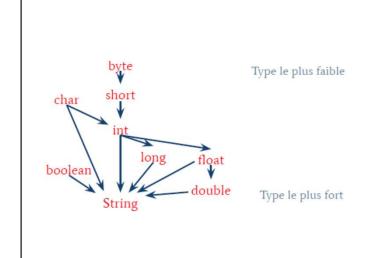
CHAPITRE II

Conversions de types et structures de contrôle en JAVA

Conversions de types en JAVA

• Conversion automatique de type

Avant d'effectuer une opération, les valeurs peuvent être converties vers un type plus fort



lors de la conversion **d'un char vers un** int c'est le code du caractère qui est renvoyé.

Ainsi par exemple 'a' a la valeur 97.

La commande java:

System.out.println('a'+3);
renvoie 100

Exemple en JAVA

```
int x = 3;
double y = x + 2.2; /* 5.2 */
String s1 = "Coucou" + x;
    /* Coucou3 */
char c = 'a';
String s2 = "Coucou" + c;
    /* Coucoua */
int d = c + 1;
    /* 98 car 'a' = 97 */
```

Conversion explicite de type

 Quand il n'existe pas de conversion implicite, on utilise la conversion explicite avec (type)

Dans ce cas, Java tronque le nombre.

```
double x = 97.7;
int y = (int)x; /* 97 */
char c = (char)y; /* 'a' car c vaut 97 */
byte b = (byte)(y * 3); /* 291 modulo 256 = 35 */
```

Il Structures de contrôle

1) Structures algorithmiques et blocs

• La structure d'un programme Java est donnée par des blocs

```
commence par un { et termine par un }
    Regroupe un ensemble de déclarations

class Test {

    public static void main(String[] args) { if(0 == 1) {

        System.out.println("Cette machine semble bizarre!");

        System.out.println("Elle pense que 0 == 1!");  // le bloc est exécuté seulement si la condition est satisfaite.
    }
}

Accolades optionnelles si une unique déclaration dans un bloc
```

- o Pour être plus précis, un bloc est simplement une déclaration...
- Même si ce n'est pas obligatoire, on indente les blocs pour la lisibilité (comme en Python)!

2) Le traitement conditionnel

• Le schéma alternatif simple :

```
• Schéma alternatif simple

• si alors ...sinon (si alors ...sinon ...)

• Parties else if et else optionnelles

If (condition1) {

Instructions 1

} else if (condition 2) {

Instructions 2

} else {

instructions 3 }
```

• Exemple en Java

```
• class Exemple {
  static void main(String[] args) {
  int a = 21*3+4;
  if(a == 67) { System.out.println("Super !");
  System.out.println("Java, c'est facile !");
   }
  }
}
```

La structure conditionnelle swich (selon)

• Cette structure conditionnelle est appelée aussi à choix multiple ou sélective car elle sélectionne entre plusieurs choix à la fois, et non entre deux choix alternatifs (comme dans le cas de la structure SI).

```
| Si val = v1 exécute instr 1 | case v1 : | instr 1 ; | break ; | Sinon si val = v3 exécute instr n | case v2 : | instr 2 ; | break ; | case v3 : | instr 2 ; | break ; | case v4 : | instr 2 ; | break ; | case v4 : | instr 2 ; | case v4 : | instr 2 ; | case v4 : | case v
```

```
switch(c) {
case 'a': System.out.println("Ceci est un a"); break; case 'b': System.out.println("Ceci est un b"); break;
...
default: System.out.println("Bizarre"); break;
}
```

Structures répétitives (boucles)

Schémas itératifs

```
Boucle while

Tant que (condition) faire

Instructions

Fin tant que

Boucle do ... while

Faire
```

Tant que (condition) (on exécute les instructions de la boucle au moins une fois)

Boucle for

Instructions

for permet d'exécuter plusieurs fois la même instruction ou série d'instructions

Instruct est l'instruction (ou bloc) dont l'exécution est conditionnée par l'instruction **for**

- •Init permet de définir des conditions de début de boucle
- •Condition est une expression booléenne qui doit être vraie pour la poursuite de la boucle, c'est-à-dire une nouvelle exécution de instruct
- •**Itération** réalise un changement d'état de la ou des variables utilisées dans le contrôle de l'itération de boucle

```
while ( condition ) {
   instructions
}
```

```
do {
    instructions
    }while
(condition);
```

```
for (init,cond,iter) {
   instruct
}
```

Exemples

• 1) Portion de programme d'un compteur affichant les entiers de 0 à 9

Avec la boucle while :

```
int x = 0 ;
while ( x<10) {
    system.out.println(x);
Avec la boucle for
for (int x=0, x<10, x++) {</pre>
    system.out.println(x) ;
Utilisation du do ... while
int x;
do {
    x = 1 + (int) Math.random() * 6;
   } while (x != 6);
```