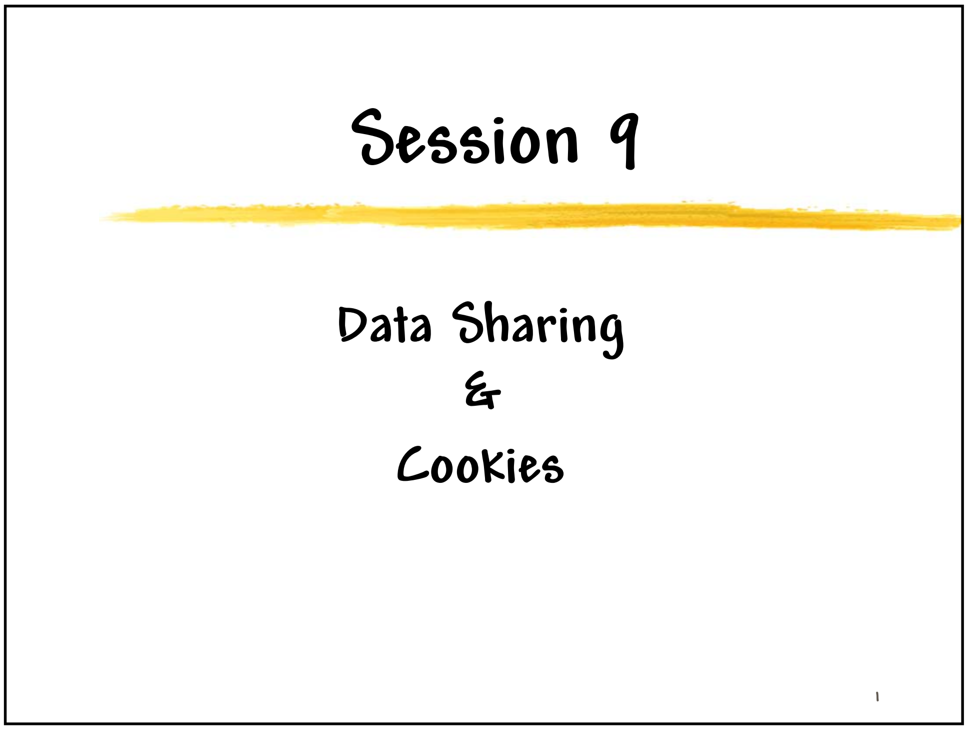
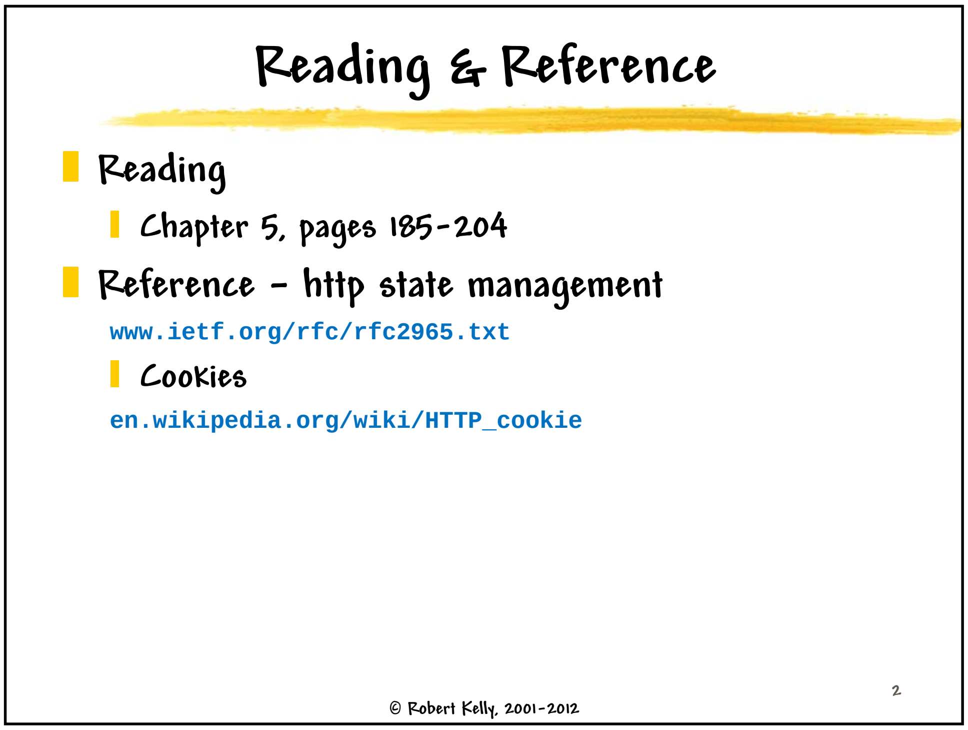


Session 9

Data Sharing
&
Cookies

1



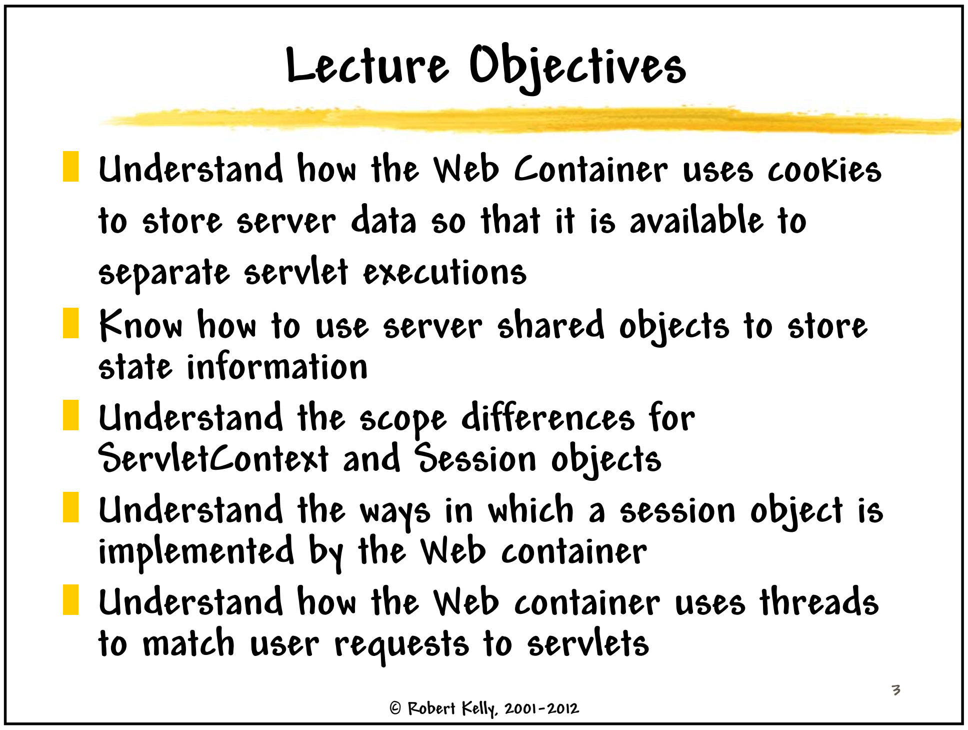
Reading & Reference

- Reading
  - Chapter 5, pages 185-204
- Reference - http state management

  www.ietf.org/rfc/rfc2965.txt
  - Cookies

  en.wikipedia.org/wiki/HTTP_cookie

2

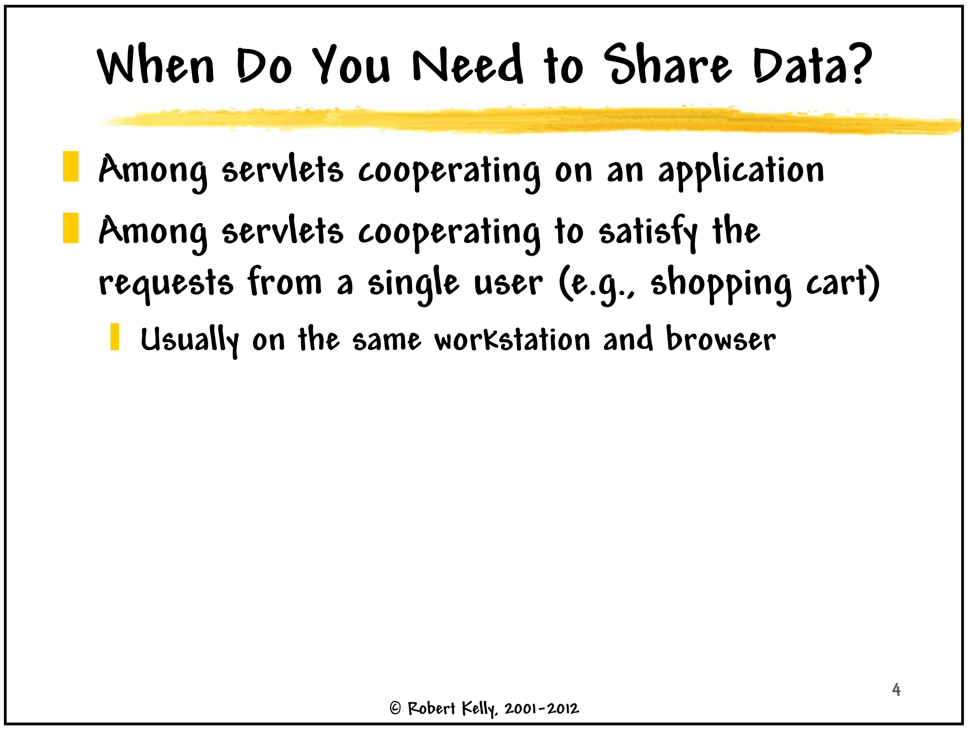


Lecture Objectives

- Understand how the Web Container uses cookies to store server data so that it is available to separate servlet executions
- Know how to use server shared objects to store state information
- Understand the scope differences for ServletContext and Session objects
- Understand the ways in which a session object is implemented by the Web container
- Understand how the Web container uses threads to match user requests to servlets

3

© Robert Kelly, 2001-2012

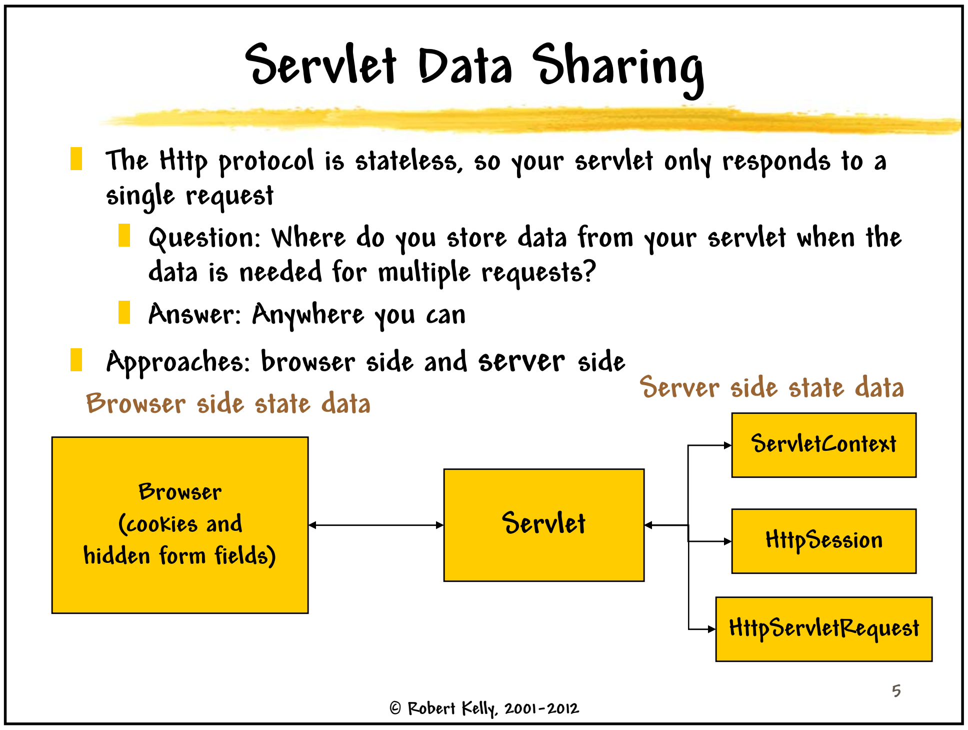


When Do You Need to Share Data?

- Among servlets cooperating on an application
- Among servlets cooperating to satisfy the requests from a single user (e.g., shopping cart)
  - Usually on the same workstation and browser

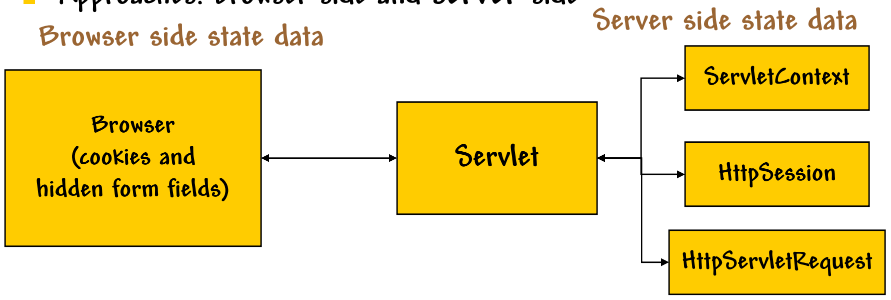
4

© Robert Kelly, 2001-2012

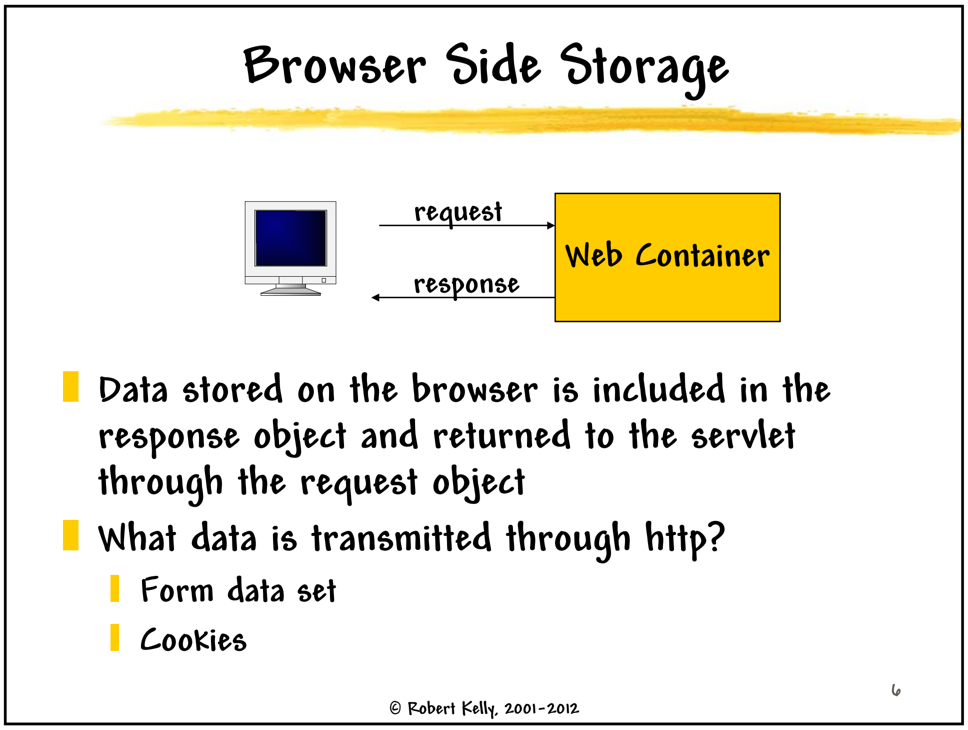


Servlet Data Sharing

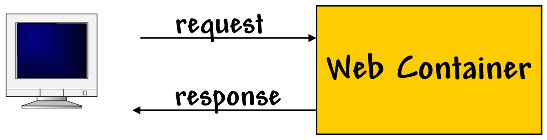
- The Http protocol is stateless, so your servlet only responds to a single request
  - Question: Where do you store data from your servlet when the data is needed for multiple requests?
  - Answer: Anywhere you can
- Approaches: browser side and server side



© Robert Kelly, 2001-2012 5



Browser Side Storage



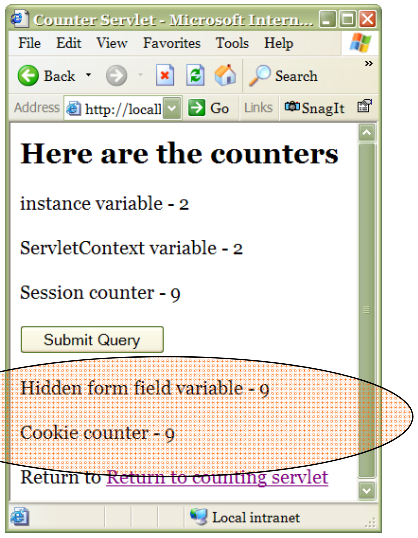
- Data stored on the browser is included in the response object and returned to the servlet through the request object
- What data is transmitted through http?
  - Form data set
  - Cookies

© Robert Kelly, 2001-2012 6

Data Sharing Example

- Implement a counter using:
  - ☑ Hidden form field
  - ☑ Cookies
  - ServletContext
  - Session

The servlet stores a counter that is updated whenever a user sends another request



Here are the counters

instance variable - 2

ServletContext variable - 2

Session counter - 9

Submit Query

Hidden form field variable - 9

Cookie counter - 9

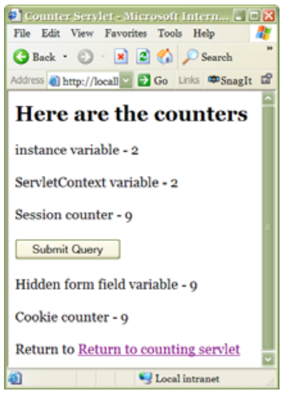
Return to Return to counting servlet

© Robert Kelly, 2001-2012 7

Browser Storage: Hidden Form Fields

- Store the data in a form field generated by your servlet.

The hidden form field is not displayed in the browser window



Here are the counters

instance variable - 2

ServletContext variable - 2

Session counter - 9

Submit Query

Hidden form field variable - 9

Cookie counter - 9

Return to Return to counting servlet

```
<form action="http://localhost:8080/CodeCSE336/datasharing">
<input type = "hidden" name = "hiddenCounter" value = "1">
<input type="submit">
</form>
```

The value of hiddenCounter is 1 the first time the servlet is invoked

© Robert Kelly, 2001-2012 8

Example - Hidden Form Field Code

Servlet Code Fragment

```
String sCounter = request.getParameter("hiddenCounter");
   if (sCounter != null)
       hffCount = Integer.parseInt(sCounter)
   else hffCount = 0;

   hffCount++;
   out.println("<form
   action=\"http://localhost:8080/CodeCSE336/datasharing\">");
   out.println("<input type = \"hidden\" name =
   \"hiddenCounter\" value = \""
                  + hffCount + "\">");
     out.println("<input type=\"submit\">");
     out.println("</form>");
     out.println("<p>Hidden form field variable - ");
     out.println(hffCount + "</p>");
```

Counter is updated and added to html

© Robert Kelly, 2001-2012 9

Hidden Form Fields

- Features
  - Visible in the html, not visible on the displayed page
  - Only available as long as the html page is "alive" in the browser
  - Only associated with a browser in a computer

There are some hidden form fields in your project - but you can remove them

© Robert Kelly, 2001-2012 10

Cookies

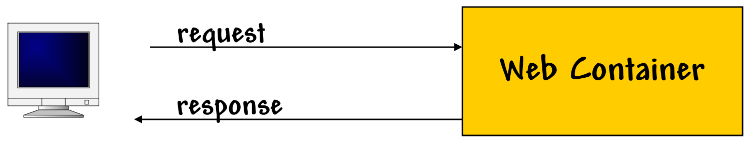
- A cookie is a small amount of information sent by the server to the browser that can later be read back from the browser
- Usually contained in a Cookies folder

Typical cookie

```
Adc1
11780|NY56|078|@NY|ISP|ISP
accuweather.com/
0
3337461760
29399690
101711582429393656
*
Adc2
5|1|40.88|-73.16|SAINT JAMES
...
```

© Robert Kelly, 2001-2012 11

Cookie Process



1. Your servlet "sets a cookie" by including it in the response

2. Your browser stores the cookie In your cookies directory on your hard disk

3. Your browser sends the cookie every time a request is made to a server "in your domain"

© Robert Kelly, 2001-2012 12

Cookies

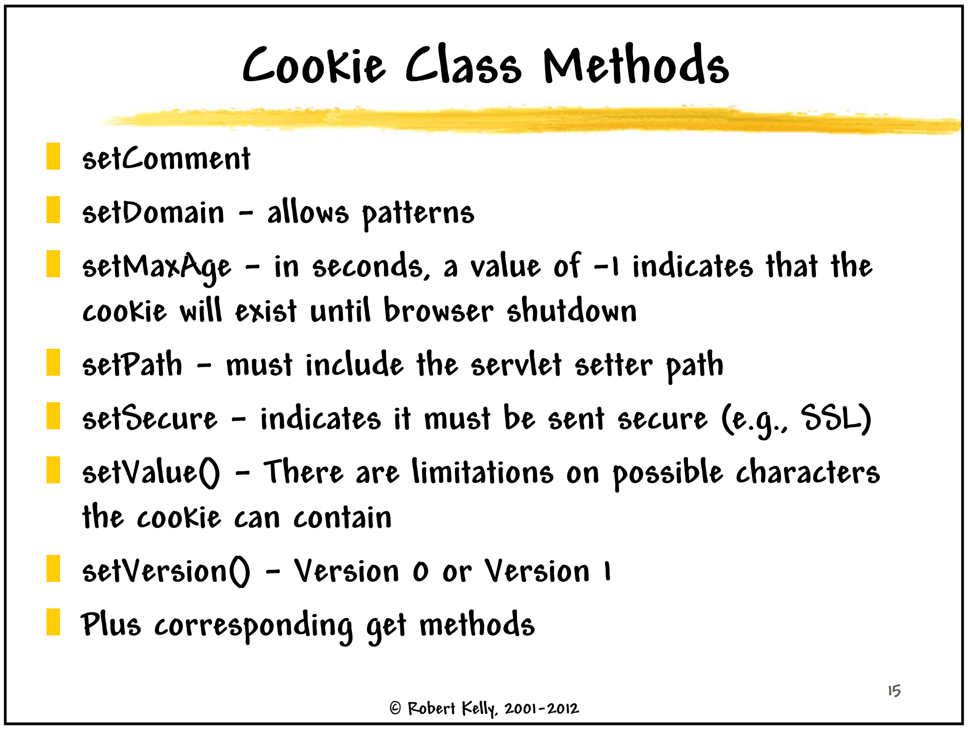
- Cookies set by a server are returned to the server each time the browser accesses a corresponding page on the server
- Cookies sent by a browser are sent based on the server name
- Cookies are included in the http header info
- Most browsers support cookies (up to 20 per site and up to 4KB per cookie)
- Multiple cookies can have the same name
- However, users can turn cookies off

© Robert Kelly, 2001-2012 13

Purposes

- Session information
- Site recognition – sometimes instead of an ID/Password
- Site customization
- Focused advertising

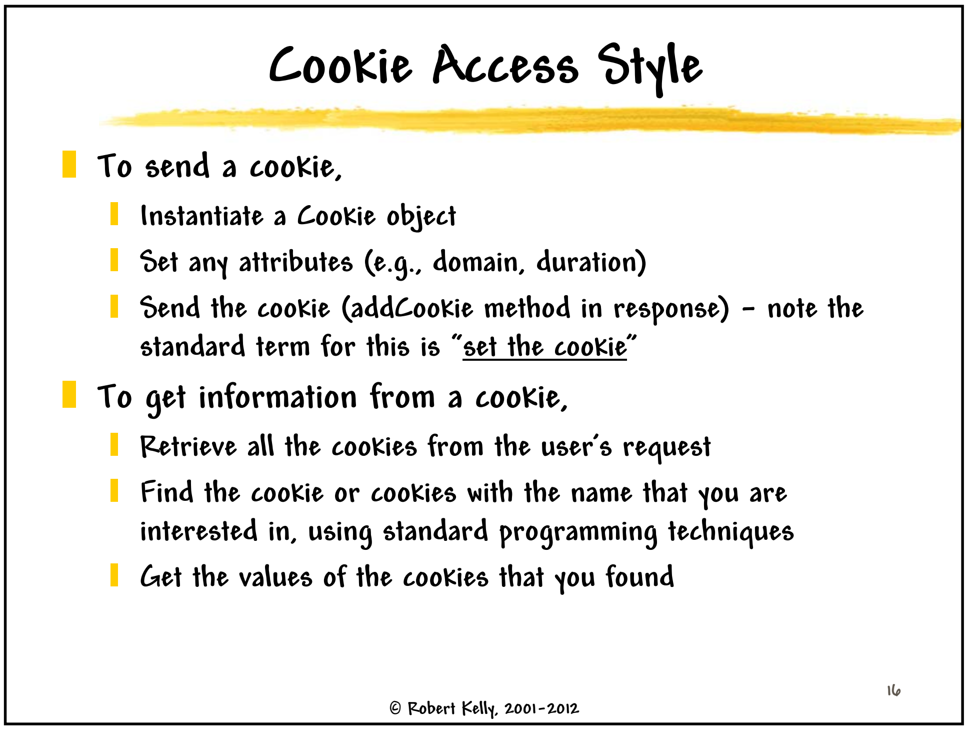
© Robert Kelly, 2001-2012 14



Cookie Class Methods

- setComment
- setDomain - allows patterns
- setMaxAge - in seconds, a value of -1 indicates that the cookie will exist until browser shutdown
- setPath - must include the servlet setter path
- setSecure - indicates it must be sent secure (e.g., SSL)
- setValue() - There are limitations on possible characters the cookie can contain
- setVersion() - Version 0 or Version 1
- Plus corresponding get methods

© Robert Kelly, 2001-2012 15



Cookie Access Style

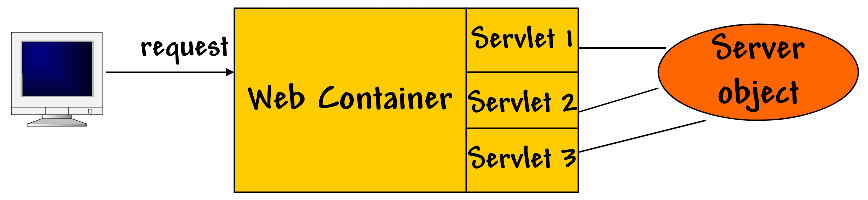
- To send a cookie,
  - Instantiate a Cookie object
  - Set any attributes (e.g., domain, duration)
  - Send the cookie (addCookie method in response) - note the standard term for this is "set the cookie"
- To get information from a cookie,
  - Retrieve all the cookies from the user's request
  - Find the cookie or cookies with the name that you are interested in, using standard programming techniques
  - Get the values of the cookies that you found

© Robert Kelly, 2001-2012 16

Cookies Summary

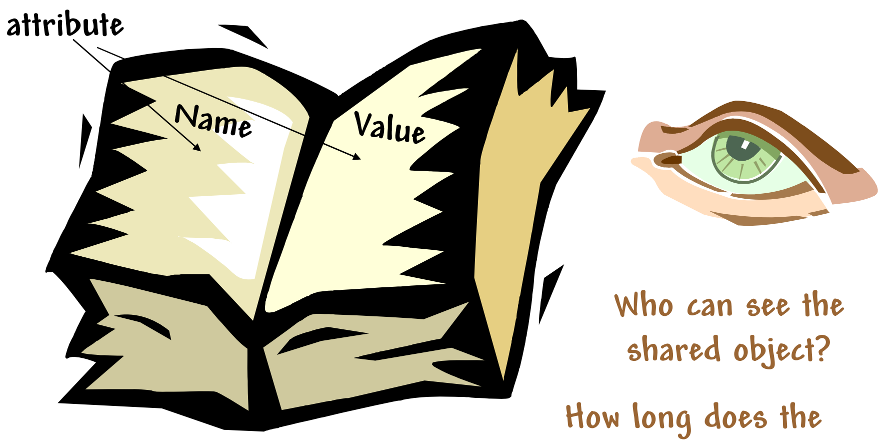
- Single system data sharing - Usable on any system that stores the server's cookie file
- Extra data traffic for all requests to the server (from that browser)
- Somewhat unreliable – user may have cookies disabled

Server Side Storage



- Data stored on the server is usually contained in an object visible to the servlet
- To access the shared object, you need to obtain a reference (handle) to the object
- Objects for sharing
  - HttpServletRequest
  - ServletContext
  - Session
  - Other predefined and private objects

Shared Objects



Who can see the shared object?

How long does the shared attribute last?

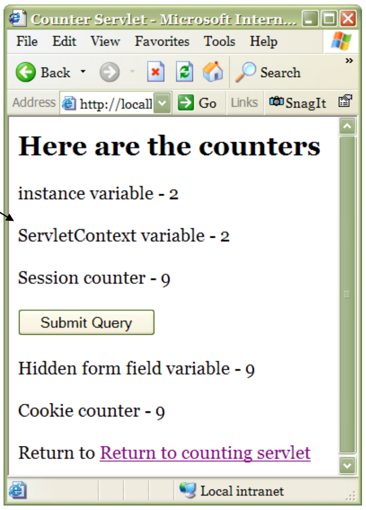
An attribute is a name/value pair

© Robert Kelly, 2001-2012

19

Example – Counter with ServletContext

- The servlet uses the ServletContext to store the access count


Counter Servlet - Microsoft Intern...
File Edit View Favorites Tools Help
Back
Search
Address http://locall
Go
Links
SnagIt
Submit Query
Local intranet

Here are the counters

instance variable - 2

ServletContext variable - 2

Session counter - 9

Hidden form field variable - 9

Cookie counter - 9

Return to Return to counting servlet

© Robert Kelly, 2001-2012

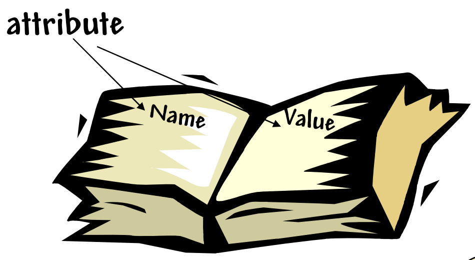
20

Example – ServletContext Code

Get a handle to the ServletContext

```
...
 ServletContext sc = this.getServletContext();
 Integer c = (Integer) sc.getAttribute("counter");
 if (c != null)
     scCount = c.intValue();
 else scCount = 0;
 scCount++;
 sc.setAttribute("counter", new Integer(scCount));
   out.println("<p>ServletContext variable - ");
   out.println(scCount + "</p>");
```

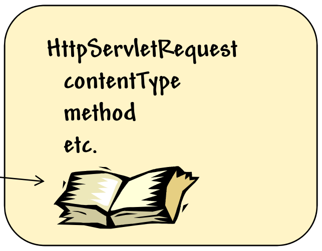
Shared Objects

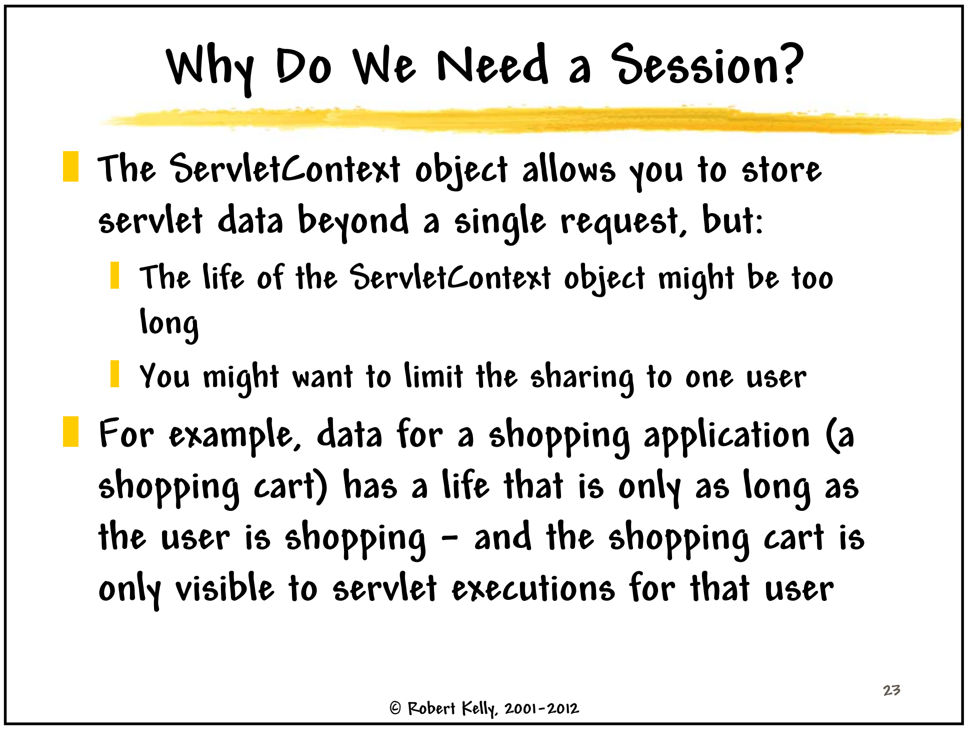


The shared objects are referred to as "scopes"

The shared scopes are contained in other objects

For example, the request object contains a scope

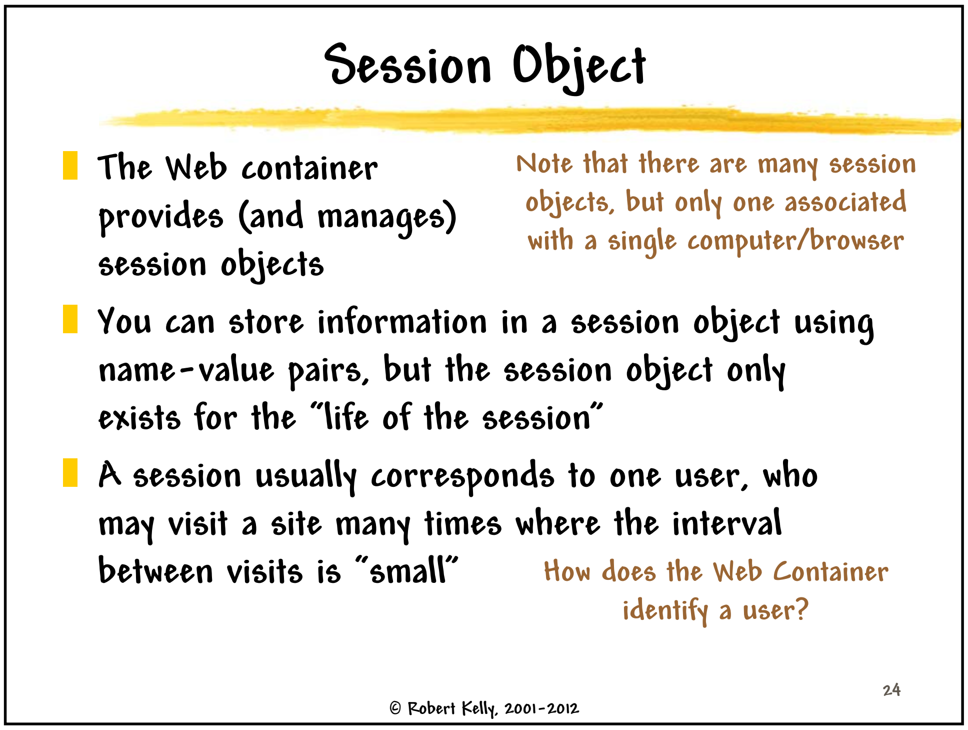




Why Do We Need a Session?

- The ServletContext object allows you to store servlet data beyond a single request, but:
  - The life of the ServletContext object might be too long
  - You might want to limit the sharing to one user
- For example, data for a shopping application (a shopping cart) has a life that is only as long as the user is shopping - and the shopping cart is only visible to servlet executions for that user

23



Session Object

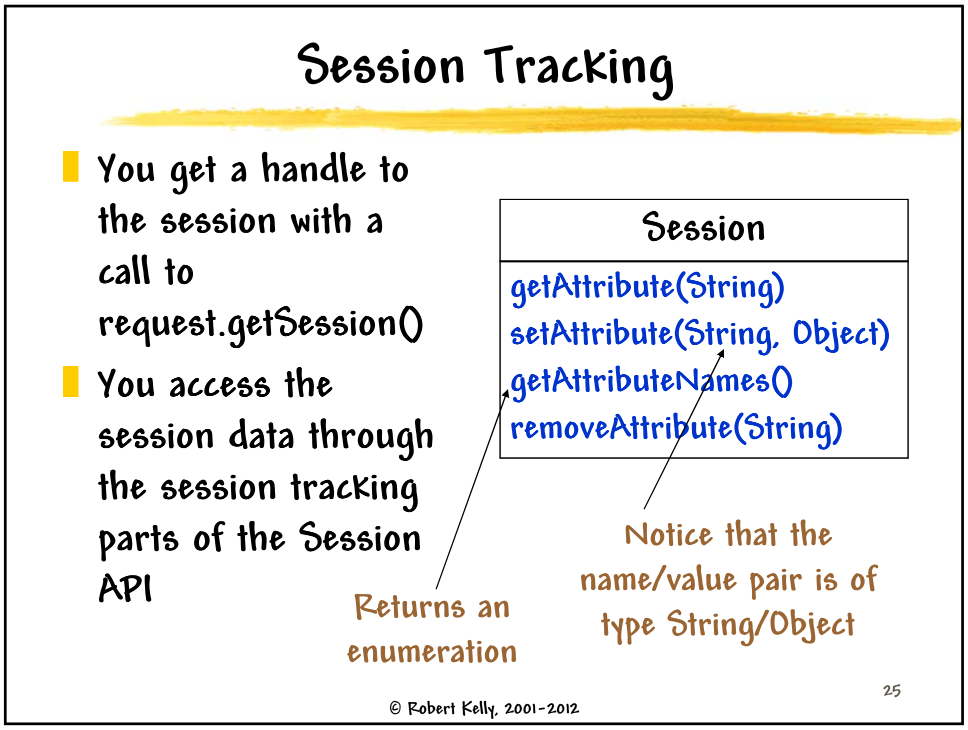
- The Web container provides (and manages) session objects

Note that there are many session objects, but only one associated with a single computer/browser

- You can store information in a session object using name-value pairs, but the session object only exists for the "life of the session"
- A session usually corresponds to one user, who may visit a site many times where the interval between visits is "small"

How does the Web Container identify a user?

24



Session Tracking

- You get a handle to the session with a call to request.getSession()
- You access the session data through the session tracking parts of the Session API

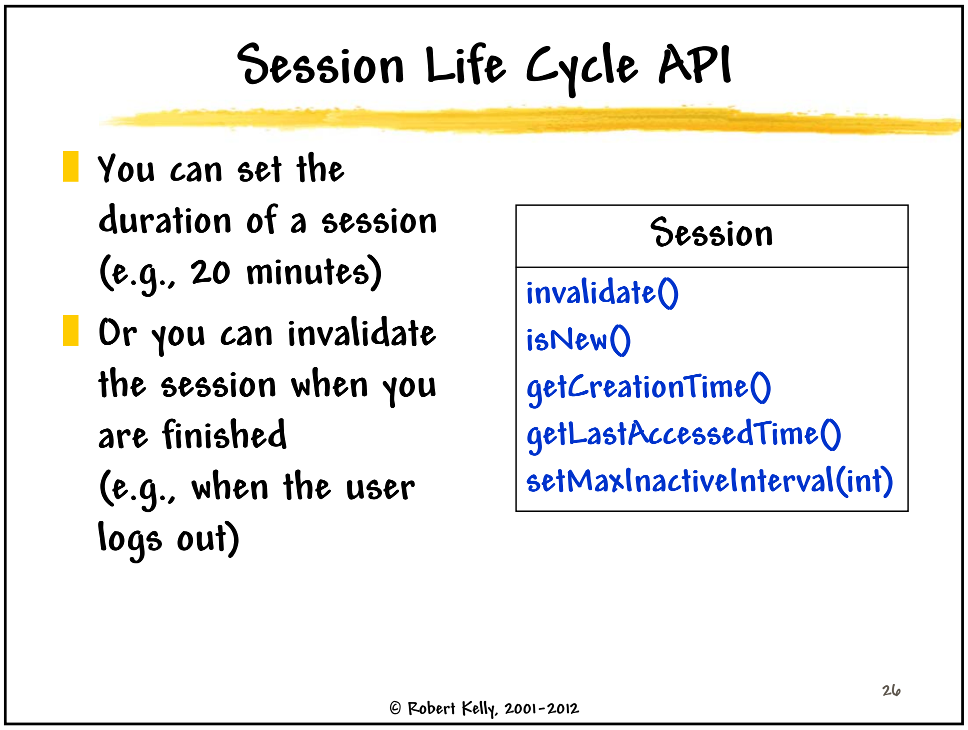
Session
getAttribute(String)
setAttribute(String, Object)
getAttributeNames()
removeAttribute(String)

Returns an enumeration

Notice that the name/value pair is of type String/Object

© Robert Kelly, 2001-2012

25



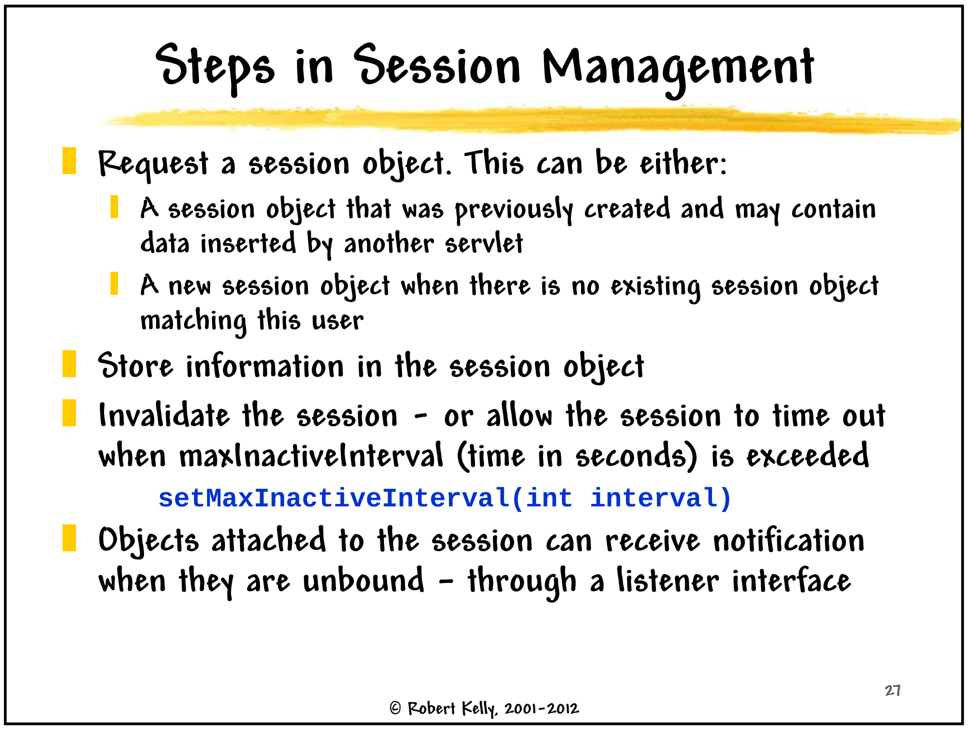
Session Life Cycle API

- You can set the duration of a session (e.g., 20 minutes)
- Or you can invalidate the session when you are finished (e.g., when the user logs out)

Session
invalidate()
isNew()
getCreationTime()
getLastAccessedTime()
setMaxInactiveInterval(int)

© Robert Kelly, 2001-2012

26



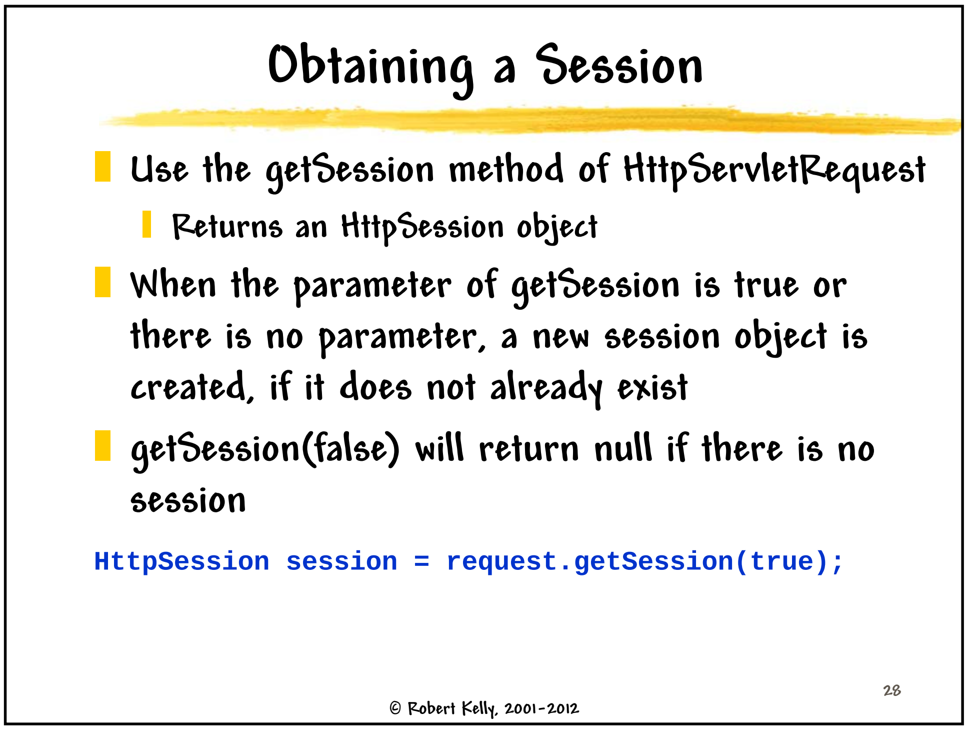
Steps in Session Management

- Request a session object. This can be either:
  - A session object that was previously created and may contain data inserted by another servlet
  - A new session object when there is no existing session object matching this user
- Store information in the session object
- Invalidate the session - or allow the session to time out when maxInactiveInterval (time in seconds) is exceeded

```
setMaxInactiveInterval(int interval)
```

- Objects attached to the session can receive notification when they are unbound - through a listener interface

© Robert Kelly, 2001-2012 27



Obtaining a Session

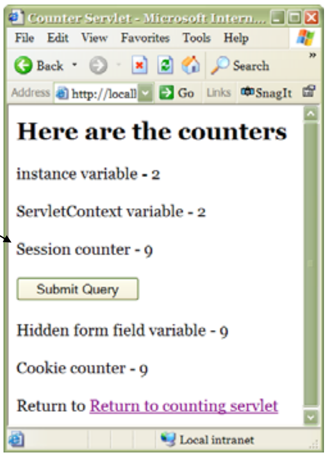
- Use the getSession method of HttpServletRequest
  - Returns an HttpSession object
- When the parameter of getSession is true or there is no parameter, a new session object is created, if it does not already exist
- getSession(false) will return null if there is no session

```
HttpSession session = request.getSession(true);
```

© Robert Kelly, 2001-2012 28

Example – Counter with Session

- The servlet uses the Session to store the access count


Counter Servlet - Microsoft Intern...
File Edit View Favorites Tools Help
Back Search
Address http://locall Go Links SnagIt
Submit Query
Local intranet

Here are the counters

instance variable - 2

ServletContext variable - 2

Session counter - 9

Hidden form field variable - 9

Cookie counter - 9

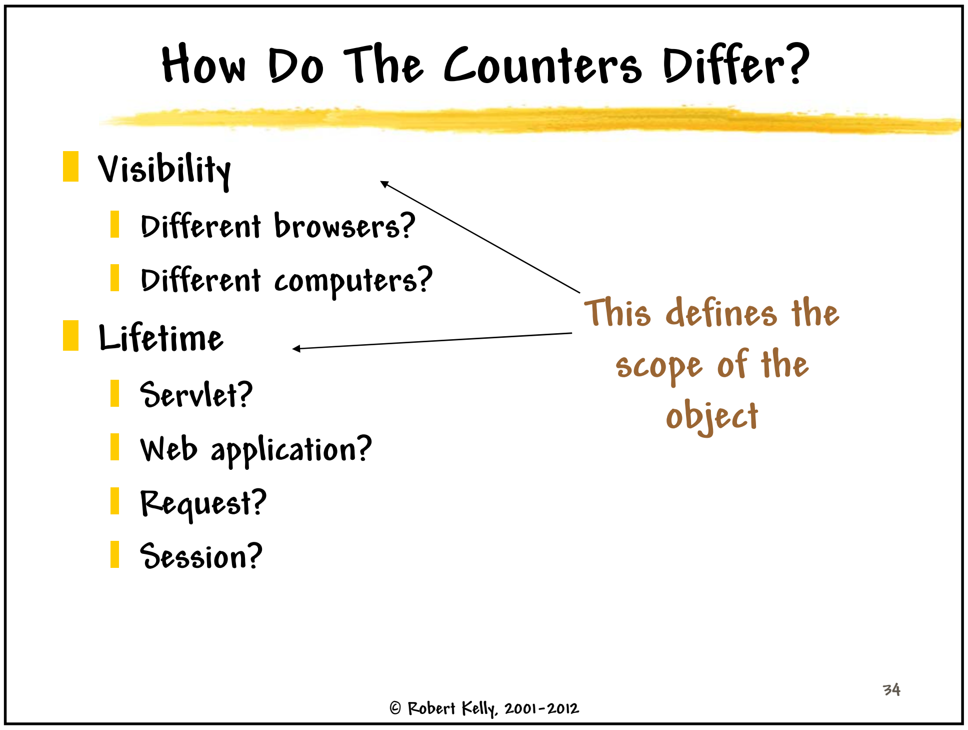
Return to Return to counting servlet

Session Counter

```
HttpSession hs = request.getSession(true);
c = (Integer) hs.getAttribute("counter");

if (c != null)
    hsCount = c.intValue();
else hsCount = 0;
hsCount++;
hs.setAttribute("counter", new Integer(hsCount));
out.println("<p>Session counter - ");
out.println(hsCount + "</p>");
```

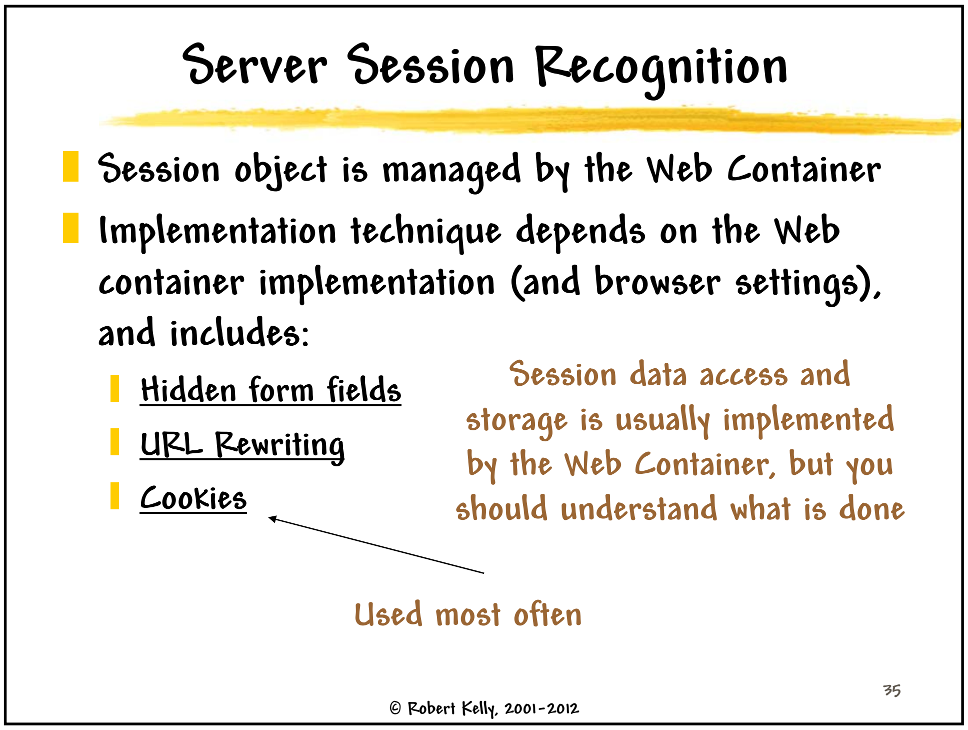
getAttribute returns an Object, which we cast to Integer



How Do The Counters Differ?

- Visibility
  - Different browsers?
  - Different computers?
- Lifetime
  - Servlet?
  - Web application?
  - Request?
  - Session?

This defines the scope of the object



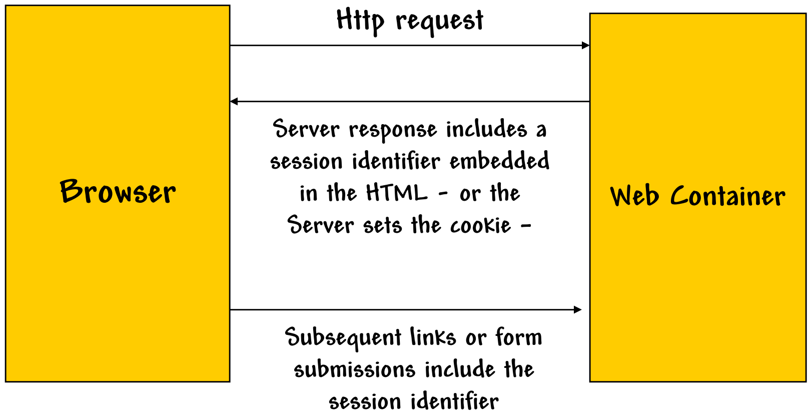
Server Session Recognition

- Session object is managed by the Web Container
- Implementation technique depends on the Web container implementation (and browser settings), and includes:
  - Hidden form fields
  - URL Rewriting
  - Cookies

Session data access and storage is usually implemented by the Web Container, but you should understand what is done

Used most often

Session Identifier Strategies


Browser
Http request
Server response includes a session identifier embedded in the HTML - or the Server sets the cookie -
Subsequent links or form submissions include the session identifier
Web Container

© Robert Kelly, 2001-2012

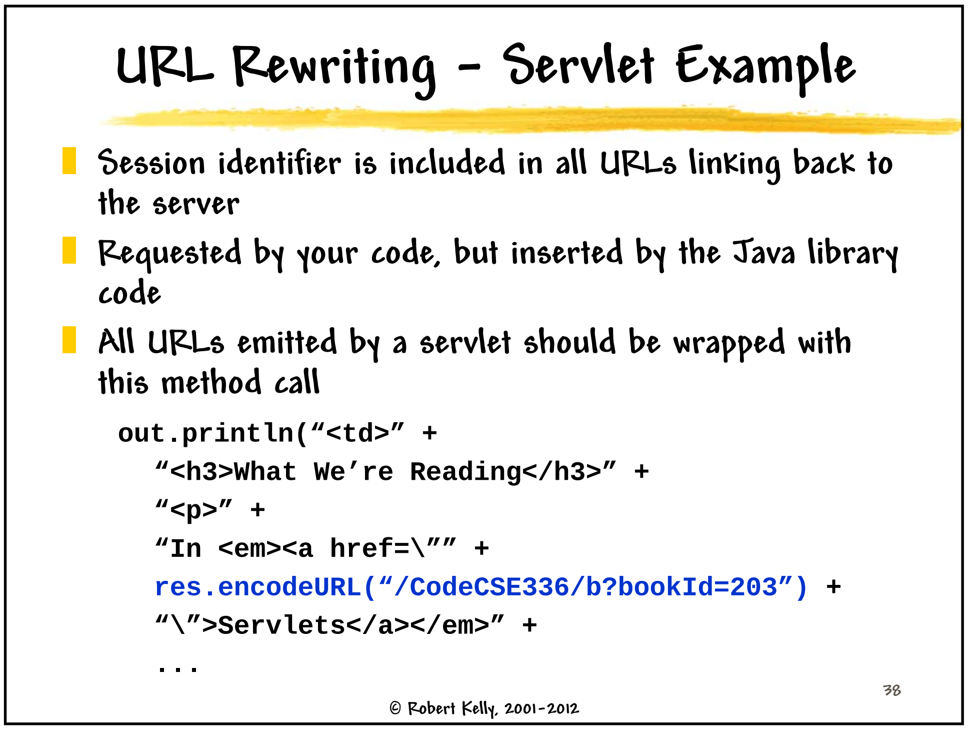
36

URL Rewriting

- The session identifier is included in all relevant URLs in the HTML sent from the server to the browser
- Subsequent links to the URL include the pre-coded session data
- HTML Example:
  `http://server:port/servlet/Rewriter?sessionid=123`
- Technique only works for dynamically generated pages
- Potential for errors

© Robert Kelly, 2001-2012

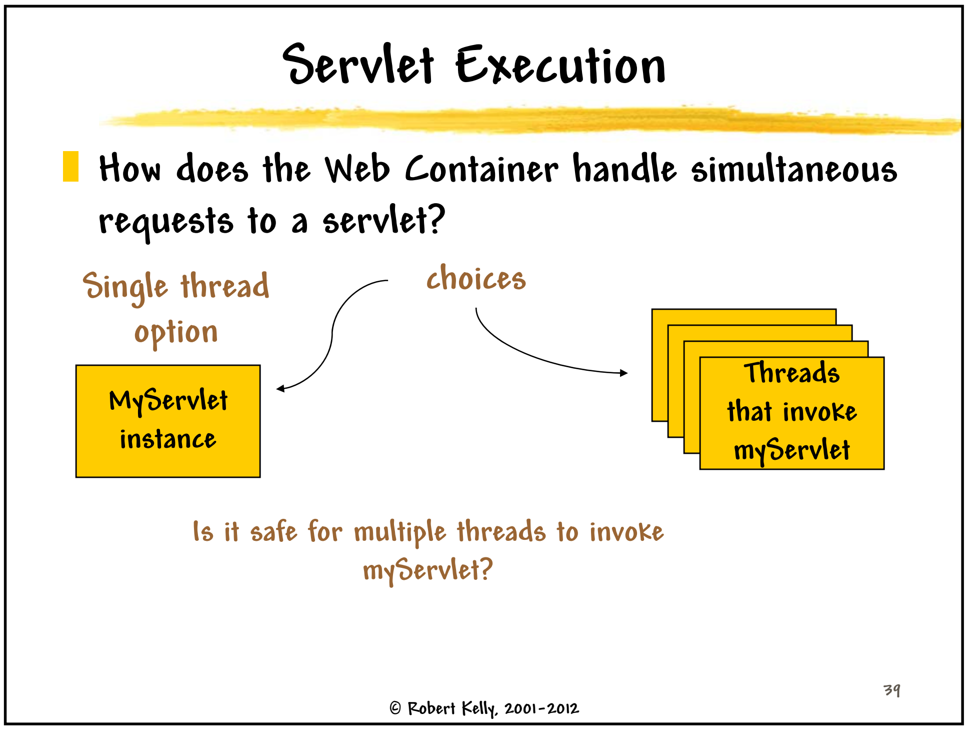
37



URL Rewriting - Servlet Example

- Session identifier is included in all URLs linking back to the server
- Requested by your code, but inserted by the Java library code
- All URLs emitted by a servlet should be wrapped with this method call

```
out.println("<td>" +
   "<h3>What We're Reading</h3>" +
   "<p>" +
   "In <em><a href=\"" +
   res.encodeURL("/CodeCSE336/b?bookId=203") +
   "\">Servlets</a></em>" +
   ...
```

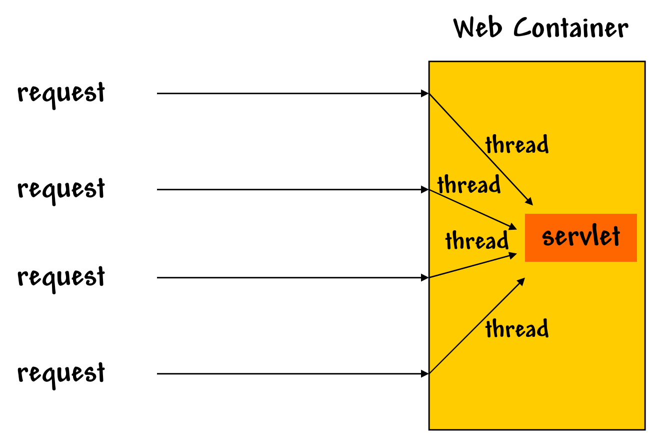


Servlet Execution

- How does the Web Container handle simultaneous requests to a servlet?

Is it safe for multiple threads to invoke myServlet?

Multi-threaded Servlet Access



© Robert Kelly, 2001-2012

40

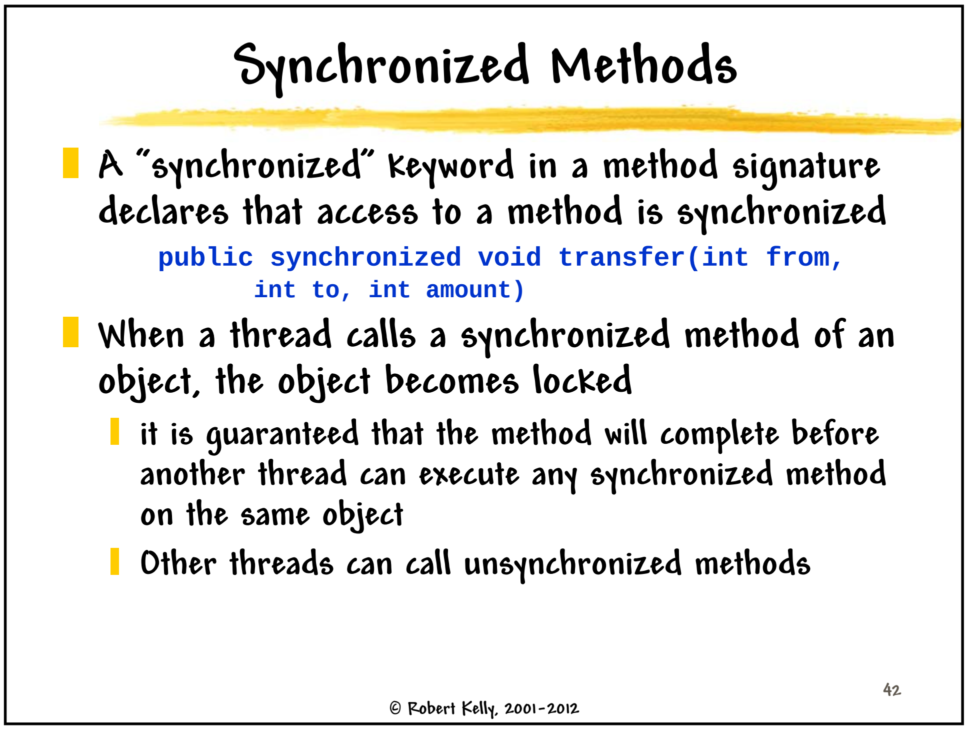
Synchronization

- It is possible for 2 or more threads to have access to the same object (or primitive)
- Most operations are not indivisible (modifications require multiple machine instructions), so corruption can result (called a <u>race condition</u>)
- To avoid simultaneous access to a shared object, you <u>synchronize</u> access to the object
  - Synchronized method
  - Synchronized block
  - Single Thread Model

Remember: a servlet local variable is not shared

© Robert Kelly, 2001-2012

41



Synchronized Methods

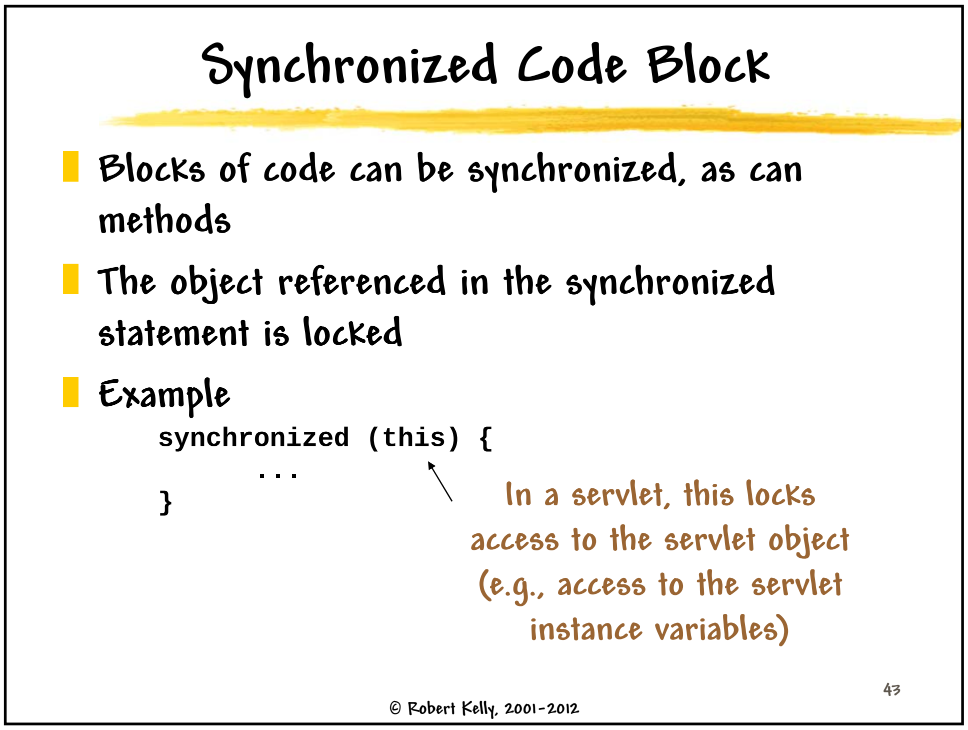
- A "synchronized" keyword in a method signature declares that access to a method is synchronized

```
public synchronized void transfer(int from,
         int to, int amount)
```

- When a thread calls a synchronized method of an object, the object becomes locked
  - it is guaranteed that the method will complete before another thread can execute any synchronized method on the same object
  - Other threads can call unsynchronized methods

© Robert Kelly, 2001-2012

42



Synchronized Code Block

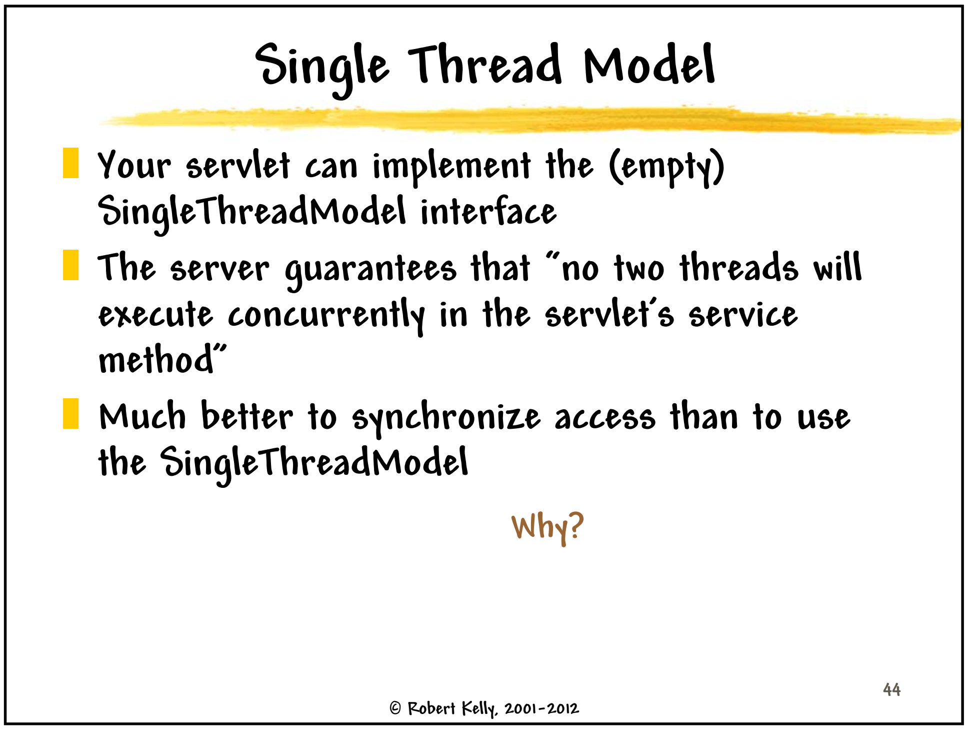
- Blocks of code can be synchronized, as can methods
- The object referenced in the synchronized statement is locked
- Example

```
synchronized (this) {
         ...
}
```

In a servlet, this locks access to the servlet object (e.g., access to the servlet instance variables)

© Robert Kelly, 2001-2012

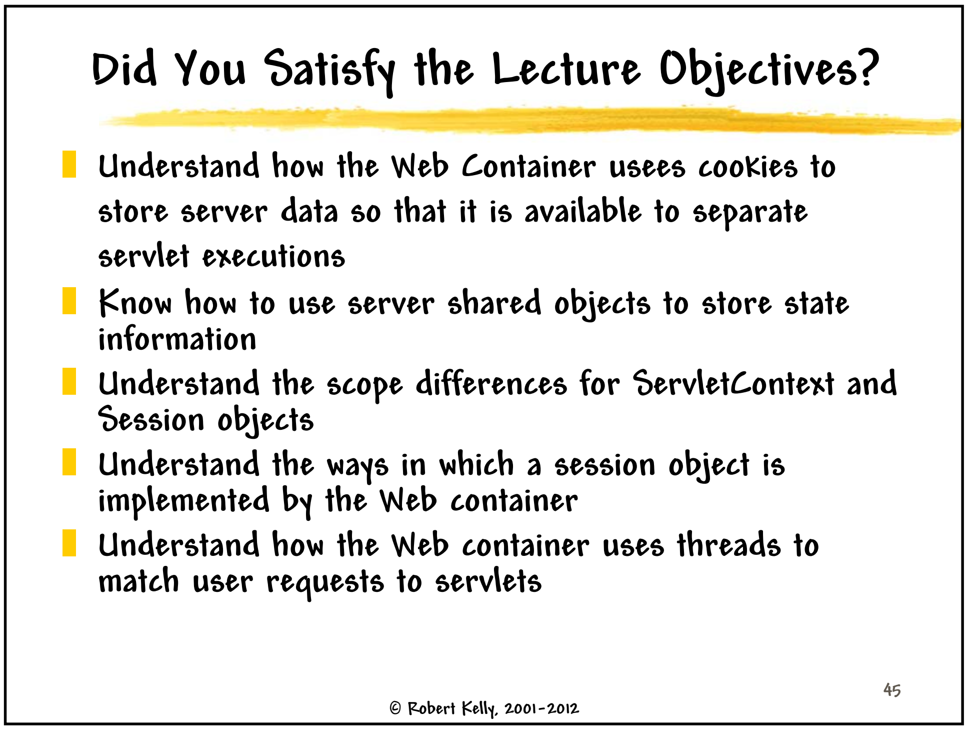
43



Single Thread Model

- Your servlet can implement the (empty) SingleThreadModel interface
- The server guarantees that "no two threads will execute concurrently in the servlet's service method"
- Much better to synchronize access than to use the SingleThreadModel

Why?



Did You Satisfy the Lecture Objectives?

- Understand how the Web Container usees cookies to store server data so that it is available to separate servlet executions
- Know how to use server shared objects to store state information
- Understand the scope differences for ServletContext and Session objects
- Understand the ways in which a session object is implemented by the Web container
- Understand how the Web container uses threads to match user requests to servlets