Extremely easy Java Tutorial

From Java Learning Academy

©http://www.javalearningacademy.com
Contents

Introduction to the course ............................................................................................................................ 3
Chapter 1: What is a programming language ............................................................................................... 4
Chapter 2: Run your first program ................................................................................................................ 6
  Java Online Compiler ................................................................................................................................ 6
  Understand your first program ................................................................................................................. 7
  Writing a function ..................................................................................................................................... 9
  System out .............................................................................................................................................. 11
Chapter 3: Add, subtract, divide, modulus ................................................................................................. 12
Chapter 4: Data Types ................................................................................................................................. 13
Chapter 5: Array .......................................................................................................................................... 15
Chapter 6: Logical constructs and operators ............................................................................................. 17
  If – Else .................................................................................................................................................... 17
  SWITCH – CASE ........................................................................................................................................ 17
  TERNARY OPERATOR ............................................................................................................................... 19
Chapter 7: Loops ......................................................................................................................................... 20
  For ........................................................................................................................................................... 20
  For Each ( we will cover this later) .......................................................................................................... 20
  While ....................................................................................................................................................... 20
  Do While.................................................................................................................................................. 21
Chapter 8: How to use Class and Objects ................................................................................................... 23
Conclusion................................................................................................................................................... 25
Introduction to the course

Congratulations! You are reading the extremely easy and simplest Java tutorial.

If you are a beginner, this course will boost your confidence and give you a jump start in the Java learning. This course is designed as baby steps. I will show you some concepts then ask you to try out certain things. Follow through every instruction to get the best out of it.

This tutorial is also meant for anyone who is new to Java or even programming. No previous experience is assumed. However, if you have a general idea about any programming language it will greatly help.

Some of the discussion here is to convey the concepts and to make it understand in a very easy way. You should keep in mind that there could be much depth in those concepts. Let me take an example, when I mention ‘a class is group of similar functions’; that is just for you to understand – actually class is much more than that.

This course is free. You are always welcome to share it with others. However, if you are using any part of this course elsewhere, please don’t forget to give credit to Java Learning Academy. If you are using it in a website, a link that follows to JLA web site is required.

Arranged as simple chapters, you will find this course easy to complete. Once you complete this course, you will understand the basic concepts in Java programming. You will be able to write your own Java programs and run it. This course also will prepare you for advanced topics in Java. So let’s get ready!
Chapter 1: What is a programming language

Simply put, programming language is a way to give instructions to a computer. Instructions to the computer have to go in a very structured way. Computer can only process one step each time. There are several programming languages and Java is one of them.

When you feed instructions to a computer in a programming language, computers will turn that in to something called ‘machine language’ which is hard for humans to read through, but could be understood by the machine.

The process of turning in to machine language is called compilation. And the tool used for compilation is called a compiler. If you did not understand what all this is about, don’t worry. You will understand it eventually. And it is not absolutely needed to know for this tutorial, but it is good to know.

What is Java

Java is a programming language. The difference of Java from other language is that, the machine language it creates is universal. For example a C++ program, once turned in to machine language in an Apple computer, it could only be understood by Apple Computer and not by an Intel Computer. In case of Java both Apple and Intel can understand it. Converting a programming language to the machine language is called compiling.

How is Java compiling universal?

Here is where the function of Java Virtual Machine (JVM) comes. JVM is very similar to an operating system. Java has JVM for each of the machines – apple/windows etc. Java code is compiled in to a form called byte codes. JVM can understand byte codes. So JVM is different for various platforms but the byte code itself is same.

Java programs are compiled in to bytecodes which could be run in any platform supported by JVM.

Helpful terminologies

You don’t have to know this for covering this tutorial. So you are free to skip it if needed.

JVM – Java virtual machine

JRE – Java run time environment. When you want to run your java program you need this. JVM is included here.

JDK – Java Development Kit- tools needed for development and compiling etc.


JEE – Java Enterprise Edition
More about this is covered in my post http://www.javalearningacademy.com/jre-jdk-and-jee-explained-in-detail/
Chapter 2: Run your first program

Now that we know in a roughly what Java is, let us run your first program. Learning too much theory is boring and being hands on will keep you excited and help you learn quickly.

Taking one at a time:

I will take you through this tutorial in step by step. This is designed for the one who is totally new to Java or even programming. So, let not the details scare you. Once you go through this tutorial, things should be very clear.

Objective Here:

Just run a program and see its output

Java Online Compiler

Normally, you need to install JDK to compile and run Java programs. To make it quick, I’m suggesting using an online compiler. For this tutorial this is good enough for you.

Now, follow the below steps:

2. Select the online compiler and get used to its UI

Copy and paste the below code in to the window of online compiler.

```java
//---- start copying from here----
public class HelloWorld {
    public static void main( String[] args ) {
        System.out.println("Hello World!");
    }
}
//--- End copying here
```

Comment using //

Note: you might have notices the line

//---- start copying from here----

This has no computational value and used for coming the code. To make a line a comment it has to start with two slashes //

3. Now click run the program.
Now you should see that "Hello World!" is printed in the output window. Congratulations!!! You successfully ran your first Java program!

Don’t worry how it all happens. But you should play around with this program. So do the following Exercises:

**Exercise 1:** Change “Hello World!” to “My name is (Your Name)” and run the program

**Exercise 2:** Play around with the message

**Understand your first program**

Note: I will be suggesting you to try out certain codes as we go through understanding the first program. You can open a new window of the online compiler so that we have our first program always handy.

Line 1: `public class HelloWorld {

public: tells that the class HelloWorld is available to all. There are ways need to make certain things hidden from others. We will discuss this later.

class: It tells that we are defining a class. Name of the class will come right after this keyword.

HelloWorld: This is the name of the class.

Conclusion: We are telling computer, define a class with the name HelloWorld that is accessible to all.

Now:

What is a class?

Before knowing what is a class; let me introduce you functions, it is also called methods.

Function is simply a group of instructions. It can take an input and it can produce an output.

Assume you are creating a function for shopping. Let the name of this function be goForShopping. We can write its instruction as below
Function goForShopping:
Step1: look in to shopping list
Step2: based on the shopping list decide which shop to go
Step3: Arrange a transport to reach the shop
Step4: Buy items from the shop
Step5: Make payment
Step 6: Come back to home
Stop Function goForShopping

Exercises for you:

1. What is the name of the function we just created?
2. Where does it start and where does it end
3. What are the instructions given in the function
4. Do you think all these steps have to be performed in the order?

Now, Let us think how this function could be used for shopping ‘ANY THING’. Simple – you have passed a shopping list as an input to this function. Function should use this shopping list then make decisions based on this. Input to the function is called an argument or parameter. Shopping list is a parameter that will be given to this function.

So it the function could be rewritten as below

Function goForShopping (input: shoppingList):
Step1: look in to input: shoppingList
Step2: based on the shopping list decide which shop to go
Step3: Arrange a transport to reach the shop
Step4: Buy items from the shop
Step5: Make payment
Step 6: Come back to home
Stop Function goForShopping

Now, whenever, you want to for shopping, you don’t need to worry about the steps. You have a function that will take care of these steps. You just need to call that function and pass a shopping list, things will be all done.

So we learned what a function – a group of instructions is. Going back to our question what is a class? Class is a group of related functions.

Here HelloWorld is a class in which we will write some functions for us to test it out. Did you notice a curly bracket ({ }) in the line number 1. In Java that is a way to start a block of code. And there will be a corresponding opposite bracket (}) to end the same block of code. There can be sub blocks under a main block. All will be separated by the same mechanism.

Let us move to line 2 of the code.
Line 2: `public static void main(String[] args) {}`

Here we are defining a function.

`public`: This function is available to all

`static`: More advanced topic, will cover later.

`void`: This is where you mention output of the function. This function do not produce any output so, we give void. Void simply means ‘nothing’.

`Main`: This is the name of the function. Main is a special function. Compiler uses this function as the starting point. Anything outside of the main function will not be executed, UNLESS, called within the main function. This will be more clear as we progress.

`(String[] args)`: args is the name of the parameter (like shopping list in your function example). Args stands for arguments. String[] is given to denote its type. You will know this when we cover data types.

**Line3: `System.out.println("Hello World!");`**

This instruction tells the computer to print the message “Hello World!”

In fact, there are many things should happen to display this message in to your screen. So println is really a function that displays ‘Hello World!”

So here is what happened.

We created a class called HelloWorld

We defined a function called `main(String[] args)`

From that function we invoked another function called `System.out.println` that displayed the message “Hello World!”

We then executed this code, the compiler looked for `main()` function and executed it

All good!

**Writing a function**

Now let us learn how to write a function and invoke.

Copy and paste the below code to our online compiler (Feel free to open a new window of the online compiler, if required):

```java
public class MethodsSample {
    public static void display(String message) {
        System.out.println(message);
    }
}
```
public static void cook() {
    System.out.println("I'm cook");
}

public static void add(int a, int b) {
    int sum = a+b;
    System.out.println("Sum is "+sum);
}

public static void main(String args[]) {
    add(100,99);
}

And run this program. What do you see as output?

You should see: Sum is 199

Let us analyze:

Go to the starting point, which is the main() function. If you look at the whole code you will see other
functions also. display(), cook(), add(), main() all of these are functions.

main() is special, because machine is going to start the execution from there. When it executes, it calls
the function add(). add() can take parameters. We sent 100 and 99 as the parameters. add() does an
addition operation using + sign and prints the result.

There is a twist to + sign. Did you notice the line:

System.out.println("Sum is "+sum);

There is a + sign here too. There is nothing to ‘add’ here. Here is it simply ‘welding together’ two Strings
“Sum is “ and the value contained in sum.

Exercises:

1. Pass 30 and 99 as the parameters to add function and execute.
2. Pass -50 and 60 as the parameters to add function and execute
3. Instead of add(), call cook() function from main function
4. Call both add() and then cook() function from main function.
5. Change the order and call cook() then add() from main function
6. Call display() function from main function. You need to pass a parameter here. It must be in
double quotes. Example: “I'm display”
If you are of the curious nature, you might be wondering `System.out.println()` – how does it work? `println()` is a function that is written inside the class `System.out`. We don’t really need to know what the instructions in `println` are. It is the whole purpose of having a function itself – others don’t have to know the details, but just call and pass the parameters.
Chapter 3: Add, subtract, divide, modulus

So far we learned

1. How to write a class
2. How to write a function
3. How to pass parameters to a function
4. How to call a function

We also added two numbers using + sign. Here the + sign is called operator. To make it more clear addition is done using + operator. There are other arithmetic operators such as:

<table>
<thead>
<tr>
<th>Operator</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>+</td>
<td>For addition</td>
</tr>
<tr>
<td>-</td>
<td>For Subtraction</td>
</tr>
<tr>
<td>*</td>
<td>For multiplication</td>
</tr>
<tr>
<td>/</td>
<td>For division</td>
</tr>
<tr>
<td>%</td>
<td>Modulus: if we do a%b, it will divide a with b then will return the reminder. Repeat, it will return the reminder. Ie, 10%3 will return 1 and 10%2 will return zero.</td>
</tr>
</tbody>
</table>

Here is a code snippet

```java
public class BasicOperators {
    public static void main(String args[]) {
        int a = 100;
        int b = 200;

        int c = a+b; //addition:+, subtraction: -, division: /, modulus: %

        System.out.println("Result=":"+c);
    }

    //exercise: apply different operators in the above example.
}
```

Exercise:

1. Write functions and use all the arithmetic operators. Play around with it.
Chapter 4: Data Types

Do you still recall the + operator? Good. When we used it among two numbers, it did the addition. When we used it between a String and a number, it welded together (or concatenated) it in to a bigger string. That is why we got the result: The Sum is 99, when we added “The Sum is” + sum.

Let us analyze the following line

```java
int sum = a+b;
```

Here is the explanation:

int : It tells that we are going to define a variable of the type integer. We simply write int for that. We call this data type. That is we are going to define something that is of data type integer.

sum: This is a variable of the type int.

= : This is assignment operator. Sum of variable a and b is calculated and assigned to the variable sum.

Here is what happened:

We defined a variable with name ‘sum’ and this is of the date type int.

We added two other variables both are of the int type and assigned to the variable ‘sum’.

Now it is the time to explore other Data types. Just like int is a data types following are the other data types.

Java has primitive data types and Class based data types. Difference? Primitive data types cannot have and function. Only operators can work on it. Class based data types ‘wraps’ a primitive data type in to a class and can have its own functions. For example, corresponding to int data type, there is a Integer class available in Java. We will discuss about class based data types later. Now let us focus on primitive data types.

<table>
<thead>
<tr>
<th>Data Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>int</td>
<td>Integer. Can hold value from minimum of -2,147,483,648 to Maximum value of 2,147,483,647. This is a 32 bit (4 bytes) data type.</td>
</tr>
<tr>
<td>byte</td>
<td>Similar to int but is of the size of a byte (8 bits). So it can only hold value from -128 to +127</td>
</tr>
<tr>
<td>Short</td>
<td>Similar to int but of the size of two bytes. Value can range from -32,768 to +32,767</td>
</tr>
<tr>
<td>long</td>
<td>Similar to int, but of bigger size. This is of 8 bytes</td>
</tr>
<tr>
<td>float</td>
<td>This is used for value with decimal point and is of the size 32 bits</td>
</tr>
<tr>
<td>double</td>
<td>Similar to float but bigger size. This is of the size 64 bits</td>
</tr>
<tr>
<td>boolean</td>
<td>This can hold only two values: true and false. I have covered Boolean data type in this post</td>
</tr>
</tbody>
</table>
Exercise:

2. Define variable of primitive data types given in the above chart.
3. Apply Arithmetic operators on variables of various data types and print the output. What the difference.
Chapter 5: Array

At this point you should know what a data type is. An integer, boolean etc. are examples of data types. Age of a person is an example of integer data type. And you can assign an integer to it – let it be 15, 20, 5, 34 anything.

Now think of a situation, you want to store age of every one in your family. Is integer good enough? No. integer can have only one value. So you need many integers that could be kept somewhere. This is the purpose of array. Array has an order, which means you can say first position of the array is you age, second position is your brother’s age etc. In java we can create an integer array as below:

```java
int[] ages;
```

you see the difference between declaring an int and declaring and int array? Two square brackets do the trick.

Now here is how you assign value to the array

```java
int[] ages = {24,16,39,46,5};
```

How to access array?

You can access the array with its index number. In java index starts with zero.

Thus, ages[0] value is 24a, ages[3] value is 46 in the above example.

Length of an array

Many times you need to programmatically identify length of an array. You can simply say ages.length to get its length. Here in this example, length will be 5.

Exercise:

1. Declare an array of type int with 10 elements in it.
2. Print out its length-1 index value

We have only mentioned primitive type array here. However, you can have array of any type. You can keep Objects also in the array. Even you can keep array inside an array – this is called multi-dimensional array!

Multi-dimensional Array:

If you want an array of integer arrays, or another way of saying this is, if you want a two dimensional array, here is how you declare it:

```java
Int[][] manyAges = { {1,4,6,2}, {34,45,21,1}, {45,2}};
```

Now if you say manyAges[1] it will get the array of index =1, or second position.

If you say manyAges[1][0] it will get array of index1 and value of index=0 from that array.

Exercise:
1. In the above two dimensional array example print out value of manyAges[0]
2. Play around with this array and print of different locations
Chapter 6 : Logical constructs and operators

So far we have been doing some basic operations and displaying the result. Time get to the next level.

Let us do this: we want to add two numbers. If it is greater than 100, you want to display “I’m big!”. Otherwise you want to display “Not that big”. We can do this using logical constructs.

Here is the code for the same:

```java
public class IfElse {

    public static void main(String args[]) {
        int variable = 100;
        if (variable > 100) {
            System.out.println("I'm big!");
        } else {
            System.out.println("Not that big");
        }
    }
//Exercise : Play around with the above number and output
}
```

**If – Else**

Look at the above code. If there is condition you want to do something. Else you want to do something else. By this time the above code should be understandable by you.

**Exercise:**

Play around with different numbers in the above code.

**SWITCH – CASE**

We can expand out If-Else code in to the following. Instead of just two checks we added one more check. This way we can have any number of checks. The code below:

```java
public class IfElse {

    public static void main(String args[]) {
        int variable = 100;
        if (variable > 100) {
            System.out.println("I'm big!");
        } else if (variable > 50) {
            System.out.println("Slightly Big");
        } else {
            System.out.println("Not that big");
        }
    }
```
Instead of keeping writing the ‘if’ condition, we can use a switch statement. Code is below:

```java
public class Switch {
    public static void main(String args[]) {
        int variable = 100;

        switch (variable) {
            case 100: {
                System.out.println("I'm big!");
                break;
            }
            case 50 : {
                System.out.println("Slightly Big");
                break;
            }
            default : {
                System.out.println("Not that big");
                break;
            }
        }
    }
}
```

We just introduced a new technique called ‘break’.

It is used to break the computation and come out of a block of code. In this case, break will help us come out of the switch block immediately break is called. To demonstrate this try out the following exercises:

**Exercises**

1. Play around with the above number and output
2. Remove the break and see the difference
TERNARY OPERATOR

In the logical constructs we learned about if and else. There is short cut to do the same, using ternary operator.

Instead of following code

```java
Int i=0;
if( i>100 ) {
  result = "I'm big";
} else {
  result = "Not so Big";
}
System.out.println(result);
```

You can use

```java
Int i=100;
String result = i>100?"I'm big":"Not So Big";
System.out.println(result);
```

Same thing - but a lot cleaner.

Ternary Operator has three parts

1. The logical check
2. What to return if the check is true
3. What to return if the check is false
Chapter 7: Loops

Logical constructs and loops are part of any language. If you know it in any language you can directly apply to others – only some minor syntax difference will be distraction.

Do you want to keep repeating something for 5 times?

This is the general syntax

For count starting at one through five, repeat it.

In java we call this For loop and here is how we write it

```java
For
for( int i=0; i<5; i++ )  {
    System.out.println(i);
}
```

It simply prints 0 through 4 in each line.

What is happening?

For loop has 4 parts.

1. The initialization part:
   a. Here we created a variable ‘i’ of type integer, and set its initial value to 0.
2. A condition check
   a. If this condition true for loop will continue, else the loop is broken or stopped. Here we are checking i<5, ie, do it until I becomes 5 (or more)
3. An operation that will do in the beginning of each loop
   a. Here we are incrementing i
4. The operations in the loop itself
   a. Here we are printing out the value of i

Exercise 1:

Define an integer array, such as int[] array = {1,3,4,5,6,7,8,10}. List all the elements of the array using a for loop.


For Each ( we will cover this later)

For Each is an upgraded version of For. Here you can loop through a Collection of Objects. It makes sense only after you learn about Collections. So, it will be covered in JLA as a different post.

While

While is yet another way to loop. Syntax of the while as follows:

```java
while( condition is true ) {
```
The `do ... while` loop is another looping construct. It is similar to the `while` loop but with a different condition evaluation. In `while` loops, the condition is evaluated before the loop body is executed. In contrast, the `do ... while` loop evaluates the condition after the loop body is executed. This means that even if the condition is false when the loop is initiated, the loop body will still be executed at least once. Here is an example:

```java
int i = 0;
while (i < 5) {
    System.out.println(i);
    i++;
}
```

You can compare it with the `for` loop and see the difference. While is suitable in some context, from practical purposes, programmers mostly use the `for` loop.

### Do While

Do-while is yet another looping mechanism. This is similar to the `while` loop itself. The difference is that it checks the condition for true after the first iteration. This means that even if the condition is false when the loop is initiated, the loop body will still be executed at least once. Here is an example:

```java
int i = 0;
do {
    System.out.println(i);
} while (i < 5);
```

You can compare it with the `for` loop and see the difference. While is suitable in some context, from practical purposes, programmers mostly use the `for` loop.
Code samples for loops

```java
public static void loopThroughArray() {
    int[] students = new int[]{80, 90, 83, 70, 99, 87, 96, 85, 92, 95};
    for (int i=0; i<students.length; i++) {
        System.out.println(students[i]);
    }
}

public static void loopThroughArrayUsingWhile() {
    int[] students = new int[]{80, 90, 83, 70, 99, 87, 96, 85, 92, 95};
    int i=0;
    while (i<students.length) {
        System.out.println(students[i]);
        i++;//remove this for infinite loop test
    }
}

public static void loopThroughArrayUsingDoWhile() {
    int[] students = new int[]{80, 90, 83, 70, 99, 87, 96, 85, 92, 95};
    int i=0;
    do {
        System.out.println(students[i]);
        i++;
    } while (i<students.length);
}

public static void loopThroughArrayAndAdd() {
    int[] students = new int[]{80, 90, 83, 70, 99, 87, 96, 85, 92, 95};
    int result = 0;
    for (int i=0; i<students.length; i++) {
        result = result+students[i];
    }
}
```

Exercise:

Declare a two dimensional array. Add each row and output the result for each row. For each result check if the result is even or odd and print “Even”/“Odd”
Chapter 8: How to use Class and Objects

Now you have everything to get started with Java. You should be able to write small programs in Java and play around with it. However, there is one more section I would like to cover before we wind up this basic course. That is Java Classes, Objects and the usage. In the next level of courses we will get in to even more details. However you should know how Java class and Object could be utilized. We are just going to focus on the usage without worrying about the technicality about it. You can read about Object Oriented Concepts for a quick understanding. If you do not understand everything- do not worry about every details of it. That is not required at this point.

What is class and object?

Relating to real life, class is simply a generic concept. Furniture, Human, Animal – these are all examples of class.

Object as the name implies is something that really exists and you can touch and feel. A Table, Gandhi, a Lion these are examples of objects corresponding to the class we discussed above.

Classes give us functionality to be used. In Java there are many prebuilt classes. We will discuss couple of them to see how they could be used.

We will focus on String and Integer class.

String class represents a string of literals. For example “This is my place” is a String.

Here is a java example using String.

```java
public static void main(String args[]) {
    String sample = "This is my place";
    System.out.println(sample);
    String substring = sample.substring(8, sample.length());
    System.out.println(substring);
}
```

Output here will be:

This is my place

my place

Here when we look at it, String is giving us a function() called substring. We have used this function to get substring of the original String and we got “my place”.

String also gives you a function called length() which we used in the substring function.

Exercise:
1. There is a function called `chartAt()` in String class. Use this function and print out the value. Do a web search – this will help you learn how to find various functions in a java library
2. Identify 3 more functions in String and use it in your program.

**Initialization of an Object**

Just like physical objects, Java objects also need space. Java Objects sits in computer memory. So we say objects needs to be initialized in memory.

We did this in the line

String sample = “This is my place”;

When we did this, we allocated space for the String variable ‘sample’ and assigned the value of “This is my place”.

Proper way to allocate memory is by using `new()` operator. Remember addition operator? Similarly we have an operator to create new objects. It is called `new()`.

So we can also rewrite the above line like below:

String sample = new String(“This is my place”);

For many reasons such as better performance, java treats String as special and you can simply assign value directly to it without using new operator.

If you want to create an Integer object, you will use something like below

Integer intVar = new Integer(10);

Once you have created the object, you can start using the variable, intVar.

**Exercise:**

1. Identify functions in Integer class, and use it in a program.
Conclusion

Congratulations!!!

You have successfully completed the first course in Java. Now it is the time to play around with the language and be more proficient. More you try better you will get.

More tutorials will be added in Java Learning Academy. So you might want to keep checking JLA.

If you have any questions about anything, you may drop a comment in the JLA website.

Also, sign up for JLA mailing list so that the updates will come to your inbox automatically. You can do it here:

http://www.javalearningacademy.com/signup