# Chapitre 1 - Introduction Java:

## Intro :

Java a été créée par Sun Microsystems en 1995. Il est encore très populaire

Sur Java il compile en bytecode, sur Android c’est différent. (plus efficace).

Il est un langage orienté objet

## Structure d’un programme Java

Un programme est composé d’une ou plusieurs classes  
Une classe contient une ou plusieurs méthodes  
Une méthode contient des instructions  
Une Application Java contient généralement une méthode appelée main  
Voir lincoln.java

### Commentaires

Les commentaires dans un programme sont appelés documentation interne, ils expliquent le but du programme et décrivent les étapes de traitement mais n’influencent pas l’exécution du programme.

Ils peuvent prendre 3 formes :

// commentaire qui s’étend sur le reste de la ligne

/\* commentaire qui peut s’étendre sur plusieurs lignes \*/

/\*\* commentaire pour javadoc

### Variables

Les identificateurs sont des mots utilisés par le programmeur, composés de lettres, de chiffres, tiret bas et signe de dollar. Ils ne peuvent commencer par un chiffre

Java fait la différence entre majuscules et minuscules

Par convention :

Le nom des classes débutent par une majuscule

Le nom d’une constante est tout en majuscule

### Mots réservés

|  |  |  |  |
| --- | --- | --- | --- |
| abstract  boolean  break  byte  case  catch  char  class  const  continue  default  do  double | else  extends  false  final  finally  float  for  goto  if  implements  import  instanceof  int | interface  long  native  new  null  package  private  protected  public  return  static  strictfp | super  switch  synchronized  this  throw  throws  transient  true  try  void  volatile  while |

### Séparateurs

Séparateur : espace, ligne vide, tabulation

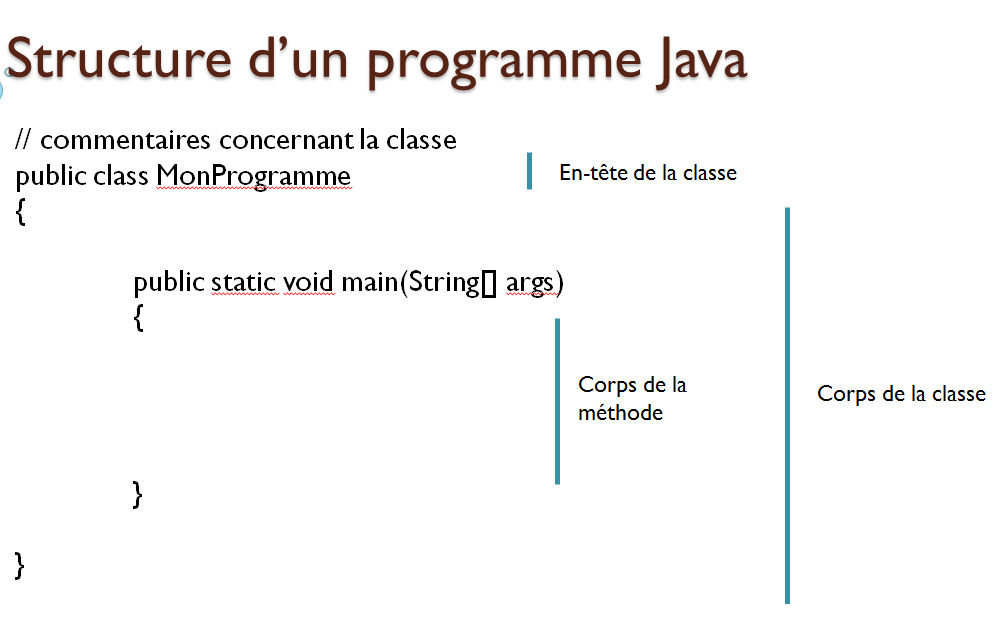
Un séparateur est utilisé pour séparer des mots et des symboles

Les séparateurs en trop sont ignorés

Un programme peut être formaté de différentes façons

Les programmes doivent être formatés de façon à augmenter la lisibilité en utilisant une indentation adéquate

Voir Lincoln2 et Lincoln3



## Niveaux des langages

Le compilateur Java traduit le code source en bytecode  
Le bytecode n’est pas un langage machine  
Un interpréteur traduit le bytecode en langage machine et l’exécute  
Donc, un compilateur Java n’est pas lié à une architecture CPU particulière  
On dit donc que Java est PORTABLE

## Environnement de développement

Il y a plusieurs environnements pour développer des logiciels Java :

Eclipse

jGRASP

Sun Java Development kit (JDK)

Sun Forte for Java

Monash BlueJ

Microsoft Visual J++

Netbeans

## Syntaxe et sémantique :

Les règles de syntaxe définissent comment nous pouvons combiner des symboles, des mots réservés et des identificateurs pour faire un programme valide

La sémantique définit la signification des instructions

Autrement dit, un programme peut être correct au niveau de la syntaxe sans l’être au niveau du sens (sémantique)

Un programme fait ce qu’on lui dit de faire et non ce qu’on pensait lui dire de faire

### On voit 3 types d’erreurs :

Erreurs de compilation : le compilateur les trouve et les signale. L’exécutable n’est pas généré

Erreur d’exécution

Erreur de logique

Erreur d’allocation (new to create, garbage collector to delete) Le garbage collector

Erreur de dépassement

# Chapitre 2 - Données et expressions :

## Chaines de caractères

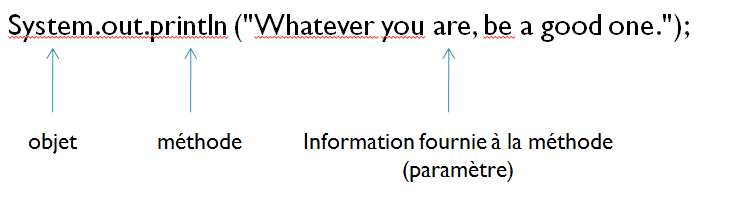
Chaque chaîne de caractères est un objet en Java, défini la classe String

Chaque chaîne littérale (entre guillements anglais) représente un objet String

## Utilisation d’objets

L’objet System.out représente une destination vers la quelle nous pouvons envoyer des sorties

Dans Lincoln, on invoque la méthode println pour l’objet System.out



## La méthode print

L’objet System.out offre un autre service

print ressemble à println mais elle ne fait pas de saut de ligne

Donc, tous les affichages avec print apparaissent sur la même ligne

Voir Countdown.java

## Chaîne de caractères

L’opération de concaténation (+) permet de coller une chaîne à une autre

Une chaîne ne peut être divisée entre 2 lignes dans un programme

Voir Facts.java

L’opérateur + est aussi utilisé pour l’addition

La fonction de cet opérateur dépend du contexte

L’opérateur + est évalué de gauche à droite

Cet ordre peut être modifié à l’aide de parenthèses

Voir Addition.java

## Séquences Échappe

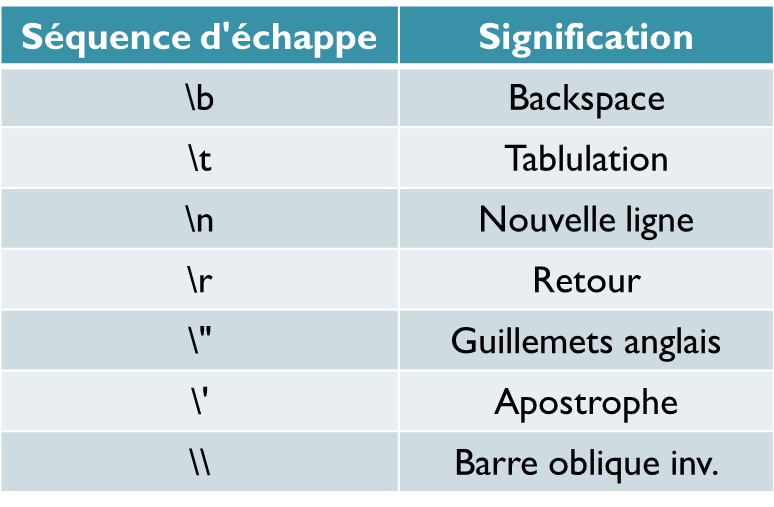
L’instruction ci-dessous est ambiguë pour le compilateur :

System.out.println("il a dit " bonjour" aux gens");

Une séquence Échappe est une série de caractères qui représente un caractère spécial

Une telle séquence commence par une barre oblique inversée indiquant que le caractère qui suit est spécial

System.out.println("Il a dit \"bonjour\" aux gens");



Voir Roses.java

## Variables

Une variable est un nom pour une zone en mémoire  
  
Une variable doit être déclarée en spécifiant :  
 son nom  
 le type de donnée qu'elle contiendra

Une variable peut recevoir une valeur initiale dans la déclaration  
 int sum = 0;  
 int min = 7, max = 149;

Quand une variable est référencée dans un programme, sa valeur courante est utilisée

Voir PianoKeys.java

## Constantes

Une constante est un identificateur qui ressemble à une variable sauf qu'elle garde UNE valeur tant que le programme est actif

Si on tente de modifier sa valeur, il y aura une erreur de compilation

En Java, on utilise final pour déclarer une constante

Constantes :

Nomment des valeurs littérales

Facilitent la modification de valeurs utilisées plusieurs fois dans le programme

Permettent d'éviter des modifications accidentelles de valeurs

## Types primitifs

Il y a exactement 8 types primitifs en Java

Quatre représentent les entiers :

byte, short, int, long

Deux représentent les réels :

float, double

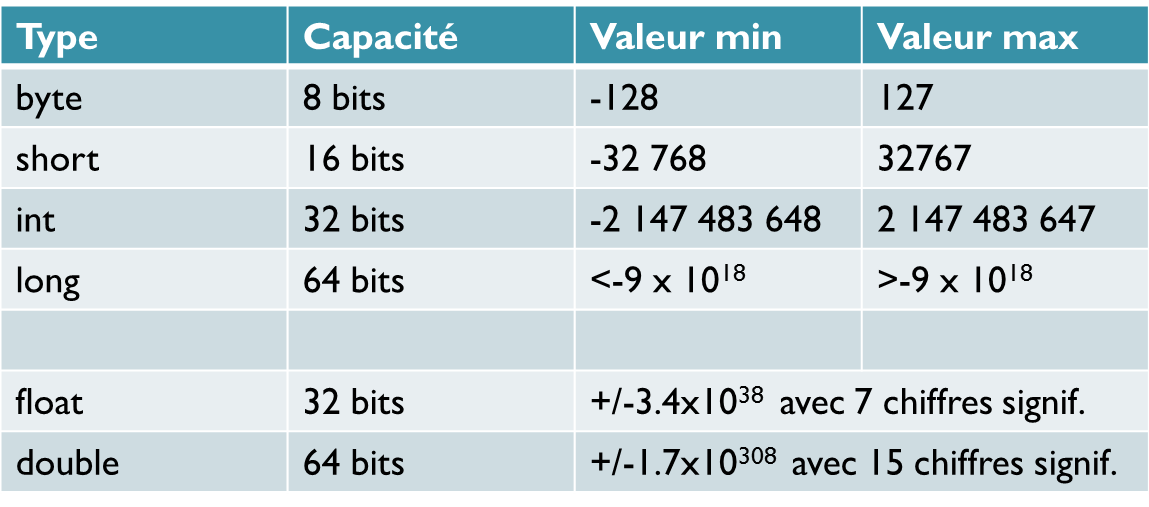
Un représente les caractères :

char

Un représente les booléens :

Boolean

Ce qui différencie les types primitifs numériques est leur taille :



## Caractères

Une variable char contient un caractère de l'ensemble Unicode

Chaque caractère correspond à un nombre unique

L'ensemble Unicode utilise 16 bits, donc il contient 65 536 caractères différents

C'est un ensemble international

Un caractère littéral est mis entre apostrophes

L'ensemble de caractères ASCII est plus vieux et plus petit qu'Unicode, cependant, il est encore populaire.

Les caractères ASCII sont un sous-ensemble de l'ensemble Unicode, incluant :

lettres majuscules et minuscules

ponctuation

chiffres

symboles spéciaux

caractères de contrôle (retour, tabulation…)

## Booléens

Une valeur booléenne est : vrai ou fausse

Exemple : ampoule allumée ou éteinte

Exemple : boolean fini = false;

## Expressions arithmétiques

Une expression est une combinaison d'un ou plusieurs opérandes et leurs opérateurs

Une expression arithmétique fait un calcul numérique en utilisant : +, -, \*, /, %

Si l'un des opérandes est réel, le résultat sera réel

Les opérateurs peuvent être combinés dans des expressions complexes

resultat = total + compteur / max - offset

Les opérateurs ont un ordre de priorité précis

\*, /, % sont plus prioritaires que + et –

Les opérateurs ayant la même priorité sont évalués de gauche à droite

Les parenthèses permettent de changer les priorités

L'opérateur d'affectation a une priorité plus basse que les opérateurs arithmétiques  
 resultat = somme / 4 + MAX \* petit;

Les côtés gauche et droit de l'affectation peuvent contenir la même variable  
 compteur = compteur + 1;

Ces opérateurs sont arithmétiques et unaires

++ incrémente de 1

-- décrémente de 1

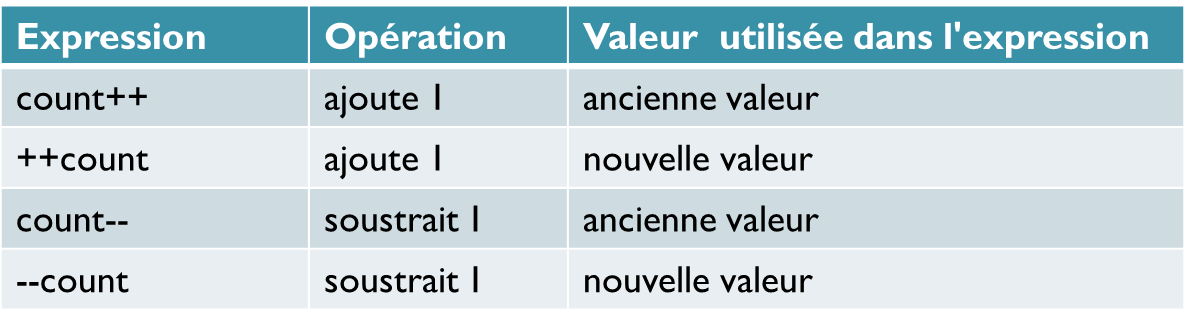
compteur++; est équivalent à :

compteur = compteur + 1;

Il y a 2 formes pour ++ et –

Postfixe : compteur++; ou compteur--;

Préfixe : ++compteur; ou --compteur;

Quand les opérateurs ++ et -- sont utilisés dans une expression plus grande, les 2 formes ont des effets différents 

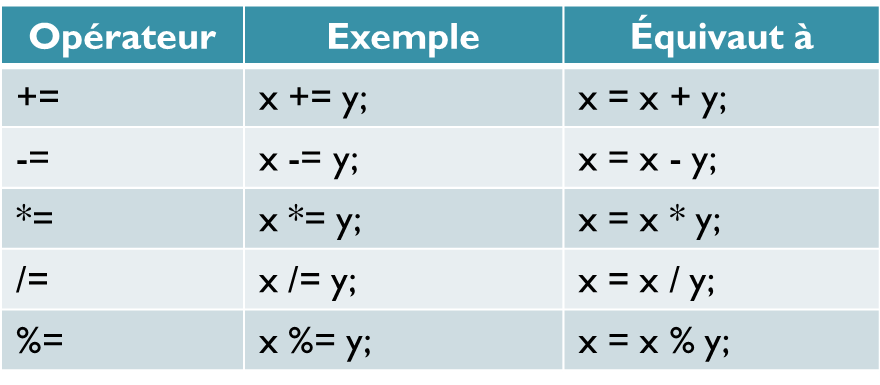
Souvent nous effectuons une opération sur une variable et nous stockons ensuite le résultat dans la variable

Java offre des opérateurs d'assignation qui simplifient ce processus

Par exemple, les instructions suivantes donnent le même résultat   
 nombre += compteur;

nombre = nombre + compteur;

Il y a plusieurs opérateurs d'assignation.



Le côté droit de l'opérateur est entièrement évalué d'abord, ensuite le résultat est combiné avec la valeur initiale de la variable

Le comportement de ces opérateurs dépend du type des opérandes

Si les opérandes de += sont des chaînes de caractères, ce sera une concaténation

## Conversions

Parfois, il est pratique de convertir les données d'un type à un autre type  
 Exemple : entier vers réel

Les conversions peuvent faire perdre de l'information

Les conversions de type "promotion numérique" sont sécuritaires

Les conversions "dégradantes" peuvent faire perdre de l'information (ex : int vers short)

En Java, les conversions peuvent se faire de 3 façons :

affectation

promotion arithmétique

cast

Conversion par affectation pour promotion numérique seulement

La promotion arithmétique se fait automatiquement quand les opérateurs convertissent leurs opérandes

Le cast est puissant mais dangereux

Exemple : total et compteur étant des nombres entiers, on peut avoir une division réelle

resultat = (float) total/compteur

## Programmes interactifs

Généralement, les programmes ont besoin de lire des données

La classe Scanner fournit des méthodes pratiques pour lire des valeurs de différents types

Un objet Scanner peut lire des données à partir de différentes sources, incluant le clavier

Les entrées à partir du clavier sont représentées par l'objet System.in

La ligne suivante crée un objet Scanner qui lit à partir du clavier :

Scanner scan = new Scanner (System.in);

L'opérateur new crée l'objet Scanner

Une fois créé, l'objet Scanner peut être utilisé pour appeler différentes méthodes de lecture :

answer = scan.nextLine();

La classe Scanner fait partie de la librairie de classes java.util, et doit être importée dans un programme pour être utilisée

la méthode nextLine lit toute une ligne

Les détails concernant la création des objets sera présenté au chapitre 3

Voir Echo.java

Les séparateurs sont utilisés pour séparer le éléments (Tokens) lors de la lecture

Les séparateurs incluent les espaces, la tabulation et le saut de ligne

La méthode next de la classe Scanner lit le token suivant et le retourne dans une chaîne de caractères

Des méthodes comme nextint et nextDouble lisent des données d'un type particulier

Voir GasMileage.java

# Questionnaires :

## Quiz 1 :

