Java Crash Course Roadmap

Short Intro

Java is a multi-platform, object-oriented, and network-centric programming language. It is among the most used languages out there. There are plenty of fancy definitions just one Google search away, but the important thing to keep in mind is that Java can be used to solve a great range of problems.

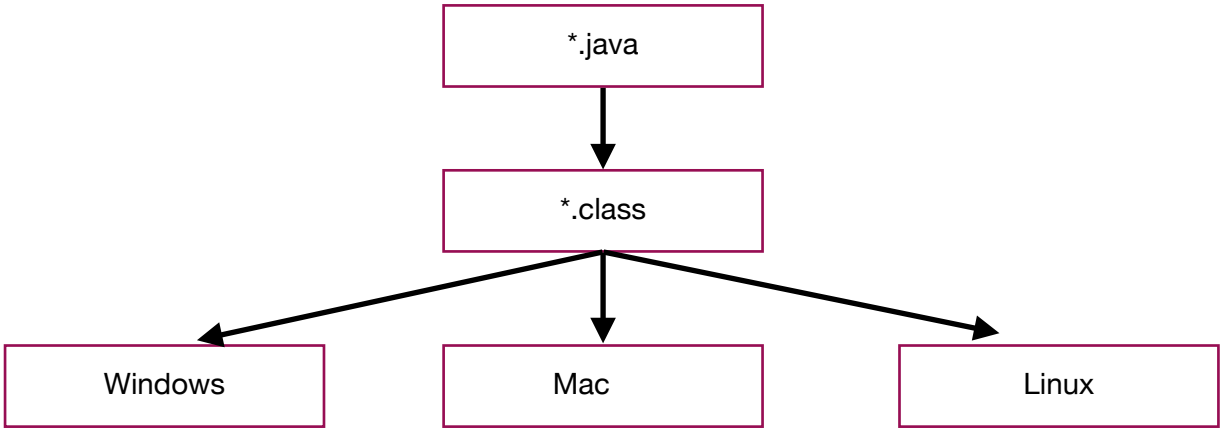
It is considered as one of the fast, secure, and reliable programming languages preferred by most organizations to build their projects.

What is Java used for?
Here are some important Java use cases that make it a really powerful programming language:

- It is used for developing Android Apps
- Helps you to create Enterprise Software
- Wide range of Mobile java Applications
- Scientific Computing Applications
- Use for Big Data Analytics
- Java Programming of Hardware devices
- Used for Server-Side Technologies like Apache, JBoss, GlassFish, etc.

A lot of people view getting started with Java as an easy study path because it is indeed easy to install and write the first “Hello world” program. However, diving directly into it can lead to some confusion and mistakes down the road and that’s why it is important to cover a few key concepts in the beginning.

First off, we write code that suspiciously similar to our language, but we know for a fact that computers don’t speak the same language. That is because Java is a high level programming language which is actually a few steps away from talking to the computer directly. So what we’re communicating with Java is then converted to what a computer understands. Here’s how:



So the code we write is going to be known as the source code. These files have the extension .java. The source code is getting compiled into bytecode (.class extension; it’s actually an intermediate state) and all of these operating systems will be able to use that bytecode. Of course this is not the full story and this doesn’t happen by magic, there are a few more components that make all of this possible. First we have the JDK (Java Development Kit) - this is what we use to develop our Java applications. Then there’s the JRE (Java Runtime Environment)

which is the one allowing the applications to run on the different operating systems. So in order to remember which is which, think of where you'll be running the application and that will be the JRE. Now what makes it possible for the same code to run on different operating systems is the JVM. JVM stands for Java Virtual Machine, and it is a virtual machine that enables a computer to run Java programs and also converts the Java bytecode into machine language.
The JVM is part of the Java Run Environment (JRE). In other programming languages, the compiler will produce machine code for a particular system. However, the Java compiler produces (in the JDK) produces code for the JVM, which in turn will "translate" it further down.

Here you can find some more information on JMV architecture.

TODO & Important concepts to cover:

1.Download and install Java

Comprehensive installation guide here.

2. Download and install IntelliJ IDEA Community Edition

https://www.jetbrains.com/idea/download/#section=windows

3. Write Your First Java Program in IntelliJ IDEA + Configuration

https://www.jetbrains.com/help/idea/creating-and-running-your-first-java-application.html

4. Java Data Types and Variables

https://beginnersbook.com/2017/08/data-types-in-java/

5. Operators

https://beginnersbook.com/2017/08/operators-in-java

6. OOP Concepts in Java

a) Classes

https://beginnersbook.com/2013/04/oops-concepts/#2

b) Objects

https://beginnersbook.com/2013/04/oops-concepts/#1

c) Constructors

https://beginnersbook.com/2013/04/oops-concepts/#3

d) The 4 key concepts of OOP

https://beginnersbook.com/2013/04/oops-concepts/#4

e) Abstract classes & Interfaces

https://beginnersbook.com/2013/04/oops-concepts/#9

7. Overriding vs Overloading

https://beginnersbook.com/2014/01/difference-between-method-overloading-and-overriding-in-java/

8. Abstract classes vs Interfaces

https://beginnersbook.com/2013/05/abstract-class-vs-interface-in-java/

9. Java Control Statements

a) If-else statement

https://beginnersbook.com/2017/08/if-else-statement-in-java/

b) Switch-case

https://beginnersbook.com/2017/08/java-switch-case/

c) For loop

https://beginnersbook.com/2015/03/for-loop-in-java-with-example/

d) While loop

https://beginnersbook.com/2015/03/while-loop-in-java-with-examples/

e) Do-while loop

https://beginnersbook.com/2015/03/do-while-loop-in-java-with-example/

f) Continue statement

https://beginnersbook.com/2017/08/java-continue-statement/

g) Break statement

https://beginnersbook.com/2017/08/java-break-statement/

10.Enums

https://beginnersbook.com/2014/09/java-enum-examples/

11. Collections - List vs Set vs Map

Collections - general overview: https://www.javatpoint.com/collections-in-java

ArrayList - https://beginnersbook.com/2013/12/java-arraylist/

Set - https://beginnersbook.com/2013/12/hashset-class-in-java-with-example/

Map - https://beginnersbook.com/2013/12/hashmap-in-java-with-example/

Please try out the methods for each collection type

12. Package & access modifiers

Packages - https://beginnersbook.com/2013/03/packages-in-java/

Access modifiers - https://beginnersbook.com/2013/05/java-access-modifiers/

13. Other reserved key words

Static - https://beginnersbook.com/2013/04/java-static-class-block-methods-variables/

Super - https://beginnersbook.com/2014/07/super-keyword-in-java-with-example/

Final - https://beginnersbook.com/2014/07/final-keyword-java-final-variable-method-class/

14. Exception handling

https://beginnersbook.com/2013/04/java-exception-handling/

a) Try catch

https://beginnersbook.com/2013/04/try-catch-in-java/

b) Checked and unchecked exceptions

https://beginnersbook.com/2013/04/java-checked-unchecked-exceptions-with-examples/

c) Finally block

https://beginnersbook.com/2013/04/java-finally-block/

d) Try-catch-finally

https://beginnersbook.com/2013/05/flow-in-try-catch-finally/

e) Throw vs throws

Throw exceptions - https://beginnersbook.com/2013/04/throw-in-java/

Example - https://beginnersbook.com/2013/12/throw-keyword-example-in-java/

Throws clause - https://beginnersbook.com/2013/04/java-throws/

Example - https://beginnersbook.com/2013/12/throws-keyword-example-in-java/

15. hashcode() - equals() contract

https://dzone.com/articles/working-with-hashcode-and-equals-in-java