

# Java Servlets : More Topics

Ye Wu & Jeff Offutt

<http://www.cs.gmu.edu/~offutt/>

SWE 642

Software Engineering for the World Wide Web

sources: Professional Java Server Programming, Patzer, Wrox

## Topics

1. `HttpServletRequest`
2. `HttpServletResponse`
3. Uploading files
4. Concurrency
5. Deploying servlets and servlet contexts

## More of the HttpServletRequest Interface

- Methods to request parameter values :  
getParameter(), getParameterValues(), getParameterNames()
- Methods to access the request as bytes :  
getInputStream(), getContentLength(), getContentType()
- Methods to request path and URL :  
getPathInfo(), getPathTranslated(), getQueryString(), getRequestURI(),  
getServletPath()
- Methods to get information about the sender  
getRemoteAddr(), getRemoteHost(), getRemotePort()

<http://java.sun.com/j2ee/1.4/docs/api/javax/servlet/ServletRequest.html>

## Topics

1. HttpServletRequest
2. HttpServletResponse
3. Uploading files
4. Concurrency
5. Deploying servlets and servlet contexts

## More of the HttpServletResponse Interface

- Methods to set content type and length  
setContentType(), setContentLength()
- Methods to create output  
getOutputStream(), getWriter()
- Methods to handle buffered output
  - setBufferSize(), getBufferSize(), flushBuffer(), isCommitted(), reset()

<http://java.sun.com/j2ee/1.4/docs/api/javax/servlet/ServletResponse.html>

## Topics

1. HttpServletRequest
2. HttpServletResponse
3. Uploading files
4. Concurrency
5. Deploying servlets and servlet contexts

## Uploading Files to Servlets

### 1. Client Side

- `<FORM ACTION="..." METHOD="POST" ENCTYPE = "multipart/form-data">`
- File to upload : `<input type="file" name="file">`

### 2. When file is submitted, browsers specify content and add a "boundary" marker

### 3. Data in the format below must be parsed ...

Content-type: multipart/form-data, boundary=AaB03x

--AaB03x

content-disposition: form-data; name="field1"

Joe Blow

--AaB03x

content-disposition: form-data; name="pics"; filename="file1.txt"

Content-Type: text/plain

... contents of file1.txt ...

--AaB03x--

<http://www.ietf.org/rfc/rfc1876.txt>

8/9/2010

© Wu and Offutt

7

## Uploading Files to Servlets (2)

### 3. Server-side

- Get data length and data  
`DataInputStream in = new DataInputStream (req.getInputStream());`  
`int formDataLength = req.getContentLength();`  
`byte dataByte[] = new byte [formDataLength];`  
`dataByte [index] = in.readByte(); // loop over each byte`
- Get boundary  
`String contentType = req.getContentType();`  
`int lastIndex = contentType.lastIndexOf ("=");`  
`String boundary = contentType.substring (lastIndex+1, contentType.length());`
- Get file name and content  
`int pos = file.indexOf ("filename="); // index of string`  
`pos = file.indexOf ("\n", pos) + 1; // index of name of file`

<http://cs.gmu.edu/~offutt/classes/642/examples/servlets/>

8/9/2010

© Wu and Offutt

8

## Topics

1. HttpServletRequest
2. HttpServletResponse
3. Uploading files
4. Concurrency
5. Deploying servlets and servlet contexts

8/9/2010

© Wu and Offutt

9

## Concurrency

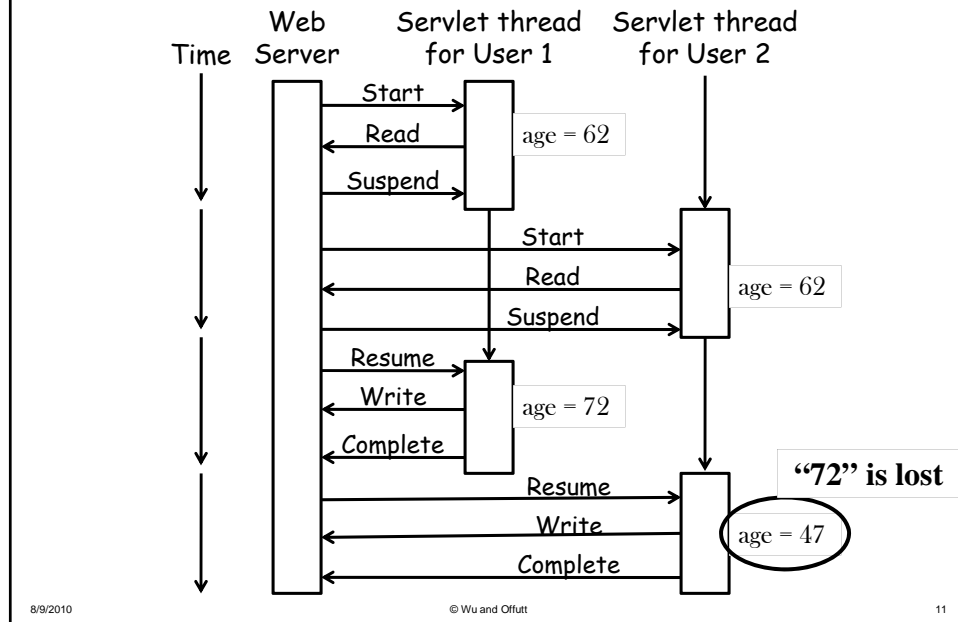
- Multiple users often access the same servlet at the same time
- Servlet objects are run as threads
- Threads on the same object share some variables
- This can cause problems when storing data into permanent storage
  1. Servlet thread for user 1 reads a data file into memory
  2. Servlet thread for user 2 reads the same data file
  3. Thread 1 changes the data
  4. Thread 2 changes the data
  5. Thread 1 writes the file onto disk
  6. Thread 2 writes the file onto disk ...
  7. Changes from user 1 ~~are~~ **lost** !
- Programmers are responsible for avoiding this problem

8/9/2010

© Wu and Offutt

10

## Concurrency Illustration



## Concurrency and Java Threads

- Tomcat creates a separate thread for each HTTP request
- Some state is unique to each thread (not shared)
  - Next statement to execute
  - The call stack :
    - Where the current method will return to
    - Where the calling method returns to
    - Parameter values and local values for each method
- Some state is shared among threads
  - Instance variable values (declared outside methods)
  - Class variables values (declared static outside methods)
  - Contents of files and other external resources

8/9/2010

© Wu and Offutt

12

## Concurrency Example

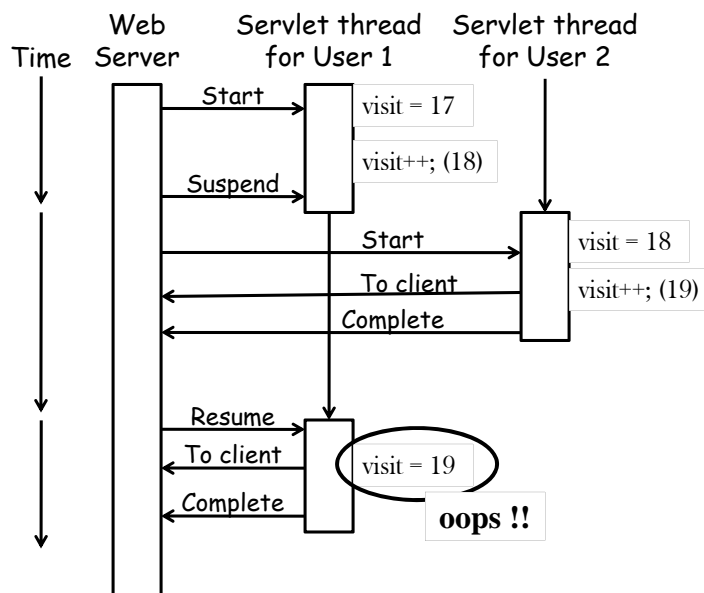
```
public class countHellos extends HttpServlet
{
    // Number of times the servlet has been executed since
    // the servlet object was created
    private int visit = 0; // shared among threads
    public void doGet (HttpServletRequest req,
                      HttpServletResponse res)
                      throws ServletException, IOException
    {
        res.setContentType ("text/html; charset=\"UTF-8\"");
        PrintWriter out = res.getWriter ();
        visit ++;
        out.println ("<HTML>");
        . . .
    }
}
```

8/9/2010

© Wu and Offutt

13

## Concurrency Illustration



8/9/2010

© Wu and Offutt

14

## Concurrency and Synchronization

- The Java “synchronized” command can be used to avoid concurrency problems
- If a method is synchronized, only one thread can execute the method at a time
  - Other threads are blocked until the first thread finishes

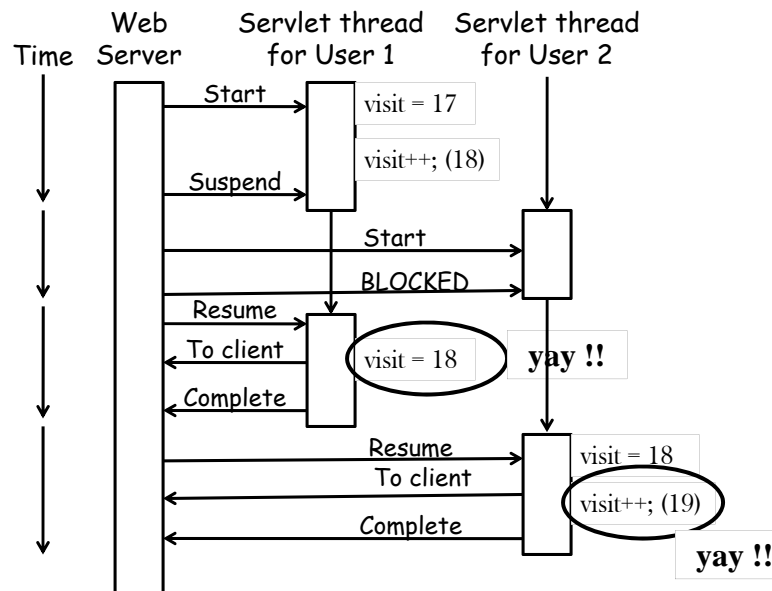
```
synchronized public void doGet (HttpServletRequest req,  
                                HttpServletResponse res)  
    throws ServletException, IOException  
{
```

8/9/2010

© Wu and Offutt

15

## Concurrency Illustration



8/9/2010

© Wu and Offutt

16

## Concurrency—Synchronization Hints

- Using synchronization will slow down users
  - They have to wait in line behind other users
- Try to synchronize “low”, not “high”
  - Takes more careful analysis—harder design
  - Is faster for users
- Synchronizing on statement blocks rather than entire methods is likewise more efficient
  - `synchronize (anyObject) { ... statements ...}`
- Be very careful synchronizing statements that make method calls ! (See Bloch)
- When two classes or servlets need to share a resource (file or database), put synchronized methods in a separate class

8/9/2010

© Wu and Offutt

17

## Topics

1. HttpServletRequest
2. HttpServletResponse
3. Uploading files
4. Concurrency
5. Deploying servlets and servlet contexts

8/9/2010

© Wu and Offutt

18

## Deploying Servlets : Servlet Contexts

- Every servlet is deployed as part of a servlet context
- Servlet context defines :
  - Location of servlet .class file
  - Servlets that it can interact with
- Servlet contexts are set up by the system administrator

8/9/2010

© Wu and Offutt

19

## Servlet Contexts : Platforms

- Most web applications are deployed on Unix or Linux servers
- Many view Windows / IIS servers as not being :
  - Reliable enough
  - Robust enough
  - Secure enough
- Administering and maintaining servlet engines is more complicated on Unix platforms than on a local standalone development platform
- Serious web application developers must know at least a little about Unix / Linux

8/9/2010

© Wu and Offutt

20

## Servlet Context

- URI : Uniform Resource Identifier
  - A string that identifies an internet resource
  - Several schemes : (`http`, `ftp`, ...)
  - domain name (`apps-642.ite.gmu.edu`)
  - port number (`8080`)
  - path (`swe64202/servlet/offutt.Hello`)
  - parameters (`?name=dumb`)
- A URL is a specific type of URI
  - http scheme
- URI path : The path inside the URI

8/9/2010

© Wu and Offutt

21

## Servlet Context (2)

- Each servlet is mapped to a specific URI:  
`apps-swe642.ite.gmu.edu:8080/swe64202/servlet/offutt.Hello`  

domain path maps to servlet
- For servlet contexts, each context must be mapped to a path prefix on disk:  
`/apps/tomcat/swe642/WEB-INF/classes/`

8/9/2010

© Wu and Offutt

22

## Installing Servlet Contexts

- The system administrator must tell :
  - Server about contexts: where classes directory is
  - Servlet engine : a name mapping for the servlet pages
  - Server : to start contexts upon initialization
- On hermes, we have one servlet context for the entire class
- This is inconvenient in some ways, but much easier to set up and administrate

8/9/2010

© Wu and Offutt

23

## Tomcat Setup on Windows

- Default location for Tomcat : `C:\Tomcat\webapps\`
- Pre-installed examples are in :
  - `C:\Tomcat\webapps\examples\WEB-INF\classes\`
  - `http://localhost:8080/example/servlet/HelloWorldExample`
- To create a new context (project) named `xyzzzy`:
  - `C:\Tomcat\webapps\xyzzzy\`
  - `WEB-INF\classes\`
    - Copy `sessions/listeners/compressionfilters/filters/` from examples
  - `WEB-INF\jsp`
    - Copy `debug-taglib.tld` & `example-taglib.tld` from examples

8/9/2010

© Wu and Offutt

24

## Summary

- The API we use to program servlets is rich with functionality
- Writing servlets is easier than
  - “Wiring” them together
  - Deploying them
  - Debugging them