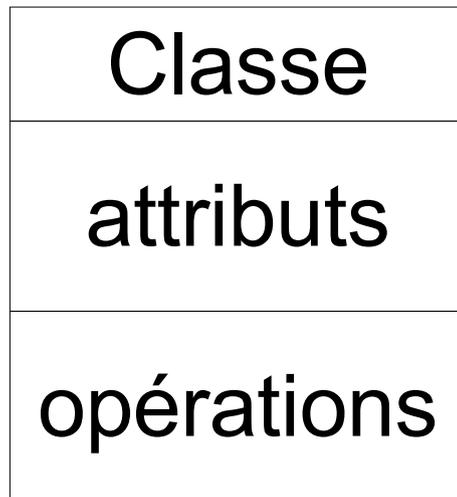


Diagramme de classes

- Diagramme entité-relation essentiel et riche.
- Représente les classes et leurs associations.
- Une classe est un type possédant des attributs et des opérations.
- Les associations sont des abstractions des liens existants entre objets.
- Pour un projet il peut exister un grand nombre de diagramme de classes.

Les classes



UML suggère que le nom d'une classe:

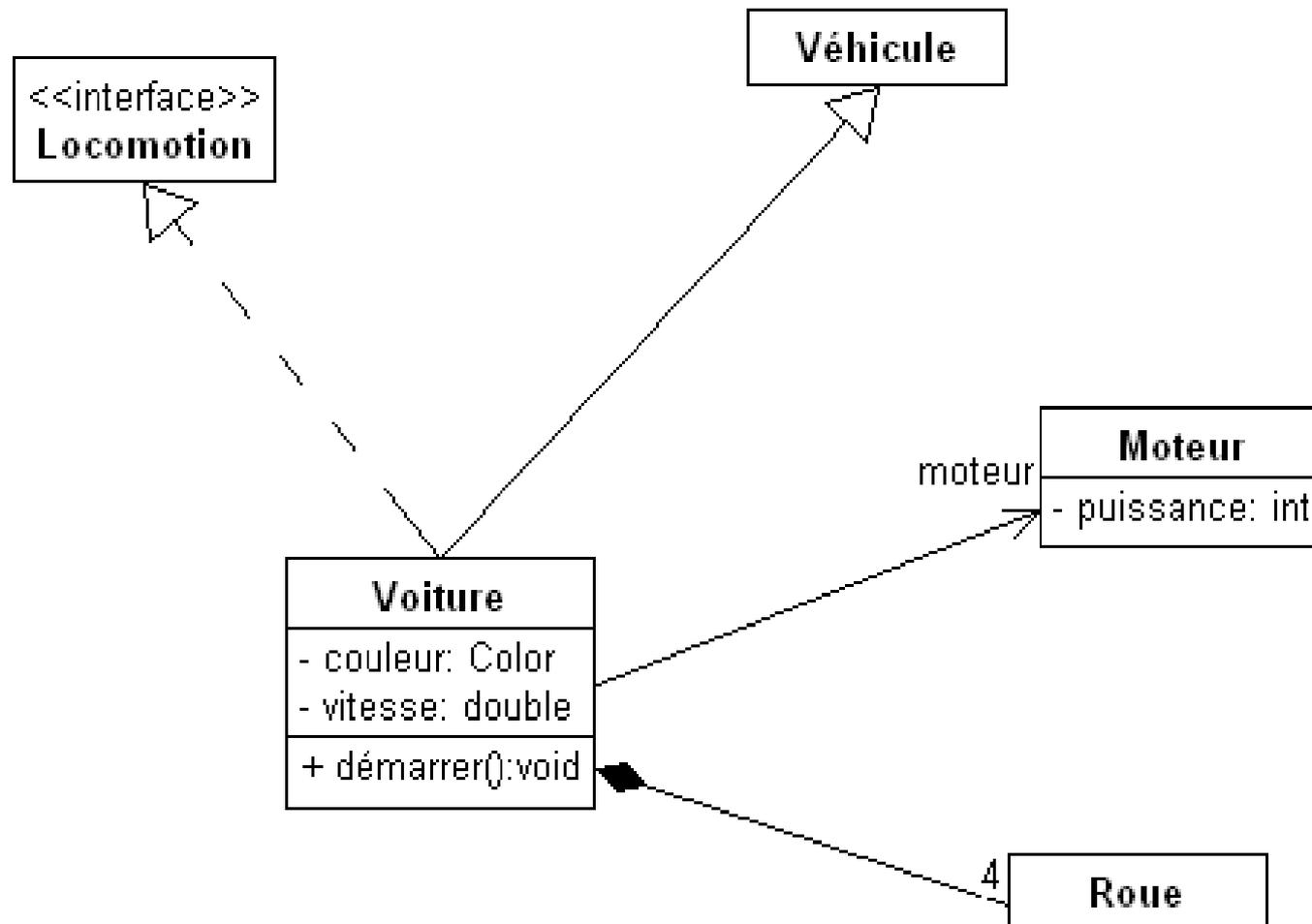
- Débute par une majuscule
- Soit centré dans le compartiment supérieur
- Soit en caractères gras
- Soit en italique s'il s'agit d'une classe abstraite

Les attributs peuvent aussi être représentés au moyen d'une notation relationnelle

UML suggère que les attributs et opérations:

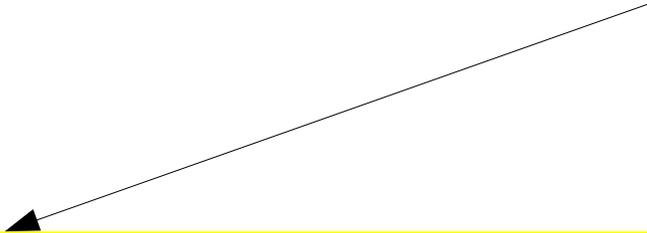
- Débute par une minuscule
- Soit cadrés à gauche
- De montrer les détails quand le contexte le demande

Exemple



Les attributs

- + : public
- : private
- # : protected
- ~ : package(default)



```
visibilité
```

```
nom : type multiplicité  
= valeur initiale  
{contraintes}
```

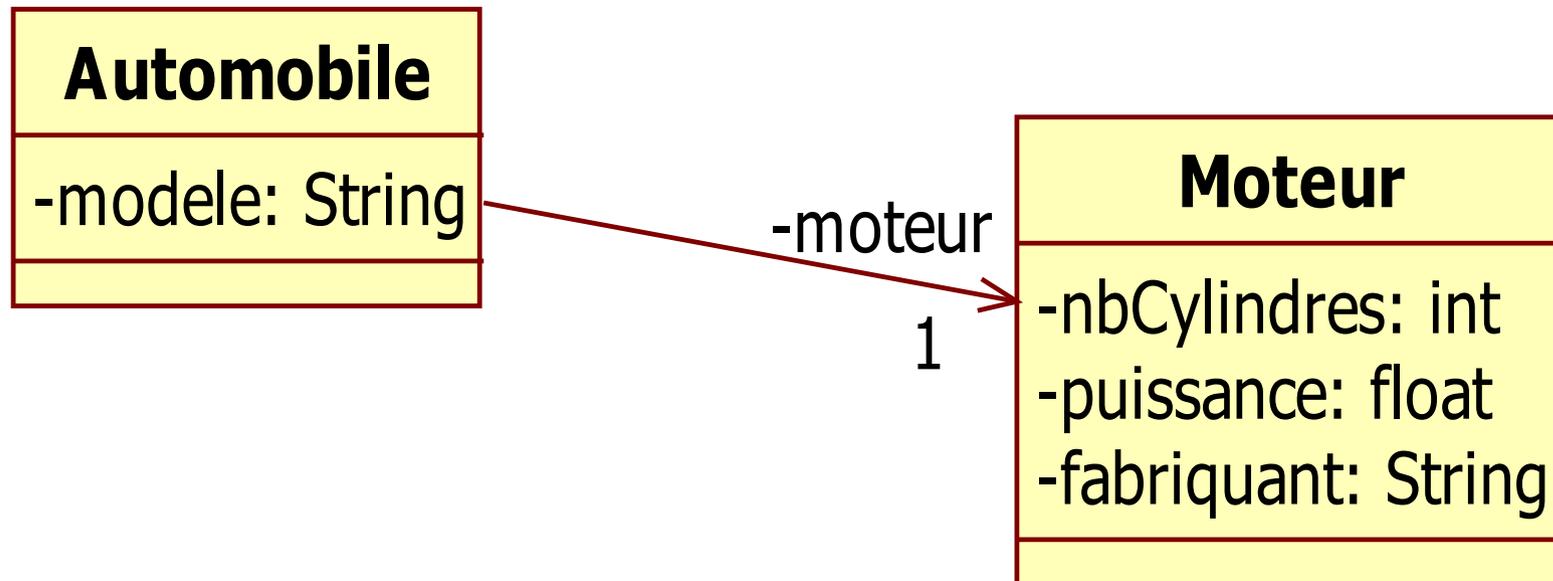
Les attributs - Exemple

ClasseExemple

- attrSimple : String = «Valeur Initiale»
- /attrDerive : boolean
- tableauP : Personne[10]
- tableauInt : Integer[*]
- liste : Salarie[1..*]{unique, ordered}
- attrStatique

Attributs par relation

- Notation utilisée pour les attributs complexes



Les opérations

```
visibilité nom(paramètres) : type_retour
```

```
par1 : type, par2 : type ...
```



Les opérations - Exemple

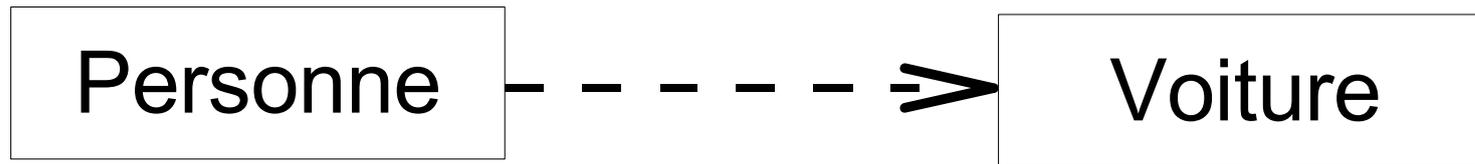
Fenetre

```
+getTaille(): Rectangle  
+setTaille(x: int, y: int): void  
+getComposants(): Composant[0..*]  
#paint(): void  
+setVisible(visible: boolean = true)  
+formatTitle(): void
```

Les relations

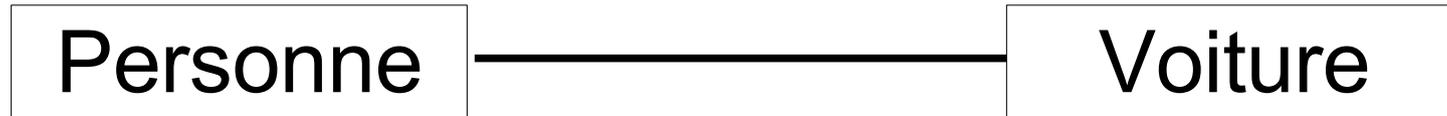
	dépendance
	association
	agrégation
	composition
	généralisation
	implémentation

La dépendance



- Relation la plus faible entre classes
- Cette association signifie qu'un objet dépend d'un objet source.
- Peut s'interpréter comme une relation de type « utilise un »

L'association



- Relation plus forte
- Cette association signifie qu'un objet est en relation avec un autre objet pendant un certain laps de temps, sans qu'ils soient étroitement associés
- Peut s'interpréter comme une relation de type « a un »

L'association

- Navigabilité



- Multiplicités

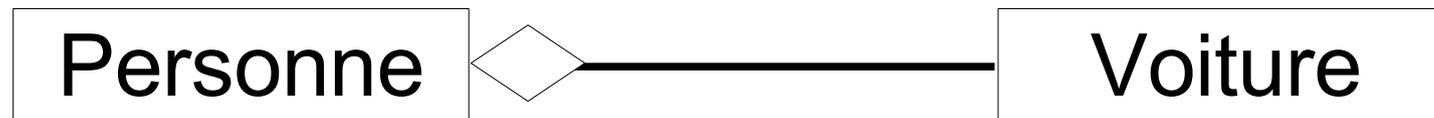
0 .. 1 : 0 ou 1 instance

* : de 0 à n instances

n : exactement n instances

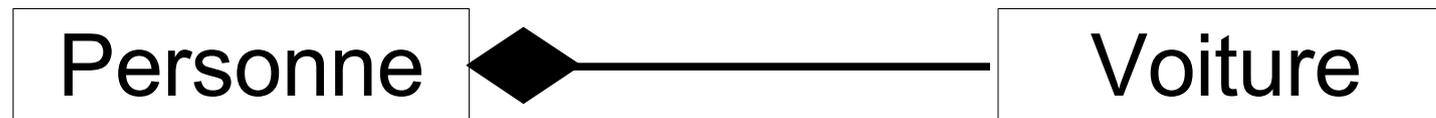
n..m : entre n et m instances

L'agrégation



- L'agrégation implique une notion de propriété plus forte que l'association
- Peut s'interpréter comme une relation de type « est propriétaire d'un »

La composition



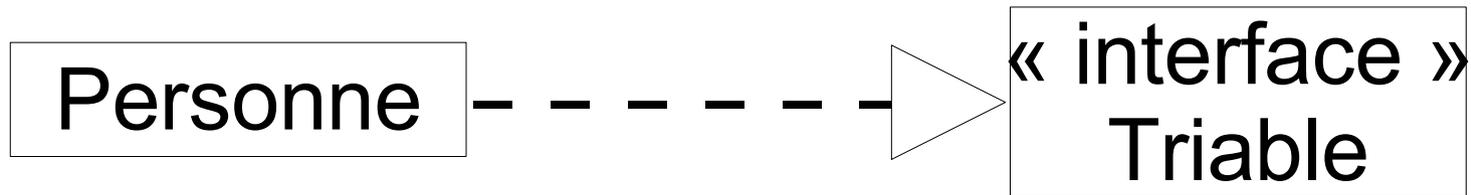
- Relation la plus forte
- Indique la possession totale d'une classe par l'autre
- Peut s'interpréter comme une relation de type « fait partie de »

La généralisation



- Permet d'indiquer qu'une classe constitue un cas plus général d'une autre classe
- Peut s'interpréter comme une relation de type « est un »

L'implémentation



- Une interface spécifie un contrat que respecte la classe qui l'implémente