Physic Sandbox

Documentation technique

Sommaire :

I. Structure du code

Vous retrouverez la liste des fichiers du projet, ainsi que leurs rôles spécifiés.

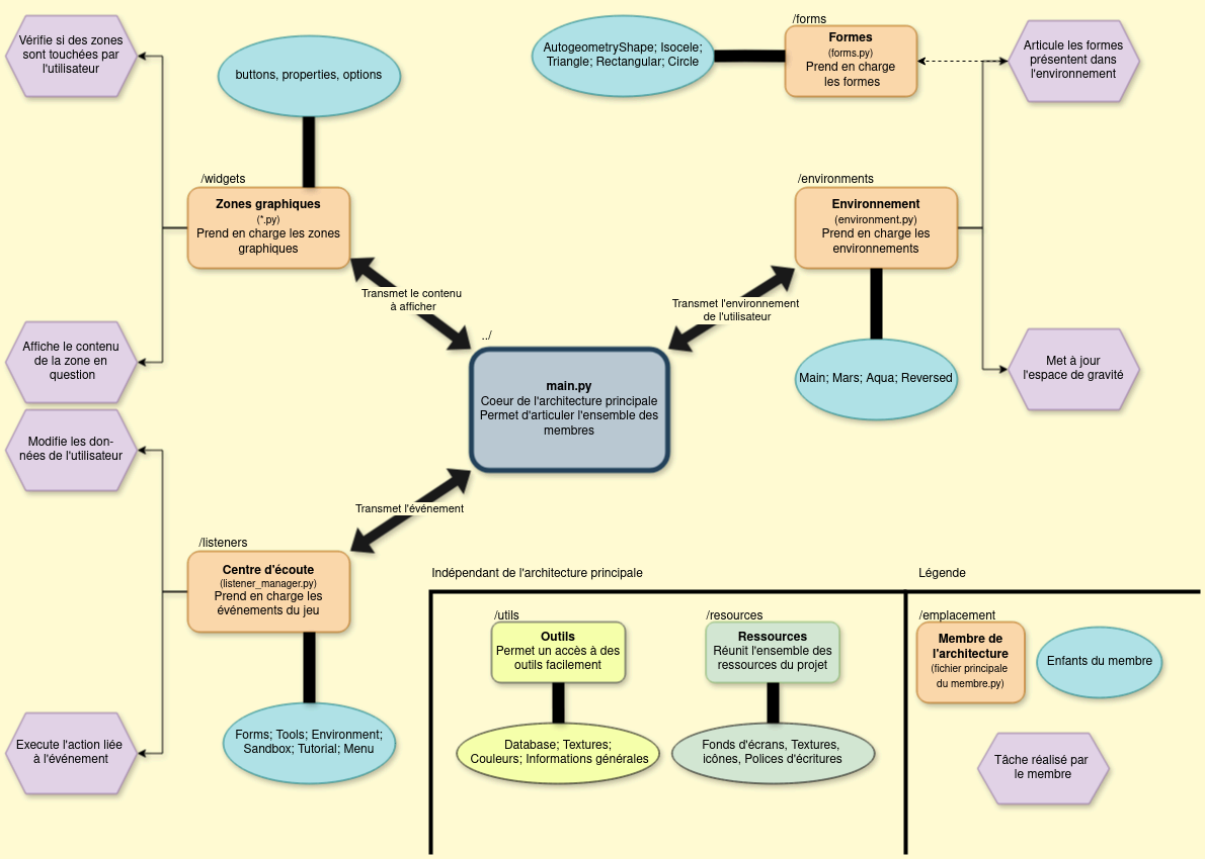
- *main.py* → `Classe principale du programme, elle permet l'articulation de l'ensemble des classes.`

❖ widgets
  ❖ buttons
    - *environnement_button.py* → `Permet d'afficher un bouton-environnement`
    - *form_button.py* → `Permet l'affichage d'un bouton-figure`
    - *menu_button.py* → `Permet d'afficher un bouton du menu`
    - tool_button.py → `Permet l'affichage d'un bouton-outil`
  ❖ sub_widgets
    - *forms_options_widget.py* → `Permet l'affichage de la boîte de dialogue des différentes actions possibles sur une figure`

        - *properties_widget.py* → Permet l'affiche des propriétés d'une figure
    - *environments_widget.py* → Permet l'affichage de la zone des environnements
    - *forms_widget.py* → Permet l'affichage de la zone des figures
    - *menu_widget.py* → Permet l'affichage du Menu
    - *tools_widget.py* → Permet l'affichage de la "boite à outils"
    - *tutorial_widget.py* → Permet l'affichage du tutoriel.
- ❖ listeners
    - *environment_surface_listener.py* → Dirige les événements liés à la surface des environnements
    - *forms_surface_listener.py* → Dirige les événements liés à la surface des formes
    - *listeners_manager.py* → Centre d'écoute des événements, les dispatche aux centres d'écoutes inférieurs
    - *menu_listener.py* → Dirige les événements liés au menu
    - *sandbox_listener.py* → *D*irige les événements liés à la surface de jeu (espace physique)
    - *tools_surface_listener.py* → Dirige les événements liés à la boîte à outils
    - *tutorial_listener.py* → Dirige le tutoriel en fonction des événements réalisé
- ❖ environments
    - *environment.py* → Classe héréditaire qui permet de définir un environnement, son espace de gravité ainsi que de nombreuses propriétés
    - *main_environment.py* → Environnement principal, gravité de la terre
    - *mars_environment.py* → Environnement avec la gravité de Mars
    - *reversed_environment.py* → Environnement avec la gravité inversée
    - *water_environment.py* → Environnement représentant l'eau
- ❖ forms
    - *autogeometry_shape.py* → Polynôme pymunk généré à partir de n'importe quelle image
    - *circle.py* → Cercle pymunk
    - *forms.py* → Polynôme pymunk, utilisable dans n'importe quel environnement
    - *isocele.py* → Polyn*ôm*e pymunk de type triangle isocèle
    - *rectangular.py* → Polynôme pymunk de type rectangle
    - *triangle.py* → Polynôme pymunk de type triangle rectangle
- ❖ utils
    - *colors.py* → Contient les codes RGB d'un certain nombre de couleurs
    - *database.py* → Permet de créer et diriger une base de données SQL
    - *texture.py* → Classe qui répertorie l'intégralité des images du projet
    - *utils.py* → Classe qui contient de nombreuses informations générales du projet

II. Architecture du projet

Notre projet repose sur une architecture qui consiste à articuler quelques membres ayant des tâches spécifiques à réaliser autour d'un membre principal, qui fait office de carrefour en permettant aux données de circuler n'importe où. De plus, deux modules sont en dehors de l'articulation, ils ont l'avantage d'être accessibles par n'importe quel membre de manière directe, mais ne fournissent uniquement des informations, et ne réalisent aucune action.



Ce schéma peut se retrouver dans le document "schema_architecture.pdf" pour une meilleure lecture.

III. Dépendances

Physic Sandbox ne peut fonctionner sans les modules Pygame, Pymunk, ainsi que Tkinter. Nous vous invitons à les installer en exécutant les commandes suivantes :

- *pip install pygame*
- *pip install pymunk*
- *pip install tkinter*

Il est aussi nécessaire que votre version python soit supérieure ou égale à la 3.x.x, nous vous conseillons d'utiliser la version 3.12.x.

IV. Déploiement

Afin de déployer le projet, rien de plus simple, télécharger le dossier "sources" présent sur GitLab, n'oubliez pas l'installation des dépendances et exécutez le fichier main.py sur votre interpréteur python. Enjoy :)

V. Ressources utilisées

De nombreuses ressources nous ont été utiles tout au long du développement du projet, les voici:

- documentation pygame (https://www.pygame.org/docs/ref/pygame.html)
- documentation pymunk (http://www.pymunk.org/en/latest/pymunk.html)
- Introduction à pymunk - Oujood (https://www.oujood.com/pymunk/index.php)
- Introduction à pymunk - Ear Of Corn Programming (Pymunk Basics - YouTube)
- Introduction à pygame - Oujood (https://www.oujood.com/pygame/index.php)