CSC584 Enterprise Programming

Chapter 3 – Servlets

MARSHIMA MOHD ROSLI

COMPUTER SCIENCE

Chapter 3 Outline

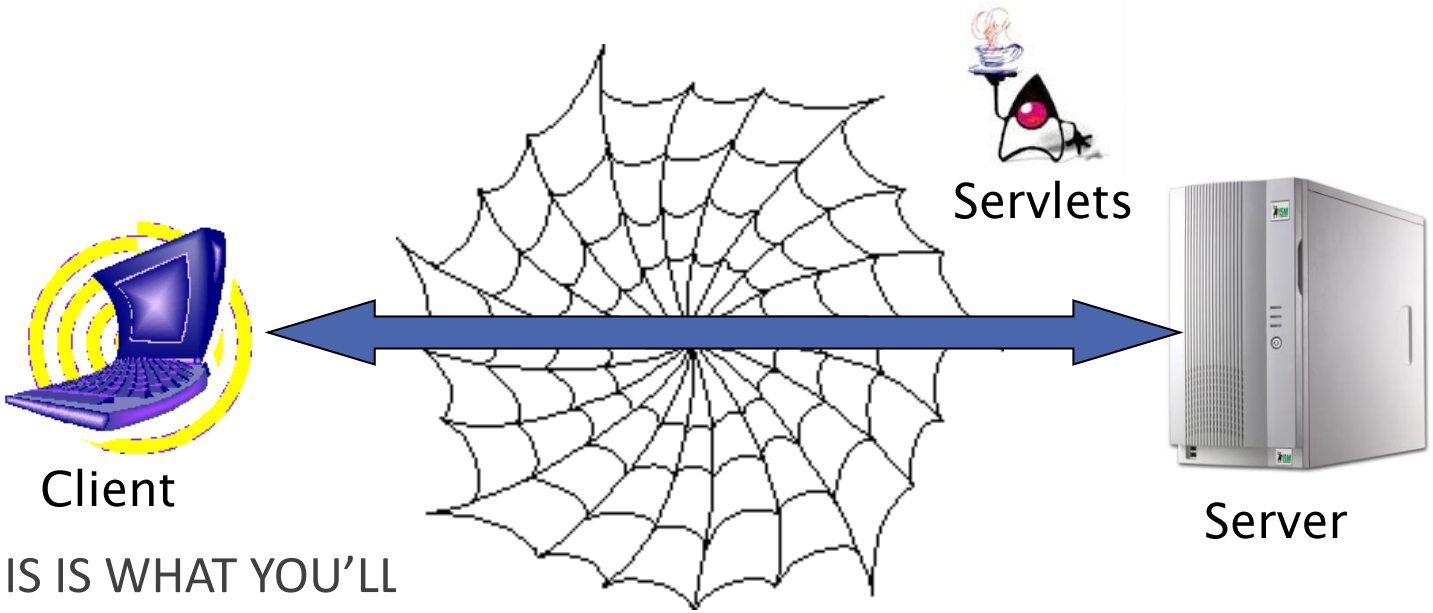
Servlets

- ❑ Creating & Running Servlets
- ❑ The Servlet API
- ❑ HTML forms
- ❑ Session tracking
- ❑ Database programming in servlets

Java on the Web: Java EE

Thin clients (minimize download)

Java all “server side”



THIS IS WHAT YOU’LL

Understand the concept of servlets

Servlet technology is primarily designed for use with the HTTP protocol of the Web.

Servlets are Java programs that run on a Web server.

Java servlets can be used to process client requests or produce dynamic Web pages.

Servlets

A servlet is like an applet, but on the server side

Client sends a request to server

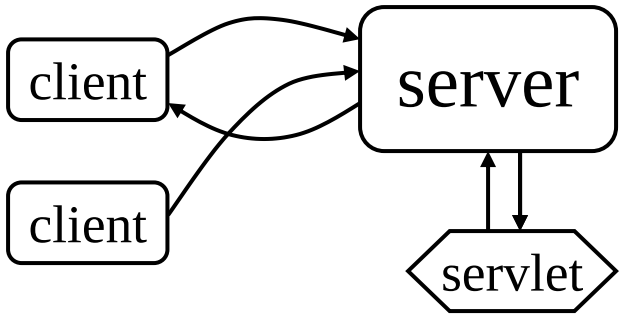
Server starts a servlet

Servlet computes a result for server and *does not quit*

Server returns response to client

Another client sends a request

Server calls the servlet again



Servlets vs. CGI scripts

Advantages:

- Running a servlet doesn't require creating a separate process each time
- A servlet stays in memory, so it doesn't have to be reloaded each time
- There is only one instance handling multiple requests, not a separate instance for every request

Disadvantage:

- Less choice of languages (CGI scripts can be in any language)

What are Servlets?

Units of Java code that run server-side.

Run in *containers* (provide context)

Helps with client-server communications

- Not necessarily over HTTP
- But usually over HTTP (we'll focus here)

What are Servlets?

A servlet is any class that implements the javax.servlet.Servlet interface

- In practice, most servlets extend the javax.servlet.http.HttpServlet class
- Some servlets extend javax.servlet.GenericServlet instead

Servlets, like applets, usually lack a main method, but must implement or override certain other methods

Why are Servlets?

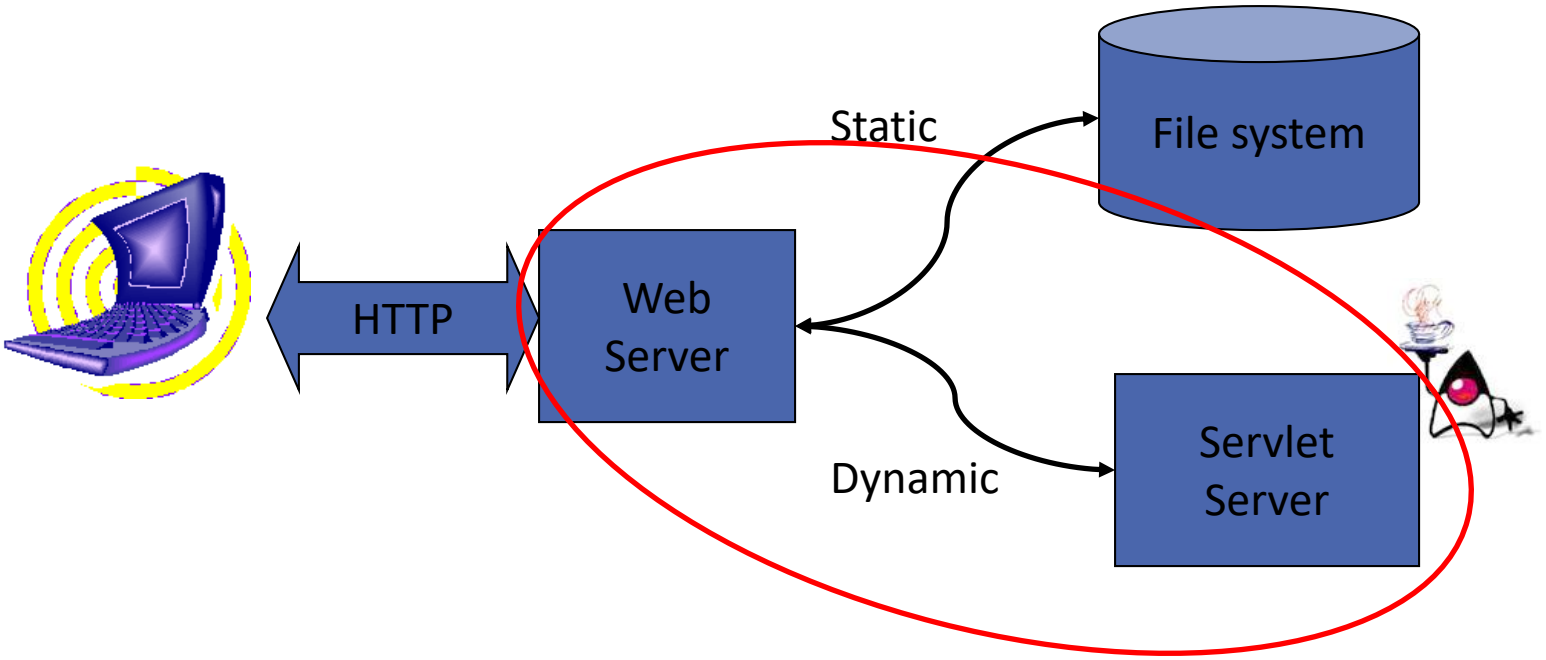
Web pages with dynamic content

Easy coordination between Servlets to make Web applications

Containers support many features

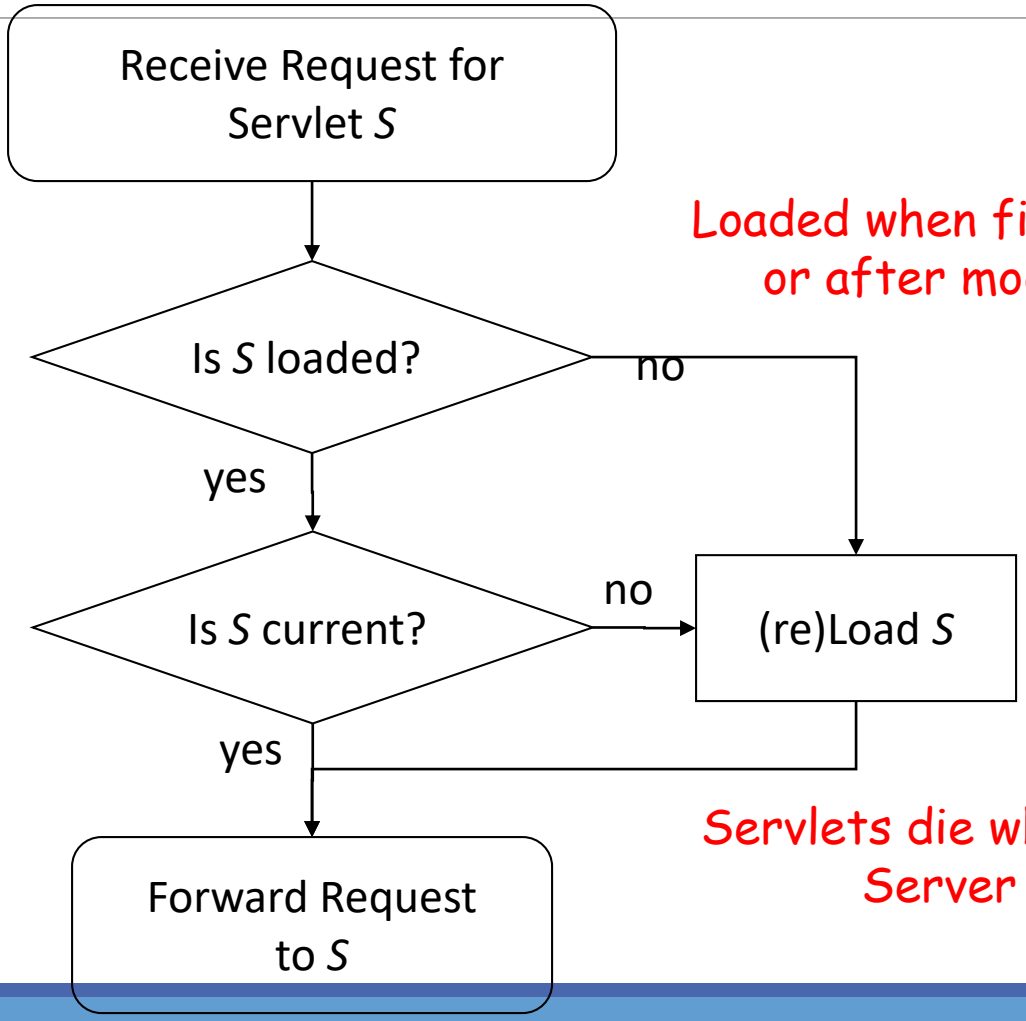
- Sessions, persistence, resource management (e.g., database connections), security, etc.

Where are Servlets?



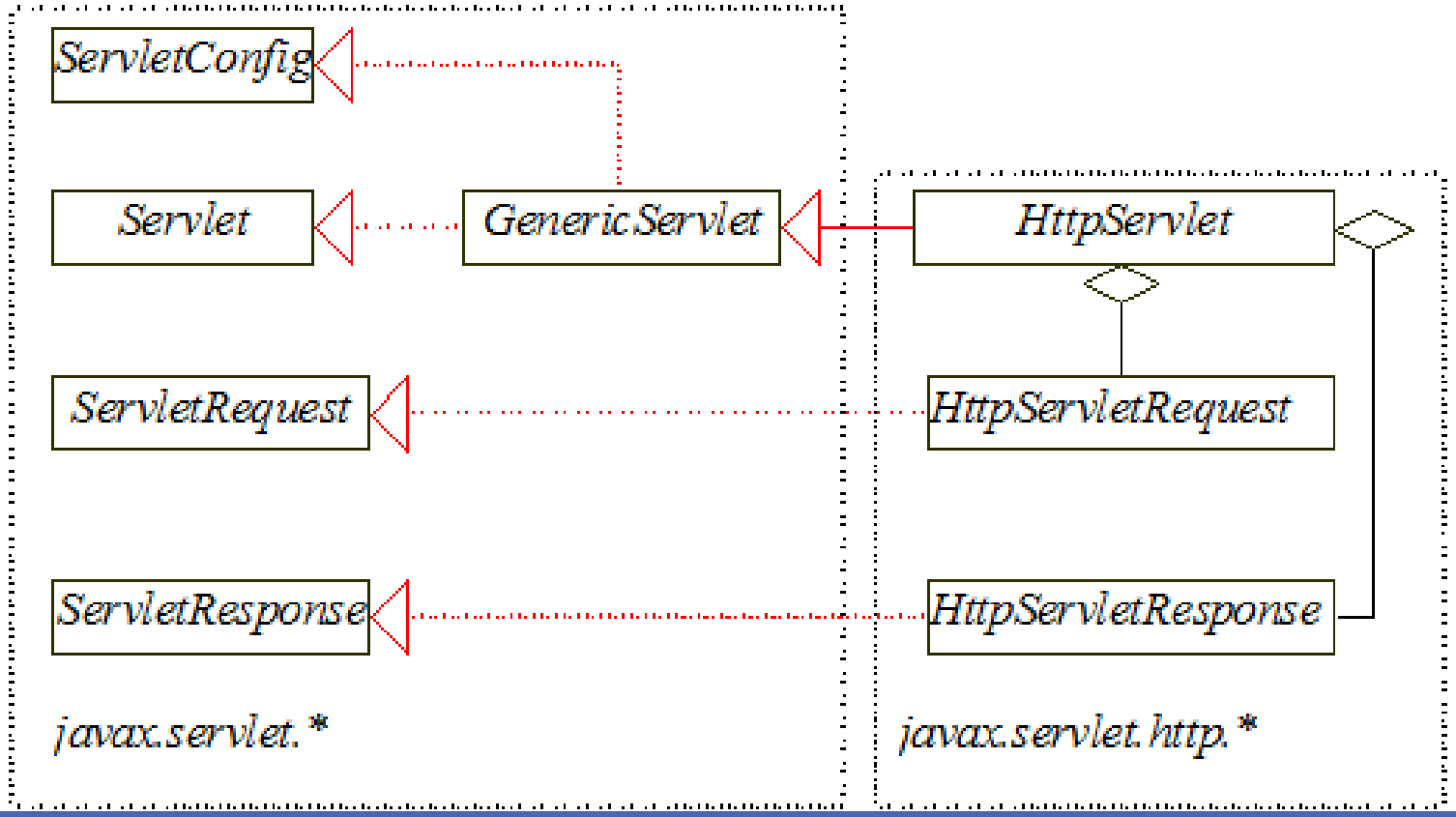
Tomcat = Web Server + Servlet Server

When are Servlets?



The Servlet API

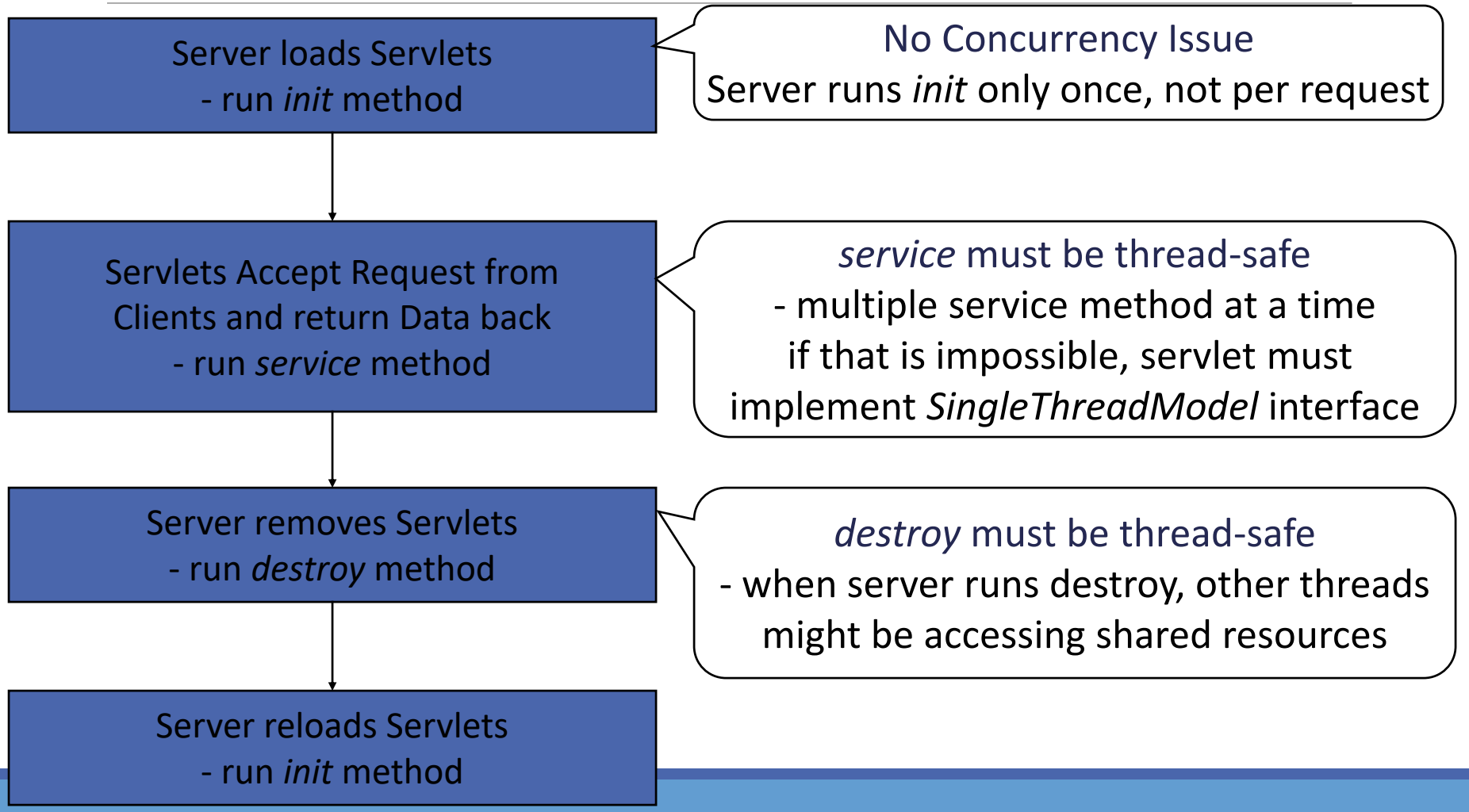
The servlet API provides the interfaces and classes that support servlets. These interfaces and classes are grouped into two packages: javax.servlet, and javax.servlet.http.



The Servlet Interface

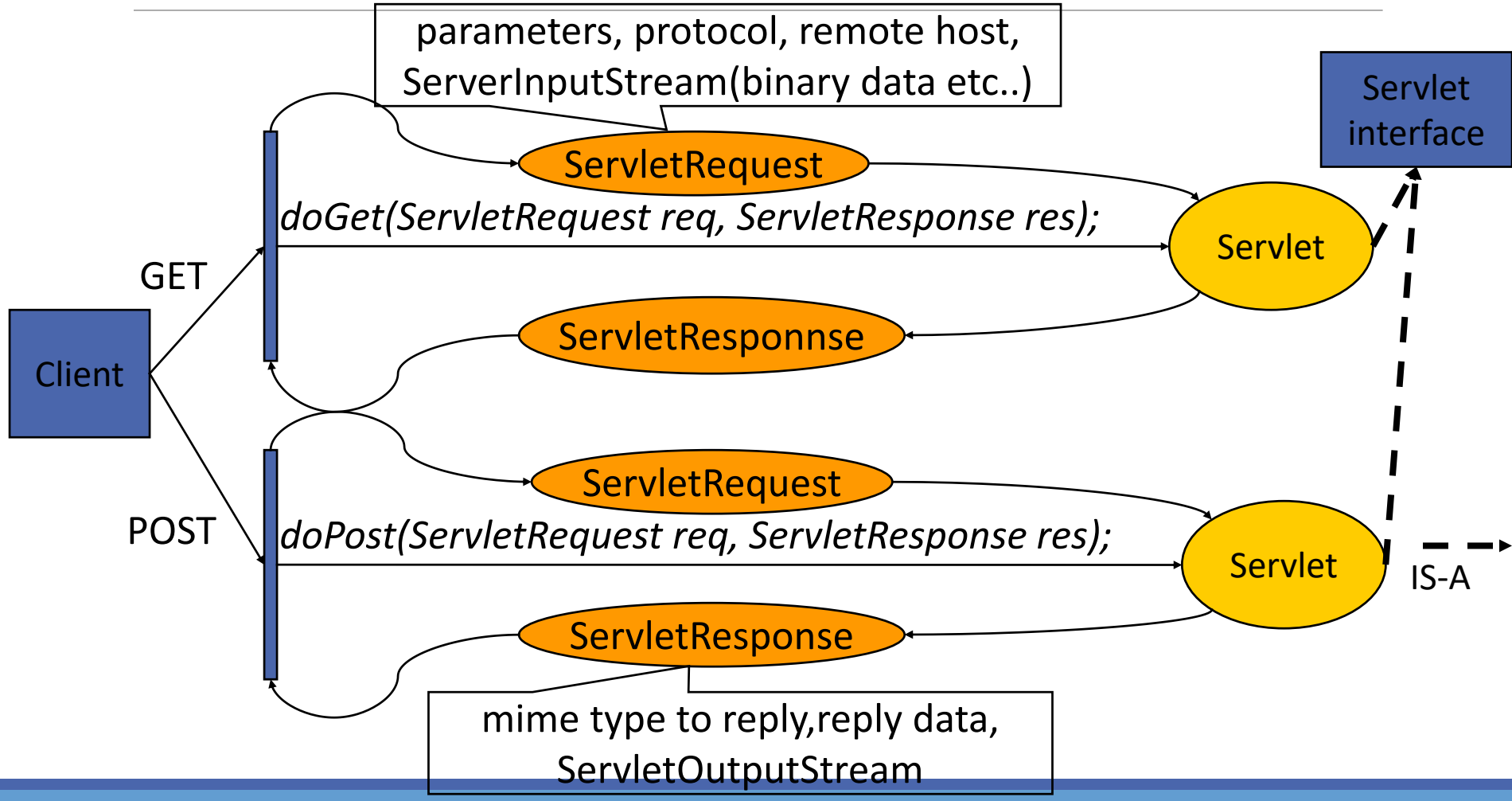
```
/**Invoked for every servlet constructed*/
public void init(ServletConfig p0) throws ServletException;

/**Invoked to respond to incoming requests*/
public void service(ServletRequest p0, ServletResponse p1)
  throws ServletException, IOException;

/**Invoked to release resource by the servlet*/
public void destroy();

/**Return information about the servlet*/
public String getServletInfo();

/**Return configuration objects of the servlet*/
public ServletConfig getServletConfig();
```

Servlet Lifecycle



Servlet Architecture Overview -

HTTP servlets



The HTTPServlet Class

1. The HttpServlet class defines a servlet for the HTTP protocol. It extends GenericServlet and implements the service method.

2. The service method is implemented as a dispatcher of HTTP requests. The HTTP requests are processed in the following methods: doGet, doPost, doDelete, doPut, doOptions, and doTrace. All these methods have the same signature as follows:

```
protected void doXxx(HttpServletRequest req,
HttpServletResponse resp)  throws ServletException,
java.io.IOException
```

The HttpServletRequest Interface

1. Every doXxx method in the HttpServlet class has an argument of the HttpServletRequest type, which is an object that contains HTTP request information including **parameter name and values, attributes, and an input stream.**
2. HttpServletRequest is a subinterface of ServletRequest. ServletRequest defines a more general interface to provide information for all kinds of clients.

The HttpServletResponse Interface

1. Every doXxx method in the HttpServlet class has an argument of the HttpServletResponse type, which is an object that assists a servlet in sending a response to the client.
2. HttpServletResponse is a subinterface of ServletResponse. ServletResponse defines a more general interface for sending output to the client.

Creating Servlets

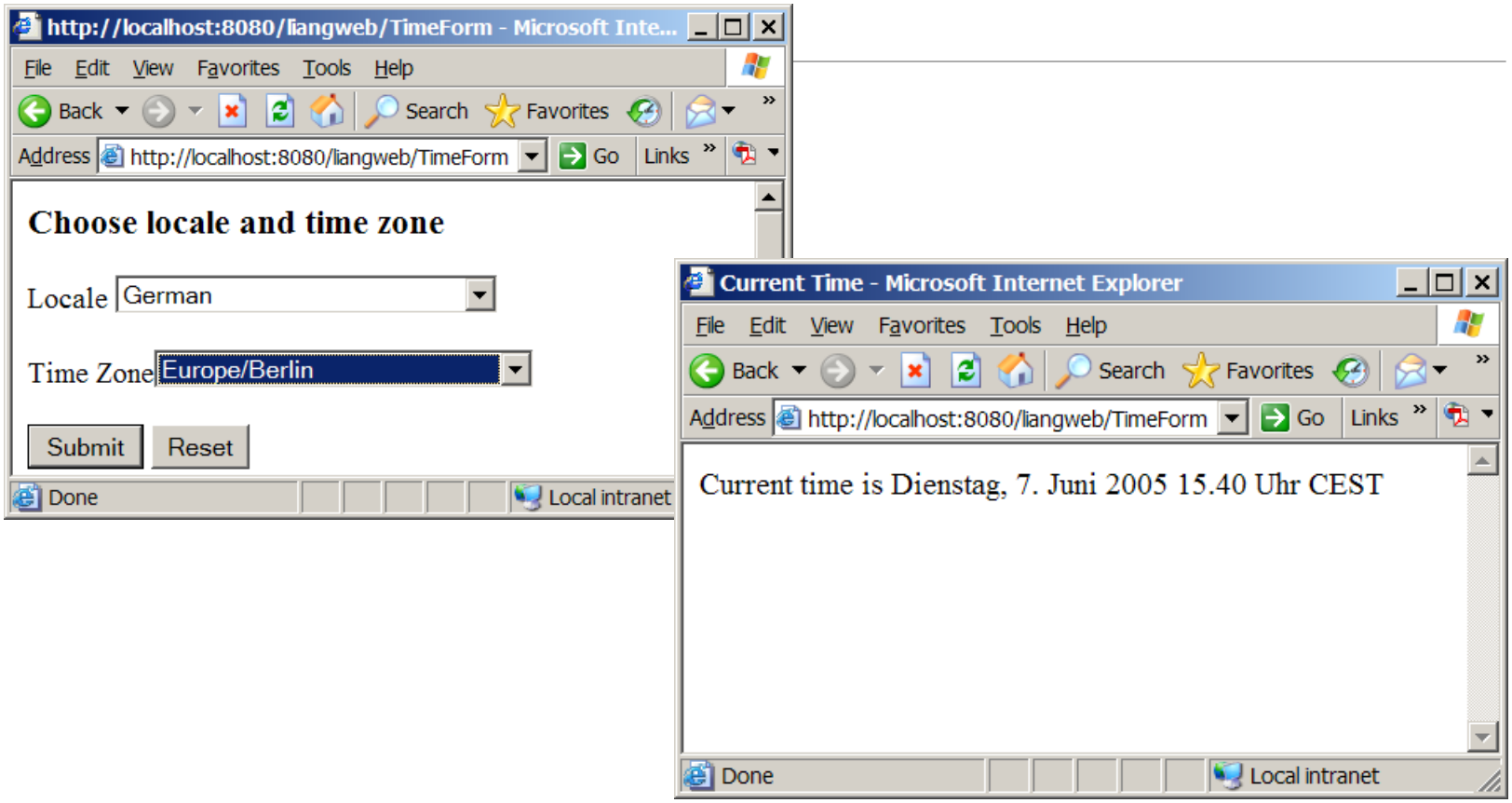
1. Servlets are opposites of the Java applets. Java applets run from a Web browser on the client side.
2. To write Java programs, you define classes.
3. To write a Java applet, you define a class that extends the Applet class.
4. The Web browser runs and controls the execution of the applet through the methods defined in the Applet class.
5. Similarly, to write a Java servlet, **you define a class that extends the HttpServlet class**.

Creating Servlets, cont.

1. The servlet engine controls the servlets using the init, doGet, doPost, destroy, and other methods. By default, the doGet and doPost methods do nothing.
2. To handle the GET request, you need to override the doGet method; to handle the POST request, you need to override the doPost method.

Example 34.1 Obtaining Current Time from Server

Example: Obtaining Current Time Based on Locale and Time Zone



TimeForm

Run

How are Servlets?

UNIVERSITI TEKNOLOGI MARA

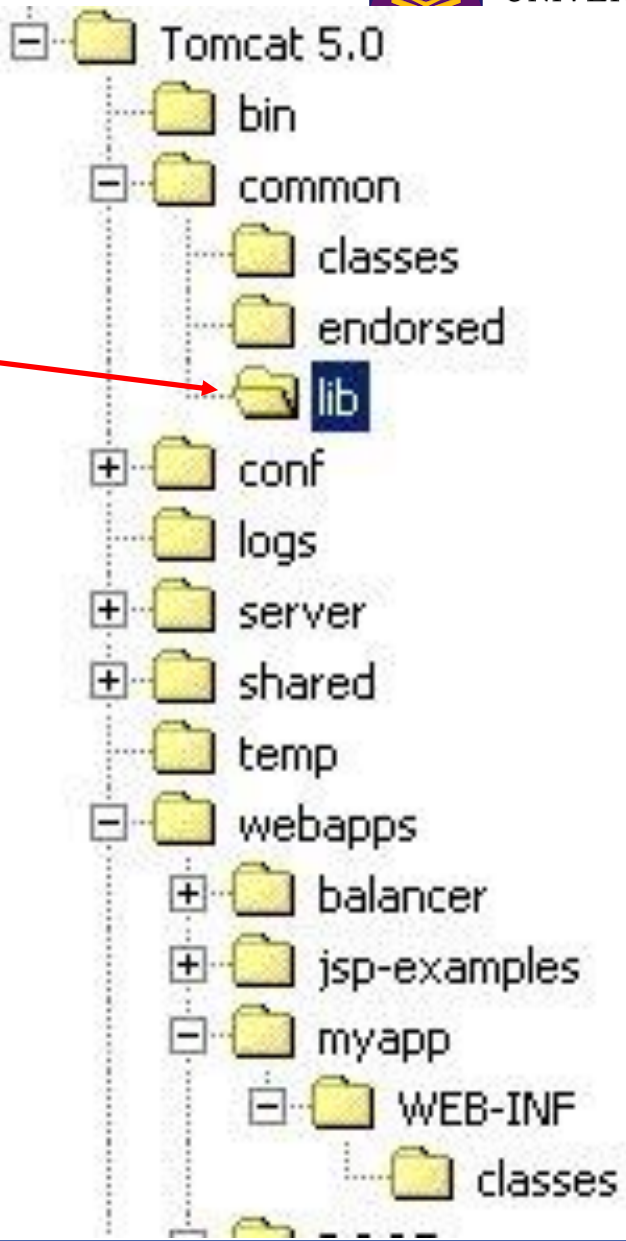
```
import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;

public class Hellox extends HttpServlet {

public void doGet(HttpServletRequest request,
                  HttpServletResponse response)
            throws IOException, ServletException
{
    response.setContentType("text/html");
    PrintWriter out = response.getWriter();
    out.println("<html>");
//    out.println("<body>");
    out.println("<head>");
    out.println("<title>Hello World!</title>");
    out.println("</head>");
    out.println("<body>");
    out.println("<h1>Hello World!</h1>");
    out.println("</body>");
    out.println("</html>");
} // doGet
} // Hellox
```

Compiling

javac –classpath

$LIB/servlet-api.jar

Hellox.java



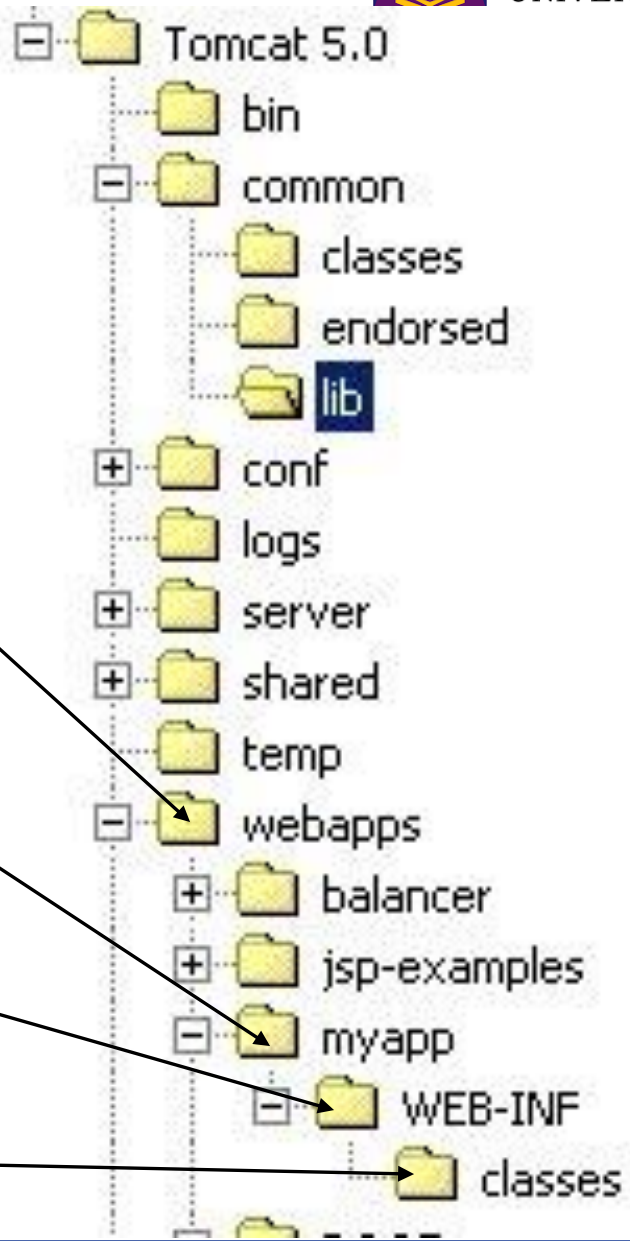
Directory Structure

Create your web applications here

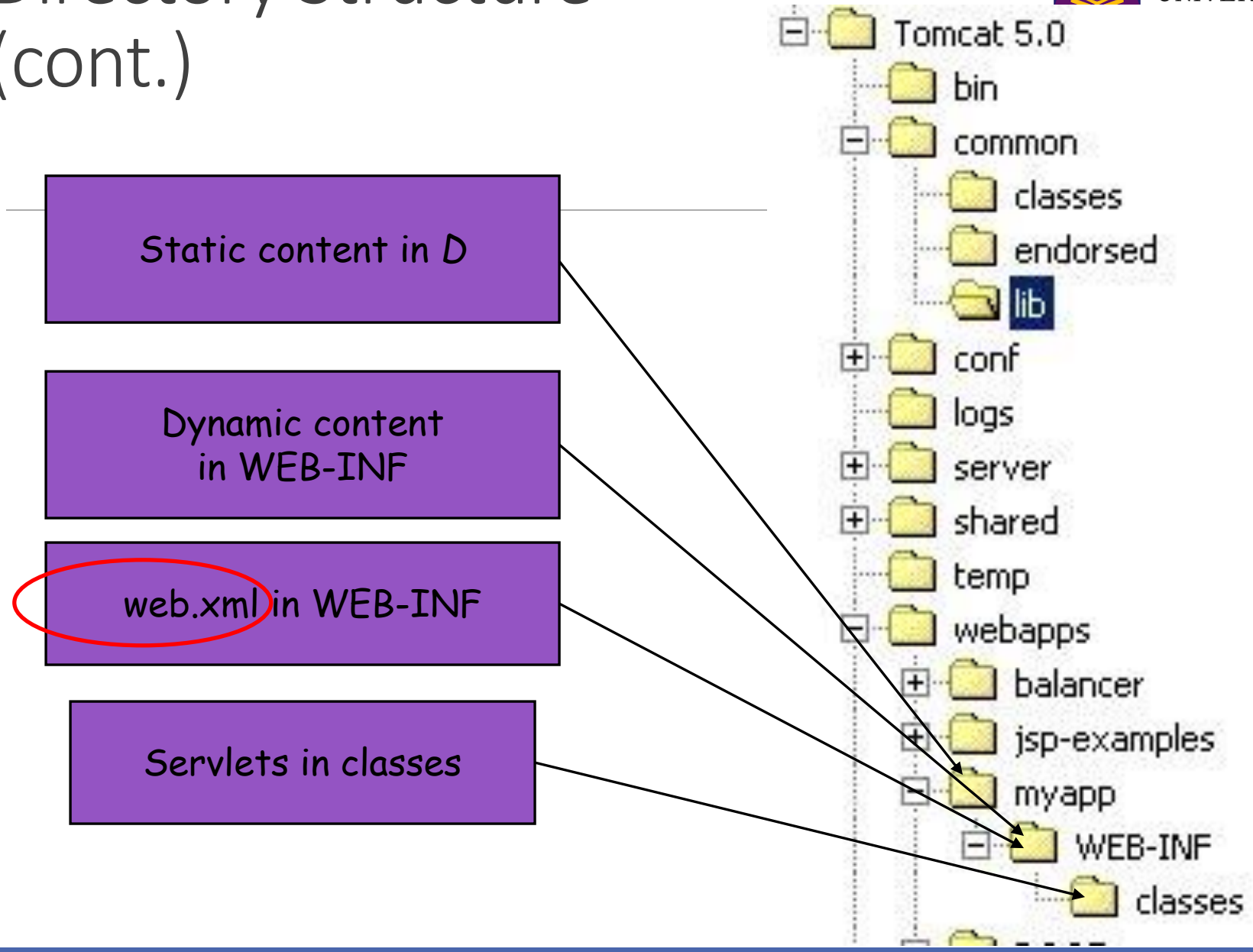
Create a directory *D* for your web application

Create "WEB-INF" under *D*

Create "classes" under "WEB-INF"



Directory Structure (cont.)



```
<?xml version="1.0" encoding="ISO-8859-1"?>

<web-app xmlns="http://java.sun.com/xml/ns/j2ee"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation=
"http://java.sun.com/xml/ns/j2ee http://java.sun.com/xml/ns/j2ee/web-app_2_4.xsd"
    version="2.4">

    <description>Examples</description>
    <display-name>Examples</display-name>

    <servlet>
      <servlet-name>Hellox</servlet-name>
      <servlet-class>Hellox</servlet-class>
    </servlet>

    <servlet-mapping>
        <servlet-name>Hellox</servlet-name>
        <url-pattern>/Hellox</url-pattern>
    </servlet-mapping> </web-app>
```

Declares servlet

abbreviation

fully qualified (e.g., java.lang.String)

Maps servlet to URL (rooted at D)

Session Tracking

1. Web servers use Hyper-Text Transport Protocol (HTTP). HTTP is a stateless protocol. The HTTP Web server cannot associate requests from a client together.
2. Each request is treated independently by the Web server. This protocol works fine for simple Web browsing, where each request typically results in an HTML file or a text file being sent back to the client.
3. Such simple requests are isolated. However, the requests in interactive Web applications are often related.

What is a Session ?

A session can be defined as a series of related interactions between a single client and the Web server over a period of time. To track data among requests in a session is known as session tracking.

Session TrackingTechniques

Using hidden values, using cookies, and using the session tracking tools from servlet API.

Session Tracking Using Hidden Values

1. You can track session by passing data from the servlet to the client as hidden value in a dynamically generated HTML form by including a field like this:

   <input type=”hidden” name=”lastName” value=”Smith”>

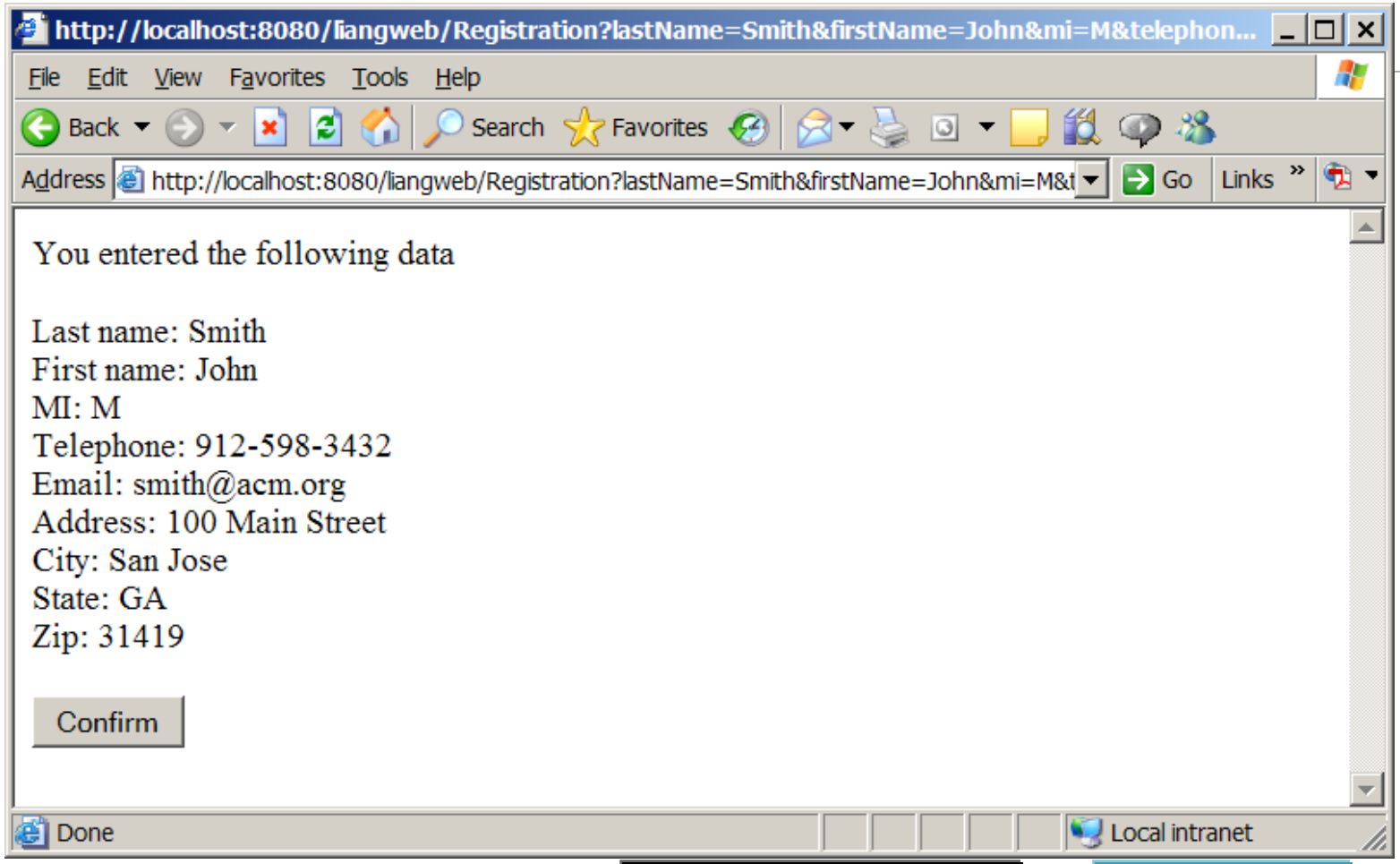
2. So the next request will submit the data back to the servlet.
3. The servlet retrieves this hidden value just like any other parameter value using the getParameter method.

Example: Using Hidden Values in the Registration form

This example creates a servlet that processes a registration form.

1. The client first submits the form using the GET method, as shown in Figure 16.18.
2. The server collects the data in the form, displays the data to the client, and asks the client for confirmation, as shown in Figure 16.20.
3. The client confirms it by submitting the request with the hidden values using the POST method.
4. Finally, the servlet writes the data to a database.

Example: Using Hidden Values in the Registration form, cont.



Registration

Run

Session Tracking Using Cookies

1. You can track sessions using cookies. Cookies are small text files that store sets of name=value pairs on the disk in the client's computer.
2. Cookies are sent from the server through the instructions in the header of the HTTP response.
3. The instructions tell the browser to create a cookie with a given name and its associated value. If the browser already has the cookie with the key name, the value will be updated.
4. The browser will then send the cookie with any request submitted to the same server. Cookies can have expiration dates set, after which the cookies will not be sent to the server.

Session Tracking Using the Servlet API

1. The problems of session tracking with hidden data and cookies are that data are not secured and difficult to deal with large set of data.
2. Java servlet API provides a session tracking tool, which enables tracking of a large set of data. Data can be stored as objects. Data are kept on the server side so they are secure.

The HttpSession Class

1. To use the Java servlet API for session tracking, first create a session object using the getSession method in the HttpServletRequest interface like this:

```
HttpSession session = request.getSession(true);
```

2. This obtains the session or creates a new session if the client does not have a session on the server.

3. The HttpSession class provides the methods for reading and storing data to the session, and for manipulating the session.

Sending Images From the Servlets

Java servlets are not limited to sending text to a browser. Java servlets can return images in GIF, JPEG, or PNG format. This section demonstrates returning images in GIF format.

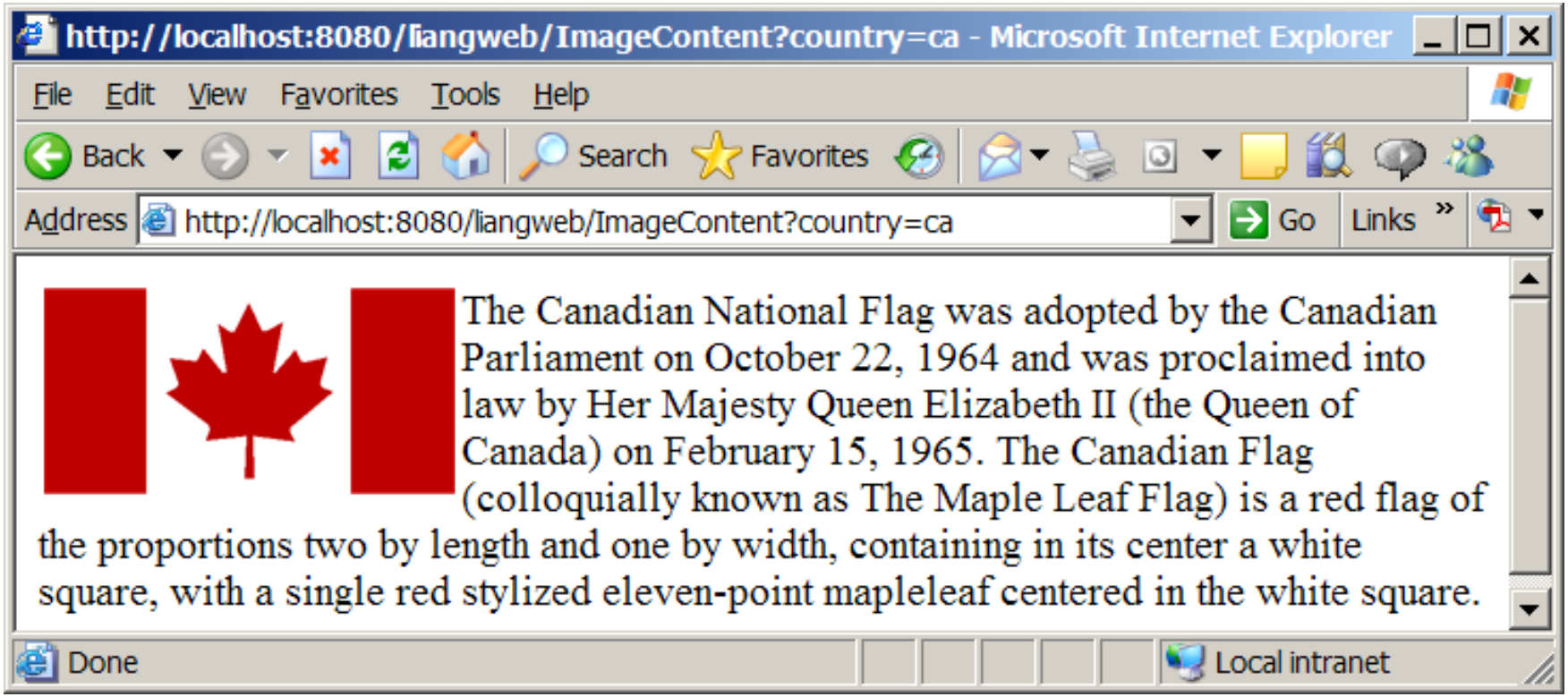
To send contents as a GIF image, the content type must be set to image/gif like this:

```
response.setContentType("image/gif");
```

Images are binary data. You have to use a binary output stream like this:

```
OutputStream out = response.getOutputStream();
```

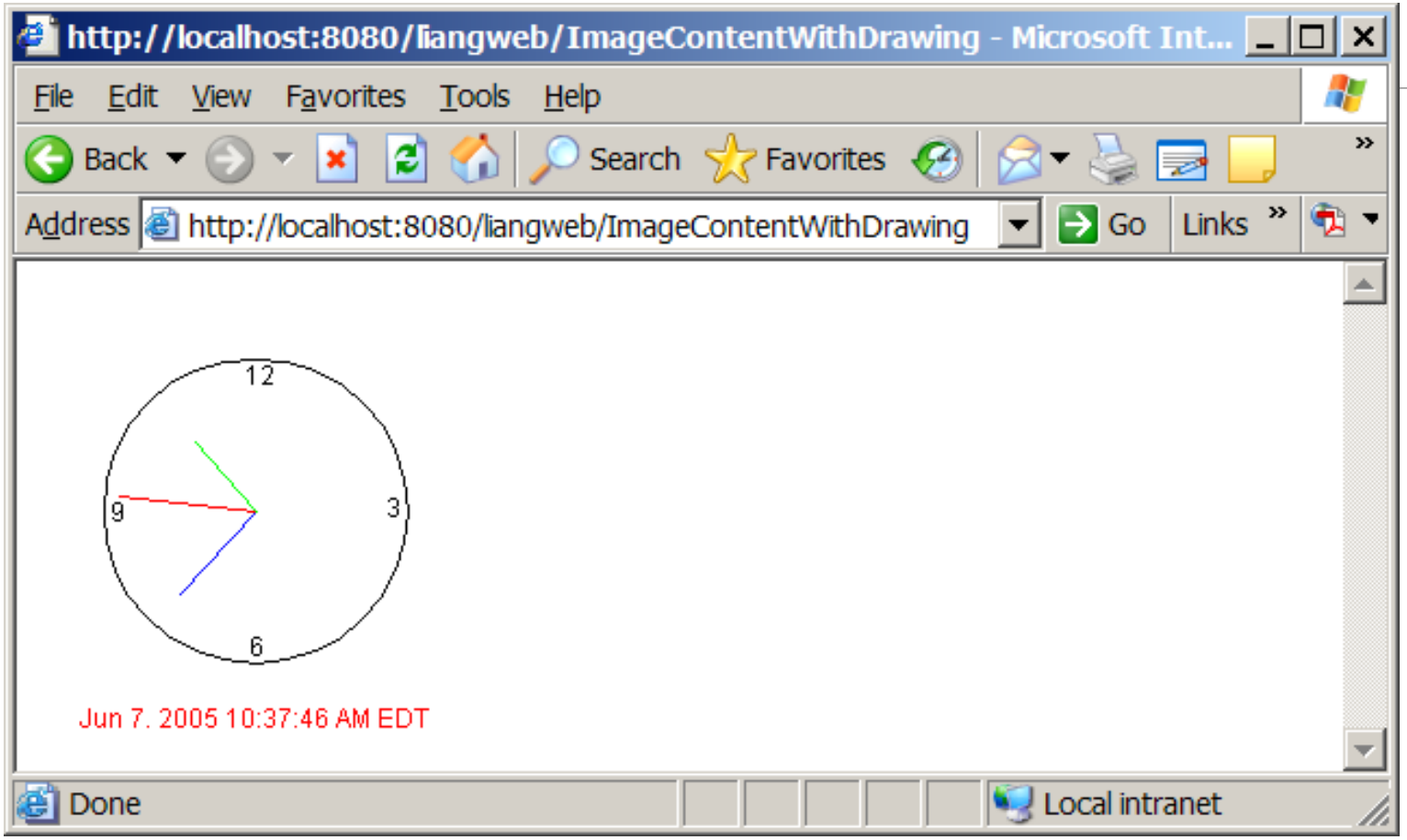
Example: Getting Images from Servlets



ImageContent

Run

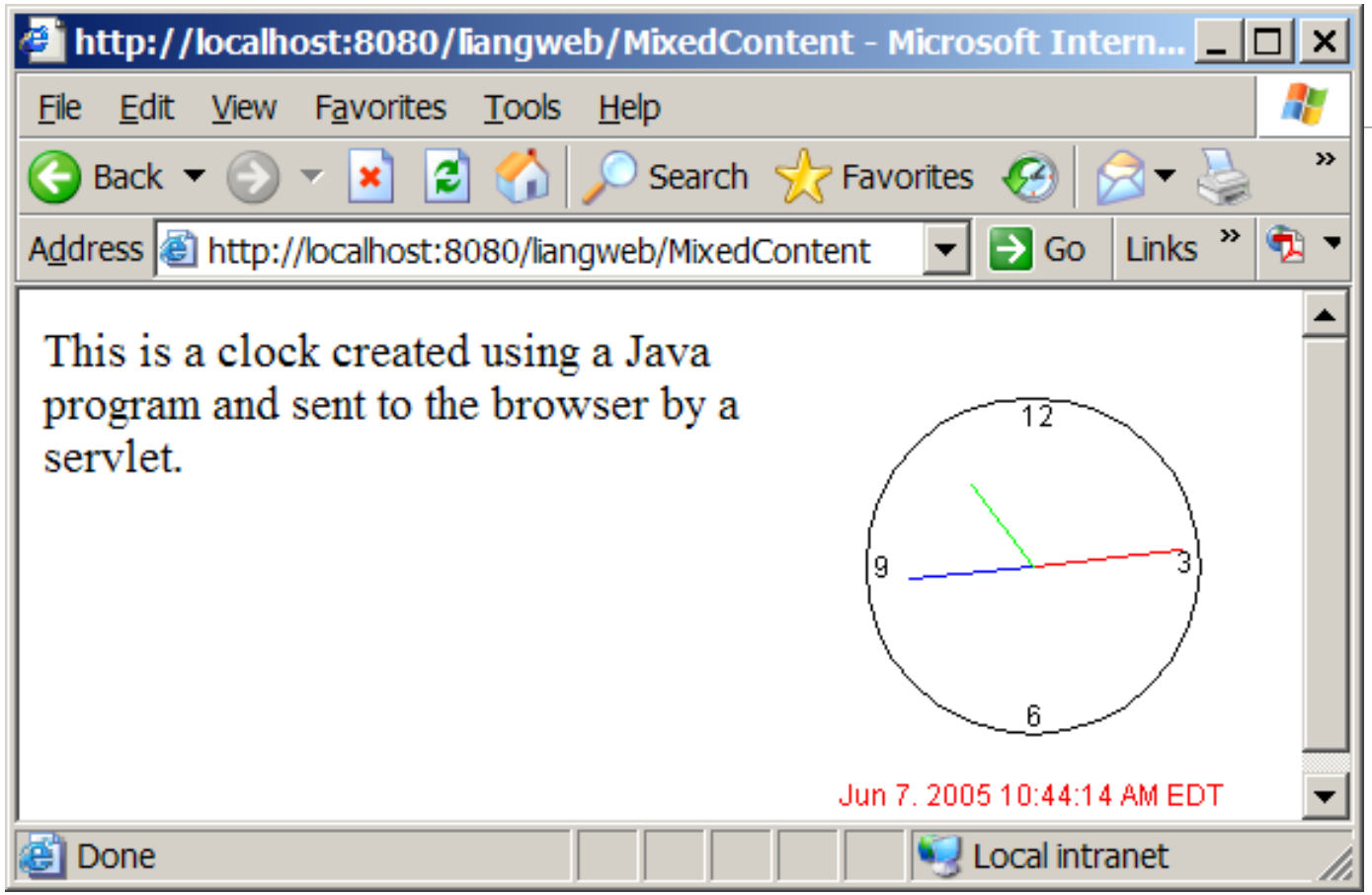
Example: Creating Images by Drawing



ImageContentWithDrawing

Run

Example: Mixing Images and Texts



MixedContent

Run