

À votre racine créez un répertoire `Projet_Programmation_1`. Placez vous dans ce répertoire et téléchargez les fichiers mis à votre disposition à l'aide de la commande :

```
wget http://dept-info.labri.fr/ENSEIGNEMENT/projetprog1/src/fichiers-1.tar.gz
```

Décompressez l'archive fournie à l'aide de la commande : `tar xvzf fichiers-1.tar.gz`. Dans votre répertoire, vous disposez des fichiers : `exemple1.py`, `exemple2.py` ainsi que des images `fleur.gif`, `bateaux.pgm` et `feuille.pgm`.

Vous pourrez consulter la documentation sur : <http://www.pythonware.com/library/index.htm>

Interface graphique : premiers pas avec Tkinter

Étape 1 – Premières expérimentations

1. Chargez sous `Emacs` le fichier `exemple1.py` et exécutez le en évaluant le buffer.
2. Expérimentez les fonctionnalités du menu de décoration de la fenêtre fournies par votre *window manager* (agrandir, iconifier, ...).
3. Modifiez le message en "Hello World, don't panic !".
4. Chargez sous `Emacs` le fichier `exemple2.py` et exécutez le. Quelle est la condition pour obtenir l'affichage de la deuxième fenêtre ? Pourquoi ?

Étape 2 – Premier bouton

Dans `exemple1.py`, ajoutez les instructions nécessaires pour créer un bouton `quitter` qui provoquera la fermeture de la fenêtre. Le bouton sera placé en dessous du texte. Modifiez ensuite votre code pour que le bouton soit à droite du texte.

Étape 3 – Affichage d'une image

1. Ouvrez un nouveau fichier de nom `affiche_image.py`. Écrivez les instructions pour ouvrir une fenêtre contenant un `canevas` de taille 100×100 et un bouton `quitter` placé sous le canevas.
2. Affichez l'image `fleur.gif` dans le canvas.
3. Ajouter un bouton `effacer` pour enlever l'image de la fenêtre.
4. Comment adapter la taille du canevas à la taille de l'image ?

Étape 4 – Boutons pour sélectionner l'image

On veut pouvoir choisir entre deux images à afficher dans la fenêtre. Pour cela, on ajoutera deux boutons `Fleur` et `Feuille` pour sélectionner respectivement l'image `fleur.gif` et l'image `feuille.pgm`. Comment faire pour utiliser le même *callback*¹ pour l'affichage de l'une ou de l'autre image ?

1. i.e., la fonction qui est exécutée suite à un événement (comme par exemple un clic à la souris sur un bouton.)

Étape 5 – Création d'un menu

Modifiez votre code en remplaçant les boutons par un menu. Pour ne pas être limité à deux fichiers image, on pourra maintenant choisir le nom du fichier à ouvrir grâce à la fonction `tkFileDialog.askopenfilename()`.



Étape 6 – Dessin de cercles de couleur

1. Dans un fichier `cercle.py`, écrivez les instructions pour ouvrir une fenêtre contenant un bouton `quitter`, un bouton `cercle` qui provoquera l'affichage dans un canevas d'un cercle de rayon 10 et de couleur verte. On utilisera pour cela la fonction `create_oval`. On essaiera différentes positions pour le centre du cercle.
2. Modifiez la fonction d'affichage du cercle de façon qu'elle affiche des cercles de rayons croissants à chaque nouvel appel.
3. Ajoutez un bouton qui modifie aléatoirement la couleur du prochain cercle affiché.

Étape 7 – Clic souris dans un canevas

1. Dans un fichier `clic_cercle.py`, écrivez les instructions pour ouvrir une fenêtre contenant un bouton `quitter`, un canevas et telle qu'un clic de souris dans le canevas provoque l'affichage d'un cercle noir de rayon 10 centré à l'endroit du clic. On utilisera pour cela la fonction `bind` qui permet de lier un événement se produisant dans un *widget*² à l'exécution d'une fonction.
2. Ajoutez un bouton `reinitialisation` qui réinitialise le canevas en effaçant les cercles.

Étape 8 – Déplacement souris dans un canevas

Reprenez la dernière version de `affiche_image.py`. Ajoutez une zone de texte sous l'image. Modifiez le code pour y afficher les coordonnées de la souris lorsqu'elle se déplace sur le canevas.

Recherche dans la documentation : comment faire également afficher la couleur du pixel situé sous la souris (quand une image est affichée) ?

2. i.e., composant graphique comme par exemple un bouton, une case à cocher, une image, etc.)