Basics of Servlets

IT Expert Trainer
By Ram Lovewanshi
Mobile: 7648904739

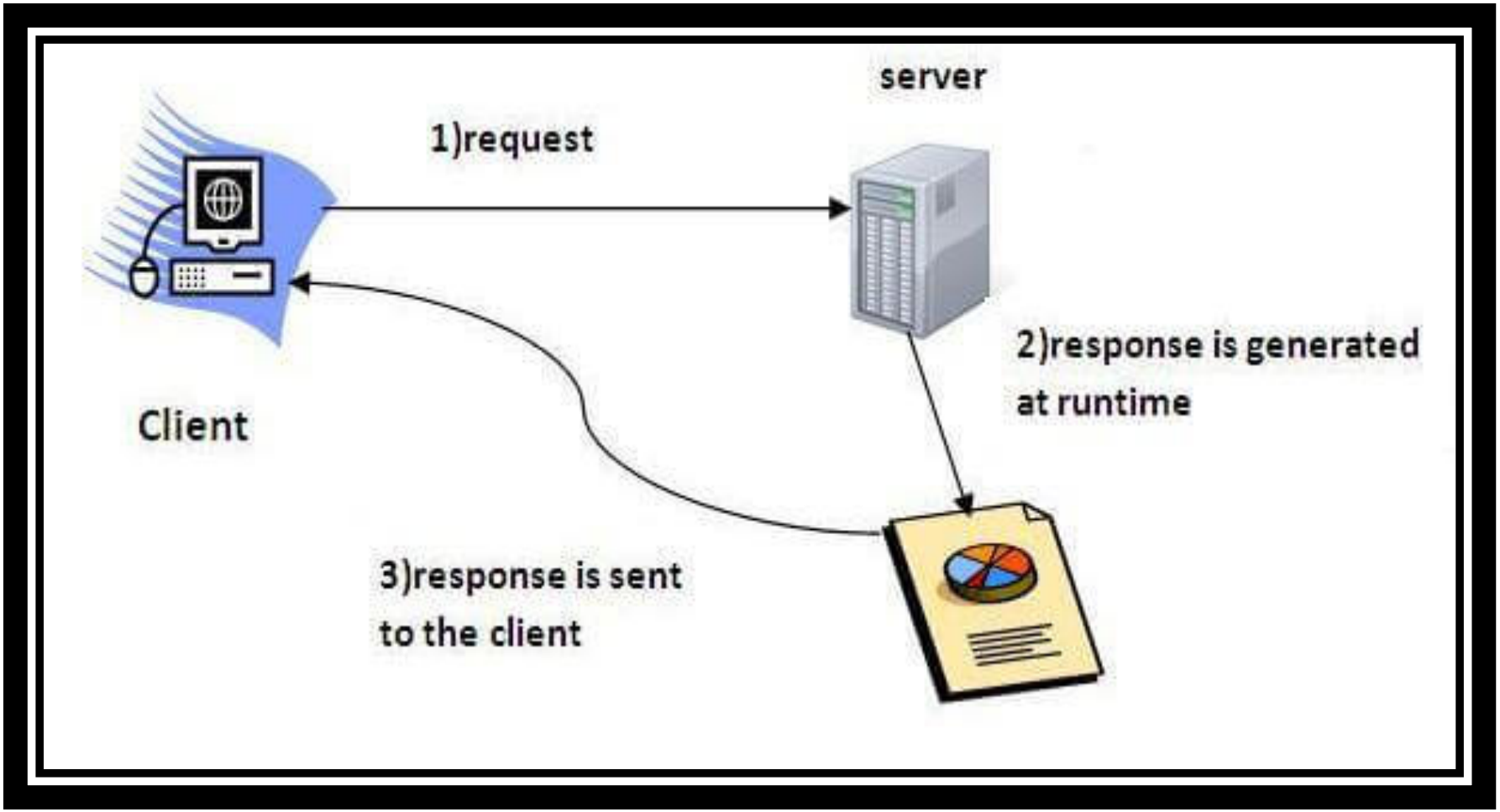
Servlets

- **Servlet** technology is used to create a web application (resides at server side and generates a dynamic web page).
- There are many interfaces and classes in the Servlet API such as Servlet, GenericServlet, HttpServlet, ServletRequest, ServletResponse, etc.

What is a Servlet?

- Servlet is a technology which is used to create a web application.
- Servlet is an API that provides many interfaces and classes including documentation.
- Servlet is an interface that must be implemented for creating any Servlet.
- Servlet is a class that extends the capabilities of the servers and responds to the incoming requests. It can respond to any requests.
- Servlet is a web component that is deployed on the server to create a dynamic web page.

Servlet


server
1)request
2)response is generated
at runtime
Client
3)response is sent
to the client

What is a web application?

- A web application is an application accessible from the web. A web application is composed of web components like Servlet, JSP, Filter, etc. and other elements such as HTML, CSS, and JavaScript. The web components typically execute in Web Server and respond to the HTTP request.

SL.NO	Static Web Page	Dynamic Web Page
1.	In static web pages, Pages will remain same until someone changes it manually.	In dynamic web pages, Content of pages are different for different visitors.
2.	Static Web Pages are simple in terms of complexity.	Dynamic web pages are complicated.
3.	In static web pages, Information are change rarely.	In dynamic web page, Information are change frequently.
4.	Static Web Page takes less time for loading than dynamic web page.	Dynamic web page takes more time for loading.
5.	In Static Web Pages, database is not used.	In dynamic web pages, database is used.
6.	Static web pages are written in languages such as: HTML, JavaScript, CSS, etc.	Dynamic web pages are written in languages such as: Servlet, JSP, PHP,ASP.NET, etc.
7.	Static web pages does not contain any application program .	Dynamic web pages contains application program for different services.

Servlet API

- The javax.servlet and javax.servlet.http packages represent interfaces and classes for servlet api.
- The **javax.servlet** package contains many interfaces and classes that are used by the servlet or web container. These are not specific to any protocol.
- The **javax.servlet.http** package contains interfaces and classes that are responsible for http requests only.

Interfaces in javax.servlet package

1. Servlet
2. ServletRequest
3. ServletResponse
4. RequestDispatcher

Interfaces in javax.servlet.http package

1. HttpServletRequest
2. HttpServletResponse
3. HttpSession

Servlet Interface

- **Servlet interface provides** common behavior t o all the servlets. Servlet interface defines methods that all servlets must implement.
- Servlet interface needs to be implemented for creating any servlet (either directly or indirectly). It provides 3 life cycle methods that are used to initialize the servlet, to service the requests, and to destroy the servlet and 2 non-life cycle methods.

Methods of Servlet interface

- There are 5 methods in Servlet interface. The init, service and destroy are the life cycle methods of servlet. These are invoked by the web container.

Method	Description
public void init(ServletConfig config)	initializes the servlet. It is the life cycle method of servlet and invoked by the web container only once.
public void service(ServletRequest request, ServletResponse response)	provides response for the incoming request. It is invoked at each request by the web container.
public void destroy()	is invoked only once and indicates that servlet is being destroyed.
public ServletConfig getServletConfig()	returns the object of ServletConfig.
public String getServletInfo()	returns information about servlet such as writer, copyright, version etc.

GenericServlet class

- **GenericServlet** class

  implements **Servlet**, **ServletConfig** and **Serializable** interfaces. It provides the implementation of all the methods of these interfaces except the service method.
- GenericServlet class can handle any type of request so it is protocol-independent.
- You may create a generic servlet by inheriting the GenericServlet class and providing the implementation of the service method.

Methods of GenericServlet class

1. **public void init(ServletConfig config)** is used to initialize the servlet.
2. **public abstract void service(ServletRequest request, ServletResponse response)** provides service for the incoming request. It is invoked at each time when user requests for a servlet.
3. **public void destroy()** is invoked only once throughout the life cycle and indicates that servlet is being destroyed.
4. **public ServletConfig getServletConfig()** returns the object of ServletConfig.
5. **public String getServletInfo()** returns information about servlet such as writer, copyright, version etc.

Methods of GenericServlet class

1. **public void init()** it is a convenient method for the servlet programmers, now there is no need to call super.init(config)
2. **public ServletContext getServletContext()** returns the object of ServletContext.
3. **public String getInitPaarameter(String name)** returns the parameter value for the given parameter name.
4. **public Enumeration getInitParameterNames()** returns all the parameters defined in the web.xml file.
5. **public String getServletName()** returns the name of the servlet object.

HttpServlet class

- The HttpServlet class extends the GenericServlet class and implements Serializable interface. It provides http specific methods such as doGet, doPost, doHead, doTrace etc.

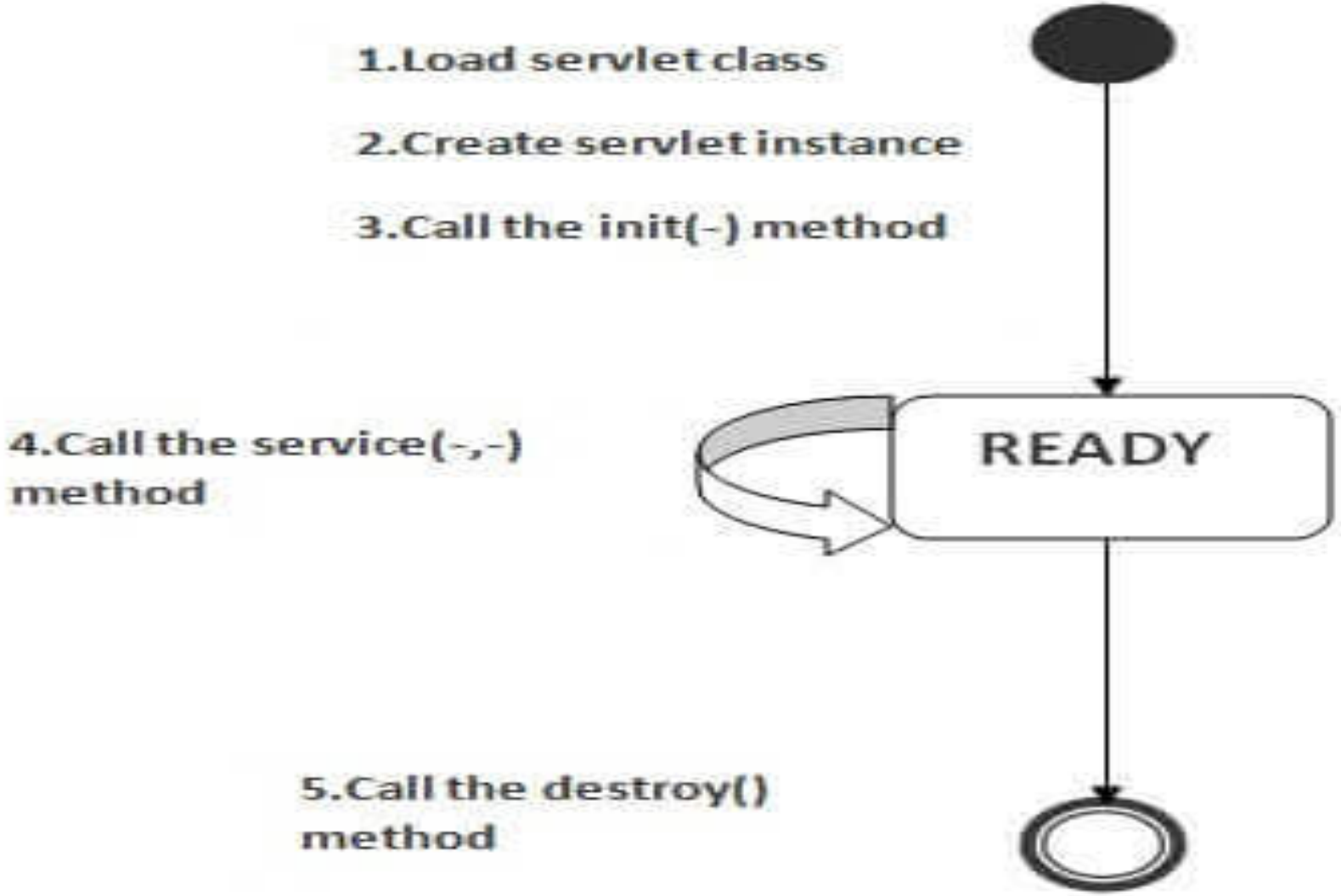
Methods of HttpServlet class

1. **public void service(ServletRequest req,ServletResponse res)** dispatches the request to the protected service method by converting the request and response object into http type.
2. **protected void service(HttpServletRequest req, HttpServletResponse res)** receives the request from the service method, and dispatches the request to the doXXX() method depending on the incoming http request type.
3. **protected void doGet(HttpServletRequest req, HttpServletResponse res)** handles the GET request. It is invoked by the web container.
4. **protected void doPost(HttpServletRequest req, HttpServletResponse res)** handles the POST request. It is invoked by the web container.

Life Cycle of a Servlet (Servlet Life Cycle)

The web container maintains the life cycle of a servlet instance. Let's see the life cycle of the servlet:

1. Servlet class is loaded.
2. Servlet instance is created.
3. init method is invoked.
4. service method is invoked.
5. destroy method is invoked.


1.Load servlet class
2.Create servlet instance
3.Call the init(-) method
READY
4.Call the service(-,-) method
5.Call the destroy() method

Life Cycle of a Servlet (Servlet Life Cycle)

- As displayed in the above diagram, there are three states of a servlet: new, ready and end. The servlet is in new state if servlet instance is created. After invoking the init() method, Servlet comes in the ready state. In the ready state, servlet performs all the tasks. When the web container invokes the destroy() method, it shifts to the end state.

1) Servlet class is loaded

- The class loader is responsible to load the servlet class. The servlet class is loaded when the first request for the servlet is received by the web container.

2) Servlet instance is created

- The web container creates the instance of a servlet after loading the servlet class. The servlet instance is created only once in the servlet life cycle.

3) init method is invoked

- The web container calls the init method only once after creating the servlet instance. The init method is used to initialize the servlet. It is the life cycle method of the javax.servlet.Servlet interface. Syntax of the init method is given below:
- **public void** init(ServletConfig config) **throws** ServletException

4) service method is invoked

- The web container calls the service method each time when request for the servlet is received. If servlet is not initialized, it follows the first three steps as described above then calls the service method. If servlet is initialized, it calls the service method. Notice that servlet is initialized only once. The syntax of the service method of the Servlet interface is given below:
- public void service(ServletRequest request, ServletResponse response) throws ServletException, IOException

5) destroy method is invoked

- The web container calls the destroy method before removing the servlet instance from the service. It gives the servlet an opportunity to clean up any resource for example memory, thread etc. The syntax of the destroy method of the Servlet interface is given below:
- **public void** destroy()

THANK YOU