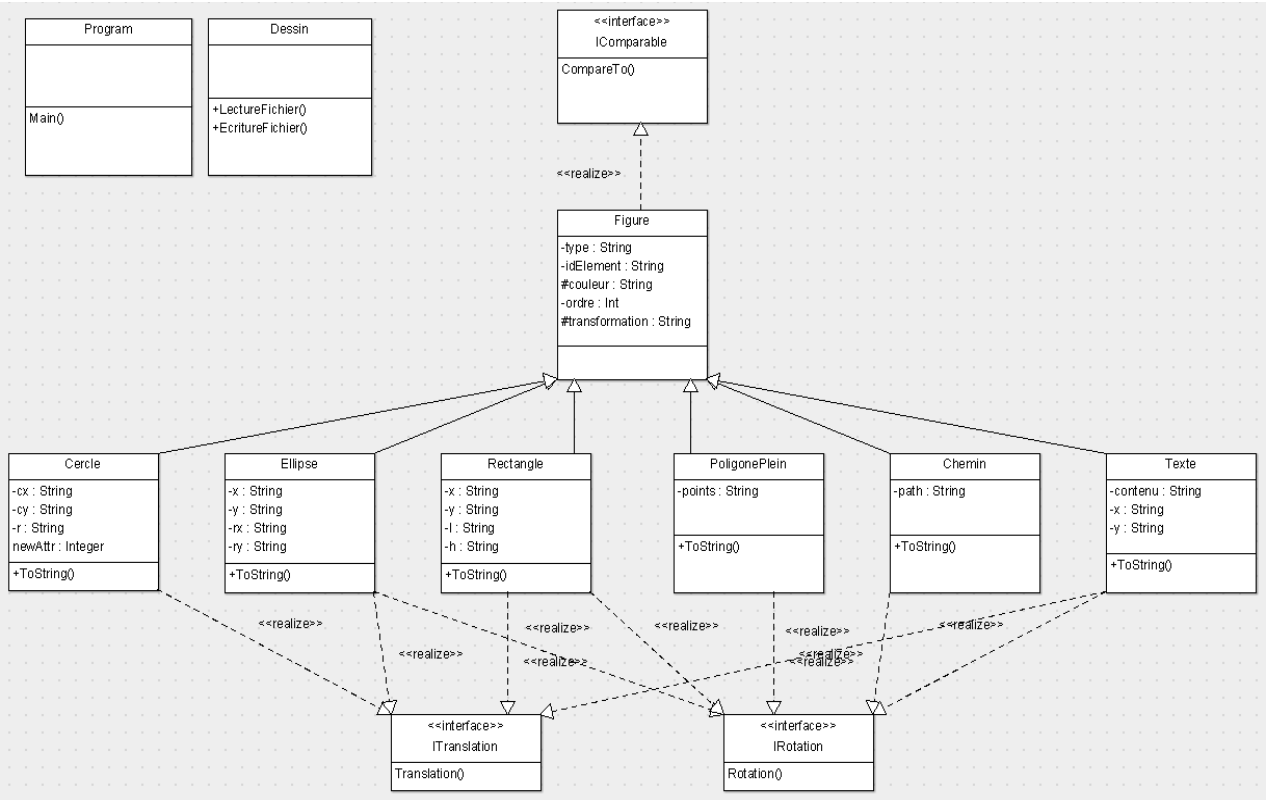
GUILLAUME Thomas

KACZMAR Guillaume

Compte Rendu POO TD Final

Exercice 1 :



La lecture du fichier CSV et l'écriture du fichier SVG doivent être rattachées à la classe Dessin.

Pour que les éléments soient écrits dans le bon ordre, on utilise le paramètre *ordre* présent pour chaque élément dans le fichier CSV. Lors de la lecture du fichier CSV, on obtient une liste d'éléments selon leur ordre dans le fichier. On trie cette liste en fonction de ce paramètre *ordre*, par l'intermédiaire de la méthode Sort. Cette méthode utilise la méthode *CompareTo*, depuis l'interface *IComparable*, que l'on a pris soin de modifier pour qu'elle compare selon *ordre* et non plus par ordre alphabétique.

Exercice2 :

Algorithme de lecture et de traitement de fichier CSV :

```
Début
   liste = nouvelle liste de figures ;
   ligne = première ligne du document ;
   TantQue (ligne != fin du document)
  {
     parametre = tableau de string, rempli par la division de la ligne quand il y a des « ; » ;
     type = parametre[0]= premier element du tableau ;
     Switch (type)
    {
       Case ' element ' : // Pour chaque type d'éléments
          id_element = parametre[1];
            attributs_particuliers = parametre[2 à x];
          couleur = {parametre[x+1] ; parametre[x+2]; parametre[x+3]};
            ordre = parametre[x+4];
            transformation = "";
            Element element = new Element(type, id, attributs_particuliers, couleur, ordre,
            transformation) ;
            liste.Add(element) ;
            Break ;

       Case 'transformation' : // Si type = Translation ou Rotation
            id_transfo = parametre[1];
            attributs_particuliers = parametre[2 à x];
            ForEach (figure dans la liste déjà remplie)
           {
             Si (id_transfo  == id_element)
            {
                Si (type == element) // Pour chaque type d'élément
              {
                  Element newelement = (Element)figure ;
                  newelement.Transformation(attributs_particuliers) ;
              }
            }
           }
           Break ;
     }
  }
   Return liste ;
Fin
```

Exercice3 :

Codage des classes.

Exercice4 :

Pour la gestion des exceptions, nous avons utilisé try…catch pour afficher un message d'erreur si l'adresse de lecture ou d'écriture est incorrecte.

Exercice5 :

La classe Figure est une classe abstraite puisqu'elle est utile uniquement afin d'en hériter des classes filles, qui sont les différents éléments possibles, sur lesquelles on agit directement. Les éléments ont en effet plusieurs attributs en communs tels que l'id, le type, la couleur, l'ordre, et la transformation. Nous n'utilisons pas la classe Figure en elle-même.

Exercice6 :

Documentation du code, cf. code.