

# Initiation à la POO

Jean-Christophe Damichel

Lycée Eiffel

03 juin 2014



# Sommaire

- 1 Présentation
- 2 Sécurité de la POO

# Présentation de la POO

- Au lieu de découper le programme en actions (fonctions), on détermine des objets
- Les objets ont des propriétés ou attributs (couleur, masse,..)
- Il y a des fonctions qui agissent sur les attributs des objets (appelées méthodes)
- La définition des attributs d'un objet et les méthodes qui agissent dessus sont regroupées dans une classe
- Lorsqu'on crée une classe, on a un nouveau type de données

# Structure d'une classe

```
class Compte {  
    int montant;  
  
    void retirer(int s) {  
        this.montant = this.montant - s;  
    }  
  
    int getMontant() {  
        return this.montant;  
    }  
  
    Compte(int s) {  
        this.montant = s;  
    }  
}
```

- *montant* est une propriété de l'objet
- *retirer* et *getMontant* sont des méthodes; *getMontant* est un accesseur, *retirer* est un mutateur
- *Compte* est un constructeur appelé lorsqu'on crée l'objet (pas de valeur de retour et même nom que la classe)
- *this* représente l'objet lui-même



# Utilisation d'une classe

```
void setup() {  
    Compte cb = new Compte(50);  
    cb.retirer(26);  
    println(cb.getMontant());  
}
```

- On déclare l'objet avec *Compte cb*; : réservation de mémoire, la variable *cb* est une référence vers l'adresse où se trouve l'objet. *cb* n'est pas un objet mais une référence.
- On crée l'objet qui est de type *Compte* avec le constructeur : on dit que l'objet est une instance de la classe *Compte*
- On appelle la méthode *retirer* de l'objet grâce à la notation .
- On appelle l'accessor de l'objet

# Exemple de sécurité de la POO

Type d'erreur qui peut arriver :

```
void setup() {  
    Compte cb = new Compte(50);  
    cb.montant = -5;  
    println(cb.getMontant());  
}
```

- Si un programmeur a accès directement aux attributs d'un objet, il peut faire des erreurs en les modifiant
- On rend les attributs inaccessibles à l'extérieur de la classe avec *private*
- On utilise des mutateurs qui contrôlent les modifications des attributs

# Exemple de sécurité de la POO

```
class Compte {
    private int montant;

    void retirer(int s) {
        if(this.montant-s <0) println("Impossible !");
        else this.montant = this.montant-s;
    }

    int getMontant() {
        return this.montant;
    }

    Compte(int s) {
        this.montant = s;
    }
}

void setup() {
    Compte cb = new Compte(50);
    cb.montant = -5;
    cb.retirer(152);
    println(cb.getMontant());
}
```

La ligne `cb.montant = -5;` provoque une erreur de compilation (en vrai Java mais pas avec Processing qui a un fonctionnement particulier) grâce à *private*

