Java Basics

Java Comments

• Comments can be used to explain Java code, and to make it more readable. It can also be used to prevent execution when testing alternative code.

Single-line Comments

• Single-line comments start with two forward slashes (//).
• Any text between // and the end of the line is ignored by Java (will not be executed).
• This example uses a single-line comment before a line of code:

```
// This is a comment
System.out.println("Hello World");
```

• This example uses a single-line comment at the end of a line of code:

```
System.out.println("Hello World"); // This is a comment
```

Java Multi-line Comments

• Multi-line comments start with /* and ends with */.
• Any text between /* and */ will be ignored by Java.
• This example uses a multi-line comment (a comment block) to explain the code:

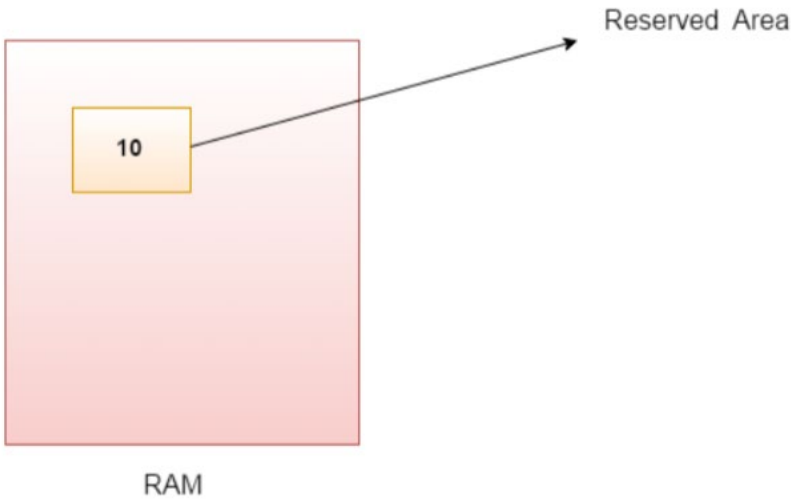
```
/* The code below will print the words Hello World
to the screen, and it is amazing */
System.out.println("Hello World");
```

Single or multi-line comments?

• It is up to you which you want to use. Normally, we use // for short comments, and /* */ for longer.

Java Variables

Variables are containers for storing data values. **Variable** is name of *reserved area allocated in memory*. In other words, it is a *name of memory location*. It is a combination of "vary + able" that means its value can be changed.



Variable in RAM space

In Java, there are different **types** of variables, for example:

- String - stores text, such as "Hello". String values are surrounded by double quotes
- int - stores integers (whole numbers), without decimals, such as 123 or -123
- float - stores floating point numbers, with decimals, such as 19.99 or -19.99
- char - stores single characters, such as 'a' or 'B'. Char values are surrounded by single quotes
- boolean - stores values with two states: true or false

Declaring (Creating) Variables

- To create a variable, you must specify the type and assign it a value:

Syntax

type variable = value;

- Where *type* is one of Java's types (such as int or String), and *variable* is the name of the variable (such as **x** or **name**). The **equal sign** is used to assign values to the variable.
- To create a variable that should store text, look at the following example:
- Create a variable called **name** of type String and assign it the value "**John**":

```
String name = "John";
System.out.println(name);
```

• To create a variable that should store a number, look at the following example:Create a variable called **myNum** of type int and assign it the value **15**:

```
int myNum = 15;
System.out.println(myNum);
```

• You can also declare a variable without assigning the value, and assign the value later:

```
int myNum;
myNum = 15;
System.out.println(myNum);
```

Note that if you assign a new value to an existing variable, it will overwrite the previous value:Change the value of myNum from 15 to 20:

```
int myNum = 15;
myNum = 20; // myNum is now 20
System.out.println(myNum);
```

Final Variables

• However, you can add the final keyword if you don't want others (or yourself) to overwrite existing values (this will declare the variable as "final" or "constant", which means unchangeable and read-only):

```
final int myNum = 15;
myNum = 20;
// will generate an error: cannot assign a
value to a final variable
```

Displaying Variables

• The println() method is often used to display variables.
• To combine both text and a variable, use the + character:

```
String name = "John";
System.out.println("Hello " + name);
```

• You can also use the + character to add a variable to another variable:

```
String firstName = "John ";
String lastName = "Doe";
String fullName = firstName + lastName;
System.out.println(fullName);
```

• For numeric values, the + character works as a mathematical operator (notice that we use int (integer) variables here):

```
int x = 5;
int y = 6;
System.out.println(x + y); // Print the value of x + y
```

From the example above, you can expect:
• x stores the value 5
• y stores the value 6
• Then we use the println() method to display the value of x + y, which is **11**

Declaring Many Variables

• To declare more than one variable of the **same type**, use a comma-separated list:

```
int x = 5, y = 6, z = 50;
System.out.println(x + y + z);
```

Java Identifiers

• All Java **variables** must be **identified** with **unique names**.
• These unique names are called **identifiers**.
• Identifiers can be short names (like x and y) or more descriptive names (age, sum, totalVolume).
• **Note:** It is recommended to use descriptive names in order to create understandable and maintainable code:

```
// Good
int minutesPerHour = 60;
// OK, but not so easy to understand what m actually is
int m = 60;
```

The general rules for constructing names for variables (unique identifiers) are:
• Names can contain letters, digits, underscores, and dollar signs
• Names must begin with a letter
• Names should start with a lowercase letter and it cannot contain whitespace
• Names can also begin with $ and _
• Names are case sensitive ("myName" and "myname" are different variables)
• Reserved words (like Java keywords, such as int or boolean) cannot be used as names