

# Java-Success.com

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


[Home](#)
[Java FAQs](#)
[600+ Java Q&As](#)
[Career](#)
[Tutorials](#)
[Member](#)
[Why?](#)
[Can u Debug?](#)
[Java 8 ready?](#)
[Top X](#)
[Productivity Tools](#)
[Judging Experience?](#)

[Home](#) > [member-paid](#) > ♦ 5 Java Object class methods interview questions & answers

## ♦ 5 Java Object class methods interview questions & answers

Posted on [August 16, 2014](#) by [Arulkumaran Kumaraswamipillai](#) — No

[Comments](#) ↓

0

Like

0

Share

G+1

Share

**Q1.** What are the non-final methods in Java Object class, which are meant primarily for extension?

**A1.** The non-final methods are

`equals( )`, `hashCode( )`, `toString( )`, `clone( )`, and `finalize( )`.

These methods are meant to be **overridden**. The `equals( )` and `hashCode( )` methods prove to be very important, when objects implementing these two methods are added to collections. If implemented incorrectly or not implemented at

[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**

[open all](#) | [close all](#)

☰ [Ice Breaker Interview](#)

- 01: ♦ 15 Ice breakers
- 02: ♥♦ 8 real life examples
- 03: ♦10+ Know your interviewer
- 04: Can you think of 10 questions to ask your interviewer?
- 05: ♥ What job interview tips should I follow?
- 06: ► Tell me about your experience
- 07: ♥ 20+ Pre interview questions

☰ [Core Java Interview Questions](#)

☰ [Java Overview \(4\)](#)

01: ♦ ♥ 17 Java Interview Questions

all, then your objects stored in a Map may behave strangely and also it is hard to debug.

The other methods like `wait( )`, `notify( )`, `notifyAll( )`, `getClass( )`, etc are final methods and therefore cannot be overridden. The methods `clone( )` and `finalize( )` have protected access.

**Q2.** Are these methods easy to implement correctly?

**A2.** No. It is not easy to implement these methods correctly because,

- 1) You must pay attention to whether your implementation of these methods will continue to work correctly if sub classed. If your class is not meant for extension, then declare your class as final.
- 2) These methods have to adhere to contracts, and when implementing or overriding a method, the contracts must be satisfied.

## Equals() Vs hashCode()

**Q3.** What are the implications of implementing them incorrectly?

**A3.** In short, excess debug and rework time. Incorrect implementations of `equals( )` and `hashCode( )` methods can result in data being lost in HashMaps and Sets. You can also get intermittent data related problems that are harder to consistently reproduce over time.

Here is an example of `equals()` and `hashCode()` methods being invoked implicitly in adding and retrieving objects from a *Map*:

02: ♥♦ Java Con

03: ♦ 9 Core Jav

04: ♦ Top 10 mos

☐ Data types (6)

01: Java data ty

02: ♥♦ 10 Java S

03: ♦ ♥ Java aut

04: Understandir

05: Java primitiv

Working with Da

☐ constructors-methc

Java initializers,

☐ Reserved Key Wor

♥♦ 6 Java Modifi

Java identifiers

☐ Classes (3)

♦ Java abstract c

♦ Java class loac

♦ Java classes a

☐ Objects (8)

► Beginner Jav

♥♦ HashMap & H

♦ 5 Java Object i

♦ Java enum inte

♦ Java immutabl

♥♦ Object equals

Java serialization

Mocks, stubs, dc

☐ OOP (10)

♥ Design princip

♦ 30+ FAQ Java

♦ Why favor cor

08: ♦ Write code

Explain abstracti

How to create a

Top 5 OOPs tips

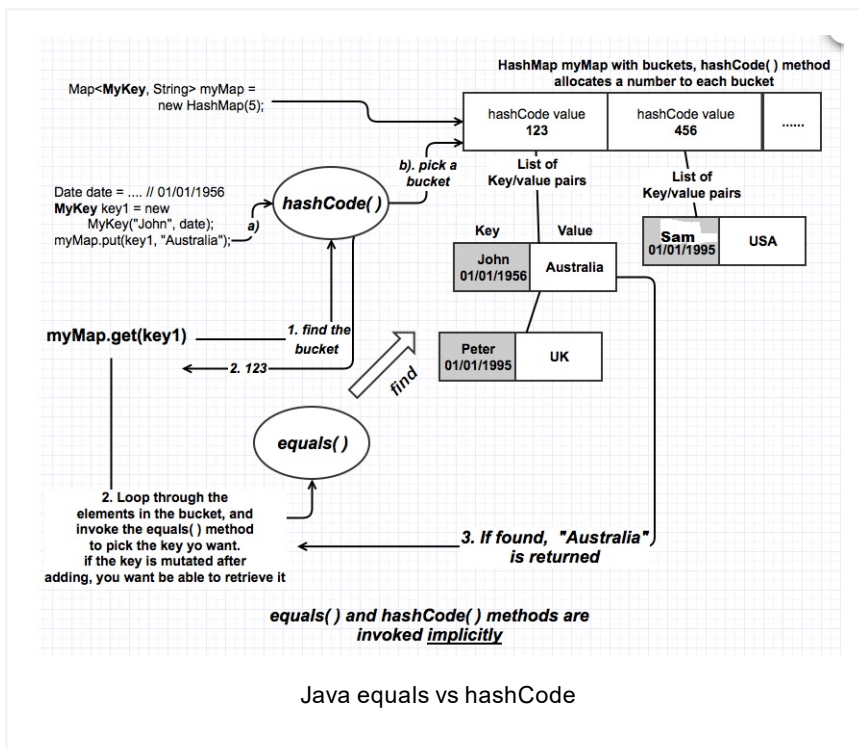
Top 6 tips to go a

Understanding C

What are good r

☐ GC (2)

♦ Java Garbage



The above example uses a custom key class **MyKey** that takes “name” and “date” as attributes. So, this key class needs to implement **equals()** and **hashCode()** methods using these 2 attributes.

**a & b)** When you put an object into a map with a key and a value, **hashCode()** method is implicitly invoked, and hash code value say 123 is returned. Two different keys can return the same hash value. A good hashing algorithm spreads out the numbers. In the above example, let’s assume that (“John”,01/01/1956) key and (“Peter”, 01/01/1995) key return the same hash value of **123**.

**Q.** So, when we retrieve the object with a specific key, how does it know which one of two objects to return?

**A.** This is where the **equals()** method is implicitly invoked to get the object value with the right key. So, it needs to loop through each key/value pair in the bucket “123” to find the right key by comparing “name” & “date” via equals() method.

**Q.** What if you change the **hashCode()** method of the “MyKey” class to always return a constant value of? Will you be able to add objects to the map?

**A.** Yes, you will be able to add objects, but all the key/value

03: Java GC tun

Generics (5)

♥ Java Generics

♥ Overloaded m

♦ 12 Java Gener

♦ 7 rules to reme

3 scenarios to ge

FP (8)

01: ♦ 19 Java 8 I

02: ♦ Java 8 Stre

03: ♦ Functional

04: ♥♦ Top 6 tips

05: ♥ 7 Java FP

Fibonacci numb

Java 8 String str

Java 8: What is c

IO (7)

♥ Reading a text

♦ 15 Java old I/C

06: ♥ Java 8 way

Processing large

Processing large

Read a text file f

Reloading config

Multithreading (12)

01: ♥♦ 15 Beginr

02: ♥♦ 10+ Java

03: ♦ More Java

04: ♦ 6 popular J

05: ♦ How a thre

06: ♦ 10+ Atomic

07: 5 Basic multi

08: ♦ ThreadLoc

09: Java FutureT

10: ♦ ExecutorSe

Java ExecutorSe

Producer and Co

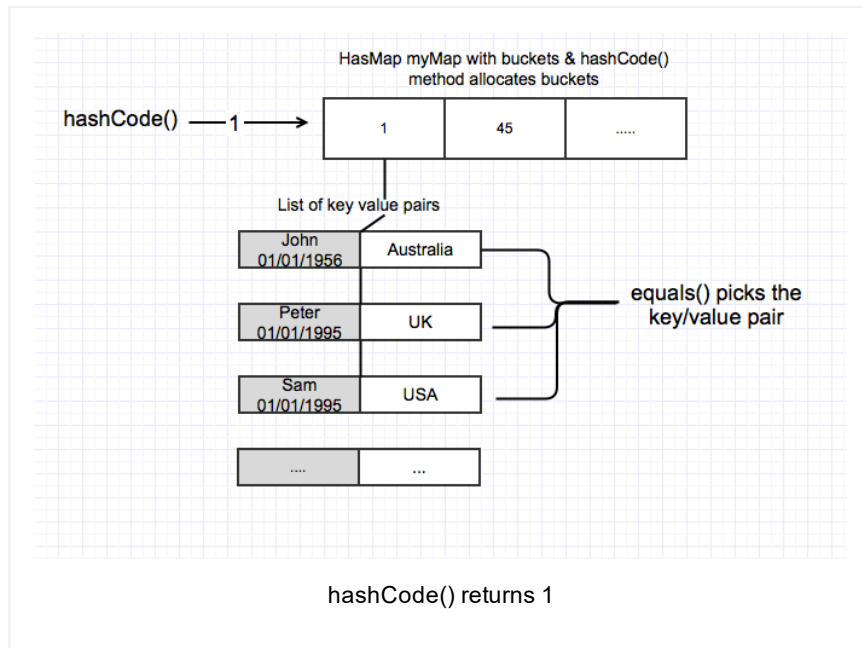
Algorithms (5)

♦ Splitting input t

♦ Tree traversal

♥ ♦ Java coding

pairs will be added to the single bucket, and the equals() method needs to loop through all the objects in the same bucket.



**1 & 2).** The hashCode() method picks the right bucket **123**. Then loop through all the values and invoke the equals() method to pick the right key, which is ("John",01/01/1956).

**Q4.** If a class overrides the equals( ) method, what other method it should override? What are some of the key considerations while implementing these methods?

**A4.** It should override the **hashCode( ) method**. The contract between the hashCode( ) and equals( ) methods is clearly defined on the Java API for the Object class under respective methods. Here are some key points to keep in mind while implementing these methods,

- 1) The equals( ) and hashCode( ) methods should be implemented together. You should not have one without the other.
- 2) If two objects return the same hashCode( ) integer value does not mean that those two objects are equal.
- 3) If two objects are equal, then they must return the same hashCode( ) integer value.
- 4) The implementation of the equals( ) method must be consistent with the hashCode( ) method to meet the previous

[Searching algorithm](#)

[Swapping, partitioning](#)

[Annotations \(2\)](#)

[8 Java Annotations](#)

[More Java annotations](#)

[Collection and Data Structures](#)

[Find the first non-repeating character in a string](#)

[Java Collections Framework](#)

[Java Iterable Interface](#)

[HashMap & HashSet](#)

[Sorting objects](#)

[Java 8 Stream API](#)

[Understanding Java 8 Streams](#)

[Java Collections Framework](#)

[If Java did not have Collections](#)

[Java 8: Different ways to iterate over a collection](#)

[Part-3: Java Tree Traversal](#)

[Sorting a Map by value](#)

[When to use which collection](#)

[Differences Between Java Collections](#)

[Java Iterable Interface](#)

[Multithreading](#)

[Why do Proxy, ProxyHandler, ProxyGenerator](#)

[Core Java Modifications](#)

[Differences between Java Collections and Java 8 Streams](#)

[Java Collections Framework](#)

[Event Driven Programming](#)

[Event Driven Programming](#)

[Event Driven Programming](#)

[Exceptions \(2\)](#)

[Java exception handling](#)

[Top 5 Core Java Interview Questions](#)

[Java 7 \(2\)](#)

[Java 7 fork and join](#)

[Java 7: Top 8 new features](#)

[Java 8 \(24\)](#)

[01: 19 Java 8 Interview Questions](#)

[02: Java 8 Stream API](#)

[03: Functional Programming](#)

[04: Top 6 tips for Java 8](#)

[04: Convert Lists to Arrays](#)

bullet points.

5) Use `@Override` annotation to ensure that the methods are correctly overridden.

6) Favor `instanceof` instead of `getClass(..)` in the `equals()` method which takes care of super types and null comparison as recommended by Joshua Bloch, but ensure that the `equals` implementation is final to preserve the symmetry contract of the method: `x.equals(y) == y.equals(x)`. If final seems restrictive, carefully examine to see if overriding implementations can fully maintain the contract established by the `Object` class.

7) Check for self-comparison and null values where required.

```

1 public final class Pet {
2     int id;
3     String name;
4
5     /**
6      * use @Override annotation to prevent the d
7      * method name or incorrect method signature
8      */
9
10    @Override
11    public boolean equals(Object that){
12        //check for self-comparison
13        if ( this == that ) return true;
14
15        /**
16         * use instanceof instead of getClass here
17         * 1. it can match any super type, and n
18         * 2. explicit check for "that == null"
19         * "null instanceof Pet" always retu
20         */
21        if ( ! (that instanceof Pet) )
22            return false;
23
24        Pet pet = (Pet)that;
25        return this.id == pet.id && this != nul
26    }
27
28    /**
29     * fields id & name are used in both equals(
30     * hashCode() methods.
31     */
32
33
34    @Override
35    public int hashCode( ) {
36        int hash = 9;
37        hash = (31 * hash) + id;
38        hash = (31 * hash) + (null == name ? 0 :
39        return hash;
40    }
41 }
42

```

04: Understanding

05: ♥ 7 Java FP

05: ♦ Finding the

06: ♥ Java 8 way

07: ♦ Java 8 API

08: ♦ Write code

10: ♦ ExecutorSe

Fibonacci numbe

Java 8 String str

Java 8 using the

Java 8: 7 useful

Java 8: Different

Java 8: Does "O

Java 8: What is c

Learning to write

Non-trivial Java 8

Top 6 Java 8 fea

Top 8 Java 8 fea

Understanding J

☰ JVM (6)

♦ Java Garbage

01: jvisualvm to

02: jvisualvm to

05: Java primitiv

06: ♦ 10+ Atomic

5 JMX and MBe

☰ Reactive Program

07: Reactive Pro

10: ♦ ExecutorSe

3. Multi-Threadir

☰ Swing & AWT (2)

5 Swing & AWT i

Q6 – Q11 Swing

☰ JEE Interview Q&A (3

☰ JEE Overview (2)

♦ 8 Java EE (aka

Java EE interview

☰ Web basics (8)

01: ♦ 12 Web ba

02: HTTP basics

03: Servlet inter



Use Apache's `HashCodeBuilder` & `EqualsBuilder` classes to simplify your implementation, especially when you have a large number of member variables. Commonclipse is an eclipse plugin for jakarta commons-lang users. It is very handy for automatic generation of `toString()`, `hashCode()`, `equals(..)`, and `compareTo()` methods.

```

1 public final class Pet2 {
2     int id;
3     String name;
4
5     @Override
6     public boolean equals(Object that) {
7         if (this == that)
8             return true;
9
10        if (!(that instanceof Pet2))
11            return false;
12
13        Pet2 pet = (Pet2) that;
14        return new EqualsBuilder().append(this.
15            this.name, pet.name).isEquals( )
16    }
17
18    /**
19     * both fields id & name are used in equals(
20     * so both fields must be used in hashCode(
21     */
22
23    @Override
24    public int hashCode( ) {
25        //pick 2 hard-coded, odd & >0 int value
26        return new HashCodeBuilder(1, 31).append
27            this.name).toHashCode( );
28    }
29 }
30

```

## toString() method

**Q5.** Why or when should you override a `toString()` method?

**A5.** You can use `System.out.println()` or `logger.info(...)` to print any object. The `toString()` method of an object gets invoked automatically, when an object reference is passed in the `System.out.println(refPet)` or `logger.info(refPet)` method. However for good results, your class should have a `toString()` method that overrides `Object` class's default implementation by formatting the object's data in a sensible way and returning a `String`. Otherwise all that's printed is the class name followed by an "@" sign and then unsigned hexadecimal representation of the `hashCode`. For example, If

[04: JSP overview](#)

[05: Web patterns](#)

[06: ♦ MVC0, MV](#)

[07: When to use](#)

[08: Web.xml inte](#)

[WebService \(11\)](#)

[01: ♥♦ 40+ Java](#)

[02: ♦ 6 Java RE](#)

[03: ♥ JAX-RS hc](#)

[04: 5 JAXB inter](#)

[05: RESTful We](#)

[06: RESTful Wel](#)

[07: HATEOAS R](#)

[08: REST constr](#)

[09: 11 SOAP We](#)

[10: SOAP Web](#)

[11: ♥ JAX-WS hc](#)

[JPA \(2\)](#)

[10: Spring, Java](#)

[8 JPA interview c](#)

[JTA \(1\)](#)

[JTA interview Q&](#)

[JDBC \(4\)](#)

[♦ 12 FAQ JDBC](#)

[JDBC Overview](#)

[NamedParamete](#)

[Spring, JavaCon](#)

[JMS \(5\)](#)

[♦ 16 FAQ JMS ir](#)

[Configuring JMS](#)

[JMS versus AMC](#)

[Spring JMS with](#)

[Spring JMS with](#)

[JMX \(3\)](#)

[5 JMX and MBe](#)

[Event Driven Pr](#)

[Yammer metrics](#)

[JNDI and LDAP \(1\)](#)

[JNDI and LDAP](#)

[Pressed for time? Jav](#)

[Job Interview Ice B](#)

the Pet class doesn't override the toString( ) method as shown below, by default Pet@162b91 will be printed via toString( ) default implementation in the Object class.

**Q6.** Can you override clone( ) and finalize( ) methods in the Object class? How do you disable a clone( ) method?

**A6.** es, but you need to do it very judiciously. Implementing a properly functioning clone( ) method is complex and it is rarely necessary. You are better off providing some alternative means of object copying through a copy constructor or a static factory method.

Unlike C++ destructors, the finalize( ) method in Java is unpredictable, often dangerous and generally unnecessary. Use finally{} blocks to close any resources or free memory. The finalize( ) method should only be used in rare instances as a safety net or to terminate noncritical native resources. If you do happen to call the finalize( ) method in some rare instances, then remember to follow the following guidelines:

1) You should call the finalize method of the super class in case it has to clean up.

```

1  protected void finalize( ) throws Throwable {
2      try{
3          //finalize subclass state
4      }
5      catch(Throwable t){
6          //log the exception
7      }
8      finally {
9          super.finalize( );
10     }
11 }
12

```

2) You should not depend on the finalize method being called. There is no guarantee that when (or if) objects will be garbage collected and thus no guarantee that the finalize method will be called before the running program terminates.

3) Finally, the code in finalize method might fail and throw exceptions. Catch these exceptions so that the finalize method can continue.

01: ♦ 15 Ice bre

02: ♥♦ 8 real life

03: ♦10+ Know y

FAQ Core Java J

♥♦ Q1-Q10: Top

♦ Q11-Q23: Top

♦ Q24-Q36: Top

♦ Q37-Q42: Top

♦ Q43-Q54: Top

01: ♥♦ 15 Beginr

02: ♥♦ 10+ Java

FAQ JEE Job Inter

♦ 12 FAQ JDBC

♦ 16 FAQ JMS ir

♦ 8 Java EE (aka

♦ Q01-Q28: Top

♦ Q29-Q53: Top

01: ♦ 12 Web ba

06: ♦ MVC0, MV

JavaScript mista

JavaScript Vs Ja

JNDI and LDAP

JSF interview Q

JSON interview

FAQ Java Web Ser

01: ♥♦ 40+ Java

02: ♦ 6 Java RE

05: RESTFul We

06: RESTful Wel

09: 11 SOAP We

Java Application Ar

001A: ♦ 7+ Java

001B: ♦ Java arc

04: ♦ How to go

Hibernate Job Inter

01: ♥♦ 15+ Hiber

01b: ♦ 15+ Hiber

06: Hibernate Fil

8 JPA interview c

Spring Job Interview

♦ 11 Spring boot

## How do you disable a clone( ) method?

```
1 public final Object clone( ) throws CloneNotSuppo
2     throw new CloneNotSupportedException( );
3 }
```

## You may also like

**a) wait() & notify()** methods are covered with examples: 6 popular Java multi-threading interview questions and answers with diagrams and code.

## Popular Posts

♦ 11 Spring boot interview questions & answers

857 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

401 views

♦ 7 Java debugging interview questions & answers

311 views

♦ 10 ERD (Entity-Relationship Diagrams) Interview Questions and Answers

302 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

01: ♥♦ 13 Spring

01b: ♦ 13 Spring

04 ♦ 17 Spring b

05: ♦ 9 Spring B

Java Key Area Ess

♦ Design pattern

♥ Top 10 causes

♥♦ 01: 30+ Writir

♦ 12 Java desigr

♦ 18 Agile Develo

♦ 5 Ways to debi

♦ 9 Java Transac

♦ Monitoring/Pro

02: ♥♦ 13 Tips to

15 Security key :

4 FAQ Performa

4 JEE Design Pa

5 Java Concurr

6 Scaling your J

8 Java memory i

OOP & FP Essenti

♦ 30+ FAQ Java

01: ♦ 19 Java 8 I

04: ♥♦ Top 6 tips

Code Quality Job I

♦ Ensuring code

♦ 5 Java unit tes

SQL, XML, UML, JSC

ERD (1)

♦ 10 ERD (Entity

NoSQL (2)

♦ 9 Java Transac

3. Understanding

Regex (2)

♥♦ Regular Expr

Regular Express

SQL (7)

♦ 15 Database d

♦ 14+ SQL interv

♦ 9 SQL scenari

Auditing databas



215 views

Bio

Latest Posts



## 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](http://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](http://Amazon.com) in 2005, and sold 35,000+ copies. Books are outdated and replaced with this subscription based site.

◀ ♦ Java immutable objects interview questions & answers

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

Posted in member-paid, Objects

Tags: Core Java FAQs, Java/JEE FAQs, Novice FAQs

## Leave a Reply

Logged in as geethika. Log out?

Comment

Deleting records

SQL Subquery ir

Transaction man

UML (1)

♦ 12 UML intervi

JSON (2)

JSON interview (

JSON, Jackson,

XML (2)

XML basics inter

XML Processing

XSD (2)

11 FAQ XSD inte

XSD reuse inter

YAML (2)

YAML with Java

YAML with Sprin

Hadoop & BigData Int

♥ 01: Q1 – Q6 Had

02: Q7 – Q15 Hado

03: Q16 – Q25 Hado

04: Q27 – Q36 Apa

05: Q37 – Q50 Apa

05: Q37-Q41 – Dat

06: Q51 – Q61 HB

07: Q62 – Q70 HDI

Java Architecture Inte

♥♦ 01: 30+ Writing

001A: ♦ 7+ Java int

001B: ♦ Java archi

01: ♥♦ 40+ Java W

02: ♥♦ 13 Tips to w

03: ♦ What should l

04: ♦ How to go ab

05: ETL architectur

1. Asynchronous pi

2. Asynchronous pi

Scala Interview Q&As

01: ♥ Q1 – Q6 Scal

02: Q6 – Q12 Scal

03: Q13 – Q18 Sca

Post Comment

04: Q19 – Q26 Sca
05: Q27 – Q32 Sca
06: Q33 – Q40 Sca
07: Q41 – Q48 Sca
08: Q49 – Q58 Sca
09: Q59 – Q65 Hig
10: Q66 – Q70 Pat
11: Q71 – Q77 – S
12: Q78 – Q80 Rec
Spring, Hibernate, & I
Spring (18)
Spring boot (4)
♦ 11 Spring bc
01: Simple Sp
02: Simple Sp
03: Spring box
Spring IO (1)
Spring IO tuto
Spring JavaConl
10: Spring, Ja
Spring, JavaC
Spring, JavaC
Spring, JavaC
01: ♥♦ 13 Spring
01b: ♦ 13 Spring
02: ► Spring DI
03: ♥♦ Spring DI
04 ♦ 17 Spring b
05: ♦ 9 Spring B
06: ♥ Debugging
07: Debugging S
Spring loading p
Hibernate (13)
01: ♥♦ 15+ Hiber
01b: ♦ 15+ Hiber
02: Understandir
03: Identifying ar
04: Identifying ar
05: Debugging H
06: Hibernate Fil
07: Hibernate mi

08: Hibernate au

09: Hibernate en

10: Spring, Java

11: Hibernate de

12: Hibernate cu

AngularJS (2)

♥ 8 AngularJS in

More Angular JS

Git & SVN (6)

♥ Git & Maven fc

♥ Merging Vs rel

♥ Understanding

6 more Git interv

8 Git Source cor

Setting up Cygw

JMeter (2)

♥ JMeter for test

♦ JMeter perform

JSF (2)

JSF interview Q&

More JSF intervi

Maven (3)

♥ Git & Maven fc

12 Maven intervi

7 More Maven ir

Testing & Profiling/Sa

Automation Testing

♥ Selenium and

Code Coverage (2)

Jacoco for unit te

Maven and Cobe

Code Quality (2)

♥ 30+ Java Code

♦ Ensuring code

jvisualvm profiling (

01: jvisualvm to :

02: jvisualvm to :

03: jvisualvm to :

Performance Testir

♥ JMeter for test

♦ JMeter perform

## Unit Testing Q&A (2)

### BDD Testing (4)

Java BDD (Be

jBehave and E

jBehave and j

jBehave with t

### Data Access Uni

♥ Unit Testing

Part #3: JPA H

Unit Test Hibe

Unit Test Hibe

### JUnit Mockito Sp

#### JUnit Mockito

Spring Con

Unit Testing

Part 1: Unit te

Part 2: Mockit

Part 3: Mockit

Part 4: Mockit

Part 5: Mockit

### Testing Spring T

Integration Un

Unit testing Sp

♦ 5 Java unit tes

JUnit with Hamc

Spring Boot in ui

## Other Interview Q&A 1

### Finance Domain In

12+ FX or Forex

15 Banking & fin

### FIX Interview Q&A

20+ FIX basics in

Finding your way

### Groovy Interview Q

#### Groovy Coding C

Cash balance

Sum grades C

♥ Q1 – Q5 Groov

♦ 20 Groovy clos

♦ 9 Groovy meta

Groovy method c

- Q6 – Q10 Groov
- JavaScript Interview
- JavaScript Top I
- ♥ Q1 – Q10 J
- ♦ Q11 – Q20
- ♦ Q21 – Q30
- ♦ Q31 – Q37
- JavaScript mis
- JavaScript Vs Ja
- JavaScript Vs
- Unix Interview Q&A
- ♥ 14 Unix intervi
- ♥ Hidden Unix, C
- sed and awk to v
- Shell script inter
- Unix history com
- Unix remoting in
- Unix Sed comm
- Free Java Interview
- Java Integration
- Java Beginner I
- 02: Spring DIP, I
- 06: Tell me abou

## 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**



[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 \(1\)](#)
- [Low Latency \(7\)](#)
- [Memory Management \(1\)](#)
- [Performance \(13\)](#)
- [QoS \(8\)](#)
- [Scalability \(4\)](#)
- [SDLC \(6\)](#)
- [Security \(13\)](#)
- [Transaction Management \(1\)](#)

## 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 Tutorial \(1\)](#)
- [JEE Tutorials \(19\)](#)
- [Scala Tutorials \(1\)](#)
- [Spring & Hibernate Tutorial \(1\)](#)
- [Tools Tutorials \(19\)](#)
- [Other Tutorials \(45\)](#)

## Preparing for Java written & coding tests

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

## 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 ▼

## © Disclaimer

The contents in this Java-Success are copy righted. The author has the right to correct or enhance the current content without any prior notice.

These are general advice only, and one needs to take his/her own circumstances into consideration. The author will not be held liable for any damages caused or alleged to be caused either directly or indirectly by these materials and resources. Any trademarked names or labels used in this blog remain the property of their respective trademark owners. No guarantees are made regarding the accuracy or usefulness of content, though I do make an effort to be accurate. Links to external sites do not imply endorsement of the linked-to sites.