Rapport de modélisation

Tangram

Réalisé par :

BEN HAMOUDA Amel
DURAND Aurélien

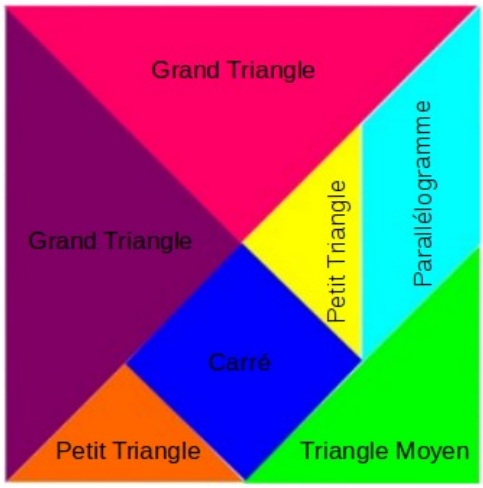
Université Paris-Est Marne-la-Vallée
Master 2 Science de l'image
2019/2020

Table des matières

I. Description du projet....3
II. Diagramme des classes....4
III. Description de l’architecture....4
1. Classe Shape....5
2. Classe Parallelogram....6
3. Classe Square....6
4. Classe RigthTriangle....7
5. Classe Button....7
6. Classe Menu....7
7. Classe Board....8
8. Classe Interface....8

I. Description du projet

Notre projet Tangram, consiste à développer un jeu d’origine chinoise composé de 7 pièces.



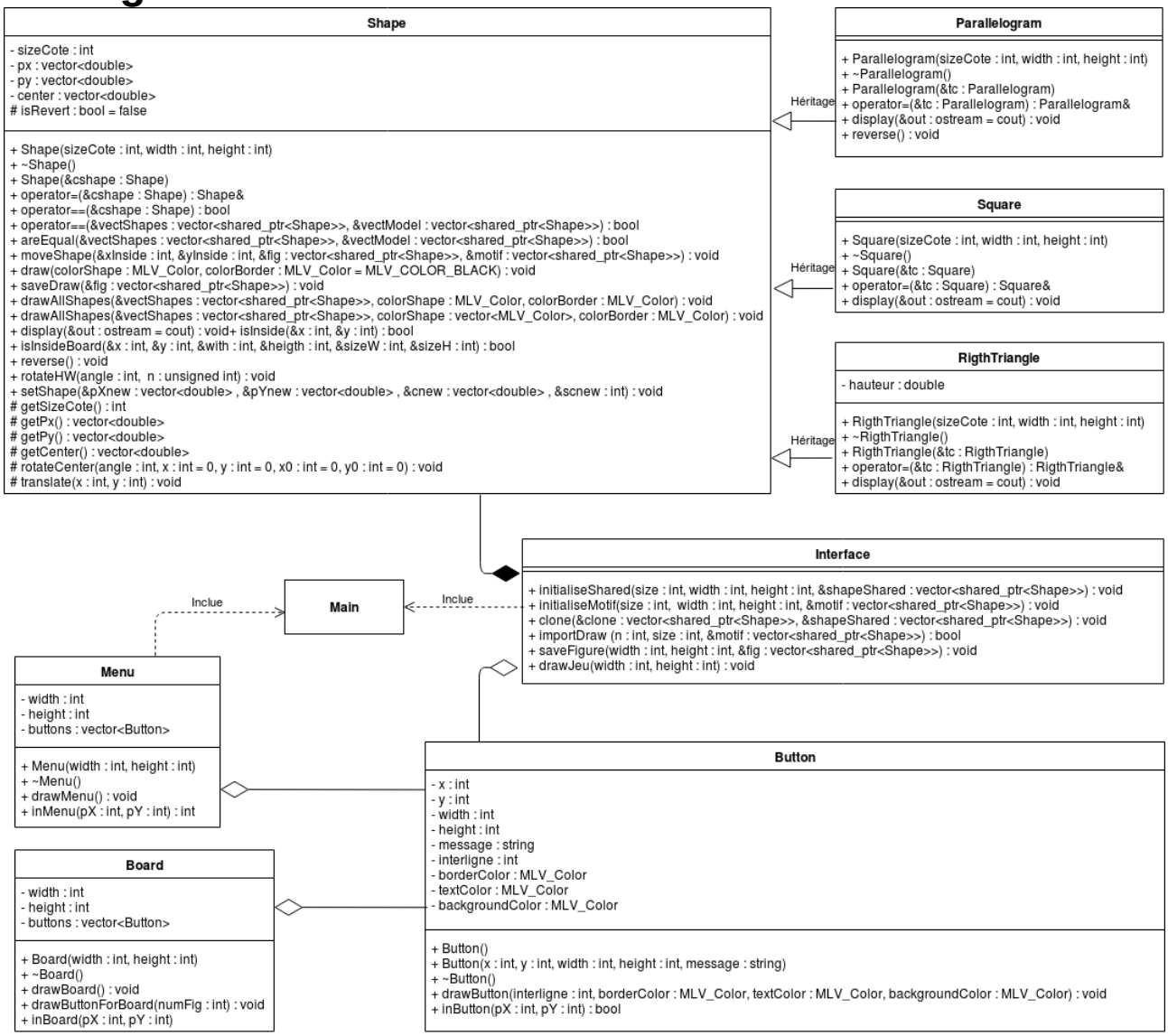
Ce casse-tête, a pour but de reproduire une forme donnée. Les règles sont simples :

→ on utilise toujours la totalité des pièces qui doivent être posées à plat,
→ et ne pas superposer les pièces.

Notre projet est principalement découpé en deux parties.
La première est la partie "mécanique" dans laquelle se trouve toute la partie de création et de mouvements des formes.
La seconde est la partie "graphique" dans laquelle se trouve toute la partie nécessaire à l’affichage graphique (visuelle) et les fonctionnalités.

Ce rapport a pour but de décrire l’architecture que nous allons développer pour notre projet.

II. Diagramme des classes



III. Description de l’architecture

Au niveau de la partie "mécanique", on a choisi de distinguer quatre classes différentes.

La classe mère est la classe *Shape* qui va comporter la majoritée des méthodes permettant de travailler avec les pièces du Tangram. Ensuite on prend trois classes filles héritant de *Shape* et ayant des constructions qui leur son propres. On a alors 3 classes filles : *Parallelogram*, *Square* et *RightTriangle*.
Ces trois classes filles permettent donc de créer chacune des pièces du Tangram séparément et héritent des méthodes de *Shape*. Elles redéfinissent les méthodes virtuels de la classe *Shape* si cela est nécessaire.

Alors qu’au niveau "graphique", on a choisi de distinguer trois classes différentes.
Les classes *Menu* et *Board* qui contiennent des bouttons (classe *Button*), ainsi que le plateau de jeu dans la classe *Board*.

Ces différentes classes sont rassemblées dans la classe *Interface* qui permet la "fusion" des deux parties différentes.

Nous allons expliquer plus en détail les différentes classes dans la suite du rapport.

1. Classe Shape

C’est la classe mère qui comprend l’ensemble des méthodes et attributs nécessaire à la fabrication et au déplacement des pièces du Tangram.

Ces attributs sont :

→ deux vecteurs de coordonnées (x,y) délimitant un polygone,
→ la taille du coté de construction, par exemple le coté d’un carré,
→ le centre de masse du polygone calculé à partir des coordonnées,
→ et un booléen pour savoir si on a effectué la symétrie de l’objet.

Ces méthodes sont :

→ un constructeur à 3 paramètres recueillant la position sur la fenêtre du premier point de construction du polygone ainsi que la taille d’un coté de construction,
→ un destructeur virtuelle,
→ un constructeur de copie,
→ une surcharge de l’opérateur de copie,
→ des getters pour chaque attributs,
→ un setter pour les classes filles,
→ une surcharge de l’opérateur de comparaison pour un polygone mais aussi pour un vecteur de polygone,
→ une méthode "areEqual" mesurant si deux vecteurs de figures sont équivalente à ± une tolérance défini arbitrairement,
→ une méthode "moveShape" permettant de prendre en considération les déplacements d’un objet en fonction de la position de la souris,
→ deux méthodes de dessins, une pour dessiner une figure et l’autre pour dessiner un vecteur de figure,
→ une méthode "display" déclarée virtuelle permettant d’afficher les informations d’un polygone, ces coordonnées, la taille de construction et son centre,
→ une méthode "inside" permettant de savoir si un point de coordonnée x et y ce trouve dans un polygone,

→ une méthode "insideBoard" pour savoir si une figure va dépasser les bords de la fenêtre de dessin,

→ une méthode "reverse" qui permet de faire la symétrie d'un objet. Cette méthode est déclarée virtuelle et de base n'effectue qu'une rotation de 180° pour des objets comme le carré/triangle dont la symétrie n'est qu'une rotation. Elle doit être redéfini pour des figures plus complexe, par exemple pour le parallélogramme du Tangram.

→ une méthode "rotateHW" permettant la rotation d'un objet autour d'une de ces coordonnées,

→ une méthode "rotate" permettant la rotation d'un objet par rapport à son centre,

→ une méthode "translate" réalisant la translation d'un polygone.

2. Classe Parallelogram

L'une des classes qui hérite de *Shape* permettant de dessiner un parallélogramme dont la hauteur est reliée par le premier et le troisième point, permettant de faire le parallélogramme que l'on retrouve dans un Tangram.

Ces méthodes sont :

→ un constructeur à trois paramètres comme pour la classe *Shape*,

→ un destructeur virtuelle,

→ un constructeur de copie,

→ une surcharge de l'opérateur=,

→ une redéfinition de la méthode virtuelle "display" de la classe *Shape*.

→ une redéfinition de la méthode virtuelle "reverse" de la classe *Shape*. Il faut pour le parallélogramme réalisé la symétrie par rapport à un axe de symétrie car une rotation par 180° n'est pas juste.

3. Classe Square

L'une des classes qui hérite de *Shape* permettant de dessiner un carré que l'on retrouve dans un Tangram.

Ces méthodes sont :

→ un constructeur à trois paramètres comme pour la classe *Shape,*

→ un destructeur virtuelle,

→ un constructeur de copie,

→ une surcharge de l'opérateur=,

→ une redéfinition de la méthode virtuelle "display" de la classe *Shape*.

4. Classe RigthTriangle

L’une des classes qui hérite de *Shape* permettant de dessiner un triangle rectangle, ce qui permet de faire les différents triangle rectangle que l’on retrouve dans un Tangram.

Ces méthodes sont :

→ un constructeur à trois paramètres comme pour la classe *Shape*,
→ un destructeur virtuelle,
→ un constructeur de copie,
→ une surcharge de l’opérateur=,
→ une redéfinition de la méthode virtuelle "display" de la classe *Shape*.

5. Classe Button

C’est la classe qui permet la création de bouton pour les fonctionnalités de la partie graphique.

Ces attributs sont :

→ la coordonnée nord ouest du bouton (x, y),
→ la longueur et la hauteur de la fenêtre,
→ le texte sur le bouton,
→ l’espacement des boutons (si nécessaire),
→ les couleurs du texte, du bouton et de la bordure du bouton.

Ces méthodes sont :

→ un constructeur sans aucun paramètre,
→ un constructeur à cinq paramètres (les cinq premiers attributs),
→ un destructeur virtuelle,
→ une méthode "drawButton" qui permet de dessiner un bouton,
→ une méthode "inButton" qui permet de savoir si le clic de la souris est sur le bouton.

6. Classe Menu

C’est la classe qui permet l’affichage du menu de départ.

Ces attributs sont :

→ la longueur et la hauteur de la fenêtre,
→ un vecteur de bouton qui permet le stockage des boutons nécessaires.

Ces méthodes sont :

→ un constructeur à deux paramètres (les deux premiers attributs),
→ un destructeur virtuelle,
→ une méthode "drawMenu" qui permet de dessiner le menu,
→ une méthode "inMenu" qui permet de connaître le choix de l’utilisateur.

7. Classe Board

C’est la classe qui permet l’affichage du plateau de jeu.

Ces attributs sont :

→ la longueur et la hauteur de la fenêtre,

→ un vecteur de bouton qui permet le stockage des boutons nécessaires.

Ces méthodes sont :

→ un constructeur à deux paramètres (les deux premiers attributs),

→ un destructeur virtuelle,

→ une méthode "drawBoard" qui permet de dessiner le plateau de jeu,

→ une méthode "drawButtonForBoard" qui permet de dessiner les boutons nécessaires,

→ une méthode "inBoard" qui permet de connaître le choix de l’utilisateur pour les boutons utilisés.

8. Classe Interface

C’est la classe qui permet la "fusion" des deux grandes parties de ce projet.

Ces méthodes sont :

→ une méthode "initialiseShared" qui permet de créer les pièces du Tangram à utiliser pour construire une figure à l’aide de la classe *Shape* et des enfants *RigthTriangle*, S*quare* et P*arallelogram*. Cette méthode renvoie un vecteur de S*hape* de pointeur sur chaque objet. On utilise des pointeurs car nos classes utilisent de l’héritage et des méthodes virtuelles.

→ une méthode "initialiseMotif" qui est similaire à la méthode "initialiseShared" et permet en fait de créer un motif de base pour le Tangram.

→ une méthode "clone" qui permet de réaliser la copie d’un vecteur de S*hape.*

→ une méthode "importDraw" qui permet d’importer des figures d’un fichier texte regroupant les coordonnées et attributs d’un motif de Tangram.

→ une méthode "saveFigure" qui permet d’effectuer une sauvegarde d’un motif qu’un utilisateur pourrait faire dans le jeu.

→ une méthode "drawJeu" qui permet d’afficher l’ensemble des figures, motifs et pièces, et d’appeler les différentes méthodes de la classe S*hape* et de ces enfants. Cette méthode est le cœur de l’interface et c’est dedans que l’on vient créer/afficher les objets ainsi que l’on appel les différentes méthodes de déplacement d’objets et d’égalité entre deux ensembles de figures. De plus on vient programmer les différents boutons nécessaire aux jeux grâce à la classe *Button*, des méthodes de l’interfaces et de la classe S*hape*.