Variable declaration

```
type name = value ;
```

Examples:

```
String message = "Hello!";
int counter = 0;
String answer = System.console().readLine();
boolean isPositive = counter > 0;
double half = x / 2;
```

Important types

`String`	Text. Ex: `"abc"`
`int`	Integer number. Ex: `10`
`double`	Decimal number. Ex: `3.14`
`boolean`	`true` or `false`

Method signature

```
static returnType name ( parameter1Type paramter2Type , ...) { ... }
```

Examples:

```
static void main(String[] args) { ... }
static void sayHello() { ... }
static boolean askYesOrNo(String question) { ... }
static double exponent(double number, double exponent) { ... }
```

Useful Java methods

```
void System.out.println(String message)
String System.console().readLine()
int Integer.parseInt(String numberAsText)
double Math.random()
double Math.round(double number)
double Math.ceil(double number)
double Math.cos(double angleRadians)
double Math.sin(double angleRadians)
```

Important operators

With numbers, returning a number:

```
3 + 4
3 - 4
3 * 4
3 / 4
```

With texts, returning a boolean:

```
"abc".equals("abc")
```

With texts, returning a text:

```
"abc" + "def"
```

With numbers, returning a boolean:

```
3 == 4
3 != 4
3 > 4
3 >= 4
3 < 4
3 <= 4
```

With booleans, returning a boolean:

```
true && false
true || false
```

Conditions

```
if ( booleanValue ) {
  …
} else if ( booleanValue ) {
  …
} else {
  …
}
```

Note: **`else if`** and **`else`** are not mandatory

Example:

```
if (number == 0) {
  return "zero";
} else if (number > 0) {
  return "positive";
} else {
  return "négative";
}
```

Loops

```
while ( booleanValue ) {
  …
}
```

Example:

```
int counter = 1;
while (counter <= 10) {
  System.out.println(counter);
  counter = counter + 1;
}
```

Working with arrays

Create an array with values:
```
String[] fruits = {"Pomme", "Orange", "Banane"};
```
Create an empty array:
```
int[] numbers = new int[10]; // Taille de 10 éléments
```
Get an element:
```
String pomme = fruits[0];    // indice 0 = 1er élément
String banane = fruits[2];   // indice 2 = 3ème élément
```
Replace an element:
```
fruits[1] = "Citron";
```
Get the size of the array:
```
int taille = fruits.length;  // taille = 3
```
Looping through elements:
```
for(String fruit : fruits) {
  System.out.println(fruit);
}
```

Classes

```
public class ClassName {
  private fieldType fieldName = value;
  // Other fields

  public ClassName(paramType paramName, …) {
    // Constructor code
  }
}
```

Instantiation (calling the constructor):

```
ClassName variableName =
  new ClassName(param1, …);
```

Reference to a class field:

```
this.fieldName
```

A well built class

- All fields have a value after the constructor execution
- All fields are **`private`**
- All fields which stays constant are **`final`**
- All methods not used outside the class are **`private`**

Packages

```
package name1.name2;

public class ClassName { … }
```

`ClassName.java` go into the `src/main/java/name1/name2` folder

Usage (at the beginning of the file):

```
import name1.name2.ClassName;
```

Java code conventions

Class names: `UppercaseUppercaseUppercase`. No - or _.

Variable, parameter, method, package names: `lowercaseUppercaseUppercase`. No - or _.

Constant names: `UPPERCASE_UPPERCASE_UPPERCASE`. No -.

`if` , `else`, `else if`, `while`, `for`: always use { and }.

Curly brackets: '{' at the end of line, and '}' at the same level as the line which opens the '{'. Example:

```
if (x > 0) {
  if (y > 0) {
    // Code
  }
}
```

Indentation: 4 spaces more at each new level. Example:

```
public class Talkative {
□□□□public void sayHello() {
□□□□□□□□System.out.println("Hello");
□□□□}

□□□□public void sayBye() {
□□□□□□□□System.out.println("Bye");
□□□□}
}
```