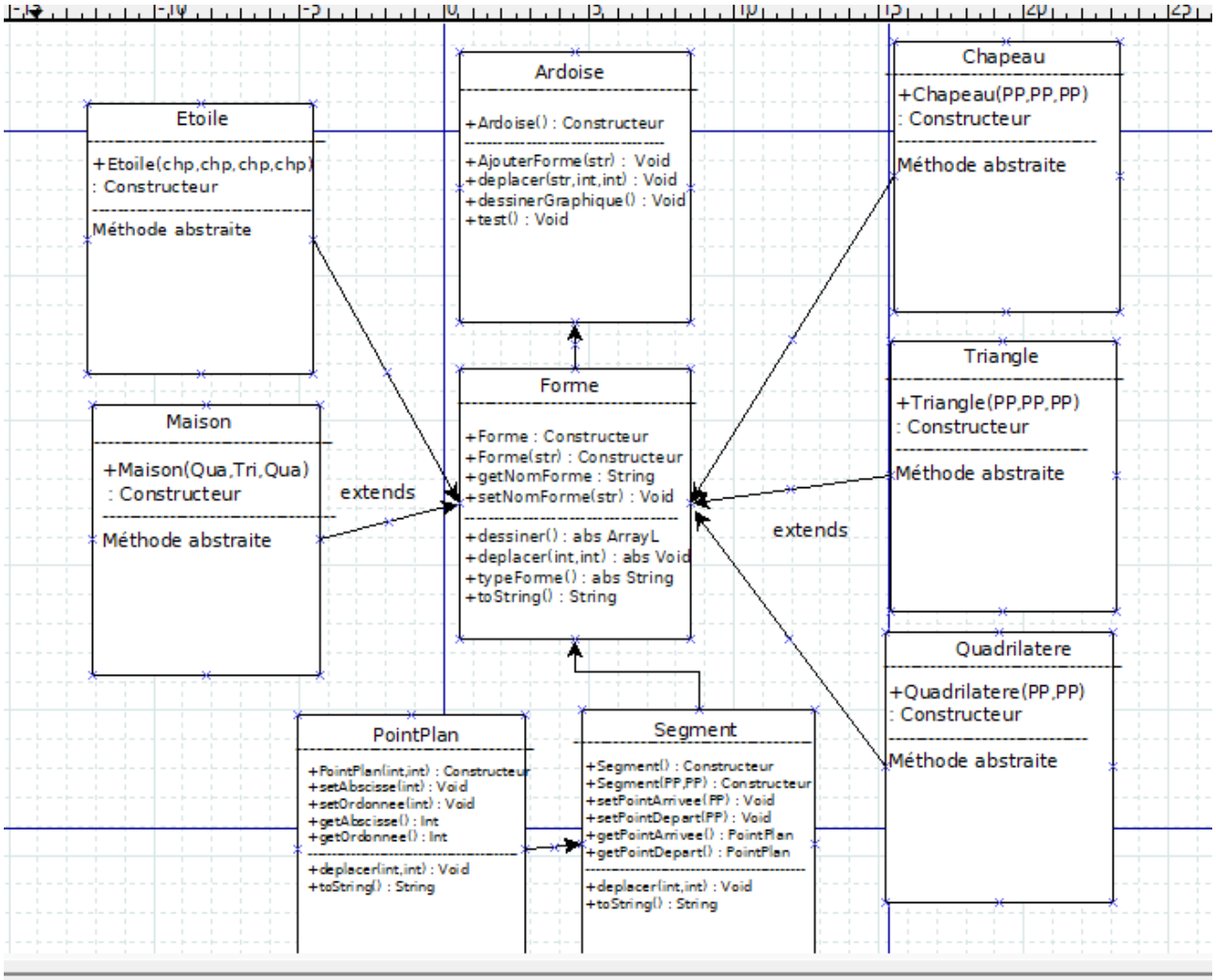
Denis-Can YUKER
Hernani Castro de Almeida
Whaitiri

Compte rendu SAE 2.01

Moi (Denis-Can) et Hernani, nous nous sommes réparti les tâches à faire, chacun a fait ses classes. Si nous avions des problèmes, nous nous entraidions.

Diagramme Finale :



Création de l’environnement de développement (Denis-Can):

J'ai commencé par configurer mon environnement de développement en installant les packages nécessaires pour la SAE (ardoise.jar et le javadoc). Ensuite, j'ai créé un nouveau projet Java (JavaSE-17) dans Eclipse et j'ai modifié l'emplacement du workspace pour pouvoir créer un dossier "libs" et ajouter ardoise.jar.
En essayant d'installer la librairie ardoise, j'avais comme erreur : "The package 'ardoise' is not accessible". Ce qui n'installais pas correctement la librairie.
En faisant des recherches, J'avais 2 solutions :
- Soit je supprime le fichier "module-info.java"
- Soit j'ajoute la ligne "requires ardoise;" sur ce même fichier

J'ai choisi l'option 2 car je ne connais pas trop ce fichier, c'est sûrement important.

Par contre l'option 2 me donne une erreur "Name of automatic module 'ardoise' is unstable, it is derived from the module's file", mais je l'ignore car elle n'a aucun impact sur le projet et la librairie fonctionne.

Répartition de classes:

Quadrilatere, Triangle et Chapeau: Denis-Can

Maison et Etoile: Hernani

Quadrilatere (Denis-Can):

J'ai commencé à créer la classe "Quadrilatere". Pour la classe "Quadrilatere", j'ai importé ardoise.jar et la classe ArrayList. J'ai hérité de la classe "Forme" car le quadrilatère nécessite les variables de forme. J'ai défini deux points, "pointSuperieurGauche" et "pointInferieurDroit". ensuite j'ai utilisé les setters et les getters pour permettre l'accès et la modification des points supérieur gauche et inférieur droit du quadrilatère. pour pointSuperieurGauche et pointInferieurDroit. Ensuite j'ai ajouté une méthode dessiner() pour dessiner le quadrilatère, j'ai ajouté un arraylist et j'ai créer 4 segments en utilisant les points du quadrilatère. Chaque segment est créé en reliant deux points du quadrilatère, formant ainsi les côtés du quadrilatère. Les segments sont ajoutés à l'ArrayList pour former les segments qui forme le quadrilatère. J'ai ajouté une méthode "deplacer()" pour déplacer le quadrilatère en utilisant les valeurs de déplacement fournies. Pour représenter le type de figure du quadrilatère, j'ai ajouté une méthode "typeForme()" qui retourne la chaîne "Q". Et enfin, j'ai ajouté une méthode "toString()" pour afficher les points supérieur gauche et

inférieur droit du quadrilatère. Cette classe était un peu compliquée à la faire car je devais utiliser les variables du javadoc de l'ardoise et où la placer (dans quel méthode).

Triangle (Denis-Can):

Jai crée la classe "Triangle", elle était moins compliqué car elle ressemblait un peu à la classe "Chapeau". J'ai importé ardoise.jar et la classe ArrayList, puis j'ai hérité de la classe "Forme". J'ai ajouté trois points pour représenter les sommets du triangle, car un triangle nécessite trois points. J'ai également ajouté des méthodes getters et setters. J'ai implémenté la méthode "dessiner()" pour dessiner le triangle en créant les segments appropriés. Pour déplacer le triangle, j'ai ajouté la méthode "deplacer()" qui déplace chaque point du triangle en utilisant les valeurs de déplacement fournies. Pour représenter le type de figure du triangle, j'ai ajouté la méthode "typeForme()" qui retourne la chaîne "T". Enfin, j'ai ajouté une méthode "toString()" pour afficher les points du triangle.

Chapeau(Denis-Can) :

J'ai créé la classe "Chapeau" car elle était ressemblait un peu à la classe Triangle. J'ai importé la classe "ArrayList". La classe "Chapeau" hérite de la classe abstraite "Forme" car elle a besoin des variables de forme. Dans la classe "Chapeau", j'ai défini deux points : "point1" et "point2". J'ai utilisé les méthodes setters et getters pour permettre l'accès et la modification de ces points du chapeau.Ensuite, j'ai ajouté une méthode "dessiner()" pour dessiner le chapeau. J'ai créé deux segments en utilisant les points du chapeau : un segment reliant le point1 et le point2, J'ai ajouté ces segments à une ArrayList pour représenter le chapeau.Pour déplacer le chapeau, j'ai ajouté une méthode "deplacer()" qui déplace chaque point du chapeau en utilisant les valeurs de déplacement fournies. Pour le type de figure du chapeau, j'ai ajouté la méthode "typeForme()" qui retourne la chaîne de caractères "CH". Enfin, j'ai ajouté une méthode "toString()" pour afficher les points du chapeau.

Maison (Hernani) :

J'ai créé la classe "Maison" qui représente une maison. Pour construire la maison, j'ai instancié deux objets de la classe "Quadrilatere" : un pour représenter le corps de la maison et un autre pour représenter la porte. J'ai également instancié un objet de la classe "Triangle" pour représenter le toit de la maison.

Le corps de la maison a été créé en utilisant un quadrilatère, défini par quatre points appropriés. J'ai utilisé les méthodes setters et getters pour permettre l'accès et la modification de ces points.
La porte de la maison a été représentée par un autre quadrilatère, défini par les points nécessaires. J'ai également utilisé les méthodes setters et getters pour manipuler ces points.
Le toit de la maison a été construit à l'aide d'un triangle, qui a été instancié avec les trois points appropriés. J'ai également utilisé les méthodes setters et getters pour manipuler ces points.
Pour dessiner la maison, j'ai utilisé les méthodes de dessin des objets Quadrilatere et Triangle, qui retournent les segments nécessaires pour représenter les différentes parties de la maison.
La classe Maison a également une méthode "deplacer()" qui permet de déplacer la position de la maison en utilisant les valeurs de déplacement fournies.
Pour afficher le type de figure, j'ai ajouté une méthode "typeForme()" qui retourne la chaîne de caractères "M".
Enfin, j'ai ajouté une méthode "toString()" pour afficher les détails de la maison, tels que les coordonnées des différents points.
Ce compte rendu décrit les étapes que j'ai suivies pour créer la classe Maison et met en évidence les composants utilisés, tels que les quadrilatères pour le corps et la porte, ainsi que le triangle pour le toit de la maison.

Etoile(Hernani) :

J'ai créé la classe "Etoile" qui représente une étoile. Pour représenter les branches de l'étoile, j'ai instancié quatre objets de la classe "Chapeau". Chaque branche de l'étoile est représentée par un chapeau.
Les chapeaux ont été instanciés avec les points nécessaires pour représenter la forme de chaque branche. J'ai utilisé les méthodes setters et getters pour manipuler ces points.
Pour dessiner l'étoile, j'ai utilisé les méthodes de dessin des objets Chapeau, qui retournent les segments nécessaires pour représenter chaque branche de l'étoile.
La classe Etoile a également une méthode "deplacer()" qui permet de déplacer la position de l'étoile en utilisant les valeurs de déplacement fournies.
Pour afficher le type de figure, j'ai ajouté une méthode "typeForme()" qui retourne la chaîne de caractères "E".
Enfin, j'ai ajouté une méthode "toString()" pour afficher les détails de l'étoile, tels que les coordonnées des différents points.

Autre :

On a réaliser la classe Main ensemble en combinant tout les classes qu’on a faits, chacun c’est occupé de ses propres classe mais Hernani a réaliser en plus l’oiseau, la tour et la montagne

L’oiseau est un objet : Instance de Chapeau.
La tour est un objet : Instance de Quadrilatere
Et la montagne est un objet : Instance de Triangle

Pour les tests unitaires comme on ne savait pas trop comment ça fonctionnais on s’est réuni tout les midis pour le faire ensemble en présentiel à la BU (Bibliothèque Universitaire) en plus des heures de saé.

Classe "ChapeauTest":

- La méthode de test "testDessiner()" a été implémentée.
- Des points et un objet "Chapeau" ont été créés.
- La méthode "dessiner()" de l'objet "Chapeau" est appelée et le nombre de segments est vérifié.
- Les segments sont vérifiés pour s'assurer qu'ils relient les points correctement.
- La méthode de test "testDeplacer()" a été implémentée.
- Des points et un objet "Chapeau" ont été créés.
- L'objet "Chapeau" est déplacé en utilisant la méthode "deplacer()".
- Les nouvelles coordonnées des points sont vérifiées.
- Les méthodes de test "testDessiner()" et "testDeplacer()" ont été réussies.

Classe "EtoileTest":

- La méthode de test "testDessiner()" a été implémentée.
- Des segments et des objets "Chapeau" ont été créés pour représenter les branches de l'étoile.
- Des objets "Etoile" ont été créés en utilisant les branches.
- La méthode "dessiner()" de l'objet "Etoile" est appelée et le nombre de segments est vérifié.
- Les segments sont vérifiés pour s'assurer qu'ils sont présents.
- La méthode de test "testDeplacer()" a été implémentée.
- Des objets "Chapeau" ont été créés pour représenter les branches de l'étoile.
- Des objets "Etoile" ont été créés en utilisant les branches.
- L'objet "Etoile" est déplacé en utilisant la méthode "deplacer()".
- Les coordonnées des points des branches sont vérifiées après le déplacement.
- La méthode de test "testTypeForme()" a été implémentée.
- Des objets "Chapeau" ont été créés pour représenter les branches de l'étoile.
- Des objets "Etoile" ont été créés en utilisant les branches.
- Le type de forme de l'étoile est vérifié.
- La méthode de test "testToString()" a été implémentée.

- Des objets "Chapeau" ont été créés pour représenter les branches de l'étoile.
- Des objets "Etoile" ont été créés en utilisant les branches.
- La méthode "toString()" de l'objet "Etoile.

Classe "TriangleTest":

- La méthode de test "testDessiner()" a été implémentée.
- Des points et un objet "Triangle" ont été créés.
- La méthode "dessiner()" de l'objet "Triangle" est appelée et le nombre de segments est vérifié.
- Les segments sont vérifiés pour s'assurer qu'ils relient les points correctement.
- La méthode de test "testDeplacer()" a été implémentée.
- Des points et un objet "Triangle" ont été créés.
- L'objet "Triangle" est déplacé en utilisant la méthode "deplacer()".
- Les nouvelles coordonnées des points sont vérifiées.
- La méthode de test "testTypeForme()" a été implémentée.
- Des points et un objet "Triangle" ont été créés.
- Le type de forme du triangle est vérifié.
- La méthode de test "testToString()" a été implémentée.
- Des points et un objet "Triangle" ont été créés.
- La méthode "toString()" de l'objet "Triangle" est appelée et le résultat est vérifié.

Classe "MaisonTest":

- J'ai créé des points représentant les coins de la maison.
- J'ai créé un objet "Maison" en utilisant ces points.
- J'ai vérifié si la méthode "dessiner()" de l'objet "Maison" était appelée correctement.
- J'ai vérifié si les segments entre les points de la maison étaient correctement reliés.
- J'ai implémenté la méthode de test "testDeplacer()" pour vérifier si la méthode "deplacer()" déplaçait correctement l'objet "Maison".
- J'ai déplacé l'objet "Maison" en utilisant la méthode "deplacer()" et vérifié les nouvelles coordonnées des points.
- J'ai implémenté la méthode de test "testTypeForme()" pour vérifier si la méthode "getTypeForme()" renvoyait le bon type de forme pour l'objet "Maison".
- J'ai vérifié si le type de forme renvoyé pour l'objet "Maison" correspondait à "Maison".
- J'ai implémenté la méthode de test "testToString()" pour vérifier si la méthode "toString()" renvoyait la représentation sous forme de chaîne de caractères correcte pour l'objet "Maison".

Classe "QuadrilatereTest":

- J'ai créé des points représentant les sommets du quadrilatère.
- J'ai créé un objet "Quadrilatere" en utilisant ces points.
- J'ai vérifié si la méthode "dessiner()" de l'objet "Quadrilatere" était appelée correctement.
- J'ai vérifié si les segments entre les points du quadrilatère étaient correctement reliés.
- J'ai implémenté la méthode de test "testDeplacer()" pour vérifier si la méthode "deplacer()" déplaçait correctement l'objet "Quadrilatere".
- J'ai déplacé l'objet "Quadrilatere" en utilisant la méthode "deplacer()" et vérifié les nouvelles coordonnées des points.
- J'ai implémenté la méthode de test "testTypeForme()" pour vérifier si la méthode "getTypeForme()" renvoyait le bon type de forme pour l'objet "Quadrilatere".
- J'ai vérifié si le type de forme renvoyé pour l'objet "Quadrilatere" correspondait à "Quadrilatere".
- J'ai implémenté la méthode de test "testToString()" pour vérifier si la méthode "toString()" renvoyait la représentation sous forme de chaîne de caractères correcte pour l'objet "Quadrilatere".