# Java Servlets

**Outline**

# 24.1 Introduction

- Java networking capabilities
  - Socket-based and packet-based communications
    - Package **`java.net`**
  - Remote Method Invocation (RMI)
    - Package **`java.rmi`**
  - Servlets and Java Server Pages (JSP)
    - Request-response model
    - Packages **`javax.servlet`**
      **`javax.servlet.http`**
      **`javax.servlet.jsp`**
      **`javax.servlet.tagext`**
    - Form the Web tier of J2EE

# 24.1 Introduction (Cont.)

- Servlets
  - Thin clients
  - Request/response mechanism
  - redirection
- Tomcat
  - Jakarta project
  - Official reference implementation of the JSP and servlet standards

# 24.2 Servlet Overview and Architecture

- Servlet container (servlet engine)
  - Server that executes a servlet
- Web servers and application servers
  - Sun ONE Application Server
  - Microsoft's Internet Information Server (IIS)
  - Apache HTTP Server
  - BEA's WebLogic Application Server
  - IBM's WebSphere Application Server
  - World Wide Web Consortium's Jigsaw Web Server

# 24.2.1 Interface `Servlet` and the Servlet Life Cycle

- Interface **`Servlet`**
  - All servlets must implement this interface
  - All methods of interface **`Servlet`** are invoked by servlet container
- Servlet life cycle
  - Servlet container invokes the servlet's **`init`** method
  - Servlet's **`service`** method handles requests
  - Servlet's **`destroy`** method releases servlet resources when the servlet container terminates the servlet
- Servlet implementation
  - **`GenericServlet`**
  - **`HttpServlet`**

# 24.2.1 Interface Servlet and the Servlet Life Cycle (Cont.)

| Method | Description |
|---|---|
| `void init( ServletConfig config )` | |
| | The servlet container calls this method once during a servlet's execution cycle to initialize the servlet. The `ServletConfig` argument is supplied by the servlet container that executes the servlet. |
| `ServletConfig getServletConfig()` | |
| | This method returns a reference to an object that implements interface `ServletConfig`. This object provides access to the servlet's configuration information such as servlet initialization parameters and the servlet's `ServletContext`, which provides the servlet with access to its environment (i.e., the servlet container in which the servlet executes). |
| `String getServletInfo()` | |
| | This method is defined by a servlet programmer to return a string containing servlet information such as the servlet's author and version. |
| `void service( ServletRequest request, ServletResponse response )` | |
| | The servlet container calls this method to respond to a client request to the servlet. |
| `void destroy()` | |
| | This "cleanup" method is called when a servlet is terminated by its servlet container. Resources used by the servlet, such as an open file or an open database connection, should be deallocated here. |

**Fig. 24.1** Methods of interface `Servlet` (package `javax.servlet`).

# 24.2.2 HttpServlet Class

- Overrides method **`service`**
- Two most common HTTP request types
  - **`get`** requests
  - **`post`** requests
- Method **`doGet`** responds to **`get`** requests
- Method **`doPost`** responds to **`post`** requests
- **`HttpServletRequest`** and **`HttpServletResponse`** objects

# 24.2.2 `HttpServlet` Class (Cont.)

| Method | Description |
|---|---|
| `doDelete` | Called in response to an HTTP *`delete`* request. Such a request is normally used to delete a file from a server. This may not be available on some servers, because of its inherent security risks (e.g., the client could delete a file that is critical to the execution of the server or an application). |
| `doHead` | Called in response to an HTTP *`head`* request. Such a request is normally used when the client only wants the headers of a response, such as the content type and content length of the response. |
| `doOptions` | Called in response to an HTTP *`options`* request. This returns information to the client indicating the HTTP options supported by the server, such as the version of HTTP (1.0 or 1.1) and the request methods the server supports. |
| `doPut` | Called in response to an HTTP *`put`* request. Such a request is normally used to store a file on the server. This may not be available on some servers, because of its inherent security risks (e.g., the client could place an executable application on the server, which, if executed, could damage the server—perhaps by deleting critical files or occupying resources). |
| `doTrace` | Called in response to an HTTP *`trace`* request. Such a request is normally used for debugging. The implementation of this method automatically returns an HTML document to the client containing the request header information (data sent by the browser as part of the request). |

**Fig. 24.2** Other methods of class `HttpServlet`.

# 24.2.3 `HttpServletRequest` Interface

- Web server
  - creates an **`HttpServletRequest`** object
  - passes it to the servlet's **`service`** method
- **`HttpServletRequest`** object contains the request from the client

# 24.2.3 `HttpServletRequest` Interface (Cont.)

| Method | Description |
|---|---|
| `String getParameter( String name )` | |
| | Obtains the value of a parameter sent to the servlet as part of a `get` or `post` request. The `name` argument represents the parameter name. |
| `Enumeration getParameterNames( )` | |
| | Returns the names of all the parameters sent to the servlet as part of a `post` request. |
| `String[] getParameterValues( String name )` | |
| | For a parameter with multiple values, this method returns an array of strings containing the values for a specified servlet parameter. |
| `Cookie[] getCookies()` | |
| | Returns an array of `Cookie` objects stored on the client by the server. `Cookie` objects can be used to uniquely identify clients to the servlet. |
| `HttpSession getSession( boolean create )` | |
| | Returns an `HttpSession` object associated with the client's current browsing session. This method can create an `HttpSession` object (`true` argument) if one does not already exist for the client. `HttpSession` objects are used in similar ways to `Cookie`s for uniquely identifying clients. |

**Fig. 24.3** Some methods of interface `HttpServletRequest`.

# 24.2.4 `HttpServletResponse` Interface

- Web server
  - creates an **`HttpServletResponse`** object
  - passes it to the servlet's **`service`** method

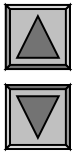
# 24.2.4 `HttpServletResponse` Interface (Cont.)

| Method | Description |
|---|---|
| `void addCookie( Cookie cookie )` | |
| | Used to add a `Cookie` to the header of the response to the client. The `Cookie`'s maximum age and whether `Cookie`s are enabled on the client determine if `Cookie`s are stored on the client. |
| `ServletOutputStream getOutputStream()` | |
| | Obtains a byte-based output stream for sending binary data to the client. |
| `PrintWriter getWriter()` | |
| | Obtains a character-based output stream for sending text data to the client. |
| `void setContentType( String type )` | |
| | Specifies the MIME type of the response to the browser. The MIME type helps the browser determine how to display the data (or possibly what other application to execute to process the data). For example, MIME type `"text/html"` indicates that the response is an HTML document, so the browser displays the HTML page. |

**Fig. 24.4** Some methods of interface `HttpServletResponse`.

# 24.3 Handling HTTP `get` Requests

- **`get`** request
  - Retrieve the content of a URL
- Example: **`WelcomeServlet`**
  - a servlet handles HTTP **`get`** requests



Outline

**WelcomeServlet**

**Lines 4-5**

**-42**

```
1  // Fig. 24.5: WelcomeServlet.java
2  // A simple servlet to process get requests.
3
4  import javax.servlet.*;
5  import javax.servlet.http.*;
6  import java.io.*;
7
8  public class WelcomeServlet extends HttpServlet {
9
10    // process "get" requests from clients
11    protected void doGet( HttpServletRequest request,
12       HttpServletResponse response )
13          throws ServletException, IOException
14    {
15       response.setContentType( "text/html" );
16       PrintWriter out = response.getWriter();
17
18       // send XHTML page to client
19
20       // start XHTML document
21       out.println( "<?xml version = \"1.0\"?>" );
22
23       out.println( "<!DOCTYPE html PUBLIC \"-//W3C//DTD " +
24          "XHTML 1.0 Strict//EN\" \"http://www.w3.org" +
25          "/TR/xhtml1/DTD/xhtml1-strict.dtd\">" );
26
```

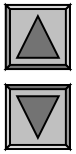
Import the javax.servlet and javax.servlet.http packages.

Extends HttpServlet to handle HTTP get requests

Override method doGet to provide custom get request processing.

Uses the response object's getWriter method to obtain a reference to the PrintWriter object that enables the servlet to send content to the client

Create the XHTML document by writing strings with the out object's println method.



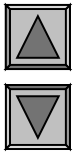
Outline

**WelcomeServlet**

**Line 41**

```
27      out.println( "<html xmlns = \"http://www.w3.org/1999/xhtml\">" );
28
29      // head section of document
30      out.println( "<head>" );
31      out.println( "<title>A Simple Servlet Example</title>" );
32      out.println( "</head>" );
33
34      // body section of document
35      out.println( "<body>" );
36      out.println( "<h1>Welcome to Servlets!</h1>" );
37      out.println( "</body>" );
38
39      // end XHTML document
40      out.println( "</html>" );
41      out.close();  // close stream to complete the page
42    }
43  }
```

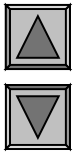
Closes the output stream, flushes the output buffer and sends the information to the client.



Outline

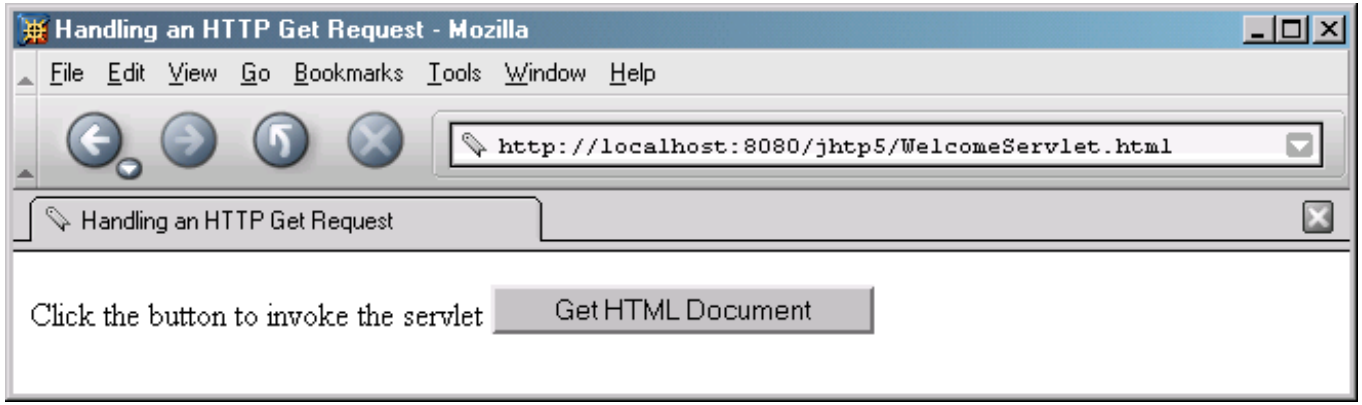
**WelcomeServlet.html**

```
<?xml version = "1.0"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
  "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">

<!-- Fig. 24.6: WelcomeServlet.html -->

<html xmlns = "http://www.w3.org/1999/xhtml">
<head>
  <title>Handling an HTTP Get Request</title>
</head>

<body>
  <form action = "/jhtp5/welcome1" method = "get">

    <p><label>Click the button to invoke the servlet
      <input type = "submit" value = "Get HTML Document" />
    </label></p>

  </form>
</body>
</html>
```





**Program output**




# 24.3.1 Setting Up the Apache Tomcat Server

- Download Tomcat (version 4.1.12)
  - jakarta.apache.org/builds/jakarta-tomcat-4.0/release/v4.1.12/bin/
- Define environment variables
  - JAVA_HOME
  - CATALINA_HOME
- Start the Tomcat server
  - startup.bat
- Connect to the Tomcat server using a Web browser
  - http://localhost:8080/

# 24.3.1 Setting Up the Apache Tomcat Server (Cont.).


Fig. 24.7 Tomcat documentation home page. (Courtesy of The Apache Software Foundation.)

# 24.3.2 Deploying a Web Application

- Web applications
  - JSPs, servlets and their supporting files
- Deploying a Web application
  - Directory structure
    - Context root
  - Web application archive file (WAR file)
  - Deployment descriptor
    - **`web.xml`**

# 24.3.2 Deploying a Web Application (Cont.)

| Directory | Description |
|---|---|
| context root | This is the root directory for the Web application. All the JSPs, HTML documents, servlets and supporting files such as images and class files reside in this directory or its subdirectories. The name of this directory is specified by the Web application creator. To provide structure in a Web application, subdirectories can be placed in the context root. For example, if your application uses many images, you might place an images subdirectory in this directory. The examples of this chapter use `jhtp5` as the context root. |
| `WEB-INF` | This directory contains the Web application *deployment descriptor* (`web.xml`). |
| `WEB-INF/classes` | This directory contains the servlet class files and other supporting class files used in a Web application. If the classes are part of a package, the complete package directory structure would begin here. |
| `WEB-INF/lib` | This directory contains Java archive (JAR) files. The JAR files can contain servlet class files and other supporting class files used in a Web application. |

**Fig. 24.8** Web application standard directories.

# Outline

```
1   <!DOCTYPE web-app PUBLIC \
2     "-//Sun Microsystems, Inc.//DTD Web Application 2.2//EN"
3     "http://java.sun.com/j2ee/dtds/web-app_2_2.dtd">
4
5   <web-app>
6
7     <!-- General descripti
8     <display-name>
9       Java How to Program JSP
10      and Servlet Chapter Examples
11    </display-name>
12
13    <description>
14      This is the Web applicatio
15      demonstrate our JSP and
16    </description>
17
18    <!-- Servlet definitions -->
19    <servlet>
20      <servlet-name>welcome1</servlet-name>
21
22      <description>
23        A simple servlet that har
24      </description>
25
```

Element web-app defines the configuration of each servlet in the Web application and the servlet mapping for each servlet.

Element display-name specifies a name that can be displayed to the administrator of the server on which the Web application is installed.

Element description specifies a description of the Web application that might be displayed to the administrator of the server.

Element servlet-name is the name for the servlet.

Element description specifies a description for this particular servlet.

**web.xml**

**Lines 5-37**

**Lines 8-11**

**Lines 13-16**

**Lines 19-29**

**Line 20**

**Lines 22-24**

**26-28**

Outline

web.xml

```
26      <servlet-class>
27        WelcomeServlet
28      </servlet-class>
29    </servlet>
30
31    <!-- Servlet mappings -->
32    <servlet-mapping>
33      <servlet-name>welcome1</servlet-name>
34      <url-pattern>/welcome1</url-pattern>
35    </servlet-mapping>
36
37  </web-app>
```

Element servlet-class specifies compiled servlet's fully qualified class name.

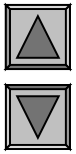
Element servlet-mapping specifies servlet-name and url-pattern elements.

es 26-28

es 32-35

# 24.4 Handling HTTP `get` Requests Containing Data

- Servlet **`WelcomeServlet2`**
  - Responds to a **`get`** request that contains data



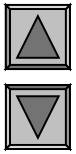
## Outline

```
1   // Fig. 24.11: WelcomeServlet2.java
2   // Processing HTTP get requests containing data.
3
4   import javax.servlet.*;
5   import javax.servlet.http.*;
6   import java.io.*;
7
8   public class WelcomeServlet2 extends HttpServlet {
9
10    // process "get" request from client
11    protected void doGet( HttpServletRequest request,
12      HttpServletResponse response )
13        throws ServletException, IOException
14    {
15      String firstName = request.getParameter( "firstname" );
16
17      response.setContentType( "text/html" );
18      PrintWriter out = response.getWriter();
19
20      // send XHTML document to client
21
22      // start XHTML document
23      out.println( "<?xml version = \"1.0\"?>" );
24
```

**WelcomeServlet2 responds to a get request that contains data.**

**Line 15**

The `request` object's `getParameter` method receives the parameter name and returns the corresponding `String` value.



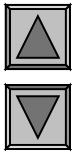
## Outline

```
25      out.println( "<!DOCTYPE html PUBLIC \"-//W3C//DTD " +
26        "XHTML 1.0 Strict//EN\" \"http://www.w3.org" +
27        "/TR/xhtml1/DTD/xhtml1-strict.dtd\">" );
28
29      out.println( "<html xmlns = \"http://www.w3.org/1999/xhtml\">" );
30
31      // head section of document
32      out.println( "<head>" );
33      out.println(
34        "<title>Processing get requests with data</title>" );
35      out.println( "</head>" );
36
37      // body section of document
38      out.println( "<body>" );
39      out.println( "<h1>Hello " + firstName + ",<br />" );
40      out.println( "Welcome to Servlets!</h1>" );
41      out.println( "</body>" );
42
43      // end XHTML document
44      out.println( "</html>" );
45      out.close();  // close stream to complete the page
46    }
47  }
```

**WelcomeServlet2 responds to a get request that contains data.**

**Line 39**

Uses the result of line 16 as part of the response to the client.



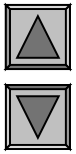
## Outline

```
1   <?xml version = "1.0"?>
2   <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
3     "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
4
5   <!-- Fig. 24.12: WelcomeServlet2.html -->
6
7   <html xmlns = "http://www.w3.org/1999/xhtml">
8   <head>
9     <title>Processing get requests with data</title>
10  </head>
11
12  <body>
13    <form action = "/jhtp5/welcome2" method = "get">
14
15      <p><label>
16        Type your first name and press the Submit button
17        <br /><input type = "text" name = "firstname" />
18        <input type = "submit" value = "Submit" />
19      </p></label>
20
21    </form>
22  </body>
23  </html>
```

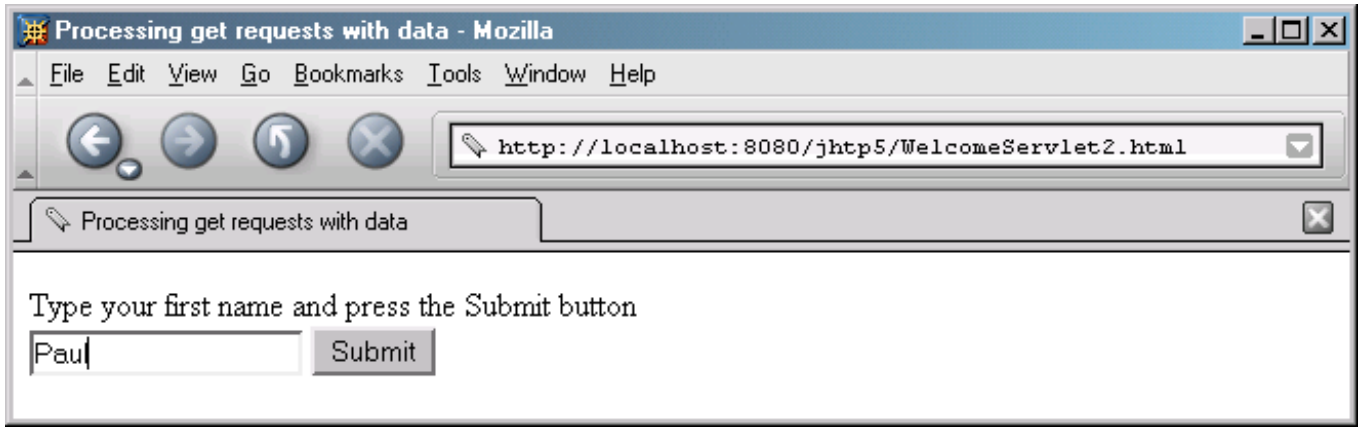
HTML document in which the form's action invokes WelcomeServlet2 through the alias welcome2 specified in web.xml.

Line 17

Get the first name from the user.



# Outline




HTML document in which the form's action invokes WelcomeServlet2 through the alias welcome2 specified in web.xml.

Program output

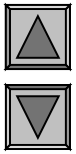
# 24.4 Handling HTTP `get` Requests Containing Data (Cont.)

| Descriptor element | Value |
|---|---|
| *`servlet` element* | |
| `servlet-name` | `welcome2` |
| `description` | `Handling HTTP get requests with data.` |
| `servlet-class` | `WelcomeServlet2` |
| *`servlet-mapping` element* | |
| `servlet-name` | `welcome2` |
| `url-pattern` | `/welcome2` |

**Fig. 24.13** Deployment descriptor information for servlet `WelcomeServlet2`.

# 24.5 Handling HTTP post Requests

- HTTP post request
  - Post data from an HTML form to a server-side form handler
  - Browsers cache Web pages
- Servlet WelcomeServlet3
  - Responds to a **`post`** request that contains data

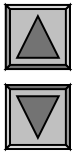


Outline

```
// Fig. 24.14: WelcomeServlet3.java
// Processing post requests containing data.

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

public class WelcomeServlet3 extends HttpServlet {

   // process "post" request from client
   protected void doPost( HttpServletRequest request,
      HttpServletResponse response )
         throws ServletException, IOException
   {
      String firstName = request.getParameter( "firstname" );

      response.setContentType( "text/html" );
      PrintWriter out = response.getWriter();

      // send XHTML page to client

      // start XHTML document
      out.println( "<?xml version = \"1.0\"?>" );

```

WelcomeServlet3 responds to a post request that contains data.

Declare a doPost method to responds to post requests.



Outline

**WelcomeServlet3.java**

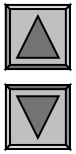
```
      out.println( "<!DOCTYPE html PUBLIC \"-//W3C//DTD " +
        "XHTML 1.0 Strict//EN\" \"http://www.w3.org" +
        "/TR/xhtml1/DTD/xhtml1-strict.dtd\">" );

      out.println( "<html xmlns = \"http://www.w3.org/1999/xhtml\">" );

      // head section of document
      out.println( "<head>" );
      out.println(
        "<title>Processing post requests with data</title>" );
      out.println( "</head>" );

      // body section of document
      out.println( "<body>" );
      out.println( "<h1>Hello " + firstName + ",<br />" );
      out.println( "Welcome to Servlets!</h1>" );
      out.println( "</body>" );

      // end XHTML document
      out.println( "</html>" );
      out.close();  // close stream to complete the page
    }
  }
```

# Outline

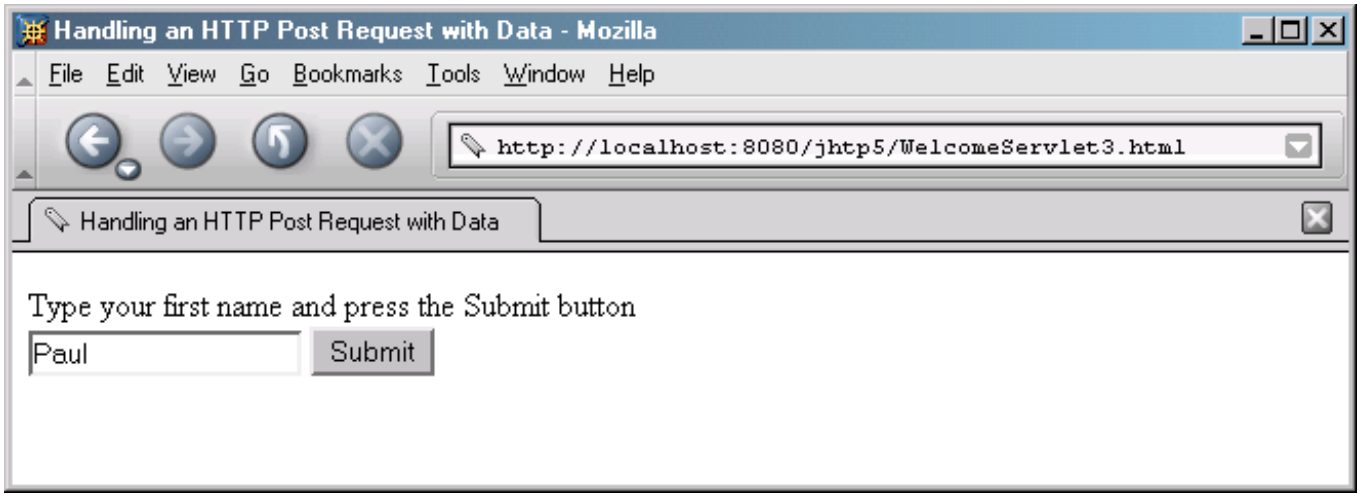
```
<?xml version = "1.0"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
  "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">

<!-- Fig. 24.15: WelcomeServlet3.html -->

<html xmlns = "http://www.w3.org/1999/xhtml">
<head>
  <title>Handling an HTTP Post Request with Data</title>
</head>

<body>
  <form action = "/jhtp5/welcome3" method = "post">

    <p><label>
      Type your first name and press the Submit button
      <br /><input type = "text" name = "firstname" />
      <input type = "submit" value = "Submit" />
    </label></p>

  </form>
</body>
</html>
```

**HTML document in which the form's action invokes WelcomeServlet3 through the alias welcome3 specified in**

Provide a form in which the user can input a name in the text input element firstname, then click the Submit button to invoke WelcomeServlet3.








HTML document in which the form's action invokes WelcomeServlet3 through the alias welcome3 specified in web.xml.


Program output

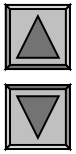
# 24.5 Handling HTTP `post` Requests (Cont.)

| Descriptor element | Value |
|---|---|
| *`servlet` element* | |
| `servlet-name` | `welcome3` |
| `description` | `Handling HTTP post requests with data.` |
| `servlet-class` | `WelcomeServlet3` |
| *`servlet-mapping` element* | |
| `servlet-name` | `welcome3` |
| `url-pattern` | `/welcome3` |

**Fig. 24.16** Deployment descriptor information for servlet `WelcomeServlet3`.

# 24.6 Redirecting Requests to Other Resources

- Servlet **`RedirectServlet`**
  - Redirects the request to a different resource



## Outline

```
1   // Fig. 24.17: RedirectServlet.java
2   // Redirecting a user to a different Web page.
3
4   import javax.servlet.*;
5   import javax.servlet.http.*;
6   import java.io.*;
7
8   public class RedirectServlet extends HttpServlet {
9
10    // process "get" request from client
11    protected void doGet( HttpServletRequest request,
12      HttpServletResponse response )
13        throws ServletException, IOException
14    {
15      String location = request.getParameter( "page" );
16
17      if ( location != null )
18
19        if ( location.equals( "deitel" ) )
20          response.sendRedirect( "http://www.deitel.com" );
21        else
22          if ( location.equals( "welcome1" ) )
23            response.sendRedirect( "welcome1" );
24
```

**RedirectServlet redirecting requests to other resources.**

**Line 15**

Obtains the page parameter from the request.

**Lines 19-23**

Determine if the value is either "deitel" or "wel

**Line 23**

Redirects the request to www.deitel.com.

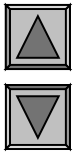
Redirects the request to the servlet WelcomeServlet.

```
25      // code that executes only if this servlet
26      // does not redirect the user to another page
27
28      response.setContentType( "text/html" );
29      PrintWriter out = response.getWriter();
30
31      // start XHTML document
32      out.println( "<?xml version = \"1.0\"?>" );
33
34      out.println( "<!DOCTYPE html PUBLIC \"-//W3C//DTD " +
35         "XHTML 1.0 Strict//EN\" \"http://www.w3.org" +
36         "/TR/xhtml1/DTD/xhtml1-strict.dtd\">" );
37
38      out.println(
39         "<html xmlns = \"http://www.w3.org/1999/xhtml\">" );
40
41      // head section of document
42      out.println( "<head>" );
43      out.println( "<title>Invalid page</title>" );
44      out.println( "</head>" );
45
46      // body section of document
47      out.println( "<body>" );
48      out.println( "<h1>Invalid page requested</h1>" );
```

Output a Web page indicating that an invalid request was made if method sendRedirect is not called.

requests to other resources.

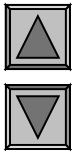
Lines 28-55



Outline

```
      out.println( "<p><a href = " +
        "\"servlets/RedirectServlet.html\">" );
      out.println( "Click here to choose again</a></p>" );
      out.println( "</body>" );

      // end XHTML document
      out.println( "</html>" );
      out.close();  // close stream to complete the page
   }
}
```

RedirectServlet redirecting requests to other resources.



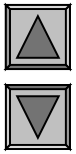
# Outline

```
1  <?xml version = "1.0"?>
2  <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
3    "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
4
5  <!-- Fig. 24.18: RedirectServlet.html -->
6
7  <html xmlns = "http://www.w3.org/1999/xhtml">
8  <head>
9    <title>Redirecting a Request to Another Site</title>
10 </head>
11
12 <body>
13   <p>Click a link to be redirected to the appropriate page</p>
14   <p>
15   <a href = "/jhtp5/redirect?page=deitel">
16     www.deitel.com</a><br />
17   <a href = "/jhtp5/redirect?page=welcome1">
18     Welcome servlet</a>
19   </p>
20 </body>
21 </html>
```

**RedirectServlet.html document to demonstrate redirecting requests to other resources.**

**Lines 15-16**

Provide hyperlinks that allow the user to invoke the servlet RedirectServlet.



# Outline

**RedirectServlet.html document to demonstrate redirecting requests to other resources.**

**Program output**

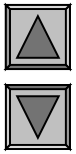

# 24.6 Redirecting Requests to other Resources (Cont.)

| Descriptor element | Value |
|---|---|
| *servlet element* | |
| servlet-name | redirect |
| description | Redirecting to static Web pages and other servlets. |
| servlet-class | com.deitel.jhtp5.servlets.RedirectServlet |
| *servlet-mapping element* | |
| servlet-name | redirect |
| url-pattern | /redirect |

**Fig. 24.19** Deployment descriptor information for servlet RedirectServlet.

# 24.7 Multi-Tier Applications: Using JDBC from a Servlet

- Three-tier distributed applications
  - User interface
  - Business logic
  - Database access
- Web servers often represent the middle tier
- Three-tier distributed application example
  - SurveyServlet
  - Survey.html
  - Cloudscape database



# Outline

```
1  // Fig.  24.20: SurveyServlet.java
2  // A Web-based survey that uses JDBC from a servlet.
3  package com.deitel.jhtp5.servlets;
4
5  import java.io.*;
6  import java.text.*;
7  import java.sql.*;
8  import javax.servlet.*;
9  import javax.servlet.http.*;
10
11 public class SurveyServlet extends HttpServlet {
12    private Connection connection;
13    private Statement statement;
14
15    // set up database connection and create SQL statement
16    public void init( ServletConfig config ) throws ServletException
17    {
18       // attempt database connection and create Statements
19       try {
20          System.setProperty( "db2j.system.home",
21             config.getInitParameter( "databaseLocation" ) );
22
23          Class.forName( config.getInitParameter( "databaseDriver" ) )
24          connection = DriverManager.getConnection(
25             config.getInitParameter( "databaseName" ) );
```

Servlets are initialized by overriding method init.

Specify database location

Loads the database driver.

Attempt to c o the animalsurvey database.

**SurveyServlet.java**
**Multi-tier Web-based survey using XHTML, servlets and JDBC.**

**Lines 16-38**

**Lines 20-21**

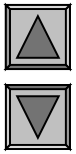
**Line 23**

```
26
27        // create Statement to query database
28        statement = connection.createStatement();
29      }
30
31      // for any exception throw an UnavailableException to
32      // indicate that the servlet is not currently available
33      catch ( Exception exception ) {
34        exception.printStackTrace();
35        throw new UnavailableException(exception.getMessage());
36      }
37
38    }  // end of init method
39
40    // process survey response
41    protected void doPost( HttpServletRequest request,
42      HttpServletResponse response )
43        throws ServletException, IOException
44    {
45      // set up response to client
46      response.setContentType( "text/html" );
47      PrintWriter out = response.getWriter();
48      DecimalFormat twoDigits = new DecimalFormat( "0.00" );
49
```

Create Statement to query database.

surveyServlet.java
Multi-tier Web-based survey using XHTML, servlets and JDBC.

Line 28



# Outline

```
50      // start XHTML document
51      out.println( "<?xml version = \"1.0\"?>" );
52
53      out.println( "<!DOCTYPE html PUBLIC \"-//W3C//DTD " +
54        "XHTML 1.0 Strict//EN\" \"http://www.w3.org" +
55        "/TR/xhtml1/DTD/xhtml1-strict.dtd\">" );
56
57      out.println(
58        "<html xmlns = \"http://www.w3.org/1999/xhtml\">" );
59
60      // head section of document
61      out.println( "<head>" );
62
63      // read current survey response
64      int value =
65        Integer.parseInt( request.getParameter( "animal" ) );
66      String query;
67
68      // attempt to process a vote and display current results
69      try {
70
71        // update total for current surevy response
72        query = "UPDATE surveyresults SET votes = votes + 1 " +
73            "WHERE id = " + value;
74        statement.executeUpdate( query );
75
```

SurveyServlet.java
Multi-tier Web-based survey using XHTML, servlets and JDBC.

Lines 64-65

Obtain the survey response

2-73

Line 74

Create query to update total
y response

Execute query to update total for current survey response

```
76      // get total of all survey responses
77      query = "SELECT sum( votes ) FROM surveyresults";
78      ResultSet totalRS = statement.executeQuery( query );
79      totalRS.next();
80      int total = totalRS.getInt( 1 );
81
82      // get results
83      query = "SELECT surveyoption, votes, id FROM surveyresults " +
84        "ORDER BY id";
85      ResultSet resultsRS = statement.executeQuery( query );
86      out.println( "<title>Thank you!</title>" );
87      out.println( "</head>" );
88
89      out.println( "<body>" );
90      out.println( "<p>Thank you for participating." );
91      out.println( "<br />Results:</p><pre>" );
92
93      // process results
94      int votes;
95
96      while ( resultsRS.next() ) {
97        out.print( resultsRS.getString( 1 ) );
98        out.print( ": " );
99        votes = resultsRS.getInt( 2 );
100       out.print( twoDigits.format(
101         ( double ) votes / total * 100 ) );
102       out.print( "%  responses: " );
103       out.println( votes );
104     }
```

Create query to get total of all survey responses

Execute query to get total of all survey responses

Create query to get survey results

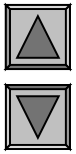
Execute query to get survey results

**Line 77**

**Line 78**

**Lines 83-84**

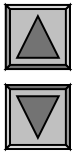
**Line 85**



Outline

```
105
106         resultsRS.close();
107
108         out.print( "Total responses: " );
109         out.print( total );
110
111         // end XHTML document
112         out.println( "</pre></body></html>" );
113         out.close();
114
115      } // end try
116
117      // if database exception occurs, return error page
118      catch ( SQLException sqlException ) {
119         sqlException.printStackTrace();
120         out.println( "<title>Error</title>" );
121         out.println( "</head>" );
122         out.println( "<body><p>Database error occurred. " );
123         out.println( "Try again later.</p></body></html>" );
124         out.close();
125      }
126
127   }  // end of doPost method
128
```

SurveyServlet.java
Multi-tier Web-based survey using XHTML, servlets and JDBC.



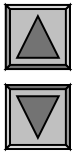
Outline

```
129   // close SQL statements and databa
130   public void destroy()
131   {
132     // attempt to close statements an
133     try {
134       statement.close();
135       connection.close();
136     }
137
138     // handle database exceptions by returning error to client
139     catch ( SQLException sqlException ) {
140       sqlException.printStackTrace();
141     }
142   }
143
144 } // end class SurveyServlet
```

Method destroy closes Statement and database connection.

SurveyServlet.java
Multi-tier Web-based survey using XHTML, servlets and JDBC.

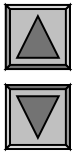
Lines 130-136



Outline

```
<?xml version = "1.0"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
  "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">

<!-- Fig.  24.21: Survey.html -->

<html xmlns = "http://www.w3.org/1999/xhtml">
<head>
  <title>Survey</title>
</head>

<body>
<form method = "post" action = "/jhtp5/animalsurvey">

  <p>What is your favorite pet?</p>

```

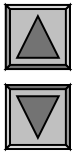
Survey.html document that allows users to submit survey responses to SurveyServlet.



## Outline

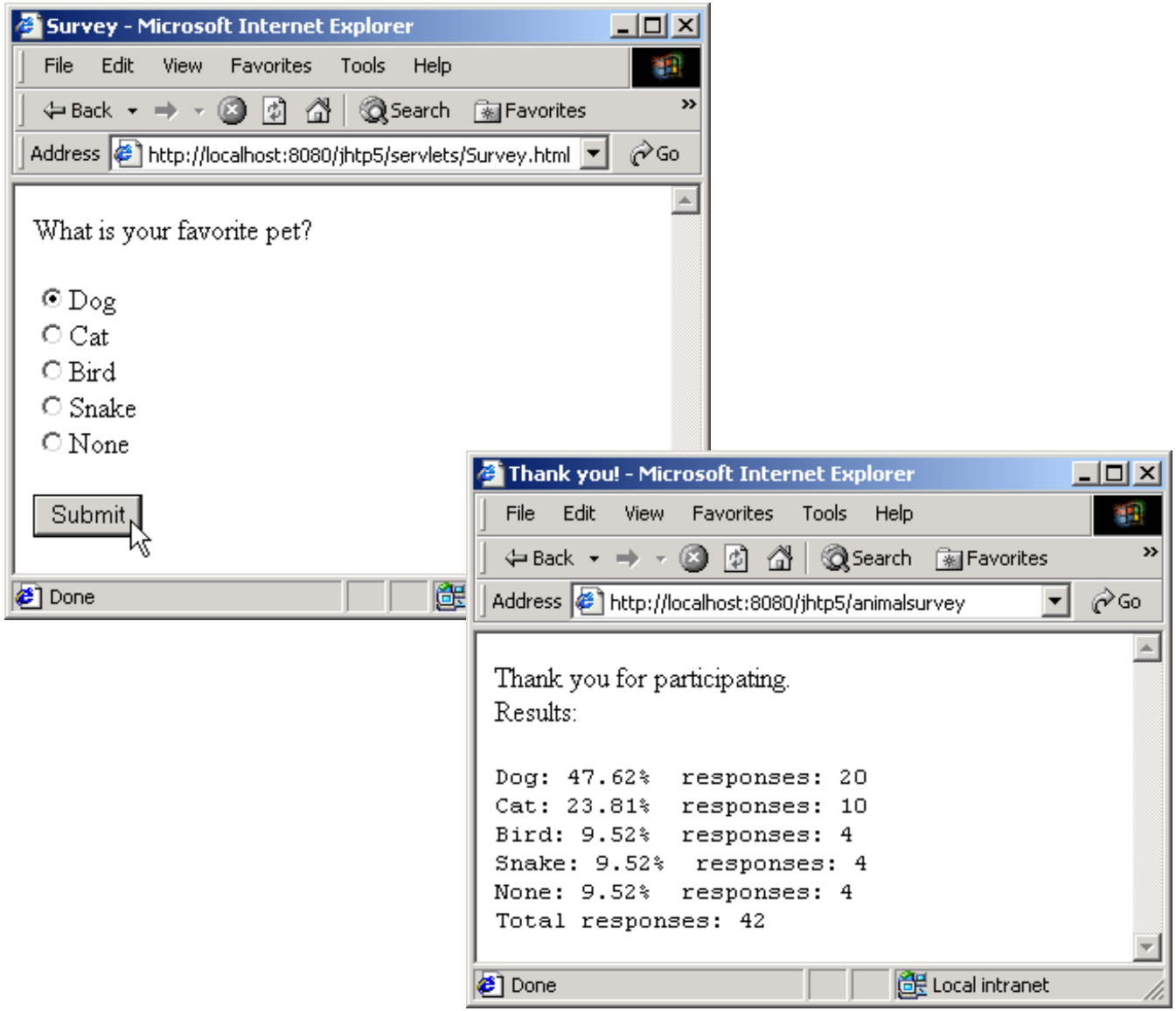
```
17    <p>
18      <input type = "radio" name = "animal"
19        value = "1" />Dog<br />
20      <input type = "radio" name = "animal"
21        value = "2" />Cat<br />
22      <input type = "radio" name = "animal"
23        value = "3" />Bird<br />
24      <input type = "radio" name = "animal"
25        value = "4" />Snake<br />
26      <input type = "radio" name = "animal"
27        value = "5" checked = "checked" />None
28    </p>
29
30    <p><input type = "submit" value = "Submit" /></p>
31
32  </form>
33  </body>
34  </html>
```

Survey.html document that allows users to submit survey responses to SurveyServlet.



# Outline

**Survey.html document that allows users to submit survey responses to SurveyServlet.**



# 24.7 Multi-Tier Applications: Using JDBC from a Servlet (Cont.)

| Descriptor element | Value |
|---|---|
| *`servlet` element* | |
| `servlet-name` | `animalsurvey` |
| `description` | `Connecting to a database from a servlet.` |
| `servlet-class` | `com.deitel.jhtp5.servlets.SurveyServlet` |
| `init-param` | |
| `param-name` | `databaseLocation` |
| `param-value` | `C:/CloudScape_5.0` |
| `init-param` | |
| `param-name` | `databaseDriver` |
| `param-value` | `com.ibm.db2j.jdbc.DB2jDriver` |
| `init-param` | |
| `param-name` | `databaseName` |
| `param-value` | `jdbc:db2j:animalsurvey` |
| *`servlet-mapping` element* | |
| `servlet-name` | `animalsurvey` |
| `url-pattern` | `/animalsurvey` |

**Fig. 24.22** Deployment descriptor information for servlet `SurveyServlet`.

# 24.8 Internet and World Wide Web Resources

- Servlet resources
  - java.sun.com/products/servlet/index.html
  - www.servlets.com
  - www.servletsource.com
  - www.servletforum.com
  - www.coolservlets.com