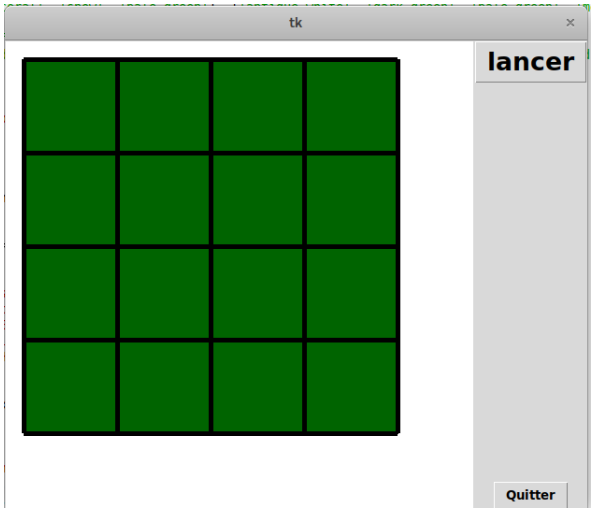
```
ref = [['snow', 'cornflower blue', 'turquoise', 'orchid1'],
       [ 'spring green',     'PaleGreen3',    'deep pink', 'pale violet red'],
       [ 'PeachPuff2', 'aquamarine2',  'LightGoldenrod1',  'gold2'],
       [  'purple3',    'orange1', 'pink4',     'LightPink1']]
```

1 présentation et premiers widgets

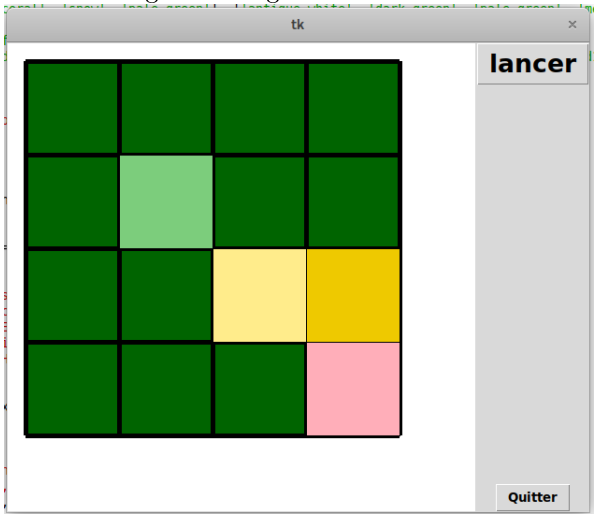
Au lancement du jeu, on a un Canvas et deux boutons "lancer" et "quitter".



Si on clique sur lancer, une "grille carrée" est dessinée avec des cases vertes, comme si les cases étaient retournées.



Ensuite si on clique sur une des cases un carré de la couleur contenue par cette case est dessiné. Cette couleur est obtenue grâce à la grille ref



1.1 mise en place

Mettre en place les widgets (dans un premier la fonction du bouton lancer ne fera rien).

1.2 fonction grilleVerte

Écrire la fonction grilleVerte qui dessine la grille verte.
Associer cette fonction au bouton lancer.

1.3 fonction cliquer

Écrire la fonction cliquer qui récupère le clic souris et affiche un carré de la bonne couleur au bon endroit ! Relier cette fonction au canvas.

2 jeu type simmons

2.1 présentation

Avant de coder, présentons un peu le "jeu".
On va programmer un jeu de mémoire. L'ordinateur affiche une case puis la cache puis le joueur doit faire de même. Si le joueur a bien répondu alors on passe à deux cases, puis 3 cases etc..... jusqu'à ce que le joueur se trompe ou arrive aux 16 cases.

On commencera par faire une version simple sans tests puis on jouera sur l'activation ou l'inactivation des boutons pour obliger le joueur à jouer "correctement".

On aura besoin de "forcer" le canvas à se mettre à jour. Ce sera avec `canvas.update()`
On fera une temporisation de l'affichage (sinon impossible de voir l'ordre d'affichage des cases par l'ordinateur).
On utilisera pour cela `canvas.after(500)` (pour une durée d'attente de 500).

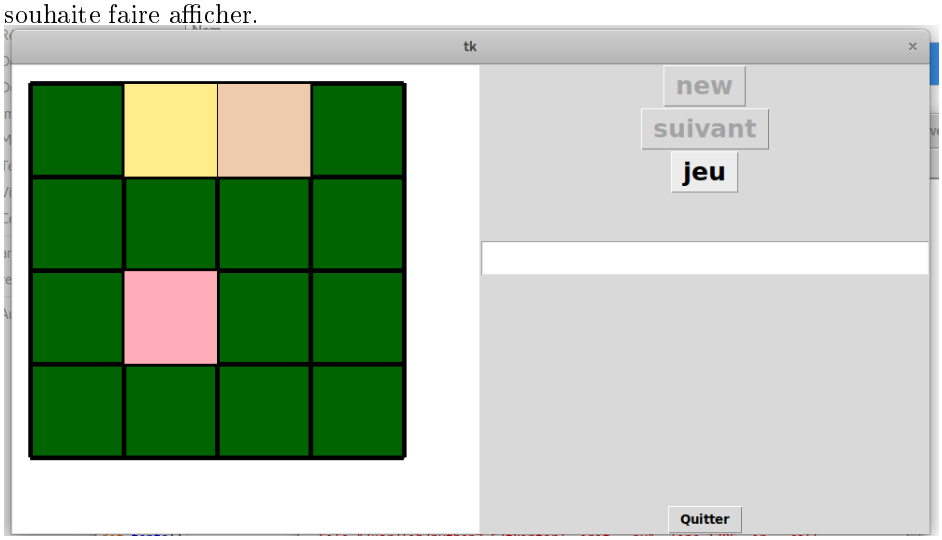
Dans un premier temps l'ordre dans lequel les cases s'affichent sera défini par une liste qui sera la liste listeCases.
On ne va donc pas jouer aléatoirement mais toujours avec les mêmes cases à retrouver dans le même ordre (plus tard on changera l'ordre et les couleurs).

```
listeCases=[[3, 1], [1, 2], [2, 1], [2, 2], [1, 0], [3, 2], [2, 0], [3, 0],
 [1, 1], [0, 2], [1, 3], [0, 1], [2, 3], [0, 0], [3, 3], [0, 3]]
```

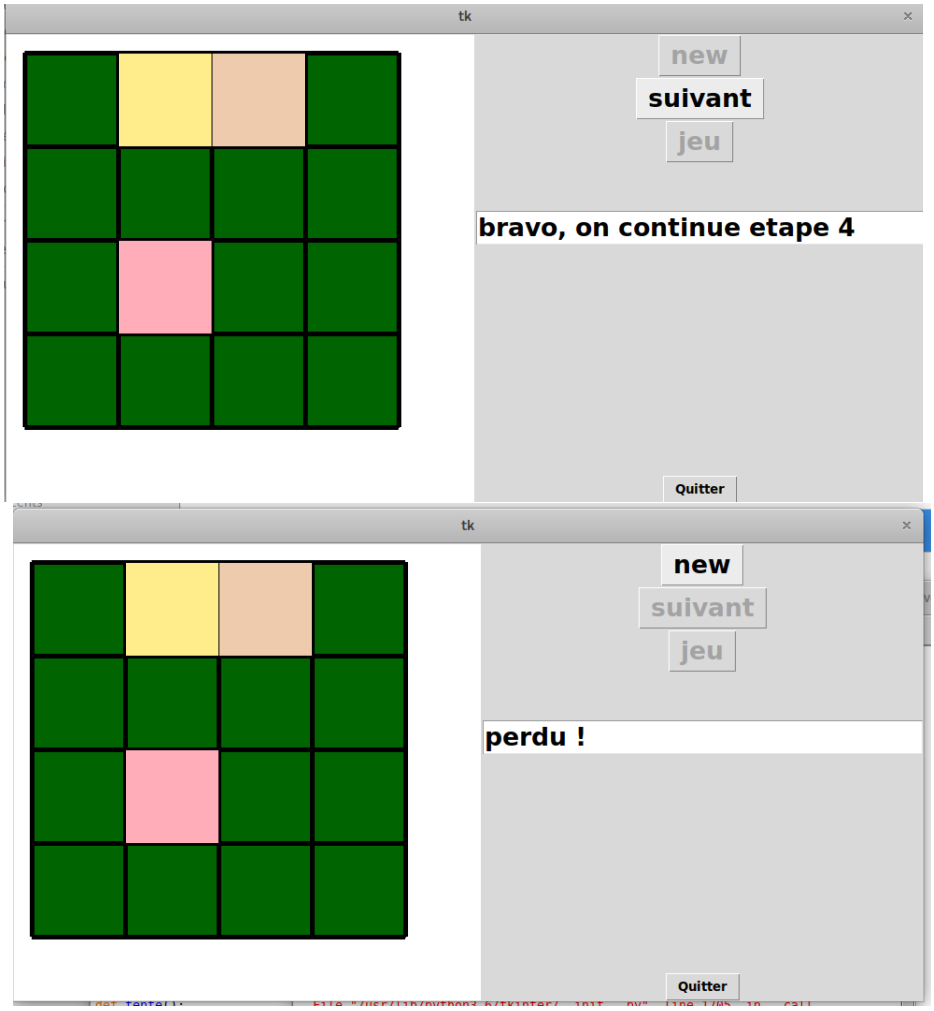
La grille ref et la liste listeCases sont dans le fichier tp3.py fourni.

Au départ, le joueur ne peut que cliquer sur le bouton new qui lance une nouvelle partie.

Une fois que la partie est initialisée, le joueur clique sur suivant pour que l'ordinateur lui affiche l'étape suivante, puis il clique sur jeu pour faire sa tentative, et pour cette tentative il clique successivement sur les cases qu'il souhaite faire afficher.



Quand il a cliqué le nombre de fois prévu (1 fois, puis 2 fois etc...) l'ordinateur affiche soit perdu, soit "btavo , on continue etape ..."



La partie s'arrête quand le joueur perd ou atteint la dernière étape !

2.2 widgets

Mettre en place les widgets nécessaires. On prévoira une zone d'affichage sous les boutons. On prendra pour les boutons en dehors du bouton quitter, 3 fonctions qui ne font "rien".

2.3 fonction des messages

Ecrire la fonction message(ch) qui gère l'affichage de ch dans la zone texte (il faut penser à la nettoyer avant d'écrire).

2.4 fonction d'affichage

Ecrire la fonction affiche(k) qui effectue l'affichage par l'ordinateur des k premières cases dans l'ordre (on utilisera can.after et can.update).

2.5 fonction du bouton suivant

A l'aide de la fonction précédente, écrire la fonction du bouton suivant. Cette fonction doit :
- effacer le canvas et la zone texte
- mettre à jour la valeur k
- dessiner la grille verte
- faire l'affichage de l'étape

2.6 fonction pour le joueur

Ecrire la fonction qu'on associera aux clics du joueur lors de sa tentative.
On aura une variable globale casesClickees qu'on remplira au fur et à mesure des clicks du joueur. A chaque click du joueur, on fera l'affichage correspondant et on ajoutera la nouvelle case cliquée aux précédentes. A chaque étape on vérifiera si le click est compatible avec la solution. Si ce n'est pas le cas on affichera que c'est perdu. Si le nombre de cases cliquées est égal à k (numéro de l'étape) alors on affichera que l'étape est réussie. Si on est arrivé à 16 que la partie est gagnée.

2.7 fonction du bouton jeu

A l'aide de la fonction précédente, écrire la fonction du bouton jeu : elle doit
- effacer le canvas , mettre à vide la liste casesClickees
- dessiner la grille verte
- récupérer les clicks et faire l'affichage au fur et à mesure

2.8 fonction nouvelle partie

Ecrire finalement la fonction du bouton new. Elle doit gérer toutes les initialisations et nettoyages...

On doit maintenant pouvoir jouer avec la liste des couleurs et des cases prévues et à condition de jouer correctement. On va maintenant améliorer cette base.

3 Améliorations

3.1 empêcher de jouer n'importe comment

Pour l'instant un joueur peut cliquer sur n'importe quel bouton à tout moment ce qui peut potentiellement fausser les résultats.
Rendre les boutons inactifs quand on ne doit pas s'en servir.

3.2 Aléatoires

3.2.1 les cases

Ecrire une fonction qui crée aléatoirement la liste des cases listeCases, puis l'ajouter dans la fonction du bouton new.

3.2.2 les couleurs

On voudrait que les couleurs ne soient pas forcément ni les mêmes ni au même endroit. Vous trouverez dans le fichier tp3.py une liste de couleurs python. Ecrire les fonctions de votre choix pour pouvoir jouer avec une grille ref différente à chaque fois.

4 Et ensuite ?

On peut ensuite avec cette base imaginer d'autres améliorations, taille de la grille, étapes..... ou d'autres jeux de mémoire !