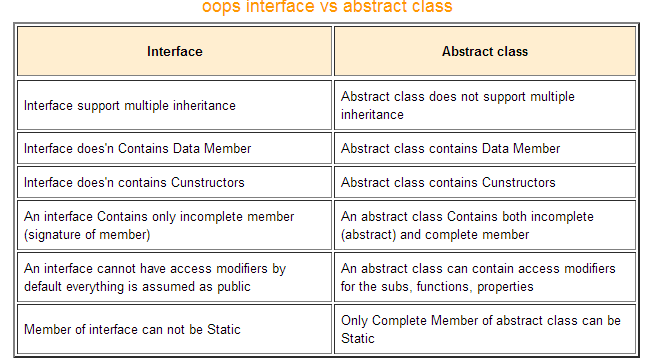
1. **Difference between interface and abstract class?**



1. **Why we use constructer in abstract class?**

* Abstract classes have constructors and those constructors are always invoked when a concrete subclass is instantiated. We know that when we are going to instantiate a class, we always use constructor of that class. Now every constructor invokes the constructor of its super class with an implicit call to super().
* We know constructor are also used to initialize fields of a class. We also know that abstract classes may contain fields and sometimes they need to be initialized somehow by using constructor.

1. **JSP implicit objects?**

|  |  |
| --- | --- |
| **Implicit Object** | **Description** |
| **request** | The **HttpServletRequest** object associated with the request. |
| **response** | The **HttpServletRequest** object associated with the response that is sent back to the browser. |
| **out** | The **JspWriter** object associated with the output stream of the response. |
| **session** | The **HttpSession** object associated with the session for the given user of request. |
| **application** | The **ServletContext** object for the web application. |
| **config** | The **ServletConfig** object associated with the servlet for current JSP page. |
| **pageContext** | The **PageContext** object that encapsulates the enviroment of a single request for this current JSP page |
| **page** | The **page** variable is equivalent to **this** variable of Java programming language. |
| **exception** | The **exception** object represents the **Throwable** object that was thrown by some other JSP page. |

1. **Difference between Servlet and jsp?**

|  |  |
| --- | --- |
| **JSP** | **Servlets** |
| JSP is a webpage scripting language that can generate dynamic content. | Servlets are Java programs that are already compiled which also creates dynamic web content. |
| JSP run slower compared to Servlet as it takes compilation time to convert into Java Servlets. | Servlets run faster compared to JSP. |
| It’s easier to code in JSP than in Java Servlets. | Its little much code to write here. |
| In MVC, jsp act as a view. | In MVC, servlet act as a controller. |
| JSP are generally preferred when there is not much processing of data required. | servlets are best for use when there is more processing and manipulation involved. |
| The advantage of JSP programming over servlets is that we can build custom tags which can directly call Java beans. | There is no such custom tag facility in servlets. |
| We can achieve functionality of JSP at client side by running JavaScript at client side. | There are no such methods for servlets. |

1. **Difference between Session factory and session?**

SessionFactory objects are one per application and Session objects are one per client.

SessionFactory is to create and manage Sessions. Session is to provide a CRUD interface for mapped classes, and also access to the more versitile Criteria API.

SessionFactory is thread safe where as Session is not thread safe.

1. **Difference between jBoss and Apache?**

**Tomcat** - is run by Apache community - Open source and has two flavors

**Tomcat** - Web profile - light weight which is only servlet container and does not support Java EE features like EJB, JMS etc.

**Tomcat EE** - This is a certified Java EE container, this supports all Java EE technologies.

No commercial support available (only community support)

**JBoss** - Run by RedHat

This is a full stack support for JavaEE and it is a certified Java EE container.

This includes TOmcat as web container internally.

This also has two flavors

Community version called Application Server (AS) - this will have only community support

Enterprise Application Server ( EAP) - For this you can have subscription based license (Its based on number of Cores you have in your servers.)

**Glassfish** - Run by Oracle

This is also a full stack certified Java EE Container. This has its own web container (not Tomcat).

This comes from Oracle itself,so all new specs will be tested and implemented with Glassfish first. So, always it would support the latest spec. I am not aware of its support models.

1. **Hibernate features?**

Hibernate is an Open Source persistence technology. Hibernate provide a way of mapping of applications class to the database table.

Main Features: It is ORM based and provide persistence management solution or persistent layer. It maps database tables to a class.

Here we are giving some features in points -

Hibernate provides Object/Relational mappings. Here is different O/R mapping strategies as multiple-objects to single-row mapping,

Polymorphic associations, bi-directional association, association filtering. It also provide XML mapping documents.

It provides different object oriented query languages. Minimal object oriented Hibernate query language (HQL), native SQL queries,

High object oriented concept of criteria.

It provide transparent persistence without byte code processing.

It introduces automatic Dirty Checking concept.

It supports detached object concept.

It supports tough concept of composite keys.

Automatic generation of primary key.

Hibernate provides Dual-layer Cache Architecture.

It provides session level cache and optional second-level cache.

It introduces Lazy initialization.

Hibernate provide outer join fetching.

It supports optimistic locking with versioning.

Optionally provide internal connection pooling and prepared statement caching.

At system initialization time it generate SQL.

It provide feature of J2EE integration.

It supports JMX and JCA.

1. **What is savepoint?**
2. **Difference between beanfactory and applicationContext?**

BeanFactory: Does not support the Annotation based dependency Injection.

ApplicationContext: Support Annotation based dependency Injection. -@Autowired, @PreDestroy

BeanFactory: Does not Support

ApplicationContext: Application contexts can publish events to beans that are registered as listeners

BeanFactory: Does not support way to access Message Bundle(internationalization (I18N)

ApplicationContext: Support internationalization (I18N) messages.

BeanFactory: Doesn’t support.

ApplicationContext: Support many enterprise services such JNDI access, EJB integration, remoting.

BeanFactory: By default its support Lazy loading

ApplicationContext: It's By default support Aggresive loading.

1. What are the types of dependency injections?

* Constructor-based dependency injection: It is accomplished when the container invokes a class constructor with a number of arguments, each representing a dependency on other class.
* Setter-based dependency injection: It is accomplished by the container calling setter methods on your beans after invoking a no-argument constructor or no-argument static factory method to instantiate your bean.

1. What is IoC container?

The IoC container is responsible to instantiate, configure and assemble the objects. The IoC container gets informations from the XML file and works accordingly. The main tasks performed by IoC container are:

* To instantiate the application class
* To configure the object
* To assemble the dependencies between the objects

There are two types of IoC containers. They are:

1. **BeanFactory**
2. **ApplicationContext**
3. What is flush method in hibernate?

**flush()** will synchronize your database with the current state of object/objects held in the memory but it does not commit the transaction. So, if you get any exception after flush() is called, then the transaction will be rolled back. You can synchronize your database with small chunks of data using flush() instead of committing a large data at once using commit() and face the risk of getting an **Out Of Memory Exception**.

1. What are strategies to solve the N+1 SELECT problem in Hibernate?

N+1 is one of the most famous question among java developers. It will give some performance issue if you where working on a large project. So it's better to resolve it using some below techniques.

**What is n+1 problem ?**

Ans: I am taking one example to let you understand that what n+1 problem is.

Suppose you have number of students in a college and every students have some number of books.

so one to many relation is between student and books.

Now suppose that you have to iterate through the collection of student and display all the books name he have. So the query will looks like this

"select \* from Students"

"select \* from Books where studentId=?"

Here you have 1 select statement for the student and if you have n number of students you have to fire n more query to select the books. So at the last you have to put n+1 select statement in order to perform this operation.

**Now the next question is how to solve it ?**

Using join fetching(it will join the parent and children and fetch all the information in a single statement) we can able to solve n+1 problem.

Now our next query will look like this

"from Students s join fetch s.Books b"

1. Which software development life cycle are you following?

AGILE