



Generating the HTTP Response

Core Servlets & JSP book: www.coreservlets.com
More Servlets & JSP book: www.moreservlets.com
Servlet and JSP Training Courses: courses.coreservlets.com

Slides © Marty Hall, <http://www.coreservlets.com>, book © Sun Microsystems Press

Agenda

- **Idea of HTTP status codes**
- **Setting status codes from servlets**
- **Common HTTP 1.1 status codes**
- **A common front end to various Web search engines**
- **Idea of HTTP response headers**
- **Setting response headers from servlets**
- **Common HTTP 1.1 response headers**
- **Persistent servlet state and auto-reloading pages**

Generating the Server Response: HTTP Status Codes

- **Example HTTP 1.1 Response**

```
HTTP/1.1 200 OK
Content-Type: text/html
```

```
<!DOCTYPE ...>
<HTML>
...
</HTML>
```

- **Changing the status code lets you perform a number of tasks not otherwise possible**
 - Forward client to another page
 - Indicate a missing resource
 - Instruct browser to use cached copy
- **Set status *before* sending document**

Setting Status Codes

- **response.setStatus(int statusCode)**
 - Use a constant for the code, not an explicit int. Constants are in HttpServletResponse
 - Names derived from standard message. E.g., SC_OK, SC_NOT_FOUND, etc.
- **response.sendError(int code, String message)**
 - Wraps message inside small HTML document
- **response.sendRedirect(String url)**
 - Relative URLs permitted in 2.2 and later
 - Sets Location header also

Common HTTP 1.1 Status Codes

- **200 (OK)**
 - Everything is fine; document follows.
 - Default for servlets.
- **204 (No Content)**
 - Browser should keep displaying previous document.
- **301 (Moved Permanently)**
 - Requested document permanently moved elsewhere (indicated in Location header).
 - Browsers go to new location automatically.

Common HTTP 1.1 Status Codes (Continued)

- **302 (Found)**
 - Requested document temporarily moved elsewhere (indicated in Location header).
 - Browsers go to new location automatically.
 - Servlets should use sendRedirect, not setStatus, when setting this header. See example.
- **401 (Unauthorized)**
 - Browser tried to access password-protected page without proper Authorization header. See example in book.
- **404 (Not Found)**
 - No such page. Servlets should use sendError to set this.
 - Problem: Internet Explorer 5.0.
 - Fun and games: <http://www.plinko.net/404/>

A Front End to Various Search Engines: Code

```
public void doGet(HttpServletRequest request,
                  HttpServletResponse response)
    throws ServletException, IOException {
    String searchString =
        request.getParameter("searchString");
    if ((searchString == null) ||
        (searchString.length() == 0)) {
        reportProblem(response, "Missing search string.");
        return;
    }
    searchString = URLEncoder.encode(searchString);
    String numResults =
        request.getParameter("numResults");
    ...
    String searchEngine =
        request.getParameter("searchEngine");
```

7

HTTP Response

www.coreservlets.com

A Front End to Various Search Engines: Code (Continued)

```
SearchSpec[] commonSpecs =
    SearchSpec.getCommonSpecs();
for(int i=0; i<commonSpecs.length; i++) {
    SearchSpec searchSpec = commonSpecs[i];
    if (searchSpec.getName().equals(searchEngine)) {
        String url =
            searchSpec.makeURL(searchString, numResults);
        response.sendRedirect(url);
        return;
    }
}
reportProblem(response,
               "Unrecognized search engine.");
```

8

HTTP Response

www.coreservlets.com

A Front End to Various Search Engines: Code (Continued)

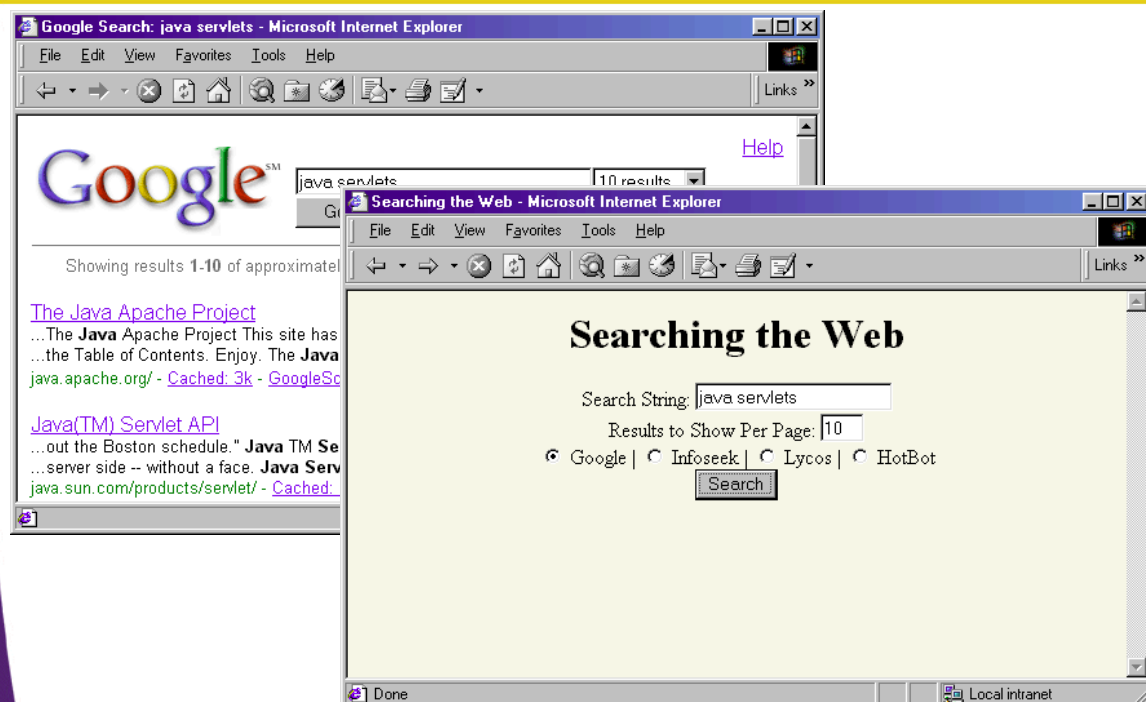
```
private void reportProblem(HttpServletRequest response,
                          String message)
    throws IOException {
    response.sendError(response.SC_NOT_FOUND,
                      message);
}
```

9

HTTP Response

www.coreservlets.com

Front End to Search Engines: Result of Legal Request

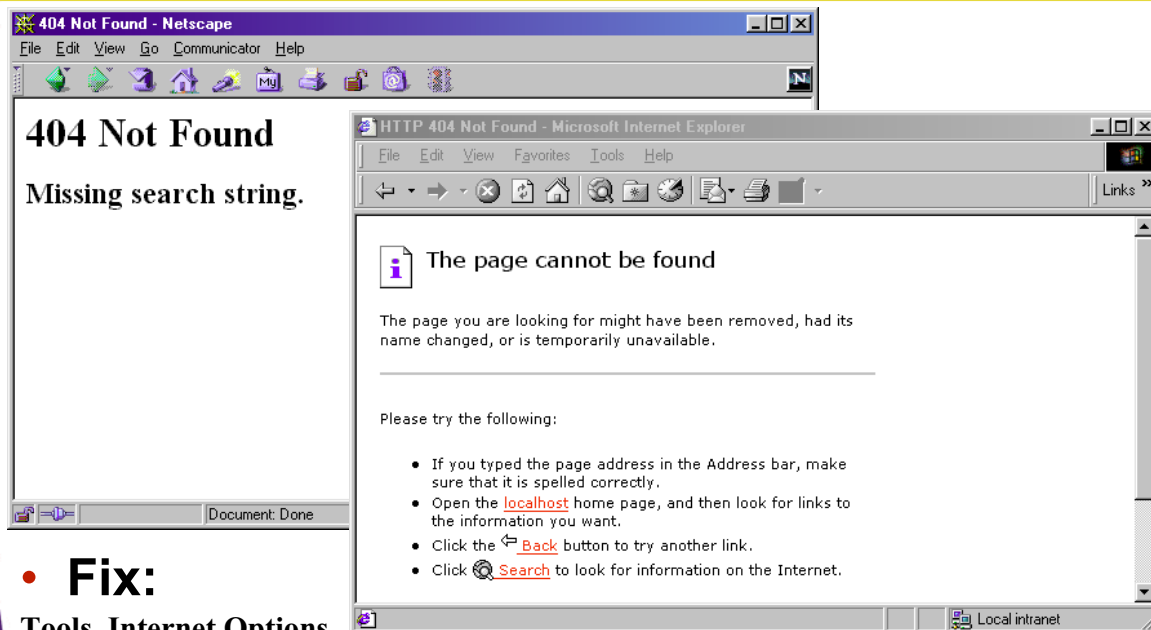


10

HTTP Response

www.coreservlets.com

Front End to Search Engines: Result of Illegal Request



- **Fix:**

Tools, Internet Options,
deselect "Show 'friendly' HTTP error messages"

Not a real fix -- doesn't help unsuspecting *users* of your pages

www.coreservlets.com

11

HTTP Response

Generating the Server Response: HTTP Response Headers

- **Purposes**

- Give forwarding location
- Specify cookies
- Supply the page modification date
- Instruct the browser to reload the page after a designated interval
- Give the document size so that persistent HTTP connections can be used
- Designate the type of document being generated
- Etc.

12

HTTP Response

www.coreservlets.com

Setting Arbitrary Response Headers

- **public void setHeader(String headerName, String headerValue)**
 - Sets an arbitrary header.
- **public void setDateHeader(String name, long millisecs)**
 - Converts milliseconds since 1970 to a date string in GMT format.
- **public void setIntHeader(String name, int headerValue)**
 - Prevents need to convert int to String before calling setHeader.
- **addHeader, addDateHeader, addIntHeader**
 - Adds new occurrence of header instead of replacing. Servlets 2.2/2.3 only.

13

HTTP Response

www.coreservlets.com

Setting Common Response Headers

- **setContentLength**
 - Sets the Content-Type header. Servlets almost always use this. See table of common MIME types.
- **setContentLength**
 - Sets the Content-Length header. Used for persistent HTTP connections. See Connection request header.
- **addCookie**
 - Adds a value to the Set-Cookie header. See separate section on cookies.
- **sendRedirect**
 - Sets the Location header (plus changes status code).

14

HTTP Response

www.coreservlets.com

Common MIME Types

Type	Meaning
application/msword	Microsoft Word document
application/octet-stream	Unrecognized or binary data
application/pdf	Acrobat (.pdf) file
application/postscript	PostScript file
application/vnd.ms-excel	Excel spreadsheet
application/vnd.ms-powerpoint	Powerpoint presentation
application/x-gzip	Gzip archive
application/x-java-archive	JAR file
application/x-java-vm	Java bytecode (.class) file
application/zip	Zip archive
audio/basic	Sound file in .au or .snd format
audio/x-aiff	AIFF sound file
audio/x-wav	Microsoft Windows sound file
audio/midi	MIDI sound file
text/css	HTML cascading style sheet
text/html	HTML document
text/plain	Plain text
text/xml	XML document
image/gif	GIF image
image/jpeg	JPEG image
image/png	PNG image
image/tiff	TIFF image
video/mpeg	MPEG video clip
video/quicktime	QuickTime video clip

Common HTTP 1.1 Response Headers

- **Cache-Control (1.1) and Pragma (1.0)**
 - A no-cache value prevents browsers from caching page. Send both headers or check HTTP version.
- **Content-Encoding**
 - The way document is encoded. Browser reverses this encoding before handling document. See compression example earlier.
- **Content-Length**
 - The number of bytes in the response.
 - See setContentLength on previous slide.
 - Use ByteArrayOutputStream to buffer document before sending it, so that you can determine size. See discussion of the Connection request header and detailed example in book.

Common HTTP 1.1 Response Headers (Continued)

- **Content-Type**
 - The MIME type of the document being returned.
 - Use `setContentTypes` to set this header.
- **Expires**
 - The time at which document should be considered out-of-date and thus should no longer be cached.
 - Use `setDateHeader` to set this header.
- **Last-Modified**
 - The time document was last changed.
 - Don't set this header explicitly; provide a `getLastModified` method instead.
See example in *CSAJSP* Chapter 2.

Common HTTP 1.1 Response Headers (Continued)

- **Location**
 - The URL to which browser should reconnect.
 - Use `sendRedirect` instead of setting this directly.
- **Refresh**
 - The number of seconds until browser should reload page.
Can also include URL to connect to.
See following example.
- **Set-Cookie**
 - The cookies that browser should remember. Don't set this header directly; use `addCookie` instead. See next section.
- **WWW-Authenticate**
 - The authorization type and realm needed in Authorization header. See example in *CSAJSP* Section 4.5.

Persistent Servlet State and Auto-Reloading Pages

- **Idea: generate list of large (e.g., 150-digit) prime numbers**
 - Show partial results until completed
 - Let new clients make use of results from others
- **Demonstrates use of the Refresh header.**
- **Shows how easy it is for servlets to maintain state between requests.**
 - Very difficult in traditional CGI.
- **Also illustrates that servlets can handle multiple simultaneous connections**
 - Each request is in a separate thread.

Generating Prime Numbers: Source Code

```
public void doGet(HttpServletRequest request,
                  HttpServletResponse response)
    throws ServletException, IOException {
    int numPrimes =
        ServletUtilities.getIntParameter(request,
                                        "numPrimes", 50);

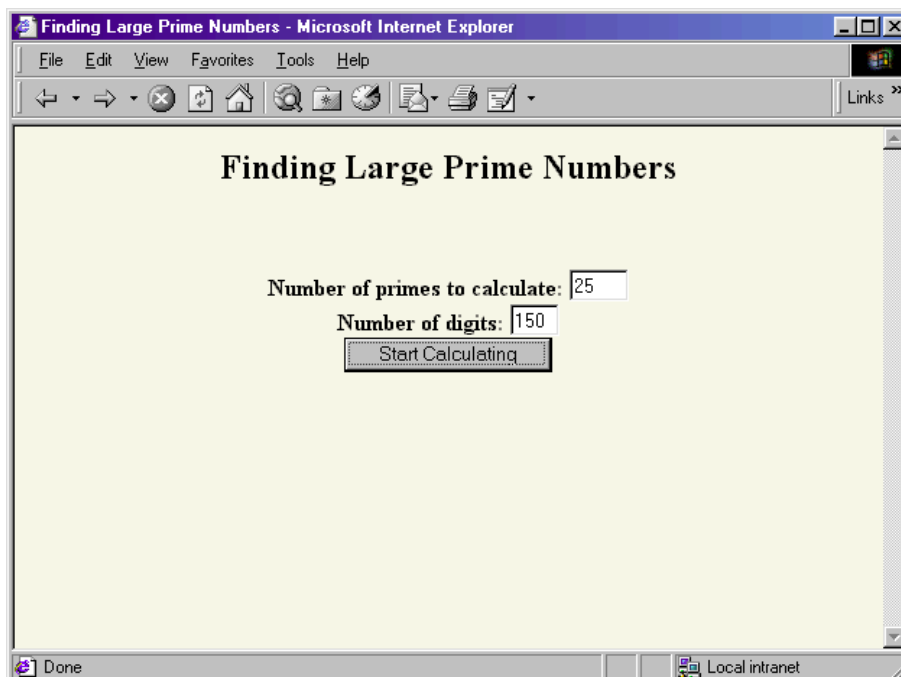
    int numDigits =
        ServletUtilities.getIntParameter(request,
                                        "numDigits", 120);

    // findPrimeList is synchronized
    PrimeList primeList =
        findPrimeList(primeListVector, numPrimes, numDigits);
    if (primeList == null) {
        primeList = new PrimeList(numPrimes, numDigits, true);
    }
}
```

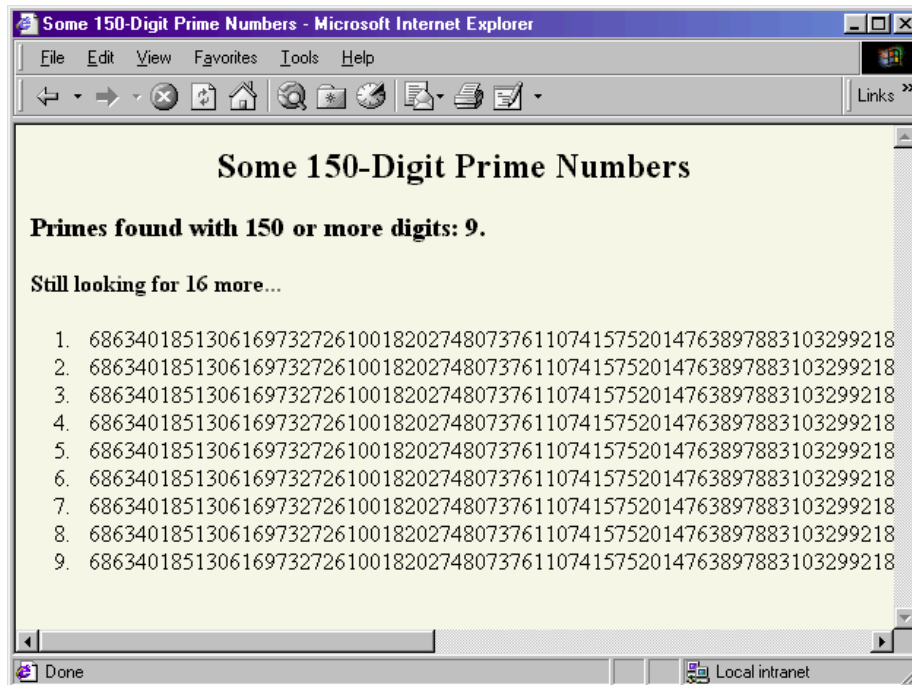
Generating Prime Numbers: Source Code (Continued)

```
synchronized(primeListVector) {  
    if (primeListVector.size() >= maxPrimeLists)  
        primeListVector.removeElementAt(0);  
    primeListVector.addElement(primeList);  
}  
}  
Vector currentPrimes = primeList.getPrimes();  
int numCurrentPrimes = currentPrimes.size();  
int numPrimesRemaining = (numPrimes - numCurrentPrimes);  
boolean isLastResult = (numPrimesRemaining == 0);  
if (!isLastResult) {  
    response.setHeader("Refresh", "5");  
}  
response.setContentType("text/html");  
PrintWriter out = response.getWriter();  
// Show List of Primes found ...
```

Prime Number Servlet: Front End



Prime Number Servlet: Initial Result

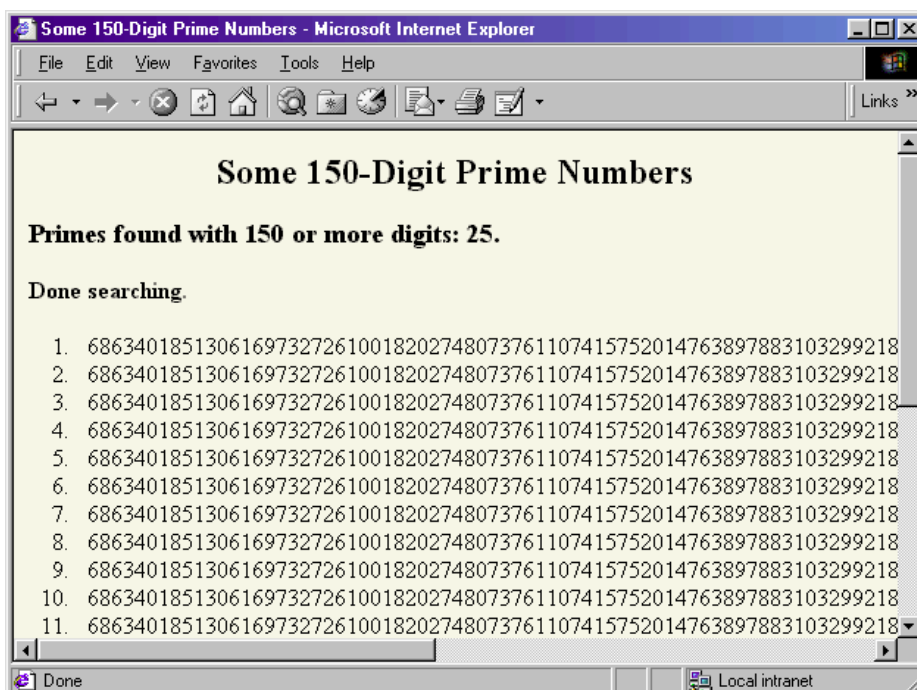


23

HTTP Response

www.coreservlets.com

Prime Number Servlet: Final Result



24

HTTP Response

www.coreservlets.com

Summary

- **Many servlet tasks can *only* be accomplished through use of HTTP status codes and headers sent to the browser**
- **Two parts of the response**
 - Status line
 - In general, set via `response.setStatus`
 - In special cases, set via `response.sendRedirect` and `response.sendError`
 - Response headers
 - In general, set via `response.setHeader`
 - In special cases, set via `response.setContentType`, `response.setContentLength`, `response.addCookie`, and `response.sendRedirect`

Summary (Continued)

- **Most important status codes**
 - 200 (default)
 - 302 (forwarding; set via `sendRedirect`)
 - 401 (password needed)
 - 404 (not found; set via `sendError`)
- **Most important headers you set directly**
 - Cache-Control and Pragma
 - Content-Encoding
 - Content-Length
 - Expires
 - Refresh
 - WWW-Authenticate