Java-Success.com

Industrial strength Java/JEE Career Companion for those who want to go places

01: ♦ ♥ 17 Java overview interview questions and answers

Posted on August 17, 2014 by Arulkumaran Kumaraswamipillai — 1 Comment ↓

You may also like

12 Java String class interview Q&A | Web Services interview Q&A | Java EE Overview interview Q&A | Multithreading scenarios in Java applications interview Q&A

Q1. Why use Java?
A1. One needs to use the best tool for the job, whether that tool is Java or not. When choosing a technology to solve your business problems, you need to consider many factors like

9 tips to earn more | What can u do to go places? | **945+** members. LinkedIn Group. **Reviews**

600+ Full Stack Java/JEE Interview Q&As ♥Free ♦FAQs

development cost, infrastructure cost, ongoing support cost, robustness, flexibility, security, performance, etc.

— Java provides **client technologies**, **server technologies**, and **integration technologies** to solve small scale to very large scale business problems.

— Firstly, Java is a **proven** and **matured** technology used in many mission critical projects, and there are millions of developers world wide, thousands of frameworks, tools, and libraries for it, and millions of sites, blogs, and books to find relevant information.

— The emergence of **open-source technologies** has truly made Java a powerful competitor in the server and integration technology space. You can always find a proven framework that best solves your business problems.

— The platform also comes with a rich set of APIs. This means developers spend less time writing support libraries, and more time developing content for their applications.

— Built-in support for multi-threading, socket communication, and automatic memory management (i.e. automatic garbage collection).

Q2. Is Java a 100% Object Oriented (OO) language? if yes why? and if no, why not?
A2. I would say Java is not 100% object oriented, but it embodies practical OO concepts. There are 6 qualities to make a programming language to be pure object oriented. They are:

1. Encapsulation – data hiding and modularity.
2. Inheritance – you define new classes and behavior based on existing classes to obtain code reuse.
3. Polymorphism – the same message sent to different objects results in behavior that's dependent on the nature of the object receiving the message.
4. All predefined **types are objects**.
5. All operations are performed by **sending messages to**

objects.
6. All user defined **types are objects**.

points 1- 3, stands for PIE (**P**olymorphism, **I**nheritance, and **E**ncapsulation).

Reason #1: The main reason why Java cannot be considered 100% OO is due to its existence of 8 primitive variables (i.e. violates point number 4) like int, long, char, float, etc. These data types have been excused from being objects for simplicity and to improve performance. Since primitive data types are not objects, they don't have any qualities such as inheritance or polymorphism. Even though Java provides immutable wrapper classes like Integer, Long, Character, etc representing corresponding primitive data as objects, the fact that it allowed non object oriented primitive variables to exist, makes it not fully OO.

Reason #2: Another reason why Java is considered not full OO is due to its existence of static methods and variables (i.e. violates point number 5). Since static methods can be invoked without instantiating an object, we could say that it breaks the rules of encapsulation.

Reason #3: Java does not support multiple class inheritance to solve the diamond problem because different classes may have different variables with same name that may be contradicted and can cause confusions and result in errors. In Java, any class can extend only one other class, but can implement multiple interfaces.

We could also argue that Java is not 100% OO according to this point of view. But Java realizes some of the key benefits of multiple inheritance through its support for multiple interface inheritance and in **Java 8, you can have multiple behavior (not state) inheritance** as you can have default methods in interfaces.

Reason #4: Operator overloading is not possible in Java except for string concatenation and addition operations. String concatenation and addition example,

```
System.out.println(1 + 2 + ”3”);//outputs 33
System.out.println(“1” + 2 + 3); //outputs 123
```

Since this is a kind of polymorphism for other operators like * (multiplication), / (division), or – (subtraction), and Java does not support this, hence one could debate that Java is not 100% OO. Working with a primitive in Java is more elegant than working with an object like BigDecimal. For example,

```
int a,b, c;
//without operator overloading
a = b – c * d
```

What happens in Java when you have to deal with large decimal numbers that must be accurate and of unlimited size and precision? You must use a BigDecimal. BigDecimal looks verbose for larger calculations without the operator overloading as shown below:

```
BigDecimal b = new BigDecimal(“25.24”);
BigDecimal c = new BigDecimal(“3.99”);
BigDecimal d = new BigDecimal(“2.78”);
BigDecimal a = b.subtract(c).multiply(d); //verbo
```

Also, the last line above is wrong. The rules of precedence have changed. With chained method calls like this, evaluation is strictly left-to-right. Instead of subtracting the product of c and d from b, we are multiplying the difference between b and c by d. We would have to rewrite the last line as shown below:

```
BigDecimal a = b.subtract(c.multiply(d)); //corre
```

So, it is error prone as well. Another point is that the BigDecimal class is immutable and, as such, each of the “operator” methods returns a new instance. In future Java versions, you may have operator overloading for BigDecimal, and it would make your code more readable as shown below.

```
BigDecimal a = b - (c * d); //much better
```

Q3. What is the difference between C++ and Java?
A3. Both C++ and Java use similar syntax and are Object Oriented, but:

1) Java does not support pointers. Pointers are inherently tricky to use and troublesome.

2) Java does not support multiple inheritances because it causes more problems than it solves. Instead Java supports multiple interface behavior inheritance (In Java 8, you can have interfaces with default & static methods, but can't have state), which allows an object to inherit many method signatures from different interfaces with the condition that the inheriting object must implement those inherited methods.

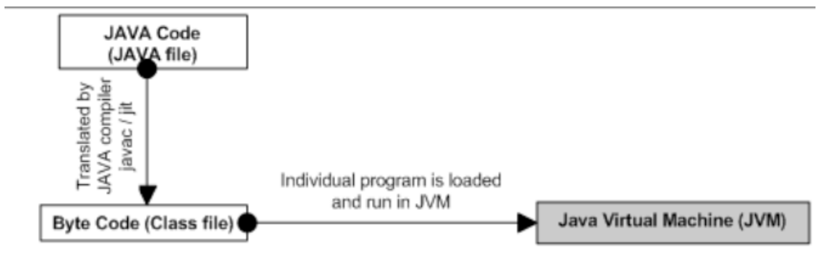
3) Java does not support destructors but rather adds a finalize() method. Finalize methods are invoked by the garbage collector prior to reclaiming the memory occupied by the object, which has the finalize() method. This means you do not know when the objects are going to be finalized. Avoid using finalize() method to release non-memory resources like file handles, sockets, database connections etc because Java has only a finite number of these resources and you do not know when the garbage collection is going to kick in to release these resources through the finalize() method.

4) Java does not include structures or unions. Java make use of the Java Collection framework.

5) All the code in Java program is encapsulated within classes therefore Java does not have global variables or functions.

6) C++ requires explicit memory management, while Java includes automatic garbage collection.

Q4. What is the main difference between the Java platform and the other software platforms?
A4. Java platform is a software-only platform, which runs on top of other hardware-based platforms like Unix, Windows, Mac OS, etc.


JAVA Code
(JAVA file)
Translated by JAVA compiler javac / jit
Byte Code (Class file)
Individual program is loaded and run in JVM
Java Virtual Machine (JVM)

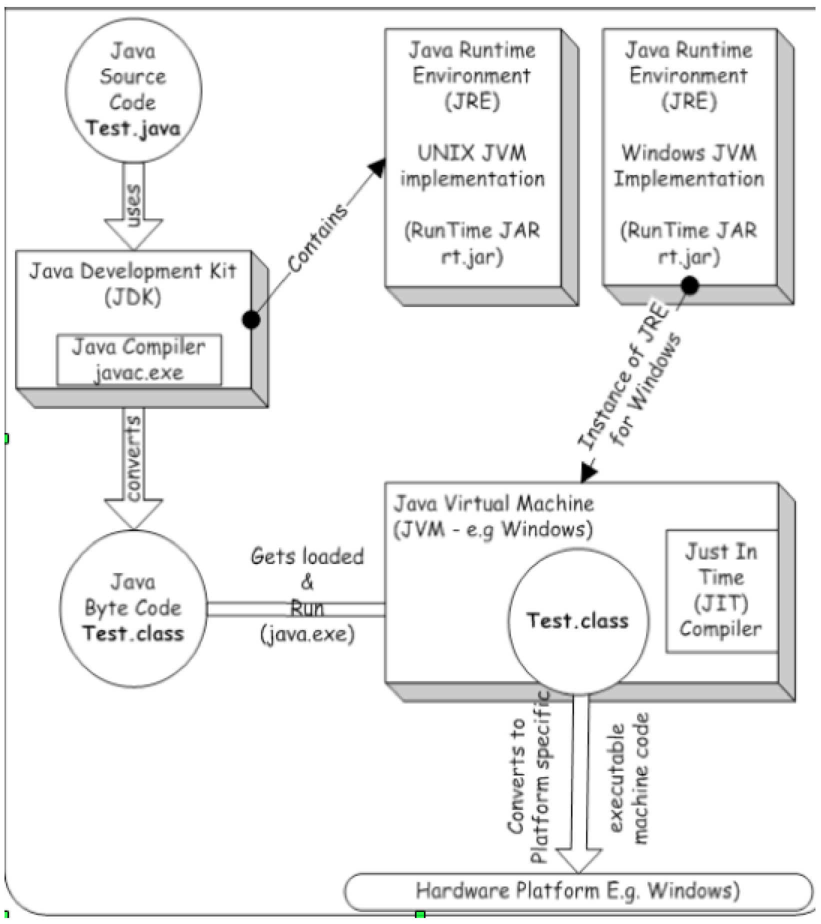
Java platform

The Java platform has 2 components:

1) Java Virtual Machine (JVM) – 'JVM' is a software that can be ported onto various hardware platforms. Byte codes are the machine language of the JVM.

2) Java Application Programming Interface (Java API) – is nothing but a set of classes and interfaces that come with the JDK. All these classes are written using the Java language and contains a library of methods for common programming tasks like manipulating strings and data structures, networking, file transfer, etc. The source *.java files are in the src.zip archive and the executable *.class files are in the rt.jar archive.

Q5. How would you differentiate JDK, JRE, JVM, and JIT?
A5. There is no better way to get the big picture than a diagram.


Java Source Code Test.java
uses
Java Development Kit (JDK)
Java Compiler javac.exe
Contains
Java Runtime Environment (JRE) UNIX JVM implementation (RunTime JAR rt.jar)
Java Runtime Environment (JRE) Windows JVM Implementation (RunTime JAR rt.jar)
Instance of JRE for Windows
converts
Java Byte Code Test.class
Gets loaded & Run (java.exe)
Java Virtual Machine (JVM - e.g Windows)
Test.class
Just In Time (JIT) Compiler
Converts to Platform specific
executable machine code
Hardware Platform E.g. Windows)

JDK, JRE, JVM, and JIT

1) **JDK:** You can download a copy of the Java Development Kit (JDK) for your operating system like Unix, Windows, etc.

2) **JRE:** Java Runtime Environment is an implementation of the JVM. The JDK typically includes the Java Runtime Environment (JRE) which contains the virtual machine and other dependencies to run Java applications.

3) **JIT:** A JIT is a code generator that converts Java byte code into native machine code. Java programs invoked with a JIT generally run much faster than when the byte code is executed by the interpreter. The JIT compiler is a standard tool that is part of the JVM and invoked whenever you use the Java interpreter command. You can disable the JIT compiler using the -Djava.compiler=NONE option to the Java VM. You might want to disable the JIT compiler if you are running the Java VM in remote debug mode, or if you want to see source

line numbers instead of the label (Compiled Code) in your Java stack traces.

Q6. Is it possible to convert byte code into source code?
A6. Yes. A Java decompiler is a computer program capable of reversing the work done by a compiler. In essence, it can convert back the byte code (i.e. the .class file) into the source code (i.e the .java file). There are many decompilers that exist today, but the most widely used JD – Java Decompiler is available both as stand-alone GUI program and as an eclipse plug-in.

Q7. When would you use a decompiler?
A7.

1) When you have *.class files and you do not have access to the source code (*.java files). For example, some vendors do not ship the source code for java class files or you accidentally lost (e.g deleted) your source code, in which case you can use the Java decompiler to reconstruct the source file.

2) Another scenario is that if you generated your .class files from another language like a groovy script, using the groovyc command, you may want to use a Java decompiler to inspect the Java source code for the groovy generated class files to debug or get a better understanding of groovy integration with Java.

3) To ensure that your code is adequately obfuscated before releasing it into the public domain.

4) Fixing and debugging .class files when developers are slow to respond to questions that need immediate answers. To learn both Java and how the Java VM works.

5) Learn and debug how code with generics has been converted after compilation.

Q8. Is it possible to prevent the conversion from byte code into source code?
A8. If you want to protect your Java class files from being decompiled, you can take a look at a Java obfuscator tool like yGuard or ProGuard, otherwise you will have to kiss your intellectual property good bye.

Q9. What are the two flavors of JVM?
A9. Client mode and server mode.

Client mode is suited for short lived programs like stand-alone GUI applications and applets. Specially tuned to reduce application start-up time and memory footprint, making it well suited for client applications. For example: c:\> java -client MyProgram

Server mode is suited for long running server applications, which can be active for weeks or months at a time. Specially tuned to maximize peak operating speed. The fastest possible operating speed is more important than fast start-up time or smaller runtime memory footprint. c:\> java -server MyProgram

Q10. How do you know in which mode your JVM is running?
A10. c:\> java -version

Q11. What are the two different bits of JVM? What is the major limitation of 32 bit JVM?
A11. JVMs are available in 32 bits (-d32 JVM argument) and 64 bits (-d64 JVM argument). 64-bit JVMs are typically only available from JDK 5.0 onwards. It is recommended that the 32-bit be used as the default JVM and 64-bit used if more memory is needed than what can be allocated to a 32-bit JVM. The Oracle Java VM cannot grow beyond ~2GB on a 32bit server machine even if you install more than 2GB of RAM into your server. It is recommended that a 64bit OS with larger memory hardware is used when larger heap sizes are required. For example, >4GB is assigned to the JVM and used for deployments of >250 concurrent or >2500 casual users.

Q12. What are some of the JVM arguments you have used in your projects?

A12. To set a system property that can be retrieved using System.getPropety("name");

```
$java -Dname=value MyApp
```

To set the classpath: -cp or -classpath

```
$java -cp library.jar MyApp
```

To set minimum and maximum heap sizes:

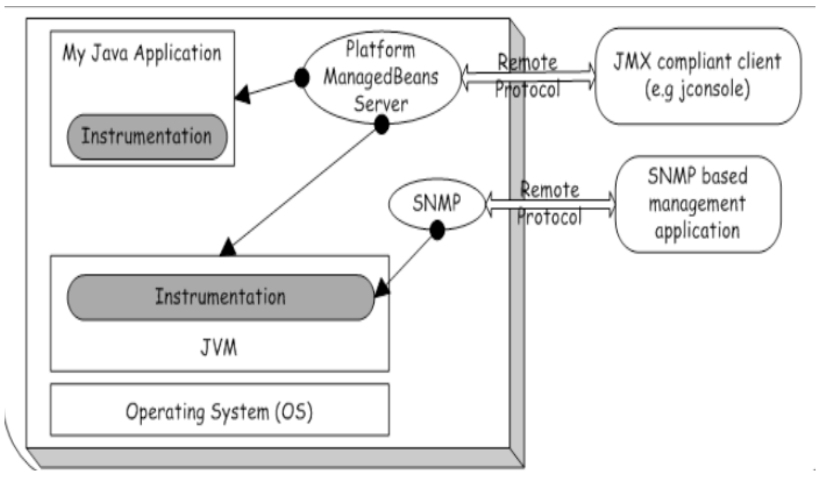
```
$java -Xms1024 -Xmx1024 MyApp
```

To set garbage collection options:

```
$ java -Xincgc MyApp
```

Q13. How do you monitor the JVMs?

A13. Since Java SE 5.0, the JRE provides a mean to manage and monitor the Java Virtual Machine. It comes in two flavors:



JVM Monitoring

1) The JVM has built-in instrumentation that enables you to monitor and manage it using **Java Management eXtension (JMX)**. You can also monitor instrumented applications with JMX. To start a Java application with the management agent for local monitoring, set the following JVM argument when you run your application.

```
$JAVA_HOME/bin/java -Dcom.sun.management.jmxremot

```

To start the JConsole:

```
$JAVA_HOME/bin/jconsole
```

2) The other is a **Simple Network Management Protocol (SNMP)** agent that builds upon the management and monitoring API of the Java SE platform, and exposes the same information through SNMP. SNMP events can be sent **Splunk**. Nagios, Zenoss, OpenNMS, and netsnmp are some of the popular SNMP tools.

Q14. What is a jar file? How does it differ from a zip file?
A14. The jar stands for Java ARchive. A jar file usually has a file name extension .jar. It mainly contains Java class files but any types of files can be included. For example, XML files, properties files, HTML files, image files, binary files, etc. You can use the "jar" application utility bundled inside /JDK1.6.0/jre/bin to create, extract, and view its contents. You can also use any zip file utility program to view its contents. **A jar file cannot contain other jar files.**, whereas a war or ear file used in Java EE.

Basically, a jar file is same as a zip file, except that it contains a META-INF directory to store meta data or attributes. The most known file is META-INF/MANIFEST.MF. When you create a JAR file, it automatically receives a default manifest file. There can be only one manifest file in an archive. Most uses of JAR files beyond simple archiving and compression require special information to be in the manifest file. For example,

— If you have an application bundled in a JAR file, you need some way to indicate which class within the JAR file is your application's entry point. The entry point is the class having a method with signature public static void main(String[ ] args). For example, Main-Class: Test.class

— A package within a JAR file can be optionally sealed, which means that all classes defined in that package must be archived in the same JAR file. You might want to seal a package, for example, to ensure version consistency among the classes in your software or as a security measure.

Name: myCompany/myPackage/
Sealed: true

Q15. What do you need to develop and run Java programs? How would you go about getting started?
A15. **Step 1:** Download and install the Java Development Kit (JDK) SE (Standard Edition) for your operating system (e.g. Windows, Linux, etc) and processor (32 bit, 64 bit, etc) .

Step 2: Configure your environment. The first environment variable you need to set is the **JAVA_HOME**.

```
JAVA_HOME=C:\DEV\java\jdk-1.6.0_11 #on windows
export JAVA_HOME=/usr/java/jdk-1.6.0_11  #on Unix

```

The second environment variable is **PATH**.

```
PATH=%JAVA_HOME%\bin; #on windows
export PATH=$PATH:$JAVA_HOME/bin #on Unix

```

Step 3: Verify your configurations with the following commands:

```
echo %JAVA_HOME% #windows
echo %PATH%  #windows
$ echo $JAVA_HOME #Unix
$ echo $PATH   #Unix

```

Step 5: Verify your installation with

```
$ javac -version
$ java -version

```

Q16. How do you create a class under a package in Java? What is the first statement in Java?

A16. ou can create a class under a package as follows with the package keyword, which is the first keyword in any Java program followed by the import statements. The java.lang package is imported implicitly by default and all the other packages must be explicitly imported. The core Java packages like java.lang.*, java.net.*, java.io*, etc and its class files are distributed in the archive file named rt.jar.

```
package com.xyz.client ;
import  java.io.File;
import  java.net.URL;

```

Q17. What do you need to do to run a class with a main( ) method in a package? For example: Say, you have a class named Pet in a project folder C:\projects\Test\src and package named com.xyz.client, will you be able to compile and run it as it is?

A17. **Step 1:** Write source code "Pet.java" as shown below

```
package com.xyz.client;

public class Pet {
    public static void main(String[ ] args) {
         System.out.println("I am found in the cl
    }
}

```

Step 2: The source code can be compiled into byte code i.e. Pet.class file as shown below:

```
C:\projects\Test\src>javac -d ../bin com/xyz/clie
```

As a Java Architect

Java architecture & design concepts interview Q&As with diagrams | What should be a typical Java EE architecture?

Senior Java developers must have a good handle on

Note: The compiled byte code file Pet.class will be saved in the folder C:\projects\Test\bin\com\xyz\client.

Step 3: If you run it inside where the Pet.class is stored, the answer is yes.

```
C:\projects\Test\bin>java com/xyz/client/Pet
```

The Pet.class file will be found since com.xyz.client.Pet class file is in the projects\Test\bin folder. If you run it in any other folder say c:\

```
C:\> java com.xyz.client.Pet
```

The answer is no, and you will get the following exception: Exception in thread "main" **java.lang.NoClassDefFoundError:** com/xyz/client/Pet. To fix this, you need to tell how to find the Pet.class by setting or providing the **classpath**. How can you do that? One of the following ways:

1) Set the operating system CLASSPATH environment variable to have the project folder c:\projects\Test\bin.

```
CLASSPATH=C:/projects/Test2/bin
```

2) Set the operating system CLASSPATH environment variable to have a jar file c:/projects/Test/pet.jar, which has the Pet.class file in it

```
CLASSPATH=c:/projects/Test/pet.jar
```

3) Run it with -cp or -classpath option as shown below.

```
C:\>java -cp projects\Test\bin com/xyz/client/Pet
C:\>java -classpath c:/projects/Test/pet.jar    c
C:\projects\Test\src>java -cp ../bin com/xyz/clie

```

open all | close all

- Best Practice (6)
- Coding (26)
- Concurrency (6)
- Design Concepts (7)
- Design Patterns (11)
- Exception Handling (3
- Java Debugging (21)
- Judging Experience I
- Low Latency (7)
- Memory Managemen
- Performance (13)
- QoS (8)
- Scalability (4)
- SDLC (6)
- Security (13)
- Transaction Managen

80+ step by step Java Tutorials

open all | close all

- Setting up Tutorial (6)
- Tutorial - Diagnosis (2
- Akka Tutorial (9)
- Core Java Tutorials (2
- Hadoop & Spark Tuto
- JEE Tutorials (19)
- Scala Tutorials (1)
- Spring & HIbernate Tu
- Tools Tutorials (19)
- Other Tutorials (45)

Preparing for Java written & coding tests

You may also like

12 Java String class interview Q&A | Web Services interview Q&A | Java EE Overview interview Q&A | Multithreading scenarios in Java applications interview Q&A

Popular Posts

- ♦ 11 Spring boot interview questions & answers
  852 views
- ♦ Q11-Q23: Top 50+ Core on Java OOP Interview Questions & Answers
  825 views
- 18 Java scenarios based interview Questions and Answers
  447 views
- 001A: ♦ 7+ Java integration styles & patterns interview questions & answers
  400 views
- ♦ 7 Java debugging interview questions & answers
  311 views
- ♦ 10 ERD (Entity-Relationship Diagrams) Interview Questions and Answers
  301 views
- 01b: ♦ 13 Spring basics Q8 – Q13 interview questions & answers
  292 views
- 01: ♦ 15 Ice breaker questions asked 90% of the time in Java job interviews with hints
  286 views
- ♦ Q24-Q36: Top 50+ Core on Java classes, interfaces and generics interview questions & answers
  263 views
- 8 Git Source control system interview questions & answers
  215 views

Bio | **Latest Posts**

Arulkumaran Kumaraswamipillai

open all | close all
- ♦ Complete the given
- Can you write code? (
- Converting from A to I
- Designing your classe
- Java Data Structures
- Passing the unit tests
- What is wrong with th
- Writing Code Home A
- Written Test Core Jav
- Written Test JEE (1)

How good are your...to go places?

open all | close all
- Career Making Know-
- Job Hunting & Resum

Mechanical Eng to freelance Java developer in 3 yrs. Contracting since 2003, and attended 150+ Java job interviews, and often got 4 - 7 job offers to choose from. It pays to prepare. So, published Java interview Q&A books via Amazon.com in 2005, and sold 35,000+ copies. Books are outdated and replaced with this subscription based site.

About Arulkumaran Kumaraswamipillai

Mechanical Eng to freelance Java developer in 3 yrs. Contracting since 2003, and attended 150+ Java job interviews, and often got 4 - 7 job offers to choose from. It pays to prepare. So, published Java interview Q&A books via Amazon.com in 2005, and sold 35,000+ copies. Books are outdated and replaced with this subscription based site.

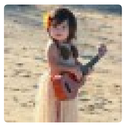
‹ ♦♥ Object equals Vs == and pass by reference Vs value

♦ 15 Database design interview Questions & Answers ›

Posted in Java Overview

Tags: Free Content, Free FAQs, Java/JEE FAQs, Novice FAQs

One comment on "01: ♦ ♥ 17 Java overview interview questions and answers"



ASIT says:

June 16, 2016 at 10:29 pm

Very well written answers. It is helpful while interview preparation for experienced one and a programming beginner too.

Reply

Leave a Reply

Logged in as geethika. Log out?

Comment

Post Comment

Empowers you to open more doors, and fast-track

Technical Know Hows

☀ Java generics in no time ☀ Top 6 tips to transforming your thinking from OOP to FP ☀ How does a HashMap internally work? What is a hashing function?
☀ 10+ Java String class interview Q&As ☀ Java auto un/boxing benefits & caveats ☀ Top 11 slacknesses that can come back and bite you as an experienced Java developer or architect

Non-Technical Know Hows

☀ 6 Aspects that can motivate you to fast-track your career & go places ☀ Are you reinventing yourself as a Java developer? ☀ 8 tips to safeguard your Java career against offshoring ☀ My top 5 career mistakes

Prepare to succeed

☀ Turn readers of your Java CV go from "Blah blah" to "Wow"? ☀ How to prepare for Java job interviews? ☀ 16 Technical Key Areas ☀ How to choose from multiple Java job offers?

Select Category ▼