Servlet API

Servlet API:

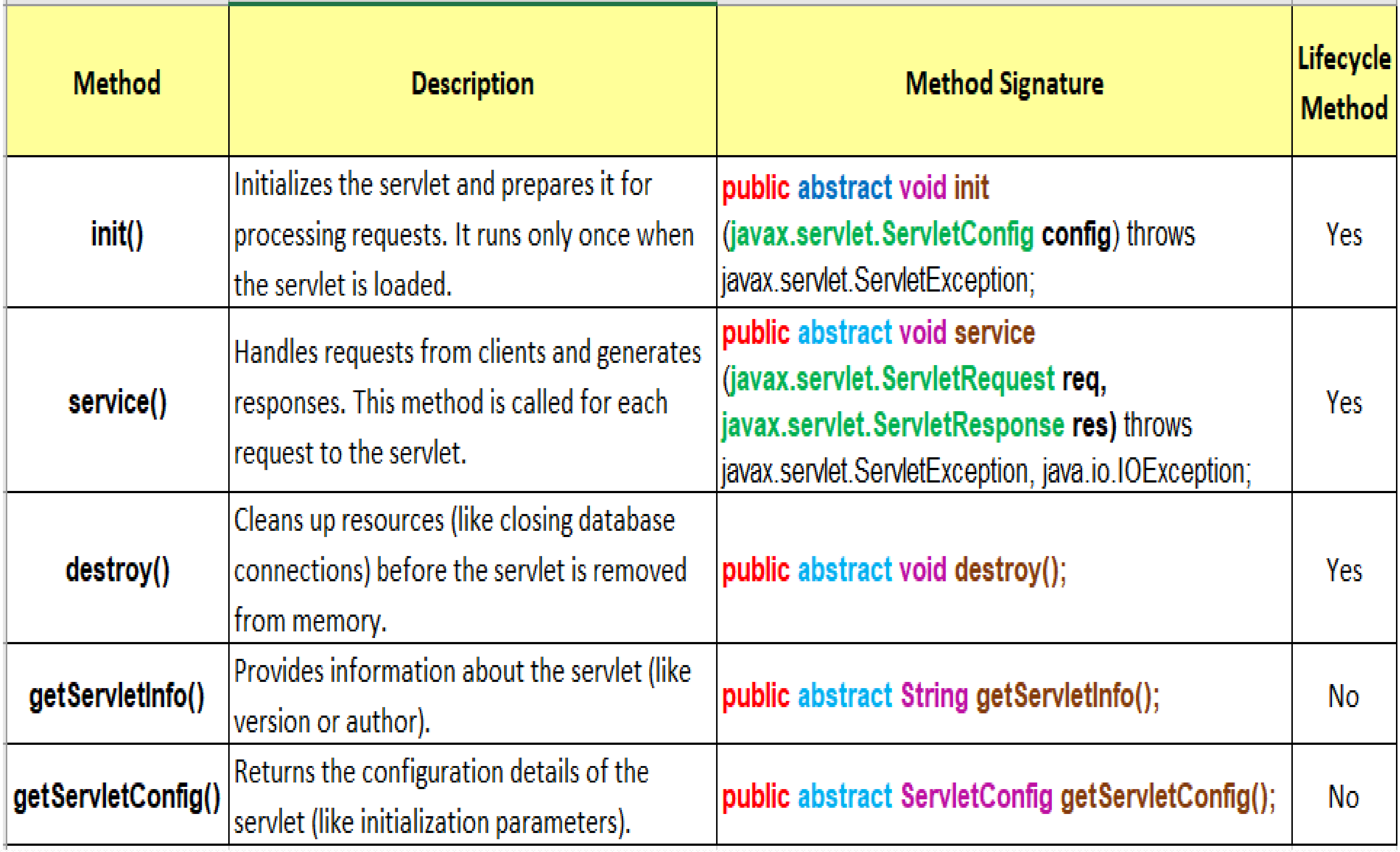
- The Servlet API is part of the Java EE platform , and it's defined in the ***javax.servlet*** package.
- By using this Servlet API we can write web applications that handle HTTP requests and generate responses.
- ***javax.servlet.Servlet*** interface is a root of Servlet-Programming or Servlet-API.
- A Servlet is a Java program that extends the functionality of a server, by responding to requests from web clients (browsers) over HTTP

Servlet Interface:

- The javax.servlet.Servlet interface is the foundation of the Servlet API.
- Servlet interface has '5' important methods.

Servlet Interface Methods

1. init() – Initialization
2. service() – Processing Requests and Sending Responses
3. destroy() – Cleanup
4. getServletInfo() – Retrieving Information About the Servlet
5. getServletConfig() – Accessing Configuration Detail

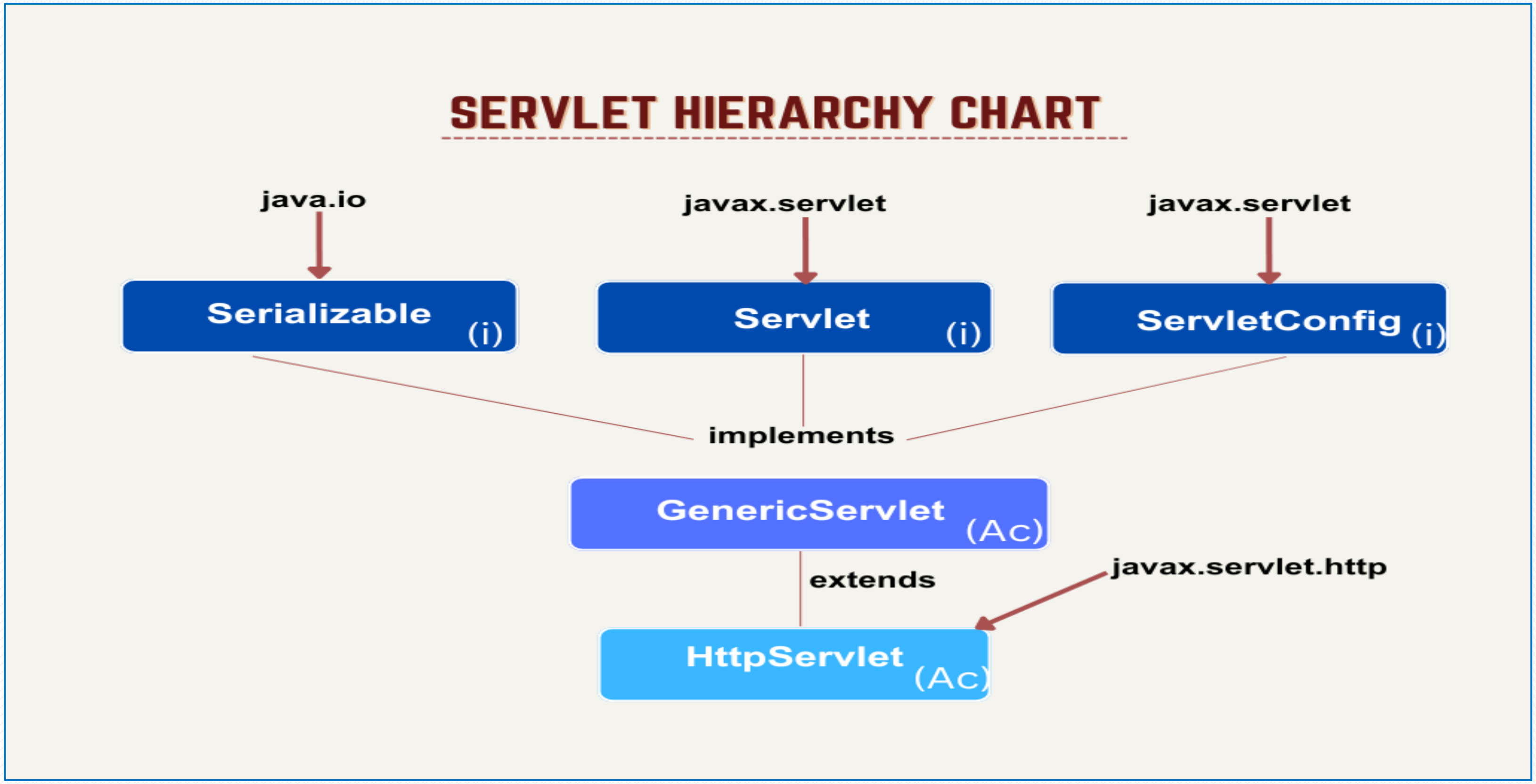


Method	Description	Method Signature	Lifecycle Method
init()	Initializes the servlet and prepares it for processing requests. It runs only once when the servlet is loaded.	public abstract void init (javax.servlet.ServletConfig **config**) throws javax.servlet.ServletException;	Yes
service()	Handles requests from clients and generates responses. This method is called for each request to the servlet.	public abstract void service (javax.servlet.ServletRequest **req,** javax.servlet.ServletResponse **res)** throws javax.servlet.ServletException, java.io.IOException;	Yes
destroy()	Cleans up resources (like closing database connections) before the servlet is removed from memory.	public abstract void destroy();	Yes
getServletInfo()	Provides information about the servlet (like version or author).	public abstract String getServletInfo();	No
getServletConfig()	Returns the configuration details of the servlet (like initialization parameters).	public abstract ServletConfig getServletConfig();	No

Lifecycle Methods: init(), service(), and destroy() are called automatically by the servlet container in the same order & follow the servlet's lifecycle (initialization, servicing requests, and destruction).

Servlet Hierarchy

- The hierarchy of the Servlet interface shows the relationships between the core interfaces and classes that implement or extend the Servlet interface in the Java Servlet API.
- Here's a simple representation of the servlet hierarchy



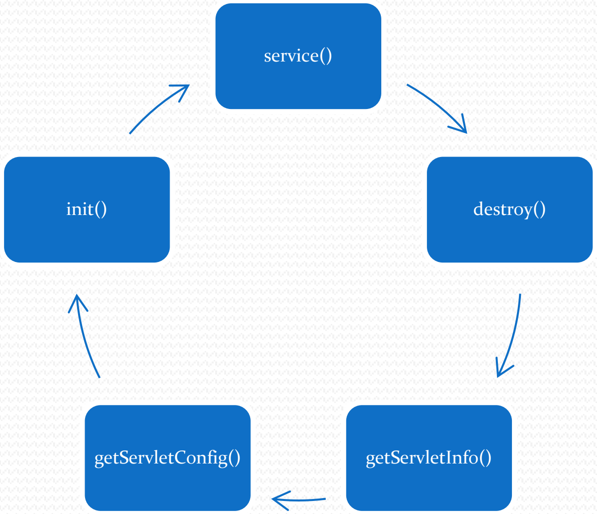
Servlet Programming Models

When constructing a servlet program, you can choose one of the following approaches:

1. Implementing the Servlet Interface
2. Extending the GenericServlet Abstract Class
3. **Extending the HttpServlet Abstract Class**

1) Implementing the Servlet Interface

If we created servlet program by implementing ***Servlet interface*** directly, we must provide implementations for all five abstract methods of the Servlet interface



2) Extending the GenericServlet Abstract Class:

If we created servlet program by extending the **GenericServlet** abstract Class, we need to implement only service(). The other lifecycle methods like init() and destroy() have default implementations, so they are optional.

3) Extending the HttpServlet Abstract Class:

If we created servlet program by extending the **HttpServlet** abstract Class, all methods are optional. We can override only the methods which we need.

- ➢ doGet() – Handles HTTP GET requests.
- ➢ doPost() – Handles HTTP POST requests.
- ➢ init() – Initialization (optional)
- ➢ destroy() – Cleanup (optional) ….etc