

Forum:[Hack Forums](#)

Thread: [WTF's Java Guide for Beginners \[Under Construction\]](#)

WTF - November 26, 2013, 3:15 pm

[img]http://ucttraining.files.wordpress.com/2013/10/java_logo.jpg[/img]

Hey Hack Forums, I'm going to be writing a Java tutorial for you guys! Firstly, I'm going to start by explaining a bit about Java and what makes it different from most other languages. Java runs on what's known as the JVM or Java Virtual Machine, this allows it to run on many different types of computing devices and operating systems. You may have read the Java is slow, but this is not the case it may have been many years ago, but now Java runs as fast as many other programming languages which is why it's being used more and more for applications such as games.

You will need to get a Java IDE, an IDE is sort of a Word for programming, for Java I recommend eclipse.

<http://www.eclipse.org/>

OK, getting into the practical side of Java programming. Variables are a crucial part to any program, a variable is a small area in memory(RAM) which is given a name and can store a value. types of variables can store different types of values for example an int can store a whole number, floats can store a number with a decimal, strings can store letter numbers and some punctuation. Below is an example of declaring a variable in Java:

```
Code int number = 10;
```

Variables can be set to different values at any time in your program, but they only need to be declared once. Also note it is good to get in the habit of naming your variables of what they are and not get random letters.

Now we will move on to selection, selection is when your code can follow a specific route depending on the outcome of a condition/question. If statements are probably the most use form of selection.

```
Code if(number == 10){
```

```
    System.out.println("Number is equal to ten!");
```

```
}
```

this if statements would run the code " System.out.println("Number is equal to ten!");" only if number was equal to 10 and if it is not it would simply skip that part of the code. Note: System.out.println prints text into the console window.

As well as having just an if statement you can have else if and else. else if works the same as an if statement, but is used if you want to have more than one possible outcome and else statements are for when you want something to happen only if none of the conditions are met.

```
Code if(number == 10){
```

```
    System.out.println("Number is equal to ten!");
```

```
}
```

```
else if(number == 5){
```

```
    System.out.println("Number is equal to five!");
```

```
}
```

```
else{
    System.out.println("Number is equal to something other then 5 or 10!");
}
```

Next I will be introducing you to iteration or loops. The two most common types of loop are while and for loops. a while loop repeats the code inside while the condition is met.

```
Code int number = 10;
```

```
while(number > 0){
    System.out.println("Number is equal to: " + number);
    number = number - 1;
}
```

The above example will print out:

Number is equal to 10

Number is equal to 9

And so on...

Loops can be used to repeat code without having to re-write it or when you don't know how many times the code will be repeated for. Note: if you want to add or subtract 1 from a number you can use variable++; or variable--;. Below I will explain the for loop:

```
Code int loopCount;
```

```
for(loopCount = 10; loopCount >= 0; loopCount--){
```

```
    System.out.println("loopCount is equal to: " + );
```

}OK, I'll break this down, the first part in the for loop can be used for setting or declaring variable that you will use for the loop condition, the second part is the condition and the last part is for changing the value of the variable that you are using for the loop. In this case the loop will run ten times as every time the loop is executed one is subtracted from loopCount.

Operators are often used in conditions, operators you may already have learned in this tutorial are "==" , ">" and "<".

== if equal to

> if greater then

< if less then

<= if equal to or less then

=> if equal to or greater then

|| or "if(number == 1 || numberTwo == 2)"

&& and "if(number == 1 && numberTwo == 2)"

OK guys I have decided to continue my tutorial that I had earlier abandoned. So, moving onto methods. Methods are great, they help you structure your code and can prevent you from having to re-write code. Now we have already been using a method, the main method(As shown below). The main method is the first bit of code that gets executed(run) when your application starts.

```
Code public static void main(str[] args){
    add(7, 4);
}
```

```
public static int add(int number1, int number2){
    return number1 + number2;
```

```
}
```

If you create a new method as I have done above it will not run unless I tell it to in the main method. You may notice there are some numbers in the brackets when I call/run the method, these are parameters. You choose what parameters you want when you declare a method, so I've told the program that I need two numbers when my method is called and in this case I have given the method 7 and 4. Now you may be wondering about the 'public int' part just before the name of the method don't worry about the public for the moment, we will get into that later. now the int is what we want the method to give us when it is finished, this can be any data type, so int, string, char, double, float, etc. in the method called add we are returning an integer which is the sum of the two numbers we gave the method.

Arrays if you have not already guessed are lists of variables. Below is an example of declaring an array. An array is not a data type, but an object. the following is an array of integers, in fact it holds 4 integers.

```
Code int[] intArray1 = new int[4];
```

```
intArray1[0] = 1;
```

```
intArray1[1] = 43;
```

The first section or variable in the array is accessed with the number 0 not 1, this can sometimes be muddled up by beginners, so keep an eye out. You can have any type of array that you want, you could have an array of strings, chars, float, etc.

Coming soon:

Working with strings

classes/OOP

access modifiers

MySQL

Multi-threading

Networking

Skulk - November 26, 2013, 4:47 pm

Looks really good, keep up the good work!

?ovu - November 26, 2013, 4:51 pm

You do know you can find all of this and more from Oracle's direct website?

<http://docs.oracle.com/javase/tutorial/>

WTF - November 26, 2013, 5:16 pm

(Nov 26, 2013 05:51 PM)Rainfur Wrote: [\[CITE\]](#) You do know you can find all of this and more from Oracle's direct website? <http://docs.oracle.com/javase/tutorial/>

If you are looking for beginner friendly tutorials, then you should search Google or maybe even YouTube.

But this may have been what you are looking for:

<http://docs.oracle.com/javase/tutorial/getStarted/index.html>

vitafortis - November 26, 2013, 5:49 pm

(Nov 26, 2013 06:16 PM)wtfwtfwtf Wrote: [\[CITE\]](#) If you are looking for beginner friendly tutorials, then you should search Google or maybe even YouTube.

But this may have been what you are looking for:

<http://docs.oracle.com/javase/tutorial/getStarted/index.html>

And here's where I disagree. The Java Docs are beginner friendly. Sources such as Stack Over Flow and youtube which may teach bad practices are not. Why? Beginners don't know the

differences between good and bad conventions so they won't know how to use code they find.

WTFζ - August 17, 2014, 5:25 am

I've recently started continuing my guide on Java for beginners. I hope you find it informative. :)

Cosa Nostra - August 17, 2014, 5:27 am

Eh... No offense, but most of this is already in a sticky.

WTFζ - August 17, 2014, 5:31 am

(Aug 17, 2014 05:27 AM)Cosa Nostra Wrote: [\[CITE\]](#) Eh... No offense, but most of this is already in a sticky.

Yeah, I know, but I enjoy writing it and it gives me something else to do while I don't have a job.

hellrider10 - August 17, 2014, 8:48 am

(Nov 26, 2013 04:15 PM)WTFζ Wrote: [\[CITE\]](#) Code public static void main(str[] args){
 add(7, 4);
}

```
public int add(int number1, int number2){  
    return number1 + number2;  
}
```

You can't call a non-static method from a static one. You should either instantiate the class and call the method on it or make the add method static.

WTFζ - August 17, 2014, 8:57 am

(Aug 17, 2014 08:48 AM)hellrider10 Wrote: [\[CITE\]](#) You can't call a non-static method from a static one. You should either instantiate the class and call the method on it or make the add method static.

Your right, I've been away from Java from a few months should have started a project to sharpen my skills.