1. Difference between callable and runnable :

The Callable interface is similar to Runnable, in that both are designed for classes whose instances are potentially executed by another thread. **A Runnable, however, does not return a result and cannot throw a checked exception.**

[**https://dzone.com/articles/java-callable-future-understanding**](https://dzone.com/articles/java-callable-future-understanding)

1. **Return type of Callable interface :**

The **callable interface** was introduced in concurrency package, which is similar to the Runnable **interface**, but it can **return** any object, and is able to throw an Exception. The Java **Callable interface** uses Generics, so it can **return** any **type** of Object.

**Interface Callable<V>**

[V](https://docs.oracle.com/javase/7/docs/api/java/util/concurrent/Callable.html) call()

throws [Exception](https://docs.oracle.com/javase/7/docs/api/java/lang/Exception.html)

Computes a result, or throws an exception if unable to do so.

Returns:

computed result

Throws:

[Exception](https://docs.oracle.com/javase/7/docs/api/java/lang/Exception.html) - if unable to compute a result

The [Executors](https://docs.oracle.com/javase/7/docs/api/java/util/concurrent/Executors.html) class contains utility methods to convert from other common forms to Callable classes.

1. **Difference between sleep and wait :**

1)wait is called from synchronized context only while sleep can be called without synchronized block

2) waiting thread can be awake by calling notify and notifyAll while sleeping thread can not be awakened by calling notify method.  
  
3) wait is normally done on condition, Thread wait until a condition is true while sleep is just to put your thread on sleep.

4) wait for release lock on an object while waiting while sleep doesn’t release lock while waiting.  
  
5) The wait() method  is called on an Object on which the synchronized block is locked, while sleep is called on the Thread  
  
Read more: <https://javarevisited.blogspot.com/2011/12/difference-between-wait-sleep-yield.html#ixzz5SNc8OiBp>

1. Wait/Notify :

<https://www.journaldev.com/1037/java-thread-wait-notify-and-notifyall-example>

1. Synchronization :

<https://javarevisited.blogspot.com/2011/04/synchronization-in-java-synchronized.html>

1. Unchecked exception is handled or not - ?
2. HashMap Set deifference:

|  |  |
| --- | --- |
| **HashMap** | **Hash Set** |
| HashMap  is an implementation of Map interface | HashSet is an implementation of Set Interface |
| HashMap Stores data in form of  key-value pair | HashSet Store only objects |
| Put method is used to add element in map | Add method is used to add element is Set |
| In hash map hashcode value is calculated using key object | Here member object is used for calculating hashcode value which can be same for two objects so equal () method is used to check for equality if it returns false that means two objects are different. |
| HashMap is faster than HashSet because unique key is used to access object | HashSet is slower than Hashmap |

Read more: <https://javarevisited.blogspot.com/2011/09/difference-hashmap-vs-hashset-java.html#ixzz5SNtIDrIr>

1. How to make map synchronized :

In order to synchronize it we are using [Collections.synchronizedMap(hashmap)](https://docs.oracle.com/javase/7/docs/api/java/util/Collections.html" \l "synchronizedMap(java.util.Map)" \t "_blank)  it returns a thread-safe map backed up by the specified HashMap.

It is also mandatory that the user manually synchronizes on the returned map when [iterating over any of its collection](http://java67.blogspot.sg/2013/02/java-iterator-example-and-tutorial.html) views  
  
Read more: <http://www.java67.com/2015/02/how-to-synchronize-hashmap-in-java-with.html#ixzz5SNui5Rhd>

1. Difference between thread and process:

**Threads** are used for small tasks, whereas **processes** are used for more 'heavyweight' tasks – basically the execution of applications. Another **difference between** a **thread** and a **process** is that **threads** within the same **process** share the same address space, whereas different **processes** do not.

1. Can we call wait outside synchronized : No

<https://javarevisited.blogspot.com/2011/05/wait-notify-and-notifyall-in-java.html>

1. What is diff between servlet and jsp?

|  |  |
| --- | --- |
| Servlet is a java code. | JSP is a html based code. |
| Writing code for servlet is harder than JSP as it is html in java. | JSP is easy to code as it is java in html. |
| Servlet plays a controller role in MVC approach. | JSP is the view in MVC approach for showing output. |
| Servlet is faster than JSP. | JSP is slower than Servlet because the first step in JSP lifecycle is the translation of JSP to java code and then compile. |
| Servlet can accept all protocol requests. | JSP only accept http requests. |
| In Servlet, we can override the service() method. | In JSP, we cannot override its service() method. |
| In Servlet by default session management is not enabled, user have to enable it explicitly. | In JSP session management is automatically enabled. |
| In Servlet we have to implement everything like business logic and presentation logic in just one servlet file. | In JSP business logic is separated from presentation logic by using javaBeans. |
| Modification in Servlet is a time consuming task because it includes reloading, recompiling and restarting the server. | JSP modification is fast, just need to click the refresh button. |

1. Difference between doGet and doPost :

|  |  |
| --- | --- |
| **DoGet** | **DoPost** |
| In doGet Method the parameters are appended to the URL and sent along with header information | In doPost, parameters are sent in separate line in the body |
| Maximum size of data that can be sent using doget is 240 bytes | There is no maximum size for data |
| Parameters are not encrypted | Parameters are encrypted |
| DoGet method generally is used to query or to get some information from the server | Dopost is generally used to update or post some information to the server |
| DoGet is faster if we set the response content length since the same connection is used. Thus increasing the performance | DoPost is slower compared to doGet since doPost does not write the content length |
| DoGet should be idempotent. i.e. doget should be able to be repeated safely many times | This method does not need to be idempotent. Operations requested through POST can have side effects for which the user can be held accountable, for example, updating stored data or buying items online. |
| DoGet should be safe without any side effects for which user is held responsible | This method does not need to be either safe |

1. difference between Diff GET and POST :

<https://www.w3schools.com/tags/ref_httpmethods.asp>

1. difference between GenericServlet and HttpServet :

<http://www.java67.com/2012/12/difference-between-genericservlet-vs-httpservlet-jsp.html>

1. Pass parameters to different jsp :

session.setAttribute("userId", userid);

int userid = session.getAttribute("userId");

1. JSP compilation:

<https://www.geeksforgeeks.org/life-cycle-of-jsp/>

### Difference between DOM and SAX XML Parser

Here are few high-level **differences between DOM parser and SAX Parser**in Java:

1) DOM parser loads whole XML document in memory while SAX only loads a small part of the XML file in memory.

2) DOM parser is faster than SAX because it access whole XML document in memory.

3) SAX parser in Java is better suitable for large XML file than DOM Parser because it doesn't require much memory.

4) DOM parser works on Document Object Model while SAX is an event based XML parser.

That’s all on the *difference between SAX and DOM parsers in Java,* now it’s up to you on which XML parser you going to choose. I recommend using DOM parser over SAX parser if XML file is small enough and go with SAX parser if you don’t know the size of XML files to be processed or they are large.

## 18. Application Server vs Web Server

1**. Application Server**supports **distributed transaction and EJB**. While Web Server only supports Servlets and JSP.

2. Application Server can contain web server in them. most of App server e.g. JBoss or WAS has Servlet and JSP container.

3. Though its not limited to Application Server but they used to provide services like **Connection pooling**, **Transaction management**, messaging, clustering, load balancing and persistence. Now Apache tomcat also provides connection pooling.  
4. In terms of l*ogical difference between web server and application server*. web server is supposed to provide http protocol level service while application server provides support to web service and expose business level service e.g. EJB.

5. Application server are more heavy than web server in terms of resource utilization.

19. Find max salary as per Dept:

SELECT DepartmentID, MAX(Salary)

FROM Employee

GROUP BY DepartmentID

SELECT DepartmentName, MAX(Salary)

FROM Employee e RIGHT JOIN Department d ON e.DepartmentId = d.DepartmentID

GROUP BY DepartmentName

20. **ClassNotFoundException** and **NoClassDefFoundError**  :

**ClassNotFoundException** and **NoClassDefFoundError** occur when a particular class is not found at runtime. However, they occur at different scenarios.

**ClassNotFoundException** is an exception that occurs when you try to load a class at run time using **Class.forName()** or **loadClass()** methods and mentioned classes are not found in the classpath.

**NoClassDefFoundError** is an error that occurs when a particular class is present at compile time, but was missing at run time.

21. Singleton :

<https://www.geeksforgeeks.org/singleton-design-pattern/>

22. Builder design pattern

<https://javarevisited.blogspot.com/2012/06/builder-design-pattern-in-java-example.html>

23. <https://javaconceptoftheday.com/statement-vs-preparedstatement-vs-callablestatement-in-java/>

23. <https://javarevisited.blogspot.com/2012/06/jdbc-database-connection-pool-in-spring.html>

24. duplicate words in an array :

public class StringsCount{

public static void main(String args[]) {

String value = "This is testing Program testing Program";

String item[] = value.split(" ");

HashMap<String, Integer> map = new HashMap<>();

for (String t : item) {

if (map.containsKey(t)) {

map.put(t, map.get(t) + 1);

} else {

map.put(t, 1);

}

}

Set<String> keys = map.keySet();

for (String key : keys) {

System.out.println(key);

System.out.println(map.get(key));

}

}

}