Olym.py

LECARME Gabriel
GUIRARDEL Colin

https://github.com/cguirardel/PAI-projet

08/02/2024

Présentation du projet :
Plan

- Présentation du projet
- Principales spécifications
- Conception
- Tests
- Conclusion

Présentation du projet

Objectif : Traiter et mettre en forme une base de données sur les Jeux Olympiques avec Python

Réalisé dans le cadre du cours Programmation pour les Activités de l'Ingénieur, IOGS 3A

Décembre 2023 – Février 2024

Présentation du projet :
Bibliothèques requises

- NumPy
- Matplotlib
- pandas
- PyQt5
- superqt
- GeoPandas

Installation superqt :
conda install conda-forge::superqt

Principales spécifications

Groupe 1 : Géographie	**Ex 1.1 : Afficher une carte du monde avec des médailles par pays**	✓
	Ex 1.2 : Afficher un tableau de classement des médailles (Or, argent, bronze, par pays)	✓
	Ex 1.1.2 et 1.2.2 : (Optionnel) Grouper par région du monde pour les deux exigences précédentes	✗
Groupe 2 : Athlètes	**Ex 2.1 : Pouvoir extraire et afficher les informations d'un athlète donné en regroupant les différentes occurrences d'un athlète**	✗
	Ex 2.2 : Pouvoir rechercher un athlète par sport/pays/nom.	✓
	Ex 2.2.2 : Usage de la syntaxe wildcards	~ (Syntaxe RegEx)
Groupe 3 : Médailles	**Ex 3.1 : Afficher un histogramme/courbe nombre de médailles vs PIB du pays à l'époque des jeux**	✗
	Ex 3.2 : Afficher pour un sport donné l'histogramme/courbe nombre de médailles sur âge	✓
	Ex 3.2.2 : Superposer ces histogrammes pour différents sports	✓
Groupe 4 : Temps	**Ex 4.1 : Sélectionner une plage temporelle (début et fin) à l'aide d'un double slider pour y restreindre les statistiques précédentes**	✓
	Ex 4.2 : Sélectionner les jeux d'été et/ou d'hiver	✓

Principales spécifications

Groupe 1 : Géographie	**Ex 1.1 : Afficher une carte du monde avec des médailles par pays**	✓
	Ex 1.2 : Afficher un tableau de classement des médailles (Or, argent, bronze, par pays)	✓
	Ex 1.1.2 et 1.2.2 : (Optionnel) Grouper par région du monde pour les deux exigences précédentes	✗

Ex 1.1 : ajout d'une colonne « médailles » à un geodataframe
Ex 1.2 : usage d'un TableModel et fonction compteMedailles
Ex 1.1.2 et 1.2.2 : manque de temps

Principales spécifications

Ex 2.1 : extraction Wikipedia
Ex 2.2 : usage de pandas.Series.str.contains
Ex 2.2.2 : syntaxe regex déjà encodée dans pandas.Series.str.contains

Groupe 2 : Athlètes	**Ex 2.1 : Pouvoir extraire et afficher les informations d'un athlète donné en regroupant les différentes occurrences d'un athlète**	✗
	Ex 2.2 : Pouvoir rechercher un athlète par sport/pays/nom.	✓
	Ex 2.2.2 : Usage de la syntaxe wildcards	~

Principales spécifications

Ex 3.1 : pas trouvé de BDD sur les PIBs en fonction des époques
Ex 3.2 : Matplotlib + fonction compteMedailles
Ex 3.2.2 : facile avec Matplotlib

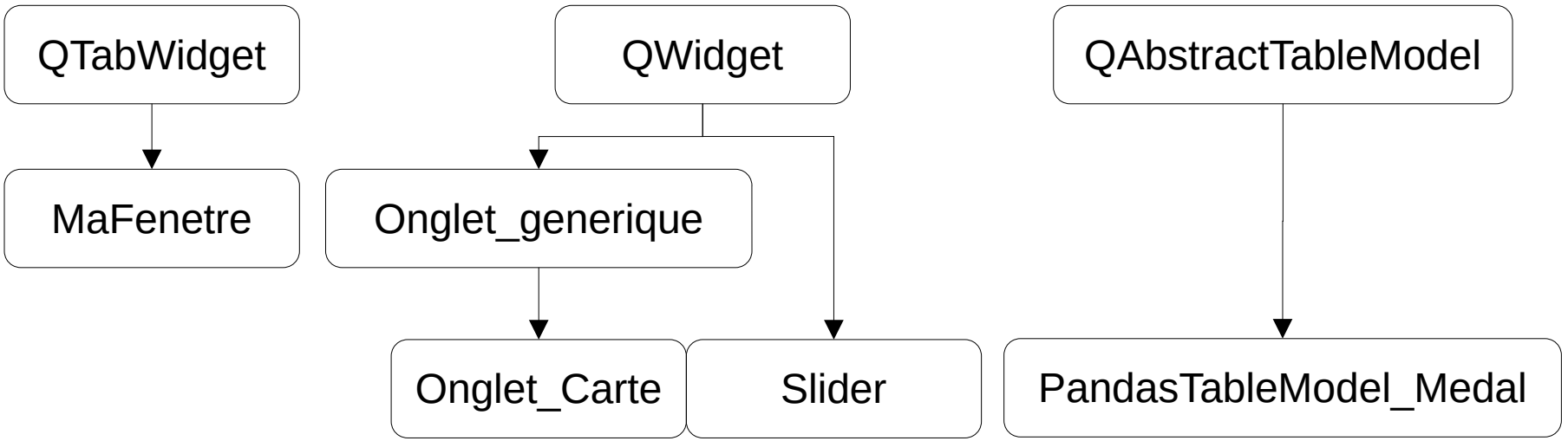
Groupe 3 : Médailles	**Ex 3.1 : Afficher un histogramme/courbe nombre de médailles vs PIB du pays à l'époque des jeux**	✗
	Ex 3.2 : Afficher pour un sport donné l'histogramme/courbe nombre de médailles sur âge	✓
	Ex 3.2.2 : Superposer ces histogrammes pour différents sports	✓

Principales spécifications

Ex 4.1 : double slider défini dans superqt
Ex 4.2 : facile avec PyQt5

Groupe 4 : Temps	**Ex 4.1 : Sélectionner une plage temporelle (début et fin) à l'aide d'un double slider pour y restreindre les statistiques précédentes**	✓
	Ex 4.2 : Sélectionner les jeux d'été et/ou d'hiver	✓

Conception :
Diagramme des classes



Conception :

Description des classes principales

- Classe MaFenetre :
  - Titre, icône
  - Instancie les différents onglets
- Modèles de tableaux :
  - Pour remplir et actualiser facilement les tableaux à partir de dataframes

Conception :
Description des classes principales

- Classe Onglet_generique :
  - Structure de base pour les autres onglets
  - Instancie le Slider
- Classe Slider :
  - Le double slider, avec l'affichage des dates et le choix des saisons
  - Méthodes :
    - update_label(self) : met à jour l'affichage des dates choisies

Conception :

Description des classes principales

- Classe Ong_Carte :
  - Carte, tableau des médailles
  - Méthodes :
    - _update(self) : appelle les deux méthodes suivantes
    - _update_table(self) : met à jour l’affichage du tableau
    - _update_map(self) : met à jour l’affichage de la carte

Conception :

Description des classes principales

- Classe Ong_Rech :
  - Zones de texte, bouton recherche, tableau de résultats
  - Méthodes :
    - search : filtre la base de données puis mets à jour le tableau de resultats

Conception :
Réunion de Bases de données

noc_region		
noc	region	notes
Code NOC	Nom du pays	Utile pour les ambiguïtés

???

NOC

countries		
geometry	name	ADM0
Géographie du pays	Nom du pays	Code du pays
La fonction d'affichage utilise ce tableau pour tracer la carte		

olympics			
athlete	NOC	medal	...
La fonction compteMedailles utilise ce tableau et agrège par pays (ou autre)			

Conception :
Réunion de Bases de données

- Construction d’un tableau de correspondance NOC <=> ADM0 :
- On parcourt la liste des NOCs
  - Correspondance directe quand NOC = ADM0
  - Correspondance quand region = countries
  - Correspondance à la main pour les ~50 NOCs restants
- Si le NOC ne correspond pas à un pays représenté sur la carte, on le met en correspondance avec la valeur NaN
- Si plusieurs NOCs correspondent à un même pays, on somme les nombres de médailles

Tests

- Programme test.py
- Exigence 1.1 – Test de la fonction constructionCarte
- Exigence 1.1 – Test de la fonction compteMedailles
- Exigence 2.2 – Extraction d’un tableau filtré
- Exigence 3.2 – Extraction du tableau médailles vs age

Conclusion :
Enjeux et résultats

- Un apprentissage de Pandas et PyQt
- Usage de classes
- Fusion de BDDs différentes
- Gestion des versions : Git
- BDD incomplète → tests difficiles
- Un résultat global satisfaisant

Conclusion :
Remerciements

Les développeurs souhaitent remercier Oriane Koellsch pour le dessin du logo

Bases de données utilisées :

Jeux Olympiques : https://www.kaggle.com/datasets/bhanupratapbiswas/olympic-data

Carte du monde : http//www.naturalearthdata.com/download/110m/cultural/ne_110m_admin_0_countries.zip

Merci à celles et ceux qui développent maintiennent les bases de données et bibliothèques !