

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](#) › What are good real life Java inheritance (or use of abstract classes) examples ?

What are good real life Java inheritance (or use of abstract classes) examples ?

Posted on [June 9, 2015](#) by [Arulkumaran Kumaraswamipillai](#)

We have already looked at [Why favor composition over inheritance?](#) with reasoning and code examples. Inheritance still has its use and let's look at some real life examples.

#1. Unit Test Classes

```
1 import javax.inject.Inject;
2
3 import org.junit.Assert;
4 import org.junit.Test;
5
6 public class EmailAddressDaoTest extends BaseTes
7
```

[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](#)

[Core Java Interview C](#)

[Java Overview \(4\)](#)

[Data types \(6\)](#)

[constructors-methc](#)

[Reserved Key Wor](#)

[Classes \(3\)](#)

[Objects \(8\)](#)

[OOP \(10\)](#)

[♥ Design princip](#)

[♦ 30+ FAQ Java](#)

```

8      @Inject
9      private EmailAddressDao emailAddressDao;
10
11     @Test
12     public void testGetEmailAddressForUserId() {
13         String emailAddress = emailAddressDao.ge
14         Assert.assertEquals("hleibowitz@stateone
15     }
16 }

```

In the base class you can wire up Spring configs, shared logic, etc. For example, you may want to load SQL scripts to test your DAO layer with a HSQLDB embedded database. So, the scripts will get loaded only once into the in memory database and then multiple unit tests can be executed. This is demonstrated in [JPA tutorial with unit testing](#).

```

1  import static org.junit.Assert.fail;
2
3  import java.lang.reflect.Field;
4  import java.lang.reflect.Modifier;
5
6  import org.junit.runner.RunWith;
7  import org.springframework.test.context.ContextC
8  import org.springframework.test.context.junit4.S
9  import org.springframework.test.context.transact
10 import org.springframework.transaction.annotatio
11
12 /**
13  * Base class for all unit tests used in this mo
14  */
15 @Transactional(readOnly = false)
16 @TransactionConfiguration(defaultRollback = fals
17 @ContextConfiguration(classes = {
18     DomainTestConfiguration.class, MyAppServiceT
19     initializers = MyAppApplicationContextTestIn
20 })
21 @RunWith(SpringJUnit4ClassRunner.class)
22 public abstract class BaseTest {
23
24     //any common logic shared all test cases go
25 }

```

#2. Hibernate Entity Base classes wiring IDs, Version, and Auditing Fields

```

1  @MappedSuperclass
2  public abstract class AbstractDomainEntity<ID ex
3      private static final long serialVersionUID =
4
5      //for pessimistic concurrency control
6      @Version
7      @Column(name = "lock_version", nullable = fa
8      private Long version;

```

[Why favor com](#)
[08: Why code](#)
[Explain abstracti](#)
[How to create a](#)
[Top 5 OOPs tips](#)
[Top 6 tips to go](#)
[Understanding C](#)
[What are good r](#)

[GC \(2\)](#)
[Generics \(5\)](#)
[FP \(8\)](#)
[IO \(7\)](#)
[Multithreading \(12\)](#)
[Algorithms \(5\)](#)
[Annotations \(2\)](#)
[Collection and Data](#)
[Differences Between](#)
[Event Driven Progr](#)
[Exceptions \(2\)](#)
[Java 7 \(2\)](#)
[Java 8 \(24\)](#)
[JVM \(6\)](#)
[Reactive Programn](#)
[Swing & AWT \(2\)](#)
[JEE Interview Q&A \(3](#)
[Pressed for time? Jav](#)
[SQL, XML, UML, JSC](#)
[Hadoop & BigData Int](#)
[Java Architecture Inte](#)
[Scala Interview Q&As](#)
[Spring, Hibernate, & I](#)
[Testing & Profiling/Sa](#)
[Other Interview Q&A 1](#)
[Free Java Interview](#)

As a Java Architect

[Java architecture &](#)
[design concepts](#)

```

9
10     @Override
11     public abstract ID getId();
12
13     //setters, getts, equals/hashcode/toString m
14
15 }

```

[interview Q&As with diagrams](#) | [What should be a typical Java EE architecture?](#)

Mapping for the database audit fields

```

1  @MappedSuperclass
2  @EntityListeners({DomainEntityListener.class})
3  public abstract class AbstractAuditableDomainEnt
4      extends AbstractDomainEntity<ID> {
5
6      private static final long serialVersionUID =
7
8      @Column(name = "created_by", nullable = true
9      private String createdBy;
10
11      @Column(name = "created_date", nullable = fa
12      @Type(type = "org.joda.time.contrib.hibernat
13      private DateTime createdDate;
14
15      @Column(name = "updated_by", nullable = true
16      private String lastModifiedBy;
17
18      @Column(name = "updated_date", nullable = fa
19      @Type(type = "org.joda.time.contrib.hibernat
20      private DateTime lastModifiedDate;
21
22      //setters, getts, equals/hashcode/toString m
23 }

```

The other entity classes can extend AbstractAuditableDomainEntity class.

```

1  @Table(name = "account_tbl")
2  @Entity(name = "Account")
3  public class Account extends AbstractAuditableDom
4
5      //...
6  }

```

#3. Template method and Composite design Patterns

are good examples of using an abstract base class and inheritance. Template and Composite design patterns are covered in [Java abstract classes Vs interfaces. Q11](#) discusses the template method design pattern — [12 java design patterns interview questions and answers](#)

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 Tuto](#)

In Java API, `AbstractStringBuilder`, `AbstractQueue`, `AbstractAction`, `AbstractButton`, `AbstractSet`, `AbstractMap`, etc provide a skeletal implementation to minimize the effort required to implement the concrete classes.

In Java 8, you can define default methods in an interface. Would this make abstract classes not required any more?

With the advent of default methods in Java 8 functional interfaces, the question arises, which one (i.e. abstract class or interfaces with default methods?) to use in a given situation. Abstract class or Interfaces with default methods. The abstract classes still have their place as

- 1) Default methods are never final.
- 2) Default methods can not be synchronized.
- 3) default methods can not override Object class's methods.
- 4) Interfaces where default methods are defined cannot hold state. That is, you can't define variables in an interface. Only behavior is allowed in default methods.

Popular Posts

♦ 11 Spring boot interview questions & answers

861 views

♦ Q11-Q23: Top 50+ Core on Java OOP Interview Questions & Answers

829 views

18 Java scenarios based interview Questions and Answers

448 views

001A: ♦ 7+ Java integration styles & patterns interview questions & answers

407 views

♦ 7 Java debugging interview questions & answers

311 views

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

303 views

✚ JEE Tutorials (19)

✚ Scala Tutorials (1)

✚ Spring & Hibernate Tutorials (1)

✚ Tools Tutorials (19)

✚ Other Tutorials (45)

Preparing for Java written & coding tests

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

✚ ♦ Complete the given

✚ Can you write code? (1)

✚ Converting from A to B

✚ Designing your classes

✚ Java Data Structures

✚ Passing the unit tests

✚ What is wrong with this

✚ Writing Code Home A

✚ Written Test Core Java

✚ Written Test JEE (1)

How good are you...to go places?

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

✚ Career Making Know-

✚ Job Hunting & Resum

01b: ♦ 13 Spring basics Q8 – Q13 interview questions & answers

294 views

01: ♦ 15 Ice breaker questions asked 90% of the time in Java job interviews with hints

288 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

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

[♥ Understanding Git terms origin, master, and head](#) >**Posted in** member-paid, OOP**Tags:** TopX

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?](#)

© 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.