# Singleon pattern ?How can your break ,then how can you resolve?

1. Swap two integers without using temp variable
2. Hibernate inheritance strategies
3. Spring core – bean lifecycle
4. In case of inheritance how throws class specifies exception names.if SQL exception is thrown by class A then child class can throw which exceptions?

[Exception handling in Method overriding with example (beginnersbook.com)](https://beginnersbook.com/2014/01/exception-handling-in-method-overriding-with-example/)

1. If a method can accept int(primitive type) and child class overridden method can accept Integer,other method can accept Long,another can accept int…Then, what is the precedence

Answer- <http://cs-fundamentals.com/java-programming/method-overloading-in-java.php>

<http://www.geeksforgeeks.org/method-overloading-autoboxing-widening-java>

first widening is preferred, then autoboxing and then varargs

Order of primitives in terms of widening

byte , short, int, long, float,  double

1. Java Method Overloading and Widening - JVM uses the method with the smallest argument that is wider than the parameter
2. Method called with fixed literals- Java by default treats integer literals int
3. Java Method Overloading and Autoboxing -If you are using wrapper class Object as an actual argument and compiler does not find the method with parameter(s) of the same reference type (i.e. class or interface type), then it starts searching a method with parameter(s) having the corresponding primitive data type.
4. Widening of primitive type gets more priority over var-args. –

**class** VarargsDemo

{

**public** **static** **void** main ([String](http://www.google.com/search?hl=en&q=allinurl%3Adocs.oracle.com+javase+docs+api+string)[] args)

{

**byte** b = 5;

aMethod(b, b);

}

**static** **void** aMethod (**byte**... b)

{

[System](http://www.google.com/search?hl=en&q=allinurl%3Adocs.oracle.com+javase+docs+api+system).out.println("byte, byte");

}

**static** **void** aMethod (**int** b, **int** c)

{

[System](http://www.google.com/search?hl=en&q=allinurl%3Adocs.oracle.com+javase+docs+api+system).out.println("int, int");

}

}

OUTPUT

======

**int**, **int**

1. compiler will choose widening over boxing- aMethod(Integer x) and aMethod(long x). Here, Integer is a wrapper reference type. If you call overloaded aMethod with an int argument,
2. Widening and boxing together-

**class** WideningAndBoxing

{

**public** **static** **void** main ([String](http://www.google.com/search?hl=en&q=allinurl%3Adocs.oracle.com+javase+docs+api+string)[] args)

{

**byte** b = 5;

aMethod(b);

}

**static** **void** aMethod ([Long](http://www.google.com/search?hl=en&q=allinurl%3Adocs.oracle.com+javase+docs+api+long) b)

{

[System](http://www.google.com/search?hl=en&q=allinurl%3Adocs.oracle.com+javase+docs+api+system).out.println("Long");

}

}

method aMethod in **class** WideningAndBoxing cannot be applied to given types;

aMethod(b);

^

required: [Long](http://www.google.com/search?hl=en&q=allinurl%3Adocs.oracle.com+javase+docs+api+long)

found: **byte**

reason: actual argument **byte** cannot be converted to [Long](http://www.google.com/search?hl=en&q=allinurl%3Adocs.oracle.com+javase+docs+api+long) by method invocation conversion

1 error

1. strangely boxing followed by widening is acceptable if this is passed to a reference of type Object.
2. **class** WideningAndBoxing
3. {
4. **public** **static** **void** main ([String](http://www.google.com/search?hl=en&q=allinurl%3Adocs.oracle.com+javase+docs+api+string)[] args)
5. {
6. **byte** b = 5;
8. *// b is first widened to Byte and then Byte passed to Object*
9. aMethod(b);
10. }
12. **static** **void** aMethod ([Object](http://www.google.com/search?hl=en&q=allinurl%3Adocs.oracle.com+javase+docs+api+object) b)
13. {
14. *//Object b is typecasted to Byte and then printed*
15. [Byte](http://www.google.com/search?hl=en&q=allinurl%3Adocs.oracle.com+javase+docs+api+byte) bb = ([Byte](http://www.google.com/search?hl=en&q=allinurl%3Adocs.oracle.com+javase+docs+api+byte))b;
16. [System](http://www.google.com/search?hl=en&q=allinurl%3Adocs.oracle.com+javase+docs+api+system).out.println("b: " + bb);
17. }
18. }
20. OUTPUT
21. ======
22. D:\JavaPrograms>javac WideningAndBoxing.java
24. D:\JavaPrograms>java WideningAndBoxing
25. b: 5
26. Add employee objects in set where age determines the equality criteria
27. From a customized list(not a java list), how to fetch last n elements.size of th

List is not known.

1. if there is circular dependency between the beans, then which injection should be used – setter or constructor?

DBS interview Questions

1. In hibernate if there are two methods one is insert (public method)and other is updating(private method).Public method is marked as @transactional (readonly=false),It calls from within itself the other method which private. if the update method which is private throws exception while updating in DB, will the whole transaction will be comiited or rolledback

<https://stackoverflow.com/questions/25076718/spring-propagation-examples-in-laymans-terms/25083505#25083505>

Consider this code...

class Service {

@Transactional(propagation=Propagation.REQUIRED)

public void doSomething() {

// access a database using a DAO

}

}

When doSomething() is called it knows it has to start a Transaction on the database before executing. If the caller of this method has already started a Transaction then this method will use that same physical Transaction on the current database connection.

When doSomething() is called it will start a new transaction if the caller has not already started a transaction.

If the caller of this method has already started a transaction then the callers' transaction is used and no new transaction is created (i.e. there is one transaction in play).

If an exception is thrown inside doSomething() then it will be rolled back, meaning that the caller will also see the transaction rolled back.

When doSomething() returns the transaction will not have been commited yet. It is the caller that will commit the transaction (or possibly rolled-back).

**PROPAGATION\_REQUIRES\_NEW**

class Service {

@Transactional(propagation=Propagation.REQUIRES\_NEW)

public void doSomething() {

// access a database using a DAO

}

}

When doSomething() is called it will always start a new transaction.

If the caller of this method has already started a transaction (TxnOuter) then the callers' transaction is suspended and a new transaction (TxnInner) is created (i.e. there are two transactions in play).

If an exception is thrown inside doSomething() then TxnInner will be rolled back, but the "suspended" transaction from the caller (TxnOuter) is unaffected.

When doSomething() returns without an Exception it will commit the transaction (TxnInner). The caller's transaction (TxnOuter) will be resumed and be unaware that another transaction was commited. The caller can then commit or roll-back TxnOuter as it sees fit.

The important point to note is that the Database views TxnOuter and TxnInner as completely independant transactions, and therefore two independant commits.

**PROPAGATION\_NESTED**

class Service {

@Transactional(propagation=Propagation.NESTED)

public void doSomething() {

// access a database using a DAO

}

}

NESTED can only be used if your JDBC driver and/or database supports [JDBC savepoints](https://stackoverflow.com/questions/7287201/java-jdbc-savepoint)

When doSomething() is called it will start a new transaction if the caller has not already started a transaction.

If the caller of this method has already started a transaction then the callers' transaction is used and no new transaction is created (i.e. there is one transaction in play). However a "savepoint" is marked on the transaction when doSomething() is entered.

If an Exception is thrown inside doSomething() then the transaction can be partially rolled back the transaction to the "savepoint". The caller will continue with the transaction.

When doSomething() returns without an Exception it is the caller who will commit the entire transaction (or roll back).

The important point to note is that the Database views only one transaction and there is only one commit.

<https://stackoverflow.com/questions/1614139/spring-transactional-read-only-propagation>

1. Webservices with security certificate, then how to call certificate or configure ?
2. Which version of JAXb is used with spring wstemplate while invoking web services
3. With business junits how did you use data source.

Answer: S*andboxing*, is dbUnit (http://dbunit.sourceforge.net). The dbUnit framework allows the developer to create a data set, which is automatically created into the real database before running the test code and can clean up its mess afterwards if necessary.

DbUnit has the ability to export and import your database data to and from XML datasets

Interfaces and classes in DBunit

|  |  |
| --- | --- |
| **Class** | **Description** |
| [IDatabaseConnection](http://dbunit.sourceforge.net/components.html#databaseconnection) | Interface representing a DbUnit connection to a database. |
| [IDataSet](http://dbunit.sourceforge.net/components.html#dataset) | Interface representing a collection of tables. |
| [DatabaseOperation](http://dbunit.sourceforge.net/components.html#databaseoperation) | Abstract class representing an operation performed on the database before and after each test. |

Example of loading data into DB from dataset file

***protected******void*** *loadInputData(****final*** *IDatabaseConnection conn,*

***final*** *String inputDataFileName)* ***throws*** *SQLException, IOException, DatabaseUnitException {*

*Resource logConfig =* ***new*** *ClassPathResource("log4j.xml");*

*DOMConfigurator.configure(logConfig.getURL());*

*Resource emptyTestInput =* ***new*** *ClassPathResource("testdata/com/annadaletech/productlink/dataaccess/EmptyTestInput.xml");*

*IDataSet emptyDataSet =* ***new*** *FlatXmlDataSet(emptyTestInput.getFile());*

*DatabaseOperation.CLEAN\_INSERT.execute(conn, emptyDataSet);*

*Resource testInput =* ***new*** *ClassPathResource(inputDataFileName);*

*IDataSet dataSet =* ***new*** *FlatXmlDataSet(testInput.getFile());*

*DatabaseOperation.CLEAN\_INSERT.execute(conn, dataSet);*

*}*

**protected** **void** populateDatabase(**final** DataSource dataSource, **final** String [] dbUnitXml) **throws** RuntimeException {

IDatabaseConnection con = **null**;

**try** {

con = **new** DatabaseConnection(dataSource.getConnection());

DatabaseConfig config = con.getConfig();

config.setProperty(DatabaseConfig.*PROPERTY\_DATATYPE\_FACTORY*, **new** CustomOracleDataTypeFactory());

String dbUnitPath = "testdata/com/annadaletech/productlink/dataaccess/";

**for** (**int** i=0; i < dbUnitXml.length; i++) {

loadInputData(con, dbUnitPath + dbUnitXml[i]);

}

con.getConnection().commit();

} **catch** (Exception e) {

e.printStackTrace(System.*out*);

**throw** **new** RuntimeException("exception loading input data", e);//NOPMD

} **finally** {

**try** {

con.getConnection().commit();

con.getConnection().close();

} **catch** (SQLException e) {

**throw** **new** RuntimeException("exception closing exception ", e);//NOPMD

}

}

}

Example usage of above method in any Dao–

String [] dbUnits = {"ActivityReportDAOImplTestDBUnit.xml"};

populateDatabase(dataSource,dbUnits);

datasource is injected into every dao in testapplicationcontext.xml

Our test file extends Spring f/w class- **AbstractTransactionalDataSourceSpringContextTests** which further extends **AbstractTransactionalSpringContextTests**

We need to override the method **getConfigLocations(belongs to AbstractSingleSpringContextTests**) to mention the applicationcontext file path.

**protected** String[] getConfigLocations() {

String[] locations = **new** String[1];

locations[0] = "/TestApplicationContext.xml";

**return** locations;

}

Also we need to override another method – **onSetUpInTransaction(belongs to AbstractTransactionalSpringContextTests)** which will populate database using XML dataset file before and after each test.

Other methods of AbstractTransactionalSpringContextTests are :

onSetUpBeforeTransaction()

onSetUpInTransaction

onTearDown

1. How datasource was configured in your application.
2. How to take the control from web.xml to different controller.(spring mvc)
3. Core java – Class A has child class B,override a method say meth(int I, int j), then output of below program?

A obj =new A();

B obj=new A();

A obj = new B();

B obj=new B();

Obj.method(1,2);

1. Analytical functions in Oracle.- rank(), dense\_rank()
2. How transaction management was done in spring
3. Concurrenthashmap vs synchronised map vs copyonwritearraylist?
4. CopyonArraylist is failsafe or failfast? Concurrenthashmap is fail safe or fail fast?Arraylist is fail safe or fail fast?

Written Test Questions

1. Abstract class can have constructor?output of some program where order of constructors to be determined.
2. Integer a=43,b=43,c=143,d=143;

SOP(a==b) ?? SOP(c=d)

1. Have try,catch and fincally block print something and ,if exception is thrown in catch block- finally will be executed? what will be the output?
2. Inner class constructors in derived class.. order of constructors called?
3. Which of these does not override hashcode and equals- Stringbuffer, Integer,Double,Character,String?
4. Enum constructors cannot be – private,public,protected or default?

The enum constructor must be either private or package scope (default). You cannot use public or protected constructors for a Java enum.

1. Write a program to print nth Fibonacci number
2. Query to print dept name,max salary of the employee dept wise,given 2 tables, employee and department.
3. Program to compress the string.e.g if the input is aabbbccaab then output should be a2b3c2a2b1
4. Output interface method – fflush,flush,clear??
5. Bufferedwriter,printwriter,filewriter- how are they wrapped
6. Assert statement usage. E.g b is boolean then assert b=false, a is integer assert a=false; and so on. Which of these statement will compile successfully.
7. Invoking static methd with null reference?
8. Calling thread first with run method and then with start method
9. Exception handling , when main method was called without arguments and trying to fetch args[0],catch finally were concatenating a common string.Secondly called main with some arguments
10. Arrays.assList – convert array to list and then change the content of array.Changing content of arraylist using set method and then adding element to arraylist.

Infy interview

1. stringutils notblank vs not empty

StringUtils.isBlank()

StringUtils.isBlank(null) = true

StringUtils.isBlank("") = true

StringUtils.isBlank(" ") = true

StringUtils.isBlank("bob") = false

StringUtils.isBlank(" bob ") = false

StringUtils.isEmpty

StringUtils.isEmpty(null) = true

StringUtils.isEmpty("") = true

StringUtils.isEmpty(" ") = false

StringUtils.isEmpty("bob") = false

StringUtils.isEmpty(" bob ") = false

Difference is that isEmpty() returns false if String parameter contains just whiltespaces. It considers whitespaces as a state of being non empty.

ADP Interview questions

1)Can string be used as primary key while working with hibernate? Can multiple fields be used as collective primary key?

2)why null key is stored at entity array 0th position

3)What is the difference between <bean id= “”> and <bean name=””>

4)Any disadvantage of lazy loading?what is n+1 problem in hibernate. How can we get rid of that. What types of fetch we have- lazy /eager?when to use which one

5)what is cascade.

6)Any other controller annotation you have worked? E.g @CacheController?

7)Difference b/w array and arraylist?

8)Underlying datastructure for arraylist?

9)Given a hashmap with 2 same keys.then how many keys we have in total?

10)Design pattern- Abstract design pattern – why is the need of abstract factory why client cant directly deal with multiple factories

11)Modify singleton pattern so that there are exactly 2 objects of the singleton class

12) Diff between singleton scope of a bean and singleton pattern

13 )Bean A’s creation depends upon Bean B. Here Bean A is singleton where as Bean B is prototype, then how can we create bean A at the time of context loading

14) What is lazy loading wrt spring bean creation.

15)get vs load in hibernate

16)Configure second level cache? where you give idle time property value?

17)what if the bean is found in second level cache but is dirty?

18)Webservices- how did you create WSDl from XSD

19)What is the name of dispatcher servlet XML file if the name of servlet in web.xml file is say mvc

20)How to configure multiple dispatcher servlets?Is it possible

21)@controller which is defined in root application context(Contextloaderlistnere configuration) and @service is is a bean defined in application context which is defined in <init-param> of dispatcher servlet , then how @controller file will access @service file and vice versa.

22)spring security- what is the default name of username and password field of the login page which is automatically created by spring.

What is the output of following program in first and second case highlighted below?

**public** **static** **void** meth(List<String> a){

// a=null; - **1st case**

// a.add("3"); -**2nd case**

}

**public** **static** **void** main(String[] args)

{

List<String> list=**new** ArrayList<>();

list.add("1");

list.add("2");

**for**(String a:list){

System.*out*.println(a);

}

*meth*(list);

**for**(String a:list){

System.*out*.println(a);

}

}

Answer:

1st case – 1,2 1,2

2nd Case- 1,2 1,2,3

Reason is : - <http://www.javaworld.com/article/2077424/learn-java/does-java-pass-by-reference-or-pass-by-value.html>

**public** **void** tricky(**Point** arg1, **Point** arg2)

{

arg1.x = 100;

arg1.y = 100;

**Point** temp = arg1;

arg1 = arg2;

arg2 = temp;

}

**public** **static** **void** main(**String** [] args)

{

**Point** pnt1 = **new** **Point**(0,0);

**Point** pnt2 = **new** **Point**(0,0);

**System**.**out**.println("X: " + pnt1.x + " Y: " +pnt1.y);

**System**.**out**.println("X: " + pnt2.x + " Y: " +pnt2.y);

**System**.**out**.println(" ");

tricky(pnt1,pnt2);

**System**.**out**.println("X: " + pnt1.x + " Y:" + pnt1.y);

**System**.**out**.println("X: " + pnt2.x + " Y: " +pnt2.y);

}

If we execute this main() method, we see the following output:

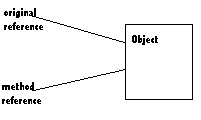
X: 0 Y: 0

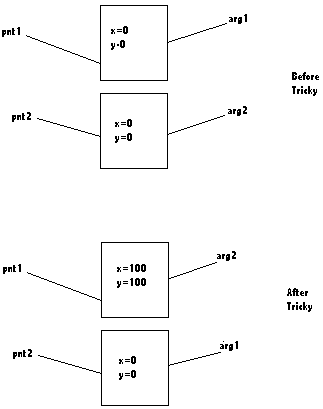
X: 0 Y: 0

X: 100 Y: 100

X: 0 Y: 0

The method successfully alters the value of pnt1, even though it is passed by value; however, a swap of pnt1 and pnt2 fails! This is the major source of confusion. In the main() method, pnt1 and pnt2 are nothing more than object references. When you pass pnt1 and pnt2 to the tricky() method, Java passes the references by value just like any other parameter. This means the references passed to the method are actually copies of the original references. Figure 1 below shows two references pointing to the same object after Java passes an object to a method.



Java copies and passes the *reference* by value, not the object. Thus, method manipulation will alter the objects, since the references point to the original objects. But since the references are copies, swaps will fail. As Figure 2 illustrates, the method references swap, but not the original references. Unfortunately, after a method call, you are left with only the unswapped original references. For a swap to succeed outside of the method call, we need to swap the original references, not the copies.

What happens if the web service response is failure?

What was the URL of your project’s webservices- http or https? How was it secured?

How to create XSD? How client stub is created?

JDK 8 streams feature

How to add configure dispatcher servlets