INFO-H-200
Programmation orientée objet

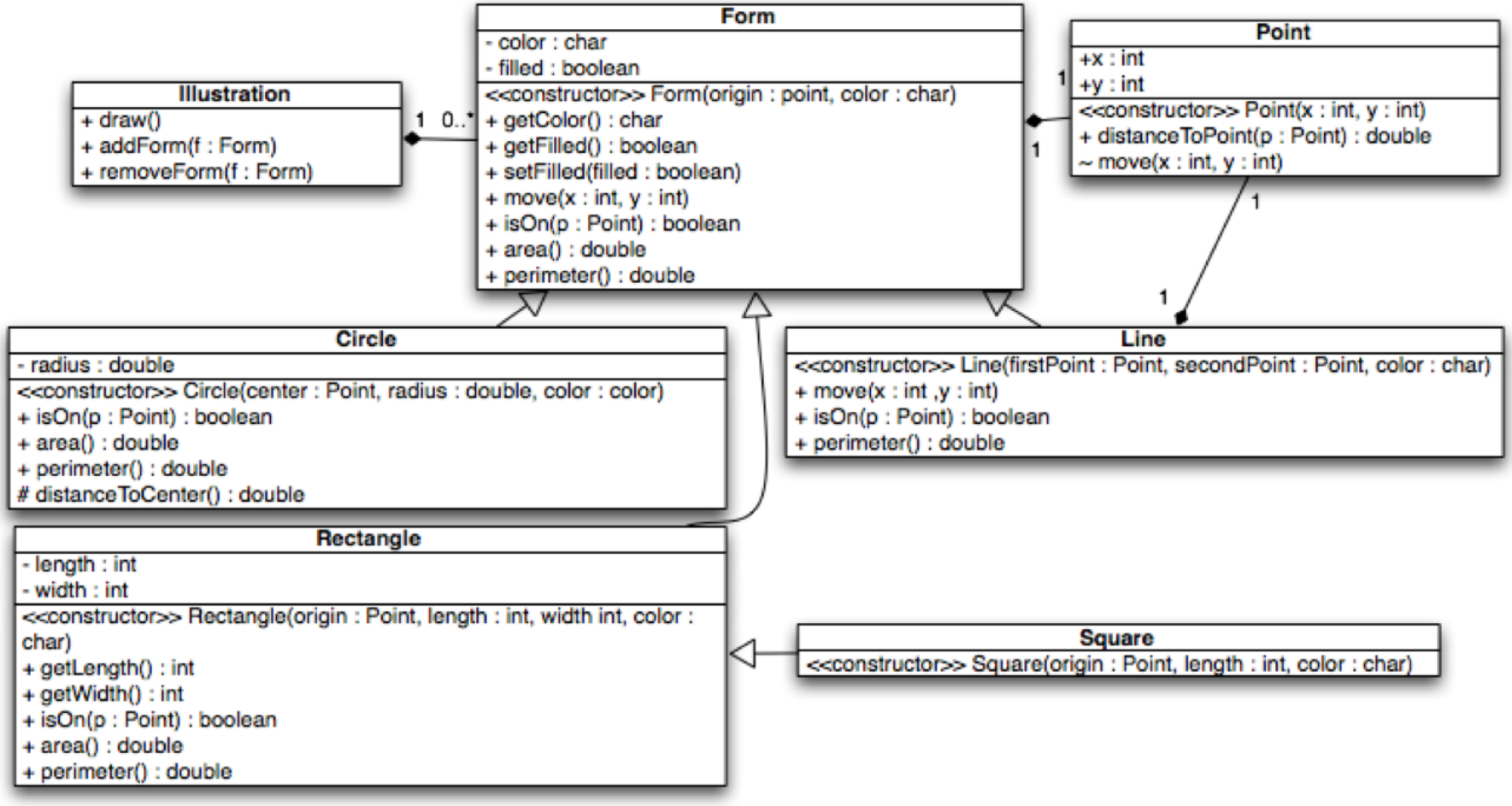
Séance d'exercices 4
UML

Université libre de Bruxelles
École polytechnique de Bruxelles

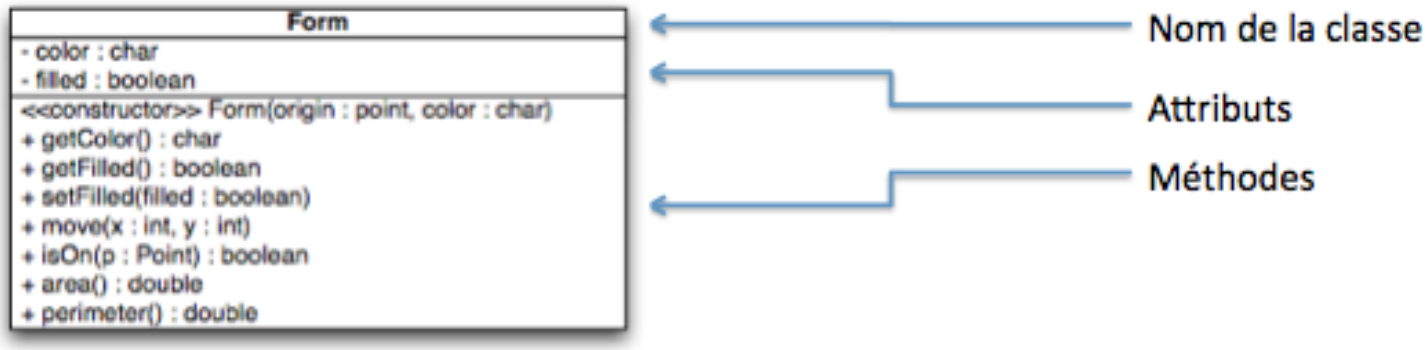
Professeur : Hugues Bersini

2014-2015

UML : Exemple de diagramme de classes



Classes, méthodes et attributs



Encapsulation :

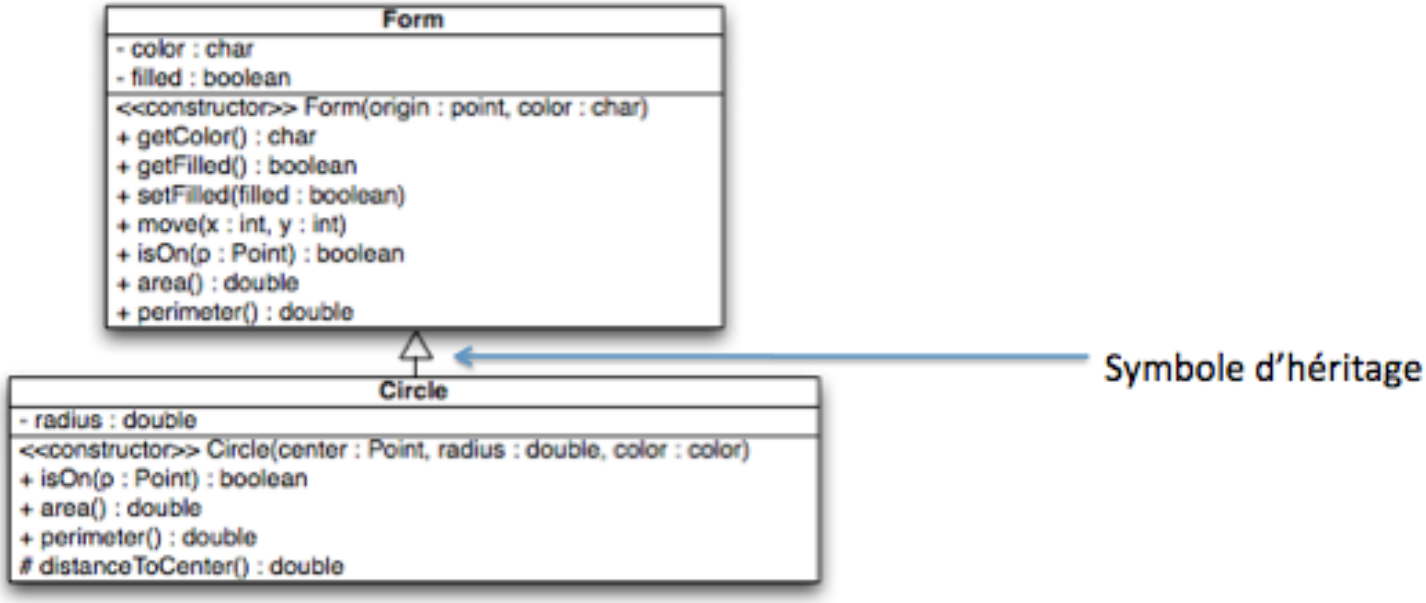
- **+** : public
- **-** : private
- # : protected
- ~ : package-private

Les types sont précisés à l'aide d'un " : "

Le constructeur est précédé par « *constructor* »

Les membres statiques sont soulignés

Héritage



- **+** : public
- **-** : private
- **#** : protected
- **~** : package-private

Les méthodes redéfinies sont réécrites dans la classe fille ainsi que les méthodes et attributs supplémentaires.

Association, composition, agrégation

- Une **association** est une relation sémantique entre des classes qui définit un ensemble de liens.
  - Un homme est associé à son épouse
- Une **agrégation** est une association dans laquelle il y a un lien d'appartenance entre les deux objets associés (contenant/contenu, possession,...)
  - Un home possède un compte en banque
- Une **composition** (ou agrégation forte) est une agrégation dans laquelle la disparition du composite entraine la disparition des composants.
  - Si un arbre meurt, ses feuilles ne servent plus à rien (on ne peut pas les mettre sur un autre arbre, au contraire de roues sur une voiture)

Il s'agit surtout d'une différente sémantique qui impliquera des changements dans votre implémentation au niveau du cycle de vie des objets.

Association, composition, agrégation (suite)

- **Association :** Une personne travaille pour une et une seule compagnie.

- **Agrégation :** Une personne possède entre 0 et n comptes en banque

- **Composition :** Une personne a un et un seul cerveau

Cardinalités :

- $*, n, m..*$ où $n > 0$ et $m >= 0$
- Par défaut, la cardinalité est 1
- Dans une composition, la cardinalité du côté de l'agrégat ne peut être que 1 ou 0..1

Le nom de l'association est facultatif

Association multiples

En Java, les associations multiples peuvent par exemple s'implémenter dans des collections comme *ArrayList*, *HashMap*,...

Par exemple :

```
import java.util.*;

...

ArrayList<Person> myList = new ArrayList<Person>();
myList.add(new Person("Jean"));
myList.add(new Person("Hugues"));

for(Person p : myList) {
    System.out.println(p);
}
```

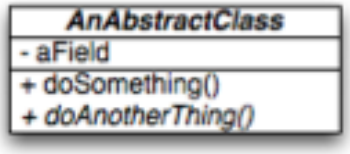
Classes abstraites

Une **classe abstraite** ne peut être instanciée mais peut être sous-classée. Elle peut contenir ou non des méthodes abstraites.

Une **méthode abstraite** est une méthode déclarée mais non implémentée.

Si une classe inclut des méthodes abstraites, elle doit être abstraite.

Une classe qui hérite d'une classe abstraite doit implémenter toutes les méthodes abstraites de la classe parente. Sinon, elle doit être déclarée abstraite.




```
public abstract class AnAbstractClass {
    private int aField;

    ...

    public void doSomething(){...}
    public abstract void doAnotherThing();

}
```

UML : Exemple de diagramme de séquence