

**Appendix  
C**

**WORKSHOP**

*Get on the  
Fast Track!*



TM

**SYS-ED/  
Computer  
Education  
Techniques, Inc.**

---

**1 Preliminary Assessment**

**Objectives**

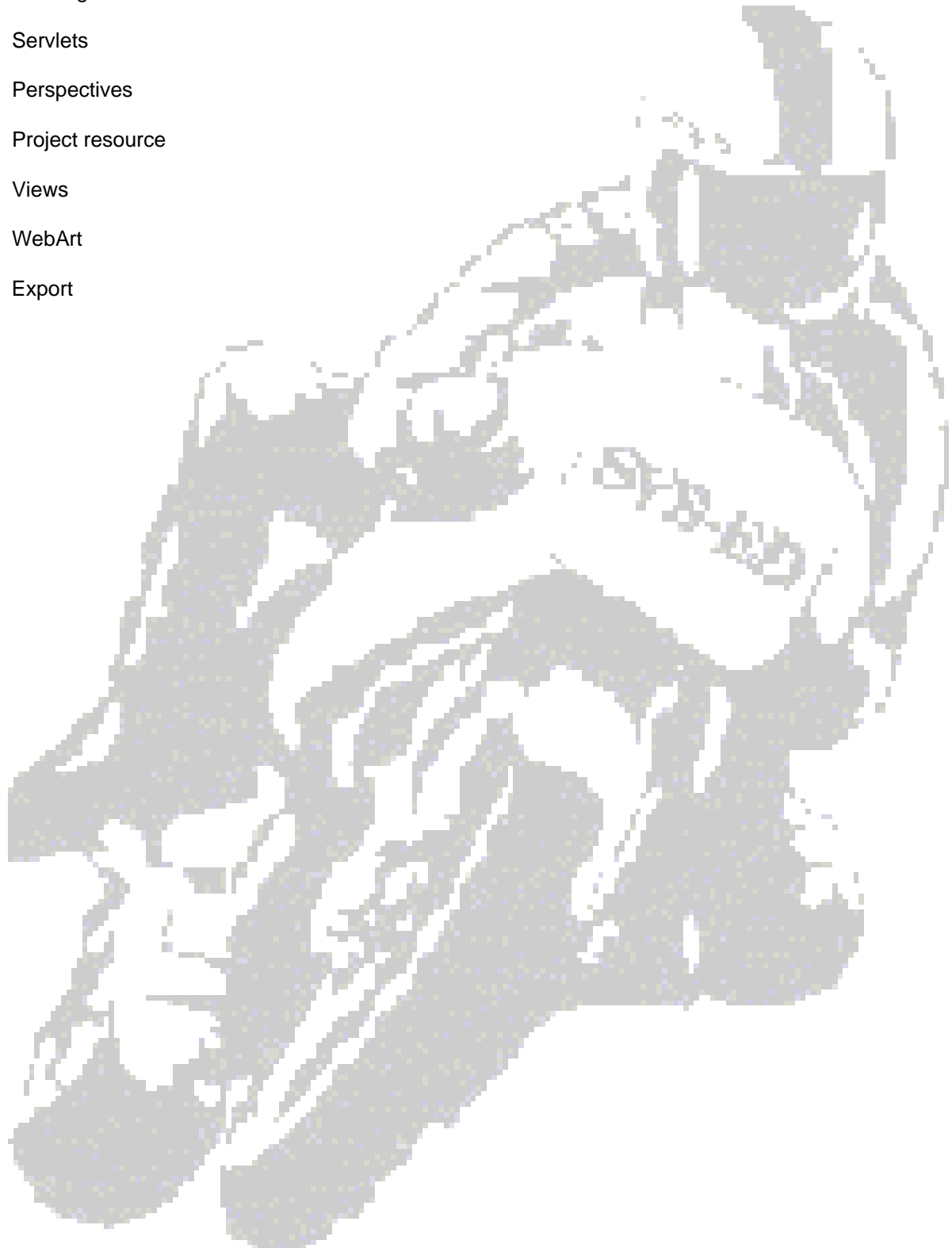
- Specify key components of WSAD.

**Questions**

1. \_\_\_\_\_ tools are used for reorganizing Java classes.
2. \_\_\_\_\_ tools are used to fine-tune applications by detecting and analyzing performance problems.
3. Server applications that execute within a web application are called \_\_\_\_\_.
4. Combinations of views and editors are known as \_\_\_\_\_.
5. \_\_\_\_\_ mappings control how the project resources map to the local file system.
6. \_\_\_\_\_ provide alternative presentations of ways of navigating through the information in the workbench.
7. \_\_\_\_\_ Designer to create graphic titles, logos, buttons, and photo frames.
8. To bring in a JAR/EAR/WAR file into a project use the \_\_\_\_\_ tool.

**Answers**

1. Refactoring
2. Profiling
3. Servlets
4. Perspectives
5. Project resource
6. Views
7. WebArt
8. Export



---

## 2 Workbench

Create a simple application, debug the program, and display various views.

### Objectives

- Create a project.
- Create a Java application.
- Build and run the application.
- Debug the application.

### Exercises

1. Create a Java project.
2. Create a new Java class and replace the generated program with the following:

```
import javax.swing.*;

public class InvoiceApp{

    public static void main(String[] args){
        String choice = "";
        while (!(choice.equalsIgnoreCase("x"))){ // begin while loop
            String inputString = JOptionPane.showInputDialog(
                "Enter order total: ");
            double orderTotal = Double.parseDouble(inputString);
            double discountAmount = 0;
            if (orderTotal >= 100)
                discountAmount = orderTotal * .2;
        }
    }
}
```

```
        else
            discountAmount = orderTotal * .1;
        double invoiceTotal = orderTotal - discountAmount;
        String message = "Order total: " + orderTotal + "\n"
            + "Discount amount: " + discountAmount + "\n"
            + "Invoice total: " + invoiceTotal + "\n\n"
            + "To continue, press Enter.\n"
            + "To exit, enter 'x': ";
        choice = JOptionPane.showInputDialog(message);
    } // end while loop
    System.exit(0);
}
}
```

3. Compile and build the above program.  
Run the program after all errors have been corrected.
4. Display the outline view.
5. Add a new high priority task with a description of "Eat a big lunch".
6. Set a breakpoint on a line.  
Run the program and stop on the breakpoint.  
Display the contents of the `orderTotal` variable.

---

### **3 Java Applications**

Use the workbench and change preferences.

#### **Objectives**

- Display source code options.
- Set preferences.

#### **Exercises**

1. Open the Java application from the previous exercise.
  - To display the selected Java file in a single element view, click on the Show Source of Selected Element Only button in the workbench toolbar.
  - To display the selected Java file in a whole (non-segmented) view, click on the Show Source of Selected Element Only button in the workbench toolbar.
2. Change the following preferences:
  - Color of the console.
  - Turn on the preference for a new line before every opening brace.
  - Display the editor tabs on the bottom.
3. Close editors after six edit windows are open.
4. Increase the font size of text to 12 points.

---

## 4 Web Project

Code a HTML page and servlet which will allow the user to enter in name and e-mail information that is passed to a servlet. The servlet will perform basic error checking and store the information. The data will then be displayed in a table on the browser.

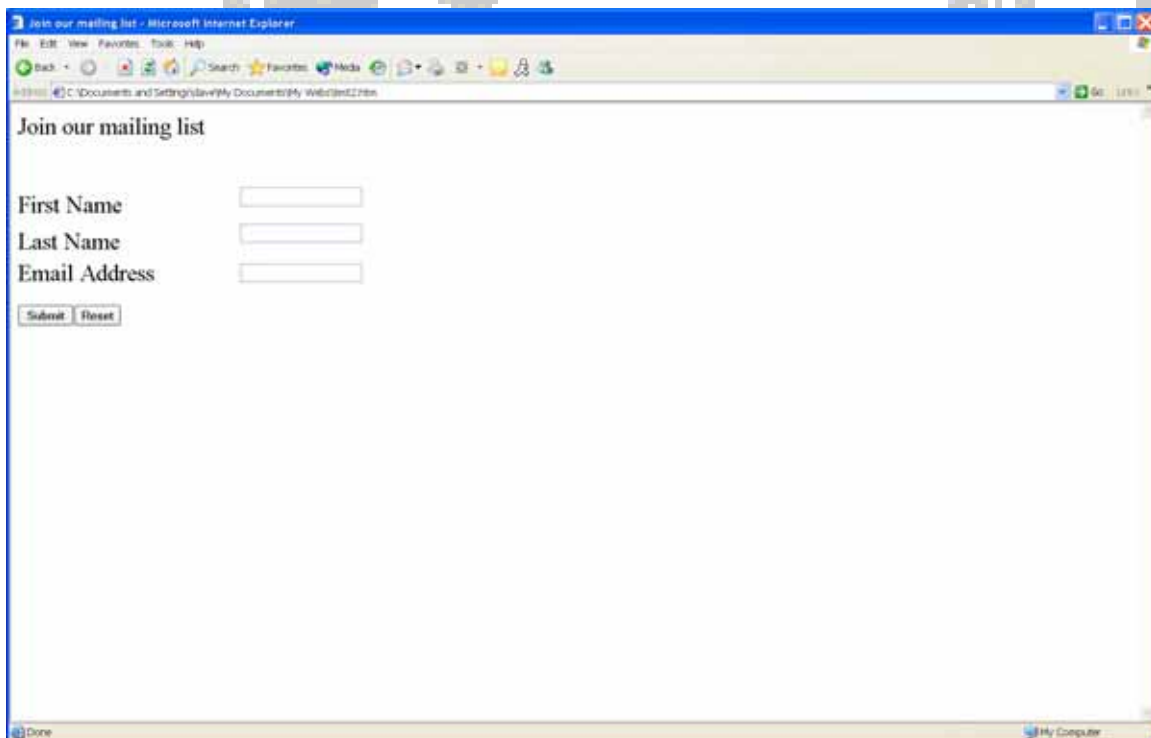
### Objectives

- Code a servlet to process data from a web form.
- Compile the servlet.
- Write a request to the servlet.
- Code a thread safe servlet.

### Exercises

1. Create the user interface for this application. The form should use the Get method.

Use HTML to code the following page:



The screenshot shows a Microsoft Internet Explorer window with the title 'Join our mailing list - Microsoft Internet Explorer'. The address bar shows the local file path 'C:\Documents and Settings\jave\My Documents\My WebSite\2.htm'. The page content includes the heading 'Join our mailing list' followed by three text input fields labeled 'First Name', 'Last Name', and 'Email Address'. Below the fields are two buttons: 'Submit' and 'Reset'.

2. Write a servlet that will run on the WSAD internal server. The source code, class and html are stored in a directory structure that uses the J2EE guidelines.
3. Code the servlet which is expecting a Get and Post method. This will affect the method that is being used to code the servlet.
4. The servlet should accept the information from the front end application and store each value passed into an thread safe variable.
5. All three fields are required: First Name, Last Name and Email address. If any field(s) are missing, display an appropriate message in the browse. Allow the user to correct the problem.
6. Once all fields are entered, store the information into a collection that will be available in a session. Access all entries in the collection and display them in the browser.
7. Test the program using the browser. Run the HTML that was created in step 1.
8. Change the html to use the Post method. Test the program using the browser.
9. Test the servlet by requesting the servlet directly in the browser. Do not use the html.

After the application is working, code the web.xml control file.

## 4.1 JSP Projects

Code a HTML page and two JSP programs that allow the user to enter in name and e-mail information which will be passed to a servlet. The JSP program will perform error checking and store the information. The data will then be displayed in a table on the browser.

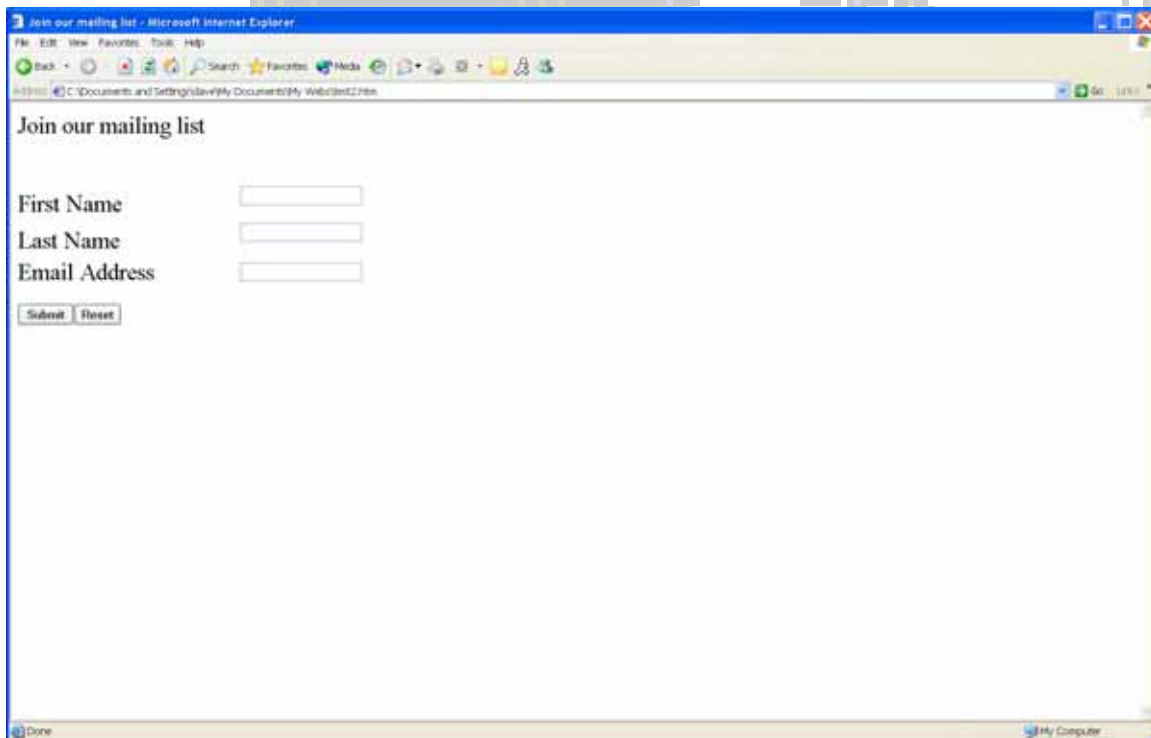
### Objectives

- Code a JSP program to process data from a web form.
- Write a request to the servlet.
- Write the code for the JSP application to call another JSP program.

### Exercises

1. Create the user interface for this application. The form should use the Get method.

Use HTML to code the following page:



The screenshot shows a Microsoft Internet Explorer browser window displaying a web page titled "Join our mailing list". The page contains a form with three input fields: "First Name", "Last Name", and "Email Address". Below the input fields are two buttons: "Submit" and "Reset". The browser's address bar shows the file path: "C:\Documents and Settings\slav\My Documents\My WebSite\2.htm".

2. Write a JSP program that runs on the WebSphere server. The source code, class, and html are to be stored in a directory structure that uses J2EE guidelines.
3. Code the JSP program.
4. The JSP program should accept the information from the front end application and store each value passed into a thread safe variable.
5. All three fields are required: First Name, Last Name and Email address. If any fields are missing, display an appropriate message in the browser. Allow the user to correct the problem.
6. Once all fields are entered, store the information into a collection that will be available in a session. In addition, store all three form fields in a cookie. Invoke another JSP application. Access all entries in the collection and display all the entries in the browser.
7. Test the program using the browser. Run the HTML that was created in step 1.
8. Change the html to use the Post method. Test the program using the browser.
9. Test the servlet by requesting the JSP directly in the browser. Do not use the html.

After the application is working, code the web.xml control file.

---

## **5 Web Pages and Database: JDBC**

Use the JSP application from the previous exercise to update the user provided data and store the information in a database. This exercise can only be performed when the appropriate database has been installed: Oracle, DB2, SQL Server, etc.

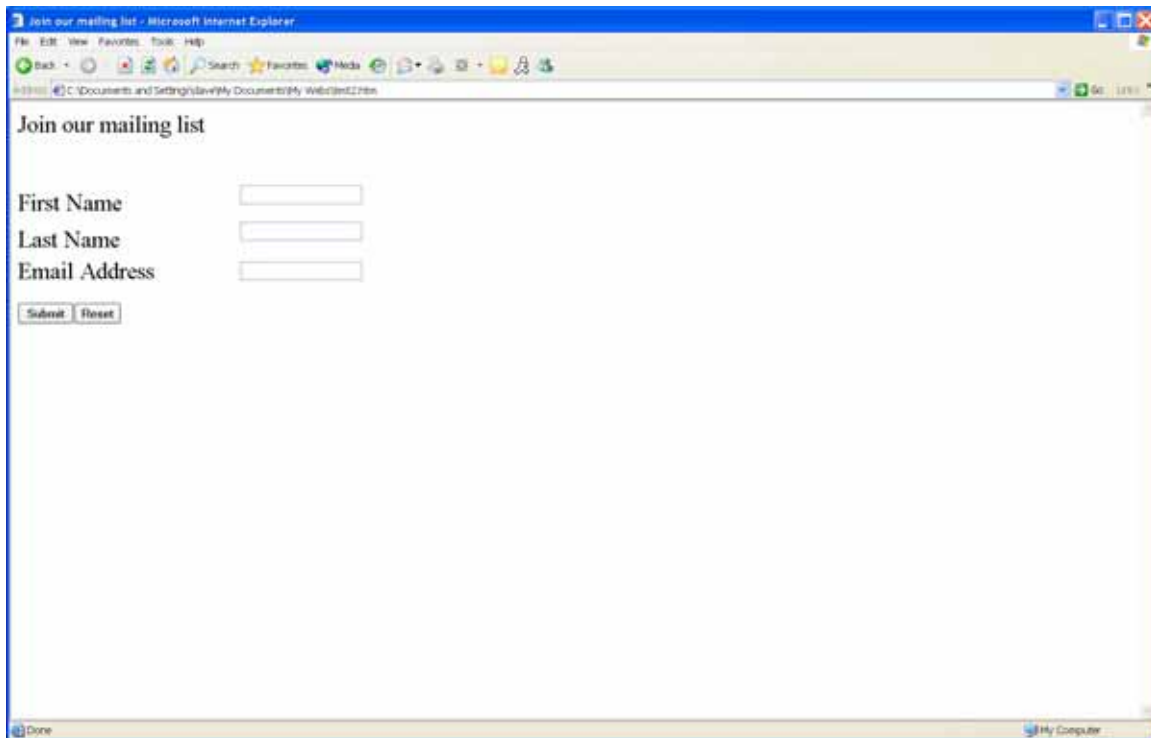
### **Objectives**

- Load a driver for a database; the database will be provided.
- Connect and disconnect from a database.
- Update the database via a JSP program.

### **Exercises**

1. Create the EMailList table in the database that contains the following columns:  
ID  
Firstname  
Lastname  
EmailAddress
2. Create the user interface for this application. The form should use the Post method.

- Use HTML to code the following page:



The screenshot shows a Microsoft Internet Explorer browser window titled "Join our mailing list - Microsoft Internet Explorer". The address bar shows the local file path: "C:\Documents and Settings\lav\My Documents\My Web Sites\2.htm". The page content includes the heading "Join our mailing list" followed by three input fields labeled "First Name", "Last Name", and "Email Address". Below the input fields are two buttons: "Submit" and "Reset".

- Write a JSP program that runs on the WAS server. The source code, class and html are to be stored in a directory structure which uses the guidelines for J2EE.
- Code the JSP program.
- The JSP program should accept the information from the front end application and store each value passed into a thread safe variable.
- All three fields are required: First Name, Last Name and Email address. If any fields are missing, display an appropriate message in the browse. Allow the user to correct the problem.

8. Once all the fields are entered, insert the data into the EMailList table.
  
9. Test the program using the browser.  
Run the HTML that was created in step 1.

After the application is working, code the web.xml control file.



### 5.1 Web Pages and Database: Manipulating the Resultset

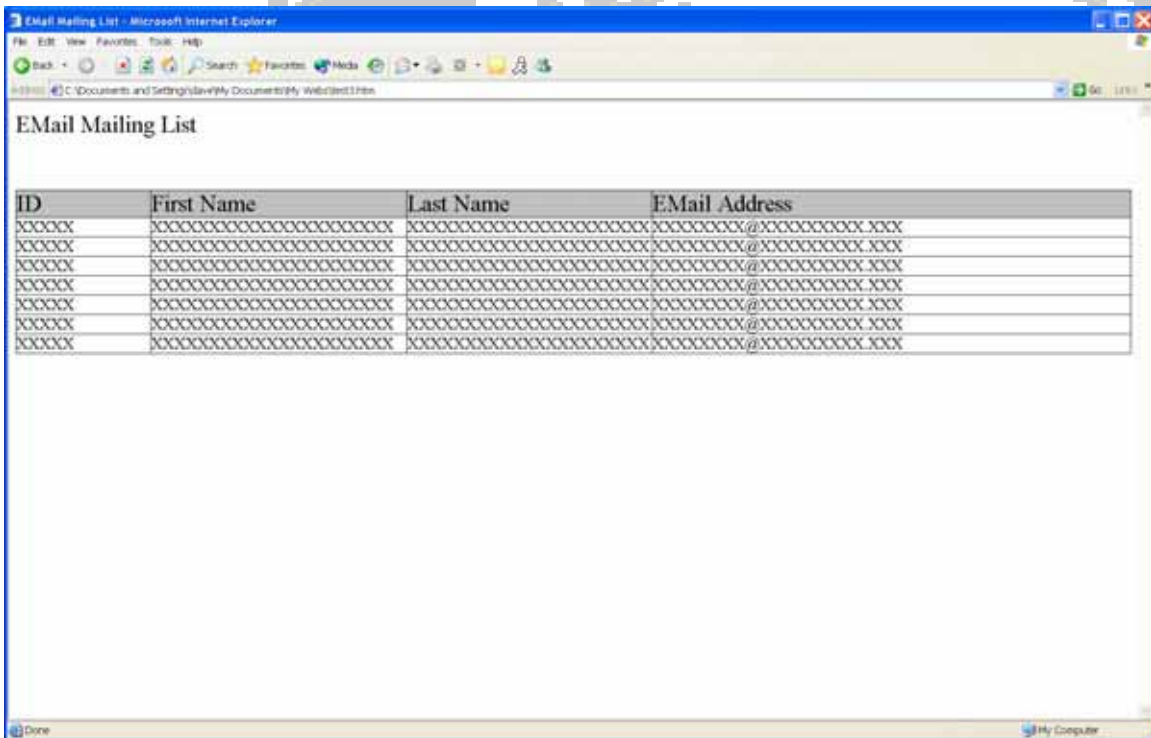
Retrieve all rows in the EMailList table and generate the HTML to display the information. The JDBC exercise must be completed in order to perform this exercise.

#### Objectives

- Determine the number of rows in a table.
- Create a resultset and perform cursor manipulation.

#### Exercise

1. Create the user interface for this application.



---

## **6 Deploying Web Applications**

Create an EAR file with an application and install the application into the server.

### **Objectives**

- Use the Export feature to create an EAR.
- Use the Administration console to install an application.

### **Exercises**

1. Export the application from the previous exercises to an EAR file using the EAR file export wizard.
2. Start the Application Admin Server console.
3. Expand the tree on the left side of the console to locate Nodes...hostname...Enterprise Applications. Select the Enterprise Applications.
4. Click the Install button displayed on the right side of the console, above the list of installed applications.
5. Follow the instructions to install the application from the EAR.

---

## 7 Profiling

Given a Java application, turn on profiling and tune the program.

### Objectives

- Invoke profiling.
- Monitor application performance.
- View class statistics.
- View method statistics.

### Exercises

1. The instructor will provide a Java application in the form of an EAR. Install and run the application.
2. Invoke the program via the profiler.
3. Determine the following performance statistics from the running program.
  - Time consuming classes and methods.
  - Memory intensive classes and methods.
  - Total base time for all methods defined by the class.
  - Total cumulative time for all methods defined by the class.
  - Memory consumption by each object instance of this type.
  - Number of calls made to each method.

---

## 8 Review Questions

Web services architecture and facilities.

### Objectives

- Mastery of the concepts and terminology of web services.

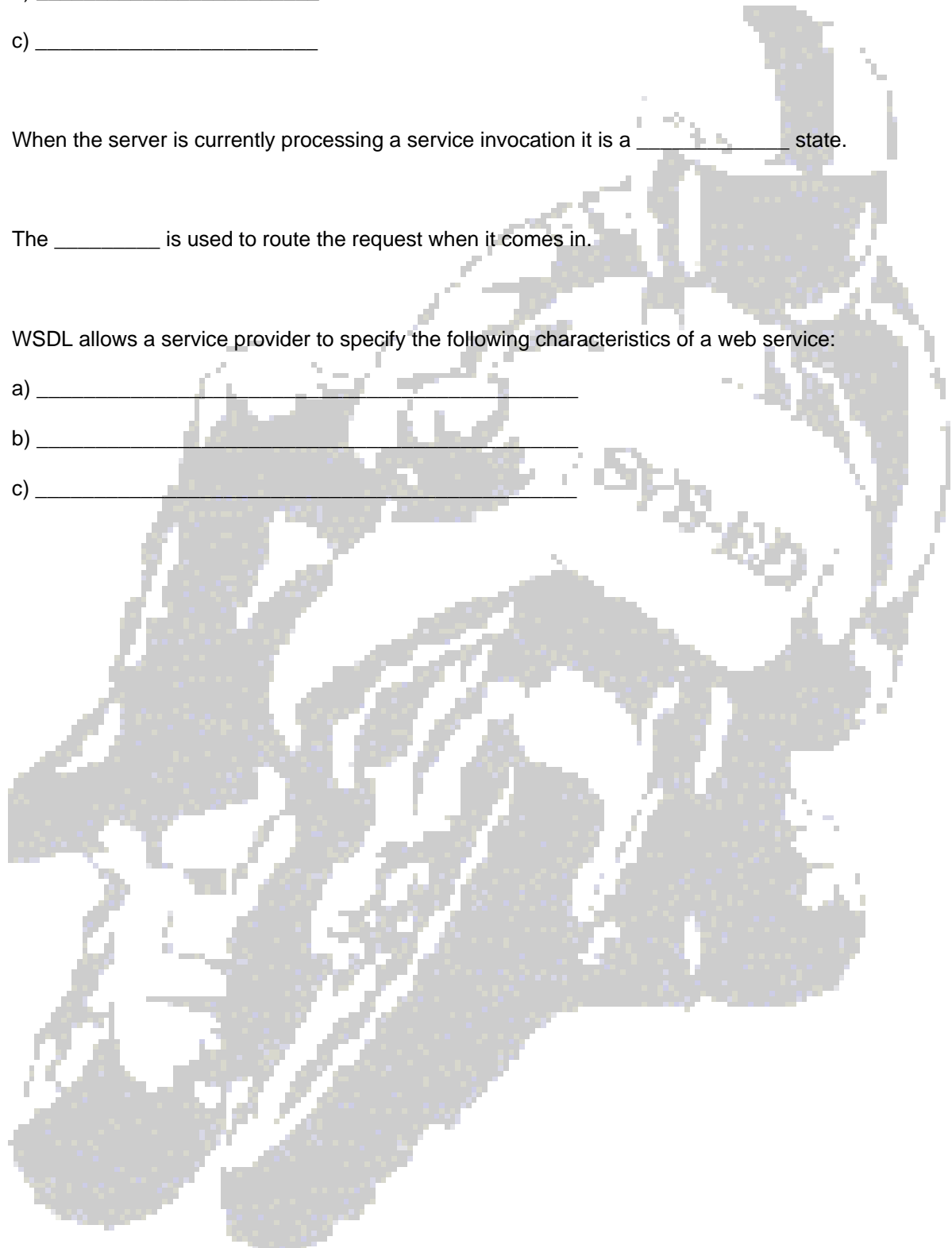
### Questions

1. The Universal data description language is \_\_\_\_\_.
2. Web services are:
  - a) \_\_\_\_\_
  - b) \_\_\_\_\_
  - c) \_\_\_\_\_
  - d) \_\_\_\_\_
3. \_\_\_\_\_ is a network, transport, and programming language neutral protocol which allows a client to call a remote service.
4. \_\_\_\_\_ is an XML based interface and implementation description language.
5. UDDI access is performed via \_\_\_\_\_ over \_\_\_\_\_.

The service provider can choose between three different development styles when defining the WSDL and the Java implementation for his/her Web service.

6. The following are the three services:
  - a) \_\_\_\_\_
  - b) \_\_\_\_\_
  - c) \_\_\_\_\_

7. There are three types of requestors including:
  - a) \_\_\_\_\_
  - b) \_\_\_\_\_
  - c) \_\_\_\_\_
  
8. When the server is currently processing a service invocation it is a \_\_\_\_\_ state.
  
9. The \_\_\_\_\_ is used to route the request when it comes in.
  
10. WSDL allows a service provider to specify the following characteristics of a web service:
  - a) \_\_\_\_\_
  - b) \_\_\_\_\_
  - c) \_\_\_\_\_



**Answers**

1. XML
2. Any of the following:
  - Self-contained
  - Self-describing
  - Modularity
  - Language independent and interoperable
  - Open and standards based
  - Dynamic
  - Composable
3. SOAP
4. WSDL
5. SOAP, HTTP
6. Top-down, Bottom-up, and Meet-in-the-middle
7. Static service, Provider-dynamic, and Type-dynamic
8. Running
9. URN
10.
  - a) Name of the web service and addressing information.
  - b) Protocol and encoding style to be used when accessing the public operations of the web service.
  - c) Type information: operations, parameters, and data types comprising the interface of the web service plus a name for this interface.

---

**9 Web Services Tools**

Create a web service by converting a EJB to a web service.

**Objectives**

- Create a web service in WSAD.

**Exercises**

1. Use the EJB bean created in a exercise and convert the bean to a web service.
2. Generate the web service web client to test the bean.

