

## ITI 1521. Introduction à l'informatique II †

Marcel Turcotte  
(contributions de R. Holte)

École de science informatique et de génie électrique  
Université d'Ottawa

Version du 30 janvier 2012

†. Pensez-y, n'imprimez ces notes de cours que si c'est nécessaire!

Marcel Turcotte ITI 1521. Introduction à l'informatique II

## Programmation orientée objet

Un logiciel est vu comme une collection d'objets qui interagissent, les uns avec les autres, afin de résoudre un problème commun.

Un objet possède :

- ▶ des propriétés qui définissent son état
- ▶ des comportements : ce que l'objet peut faire, ses réponses aux requêtes

Marcel Turcotte ITI 1521. Introduction à l'informatique II

## Interface

- ▶ Les éléments (variables et méthodes) publics définissent l'**interface** de la classe.
- ▶ Afin d'utiliser un objet (ou une classe) tout ce qu'il faut c'est en connaître l'interface
- ▶ **L'interface d'une classe doit être définie soigneusement.**
- ▶ Seules les changements portant sur l'interface de la classe affecteront les autres parties du système logiciel.

Marcel Turcotte ITI 1521. Introduction à l'informatique II

```
public class Ticket {  
    private static int lastSerialNumber = 0;  
    private int serialNumber;  
  
    public Ticket() {  
        serialNumber = lastSerialNumber;  
        lastSerialNumber++;  
    }  
    public int getSerialNumber() {  
        return serialNumber;  
    }  
}
```

Marcel Turcotte ITI 1521. Introduction à l'informatique II

## Variable d'instance et variable de classe

**Variable d'instance** : Chaque objet réserve des espaces mémoire pour ses variables d'instance (propriétés, état de l'objet, ce que chaque objet mémorise)

**Variable de classe** : Emplacement mémoire **unique** associé à la classe (et non pas aux objets)

Marcel Turcotte ITI 1521. Introduction à l'informatique II

## Méthode d'instance et variable de classe

**Méthode d'instance** : Évaluée dans un contexte objet, a donc accès aux variables d'instance de l'objet

**Méthode de classe** : N'a pas accès aux variables d'instances

Marcel Turcotte ITI 1521. Introduction à l'informatique II

## Qu'en pensez-vous?

```
public class Ticket {  
    private static int lastSerialNumber = 0;  
    private int serialNumber;  
  
    public Ticket() {  
        serialNumber = lastSerialNumber;  
        lastSerialNumber++;  
    }  
    public static int getSerialNumber() {  
        return serialNumber;  
    }  
}
```

Ticket.java:10: non-static variable serialNumber cannot be referenced from a static context  
return serialNumber;  
^

1 error

Marcel Turcotte ITI 1521. Introduction à l'informatique II

## Exemples de variables et méthodes de classe

```
public class Math {  
  
    public static final double E = 2.718281828459045;  
  
    static int min( int a, int b ) { ... }  
    static double sqrt( double a ) { ... }  
    static double pow( double a, double b ) { ... }  
  
    public static double toDegrees( double angrad ) {  
        return angrad * 180.0 / PI;  
    }  
    ...  
}
```

Marcel Turcotte ITI 1521. Introduction à l'informatique II

## Exemple : représenter le temps

On souhaite modéliser le temps sur une période de 24 heures.  
En particulier, il nous faut représenter les **informations** suivantes :

- heures : sur l'intervalle 0 ... 23 (inclusivement)
- minutes : sur l'intervalle 0 ... 59 (inclusivement)
- secondes : sur l'intervalle 0 ... 59 (inclusivement)

## Déclaration de classe

La déclaration d'une classe (aspect déclaratif) débute par le mot réservé `class` suivi du nom de la classe (un identificateur dont la première lettre est une majuscule, on choisi en général un nom singulier).

Est-ce que cette déclaration est valide ?

```
public class Time {  
}
```

Cette déclaration peut être mise dans un fichier nommé **Time.java**, puis compilé,

```
> javac Time.java
```

## Déclaration de classe

Peut-on utiliser la classe **Time** ? Comment ?  
Est-valide ?

```
class Test {  
    public static void main(String[] args) {  
        Time t0;  
    }  
}
```

## Déclaration de classe

Est-valide ?

```
class Test {  
    public static void main(String[] args) {  
        Time t0;  
        t0 = new Time();  
    }  
}
```

Hum, mais il n'y a pas de constructeur !

## Déclaration de classe : constructeur

En effet, Java introduit automatiquement un constructeur par défaut :

```
public class Time {  
    public Time() {  
    }  
}
```

## Déclaration de classe : constructeur

Attention ! Le constructeur par défaut existe, à moins que vous définissiez votre propre constructeur :

```
public class Time {  
    private int hours;  
    private int minutes;  
    private int seconds;  
    public Time( int h, int m, int s ) {  
        hours = h;  
        minutes = m;  
        seconds = s;  
    }  
}
```

Ainsi, le second énoncé produira une erreur de compilation.

```
Time t;  
t = new Time();
```

## Constructeur

Qu'est-ce qu'un constructeur ?

Se comporte comme une méthode d'instance ayant des propriétés spéciales :

- ▶ Ne peut être appelée qu'une seule fois et que dans le contexte "new ...";
- ▶ Le constructeur porte le nom de la classe;
- ▶ N'a pas de valeur de retour;
- ▶ Java fournit un constructeur par défaut, mais seulement si vous ne définissez aucun constructeur.

**Puisque le constructeur est appelé au moment de la création de l'objet seulement, il sert généralement à initialiser le contenu des variables d'instances.**

## <<Getters>>

```
public int getHours() {  
    return hours ;  
}  
  
public int getMinutes() {  
    return minutes ;  
}  
  
public int getSeconds() {  
    return seconds ;  
}
```

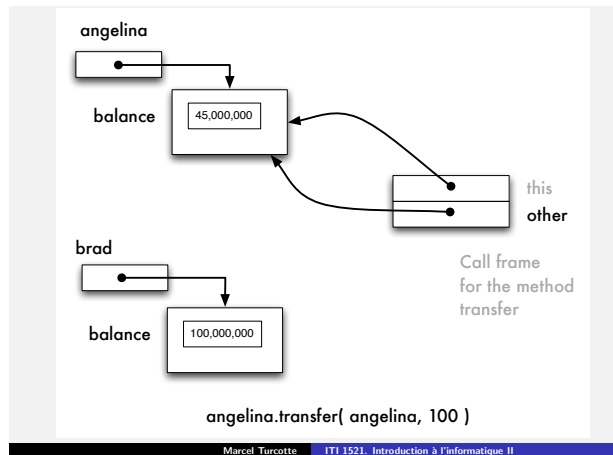
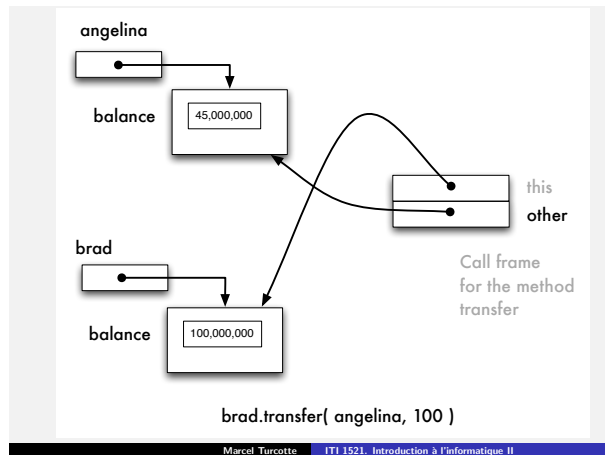
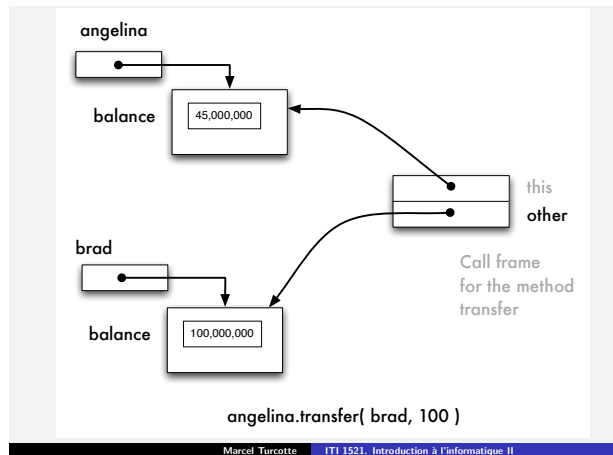
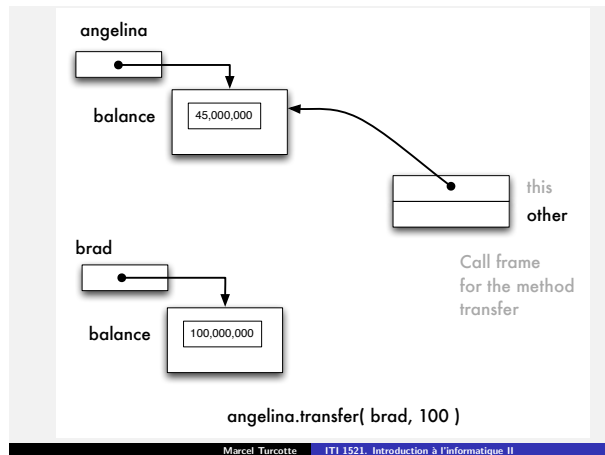
```
public boolean equals( Time t ) {
    return (( hours == t.getHours() ) &&
           ( minutes == t.getMinutes() ) &&
           ( seconds == t.getSeconds() ) );
}
```

```
public boolean equals( Time t ) {
    return (( hours == t.hours ) &&
           ( minutes == t.minutes ) &&
           ( seconds == t.seconds ) );
}
```

this?

<< this >> est une référence vers cet objet.  
Exemple : BankAccount

```
public class BankAccount {
    private double balance;
    // ...
    public boolean transfer( BankAccount other , double a ) {
        if ( this == other ) {
            return false;
        }
        // ...
    }
}
```



this?

```
public class Date {
    private int day;
    private int month;
    public Date( int day , int month ) {
        this.day = day;
        this.month = month;
    }
    // ...
}
```

-  E. B. Koffman and Wolfgang P. A. T.  
*Data Structures : Abstraction and Design Using Java.*  
John Wiley & Sons, 2e edition, 2010.
-  P. Sestoft.  
*Java Precisely.*  
The MIT Press, second edition edition, August 2005.
-  D. J. Barnes and M. Kölling.  
*Objects First with Java : A Practical Introduction Using BlueJ.*  
Prentice Hall, 4e edition, 2009.
-  Petar Tahchiev, Felipe Leme, Vincent Massol, and Gary Gregory.  
*JUnit in Action.*  
Manning Publications Co., second edition edition, 2010.



Pensez-y, n'imprimez ces notes de cours que si c'est nécessaire !