Conception

Ce document a pour but de présenter la structure générale du code, si vous voulez connaitre en détail le fonctionnement des classes, se référer aux .h qui contiennent des commentaires explicatifs.

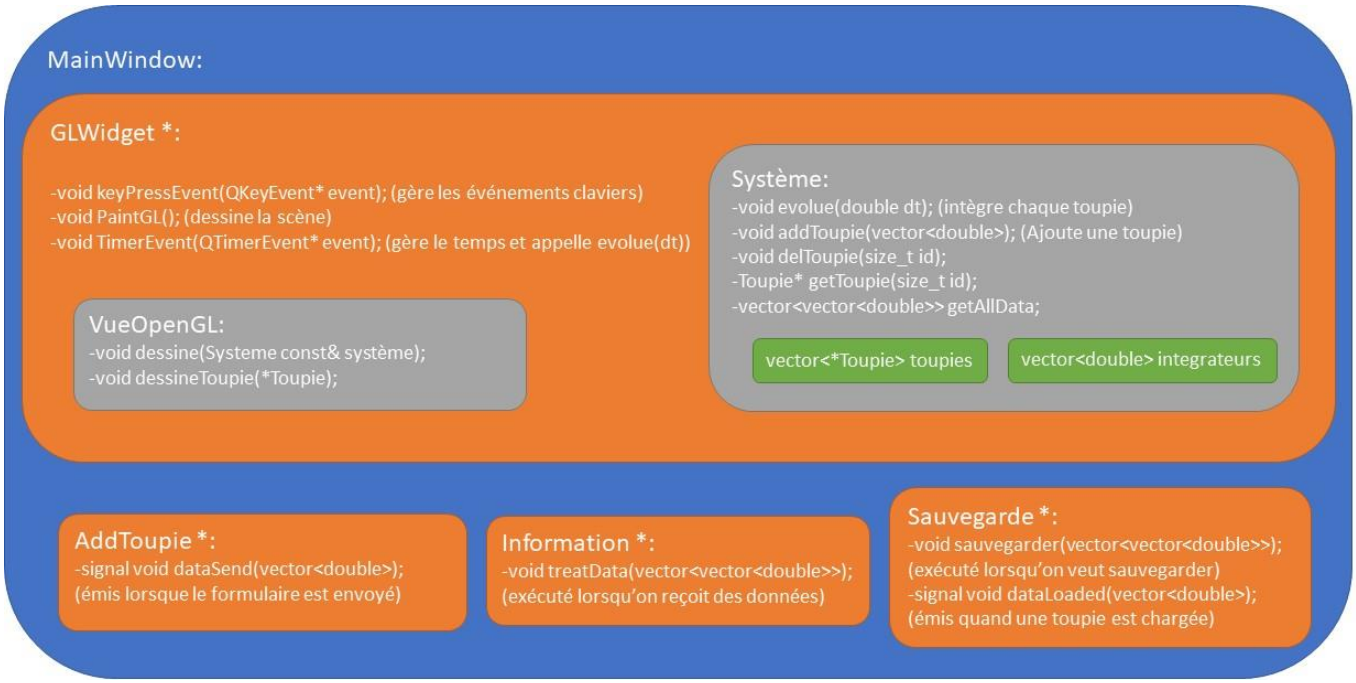


Schéma résumant la conception générale du projet BeyBlade

MainWindow :

C'est la classe qui correspond à la fenêtre principale. Elle est directement déclarée dans le int main()

Elle contient un **GLWidget *** à qui elle renvoie les évènements claviers. Elle permet, via un **QMenu,** d'afficher son formulaire **AddToupie ***, de recevoir les données inscrites par l'utilisateur et de les transmettre à son **GlWidget ***. Elle permet aussi de supprimer des toupies à partir d'un autre **QMenu** et aussi de charger et sauvegarder le système via son objet **Sauvegarde *.**

Elle permet enfin, d'afficher des fenêtres **Information *,** à qui elle transmet, quand elle reçoit le signal *emitEverySecond()* de **GLWidget ***, un vector<vector<double>> qui contient toutes les données du système**.**

Sauvegarde :

C'est une classe qui hérite de **QWidget** pour permettre d'utiliser **QFileDialog** (afin que l'utilisateur sélectionne un fichier .txt).

Elle permet la lecture et l'écriture de fichier de sauvegarde du système en recevant/envoyant un vector<vector< double>> qui contient l'ensemble des paramètres des toupies du système.

Afin de garantir l'authenticité des données nous avons ajouté une fonction de hash qui permet de détecter les fichiers corrompus, cela empêche l'écriture directe dans le fichier texte.

Information :

C'est une classe qui permet l'affichage des données d'une toupie du système. Elle hérite de **QWidget**. Elle reçoit de **MainWindow** un vector<vector<double>> qui contient toutes les données du système, et affiche celles qui portent un intérêt. L'utilisateur sélectionne la toupie qu'il souhaite observer.

AddToupie :

C'est une classe qui permet l'affichage d'un formulaire de création de nouvelle toupie. L'utilisateur rentre ses données et celles-ci sont envoyé à **MainWindow** via un signal émis lorsque l'utilisateur valide ses choix**.** Le formulaire s'adapte en fonction du choix de type de toupie.

GLCone :

C'est une classe qui permet l'affichage d'un cône générique. Elle calcule une seule fois les coordonnées des coins d'un cône de x cotés et permet via une méthode l'affichage d'un cône. Celle-ci revoit en argument un double qui fait varier la couleur du cône.

GLSphere :

C'est une classe qui permet l'affichage d'une sphère, elle est- reprise des exemples du professeur.

GLChinoise :

C'est une classe qui permet l'affichage d'une toupie chinoise, elle calcule une seule fois toutes les coordonnées d'une sphère. Pour dessiner la toupie elle reçoit une hauteur et tronque la toupie.

GLWidget :

C'est une classe qui permet l'affichage graphique 3D du système via OpenGL, elle hérite de **OpenGLWidget** et est contenu dans **MainWindow**.

Elle possède une **VueOpenGL*** avec laquelle elle initialise son Système * (ainsi Système.dessine() appelle vue.dessine(Système)).

C'est elle qui gère les évènement clavier, la gestion du temps (dt) et elle permet le dialogue entre **MainWindow** et **Système** via des slots et des signaux.

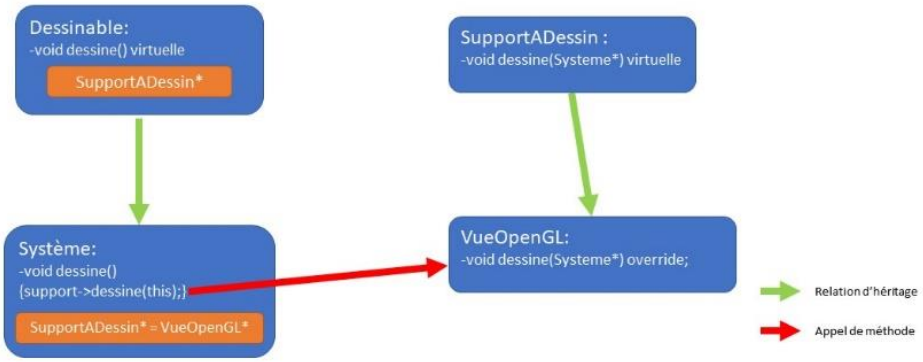


Schéma explicatif de l'appel de Systeme::dessine

Dessinable :

C'est une classe abstraite qui est mère de **Système**. Elle possède un **SupportADessin** et une méthode dessine.

Système :

C'est un objet **Dessinable**. C'est la classe qui va gérer concrètement les **Toupies**. Elle possède un vecteur de pointeurs de **Toupies**, un vector de int qui correspond aux intégrateurs liés à chaque toupie, et un vector de pointeurs de balles.

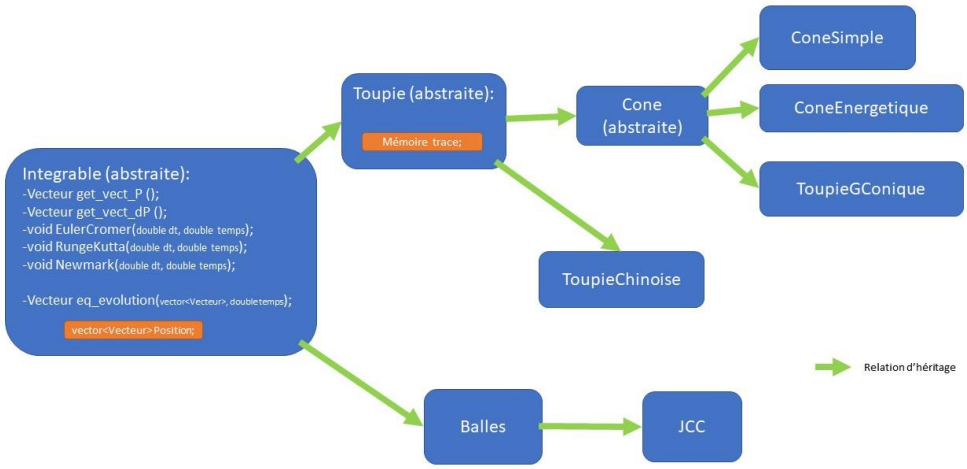
A travers différentes méthodes elle permet d'ajouter supprimer des toupies et **Balles** ainsi que de les faire évoluer, avec l'intégrateur choisi, via sa méthode *evolue* qui reçoit un dt de **GLWidget.**

SupportADessin :

C'est une classe virtuelle qui représente les différents moyens d'affichage.

VueOpenGL :

C'est le support à dessin graphique. A l'aide de sa fonction dessine qui reçoit un **Système** il va dessiner toutes les toupies de ce système à l'aide de **GLSphere, GLChinoise** et **GLCone.** Elle reçoit régulièrement l'ordre de dessiner le système de **GLWidget** via l'appel de PaintGl(). C'est elle qui va gérer les QMatrix4x4 pour les transformations d'espaces, les textures, les couleurs…



On surcharge *eq_evolution* dans chaque classe fille de Integrable non abstraite

Structures des objets concrets

Integrable :

C'est un objet virtuel dont va hériter tout objet qui représente quelque chose de concret ; **Toupie, Balle, (Planete**). Elle possède des **Vecteurs** dans un vector qui représentent la position et la vitesse de l'objet. On utilise un vector de Vecteur pour pouvoir utiliser des objets dont les équations d'évolution sont de tous degrés.

Elle possède une méthode qui retourne le vecteur « accélération » de l'objet et enfin 3 méthodes d'intégration qui permettent de faire évoluer l'objet.

Balle :

C'est un **Integrable** très simple qui correspond à un ballon en chute libre qui rebondit .

Toupie :

C'est un **Integrable** abstrait (ne possède pas d'équation d'évolution) qui représente une toupie avec 2 vecteurs position et vitesse à 5 dimensions (psi thêta phi X Y). Elle possède une masse volumique une hauteur et un rayon (qui sont interprétés différemment chez un **Cone** et une **ToupieChinoise**). Elle a des méthodes de calculs de différentes quantités intéressantes. Elle possède une **Mémoire** et des méthodes renvoyant des Vecteur permettant l'affichage d'une trace.

Cone :

C'est une toupie abstraite qui définit certains concepts mathématiques d'un cône.

ConeSimple :

Il hérite de **Cone** et correspond au cône simple du complément mathématique.

ConeEnergetique :

Il hérite de **Cone** et correspond au cône énergétique du complément mathématique.

ToupieGConique :

Il hérite de **Cone** et correspond la une toupie générale du complément mathématique.

Vecteur :

C’est une classe qui correspond à un vecteur. On peut directement l’utiliser pour des calculs

Matrice33 :

C’est une classe qui correspond à une matrice 3x3, on peut l’utiliser directement pour des calculs.

Mémoire :

Elle permet le stockage des traces des toupies via un std::deque.

Erreur :

Elle recueille tous les throw du code et affiche l’erreur correspondante.

JCC :

SURPRISE.