Class 1: Java Basics

Agenda

- Java Session FAQ & Key Points
- What is Program / Programming language / Java?
- Java program structure & execution flow
- Java main method
- First Java Program
- Identifier
- Java Keywords

Java Session FAQ and Key Points

- Can I ask questions during the class?
- Do I need to be on video?
- When we'll have a break?
- I got very silly question, Can I ask?

- Give sometime to get familiar with the concept
- Don't judge yourself too quickly
- Practice EVERY DAY!!!!

What is Program / Programming language / Java?

IDE for Java

A **Java IDE** (for Integrated Development Environment) is a software application which enables users to more easily write and debug **Java** programs. Many **IDEs** provide features like syntax highlighting and code completion, which help the user to code more easily.

Examples of IDE's:

- Eclipse
- Java NetBeans
- Visual Studio
- IntelliJ

Java Main Method

Java ***main*** method.

Starting point of executing any Java code is the ***main*** method. Without main method we won't be able to run our code and get output.

Syntax:

public static void main (String[] args) {

    Your code

}

Identifiers

Java Identifiers are used to identify a class name, method or variable name and also a label.

Rules:

- identifiers must be composed of letter, numbers, the underscore _ and the dollar sigh $.
- identifiers may only begin with a letter, the underscore _ and the dollar sign $
- Variable names are case-sensitive.
- Its should not contains keyword that is reserved by java

Identifiers

Examples of valid identifier:

- variable
- _variable
- $variable
- _7variable

Examples of invalid identifiers:

- My Variable - (it contains a space)
- 123gkk - (it begins with numbers)
- a+c - (plus sign is not an alphanumeric character)
- variable-2 - (the hyphen is not allowed)
- sum_&_difference - (ampersand is not an alphanumeric character)
- O’Reilly - (the apostrophe is not an alphanumeric character)

Java keywords

Java has a set of keywords that are reserved words that cannot be used as variables, methods, classes, or any other identifiers

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

Data types & variables in Java

In Java, data types are associated with variables.

In Java, **a variable is a named reference to a memory area** where value of the variable is stored. I n other words variable is like a container where we store value.

A variable declaration has following syntax:

```
[data_type] [variable_name] = [variable_value];
```

Data types

Java has 2 type of Data Types:

Primitive:

- byte
- short
- int
- long
- float
- double
- char
- boolean

Non Primitive:

- String

Numeric data types range

Type	Size	Range
byte	8 bits	-128 .. 127
short	16 bits	-32,768 .. 32,767
int	32 bits	-2,147,483,648 .. 2,147,483,647
long	64 bits	-9,223,372,036,854,775,808 .. 9,223,372,036,854,775,807
float	32 bits	$3.40282347 \times 10^{38}$, $1.40239846 \times 10^{-45}$
double	64 bits	$1.7976931348623157 \times 10^{308}$, $4.9406564584124654 \times 10^{-324}$

Variable declaration

Variables have 3 properties :

- data type
- name
- value

int i; - declaring variable of integer data type

int i = 10; - declaring variable and assigning the value

int a, b, c; - declaring multiple variables of the same type (integer)

```
Example of variable declarations

int i = 10;          //Variable of int type
```

Operators in Java

Java divides the operators into the following groups:

- Assignment operators
- Arithmetic operators
- Comparison / Relational operators
- Logical operators

Assignment operator =

We use the **assignment** operator (=) to assign the value to a variable:

```
int x = 10;
```

Arithmetic operators in Java

Operator	Name	Description	Example
+	Addition	Adds together two values	x + y
-	Subtraction	Subtracts one value from another	x - y
*	Multiplication	Multiplies two values	x * y
/	Division	Divides one value from another	x / y
%	Modulus	Returns the division remainder	x % y

Modulus operator

- The **remainder** or **modulus operator** in **Java**.
- The % **operator** returns the **remainder** of two numbers. For instance 10 % 3 is 1 because 10 divided by 3 leaves a **remainder** of 1.

```
int q=23;
int v=11;

System.out.println(q%v);//the remainder is 1 cos in 23 (11+11)+1

int e=-5+4*6;
System.out.println(e);//19

int x=(22+9)%7;
System.out.println(x);//3

int z= 5+15/3*2 - 8%3; //(5+10-2)
System.out.println(z); //13
```

Expression

- Expression is a combination of operand and operator that evaluates to a single value.
- The order of evaluation of operators in an expression is determined by the precedence and associativity of the operators
- Examples:
  - 12 + 4 - 9
  - 12 * 7 / 3

Priority and Associativity of operators

Operator	Precedence	Associativity
* / %	1	Left to Right
+ -	2	Left to Right
< <= > >=	3	Left to Right
== !=	4	Left to Right
&&	5	Left to Right
||	6	Left to Right
=	7	Right to Left

String

- **Java String** represents a sequence of characters and cannot be changed once created.
- **To create Strings** in Java we simply assign the characters in double quotes – this way is called String literals.

```
String literal example

String blogName = "howtodoinjava.com";

String welcomeMessage = "Hello World !!";
```

String concatenation

- The + operator can be used between strings to combine them. This is called **concatenation**: