Préalable

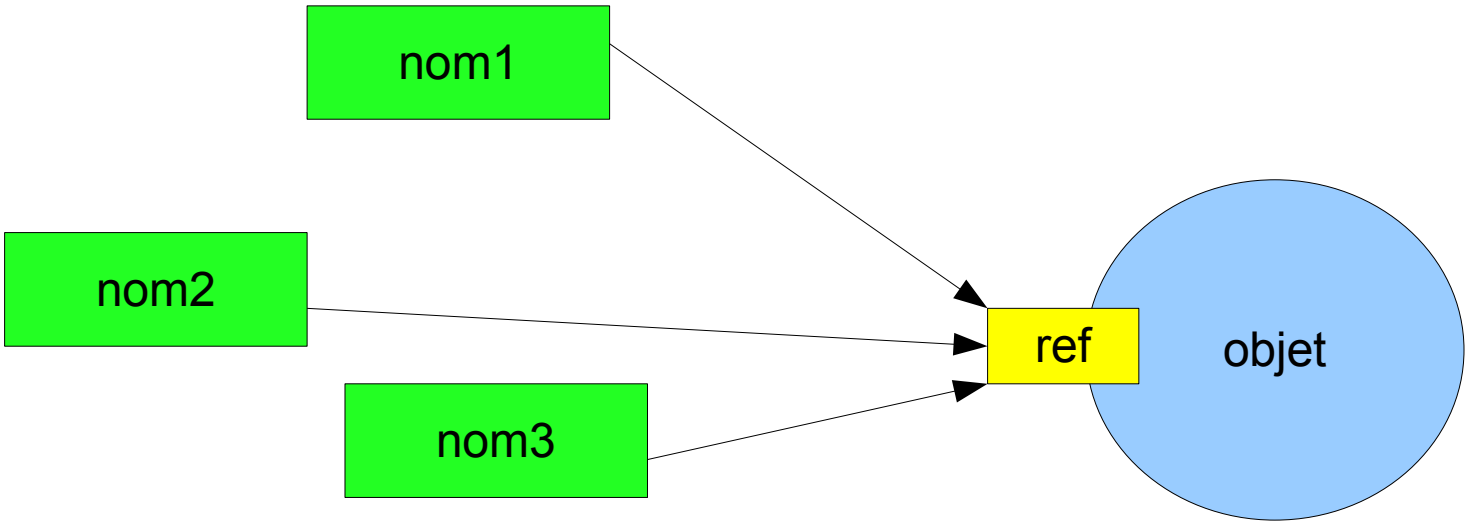
- Description des objets à l'aide du trio

  <entité, attributs, valeurs>

- Chaque objet se caractérise par un ensemble d'attributs qui lui est propre
- Les attributs peuvent être attributs d'un grand nombre d'objets différents
- Le monde des attributs est moins diversifié que le monde des objets

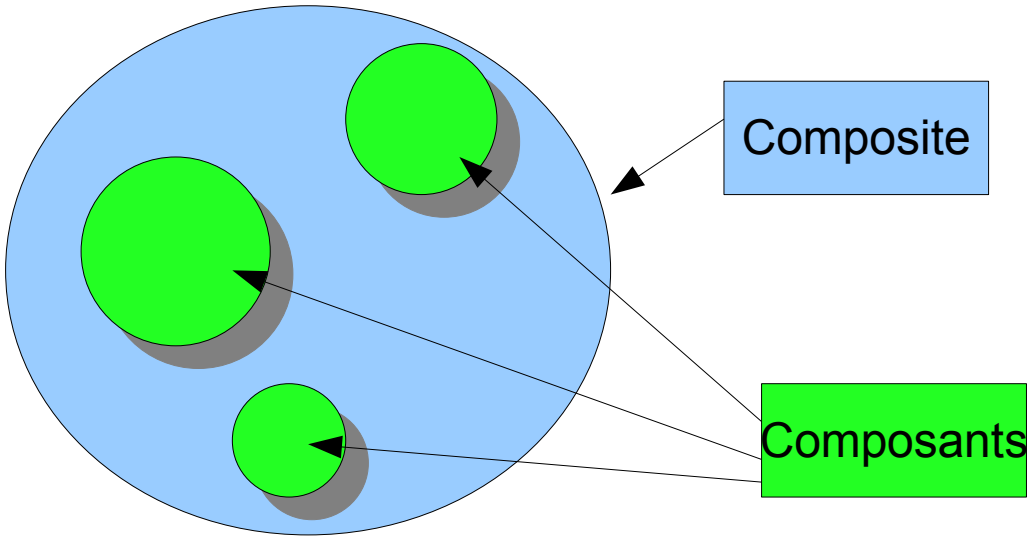
Stockage des objets en mémoire

- Ensemble de couples <attribut : valeur> stocké à une adresse physique connue
- L'adresse physique de l'objet est sa référence
- Un même objet, une seule référence peut avoir plusieurs noms



Objet composite

- Certains objets sont des compositions d'autres objets
- Les objets composants se trouvent contenus dans d'autres et ne sont accessibles qu'à partir de ces autres. Leur existence dépend entièrement de celles des objets qui les contiennent



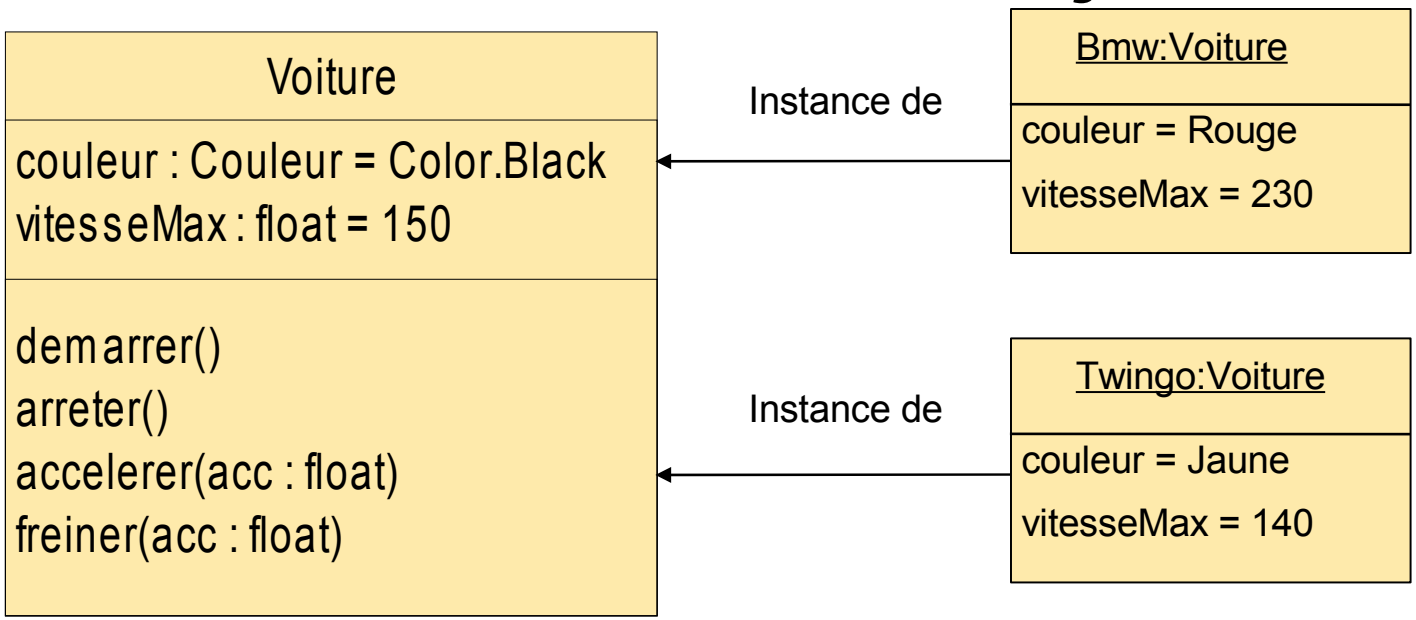
Activité des objets

- Les objets sont dynamiques
- La valeur de leurs attributs changent dans le temps. Ils changent d'état
- Le changement d'un attribut n'affecte en rien son adresse
- L'objet naît, vit une succession de changements d'états et disparaît de la mémoire
- Les opérations (ou méthodes) sont responsables du changement d'état

Notion de classe

- La classe unit tous les attributs et méthodes de l'objet
- Il s'agit d'un contrat logiciel qui lie les attributs et les méthodes de l'objet
- Tout attribut sera typé comme indiqué dans sa classe
- Toutes les méthodes affectant l'objet seront uniquement celles prévues dans sa classe

Classe et objets



- Valorisation d'un attribut : **objet.attribut = valeur**
  - Ex : Bmw.couleur = Rouge
- Utilisation d'une Méthode : **objet.méthode**
  - Ex : Bmw.demarrer()

Des concepts au code

```
class Voiture {

   Couleur couleur = Color.black;
   float   vitesseMax = 150;

   void demarrer() {...}
   void arreter() {...}
   void accelerer(float acc) {...}
   void freiner(float frein) {...}

   void setVitesseMax (int vitesse){
     vitesseMax = vitesse;
   }
   int getVitesseMax () {
     return vitesseMax;
   }
}
```

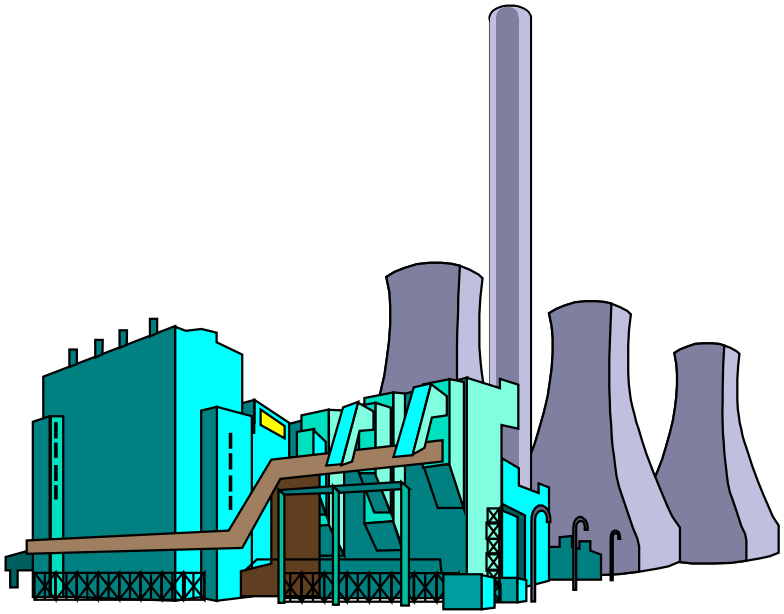
Attributs

Méthodes

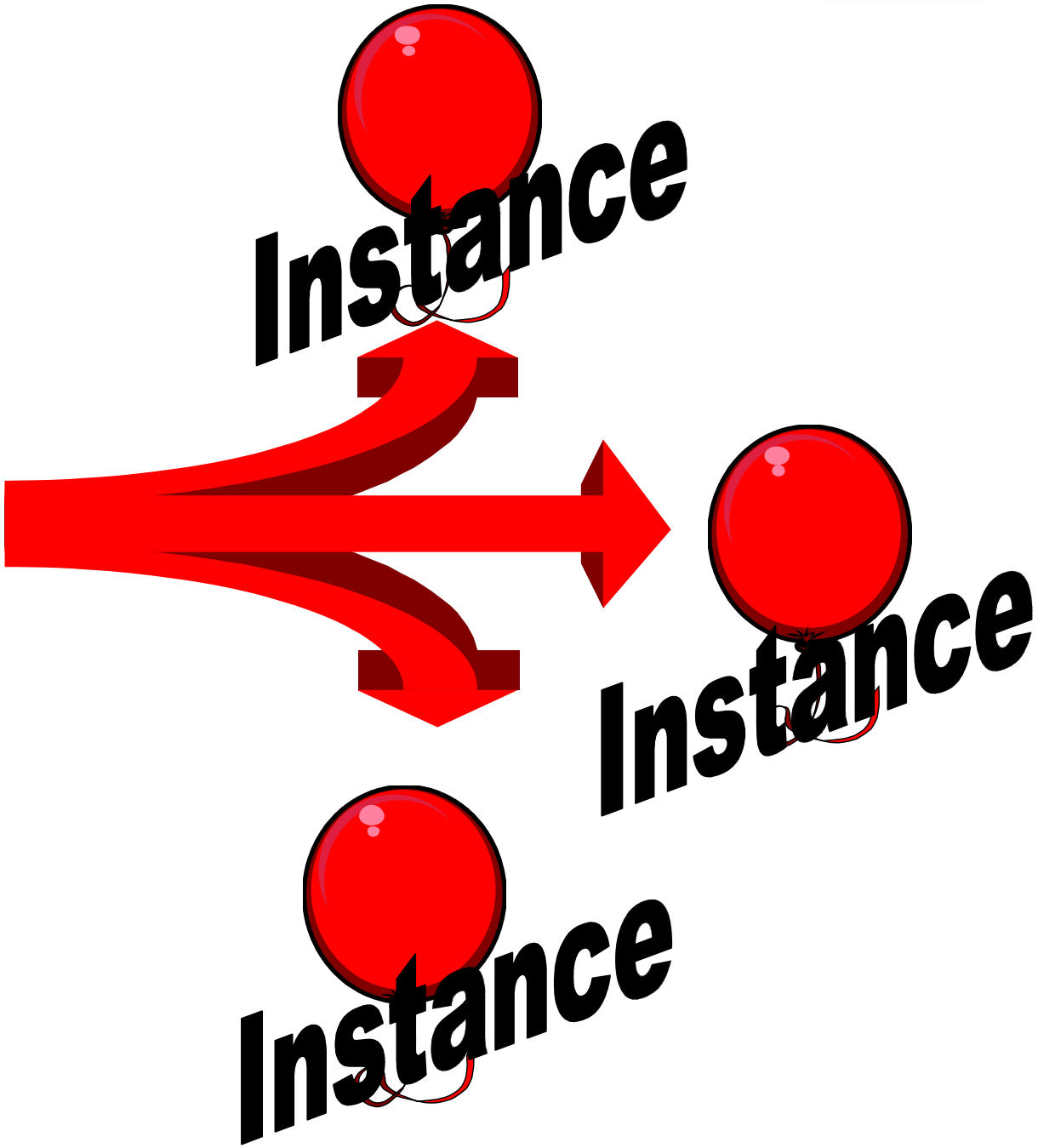
un objet

```
...
Voiture bmw = new Voiture();
bmw.setVitesseMax(200);
bmw.demarrer();
bmw.accelerer(65);
...
```

Classe et instances



Classe



Communication entre objets

- Le seul mode de communication entre deux objets revient à la possibilité pour le premier de d'exécuter une méthode sur le second

Objet feu tricolore

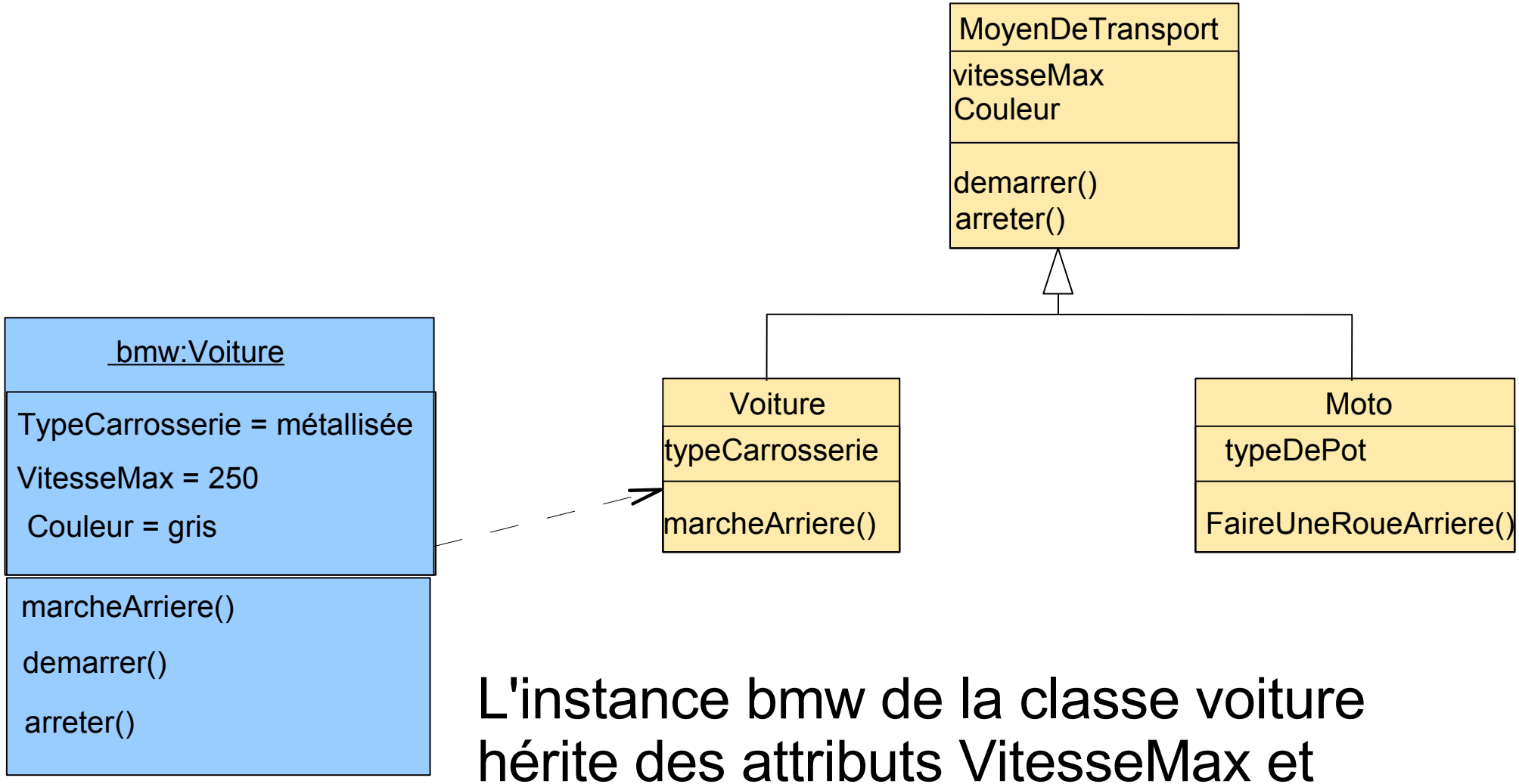
```
...
bmw.accelerer(50);
...
```

Ici l'objet feu tricolore envoie le message « accelerer » à l'objet bmw

Héritage

- Une pratique courante de l'orienté objet est d'organiser les classes de manière hiérarchique
- Ce classement de type taxonomique va du plus général au plus particulier
- Un objet instance d'une classe sera, à la fois instance de cette classe et instance de toutes les classes dont elle hérite

Héritage

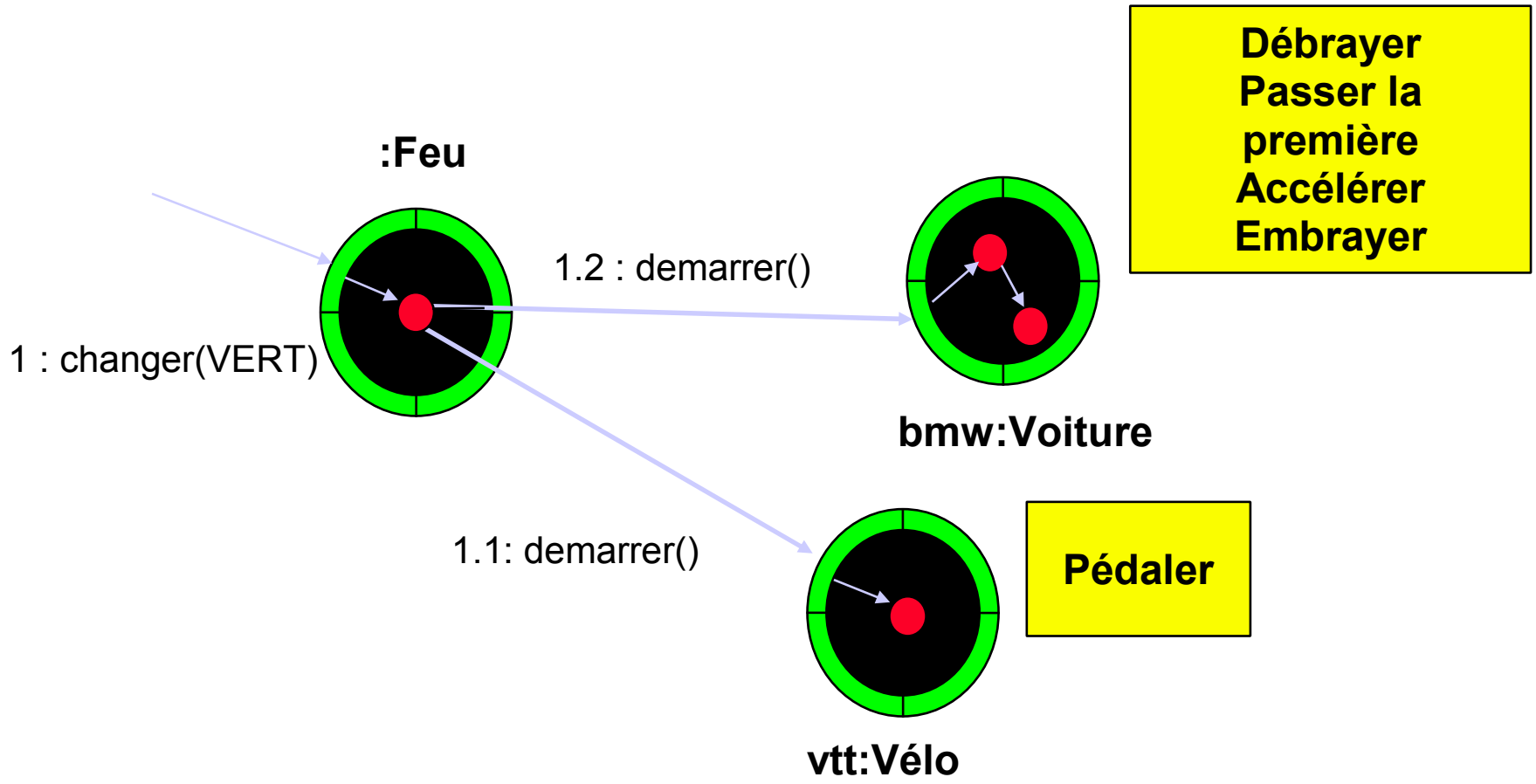


L'instance bmw de la classe voiture hérite des attributs VitesseMax et Couleur, et des méthodes demarrer() et arreter()

Polymorphisme

- Le polymorphisme est une conséquence directe de l'héritage
- Un objet est du type de la classe qu'il instancie et du type de toutes les « super-classes », mais il possède la « particularité » de la classe dont il hérite directement
- Permet à une même méthode d'avoir un comportement différent selon le type de l'objet

Conséquence du polymorphisme



Méthode et signature

- Le nom de la méthode et la liste de ses arguments constituent la signature de cette méthode
- Le langage d'interaction entre deux objets se réduit à l'utilisation des méthodes et de leurs signatures
- Il n'est pas possible d'avoir deux méthodes avec la même signature

Le constructeur

- Méthode particulière, de même nom que la classe et sans retour (même pas void)
- Cette méthode est appelée lors de la création d'un objet
- Elle a pour mission d'initialiser les attributs de l'objet
- Une version par défaut est toujours fournie par les langages de programmation
- C'est une méthode souvent surchargée

Attributs et méthodes statiques

- Ce sont des attributs dont la valeur est commune à l'ensemble des objets (instances) de la classe
- Ce sont des méthodes qui peuvent s'exécuter directement à partir de la classe, en l'absence même de tout objet
- Une méthode statique ne peut utiliser que des attributs et des méthodes statiques

Encapsulation des attributs

- L'accès aux attributs d'une classe ne doit se faire que par l'utilisation des méthodes de cette classe
- Les attributs doivent donc être déclarés privés, c'est à dire accessible que de la classe
- Leur simple lecture ou modification se fera par des méthodes d'accès
- Les classes doivent être responsable de leurs intégrités

Encapsulation des méthodes

- Une partie des méthodes d'une classe peuvent être déclarées privées
- Les méthodes publiques de la classe constituent l'interface et sont les seuls messages possibles
- Les méthodes privées constituent l'implémentation et sont appelées des méthodes publiques ou d'autres méthodes privées de la classe