What is Servlet

- Servlets are the java classes that extend the functionality of a web server by dynamically generating web pages.
- A runtime environment known as *Servlet Container* manages servlet loading and unloading, and works with the web server to direct requests to servlets and to send output back to web clients.

Key Advantages

- Performance
  - As compared to Common Gatewat Interface (CGI).
- Simplicity
  - Unlike Applets no web browser provided virtual machine is needed to run it.
- HTTP Sessions
  - HttpSession Class provided by servlet api.
- Access to Java Technology
  - Access to full range of java technologies.

Servlet Lifecycle

- Servlets operate in the context of a request and response model managed by a servlet engine which does the following:
  - Loads the servlet when it is first requested.
  - Calls the servlet's init() method.
  - Handles any number of requests by calling the servlet's service() method.
  - When shutting down, calls the destroy() method of each active servlet.

Servlet Lifecycle

- There are standard base classes that implement the servlet callback methods:
  - javax.servlet.GenericServlet
  - javax.servlet.http.HttpServlet
- Servlet programming, then, consists of subclassing one of these classes and overriding the necessary method to accomplish the specific task at hand.
- These methods are init(), service() and destroy().

Servlet Lifecycle

- init:
  - After the servlet is loaded, but before it services any requests, the servlet engine calls an initialization method with the following signature:
  - public void init(ServletConfig config) throws ServletException
  - This method is called only once, just before the servlet is placed into service.
  - To maintain a reference to the servlet context, the config object must be stored as an instance variable which is done by this init() method.

Servlet Lifecycle

- service:
  - After the init() method completes successfully, the servlet is able to accept requests.
  - By default, only a single instance of the servlet is created, and the servlet container dispatches each request to the instance in a separate thread.
  - public void service(ServletRequest request, ServletResponse response) throws ServletException, IOException

Servlet Lifecycle

- ServletRequest object is constructed by the servlet container and acts as a wrapper for information about the client and the request.
  - This includes the identity of the remote system, the request parameters and any input stream associated with the request.
- ServletResponse object provides means for a servlet to communicate its results back to the original requester.
  - It includes methods for opening an output stream and for specifying the content type and length.

Servlet Lifecycle

- Being more specific to http requests and responses, extends HttpServlet class which provides specialized methods corresponding to each HTTP request method.
- GET requests are handled by doGet() and POST requests are handled by doPost() method.
  - public void **doGet**(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException
  - public void **doPost**(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException

Servlet Lifecycle

- destroy:
  - The destroy() method is used at the time of unloading any servlet by the servlet container.
  - The servlet container notifies each loaded servlet about this event by calling destroy() method.
  - By overriding destroy(), you can release any resources allocated during init().
  - Calling destroy() yourself won't actually unload the servlet. Only the servlet container can cause this to happen. There is no architected way for a servlet to unload itself programmatically.

ServletRequest Interface

- It encapsulates the details of the client request.
- Generic Request: Protocol Independent
  - Finds the host name and IP address of the client.
  - Retrieves request parameters.
  - Gets and sets attributes.
  - Gets the input and output streams.
- HTTP-Specific Request:
  - Reads and writes HTTP headers.
  - Gets and sets cookies.
  - Gets path information.
  - Identifies the HTTP session, if any.

ServletResponse Interface

- The function of the servlet response object is to convey results generated by a servlet back to the client that made the request.
- A ServletResponse operates mainly as a wrapper for an input stream, as well as information about its content type and length.
- It is created by the servlet container.
- Two types: Generic and HTTP-Specific

ServletResponse Interface

- Generic Response:
  - It has both a generic protocol-independent interface and an HTTP-specific one.
- HTTP-Specific Response:
  - It adds methods for manipulating the status code, status message, and response headers.

Servlet Context

- It is very important interface supplied by the servelt container to provide services to a web application.
- It is used to get
  - The capability to store and retrieve attributes between invocations, and to share these attributes with other servlets.
  - The capability to read the contents of files and other static resources in the web application.
  - A means to dispatch requests to each other.
  - A facility for logging errors and informational messages.

Thread Models

- Default Threading Model:
  - Servlet container loads only a single instance of a servlet.
  - Requests serviced by the servlet will run in separate threads, but share the same instance.
  - Applications become scalable as fewer resources are needed.
  - Instance variables are not thread safe here.
- Single Thread Model:
  - One instance and corresponding thread generated for each new request and hence there is no overlap.

What is a Web Server???

- Web server is a place where your actual web application resides and executes from.
- Web server provides a platform to the web application to get executed and displayed to the user.
- A web server listens to the requests at a well known port number.
- E.g. Apache Tomcat, IBM Websphere, Sun Java Application Server, Glass Fish etc.
- Here we will take reference of Tomcat versions.

How to run a Servlet???

- Set path of ***servlet-api*** in compiler's classpath.
- Compile servlet class.
- Prepare/Update Web Application Deployment Descriptor file ***web.xml***.
- Place the class file of servlet and web.xml at a proper place in directory structure followed by a specific web server directory for your web application.
- Start the web server.
- Run servlet in the web browser.

How to run a Servlet???

- Setting path of servlet api:
  - My Computer-Properties-Advanced-Environment Variables-Path/Classpath variable to be edited.
  - You can find servlet.jar (in Tomcat 4.0 or below versions) or servlet-api.jar (in Tomcat 5.0 and higher versions) at $CATALINA_HOME (Apache/Tomcat/) /common/lib directory.
  - Place the fully qualified path of this servlet api along with its name in that Classpath variable.

How to run a Servlet???

- Compile the servlet:
  - As servlet is after all a java class only, the compilation process remains the same here as with any simple java class.
  - On the command prompt:
    - c:/>javac ServletClass.java
  - Hence you will get a class file with the same file name as of the java file and at the same location of that java file.

How to run a Servlet???

- Dealing with web.xml:
  - Here you are specifying description of all the servlets you have build for your web application.

```
<?xml version="1.0">
<web-app>
        …
        <servlet>
            <servlet-name>Demo</servlet-name>
            <servlet-class>DemoServlet</servlet-class>
        </servlet>
        …
</web-app>
```

How to run a Servlet???

- Dealing with web.xml:
  - Here, servlet-name and servlet-class can be same or different as per the choice of user.
  - Servlet-name is how would you like to identify your servlet-class.
  - It is also possible to specify a mapping of URIs to servlets with the <servlet-mapping> element in web.xml.
  - In most case, modifying the web.xml file requires either the Web Application or servlet container to be restarted before any changes take effect.

How to run a Servlet???

- Placing servlet class file and web.xml at proper place:
  - web.xml file is to be placed in the WEB-INF folder of your web application.
    - webapps/DemoApplication/WEB-INF/web.xml
  - Class file of your servlet is to be placed in classes folder i.e. residing inside the WEB-INF folder of your web application.
    - webapps/ DemoApplication/ WEB-INF/ classes/ DemoServlet.class

How to run a Servlet???

- Starting and Stopping Web Server:
  - In the web server directory, you can find files in the .exe or .bat form to start and stop the services of web server.
  - For Tomcat, you can find these files in $CATALINA_HOME/bin directory.
  - You can start the web server by executing the startup file over there and then can execute your web application in the browser.
  - After closing the web application, you can stop the web browser by relevant shutdown file in the same bin directory.

How to run a Servlet???

- Run servlet in the web browser:
  - You need to specify a URL in the address bar of your web browser to invoke your web application.
  - The general format is as given below:

  http://<servername>/<webappname>/servlet/<servletname>

  E.g.

  http://localhost:8080/DemoApp/servlet/Demo