

PPRI0301 – Analyse de Données en Python

TP – Interfaces Graphiques

Dans ce TP Vous devez réaliser trois applications (interface desktop TkInter, micro-app web avec Flask et dashboard Streamlit), chacune exploitant le dataset "Palmer Archipelago penguin data" (trois espèces de manchots, avec diverses mesures biologiques). Vous manipulerez ce dataset avec Pandas, puis construirez des interfaces permettant d'explorer, filtrer, et visualiser ces données.

Le dataset est fourni sous forme CSV, et disponible à l'adresse suivante :

```
https://raw.githubusercontent.com/allisonhorst/palmerpenguins/master/inst/extdata/penguins\_raw.csv
```

Comme ces trois applications partagent des fonctionnalités, vous devez créer également un « cœur logiciel commun » implémenté sous forme d'une classe mère Python utilisant Pandas et matplotlib. Ce cœur commun permettra la lecture des données et la génération des graphiques.

Objectifs pédagogiques

- Appliquer les concepts de programmation orientée objet (POO) pour réaliser une classe commune
- Manipuler des données via Pandas chargées directement à partir d'une URL
- Construire des interfaces graphiques dans trois environnements différents
- Implémenter une gestion commune des données et des visualisations dans la classe partagée
- Développer progressivement les applications avec une organisation modulaire

Partie 1 : Création de la classe centrale

Créez une classe PenguinData regroupant les fonctionnalités suivantes :

- Chargement des données dans un DataFrame Pandas depuis l'URL : https://raw.githubusercontent.com/allisonhorst/palmerpenguins/master/inst/extdata/penguins_raw.csv
- Si nécessaire, nettoyer le dataset et sélectionner les colonnes d'intérêt
- Méthodes pour appliquer des filtres simples (ex : filtre par espèce, sexe, île, taille minimum du bec).
- Méthodes retournant des statistiques descriptives (describe(), etc.).
- Méthodes pour la génération de graphiques matplotlib (ex. histogramme de la masse selon espèce).

Attention : Cette classe ne contient aucune notion d'interface graphique, elle est indépendante.

Partie 2 : TkInter

- Utilisez la classe *PenguinData* pour charger et manipuler les données.
- Ajoutez une fenêtre TkInter avec des widgets pour contrôler les filtres sur le DataFrame.
- Affichez les résultats filtrés dans la fenêtre via un widget texte ou tableau.
- Ajoutez des boutons permettant de générer un graphique en appelant les méthodes de la classe qui retournent des graphiques matplotlib. Essayez différents modes (RadioButton, menus) pour rendre l'interface accueillante et pratique.
- Affichez les graphiques dans une fenêtre TkInter à part ou intégrée dans la même fenêtre grâce à un *Canvas*.

Partie 3 : Flask

- Créer le template HTML et le code Python dans deux fichiers séparés, cf. le cours.
- Chargez les données dans la classe *PenguinData* directement depuis l'URL à chaque lancement de l'application.
- Affichez un tableau HTML des statistiques descriptives.
- Affichez un tableau HTML du DataFrame filtré par espèce (via un *dropdown*, avec le filtre appliqué dans la classe).
- Faites afficher des graphiques matplotlib du poids des manchots selon filtre, avec rendu en PNG dans la page HTML.

La logique de données et filtrage doit être gérée via la classe *PenguinData*.

Partie 4 : Streamlit

- Utilisez la classe *PenguinData* pour charger et filtrer les données.
- Ajoutez des widgets Streamlit pour filtrer interactivement les données (sexe, espèce, taille).
- Affichez le DataFrame filtré avec `st.dataframe`.
- Affichez des graphiques dynamiques (histogramme, camembert, etc.) en appelant la/les méthode(s) graphique(s) de la classe.

Toute manipulation de données et visualisation passe par la classe commune.

Partie 5 : Bonus

- Ajoutez des filtres avancés dans la classe et utilisez-les dans les trois applications.
- Permettez l'export en CSV du DataFrame filtré dans chaque UI.
- Améliorez la présentation (titres, couleurs, etc.). Avec Streamlit, pensez à utiliser *Sidebar* et des *Selectbox*.
- Documentez le cœur logiciel et chaque interface.

Consignes pour le rendu :

1. Créer un répertoire par application.
2. Rajouter également un petit fichier (texte ou markdown) avec les consignes pour le lancement de vos applications.
3. Déposer un seul fichier zip dans la zone dédiée sur Moodle.