Complete Servlet Overview

# 1. Introduction to Servlets

Servlets are Java-based server-side programs used to handle requests and generate dynamic web content. They operate inside a servlet container (like Tomcat) and support multithreading, ensuring scalability and efficiency.

# 2. Servlet Lifecycle Methods

1. init(): Initializes the servlet when it is loaded by the container.  
2. service(): Handles client requests (e.g., GET or POST) and generates responses.  
3. destroy(): Releases resources before the servlet is unloaded from memory.  
Non-lifecycle Methods:  
- getServletConfig(): Retrieves servlet-specific configuration.  
- getServletInfo(): Provides metadata about the servlet (like version and author).

# 3. Web Application Architecture

A web application runs on a web server and is accessed via a browser. It combines server-side scripts (like Servlets) with client-side technologies (like JavaScript) to deliver a dynamic and interactive user experience. These applications provide features like authentication, session management, and database integration.

# 4. HTTP Methods - GET vs POST

GET: Retrieves data, sends parameters in the URL, and is idempotent (safe for caching).  
POST: Sends data in the request body, useful for form submissions, and may change the server state.

# 5. ServletConfig vs ServletContext

- ServletConfig: Holds initialization parameters specific to a servlet (e.g., database URL).  
- ServletContext: Shared across all servlets, providing application-wide settings (e.g., app name).

# 6. Session Management Techniques

State Management:  
- Cookies: Store small data on the client’s browser.  
- Sessions: Maintain user-specific data on the server.  
- Hidden Form Fields: Embed temporary data inside HTML forms.  
- URL Rewriting: Append session IDs to URLs when cookies are disabled.

# 7. Servlet Filters and Listeners

Filters:  
- Authentication: Verify users before accessing secure areas.  
- Compression: Reduce response size to improve performance.  
- Logging: Record request/response details.  
- CSRF Protection: Prevent unauthorized cross-site requests.  
Listeners:  
- ServletContextListener: Monitors web application events.  
- HttpSessionListener: Tracks session lifecycle events.

# 8. Servlet Container Responsibilities

The servlet container manages the lifecycle of servlets, handles resource pooling, and creates a new thread for each request. It also converts JSP pages into servlets and provides memory optimizations.

# 9. Types of Servlet Containers

- Standalone: Tomcat running as a self-contained server.  
- In-process: Tomcat embedded in a larger server (e.g., JBoss).  
- Out-of-process: Apache and Tomcat running separately.

# 10. WAR Files (Web Archive)

WAR files package the entire web project (HTML, CSS, JSP, servlets) for easy deployment on a server.

# 11. MIME Types

MIME types tell the browser what type of content to expect (e.g., 'text/html' or 'application/json').

# 12. RequestDispatcher vs sendRedirect

- RequestDispatcher: Forwards requests without changing the browser’s URL.  
- sendRedirect: Redirects to a new URL, changing the browser’s address bar.

# 13. Authentication Methods

- Basic Authentication: Credentials are sent encoded in Base64.  
- Digest Authentication: Uses a hashed value with a nonce.  
- HTTPS Authentication: Secures communication with SSL/TLS.  
- Form-based Login: Users submit credentials via an HTML form.

# 14. Key Servlet Methods

- request.getRemoteAddr(): Retrieves the client’s IP address.  
- getServletContext().getServerInfo(): Provides server information.  
- getServletContext().getRealPath(): Gets the real path of the servlet on the server.

# 15. Servlet 3.0 Features

- File Uploads  
- Asynchronous Processing  
- Annotations-based Configuration  
- Declarative Security

# 16. PrintWriter vs ServletOutputStream

- PrintWriter: For writing text-based content like HTML or JSON.  
- ServletOutputStream: For writing binary data like images or files.

# 17. Deployment Descriptor (web.xml)

Defines servlet mappings, filters, listeners, and error handlers. With Servlet 3.0, annotations can be used to reduce the reliance on web.xml.

# 18. URL Encoding

- encodeURL(): Encodes URLs within the same response.  
- encodeRedirectURL(): Encodes URLs used in redirects.

# 19. Interview Tips and Tricks

- Understand how sessions are used for state management and how to secure them.  
- Know the differences between JSP and Servlets.  
- Be familiar with filters and their role in enhancing security and performance.