http://www.oracle.com/technetwork/java/index-138747.html

http://www.oracle.com/technetwork/java/basicjava2-138746.html

https://www.javacodegeeks.com/2014/07/step-by-step-path-to-becoming-a-great-software-developer.html

Java developers knowledge expectation changes based on the profile. In this post I have divided it into 3 profiles:

College Graduate,

Experienced Java Developer,

Experienced Java Web Developer.

7 Things a College graduate must know to get a Java developer job

If you are a college graduate with no job experience then as a Java developer you need to understand the following basic things.

How Java Virtual Machine works? e.g. (Platform Independence, Garbage Collection, class files etc)

What are the Object Oriented Programming Concepts Implemented in Java?

Multi-threading

Java Collection framework

Good understanding of data types and few java.lang classes like String, Math, System etc. java.io stream concepts.

Understand concept of Swing/AWT event based programming. Do not spend a lot of time on this but understand the best practices.

Servlets and JSP concepts.

9 Things an experienced Java Developer must know to thrive

If you are a experienced professional then as a Java developer you may also need to understand the following basic things in addition to the ones listed above.

Understand design patterns and its usage in Java

Improvements on language from major version changes (Generics, Annotations, Enums, ...).

Coding Conventions.

Build tool (Ant) or Project Management Tool (Maven).

Version control System like GIT/SVN/Perforce/Clearcase.

Apache Commons Libraries & few other common open source libraries.

Continuous Integration Tools e.g. Jenkins and Hudson.

Unit testing - e.g. Junit and TestNG

Unit testing Mocking libraries like Mockito

Fundamental understanding of XML

Understand Business layer frameworks - like Spring

Understanding dependency injection (e.g. Spring, Google Guice and Plain Java Dependency injection)

4 Things a Java Web Developer (JEE) Developer must know

If you are a experienced professional working on Web based development then as a JEE developer you also need to understand the following basic things in addition to the ones (7+9) listed above.

Understanding of MVC Frameworks - Open source J2EE frameworks like - Struts, Webwork, Spring MVC, Tapestry.

Important Note : A lot of Front End UI development is now shifted to JavaScript frameworks. Therefore do not focus on Java based frameworks that focus on user interface (e.g. JSF or related frameworks). Instead learn JavaScript related frameworks like Angular.js or Backbone.js

Fundamental understanding of Web Services and REST based service development.

Good understanding of Web/Application server like Tomcat, Glassfish, WebLogic, WebSphere, Jetty etc.

Unix environment - A working knowledge of Unix environment can be beneficial as most of the Java servers are hosted on Unix based environment in production.

Looking at the list of things it really feels difficult for a person to know each and everything in depth. As I already said it is more important and valuable to know the language fundamentals thoroughly and rest can be learned quickly when required.

https://devskiller.com/how-to-screen-java-developers-skills-to-find-the-best-guide-for-it-recruitment/

https://www.quora.com/What-skills-do-self-taught-programmers-commonly-lack

programer path

Systems Architecture (Binary logic and the physics of implementation, parts of a computer and how they interact, understanding the speed order of magnitude of each part, computing as a hierarchy of caches, cache locality.) [1 week]

Operating Systems (The kernel: abstracting hardware into useful services, POSIX and how does it relates to the standard libraries of your favorite programming language.) [1 week]

Data Structures and Algorithms (A compressed version of your standard data structures/algorithms class, taught by Bloomberg instructors.) [2 weeks]

Decidability and tractability (Formal models of computing, Complexity Theory, important complexity classes, and reductions to known problems.) [1 week]

Programming Languages (Historical problems and the language features that solve them, type systems, polymorphism, higher order functions, scope and closure, type safety, lexing, parsing, optimization, JIT, etc.) [2 weeks]

Special topic: Concurrency (The failure of Moore's Law, shared memory models, Threads, locking, actor models, the c10k problem, asynchronous IO, vector clocks, consensus algorithms, etc.) [1 week]

Special topics: Artificial Intelligence and Machine Learning (State spaces and search, probability and bayes networks, planning, supervised learning, regression, overfitting, unsupervised learning, reinforcement learning) [2 weeks]

Read lot of puzzle and how to implement.

The Java professional need following skills (depends on the experience level):

Core Java

OO Concepts

Java Design Patterns (GoF design patterns)

Java EE Concepts ( Java EE Enterprise Application tiers knowledge for OCMJEA Certification - MyExamCloud )

JSP and Servlets

JSF

EJB

JPA

Java Web Services (JAX-RS, JAX-WS)

Java EE Design Patterns

Application Servers (JBoss, GlashFish, WebSphere..etc)

Databases (MySQL, Oracle….SQL queries)

UML Diagrams (Use Case, Class, Sequence, Component, Deployment…)

Process Knowledge (Agile…)

1. Have strong foundation and understanding on OO Principles

For a Java developer, having strong understanding on Object Oriented Programming is a must. Without having a strong foundation on OOPS, one can't realize the beauty of an Object Oriented Programming language like Java. If you don't have good idea on what OOPS is, even though you are using OOP language you may be still coding in procedural way. Just studying OO principle definitions won't help much. We should know how to apply those OO principles in designing a solution in OO way. So one should have a strong knowledge on Object modeling, Inheritance, Polymorphism, Design Patterns.

2. Master the core APIs

It doesn't matter how strong you are in terms of theoretical knowledge if you don't know the language constructs and core APIs. In case of Java, one should have very strong hands-on experience with core APIs like java.lang.\*, I/O, Exceptions, Collections, Generics, Threads, JDBC etc. When it comes to Web application development, no matter which framework you are using having strong knowledge on Servlets, JSPs is a must.

3. Keep coding

Things look simpler when talking about them theoretically. We can give a solution to a problem very easily in theory. But we can realize the depth of the problem when we start implementing our approach. You will come to know the language limitations, or design best practices while coding. So keep coding.

4. Subscribe to forums

We are not alone. There are lots of people working on the same technologies that we are working on. While doing a simple proof of concept on a framework may not give you real challenges, when you start using it on real projects you will face weird issues and you won't find any solution in their official documentation. When starting to work on a new technology the best and first thing to do is subscribe to the relevant technology forums. Whatever the issue you are facing, someone else in the world might have already faced it earlier and might have found the solution. And it would be really really great if you can answer the questions asked by other forum users.

5. Follow blogs and respond

As I already told you are not alone. There are thousands of enthusiastic technology freaks around the world blogging their insights on technology. You can see different perspectives of same technology on blogs. Someone can find great features in a technology and someone else feels its a stupid framework giving his own reasons of why he felt like that. So you can see both good and bad of a technology on blogs. Follow the good blogs and respond/comment on posts with your opinion on that.

6. Read open source frameworks source code

A good developer will learn how to use a framework. But if you want to be an outstanding developer you should study the source code of various successful, popular frameworks where you can see the internal working mechanism of the framework and lot of best practices. It will help a lot in using the frameworks in very effective way.

7. Know the technology trends

In the open source software development technology trends keep on changing. By the time you get good idea on a framework that might become obsolete and some brand new framework came into picture with super-set of features. The problem which you are trying to solve with your current framework may be already solved by the new framework with a single line of configuration. So keep an eye on whats coming in and whats going out.

8. Keep commonly used code snippets/utilities handy

Overtime you may need to write/copy-paste same piece of code/configuration again and again. Keeping those kind of configuration snippets like log4.properties, jdbc configuration etc and utilities like StringUtils, ReflectionUtils, DBUtils will be more helpful. I know it itself won't make you outstanding developer. But just imagine some co-developer asks you to help in fetching the list of values of a property from a collection of objects and then you just used your ReflectionUtil and gave the solution in few minutes. That will make you outstanding.

9. Know different development methodologies

Be familiar with various kinds of methodologies like Agile, SCRUM, XP, Waterfall etc. Nowadays choosing the development methodology depends on the client. Some clients prefer Agile and some clients are happy with waterfall model. So having an idea on various methodologies would be great.

10. Document/blog your thoughts on technology

In day to day job you may learn new things, new and better way of doing things, best practices, architectural ideas. Keep documenting those thoughts or blog it and share across the community. Imagine you solved a weird problem occurred while doing a simple POC and you blogged about it. May be some developer elsewhere in the world is facing the same issue on a production deployed application. Think how important that solution for that developer. So blog your thoughts, they might be helpful for others or to yourself.

http://eclipsesource.com/blogs/2012/09/18/must-reads-for-java-developers-from-beginner-to-professional-2/