core WEB programming

Using Applets as Front Ends to Server-Side Programs

Agenda

- Sending GET data and having the browser display the results
- Sending GET data and processing the results within the applet (HTTP tunneling)
- Using object serialization to exchange high-level data structures between applets and servlets
- Sending POST data and processing the results within the applet (HTTP tunneling)
- Bypassing the HTTP server altogether

Sending GET Request and Displaying Resultant Page

- **Applet requests that browser display page – showDocument**

```
try {
  URL programURL =
       new URL(baseURL + "?" + someData);
  getAppletContext().showDocument(programURL);
} catch(MalformedURLException mue) { ... };
```

- **URL-encode the form data**

```
String someData =
  name1 + "=" + URLEncoder.encode(val1) + "&" +
  name2 + "=" + URLEncoder.encode(val2) + "&" +
  ...
  nameN + "=" + URLEncoder.encode(valN);
```

GET Request Example: Applet

```
public class SearchApplet extends Applet
                          implements ActionListener {
 ...
 public void actionPerformed(ActionEvent event) {
   String query =
     URLEncoder.encode(queryField.getText());
   SearchSpec[] commonSpecs =
     SearchSpec.getCommonSpecs();
   for(int i=0; i<commonSpecs.length-1; i++) {
     try {
       SearchSpec spec = commonSpecs[i];
       URL searchURL =
         new URL(spec.makeURL(query, "10"));
       String frameName = "results" + i;
       getAppletContext().showDocument(searchURL,
                                       frameName);
     } catch(MalformedURLException mue) {}
   }
 }
}
```

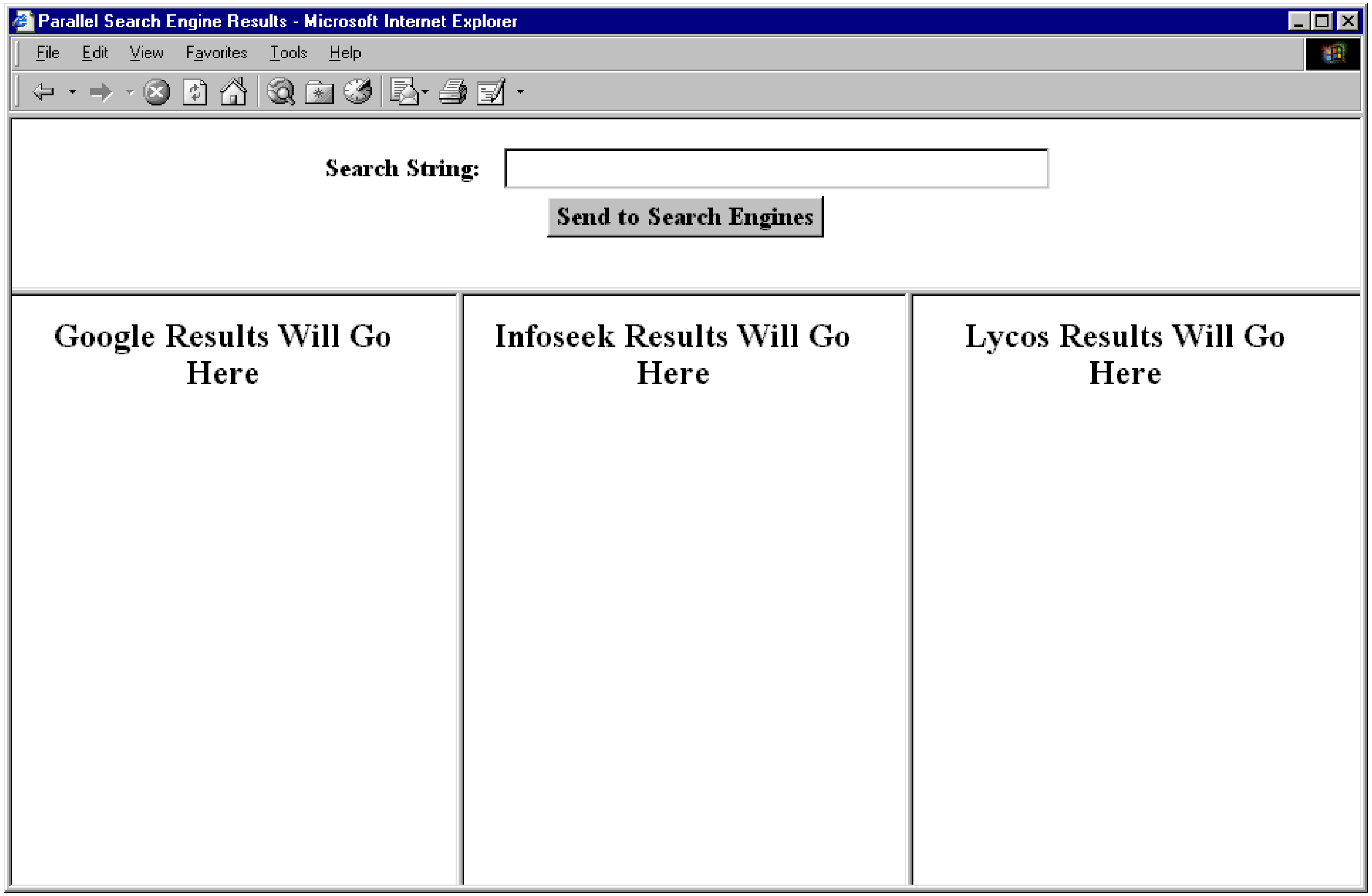
GET Request Example: Utility Class

```
public class SearchSpec {
  private String name, baseURL, numResultsSuffix;

  private static SearchSpec[] commonSpecs =
    { new SearchSpec("google",
                     "http://www.google.com/search?q=",
                     "&num="),
      ... };

  public String makeURL(String searchString,
                        String numResults) {
    return(baseURL + searchString +
           numResultsSuffix + numResults);
  }
  ...
}
```

Get Request Example: HTML File

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Frameset//EN">
<HTML>
<HEAD>
  <TITLE>Parallel Search Engine Results</TITLE>
</HEAD>

<FRAMESET ROWS="120,*">
  <FRAME SRC="SearchAppletFrame.html" SCROLLING="NO">
  <FRAMESET COLS="*,*,*">
    <FRAME SRC="GoogleResultsFrame.html" NAME="results0">
    <FRAME SRC="InfoseekResultsFrame.html" NAME="results1">
    <FRAME SRC="LycosResultsFrame.html" NAME="results2">
  </FRAMESET>
</FRAMESET>
```

Get Request: Initial Result

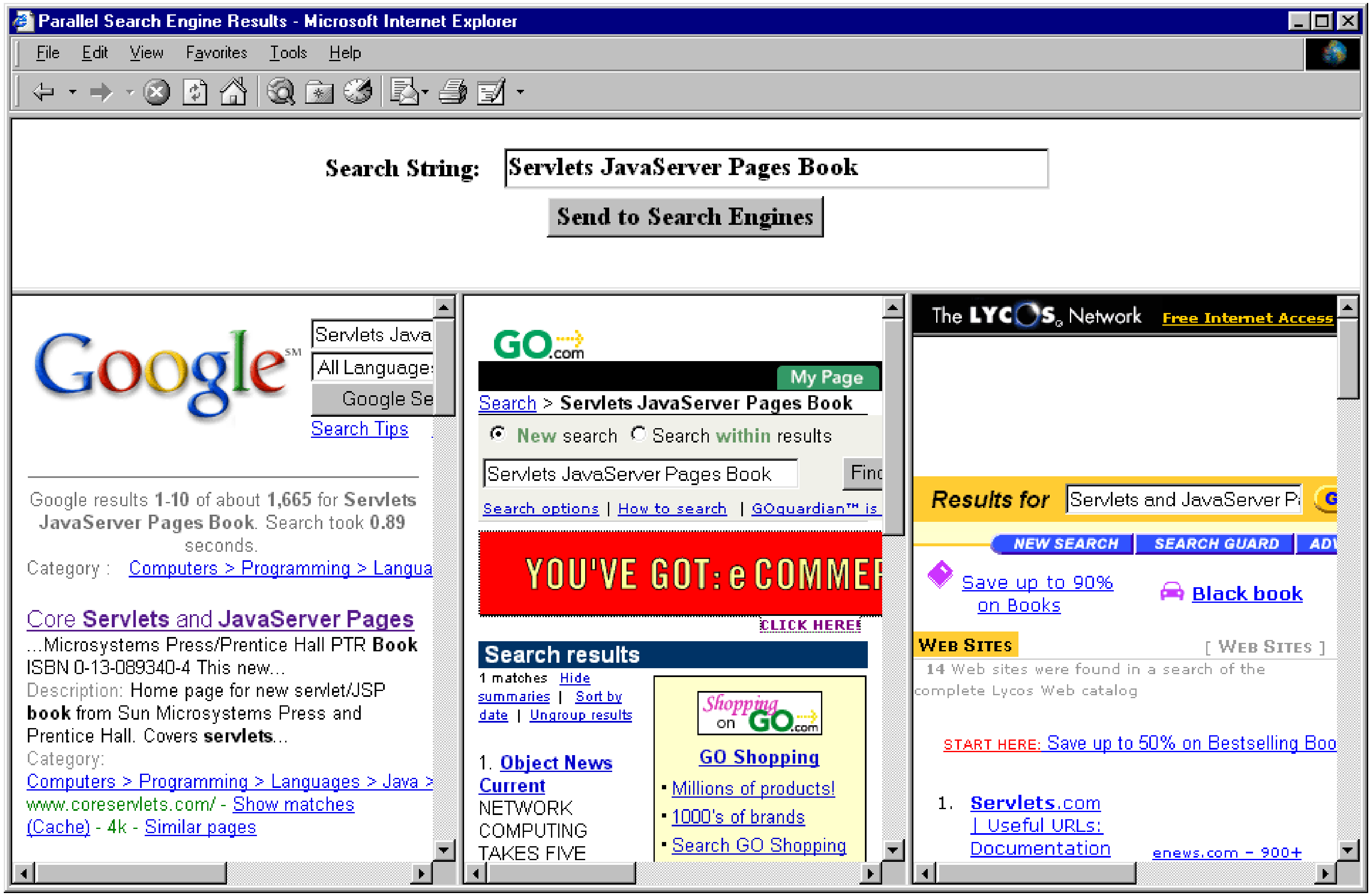

Parallel Search Engine Results - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Search String:
Send to Search Engines

Google Results Will Go Here

Infoseek Results Will Go Here

Lycos Results Will Go Here

GET Request: Submission Result


Parallel Search Engine Results - Microsoft Internet Explorer

File Edit View Favorites Tools Help

Search String: Servlets JavaServer Pages Book

Send to Search Engines

Google results 1-10 of about 1,665 for Servlets JavaServer Pages Book. Search took 0.89 seconds.
Category : Computers > Programming > Langua

Core Servlets and JavaServer Pages
...Microsystems Press/Prentice Hall PTR Book ISBN 0-13-089340-4 This new...
Description: Home page for new servlet/JSP book from Sun Microsystems Press and Prentice Hall. Covers servlets...
Category: Computers > Programming > Languages > Java >
www.coreservlets.com/ - Show matches (Cache) - 4k - Similar pages

Search > Servlets JavaServer Pages Book
New search Search within results
Servlets JavaServer Pages Book Find
Search options | How to search | GOquardian™ is

YOU'VE GOT: e COMMER
CLICK HERE!

Search results
1 matches Hide summaries | Sort by date | Ungroup results

1. Object News Current
NETWORK COMPUTING TAKES FIVE

GO Shopping
- Millions of products!
- 1000's of brands
- Search GO Shopping

Results for Servlets and JavaServer P
NEW SEARCH SEARCH GUARD ADV
Save up to 90% on Books
Black book

WEB SITES [WEB SITES]
14 Web sites were found in a search of the complete Lycos Web catalog

START HERE: Save up to 50% on Bestselling Boo

1. Servlets.com | Useful URLs: Documentation
enews.com - 900+

HTTP Tunneling

- **Idea**
  - Open a socket connection to port 80 on the server and communicate through HTTP
- **Advantages**
  - Communicate through firewalls
  - Server-side programs only needs to return the data, not a complete HTML document
- **Disadvantages**
  - Can only tunnel to server from which the applet was loaded
  - *Applet*, not *browser*, receives the response
    - Cannot easily display HTML

HTTP Tunneling and GET Requests

- **Create URL object referring to applet's host**
  ```
  URL dataURL = new URL(…);
  ```
- **Create a URLConnection object**
  ```
  URLConnection connection = dataURL.openConnection();
  ```
- **Instruct browser not to cache URL data**
  ```
  connection.setUseCaches(false);
  ```
- **Set any desired HTTP headers**
- **Create an input stream**
  - Call `connection.getInputStream`; wrap in higher-level stream
- **Read data sent from server**
  - E.g., call `readLine` on `BufferedReader`
- **Close the input stream**

HTTP Tunneling Template: Client Side

```
URL currentPage = getCodeBase();
String protocol = currentPage.getProtocol();
String host = currentPage.getHost();
int port = currentPage.getPort();
String urlSuffix = "/servlet/SomeServlet";
URL dataURL = new URL(protocol, host, port, urlSuffix);
URLConnection connection = dataURL.getConnection();
connection.setUseCaches(false);
connection.setRequestProperty("header", "value");

BufferedReader in = new BufferedReader(
  new InputStreamReader(connection.getInputStream()));
String line;
while ((line = in.readLine()) != null) {
  doSomethingWith(line);
}
in.close();
```

Using Object Serialization with HTTP Tunneling

- **Idea**
  - Server-side program (servlet) sends complete Java object
  - Client-side program (applet) reads it
- **Client-side program (applet) template:**

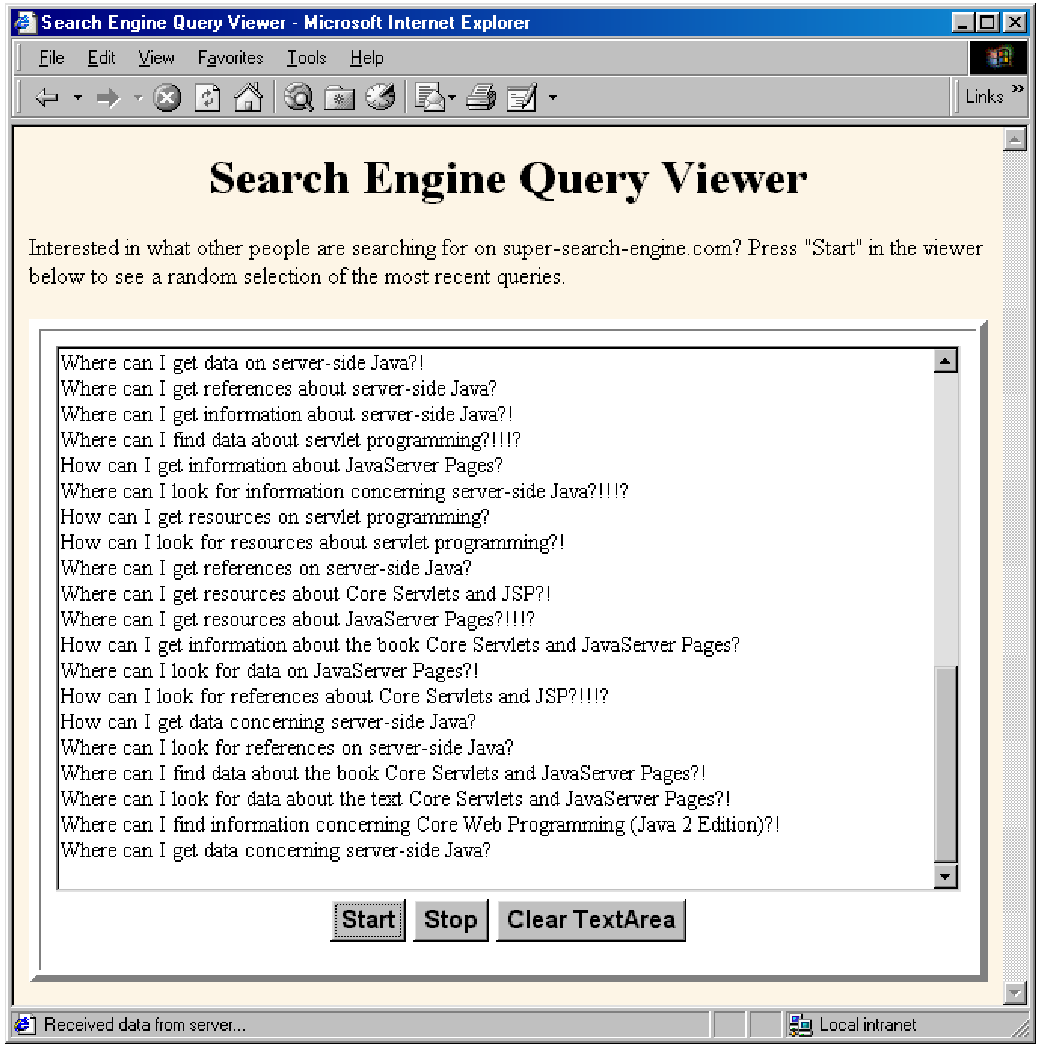
```
ObjectInputStream in =
  new ObjectInputStream(
   connection.getInputStream());

SomeClass object = (SomeClass)in.readObject();
doSomethingWith(object);
```

Using Object Serialization with HTTP Tunneling (Continued)

- **Server-side program (servlet) template:**

```
String contentType =
  "application/x-java-serialized-object";
response.setContentType(contentType);

ObjectOutputStream out =
  new ObjectOutputStream(
    response.getOutputStream());

SomeClass object = new SomeClass(...);
out.writeObject(value);
out.flush();
```

Example: Live Scrolling Data



Search Engine Query Viewer - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Links

Search Engine Query Viewer

Interested in what other people are searching for on super-search-engine.com? Press "Start" in the viewer below to see a random selection of the most recent queries.

```
Where can I get data on server-side Java?!
Where can I get references about server-side Java?
Where can I get information about server-side Java?!
Where can I find data about servlet programming?!!!?
How can I get information about JavaServer Pages?
Where can I look for information concerning server-side Java?!!!?
How can I get resources on servlet programming?
How can I look for resources about servlet programming?!
Where can I get references on server-side Java?
Where can I get resources about Core Servlets and JSP?!
Where can I get resources about JavaServer Pages?!!!?
How can I get information about the book Core Servlets and JavaServer Pages?
Where can I look for data on JavaServer Pages?!
How can I look for references about Core Servlets and JSP?!!!?
How can I get data concerning server-side Java?
Where can I look for references on server-side Java?
Where can I find data about the book Core Servlets and JavaServer Pages?!
Where can I look for data about the text Core Servlets and JavaServer Pages?!
Where can I find information concerning Core Web Programming (Java 2 Edition)?!
Where can I get data concerning server-side Java?
```

Sending POST Data to Server

- Applet sends POST request to server
- Processes the response directly

```
Url currentPage = getCodeBase();
String protocol = currentPage.getProtocol();
String host = currentPage.getHost();
int port = currentPage.getPort();
String urlSuffix = "/servlet/SomeServlet";
URL dataURL = new URL(protocol, host, port,
                      urlSuffix);

URLConnection connection =
  dataURL.openConnection();
connection.setUseCaches(false);
connection.setDoOutput(true);
```

Sending POST Data to Server (Continued)

- **Character or Binary Data**

```
ByteArrayOutputStream byteStream =
  new ByteArrayOutputStream(512);
PrintWriter out = new PrintWriter(byteStream, true);
out.print(data);
out.flush();

connection.setRequestProperty(
              "Content-Length",
              String.valueOf(byteStream.size()));
connection.setRequestProperty(
            "Content-Type",
            "application/x-www-form-urlencoded");
byteStream.writeTo(connection.getOutputStream());
```

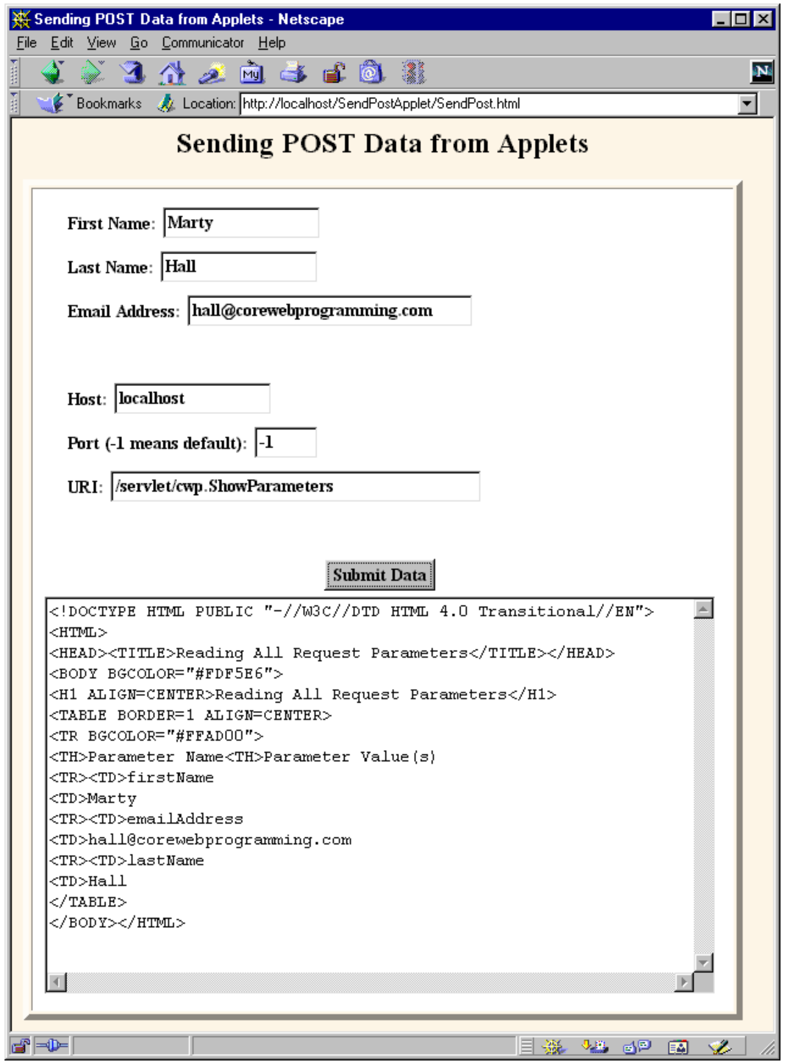
Sending POST Data to Server

- **Serialized Data**

```
ByteArrayOutputStream byteStream =
  new ByteArrayOutputStream(512);
ObjectOutputStream out =
  new ObjectOutputStream(byteStream);
out.writeObject(data);
out.flush();

connection.setRequestProperty(
              "Content-Length",
              String.valueOf(byteStream.size()));
connection.setRequestProperty(
            "Content-Type",
            "application/x-java-serialized-object");
byteStream.writeTo(connection.getOutputStream());
```

Sending POST Data: Example

- **Sends data to a servlet that returns an HTML page showing form data it receives**
  - Displays result in an AWT TextArea


Sending POST Data from Applets - Netscape
File Edit View Go Communicator Help
Bookmarks
Location: http://localhost/SendPostApplet/SendPost.html

Sending POST Data from Applets

First Name: Marty

Last Name: Hall

Email Address: hall@corewebprogramming.com

Host: localhost

Port (-1 means default): -1

URI: /servlet/cwp.ShowParameters

Submit Data

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">
<HTML>
<HEAD><TITLE>Reading All Request Parameters</TITLE></HEAD>
<BODY BGCOLOR="#FDF5E6">
<H1 ALIGN=CENTER>Reading All Request Parameters</H1>
<TABLE BORDER=1 ALIGN=CENTER>
<TR BGCOLOR="#FFAD00">
<TH>Parameter Name<TH>Parameter Value(s)
<TR><TD>firstName
<TD>Marty
<TR><TD>emailAddress
<TD>hall@corewebprogramming.com
<TR><TD>lastName
<TD>Hall
</TABLE>
</BODY></HTML>
```

Bypassing the HTTP Server

- **If you are using applets, you don't have to communicate via HTTP**
  - JDBC
  - RMI
  - SOAP (perhaps via JAX-RPC)
  - Raw sockets
- **Advantages**
  - Simpler
  - More efficient
- **Disadvantages**
  - Can only talk to server from which applet was loaded
  - Subject to firewall restrictions
  - You have to have a second server running

Summary

- **Send data via GET and showDocument**
  - Can access any URL
  - Only browser sees result
- **Send data via GET and URLConnection**
  - Can only access URLs on applet's home host
  - Applet sees results
  - Applet can send simple data
  - Server can send complex data (including Java objects)
- **Send data via POST and URLConnection**
  - Can only access URLs on applet's home host
  - Applet sees results
  - Applet can send complex data (including Java objects)
  - Server can send complex data (including Java objects)
- **Bypass Web Server**

core WEB programming

Questions?