Chapter 37 Servlets

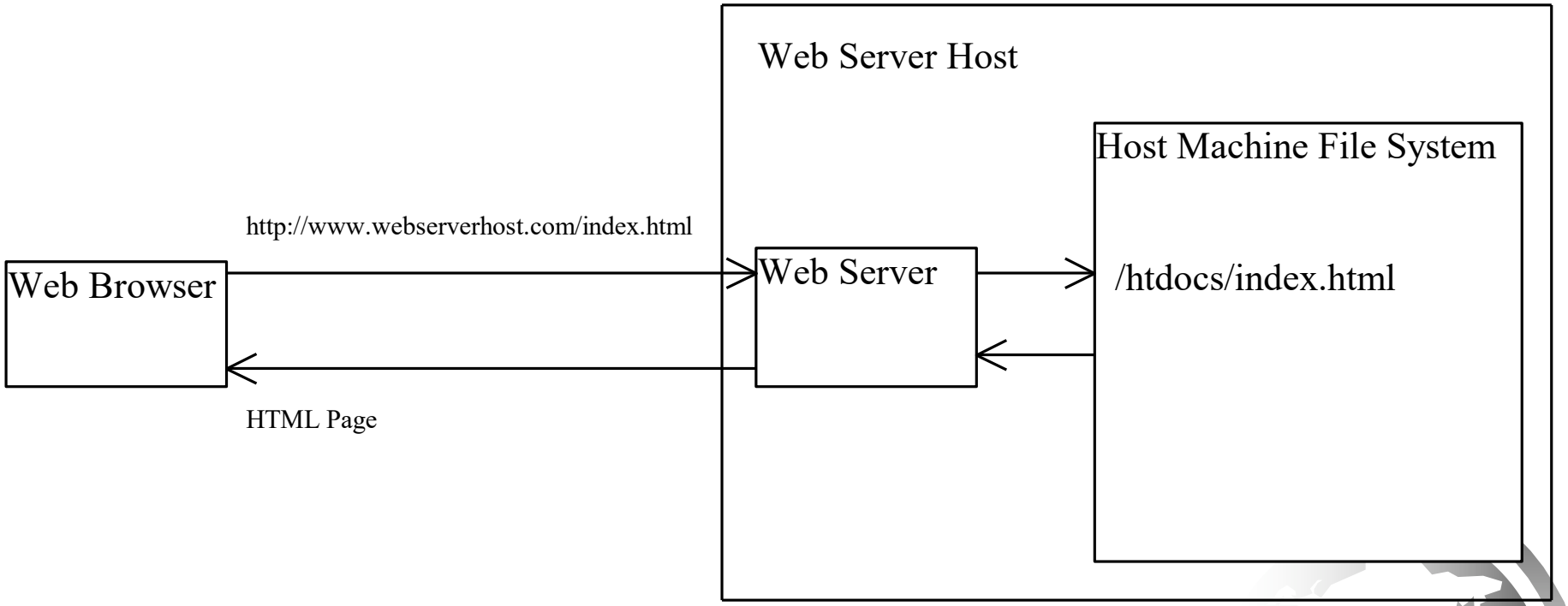
Objectives

- To explain how a servlet works (§37.2).
- To create/develop/run servlets (§37.3).
- To deploy servlets on application servers such as Tomcat (§37.3).
- To describe the servlets API (§37.4).
- To create simple servlets (§37.5).
- To create and process HTML forms (§37.6).
- To develop servlets to access databases (§37.7).
- To use hidden fields, cookies, and HttpSession to track sessions (§37.8).
- To send images from servlets (§37.9).

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.

HTTP and HTML

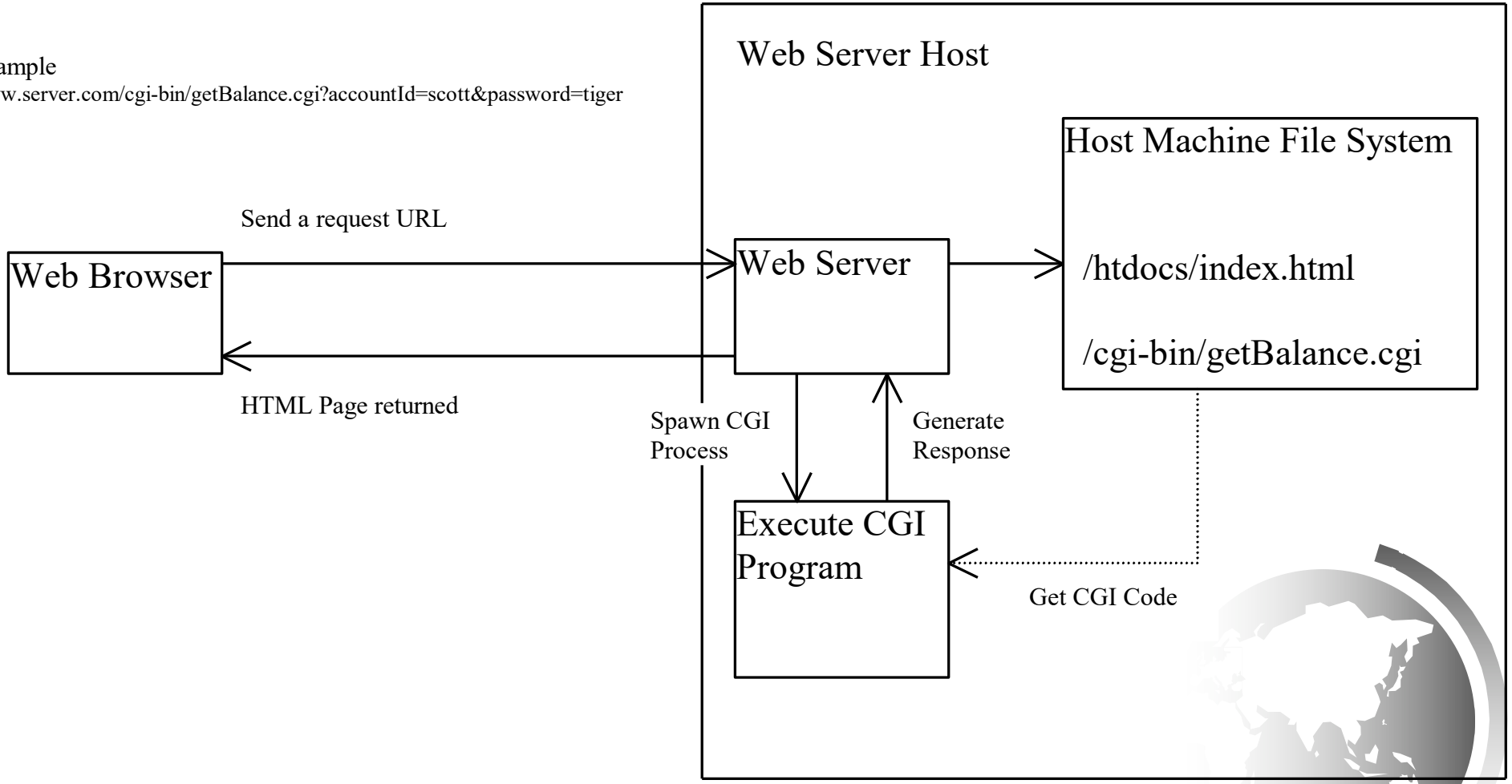


CGI

The Common Gateway Interface, or CGI, was proposed to generate dynamic Web contents. The interface provides a standard framework for Web servers to interact with external programs, known as the CGI programs.

How Does CGI Work?

URL Example
http://www.server.com/cgi-bin/getBalance.cgi?accountId=scott&password=tiger



The GET and POST Methods

The two most common HTTP requests, also known as methods, are GET and POST. The Web browser issues a request using a URL or an HTML form to trigger the Web server to execute a CGI program. When issuing a CGI request directly from a URL, the GET method is used. This URL is known as a query string.

Query String

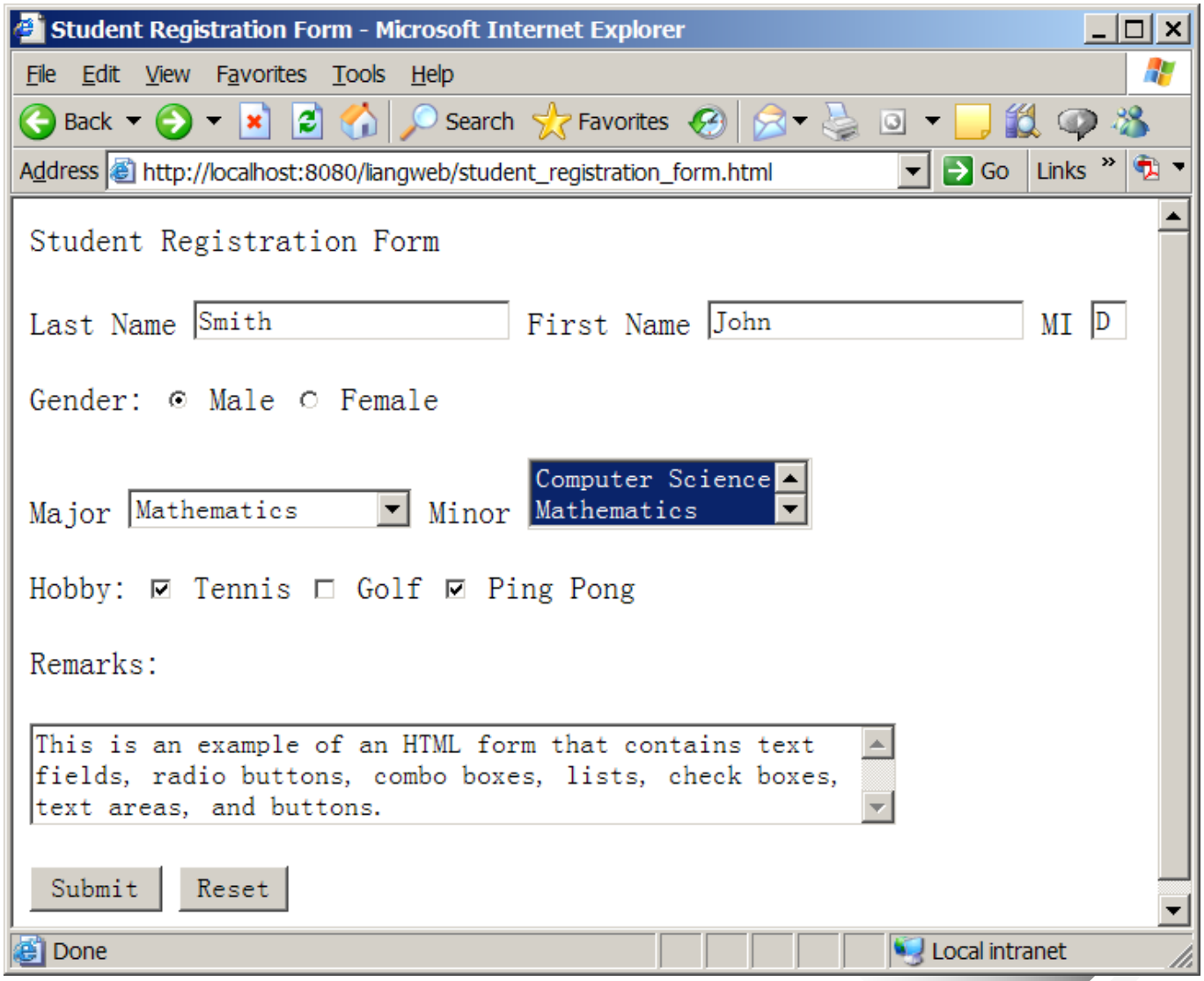
The URL query string consists of the location of the CGI program, parameters and their values.

http://www.webserverhost.com/cgi-bin/getBalance.cgi?accountId=scott+smith&password=tiger

The ? symbol separates the program from the parameters. The parameter name and value are associated using the = symbol. The parameter pairs are separated using the & symbol. The + symbol denotes a space character.

HTML Forms

HTML forms enable you to submit data to the Web server in a convenient form. The form can contain text fields, text area, check boxes, combo boxes, lists, radio buttons, and buttons.



From CGI to Java Servlets

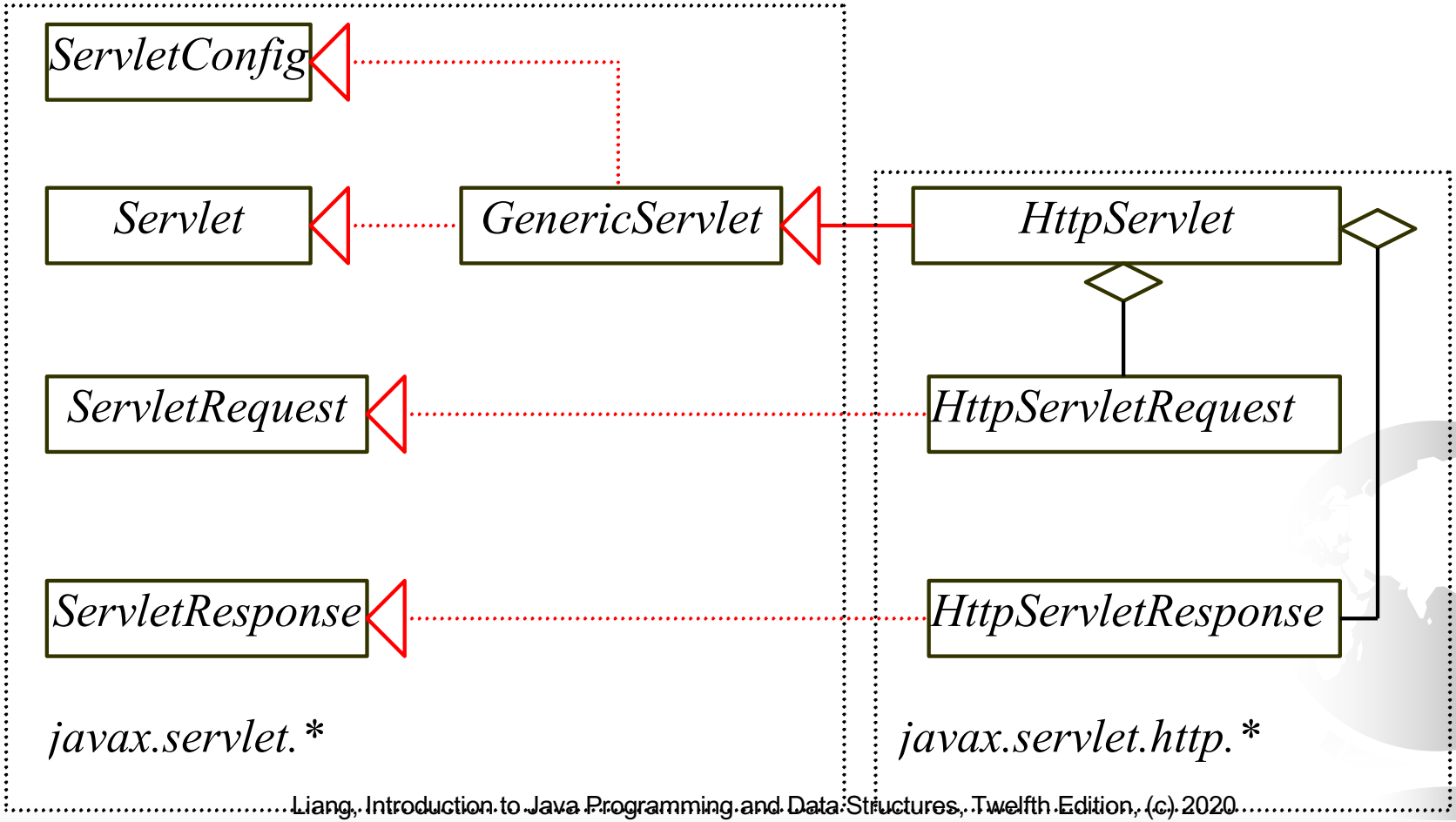
Java servlets are Java programs. They function like CGI programs. They are executed upon the request from Web browser. All the servlets run inside a servlet container. A servlet container is also referred to as a servlet server, or a servlet engine. A servlet container is a single process that runs a JVM. The JVM creates a thread to handle each servlet. Java threads have much less overhead than full-brown processes. All the threads share the same memory allocated to the JVM.

Creating and Running Servlets from

To run Java servlets, you need a servlet container. Many servlet containers are available. Tomcat, developed by Apache (www.apache.org), is a standard reference implementation for Java servlet 2.2 and Java Server Pages 1.1.

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 Life-Cycle

1. The init method is called when the servlet is first created, and is not called again as long as the servlet is not destroyed. This resembles the applet's init method, which is invoked when the applet is created, and is not invoked again as long as applet is not destroyed.

2. The service method is invoked each time the server receives a request for the servlet. The server spawns a new thread and invokes service.

3. The destroy method is invoked once all threads within the servlet's service method have exited or after a timeout period has passed. This method releases resources for the servlet.

The HTTPServlet Class

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

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. HttpServletRequest is a subinterface of ServeletRequest. ServletRequest defines a more general interface to provide information for all kinds of clients.

The HttpServletResponse Interface

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. HttpServletResponse is a subinterface of ServeletResponse. ServletRequest defines a more general interface for sending output to the client.

Creating Servlets

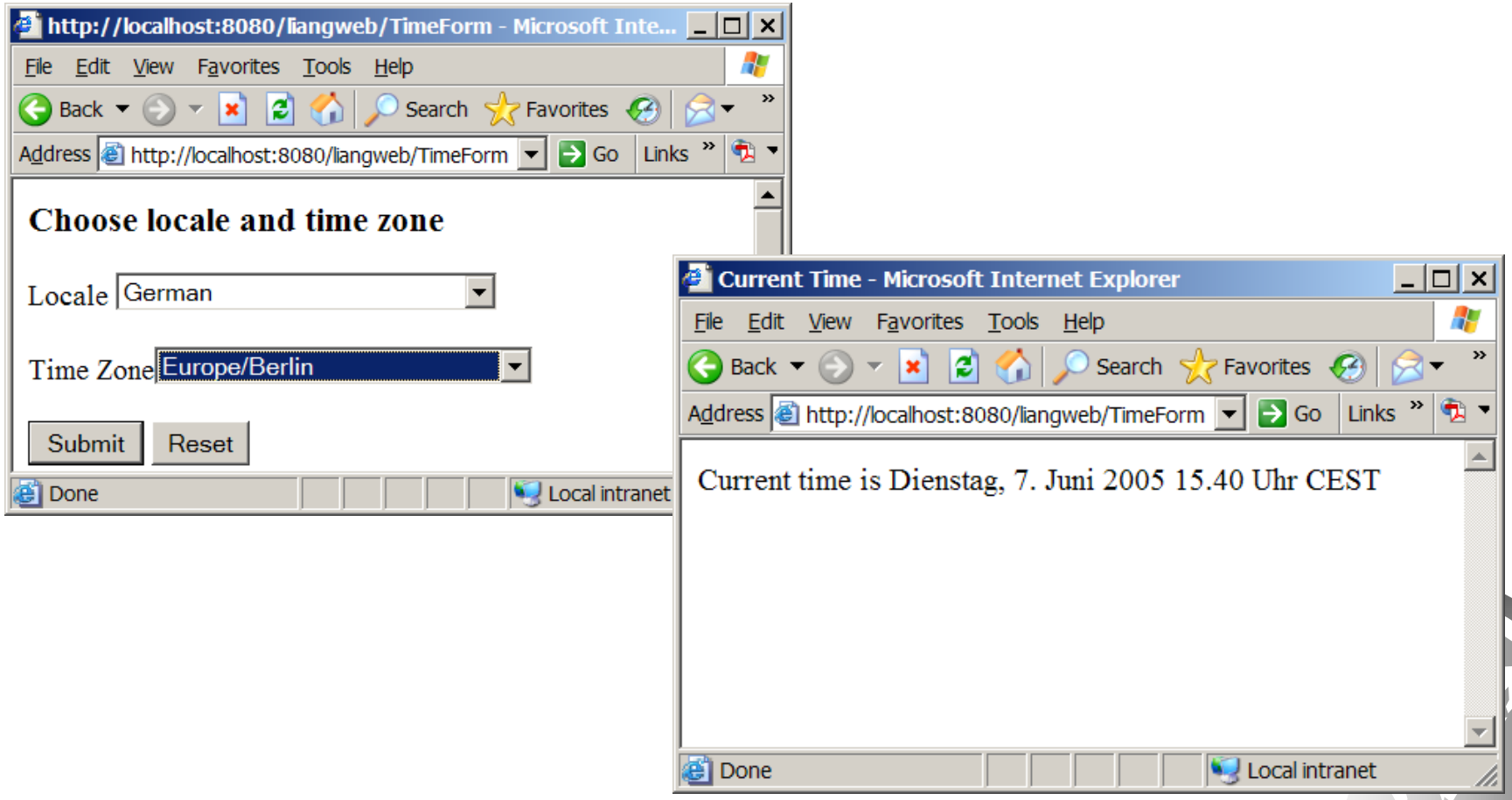
Servlets are opposites of the Java applets. Java applets run from a Web browser on the client side. To write Java programs, you define classes. To write a Java applet, you define a class that extends the Applet class. The Web browser runs and controls the execution of the applet through the methods defined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

Creating Servlets, cont.

The servlet engine controls the servlets using the init, doGet, doPost, destroy, and other methods. By default, the doGet and doPost methods do nothing. 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



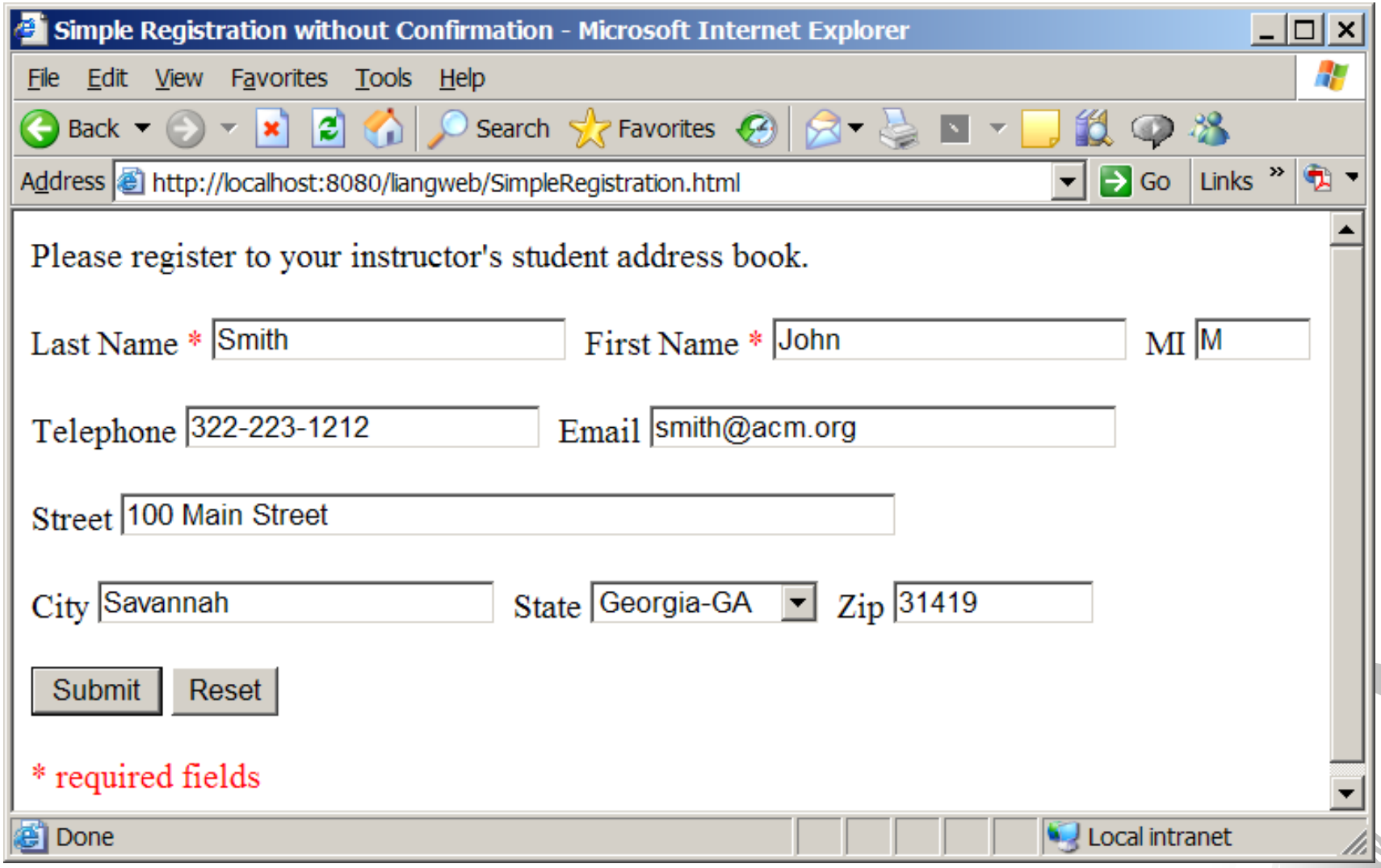
Run

Database Programming Using Servlets

Many dynamic Web applications use databases to store and manage data. Servlets can connect to any relational database via JDBC. Connecting a servlet to a database is no different from connecting a Java application or applet to a database. If you know Java servlets and JDBC, you can combine them together to develop interesting and practical Web based interactive projects immediately.

Example 34.3 Registering Student into a Database

Example: Registering Student into a Database



Run

Session Tracking

Web servers use Hyper-Text Transport Protocol (HTTP). HTTP is a stateless protocol. The HTTP Web server cannot associate requests from a client together. 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. 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 Tracking Techniques

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

Session Tracking Using Hidden Values

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:

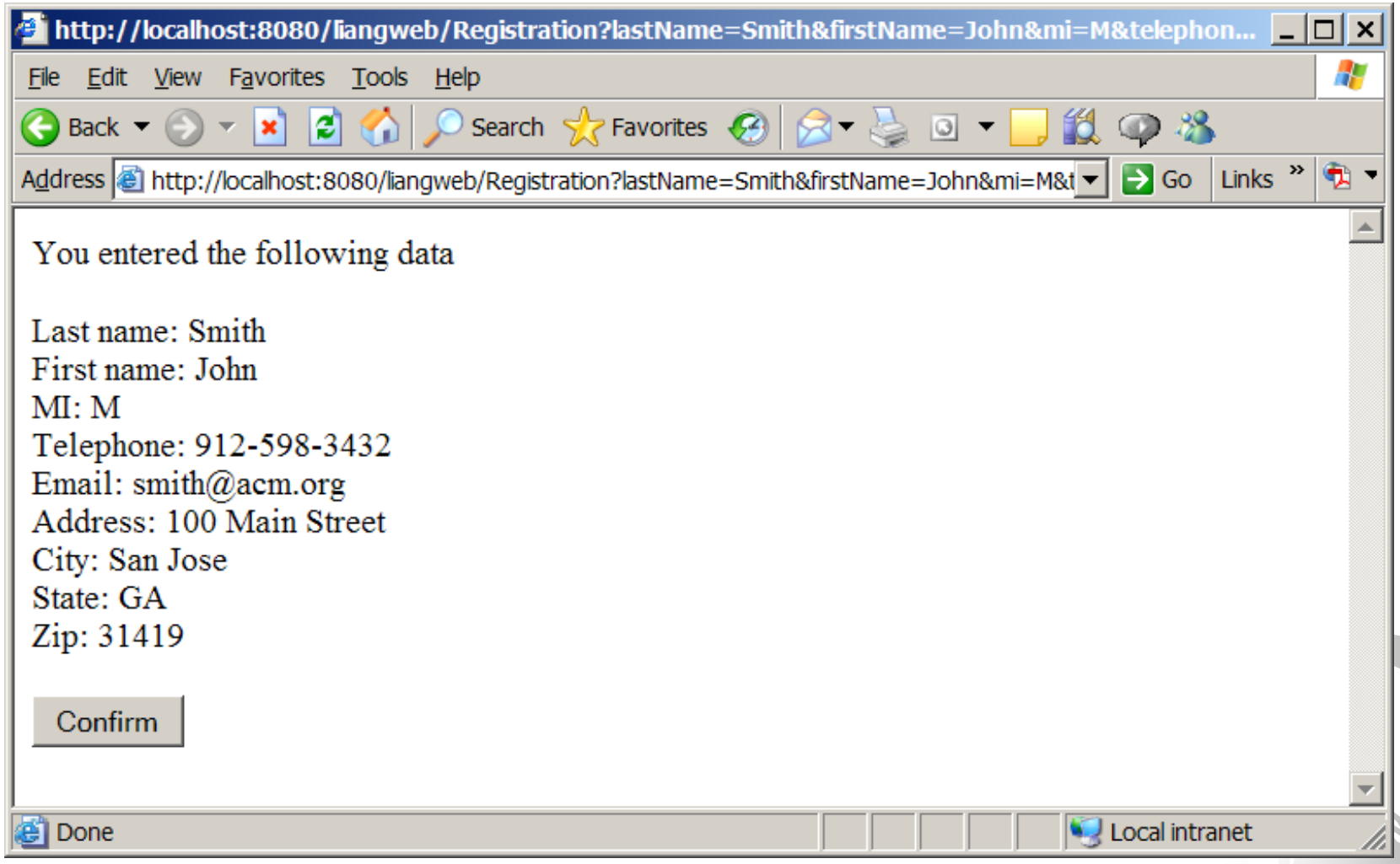
```
<input type=”hidden” name=”lastName” value=”Smith”>
```

So the next request will submit the data back to the servlet. 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. The client first submits the form using the GET method. The server collects the data in the form, displays the data to the client, and asks the client for confirmation. The client confirms it by submitting the request with the hidden values using the POST method. Finally, the servlet writes the data to a database.

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



Run

Session Tracking Using Cookies

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. Cookies are sent from the server through the instructions in the header of the HTTP response. 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. 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

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.

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

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);
```

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

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.

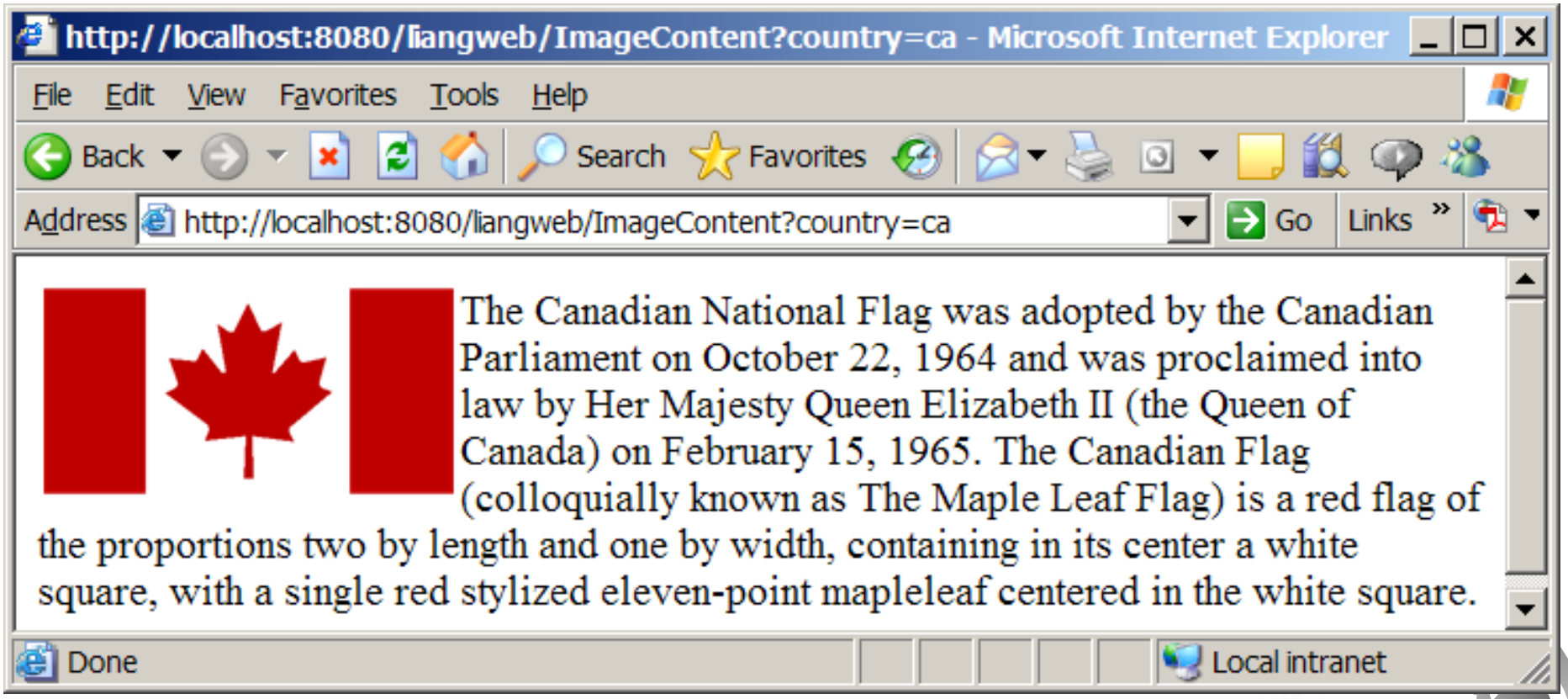
To send contents as a GIF image, the content type must be set to image/gif like this:

```
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



Run