

✔ **PART I: Short Questions (Exercise Answers)**

Q1. What is a Servlet?

- A **Servlet** is a **Java program** that runs on a web server.
- It handles **client requests** (HTTP) and generates **dynamic responses** (HTML, JSON, XML).

Q2. Write any two advantages of Servlet.

1. Platform independent (runs on any OS with JVM).
2. More efficient and secure than CGI (Common Gateway Interface).

Q3. Define a Generic Servlet.

- A **Generic Servlet** is a class that extends `javax.servlet.GenericServlet`.
- It provides a framework for handling any type of request (not just HTTP).

Q4. What is an HTTP Servlet? Explain with diagram.

- An **HTTP Servlet** is a servlet class that extends `HttpServlet`.
- It is specifically designed to handle **HTTP requests** (GET, POST).
- Example: A **login servlet** that processes username & password.

Diagram (Textual):

Browser (Client) ---> HTTP Request ---> Servlet Container ---> HttpServlet

Browser (Client) <--- HTTP Response <--- Servlet Container <--- HttpServlet

Q5. Enlist steps of Writing Basic Hello World Servlet.

1. Create a new servlet class extending `HttpServlet`.
2. Override `doGet()` or `service()` method.
3. Write response using `PrintWriter`.
4. Compile and deploy in **WEB-INF/classes**.
5. Configure in **web.xml**.
6. Run on server (Tomcat).

Example:

```
public class HelloServlet extends HttpServlet {  
    public void doGet(HttpServletRequest req, HttpServletResponse res)  
        throws IOException {  
        res.setContentType("text/html");  
        PrintWriter out = res.getWriter();  
        out.println("<h1>Hello World Servlet</h1>");  
    }  
}
```

Q6. What are the processes of Servlet Life Cycle?

1. **Loading & Instantiation**

Chapter # 5

2. **HTTP Servlet** → Extends HttpServlet, designed for HTTP requests (doGet(), doPost()).

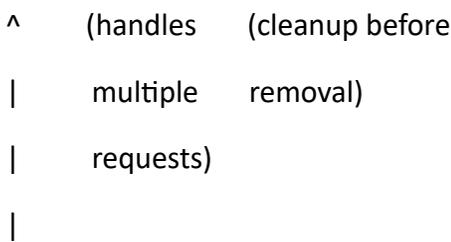
Q3. Discuss the Servlet Life Cycle in detail with its Architectural Diagram.

• **Phases:**

1. **Loading & Instantiation** → Servlet loaded in memory.
2. **Initialization (init())** → Called once at startup.
3. **Request Handling (service())** → Called for each request.
4. **Destruction (destroy())** → Called before unloading.

Diagram (Textual):

init() --> service() --> destroy()



Servlet Container

★ **Extra Important Short Questions with Answers**

Q1. Differentiate between Servlet and JSP.

- **Servlet:** Java code inside Java class.
- **JSP:** Java code inside HTML.

Q2. What is Servlet Container?

- A part of web server (e.g., Tomcat) that manages servlet lifecycle and communication.

Q3. What is the difference between doGet() and doPost()?

- **doGet():** Sends data in URL, less secure, limited length.
- **doPost():** Sends data in request body, more secure.

Q4. What is web.xml in servlet?

- Deployment descriptor that defines servlet mapping and configuration.

Q5. What is RequestDispatcher in Servlet?

- Used to forward a request from one servlet to another or include content.

Q6. What is Session Management in Servlet?

- A way to track user data across multiple requests (e.g., login session).

★ **Extra Important Long Questions with Answers**

Q1. Differentiate between Servlet and CGI.

Feature	Servlet	CGI
Language	Java	Multiple (C, Perl)
Performance	Fast, multithreaded	Slow, new process for each request
Portability	Platform independent	Less portable

Q2. Explain how Servlets handle Form Data with Example.

- Form data sent via HTML form → Servlet reads using `request.getParameter("field")`.

Example:

```
<form action="Register" method="post">
```

```
  Name: <input type="text" name="username">
```

```
  <input type="submit">
```

```
</form>
```

```
public void doPost(HttpServletRequest req, HttpServletResponse res) throws IOException {
```

```
  String name = req.getParameter("username");
```

```
  PrintWriter out = res.getWriter();
```

```
  out.println("Welcome " + name);
```

```
}
```

Q3. Explain Form Validation using Servlet.

- Servlet checks if fields are empty or invalid.
- Example: Rejects empty email or wrong format.

```
if(email == null || !email.contains("@")) {
```

```
  out.println("Invalid Email!");
```

```
}
```