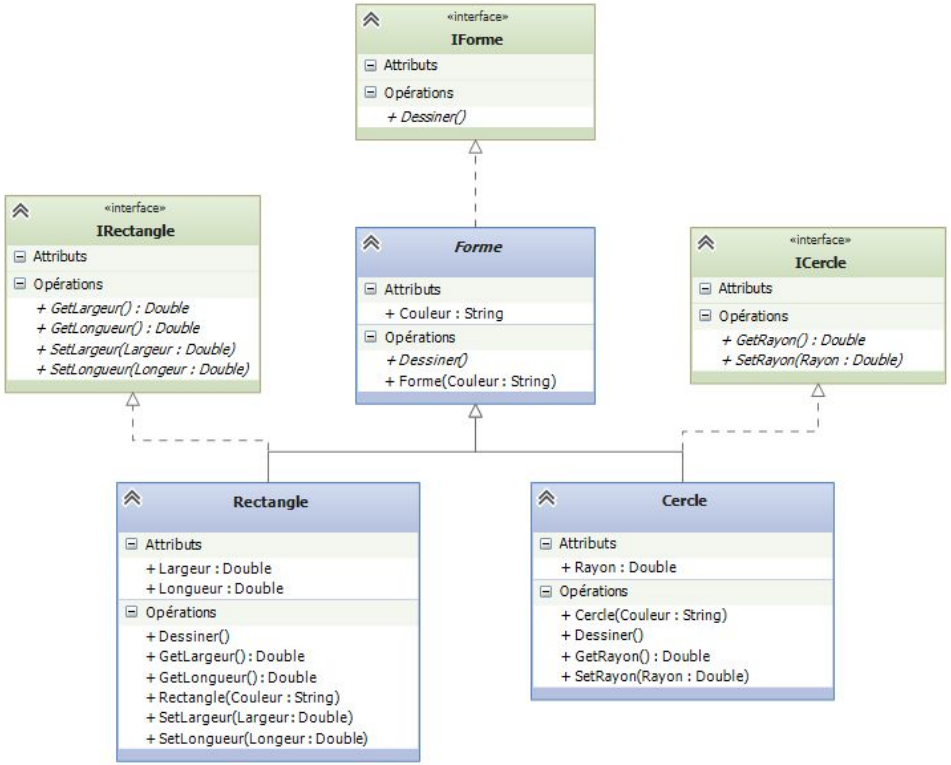
INTERFACE

INTRODUCTION À L'APPROCHE ORIENTÉ OBJET

- Définition et contrainte
- Interface et polymorphisme
- Classes abstraites vs Interfaces

DÉFINITION ET CONTRAINTE



Les interfaces décrivent un groupe de fonctionnalités communes qui peuvent appartenir à n'importe quelle classe.

Les interfaces sont composées de méthodes* qui ne contiennent pas de fonctionnel.

D'ailleurs, on les compare souvent à un contrat.

Lorsque nous disons qu'une classe implémente une interface, cela signifie que la classe s'engage à implémenter tous les membres définis par l'interface avec le modificateur d'accès public.

Elles sont également utiles, pour les langages qui n'implémente pas l'héritage multiple.

* Peut varier en fonction des langages de programmation

INTERFACE ET POLYMORPHISME

Dans un contrat, nous avons des obligations mais nous en retirons le plus souvent un certain bénéfice.

Lorsque nous implémentons une interface nous sommes tenu d'implémenter les fonctionnalités de la dite interface.

Mais d'un autre côté, nous y gagnons un nouveau type. Ce nouveau type sera donc utilisable avec le principe de polymorphisme d'héritage.

Il est bon à rappeler, que l'utilisation du polymorphisme d'héritage limite aux fonctionnalités du type de la variable.

CLASSES ABSTRAITES VS INTERFACES

Reste à ne pas confondre interfaces et classes abstraites!!

Car bien que pouvant être très proche dans le concept d'implémentation, plusieurs notions différencie l'utilisation des classes abstraites par rapport aux interfaces.

Cas	Interface	Classe abstraite
Les classes peuvent Hériter/implémenter plusieurs …	OUI	Dépend du langage de programmation
Peut contenir des variables membres	NON	OUI
Peut contenir des blocs d'instructions	NON	OUI
Peut contenir des membres private, protected ou package	NON	OUI
Peut contenir des classes imbriquées	NON	OUI
…	…	…