HUMBER

J2EE Web Component

Week 2

Lecturer: Shahdad Shariatmadari

September 2019

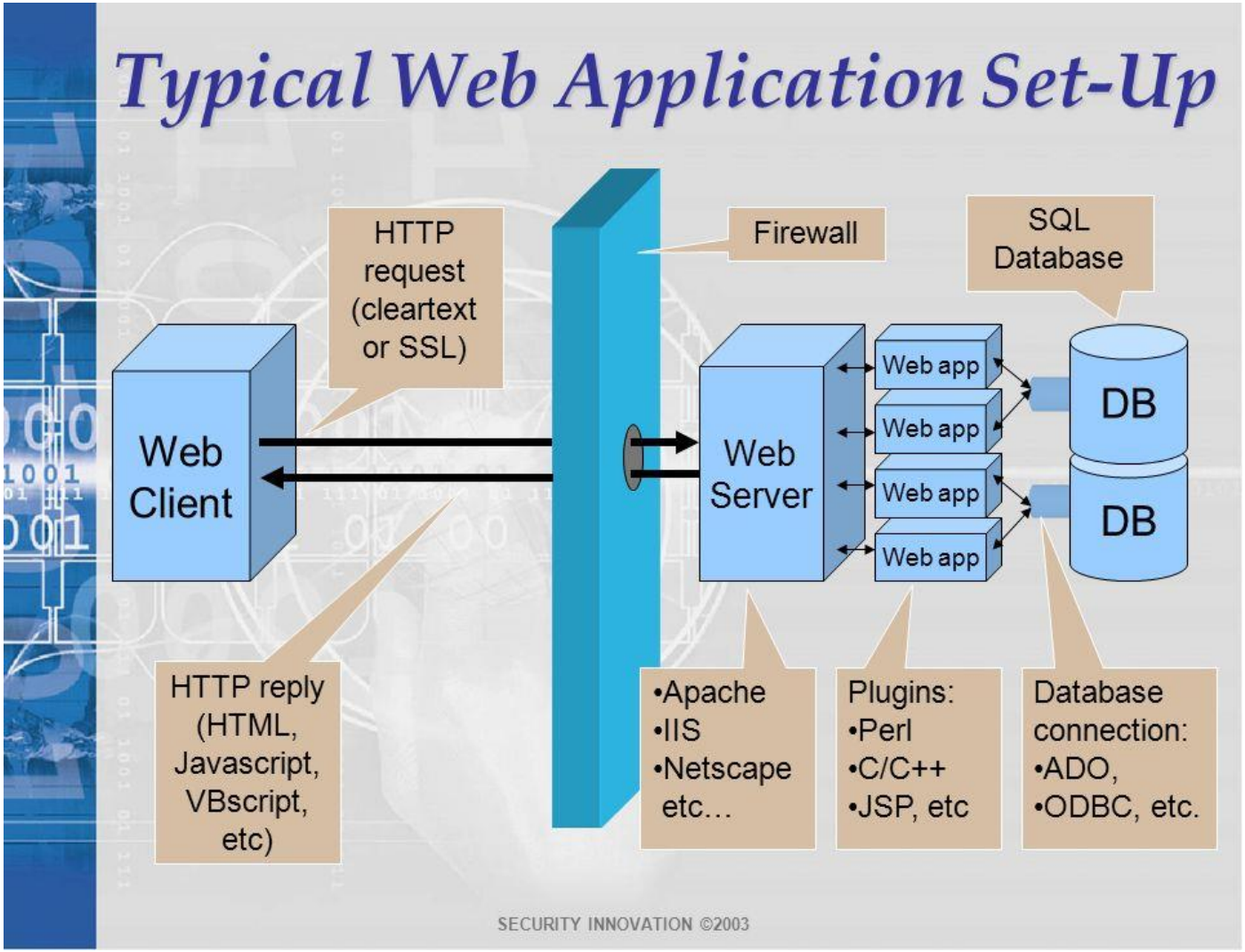
What we have learned…

- Web Application
- Dynamic Web development
- J2EE Architecture

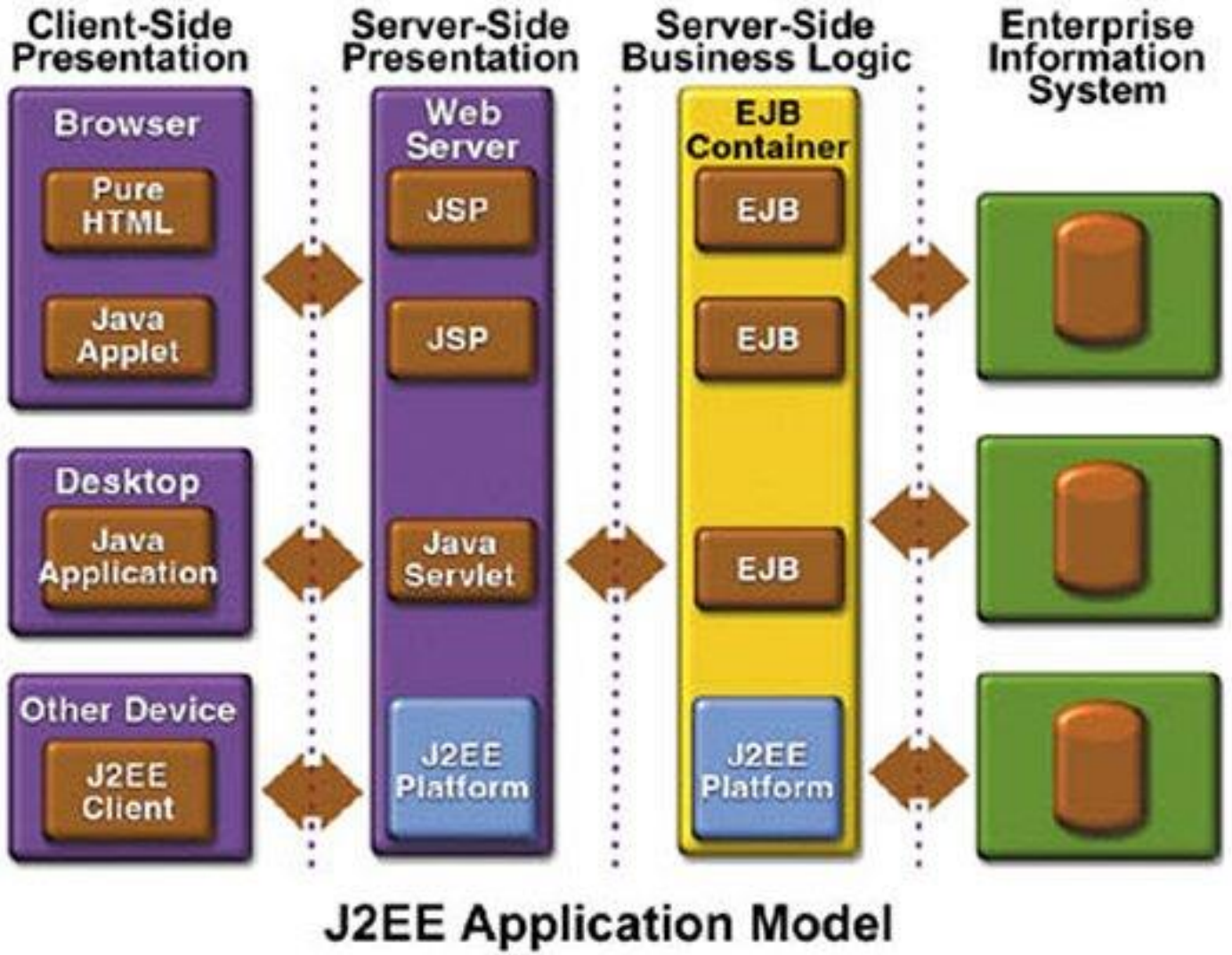
Agenda

- The basic structure of servlets
- A simple servlet that generates plain text
- A servlet that generates HTML
- Servlets and packages
- Some utilities that help build HTML
- The servlet life cycle

Web Application



J2EE Application Model

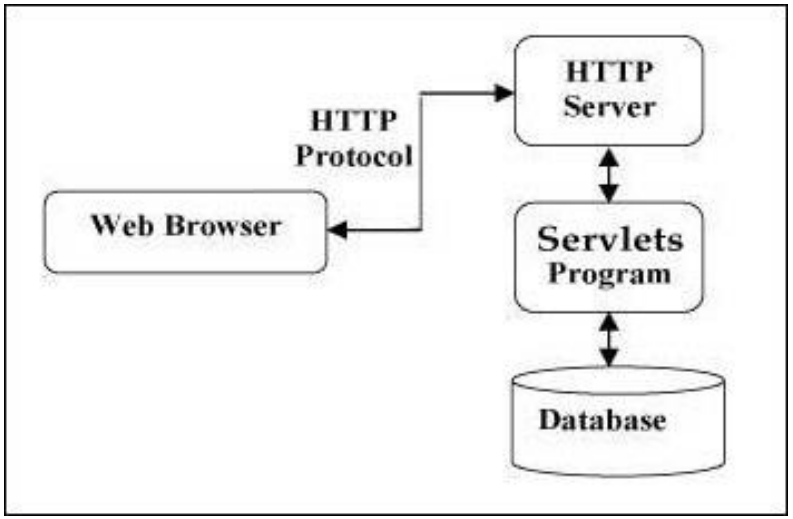


Web components

- Component
  - An application-level software unit
- A web component is a software unit that provides a response to a request
- Act as the user interface for a web-based application
- Java EE platform specifies few Web components
  - Servlets
  - Java Server Pages (JSP)

Servlet

- Java Servlets are programs that run on a Web or Application server and act as a middle layer between a requests coming from a Web browser or other HTTP client and databases or applications on the HTTP server.

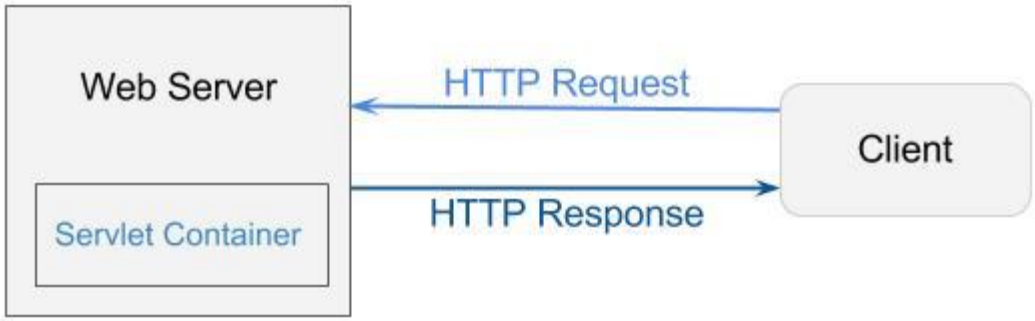


Web Container

- A web container ,also known as a servlet container is the component of a web server that interacts with Java servlets.
- The Web container creates servlet instances, loads and unloads servlets, creates and manages request and response objects, and performs other servlet-management tasks.
- A web container implements the **web component** contract of the **Java EE architecture**, specifying a runtime environment for web components that includes **security, concurrency, lifecycle management, transaction, deployment**, and other services.

Servlet Container

- The basic idea of Servlet container is using Java to dynamically generate the web page on the server side.
- So servlet container is essentially a part of a web server that interacts with the servlets.



Servlet Container

- **Catalina** is **Tomcat's** servlet container.
- **Catalina** implements Sun Microsystems's specifications for servlet and JavaServer Pages (JSP).
- Question: Can you find how to set/start Catalina in the Tomcat?

Popular servlet engines

- Tomcat: jakarta.apache.org (free)
- JRun: www.livesoftware.com
- Jetty: www.mortbay.com (free)
- J2EE application servers: IBM WebSphere, BEA WebLogic, Orion

Servlets Packages

- Java Servlets are Java classes run by a web server that has an interpreter that supports the Java Servlet specification.
- Servlets can be created using the **javax.servlet** and **javax.servlet.http** packages, which are a standard part of the Java's enterprise edition.
- In practice, all servlets extend the HttpServlet class. To extend this class, the servlet must import some of the classes in the java.io, javax.servlet, and javax.servlet.http packages.

Creating a Servlet

- There are two steps to creating a servlet.
  - You must **code the class** for the servlet, and you must **map that class to a URL**.
- Prior to the servlet 3.0 specification (Tomcat 7.0), you had to use the web.xml to map a servlet to a URL.
- With the servlet 3.0 specification and later, you can use the @WebServlet annotation to map a servlet to one or more URL patterns.

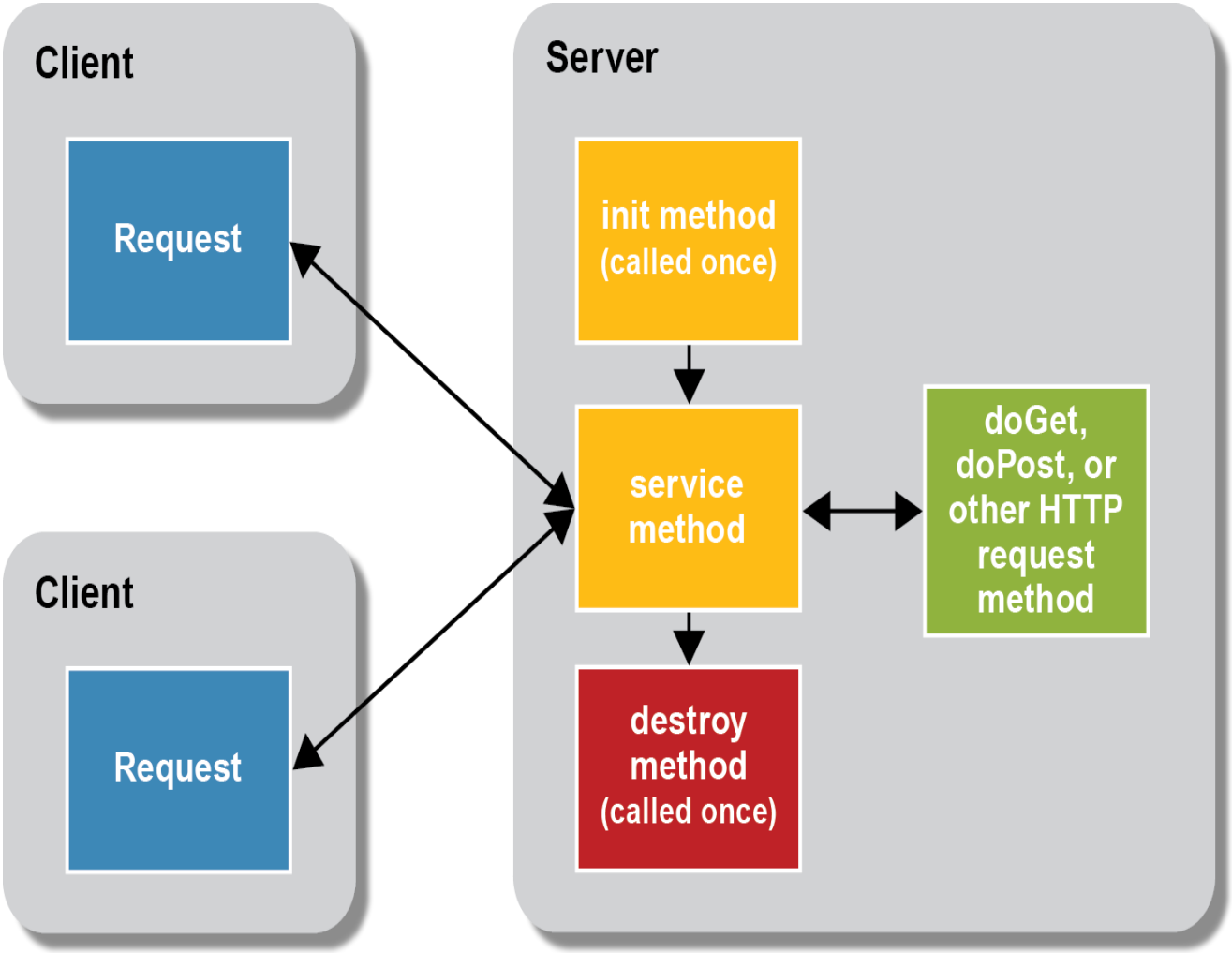
Sample Servlet

```
// Import required java libraries
import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;

// Extend HttpServlet class
public class HelloWorld extends HttpServlet {
  private String message;
  public void init() throws ServletException {
    // Do required initialization
    message = "Hello World";
  }

  public void doGet(HttpServletRequest request, HttpServletResponse response)
    throws ServletException, IOException {

    // Set response content type
    response.setContentType("text/html");

    // Actual logic goes here.
    PrintWriter out = response.getWriter();
    out.println("<h1>" + message + "</h1>");
  }

  public void destroy() {
    // do nothing.
  }
}
```

Servlets - Life Cycle

- A servlet life cycle can be defined as the entire process from its creation till the destruction. The following are the paths followed by a servlet.
  - The servlet is initialized by calling the **init()** method.
  - The servlet calls **service()** method to process a client's request.
  - The servlet is terminated by calling the **destroy()** method.
  - Finally, servlet is garbage collected by the garbage collector of the JVM.

The lifecycle of a servlet



The init() Method

- The init method is called only once, when the servlet is created.
- The servlet is normally created when a user first invokes a URL corresponding to the servlet, but you can also specify that the servlet be loaded when the server is first started.

The init() Method

- The init() method simply creates or loads some data that will be used throughout the life of the servlet.
- `public void init() throws ServletException {`
  - `// Initialization code...`
- `}`

The service() Method

- The service() method is the main method to perform the actual task.
- The **servlet container** calls the service() method to handle requests coming from the client(browsers) and to write the formatted response back to the client.

The service() Method

- Each time the server receives a request for a servlet, the server spawns a new thread and calls service.
- The service() method checks the HTTP request type (GET, POST, PUT, DELETE, etc.) and calls doGet, doPost, doPut, doDelete, etc. methods as appropriate.
- The doGet() and doPost() are most frequently used methods with in each service request.

doGet and doPost

- public void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
  - // Servlet code
- }
- public void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
  - // Servlet code
- }

Servlet concepts

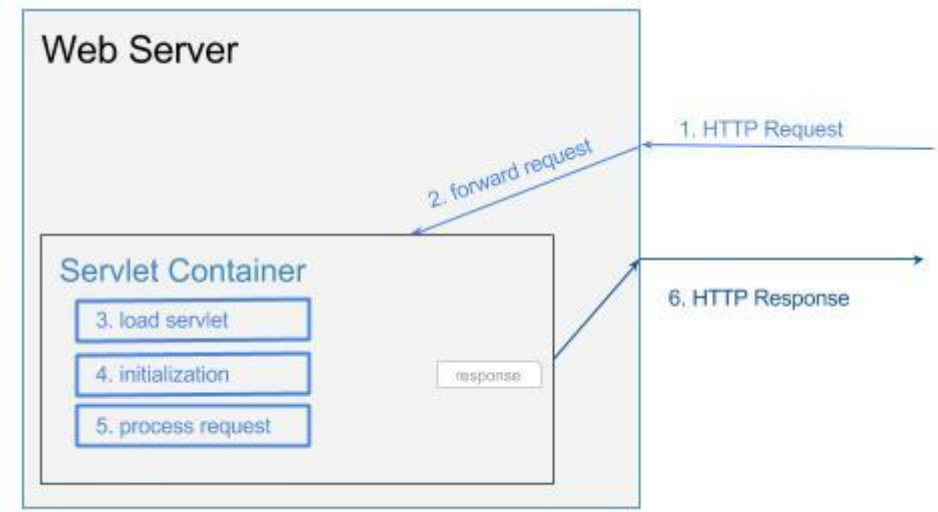
- The doGet method processes all HTTP requests that use the GET method.
- The doPost method processes all HTTP requests that use the POST method.
- The doGet and doPost methods both accept (1) the HttpServletRequest object, or the *request object*, and (2) the HttpServletResponse object, or the *response object*.

The destroy() Method

- The destroy() method is called only once at the end of the life cycle of a servlet.
- This method gives your servlet a chance to close database connections, halt background threads, write cookie lists or hit counts to disk, and perform other such cleanup activities
- After the destroy() method is called, the servlet object is marked for garbage collection.
  - public void destroy() { // Finalization code... }

How Servlet container and web server process a request?

1. Web server receives HTTP request
2. Web server forwards the request to servlet container
3. The servlet is dynamically retrieved and loaded into the address space of the container, if it is not in the container.
4. The container invokes the init() method of the servlet for initialization(invoked once when the servlet is loaded first time)
5. The container invokes the service() method of the servlet to process the HTTP request, i.e., read data in the request and formulate a response. The servlet remains in the container's address space and can process other HTTP requests.
6. Web server return the dynamically generated results to the correct location



Activity 1

- Open Blackboard->Learning Material -> Module 2
- Download the sample-Servlet-1
- Add new servlet (name it Servlet2) to the java project
- Copy the given code to the servlet2.

Activity 1… cont

- Add the init() method as follow:
- **public void init() throws ServletException {**
  - System.***out.println("Init method");***
- }
- Run the project and see the result (in colsole)
- Develop the destroy() method. Run the project and try to get the output from thedtroy() method (same as init())

PrintWriter

- PrintWriter out = response.getWriter();
  - **PrintWriter** is an abstract class for writing to character streams. Print formatted representations of objects to a text-output ,HTML, XML etc stream.
  - we then call the .**getWriter()** method for the **response** obj that gets us the stream on which we can write our output.
- response.setContentType("text/html");
  - The **setContentType**(java.lang.String type): Sets the content type of the **response** being sent to the client,

Add to the 1st activity

- Display your name with GREEN color on the out put after HELLO(2) message using H5 tag.
- Add a css file to the project.
  - Link it to the servlet2 (HTML output)
  - Try to apply the background color using linked-css file.

Activity 2

- Open Blackboard->Learning Material -> Module 2
- Download the sample-Servlet-2
- Add new servlet (name it Servlet3) to the java project
- Copy the given code to the servlet3.
- Explain how the servlet3 works.
  - Based on the Servlet Life Cycle, in your own word, explain how Servlet3 works?

Activity 3

- Develop a new Servlet (name it: **yourname-Time**) which displays the current TIME/DATE in blue color using H3 heading.
  - Link the css file that you created in Activity 1 to this page.

The role of JVM

- Using servlets allows the JVM to handle each request within a separate Java thread
- In most cases servlet container runs in a single JVM, but there are solutions when container need multiple JVMs.

Summary

- Servlets provide a component-based, platform-independent method for building Web based applications
- Servlets have access to the entire family of Java APIs, including the JDBC API to access enterprise databases.