
Web Search

Interfaces

1

Web Search Interface

- Web search engines of course need a web-based interface.
- Search page must accept a query string and submit it within an HTML `<form>`.
- Program on the server must process requests and generate HTML text for the top ranked documents with pointers to the original and/or cached web pages.
- Server program must also allow for requests for more relevant documents for a previous query.

2

Submit Forms

- HTML supports various types of program input in forms, including:
 - Text boxes
 - Menus
 - Check boxes
 - Radio buttons
- When user submits a form, string values for various *parameters* are sent to the server program for processing.
- Server program uses these values to compute an appropriate HTML response page.

3

Simple Search Submit Form

```
<form action="http://prospero.cs.utexas.edu:8082/servlet/irs.Search" method="POST">
<p> <b> Enter your query: </b>
  <input type="text" name="query" size=40>
<p> <b> Search Database: </b>
  <select name="directory">
    <option selected value="/u/mooney/ir-code/corpora/cs-faculty/"> UT CS Faculty
    <option value="/u/mooney/ir-code/corpora/yahoo-science/"> Yahoo Science
  </select>
<p> <b> Use Relevance Feedback: </b>
  <input type="checkbox" name="feedback" value="1">
<br> <br>
  <input type="submit" value="Submit Query">
  <input type="reset" value="Reset Form">
</form>
```

4

What's a Servlet?

- Java's answer to CGI programming for processing web form requests.
- Program runs on Web server and builds pages on the fly.
- When would you use servlets?
 - Page is based on user-submitted data e.g search engines.
 - Data changes frequently e.g. weather-reports.
 - Page uses information from a databases e.g. on-line stores.
- Requires running a web server that supports servlets.

5

Basic Servlet Structure

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

public class SomeServlet extends HttpServlet {
  // Handle get request
  public void doGet(HttpServletRequest request, HttpServletResponse
  response) throws ServletException, IOException {
    // request - access incoming HTTP headers and HTML form data
    // response - specify the HTTP response line and headers
    // (e.g. specifying the content type, setting cookies).
    PrintWriter out = response.getWriter(); //out - send content to
    browser
  }
}
```

6

A Simple Servlet

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

public class HelloWorld extends HttpServlet {
    public void doGet(HttpServletRequest request,
        HttpServletResponse response) throws ServletException,
        IOException {

        PrintWriter out = response.getWriter();
        out.println("Hello World");
    }
}
```

7

Generating HTML

```
public class HelloWWW extends HttpServlet {
    public void doGet(HttpServletRequest request, HttpServletResponse
        response) throws ServletException, IOException {

        response.setContentType("text/html");
        PrintWriter out = response.getWriter();
        out.println("<HTML>\n"+
            "<HEAD><TITLE>HelloWWW</TITLE></HEAD>\n"+
            "<BODY>\n" + "<H1>Hello WWW</H1>\n" +
            "</BODY></HTML>");
    }
}
```

8

HTML Post Form

```
<FORM ACTION="/servlet/hall.ThreeParams"
    METHOD="POST">
    First Parameter: <INPUT TYPE="TEXT"
    NAME="param1"><BR>
    Second Parameter: <INPUT TYPE="TEXT"
    NAME="param2"><BR>
    Third Parameter: <INPUT TYPE="TEXT"
    NAME="param3"><BR>
    <CENTER>
        <INPUT TYPE="SUBMIT">
    </CENTER>
</FORM>
```

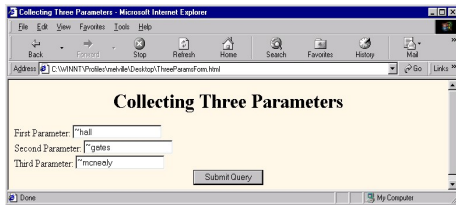
9

Reading Parameters

```
public class ThreeParams extends HttpServlet {
    public void doGet(HttpServletRequest request,
        HttpServletResponse response) throws ServletException,
        IOException {
        response.setContentType("text/html");
        PrintWriter out = response.getWriter();
        out.println("<UL>\n" +
            "<LI>param1: " + request.getParameter("param1") + "\n" +
            "<LI>param2: " + request.getParameter("param2") + "\n" +
            "<LI>param3: " + request.getParameter("param3") + "\n" +
            "</UL>\n" + ...);
    }
    public void doPost(HttpServletRequest request,
        HttpServletResponse response) throws ServletException,
        IOException {
        doGet(request, response);
    }
}
```

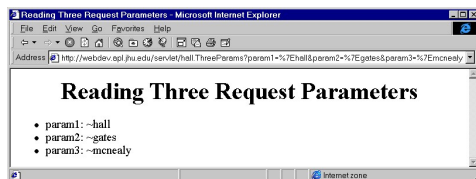
10

Form Example



11

Servlet Output



12

Session Tracking

- Typical scenario – shopping cart in online store.
- Necessary because HTTP is a "stateless" protocol.
- Common solutions: Cookies and URL-rewriting.
- Session Tracking API allows you to:
 - Look up session object associated with current request.
 - Create a new session object when necessary.
 - Look up information associated with a session.
 - Store information in a session.
 - Discard completed or abandoned sessions.

13

Session Tracking API - I

- Looking up a session object:
 - `HttpSession session = request.getSession(true);`
 - Pass `true` to create a new session if one does not exist.
- Associating information with session:
 - `session.setAttribute("user", request.getParameter("name"))`
 - Session attributes can be of any type.
- Looking up session information:
 - `String name = (String) session.getAttribute("user")`

14

Session Tracking API - II

- **getId**
 - The unique identifier generated for the session.
- **isNew**
 - `true` if the client (browser) has never seen the session.
- **getCreationTime**
 - Time in milliseconds since session was made.
- **getLastAccessedTime**
 - Time in milliseconds since the session was last sent from client.
- **getMaxInactiveInterval**
 - # of seconds session should go without access before being invalidated.
 - Negative value indicates that session should never timeout.

15

Simple Search Servlet

- Based on `directory` parameter, creates or selects existing `InvertedIndex` for the appropriate corpus.
- Processes the query with VSR to get ranked results.
- Writes out HTML ordered list of 10 results starting at the rank of the `start` parameter.
- Each item includes:
 - Link to the original URL saved by the spider in the top of the document in `BASE` tag.
 - Name link with page `<TITLE>` extracted from file.
 - Additional link to local cached file.
- If all retrievals not already shown, creates a submit form for “**More Results**” starting from the next ranked item.

16

Simple Search Interface Refinements

- For “**More results**” requests, stores current ranked list with the user session and displays next set in the list.
- Integrates relevance feedback interaction with “radio buttons” for “NEUTRAL,” “GOOD,” and “BAD” in HTML form.

17

Other Search Interface Refinements

- Highlight search terms in the displayed document.
 - Provided in cached file on [Google](#).
- Allow for “advanced” search:
 - Phrasal search (“.”)
 - Mandatory terms (+)
 - Negated term (-)
 - Language preference
 - Reverse link
 - Date preference
- Machine translation of pages.

18

Clustering Results

- Group search results into coherent “clusters”:
 - “microwave dish”
 - One group of on food recipes or cookware.
 - Another group on satellite TV reception.
 - “Austin bats”
 - One group on the local flying mammals.
 - One group on the local hockey team.
- Northern Light used to group results into “folders” based on a pre-established categorization of pages (like DMOZ categories).
- Alternative is to dynamically cluster search results into groups of similar documents.

19

User Query Length

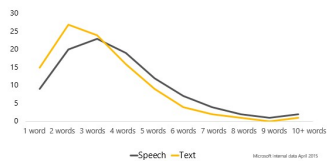
- Users tend to enter short queries.
 - Study in 1998 gave average length of 2.35 words.
- Evidence that queries are getting longer.

Subject	Jan-08	Dec-08	Jan-09	Year-over-year percent change
1 word	20.96%	20.70%	20.29%	-3%
2 words	24.91%	24.13%	23.65%	-6%
3 words	22.03%	21.94%	21.92%	0%
4 words	14.54%	14.67%	14.89%	2%
5 words	8.20%	8.37%	8.68%	6%
6 words	4.32%	4.47%	4.65%	8%
7 words	2.23%	2.40%	2.49%	12%
8+ words	2.81%	3.31%	3.43%	22%

Note: Data is based on four-week rolling periods (ending Jan. 31, 2009; Dec. 27, 2008; and Jan. 26, 2008) from the Hitwise sample of 10 million U.S. Internet users.
Source: Hitwise, an Experian company.

20

Speech Queries are Longer



21
