

Top 55 Servlets Interview Question You Need to Know

Last updated on Dec 03,2019 9.5K Views



Ravi Kiran

Tech Enthusiast working as a Research Analyst at Edureka. Curious about learning...

Java Servlets are the main reason behind the simplicity in developing the **High-end** Web Application as Web Pages due to which the Java Web Application Technology is on the highest demand in present days. In this article, we will discuss the most frequently asked interview questions based on [Java Servlets](#).

- [Beginner Level Interview Questions](#)
- [Intermediate Level Interview Questions](#)
- [Advanced Level Interview Questions](#)

Beginner Level Interview Questions



Q1. What is a Servlet?

Ans: A **Servlet** is a Java program that runs on a Web server. It is similar to an applet but is processed on the server rather than a client's machine. **Servlets** are often run when the user clicks a link, submits a form, or performs another type of action on a website



Q2. What is a Cookie?

Ans: A **cookie** is a piece of information that is present between multiple client requests. A cookie has a *name*, a *single value*, and *optional attributes* such as a comment, path and domain qualifiers, a maximum age, and a version number.

Q3. How PrintWriter is different from ServletOutputStream?



Ans: PrintWriter is basically a character-stream class. On the other hand, **ServletOutputStream** is a byte-stream class. The PrintWriter class can be used to write only character-based information whereas ServletOutputStream class can be used to write

Subscribe to our Newsletter, and get personalized recommendations. 

 Sign up with Google

 Signup with Facebook

Already have an account? [Sign in](#).

Q5. What are the annotations used in Servlet 3?

Ans: The important 3 annotations used in the servlets are.

- **@WebServlet** : for servlet class.
- **@WebListener** : for listener class.
- **@WebFilter** : for filter class.

Q6. What is the difference between a Generic Servlet and HTTP Servlet?

Ans: A **common feature** between Generic Servlet and HTTP Servlet is both these Classes are [Abstract Classes](#). But, they do have differences between them which discussed as follows

Generic Servlet	HTTP Servlet	
Protocol Independent	Protocol Specific	
Belongs to javax.servlet package	Belongs to javax.servlet.http package	
supports only service() method	supports doGet(), doPost(), doHead() methods	

Q7. What is the use of RequestDispatcher Interface?

Ans: The **RequestDispatcher** interface defines the object that receives the request from the client and dispatches it to the resources such as a servlet, JSP, HTML file. The RequestDispatcher interface has the following two methods:

- 1 | `public void forward(ServletRequest request, ServletResponse response)`

Forwards request from one servlet to another resource like servlet, JSP, HTML etc.

- 1 | `public void include(ServletRequest request, ServletResponse response)`

Includes the **content** of the resource such as a servlet, JSP, and HTML in the response.

Q8. Can a JSP be called using a Servlet?

Ans: Yes, Servlet can call a **JSP** using **RequestDispatcher** interface.

Example:

```
1 | RequestDispatcher reqdis=request.getRequestDispatcher("log.jsp");  
2 | reqdis.forward(request,response);
```

Q9. Explain the Servlet Filter.

Ans: A **Filter** is defined as a **pluggable** object that is invoked either at the pre-processing or post-processing of a **request**.

Q10. Why do we need Servlet Filter?



Ans: We need **Servlet Filters** for the following reasons:

- **Logging** the request parameters to log files

Subscribe to our Newsletter, and get personalized recommendations. ✕



Sign up with Google



Signup with Facebook

Already have an account? [Sign in](#).

Q12. What is load-on-startup in Servlet?

Ans: The **load-on-startup** element of servlet in **web.xml** is used to load the servlet at the time of deploying the **project** or the **server** to start. This saves time for the **response** of the first request.

Q13. What is a WAR file?

Ans: The **WAR(Web Application Resource) file** specifies the web elements. Either a **Servlet** or **JSP** project can be converted into a war file. Moving one Servlet project from one place to another will be fast as it is combined into a single file.

Q14. What does the following code snippet in an XML file mean?

```
1 | <load-on-startup>1</load-on-startup>
```

Ans: Whenever a request for a **servlet** is placed, then the **servlet container** will initialize the servlet and load it. This process is defined in our config file called **web.xml**. But, by default, Container will not initialize the servlet, when the context is loaded. This can be achieved by defining the servlet in a **pre-initialization** procedure syntax **<load-on-startup>1</load-on-startup>**. Then, the servlet that we have defined in this tag will be **initialized** at the start when the context gets loaded before even getting the **request**.

Q15. How to get the server information in a servlet?

Ans: Yes, we can retrieve the information of a server in a servlet. We can use below code snippet to get the servlet information in a servlet through servlet context object.

```
1 | getServletContext().getServerInfo()
```

Q16. Explain MVC pattern.

Ans: **Model-View-Controller (MVC)** is a design pattern that divides a software application into three segments namely the **Model**, the **View** and the **Controller**.

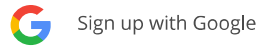
- A **model** deals with the behaviour of the application. It contains the data and business logic of the application. It notifies views and controllers when there is a change in its state.
- A **view** renders the information to the user so that it looks attractive and appealing. It takes information from the model using which it generates output.
- A **controller** takes input from a user and sends the command to model or view. It controls the flow of the application.



Q17. What is the workflow of a servlet?

Ans: **Servlet** is a Java tool that is used to create **Java Web Applications**. It is located at the server-side and helps to generate

Subscribe to our Newsletter, and get personalized recommendations. ✕



Already have an account? [Sign in.](#)



A **request** is received from a webpage by the servlet. The servlet redirects the request to the appropriate **JSP** page and the JSP page sends the **response** as a result page that is visible to the client.

Q18. Write a Hello World Program using Servlets.

Ans: The code to write a Hello World Program using Servlets is as follows:

```
1  import java.io.*;
2  import javax.servlet.*;
3  import javax.servlet.http.*;
4
5  public class HelloWorld extends HttpServlet {
6      private String message;
7      public void init() throws ServletException {
8          message = "Hello World";
9      }
10     public void doGet(HttpServletRequest request, HttpServletResponse response)
11     throws ServletException, IOException {
12         response.setContentType("text/html");
13         PrintWriter out = response.getWriter();
14         out.println("<h1>" + message + "</h1>");
15     }
16     public void destroy() {
17     }
18 }
```



[Java Certification Training Course](#)

[Instructor-led Sessions](#)

[Real-life Case Studies](#)

[Assignments](#)

[Lifetime Access](#)

[Explore Curriculum](#)

Q19. What are the life cycle methods of a servlet?

Ans: The Servlet Life Cycle consists of three methods:

- **public void init(ServletConfig config):** This method is used by the container to initialize the servlet, this method is invoked only once in the lifecycle of the servlet.
- **public void service(ServletRequest request, ServletResponse response):** This method is called once for every request, a container can't invoke service() method until unless init() method is executed.
- **public void destroy():** This method is invoked once when a servlet is unloaded from memory.



Subscribe to our Newsletter, and get personalized recommendations. 



Sign up with Google



Signup with Facebook

Already have an account? [Sign in.](#)

Q20. Can you create a Deadlock condition on a servlet?

Ans: Yes, a **Deadlock** situation can be created on a servlet by calling **doPost()** method inside **doGet()** method, or by calling a **doGet()** method inside **doPost()** method will successfully create a **deadlock** situation for a servlet.

Q21. Can we fetch the attributes related to a servlet on a different servlet?

Ans: The **attributes** of another servlet that we are looking for can be accessed by using its **name**. The syntax is described as follows:

```
1 | Context.setAttribute ("name"," value")
2 | Context.getAttribute ("name")
```

Q22. What do you mean by MIME type?

Ans: The **"Content-Type"** response header is called **MIME** (Multipurpose Internet Mail Extensions) Type. The server sends MIME type to the client to let the client know the kind of data it's sending. It helps the client in rendering the data for the user. Some of the most commonly used mime types are **text/HTML**, **text/XML**, **application/XML**.

Q23. What is the ServletConfig object?

Ans: **javax.servlet.ServletConfig** is used to pass configuration information to Servlet. Every servlet has its own **ServletConfig** object and servlet container is responsible for instantiating this object. We can provide servlet **init** parameters in **web.xml** file or through use of **WebInitParam** annotation.

Q24. What is the difference between ServletConfig and ServletContext?

Ans: Some of the major differences between **ServletConfig** and **ServletContext** are:

ServletConfig	ServletContext	
ServletConfig is a unique object per servlet.	ServletContext is a unique object for a complete application.	
ServletConfig is used to provide the init parameters to the servlet.	ServletContext is used to provide the application-level init parameters that all other servlets can use.	
Attributes in the ServletConfig object cannot be set.	We can set the attributes in ServletContext that other servlets can use.	

Syntax for ServletConfig:

```
1 | public ServletConfig getServletConfig();
```

Example of ServletConfig:



```
1 | ServletConfig config=getServletConfig();
```

Subscribe to our Newsletter, and get personalized recommendations. ✕



Sign up with Google



Signup with Facebook

Already have an account? [Sign in.](#)

Q25. What is the use of `HttpServletRequestWrapper` and `HttpServletResponseWrapper`?

Ans: Both `HttpServletRequestWrapper` and `HttpServletResponseWrapper` classes are used to help developers with a custom implementation of a servlet **request** and **response** types. Programmers can extend these classes and override only the specific methods that they need to implement for customized **request** and **response** objects.

Q26. Write a simple Servlet program to print the contents of HTML.

Ans: We can print the contents of HTML using the following steps:

Step 1: Get the object of `PrintWriter` using request.

```
1 | PrintWriter out = response.getWriter();
```

Step 2: Now print HTML.

```
1 | out.println("Hello World");
```

Q27. What do you mean by InterServlet communication?

Ans: **InterServlet communication** is a method to invoke another servlet using `RequestDispatcherforward()` and `include()` methods and provide additional attributes in the request for other servlet use.

Q28. How can we refresh automatically when new data is entered into the database?

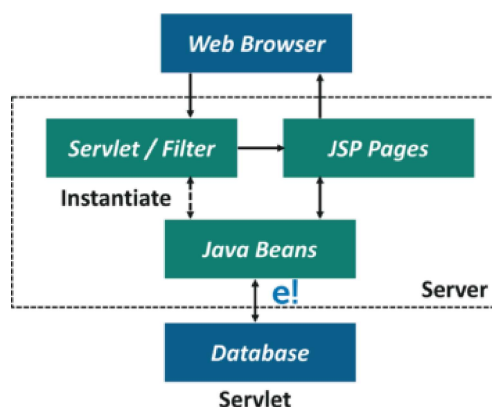
Ans: The following code segment can be used to **automatically update** the database.

```
1 | response.setHeader("Refresh",5);
```

This will update the browser after every **5 seconds**

```
1 | response.setHeader("Refresh","5", URL="http://localhost:8080/myServlet/Request.do0");
```

This will refresh Request.do after **5 seconds**



Subscribe to our Newsletter, and get personalized recommendations. 



Sign up with Google



Signup with Facebook

Already have an account? [Sign in.](#)

The `<input type="hidden">` defines a **hidden input field**. A **hidden field** let web developers include data that cannot be seen or modified by users when a **form** is submitted. A **hidden field** often stores what database record that needs to be updated when the **form** is submitted

- **Cookies**

A small text file created by a website that is stored in the user's computer either temporarily for that session only or permanently on the hard disk. **Cookies** provide a way for the website to recognize you and keep track of your preferences

- **URL Rewriting**

URL rewriting is an automatic process of altering a program written for manipulating the parameters in a **URL** (Uniform Resource Locator). **URL** manipulation is employed as a convenience by a Web server administrator, or for nefarious purposes by a hacker.

- **Session Management API**

Session Management API is built on top of the **Request-Response** methods for session tracking. Session Tracking is a way to maintain state/data of a user. It is also known as session management in servlet.

Q30. How do you get the IP address of the client in servlet?

Ans: We can use the following code to get the **client IP address** in servlet.

```
1 | request.getRemoteAddr()
```

Q31. How is an application exception handling is done using a Servlet?

Ans: The **doGet()** method and **doPost()** method throw the **ServletException** and **IOException**. The browser understands only HTML. Hence, when an exception is thrown by an application, then, the servlet container processes the exception and generates an **HTML response**. Same happens with other errors such as error 404.

Servlet API supports **customized** Exception and Error Handler servlets that can be configured in the deployment descriptor, the purpose of these servlets is to handle the Exception thrown by application and send HTML response that is useful for the user. We can provide a link to the application home page or the details that let the user know what went wrong.

The **Web.XML** configuration is as follows:


```
1 | <error-page>
2 |     <error-code>404</error-code>
3 |     <location>/AppExceptionHandler</location>
4 | </error-page>
5 | <error-page>
6 |     <exception-type>javax.servlet.ServletException</exception-type>
7 |     <location>/AppExceptionHandler</location>
8 | </error-page>
```

Q32. What is Servlet Interface?



Ans: The central abstraction in the Servlet API is called as the **Servlet Interface**. All servlets in the Web Application implement this interface, either directly or, most commonly by extending a class that implements it.

Subscribe to our Newsletter, and get personalized recommendations. ✕

 Sign up with Google

 Signup with Facebook

Already have an account? [Sign in](#).

Q34. What do you mean by CGI and what are its drawbacks?

Programming & Frameworks Training

[FULL STACK WEB
DEVELOPMENT INTERNSHIP
PROGRAM](#)

Full Stack Web
Development Internship
Program

Reviews

★★★★★ 5(2020)



[JAVA CERTIFICATION
TRAINING COURSE](#)

Java Certification Training
Course

Reviews

★★★★★ 4(49763)



[PYTHON SCRIPTING
CERTIFICATION
TRAINING](#)

Python Scripting
Certification Training

Reviews

★★★★★ 5(10541)



[PYTHON DJANGO
CERTIFICATION
TRAINING COURSE](#)

Python Django
Certification Training
Course

Reviews

★★★★★ 5(6022)

Ans: CGI is an abbreviation for **Common Gate Interface**. It consists of a set of code segments at the **server-side** using which the server interacts with the client running on the web. The drawbacks of CGI are as follows:

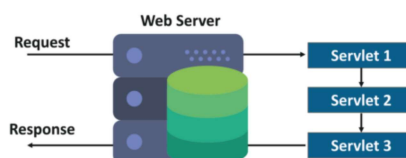
- If there are multiple incoming requests, then the response generated will be very slow, which results in **low efficiency**.
- CGI is **platform dependent**.

Q35. Explain Web Container.

Ans: A web container or a Servlet container is used to interact with the Servlet and includes all the **Servlet, JSP, XML** files inside it. Web Container's responsibility is to manage the life cycle of a servlet and to help to map the **URL** of a specific servlet. A web container is also used creates the object of a servlet.

Q36. What do you mean by the Servlet Chaining?

Ans: Servlet Looping or Chaining is a process where the output of one servlet is given as an input to another servlet and the last servlet output is considered as the **actual output** which is provided to the client.



Q37. Why do we use sendredirect() method?

Ans: The **sendredirect()** method basically works at the client-side. It is used to redirect the response to another resource like **Servlet, JSP, HTML**.

The syntax for **sendredirect()** method is as follows:

```
1 | void send Redirect(URL);
```



Example:

```
1 | response.sendRedirect("http://www.google.com");
```

Subscribe to our Newsletter, and get personalized recommendations. ✕

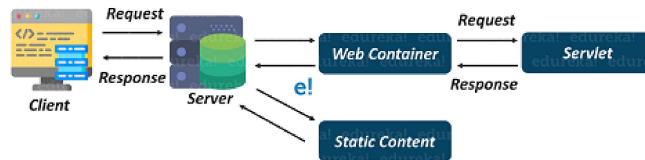


Sign up with Google



Signup with Facebook

Already have an account? [Sign in.](#)



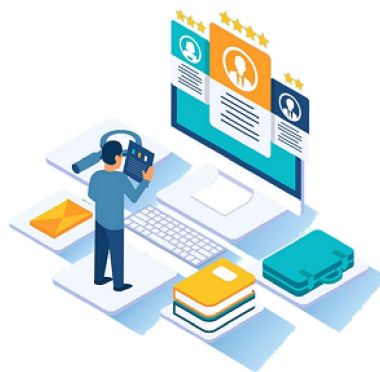
Q39. What do you mean by Servlet Context?

Ans: **Servlet Context** is referred to as an object that has information about the application and the Web Container. Using the Servlet context we can **log events**, get the **URL** of the specific resource, and store the attributes for other servlets to use.

The important methods of servlet context are as follows:

- **getInitParameter ()**: return the value of parameter.
- **getInitParameterNames ()**: returns the name of parameter.
- **void setAttribute ()**: used to set the values of attributes.
- **void getAttribute ()**: used to get the values of attributes.
- **void removeAttribute ()**: used to remove the attribute.

Advanced Level Interview Questions



Q40. What is the difference between Context Parameter and Context Attribute?

Ans: The main difference is, **Context Parameter** is a value stored in the deployment descriptor, which is the **web.xml** and is loaded during the deployment process. On the other hand, **Context Attribute** is the value which is set dynamically and can be used throughout the application.

Q41. Can you refresh servlet in client and server-side automatically?

Ans: Yes, There are a couple of primary ways in which a servlet can be **automatically** refreshed. One way is to add a **"Refresh"** response header containing the number of seconds after which a refresh should occur. The **TestServlet** class below shows an example of this.



```

1 import java.io.IOException;
2 import java.io.PrintWriter;
3 import java.util.Date;
4 import javax.servlet.ServletException;

```

Subscribe to our Newsletter, and get personalized recommendations. ✕



Sign up with Google



Signup with Facebook

Already have an account? [Sign in.](#)

```

18 }
19     response.setContentType("text/html");
20     response.addHeader("Refresh", "5");
21     PrintWriter out = response.getWriter();
22     out.println("TestServlet says hi at " + new Date());
23 }
24 }

```

Q42. What is Pure Servlet?

Ans: **Pure servlet** is known as a servlet that is used to create java objects that can be implemented from **javax.servlet.Servlet** interface.

Q43. What do you mean by HttpServlet and how it is different from the GenericServlet?

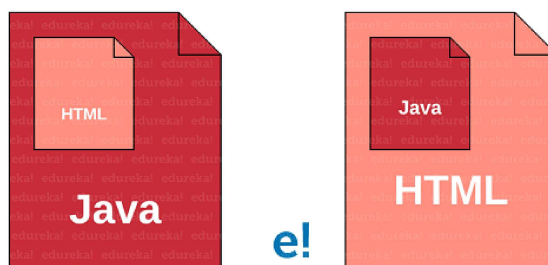
Ans: **HttpServlet** is basically extended from GenericServlet and it also inherits the properties of **GenericServlet**. HttpServlet defines an **HTTP protocol** servlet while GenericServlet defines a **generic, protocol-independent** servlet.

Q44. Which exception is thrown if the servlet is not initialized properly?

Ans: The **Servlet Exception** or **Unavailable Exception** is thrown if the servlet is not initialized properly.

Q45. How do we translate JSP?

Ans: In a servlet, the Java code is written in HTML but **JSP(Java Server Page)** allows us to write Java code in HTML. JSP allows easy development of web pages and allows web designer and web developer to work independently. All JSP pages are translated into servlet and web container is responsible for translating JSP into the servlet.



Q46. How can you create a session in servlet?

Ans: We can get **HttpSession** object by calling the **public method getSession()** of **HttpServletRequest**. The following code segment will help us.



```
1 | HttpSession session = request.getSession();
```

Subscribe to our Newsletter, and get personalized recommendations. 



Sign up with Google



Signup with Facebook

Already have an account? [Sign in.](#)

Ans: The difference between **sendRedirect()** and **Forward()** can be explained as follows:

sendRedirect():

- **sendRedirect()** method is declared in **HttpServletResponse** Interface.
- The **syntax** for the function is as follows:

```
1 | void sendRedirect(String URL)
```

- This method redirects the client request for further processing, the new location is available on a different server or different context. The web container handles this and transfers the request using the browser, this request is visible in the browser in the form of a new request. It is also called a client-side redirect.

Forward():

- **Forward()** method is declared in the **RequestDispatcher** Interface.
- The syntax for the function is as follows:

```
1 | forward(ServletRequest request, ServletResponse response)
```

- This passes the request to another resource for processing within the same server, another resource could be any servlet, JSP page. Web container handles the **Forward()** method. When we call **Forward()** method, a request is sent to another resource without informing the client, about the resource that will handle the request. It will be mentioned on the **requestDispatcher** object which we can be got in two ways. Either using **ServletContext** or **Request**.

Q49. Explain the working of service() method of a servlet.

Ans: The **service()** method is actually the main method that is expected to perform the actual task. The servlet container calls the **service()** method to handle requests coming from the **client/browsers** and to provide the response back to the client.

Each time the server receives a request for a **servlet**, the server creates a new thread and calls for the service. The **service()** method checks the **HTTP** request type and calls the respective methods as required.

The sample code for the same is as follows:

```
1 | public void service(ServletRequest request, ServletResponse response)
2 |     throws ServletException, IOException{
3 | }
```

The **container** calls the **service()** method and service method invokes **doGet()**, **doPost()**, **doPut()**, **doDelete()**, methods as needed. So you have nothing to do with **service()** method but you override either **doGet()** or **doPost()** depending on the request you receive from the client-end.

Q50. Can you send an Authentication error from a Servlet?

Ans: Yes, we can use **setStatus(statuscode)** method of **HttpServletResponse** to send an authentication error. All we have to do is to set an error code and a valid reason along with the error code.

```
1 | response.sendError(404, "Page not Found!!!" );
```



Q51. How do you configure a centralized error handler in servlets?

Subscribe to our Newsletter, and get personalized recommendations. ✕



Sign up with Google



Signup with Facebook

Already have an account? [Sign in.](#)

Ans: Creating and Setting cookies using servlet involves the following steps:

Step 1: Creating a Cookie object



[Java Certification Training Course](#)

[Weekday / Weekend Batches](#)

[See Batch Details](#)

Call the Cookie constructor with a cookie **name** and a cookie **value**, both of String Datatype. Care should be taken that these particular symbols([] () = , " / ? @ : ;) are excluded while declaring name and values.

```
1 | Cookie cookie = new Cookie("key", "value");
```

Step 2: Setting the maximum age

We shall use **setMaxAge** to specify how long the cookie should be valid. Following code-segment would set up a cookie for 24 hours.

```
1 | cookie.setMaxAge(60*60*24);
```

Step 3: Sending the Cookie into the HTTP response headers

We can use **response.addCookie** to add cookies in the **HTTP response header** as follows:

```
1 | response.addCookie(cookie);
```

Q53. How can you use a servlet to generate a plain text instead of HTML?

Ans: A servlet basically generates HTML. We can set the content-type response header by any of the methods. The example for producing text message EDUREKA using the HTML code inside the servlets.

```
1 | import java.io.*;
2 | import javax.servlet.*;
3 | import javax.servlet.http.*;
4 | public class EDUREKA extends HttpServlet{
5 |     public void doGet(HttpServletRequest request, HttpServletResponse response)
6 |         throws ServletException, IOException{
7 |         response.setContentType("text/html");
8 |         PrintWriter out = response.getWriter();
9 |         out.println("<!DOCTYPE HTML PUBLIC \"-//W3C//DTD HTML 4.0 \" +\"Transitional//EN\">n\" +\"<H1
10 |     }
11 | }
```

Q54. Explain the steps involved in placing a servlet within a package?



Ans: The Packages are used to separate the servlets under a directory to eradicate confusion among programmers working simultaneously on creating servlets on the same server. This can be done through the following steps:

Subscribe to our Newsletter, and get personalized recommendations. 



Sign up with Google



Signup with Facebook

Already have an account? [Sign in.](#)

```
7         response.setContentType("text/html");
8         PrintWriter out = response.getWriter();
9         String docType = "<!DOCTYPE HTML PUBLIC \"-//W3C//DTD HTML 4.0 \" + \"Transitional//EN\">\n";
10        out.println(docType + "<HTML>\n" + "<HEAD><TITLE>Edureka</TITLE></HEAD>\n" + "<BODY BGCOLOR="
11    }
12 }
```

55. Explain steps to establish JDBC Connections.

Ans: The steps to be followed to establish a [JDBC connection](#) are as follows:

- **Import JDBC Packages:** Add **import** statements to your [Java program](#) to import required classes in your Java code.
- **Register JDBC Driver:** In this step, [JVM](#) loads the desired driver implementation into memory so that it can fulfill the JDBC requests. There are 2 approaches to register a driver.
- The most suitable approach to register a driver is to use Java's **forName()** method to dynamically load the driver's class file into memory, *which automatically registers it*. This method is suitable as it allows you to make the driver registration configurable and portable.

```
1  try {
2      Class.forName("oracle.jdbc.driver.OracleDriver");
3  }
4  catch(ClassNotFoundException ex)
5      System.out.println("Error: unable to load driver class!");
6      System.exit(1);
7  }
```

- The second approach you can use to register a driver is to use the static **registerDriver()** method.

```
1  try {
2      Driver myDriver = new oracle.jdbc.driver.OracleDriver();
3      DriverManager.registerDriver( myDriver );
4  }
5  catch(ClassNotFoundException ex){
6      System.out.println("Error: unable to load driver class!");
7      System.exit(1);
8  }
```

- You should use the *registerDriver()* method if you are using a non-JDK compliant JVM, such as the one provided by Microsoft. Here each form requires a database **URL**.
- **Database URL Formulation:** URL Formulation is necessary to create a properly formatted address that points to the database to which you want to connect. Once you loaded the driver, you can establish a connection using the **DriverManager.getConnection()** method. **DriverManager.getConnection()** methods are—
 - getConnection(String url)
 - getConnection(String url, Properties prop)
 - getConnection(String url, String user, String password)

- **Create a connection object**

You can create a connection using the database URL, username, and password and also using properties object.

