# Challenge 2024-06-08: Vinos Ibericos

Table des matières

[Challenge 2024-05-31 : Classement par moyennes 1](#_Toc168519328)

[1) Enoncé 3](#_Toc168519329)

[2) Logigramme des modules 4](#_Toc168519330)

[a. Logigramme simple du code 4](#_Toc168519331)

[3) Variables du code 5](#_Toc168519332)

[4) Code 6](#_Toc168519333)

[b. constantes.py 6](#_Toc168519334)

[c. main.py 7](#_Toc168519335)

[1) Résulta **Erreur ! Signet non défini.**](#_Toc168519336)

## Enoncé

El AVE llega esta noche a Madrid y voy a embarcarme en la ruta de los vinos españoles... ¿Pero por dónde empezar? ¿Y si desarrollara mi propia aplicación...? Cette fois-ci le challenge va nous conduire dans l'univers des interfaces graphiques, à l'aide du module [Tkinter](https://docs.python.org/fr/3/library/tkinter.html). Il s'agira pour ce challenge d'une simple prise en main du module. Mais nous allons tout de même proposer une petite application sympathique qui fera appel à des modules tiers que chacun aura le loisir d'explorer et d'exploiter. Comme dit en introduction, nous allons donc profiter de ce challenge pour découvrir quelques [régions viticoles espagnoles](https://fr.wikipedia.org/wiki/Viticulture_en_Espagne), en développant une application graphique avec laquelle nous devrons cliquer sur des boutons portant les noms de ces régions viticoles, et dans notre widget central (carte de l'Espagne) apparaîtront des icônes situant exactement l'emplacement de ces régions. Comme vous l'aurez deviné, nous ferons appel aux [coordonnées GPS](https://fr.wikipedia.org/wiki/Global_Positioning_System).

**Étapes** :

1. Création de l'interface graphique (cf. modèles ci-dessous)
2. Utilisation des données (cf. tableau ci-dessous)
3. Positionnement sur la carte de nos icônes (marqueurs)
4. Utiliser des icônes en rapport avec le thème (cf. les deux fichiers images fournis ci-dessous)

**Conditions**

* **Utilisation des informations suivantes (oui, nous sommes sympas, nous vous fournissons les coordonnées GPS)**:

DO\_VINOS = {

"Alicante": ((38.3436365, -0.4881708), "Tinto"),

"Calatayud": ((41.3527628, -1.6422977), "Tinto"),

"Cariñena": ((41.3382122, -1.2263149), "Tinto"),

"Condado de Huelva": ((37.3382055, -6.5384658), "Blanco"),

"Jumilla": ((38.4735408, -1.3285417), "Tinto"),

"La Gomera": ((28.116, -17.248), "Blanco"),

"Málaga": ((36.7213028, -4.4216366), "Blanco"),

"Rías Baixas": ((42.459627886165265, -8.722862824636783), "Blanco"),

"Ribera del Duero": ((41.49232, -3.005), "Tinto"),

"Rioja": ((42.29993373411561, -2.486288477690506), "Tinto"),

"Rueda": ((41.4129785, -4.9597533), "Blanco"),

"Somontano": ((42.0883878, 0.0994041), "Tinto"),

"Tarragona": ((41.1172364, 1.2546057), "Tinto"),

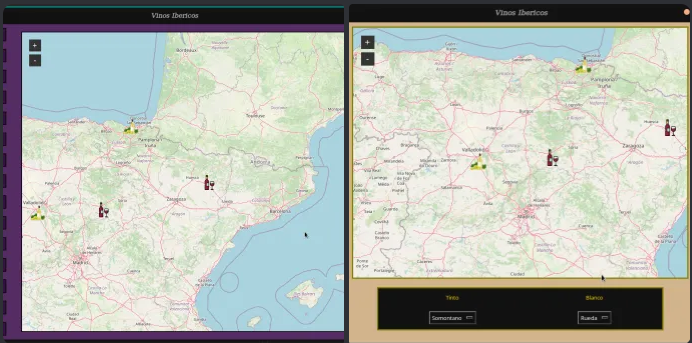
"Txakoli de Getaria": ((43.29428414467608, -2.202397625912913), "Blanco"),

"Xérès": ((36.6816936, -6.1377402), "Blanco")

}

* **Création de l'interface graphique**:

Une des interfaces utilise des boutons cliquables et l'autre des menus déroulants. Il n'y a pas de modèle imposé, à vous de choisir celui qui vous convient, comme pour les couleurs : c'est votre interface.



 **Les icônes**:

Nous utilisons deux fichiers images que nous convertissons en icônes. Un représente une bouteille de vin blanc et l’autre une bouteille de vin rouge. En fait cette distinction apporte un défi supplémentaire puisqu'en fonction de la région (précision avec les données fournies : tinto = rouge, blanco =), l’icône qui s’affiche va représenter l’une ou l’autre des deux images.





**Et s'il devait y avoir un bonus...**

Libre à chacun d’exploiter les possibilités offertes par les divers modules utilisés… et là c’est sans limite. Proposez nous vos belles créations...

## Logigramme des modules

### Logigramme simple du code

## Variables du code

|  |  |  |  |
| --- | --- | --- | --- |
| Modules | Nom de la variable | Type | Commentaires |
| Constantes | DO\_VINOS | Dict | Données d’entrées sous forme de dictionnaire. Chaque clé possède une liste avec les coordonnées et type de vin |
| CURRENT\_PATH | Path | Contient la varibale du chemin du projet |
|  |  |  |  |
|  |  |  |  |

## Code

### constantes.py

# constantes.py  
# But:  
# Contient les constantes du code  
# -----------------------------------  
# Date de création: 2024-06-08  
# Date de dernière modification: 2024-06-16  
# ------------------------------------------  
# version: 1.0  
# -  
#-------------------------------------------  
import os  
  
DO\_VINOS = {  
 "Alicante": ((**38.3436365,** -**0.4881708**)**,** "Tinto")**,** "Calatayud": ((**41.3527628,** -**1.6422977**)**,** "Tinto")**,** "Cariñena": ((**41.3382122,** -**1.2263149**)**,** "Tinto")**,** "Condado de Huelva": ((**37.3382055,** -**6.5384658**)**,** "Blanco")**,** "Jumilla": ((**38.4735408,** -**1.3285417**)**,** "Tinto")**,** "La Gomera": ((**28.116,** -**17.248**)**,** "Blanco")**,** "Málaga": ((**36.7213028,** -**4.4216366**)**,** "Blanco")**,** "Rías Baixas": ((**42.459627886165265,** -**8.722862824636783**)**,** "Blanco")**,** "Ribera del Duero": ((**41.49232,** -**3.005**)**,** "Tinto")**,** "Rioja": ((**42.29993373411561,** -**2.486288477690506**)**,** "Tinto")**,** "Rueda": ((**41.4129785,** -**4.9597533**)**,** "Blanco")**,** "Somontano": ((**42.0883878, 0.0994041**)**,** "Tinto")**,** "Tarragona": ((**41.1172364, 1.2546057**)**,** "Tinto")**,** "Txakoli de Getaria": ((**43.29428414467608,** -**2.202397625912913**)**,** "Blanco")**,** "Xérès": ((**36.6816936,** -**6.1377402**)**,** "Blanco")  
}  
  
CURRENT\_PATH = os.path.join(os.path.dirname(os.path.abspath(\_\_file\_\_))) # Variable pour le répertoire de travail

### window.py

# window.py  
# But:  
# Contient le code du challenge  
# -----------------------------------  
# Date de création: 2024-06-08  
# Date de dernière modification: 2024-06-16  
# ------------------------------------------  
# version: 1.0  
# -  
#-------------------------------------------  
# Appel des modules externes  
from tkinter import ttk**,** Tk  
import tkintermapview  
import os  
from PIL import Image**,** ImageTk  
  
# Appel des modules internes  
from constantes import DO\_VINOS**,** CURRENT\_PATH

class MainWindow(Tk):  
 *"""  
 Classe MainWindow:  
 Permet de définir toutes les caractéristiques de la fenêtre d'application  
 Hérite de la classe Tk  
  
 3 méthodes:  
 - set\_marker: Pour la gestion des marqueurs sur la carte  
 - set\_coordonnees: Pour gérer le centre de la carte avec la ville sélectionnée  
 - set\_button: Pour gérer la création des boutons* ***:param*** *- Tk: Permet à MainWindow d'hériter de la classe Tk  
 """* # Variables de la classe  
 POS\_ROW = **10** POS\_COLUMN = **25** WIDTH = **130** def \_\_init\_\_(self):  
 super().\_\_init\_\_()  
 # Windows options  
 self.title(f"VINOS IBEROS")  
 self.configure(bg="purple")  
 self.geometry('1000x700')  
  
 # Coordonnées GPS de Madrid (par défaut)  
 self.POS\_X = **40.427** self.POS\_Y = -**3.738** # create map widget  
 self.map\_widget = tkintermapview.TkinterMapView(self**,** width=**745,** height=**650,** corner\_radius=**0**)  
 # set current widget position and zoom  
 self.map\_widget.set\_position(self.POS\_X**,** self.POS\_Y) # Centrer sur Madrid  
 self.map\_widget.set\_zoom(**7**)  
 self.map\_widget.place(x=**230,** y=**25**)

# Style options  
 self.style = ttk.Style(self)  
 # Configure le style des boutons  
 self.style.configure('TButton'**,** font=('Helvitica'**, 10**)**,** bd=**10,** foreground="#FFCC00"**,** bg="#FFFF33"**,** activebackground="blue"**,** padding=**5**)  
  
  
 def set\_marker(self):  
 *"""  
 Méthode pour afficher le type de vin en fonction de la ville* ***:return****: Images (vin blanc ou rouge) sur les coordonnées GPS de la ville en fonction des valeurs du dictionnaires  
 """* blanco = ImageTk.PhotoImage(Image.open(os.path.join(CURRENT\_PATH**,** "images"**,** "blanco.webp")).resize((**50, 50**)))  
 tinto = ImageTk.PhotoImage(Image.open(os.path.join(CURRENT\_PATH**,** "images"**,** "tinto.webp")).resize((**50, 50**)))  
 for i in list(DO\_VINOS):  
 if DO\_VINOS[i][**1**].lower() == "tinto":  
 self.map\_widget.set\_marker(DO\_VINOS[i][**0**][**0**]**,** DO\_VINOS[i][**0**][**1**]**,** icon=tinto) # change position  
 elif DO\_VINOS[i][**1**].lower() == "blanco":  
 self.map\_widget.set\_marker(DO\_VINOS[i][**0**][**0**]**,** DO\_VINOS[i][**0**][**1**]**,** icon=blanco) # change position  
 def set\_coordonnees(self**,** i):  
 *"""* ***:param*** *i: Récupère le nom de la ville de la méthode set\_button* ***:return****: Permet de centrer la carte sur la ville sélectionnée  
 """* self.POS\_X = DO\_VINOS[i][**0**][**0**]  
 self.POS\_Y = DO\_VINOS[i][**0**][**1**]  
 self.map\_widget.set\_position(self.POS\_X**,** self.POS\_Y) # Centrer sur la ville sélectionnée  
 self.map\_widget.set\_zoom(**8**) # Permet de faire un zoom sur la ville sélectionnée  
  
 def set\_button(self):  
 *"""  
 Méthode pour la création des boutons* ***:return****: Créer tout les boutons du dictionnaire.  
 Action des boutons: Centrer la carte sur la ville sélectionnée  
 """* for i in list(DO\_VINOS):  
 # Création de tout les boutons pour quitter la fenêtre avec position dans la grille  
 self.button = ttk.Button(self**,** text=i**,** command=lambda i=i:self.set\_coordonnees(i))  
 self.button.place(x=self.POS\_COLUMN**,** y=self.POS\_ROW**,** width=self.WIDTH**,** height=**40**)  
 self.POS\_ROW += **45**

if \_\_name\_\_ == "\_\_main\_\_":  
 main\_window = MainWindow() # Créer l'instance main\_window  
 main\_window.set\_marker() # Appel de la méthode pout afficher le type de vin en fonction de la ville  
 main\_window.set\_button() # Appel de la méthode pour créer et gérer les boutons  
 main\_window.mainloop() # Affiche la fenêtre

## Détails du code

*A écrire !!*

## Résultat

