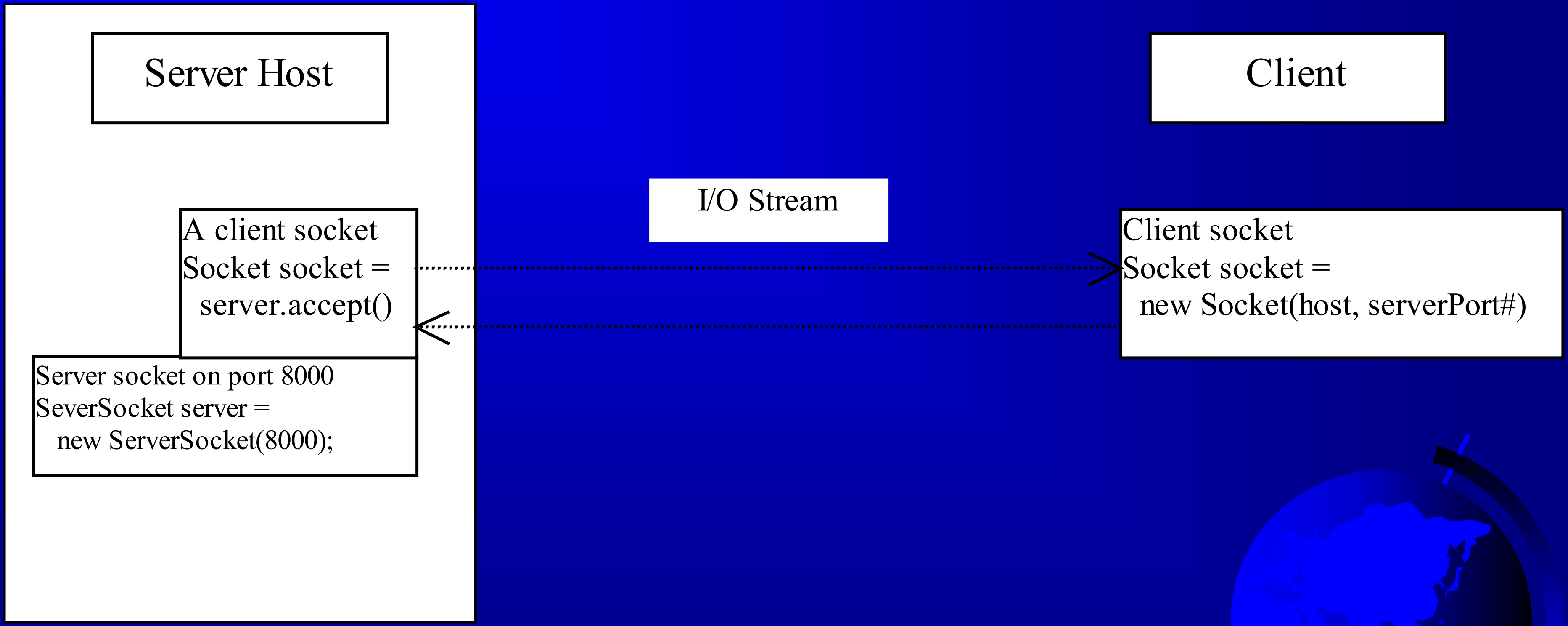
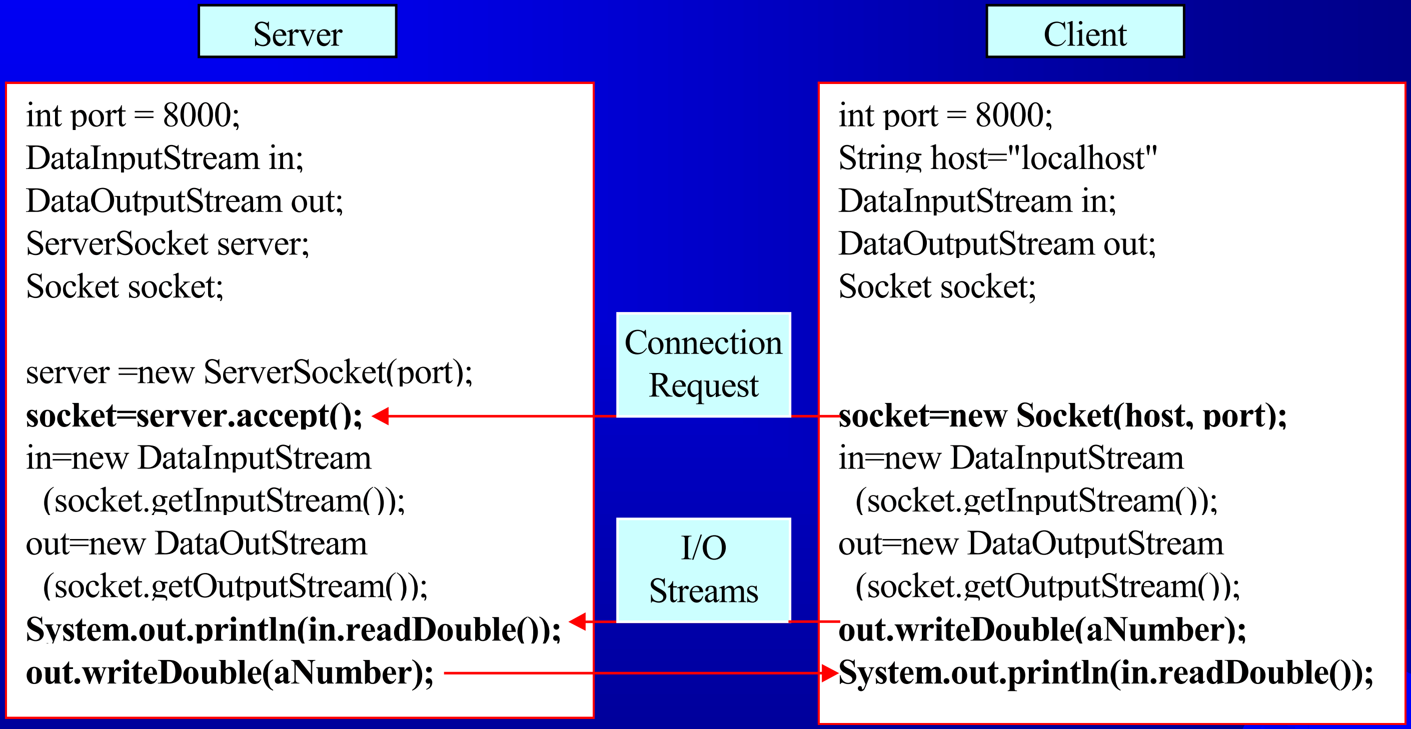
Chapter 16 Networking

- Client/Server Communications
- Simple Client/Server Applications
- Serve Multiple Clients
- Create Applet Clients
- Send and Retrieve Obects on the Network
- The `URL` Class
- Retrieve Files from the Network
- Retrieve Files from Web Servers
- View HTML Pages

Client/Server Communications



Coding Client and Server



```
int port = 8000;
DataInputStream in;
DataOutputStream out;
ServerSocket server;
Socket socket;

server =new ServerSocket(port);
socket=server.accept();
in=new DataInputStream
  (socket.getInputStream());
out=new DataOutStream
  (socket.getOutputStream());
System.out.println(in.readDouble());
out.writeDouble(aNumber);
```

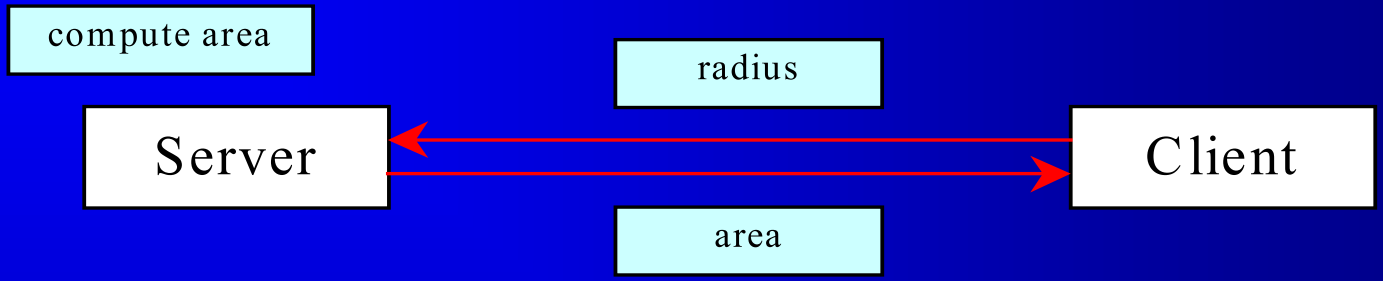
```
int port = 8000;
String host="localhost"
DataInputStream in;
DataOutputStream out;
Socket socket;


socket=new Socket(host, port);
in=new DataInputStream
  (socket.getInputStream());
out=new DataOutputStream
  (socket.getOutputStream());
out.writeDouble(aNumber);
System.out.println(in.readDouble());
```

Example 16.1 A Client/Server Example

- Objective: Write a client to send data to a server. The server receives the data, uses it to produce a result, and then sends the result back to the client. The client displays the result on the console. In this example, the data sent from the client is the radius of a circle, and the result produced by the server is the area of the circle.

Example 16.1, cont.



Server Code

Run

Client Code

Run

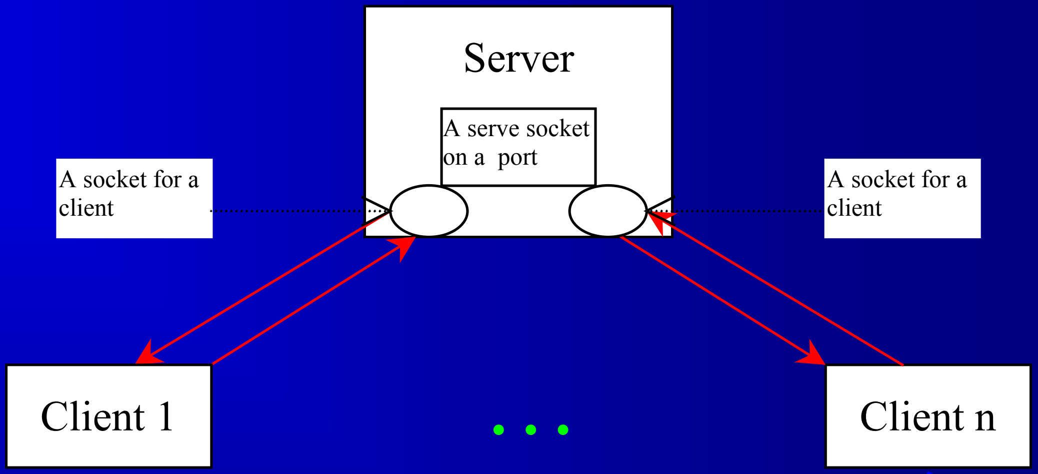
Note: Run Server first, then Client. Press Ctrl+C to close the window.

Example 16.2 Serving Multiple Clients

Server for Multiple Clients

Run Server

Run Client



Note: Run Server first, then Client. Press Ctrl+C to close the window.

Applet Clients

Due to security constraints, applets can only connect to the host from which they were loaded. Therefore, the HTML file must be located on the machine on which the server is running.

Example 16.3 Creating Applet Clients

Objective: shows how to use an applet to register students. The client collects and sends registration information to the server. The server appends the information to a data file using a random access file stream.

Server Code

Client Code

Run

Click Run. Type `java RegistrationServer` and press Enter. Alt+Tab back to this window and click Run again. Type `appletviewer registrationClient.html` in the second DOS window. Display the Applet Viewer on top of the server window, and enter student information. To end the server session, press Ctrl+C. (Note: This program cannot be run from the CD.)

Example 16.4 Passing Objects in Network Programs

Objective: This example rewrites Example 16.5, using object streams on the socket. Instead of passing name, street, state, and zips separately, this program passes the student object as a whole object.

Server Code　　Client Code　　Run

Click Run. Type `java RegistrationServer` and press Enter. Alt+Tab back to this window and click Run again. Type `appletviewer registrationClient.html` in the second DOS window. Display the Applet Viewer on top of the server window, and enter student information. To end the server session, press Ctrl+C. (Note: This program cannot be run from the CD.)

Viewing HTML Pages

- Given the URL of the page, a Web browser can view an HTML page—for example, `http://www.sun.com`.

- HTTP (Hypertext Transfer Protocol) is the common standard used for communication between a Web server and the Internet. You can open a URL and view a Web page in a Java applet.

- A URL is a description of a resource location on the Internet. Java provides a class—`java.net.URL`—to manipulate URLs.

Creating a URL Instance

The following statement creates a Java URL object:

```
try
{ URL location = new URL("http://www.sun.com");
}
catch(MalformedURLException e)
{ }
```

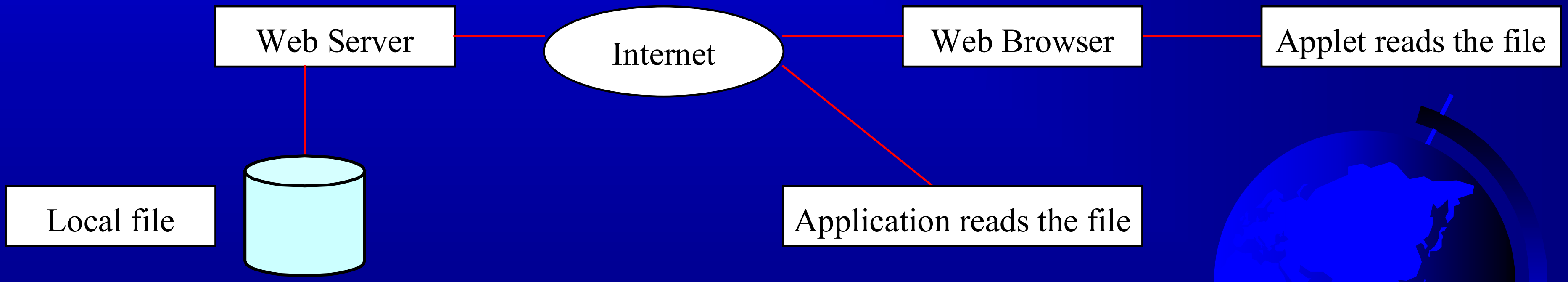
ViewingWebPages

Run

If necessary, click the Run button to access the DOS prompt. Using a JDK 1.2-enabled Web browser. This applet cannot run using the Applet Viewer utility.

Retrieving Files from Web Servers

The following figure shows the process by which an applet reads the files on the Web server:



Example 16.6 Retrieving Remote Files

- Objective: Compute and display student exam scores. The example is similar to Example 15.5. Rather than reading the file from the local system, this example reads the file from a Web server.

ViewRemoteFile

Run

If necessary, click the Run button to access the DOS prompt. Using a Web browser (such as the HotJava Browser), enter the following URL:

`http://www.cs.armstrong.edu/liang/intro3e/ViewRemoteFile.html`

The Web Server

- You need to place three files on the Web server:
  - ViewRemoteFile.class
  - ViewRemoteFile.html
  - in.dat
- For convenience, place them in one directory.

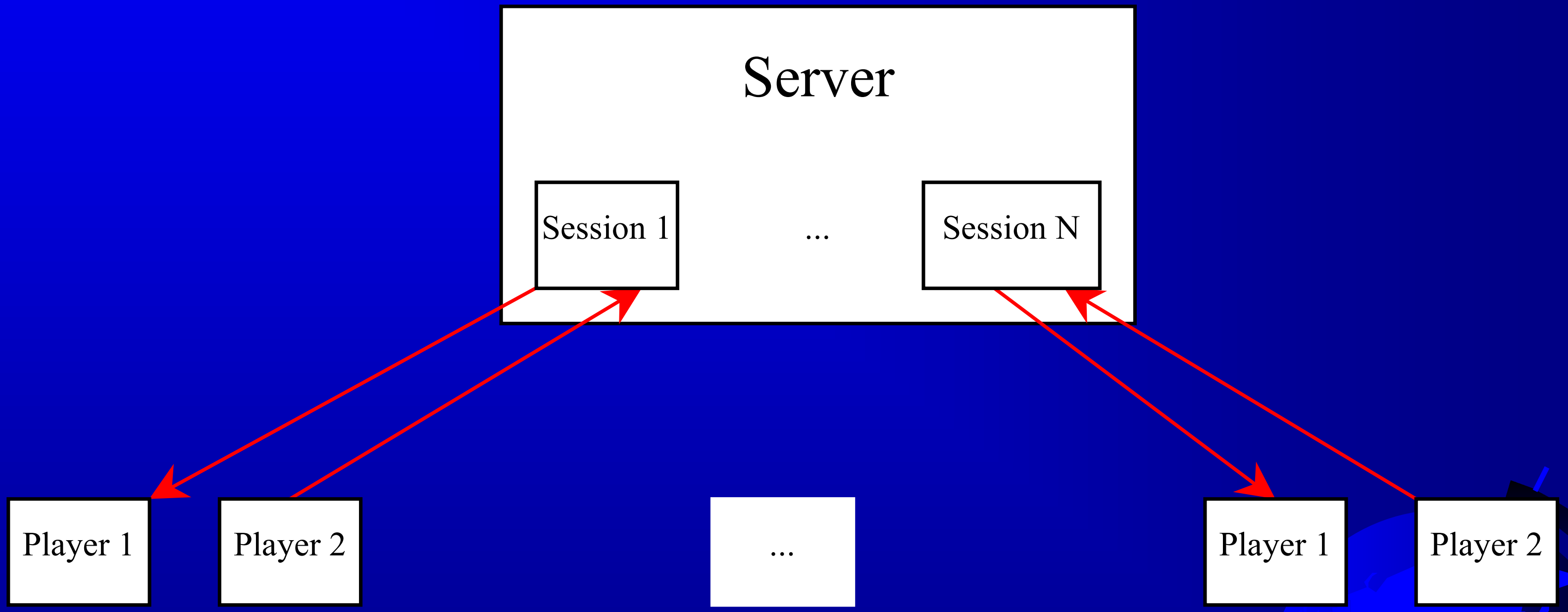
Viewing HTML Files Using the `JEditorPane`

`JEditorPane` can be used to display HTML files.

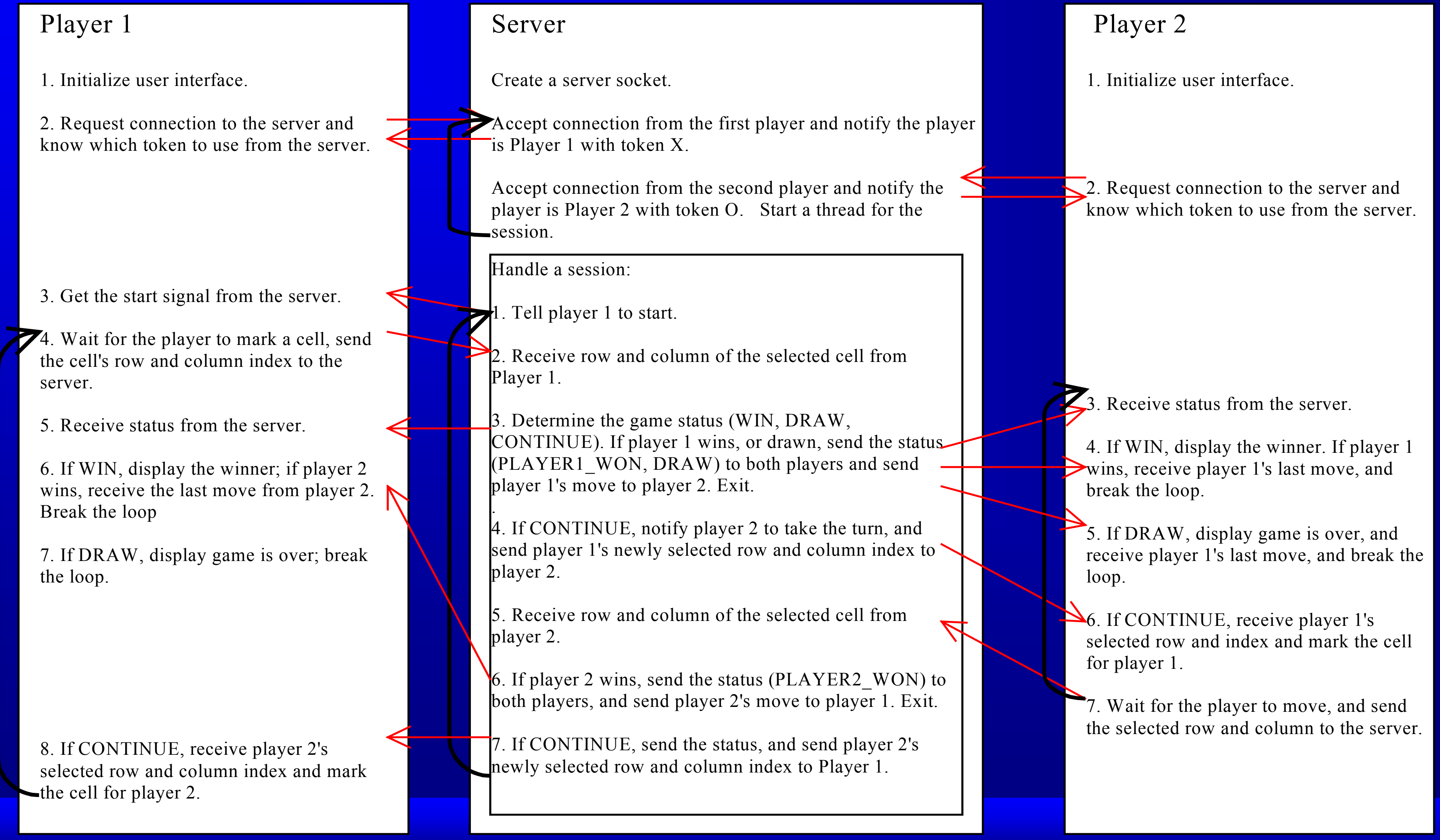
WebBrowser

Run Applet Viewer

Distributed TicTacToe Game



Distributed TicTacToe Game



Player 1

1. Initialize user interface.

2. Request connection to the server and know which token to use from the server.

3. Get the start signal from the server.

4. Wait for the player to mark a cell, send the cell's row and column index to the server.

5. Receive status from the server.

6. If WIN, display the winner; if player 2 wins, receive the last move from player 2. Break the loop

7. If DRAW, display game is over; break the loop.

8. If CONTINUE, receive player 2's selected row and column index and mark the cell for player 2.

Server

Create a server socket.

Accept connection from the first player and notify the player is Player 1 with token X.

Accept connection from the second player and notify the player is Player 2 with token O. Start a thread for the session.

Handle a session:

1. Tell player 1 to start.

2. Receive row and column of the selected cell from Player 1.

3. Determine the game status (WIN, DRAW, CONTINUE). If player 1 wins, or drawn, send the status (PLAYER1_WON, DRAW) to both players and send player 1's move to player 2. Exit.

.

4. If CONTINUE, notify player 2 to take the turn, and send player 1's newly selected row and column index to player 2.

5. Receive row and column of the selected cell from player 2.

6. If player 2 wins, send the status (PLAYER2_WON) to both players, and send player 2's move to player 1. Exit.

7. If CONTINUE, send the status, and send player 2's newly selected row and column index to Player 1.

Player 2

1. Initialize user interface.

2. Request connection to the server and know which token to use from the server.

3. Receive status from the server.

4. If WIN, display the winner. If player 1 wins, receive player 1's last move, and break the loop.

5. If DRAW, display game is over, and receive player 1's last move, and break the loop.

6. If CONTINUE, receive player 1's selected row and index and mark the cell for player 1.

7. Wait for the player to move, and send the selected row and column to the server.