## **UML:** relations

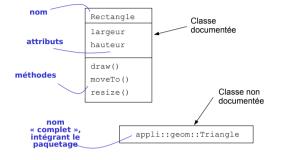
Programmation Orientée Objet

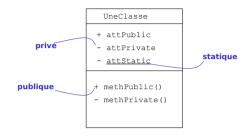
Jean-Christophe Routier Licence mention Informatique Université des Sciences et Technologies de Lille



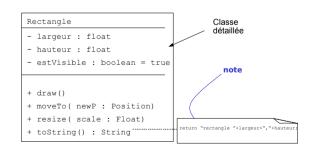
Informatique

#### Classes





USTL - Licence Informatio



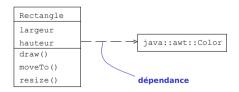
## Relations

Identifier les classes ne suffit pas, elles coopèrent/interagissent entre elles, il faut exprimer ces relations (le plus souvent binaires).

- Dépendances
- relations d'utilisation
- Généralisations
- Associations
  - relations structurelles, connexion sémantique

## Dépendance

Exprimer le fait qu'une classe en utilise une autre. Toute modification sur la classe utilisée peut avoir un impact sur la classe utilisante.



UML : relations

## **Association**

C'est une relation structurelle qui exprime une relation sémantique entre (le plus souvent) deux classes.

Elle est le plus souvent réflexive.

On peut la compléter de 4 informations :

- Nom
- Rôles
- Multiplicité
- Agrégation

USTL - Licence Informatique

Programmation Orientée Ob

Cardinalités

cardinalités

nom

direction

Personne

employé

employé

employeur

Entreprise

USTL - Licence Informatique

Programmation Orientée Objet

association

UML : relat

## Cardinalités

- Définissent la multiplicité des rôles
- Une cardinalité à une extrémité signifie qu'à l'autre extrémité ce nombre d'éléments doit exister pour tout objet de la classe.
- Expressions possibles :
  - n : exactement n
  - n..m : de n à m
  - \* : quelconque (équivalent à « 0..n » ou « 0.. »)
  - n..\* : *n* ou plus
  - liste de cardinalités : 1..2,3..5 = 1 à 5 sauf 4

STL - Licence Informatique

Programmation Orientée Obje

UML : relations

Il peut exister plusieurs relations entre les mêmes classes

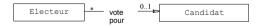


UML : relatio

## Navigabilité restreinte

Rendre unidirectionnelle la relation

Pour indiquer que les instances d'une classe ne "connaissent" pas les instances d'une autre.

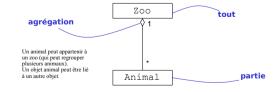


UML : relatio

## Agrégation/Composition

Association « tout/partie », relation de possession « has-a »

**Agrégation :** le tout est responsable de la gestion de ses parties. Relation de subordination.



L - Licence Informatique Programmation Orientée Ob

USTL - Licence Informatique

rogrammation Orientée Objet

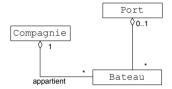
USTL - Licence Informatique

rogrammation Orientée Objet

12

UML : relation

# la partie est éventuellement partagée

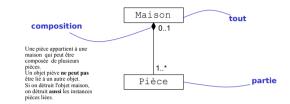


USTL - Licence Informatique

Programmation Orientée Obje

HML · relations

#### Composition: agrégation forte, la partie n'est pas partagée



USTL - Licence Informatique Programmation Orientée Objet

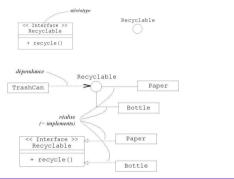
## Exemple : compteur Counter 4 affiche ounterGraphicInterface value : int incF:IncrementFunction getValue() : int Javax::swing::JFrame increment() initValue(value:int) « interface » value = incF.increment(value); IncrementFunction SimpleIncrement ModularIncrement AnotherIncrement

USTL - Licence Informatique Programmation Orientée Objet

grid::GridDisplayer grid::Grid <interface> Fish wator Position Shark Tuna Fish grid::Grid Environment fait évoluer grid::GridDisplayer Wator grid::GraphicalGridDisplayer grid::TextGridDisplayer USTL - Licence Informatique

UML : relations

## Interfaces



USTL - Licence Informatique Programmation Orientée Objet