Polytech Paris-Saclay

Rapport de projet - Global Warming 3D

23 juin 2020 - par **Antonin D.**

Objectifs du projet

Ce projet s'inscrit dans le module d'Interaction Homme Machine en 1ère année de cycle ingénieur à Polytech Paris-Saclay en spécialité Informatique.

Le but est de réaliser une application permettant de visualiser les anomalies de température (écarts de température aux températures « normales » calculées par la NASA) sur un globe terrestre en 3D.

Les technologies utilisées sont Java (pour la partie applicative) et JavaFX (pour l'interface).

J'ai fait le choix d'utiliser le Kit de Développement Java 8, l'Environnement de Développement Intégré Netbeans 8.2 ainsi que le logiciel SceneBuilder (afin d'éditer le fichier .fxml servant à décrire l'interface utilisateur).

Description fonctionnelle

La partie applicative du projet doit répondre à un certain nombre de fonctionnalités :

Caractéristiques de l'interface :

- visualiser en 3D sur le globe les anomalies de température
  - proposer 2 modes de visualisation : zones de couleur ou histogrammes
- sélectionner l'année à afficher et animer l'évolution
- afficher une légende
- sélectionner une zone du globe
- afficher graphique évolution

Fonctionnalités de recherche :

- récupérer chaque valeur des anomalies de température
- récupérer les valeurs pour toutes les zones.
- récupérer les valeurs pour toutes les années pour une zone donnée.

Récupérer, parser et stocker les données contenues dans un fichier .csv

lat	lon	1880	1881	1882	... (3)	2018	2019	2020
-88	-178	NA	NA	NA	...	1.1591666	0.7566666	0.5124999
-88	-174	NA	NA	NA	...	1.1591666	0.7566666	0.5124999
-88	-170	NA	NA	NA	...	1.1591666	0.7566666	0.5124999
...	...	...	...	...		...	...	...
88 (2)	170	NA	NA	NA	...	3.0783332	2.8291666	3.9999999
88	174	NA	NA	NA	...	3.0783332	2.8291666	3.9999999
88	178	NA	NA	NA	... (1)	3.0783332	2.8291666	3.9999999

Illustration du formatage des données (sous forme de tableau) contenues dans le fichier *tempanomaly_4x4grid.csv*

Description technique

Afin de réaliser le plus efficacement les opérations de recherche, il a fallu déterminer la structure de données appropriée pour (cette utilisation | stocker les anomalies).

Après avoir envisagé une ArrayList (collection ordonnée et ... mais trop coûteuse en recherche), la structure de données définitive a adoptée une double LinkedHashMap pour permettre des chargements performants et dans le but de conserver l'ordre du .csv pour "facilement afficher" les graphiques plus tard.

Pour implémenter les fonctions de recherche, trois méthodes ont été ajoutées à la classe *CoordAnomaliesMap* (héritant de LinkedHashMap) pour retourner les (1), (2), (3) illustrés par le tableau précédent.

Diagramme(s) UML

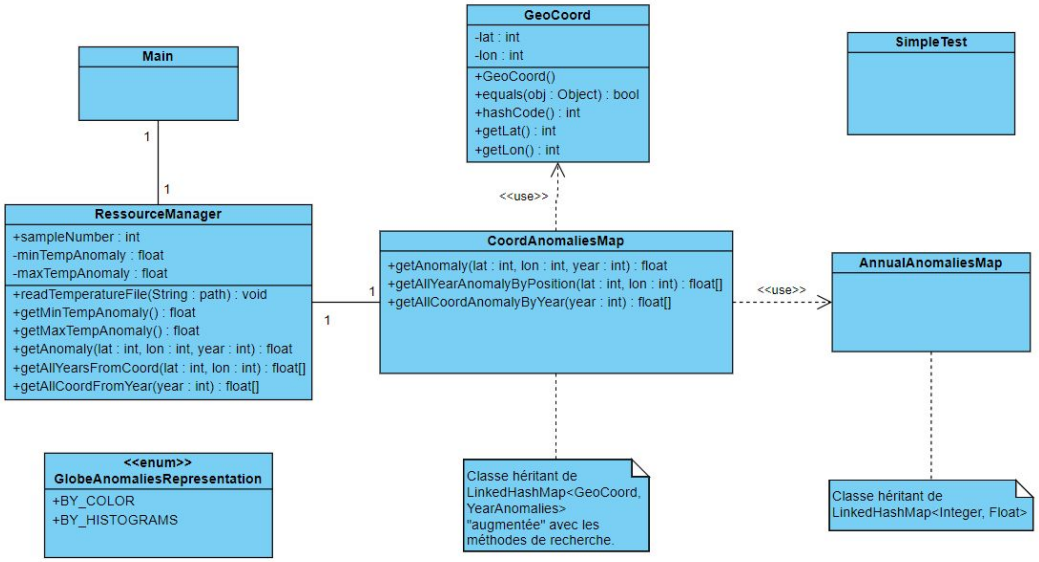


Diagramme de classe final du projet

Autre Image ici ??

De plus, des tests unitaires ont permis de valider le bon fonctionnement de la partie applicative.

Pour découvrir l'implémentation plus en détail, le code source de l'application est disponible à l'adresse suivante :

Github - Code source du projet

Gestion de la 3D

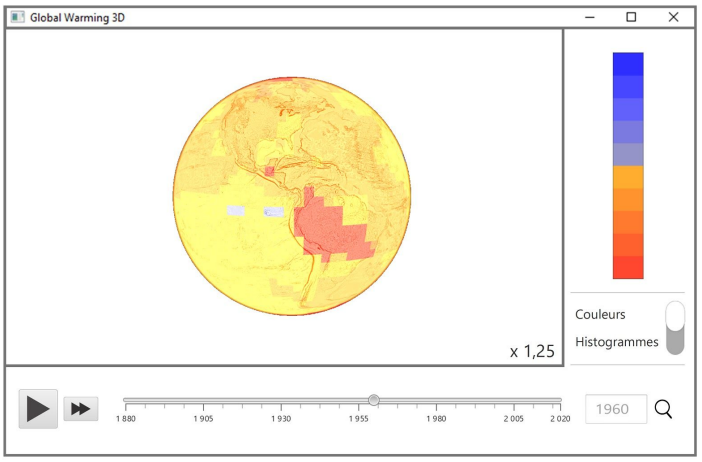
Il a été nécessaire d'inclure une librairie externe (*jimObjModelImporterJFX*) pour charger le modèle 3D d'une sphère texturée (la planète Terre) fournie au format WaveFront OBJ dans la scène 3D.

La classe *CameraManager*, permettant de gérer de déplacement de la caméra au clavier et à la souris, a également été fournie.

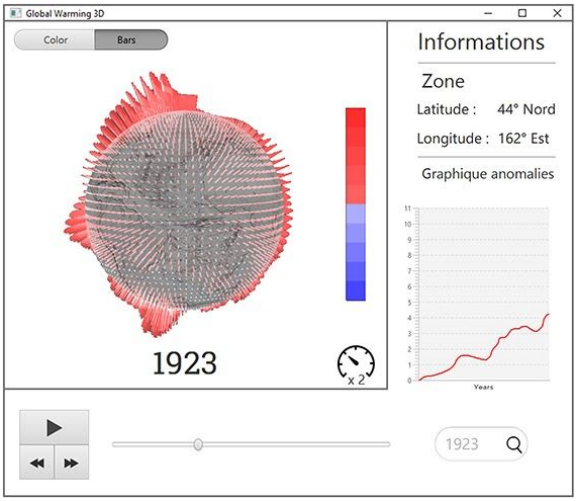
Interface et prototypes

Visuellement, l'interface est "découpée" en deux : une partie 2D (pour faire ci) et un espace 3D (pour faire ça).

Dans un premier temps, deux premières versions ont été réalisées de la maquette au prototype, philosophies similaires...



Prototype de l'interface dans sa première version



Prototype de l'interface dans sa deuxième version

Pour / Contre des v1 et v2 :

	Pour	Contre
Version 1		Pas d'infos sur la zone sélectionnée (aurait pu être indiqué au survol dans une infobulle + graphique au clic dans une autre fenêtre)
Version 2	Année courante déportée sur 3d	Un peu surchargé (indicateur de vitesse notamment) Résolution "trop carrée" Slider étroit et pas indiqué

Après "expérimentation" avec utilisateur "cobaye", il a perçu les btn >> et << comme "avancer" et "reculer" dans le temps et pas "augmenter/diminuer" la vitesse de défilement.

Pour la v3, << et >> ont donc été modifiés en - et +, plus explicite en terme de vitesse. L'indicateur de vitesse associé a été simplifié (par rapport à la v2) et déplacé entre - et + (car là où il était (bas droite 3d) pas visible).

Comment j'ai appliqué les principes IHM vus en classe ??

Prototype final de l'interface :