Haute École Bruxelles-Brabant
École Supérieure d'Informatique
Bachelor en Informatique

2024 - 2025
3dev3a

3dev3a – Développement 3

Mise en pratique : ASCII Paint

Le modèle

Dans cet exercice vous allez implémenter la partie modèle de l'application `asciipaint` une petite application de dessin en console.

Table des matières

Version du document
23 septembre 2024

1 Mini Projet : AsciiPaint - remise 1

Le but de cet exercice est de créer une application permettant de créer et afficher différentes formes géométriques dans la console. Dans un premier temps vous allez implémenter le modèle.

Voici à quoi pourrait ressembler une illustration contenant 4 cercles dans la console :

Votre application permettra

1. d'ajouter une nouvelle forme : cercle, rectangle ou carré ;
2. d'afficher l'illustration ;
3. d'afficher la liste des formes présentes dans le dessin ;
4. de bouger une forme ;
5. de supprimer une forme ;
6. de changer sa couleur, ici la couleur est un caractère.

2 Diagramme de classes

Vous trouverez sur la figure 1 page suivante un diagramme de classes (incomplet) dont vous pouvez vous inspirer pour votre implémentation.

L'interface `Shape` représente une forme et définit les comportements attendus par toute forme. Elle déclare les méthodes :

- `move(double dx, double dy)` permettant de déplacer une forme ;
- `isInside(Point p)` retournant vrai si le point donné se trouve à l'intérieur de la forme, et faux sinon ;
- `getColor()` retournant un caractère d'affichage (sa couleur), par exemple le caractère 'c' ;
- `setColor(char color)` modifiant la couleur de la forme ;

Les classes `Circle`, qui représente un cercle, et `Rectangle`, qui représente un rectangle, étendent la classe abstraite `ColoredShape` qui elle même implémente l'interface `Shape`.

La classe `Square`, représentant un carré, est une sous-classe de `Rectangle`.

La classe `Drawing` représente une illustration sous la forme d'une collection de formes. Elle a une longueur et une largeur (50x50, 100x30, etc) et propose des accesseurs (pour `height`, `width` et `Shape`).

`AsciiPaint` est la façade du modèle et contient les méthodes permettant de modifier le modèle : ajouter un forme, bouger une forme, changer la couleur, etc.

La façade contient aussi les méthodes permettant de récupérer les informations nécessaires à l'affichage.

Associations

Notez l'association `shapes` entre `Shape` et `Drawing`, cette association possède une multiplicité 1 à n, cela signifie qu'il faut un attribut de type `List<Shape>` nommée `shapes` dans la classe `Drawing`.

Il y a également des associations simples entre `Point` et `Circle` et `Rectangle`. Cela signifie qu'il faut un attribut de type `Point` dans chacune de ces classes (`Circle` et `Rectangle`).

Finalement l'association entre `Drawing` et `AsciiPaint` signifie également qu'un attribut de type `Drawing` se trouve dans la class `AsciiPaint`.

3 Structure du code

Vous devez structurer votre code en suivant l'architecture MVC (*model view controller*).

Vous aurez au minimum 3 *packages* :

- pour le modèle : `g12345.dev.ascii.model` ;
- pour la vue : `g12345.dev.ascii.view` ;
- pour le contrôleur : `g12345.dev.ascii.controller`.

3.1 Le modèle

Le modèle contient les classes et une interface : `Point`, `Shape`, `Circle`, `Rectangle`, `Square`, `Drawing` et `AsciiPaint`.

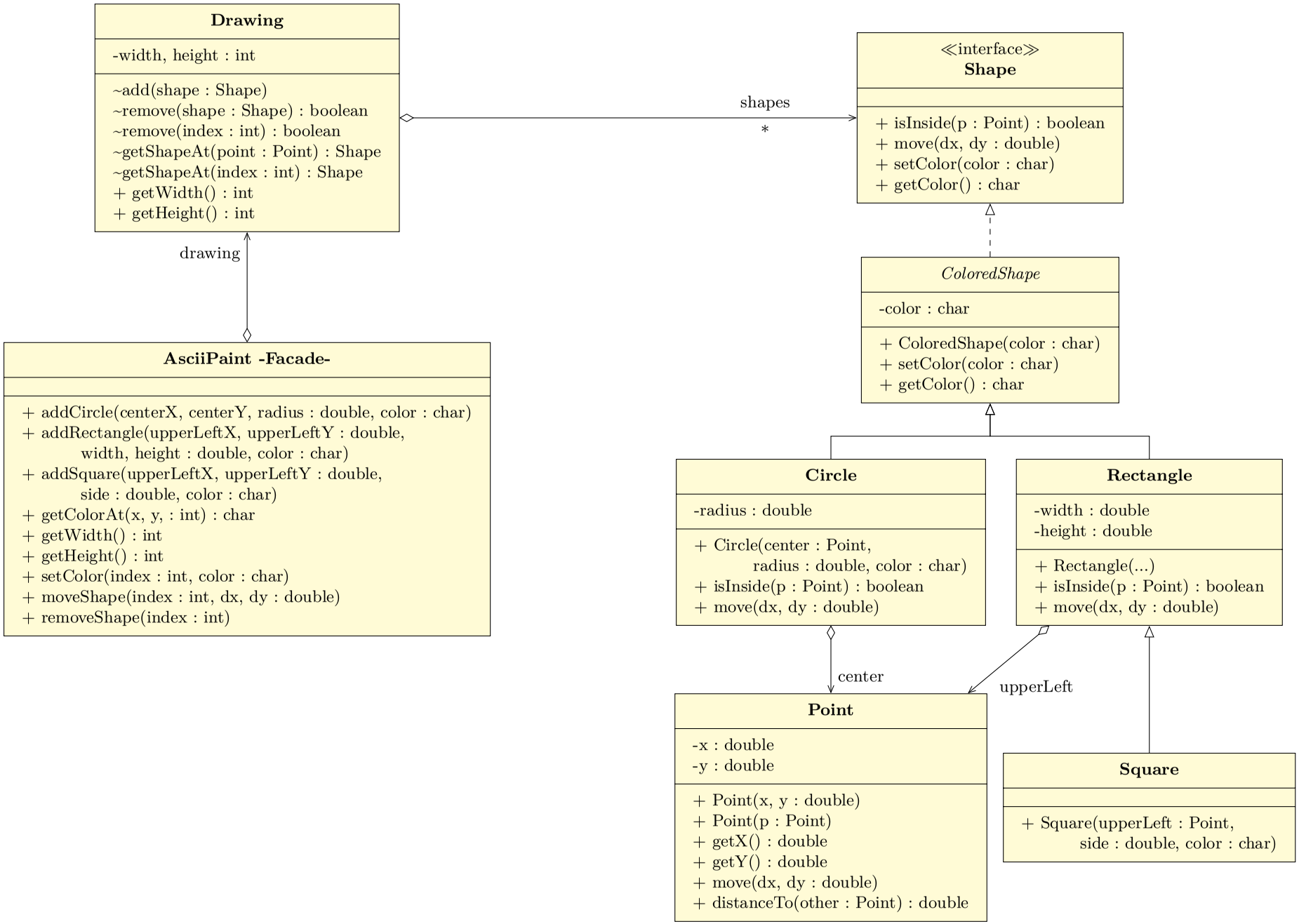

FIGURE 1 – Diagramme de classes

3.2 La vue

La vue n'est pas présente dans le diagramme de classe fourni.

Dans cette itération elle propose uniquement une méthode permettant d'afficher le dessin : `public void display(AsciiPaint paint)`.

Notez que cette méthode n'est **pas** `static`.

Tip

Pour afficher l'illustration : parcourir chaque point de chaque ligne de l'illustration (en fonction de la hauteur et la largeur du dessin), et vérifier si une forme occupe ce point de façon à afficher un blanc, si aucune forme n'occupe cette case, ou le caractère d'affichage de la forme (sa couleur) sinon.

3.3 Le contrôleur

La classe `g12345.dev.ascii.controller.AsciiController` est le contrôleur. Pour le moment il ne fait rien.

3.4 Méthode principale

La méthode principale se trouve dans la classe `g12345.dev.ascii.App`. Cette méthode :

- ▷ instancie `AsciiPaint` et la vue ;
- ▷ ajoute quelques formes ;
- ▷ les déplace et
- ▷ affiche le dessin afin de vérifier que tout semble fonctionner.

4 Tests unitaires de la Façade

Vous devez écrire des tests unitaires pour toutes les méthodes de la classe `AsciiPaint` (la façade du modèle).

Par exemple après avoir ajouté un cercle assurez vous qu'il se trouve bien sur l'illustration.