Web Development with Jakarta Server Pages and Servlets

Session: 3

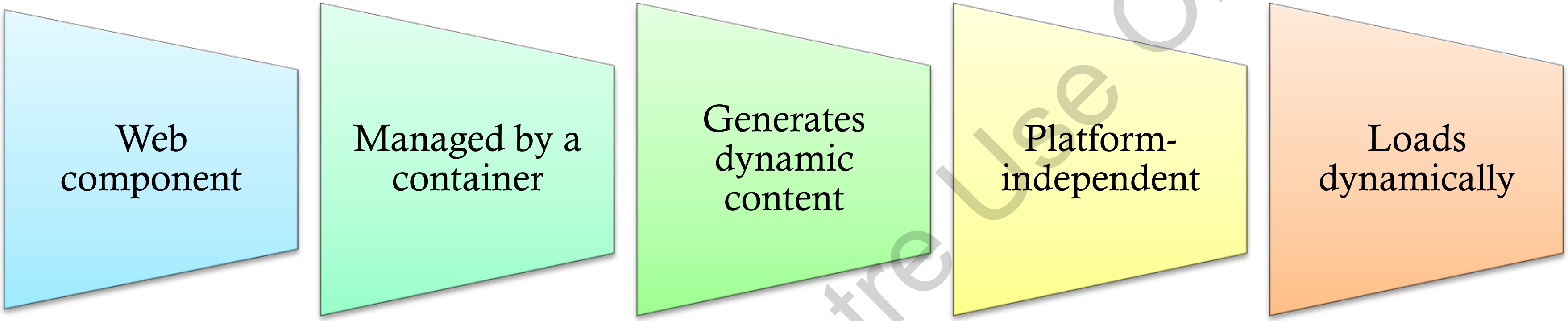
Jakarta Servlets

Objectives

- Explain the Jakarta Servlet API
- Explain the servlet architecture and lifecycle of Servlet
- Describe the servlet mapping specifications
- Explain the methods of ServletRequest and HttpServletRequest interfaces
- Explain the methods of ServletResponse and HttpServletResponse interfaces
- Describe the use of response headers
- Explain how to read text and binary data from a request
- Explain the ServletConfig and ServletContext interface
- Explain Servlet context Attribute
- Explain redirection of client requests
- Explain RequestDispatcher interface
- Explain error handling in Servlet
- Explain uploading files with Jakarta Servlet Technology

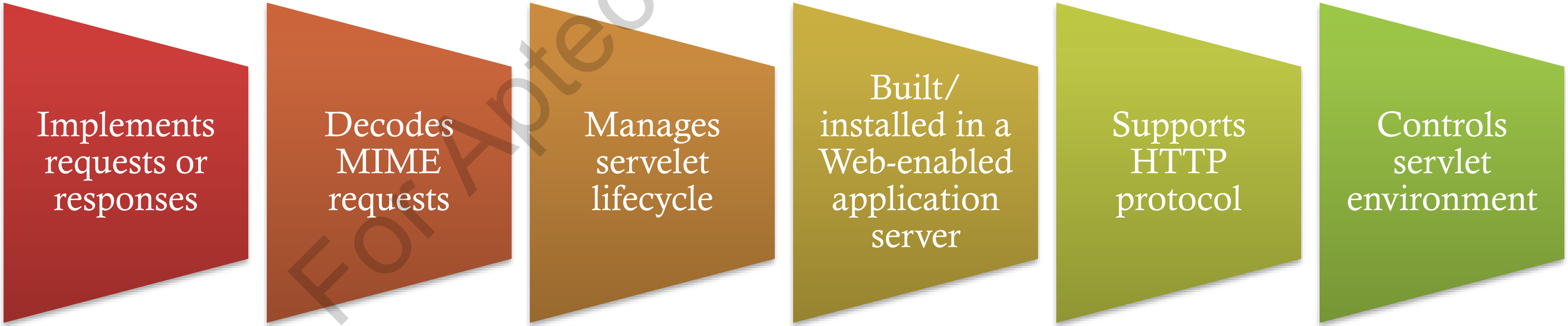
Introduction

Servlet



- Web component
- Managed by a container
- Generates dynamic content
- Platform-independent
- Loads dynamically

Servlet Container



- Implements requests or responses
- Decodes MIME requests
- Manages servelet lifecycle
- Built/ installed in a Web-enabled application server
- Supports HTTP protocol
- Controls servlet environment

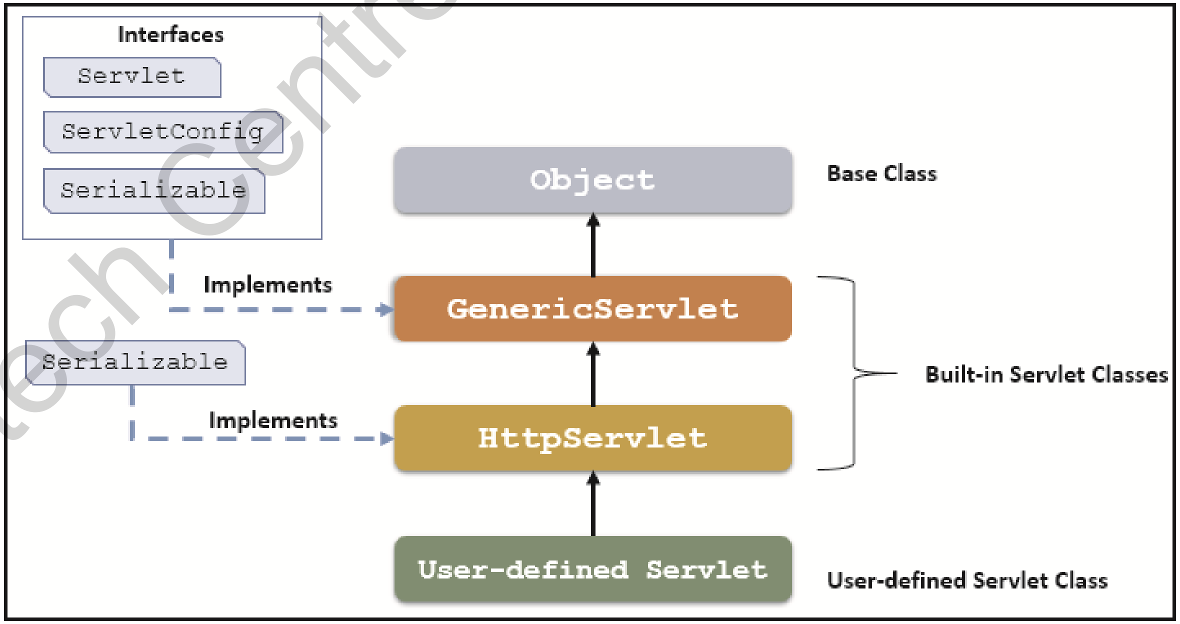
Servlet API

Packages containing classes related to developing and managing Servlets

- `jakarta.servlet`
- `jakarta.servlet.http`

The `GenericServlet` implements three interfaces

- `Servlet`
- `ServletConfig`
- `Serializable`

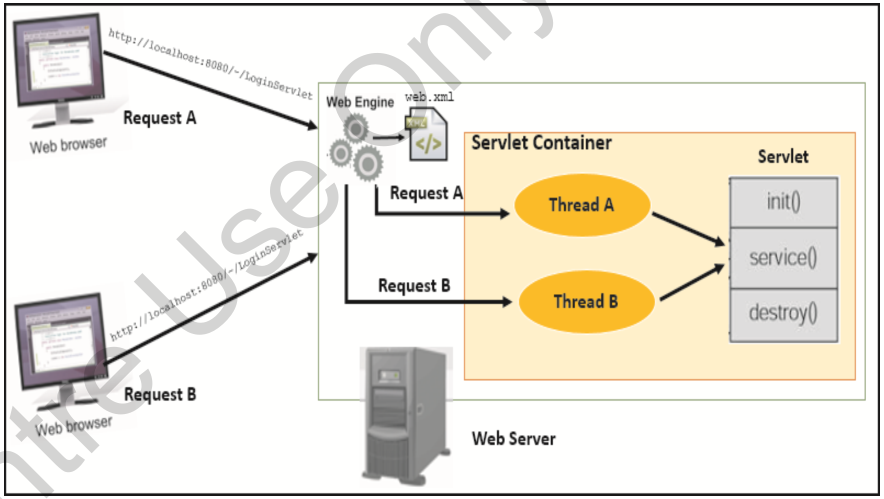


Servlet Hierarchy

Servlet Lifecycle

Servlet Lifecycle

1. Instantiation
2. Initialization
3. Service
4. Destroy
5. Unavailable



Servlet Multithreaded Model

init()
service ()
destroy ()

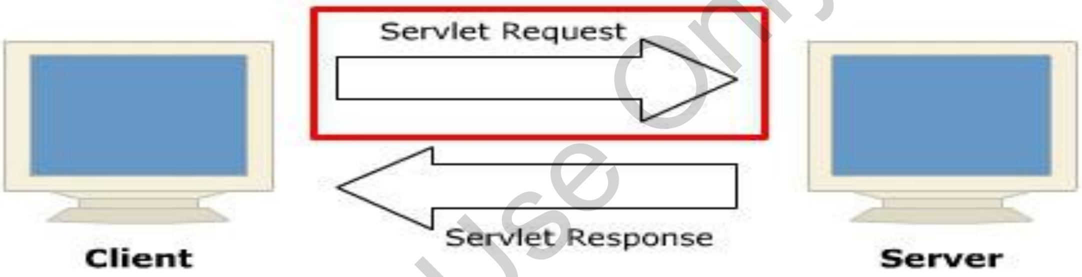
Servlet Lifecycle Methods

- doGet()
- doPost()
- doPut()
- doDelete()
- doHead()
- doOptions()
- doTrace()

HTTP Specific Request Handling Methods

Handling Servlet Request

Servlet Request



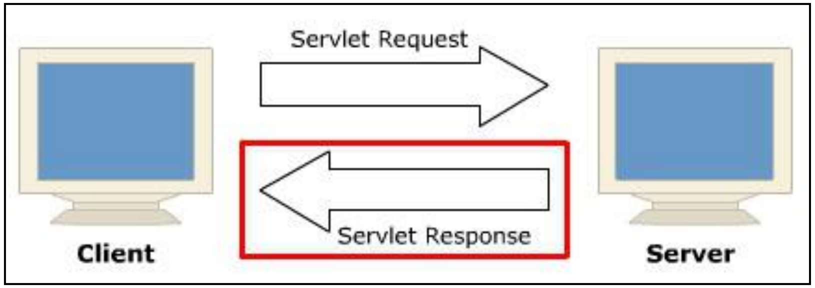
- Accept
- Accept-Charset
- Accept-Encoding
- Accept-Language
- Authorization

HTTP Request Headers

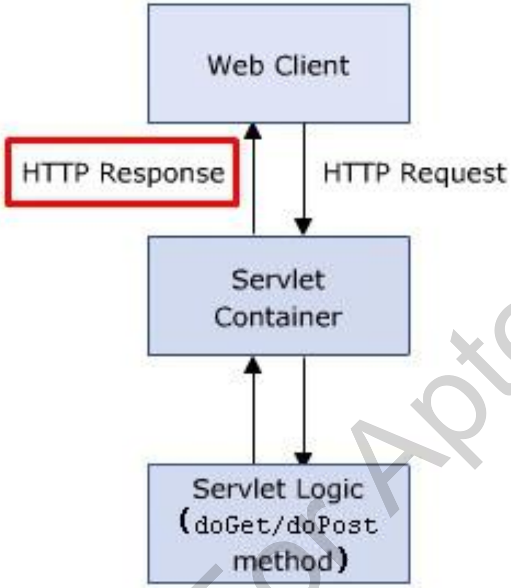
HTML Forms

- Text input
- Drop-down list input
- Submit button

Handling Servlet Response



Servlet Response



HTTP Servlet Response

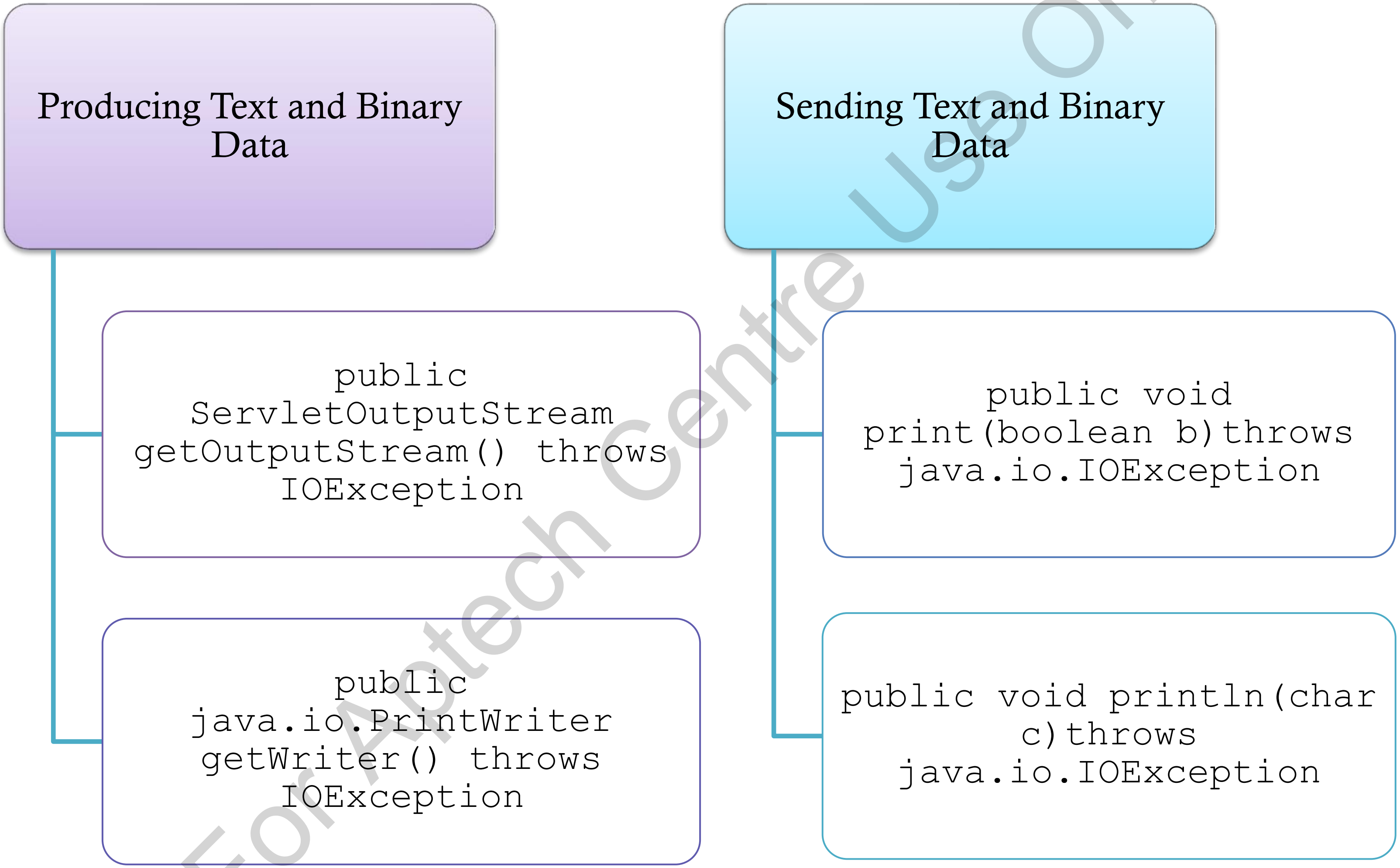
Response Headers

Contains data about the server and file date, size, and type that the server sends back to the client.

Sending Headers

- `addHeader()`
- `addDateHeader()`
- `addIntHeader()`
- `containsHeader()`

Reading Binary Data



Producing Text and Binary Data

```
public ServletOutputStream getOutputStream() throws IOException
```

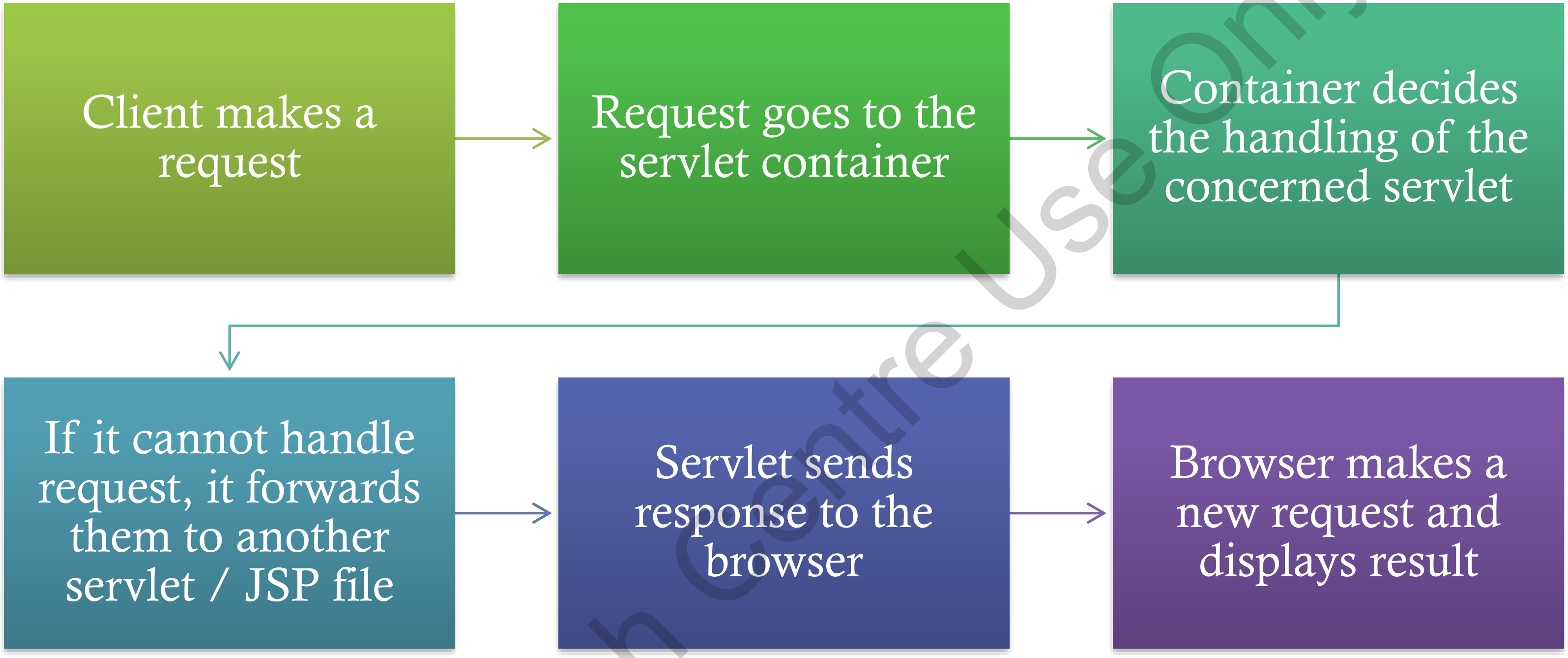
```
public java.io.PrintWriter getWriter() throws IOException
```

Sending Text and Binary Data

```
public void print(boolean b)throws java.io.IOException
```

```
public void println(char c)throws java.io.IOException
```

Redirecting Requests



Client makes a request

Request goes to the servlet container

Container decides the handling of the concerned servlet

If it cannot handle request, it forwards them to another servlet / JSP file

Servlet sends response to the browser

Browser makes a new request and displays result

Servlets of same application use these methods to share data:

forward()

include()

Initializing Servlets

Method	Description
• `public String getInitParameter(String name)`	Returns parameter value for the specified parameter name.
• `public Enumeration getInitParameterNames()`	Returns names of the initialization parameters of the context.
• `public void setAttribute(String name, Object object)`	Sets given object in the application scope.
• `public Object getAttribute(String name)`	Returns attribute for the specified name.
• `public Enumeration getInitParameterNames()`	Returns names of the context's initialization parameters as an Enumeration of String objects.
• `public void removeAttribute(String name)`	Removes attribute with the given name from the servlet context.

Commonly Used Methods of ServletContext

Error Handling in Servlets

Status Code	Associated Message	Meaning
301	Moved Permanently	Document moved to separate location as mentioned in URL.
302	Found	Temporary replacement of file from one location to the other as specified.
400	Bad Request	The request placed is syntactically incorrect.
401	Unauthorized	Authorization is not given to access a password-protected page.
404	Not Found	Resource not found in the specified address.
408	Request Timeout	Time taken by client is very long to send the request.
500	Internal Server Error	Server is unable to locate the requested file due to servlet deletion or crash.

Status Codes

File Upload

When data is sent as form data or as multipart, files can be uploaded using these methods:

```
public Collection<Part> getParts()
```

```
public Part getPart(String name)
```

Summary (1-2)

- A servlet is a Web component based on Jakarta EE (formerly Java EE) technology that is managed by a container and is responsible for generating dynamic content.
- The init(), service(), and destroy() methods are a servlet's lifecycle methods.
- A GenericServlet class defines a servlet that is not protocol dependent. To have better control over required servlets, HttpServlet is extended to GenericServlet.
- A servlet request contains the data to be passed between a client and the servlet. All requests implement the ServletRequest interface, which defines the methods for accessing the relevant information.
- A servlet response contains data to be passed between a server and the client. All responses implement the ServletResponse interface. This interface defines various methods to process response.

Summary (2-2)

- You can use getInputStream() to retrieve the input stream as a ServletInputStream object. ServletOutputStream provides an output stream for sending binary data to the client.

- Resources are included to a servlet by forwarding request from one servlet to another by using the forward() and include() methods along with RequestDispatcher interface. Inter-servlet communication can be used by the servlets to gain access to other currently loaded servlets and perform some tasks on other servlets.

- Errors in Web applications with servlets are reported using sendError() and setStatus() methods. Errors are logged using the log() method of ServletContext and forwarded to an error page using the RequestDispatcher interface.

- A RequestDispatcher object is used to redirect the client request to a Web page on receiving an error message on the server.