open.michigan

Unless otherwise noted, the content of this course material is licensed under a Creative Commons Attribution 3.0 License.
http://creativecommons.org/licenses/by/3.0/.

Copyright © 2009, Charles Severance.

You assume all responsibility for use and potential liability associated with any use of the material. Material contains copyrighted content, used in accordance with U.S. law. Copyright holders of content included in this material should contact open.michigan@umich.edu with any questions, corrections, or clarifications regarding the use of content. The Regents of the University of Michigan do not license the use of third party content posted to this site unless such a license is specifically granted in connection with particular content. Users of content are responsible for their compliance with applicable law. Mention of specific products in this material solely represents the opinion of the speaker and does not represent an endorsement by the University of Michigan. For more information about how to cite these materials visit http://michigan.educommons.net/about/terms-of-use.

Any medical information in this material is intended to inform and educate and is not a tool for self-diagnosis or a replacement for medical evaluation, advice, diagnosis or treatment by a healthcare professional. You should speak to your physician or make an appointment to be seen if you have questions or concerns about this information or your medical condition. Viewer discretion is advised: Material may contain medical images that may be disturbing to some viewers.

University of Michigan

CC BY

Cookies and Sessions

Charles Severance
www.dr-chuck.com

How stuff looks

How stuff gets back and forth...

How stuff gets made and stored

For each of these aspects of the web, we have many standards and languages and techniques to learn.

www.umich.edu

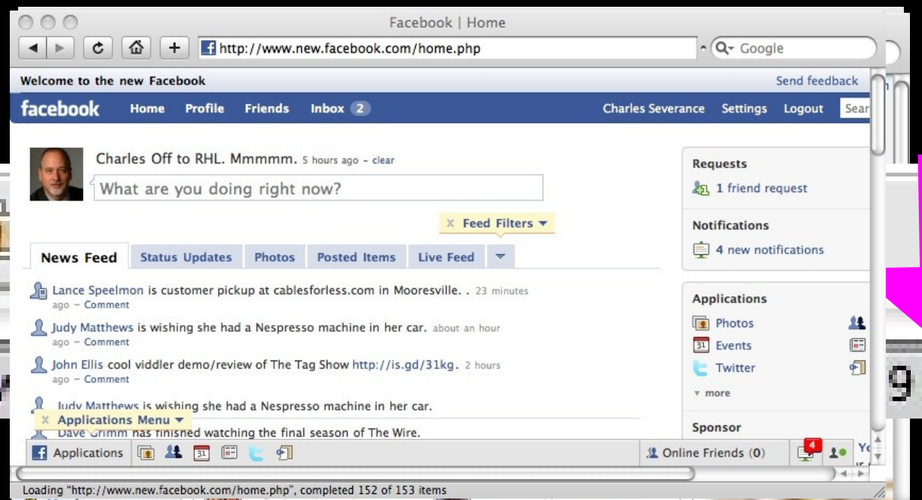
www.facebook.com

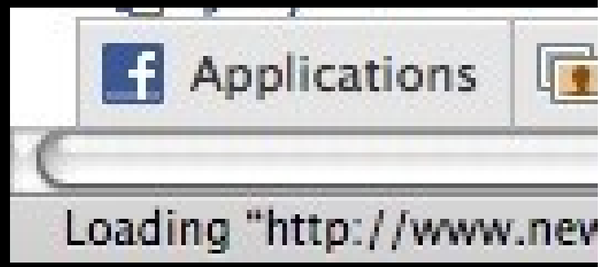
www.yahoo.com

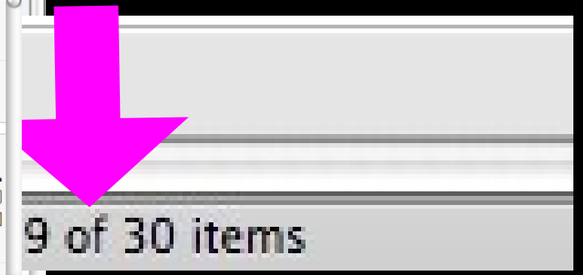
29 times

images.yahoo.com

The read-only web: hypertext navigation and lots of GETs





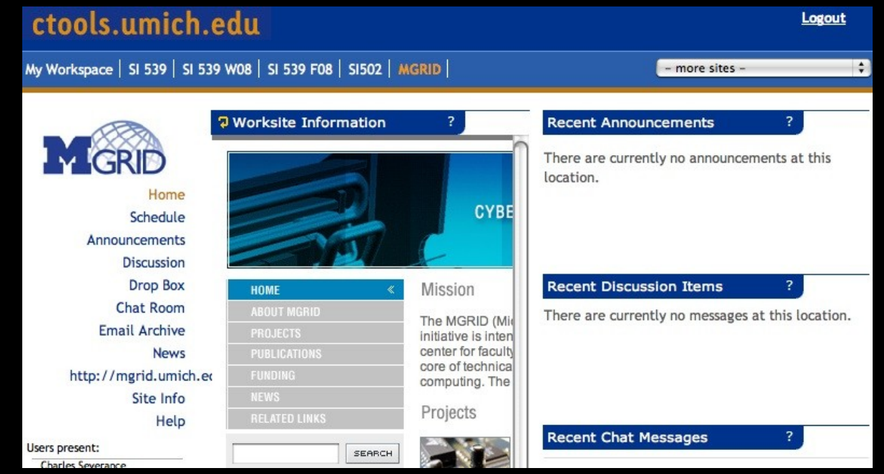


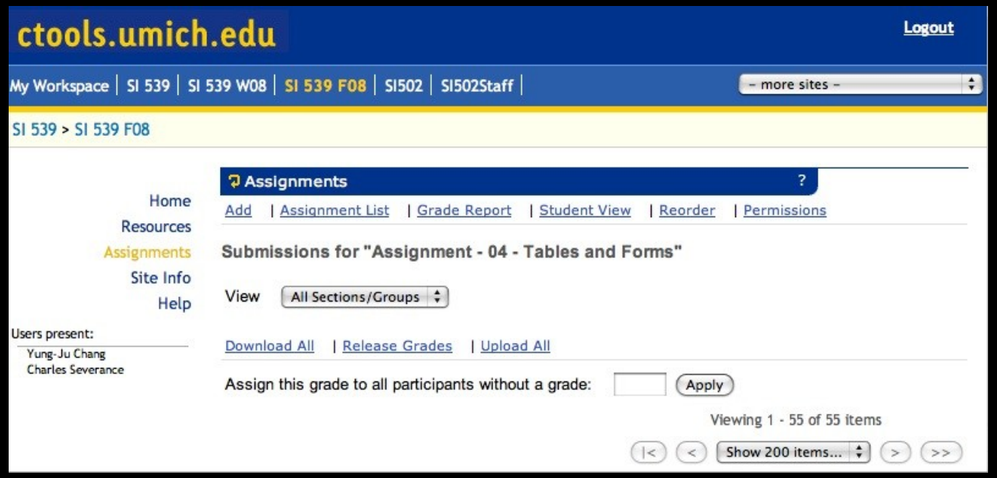
(Screenshot) Source: www.facebook.com
(Globe) source: http://www.clker.com/clipart-2123.html
(Server) source: http://www.clker.com/clipart-server.html

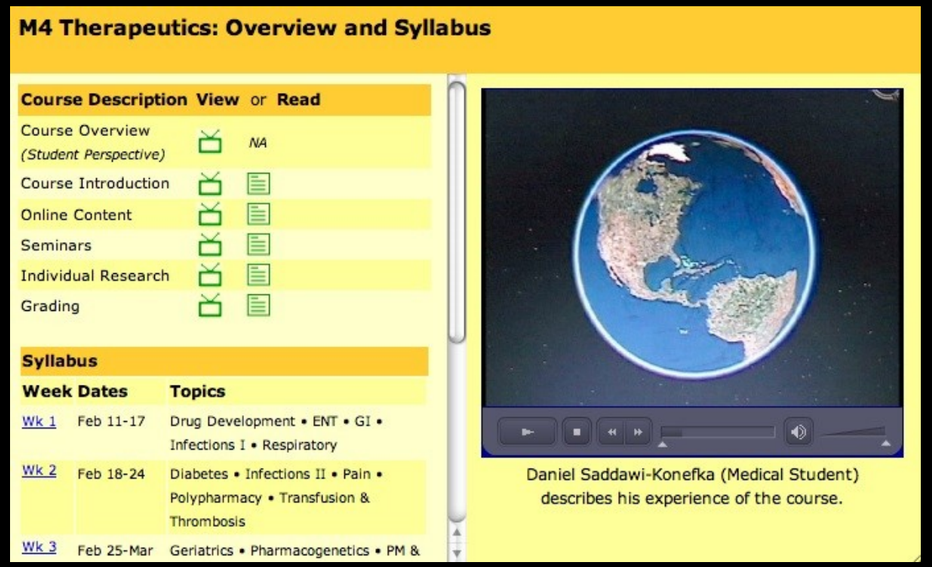
(Screenshots) Source: ctools.umich.edu
(Globe) source: http://www.clker.com/clipart-2123.html
(Server) source: http://www.clker.com/clipart-server.html

Servers get used by many users at the same time.

ctools.umich.edu





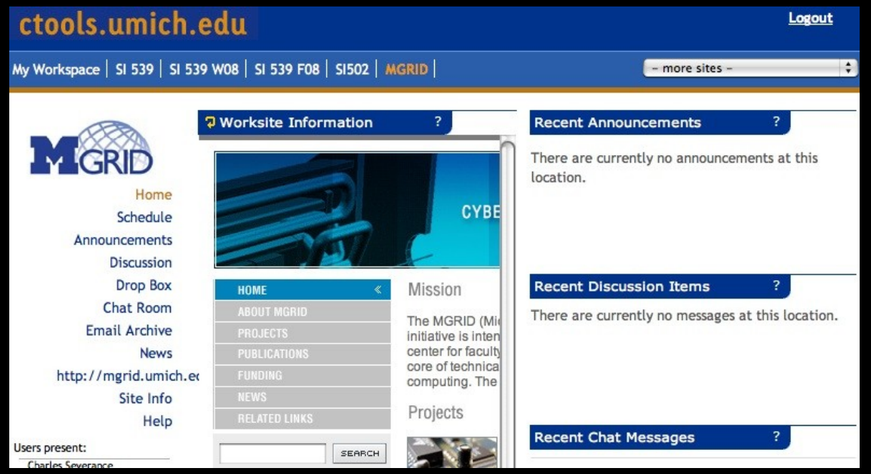


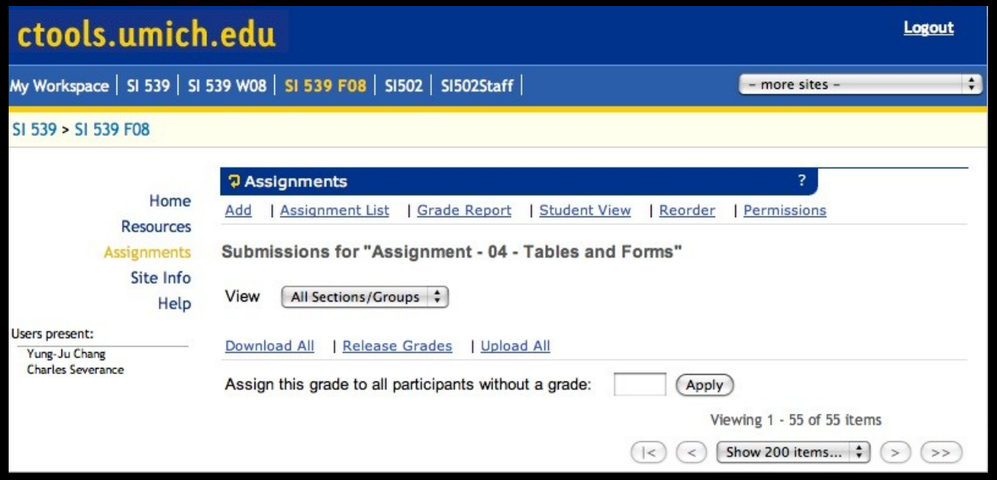
When folks hit a button...
everyone POSTs

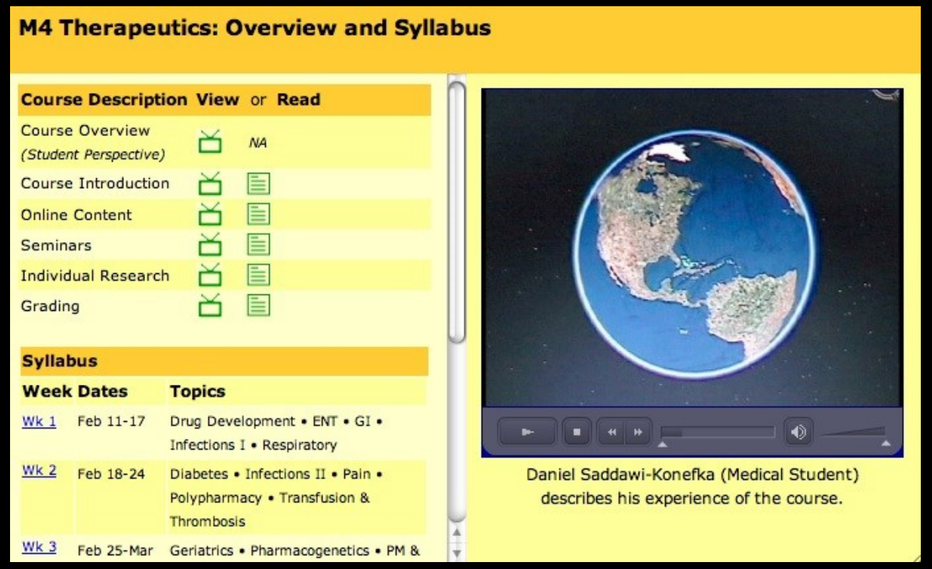
???

ctools.umich.edu

(Screenshots) Source: ctools.umich.edu
(Globe) source: http://www.clker.com/clipart-2123.html
(Server) source: http://www.clker.com/clipart-server.html







??? Server Questions ???

- Who is this user?
- Are they logged in yet?
- What screen did they come from?
- What button did they push?
- Where do we store this data?
- What screen do they want next?

??? Server Questions ???

- Who is this user?
- Are they logged in yet?
- What screen did they come from?
- What button did they push?
- Where do we store this data?
- What screen do they want next?

Over and over and over and over and over and over and over and over....

same as it ever was

Cookies and Sessions
Maintaining State in HTTP

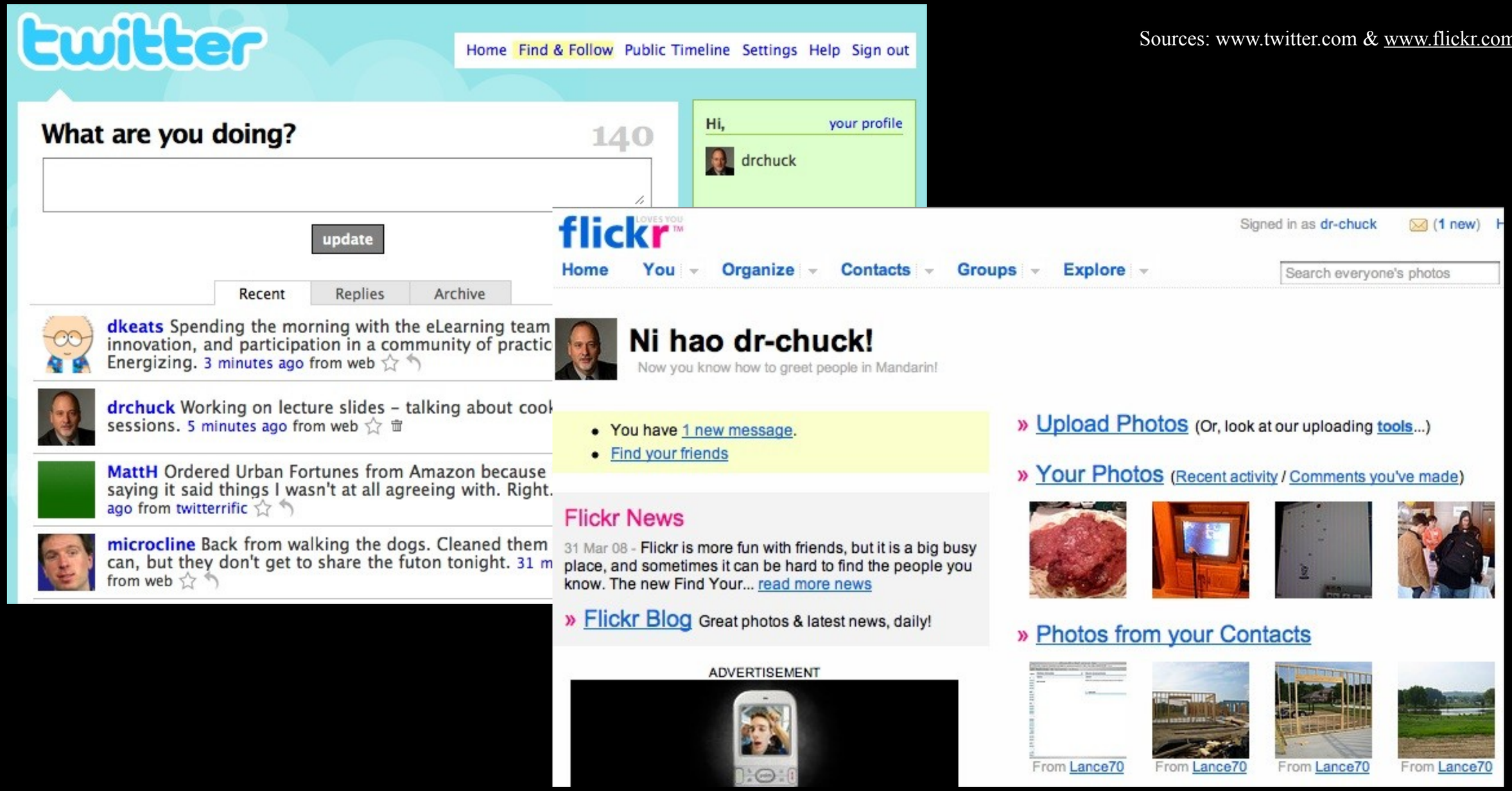
High Level Summary

- The web is “stateless” - the browser does not maintain a connection to the server while you are looking at a page. You may never come back to the same server - or it may be a long time - or it may be one second later
- So we need a way for servers to know “which browser is this?”
  - In the browser state is stored in “Cookies”
  - In the server state is stored in “Sessions”

Source: https://weblogin.umich.edu/

Some Web sites always seem to want to know who you are!

Sources: www.twitter.com & www.flickr.com



Other Web sites always seem to know who you are!

Source: http://www.youtube.com/watch?v=f90ysF9BenI

Browser

Click

Draw

You watch the YouTube video for 30 seconds

Click

Draw

Whole Page

Whole Page

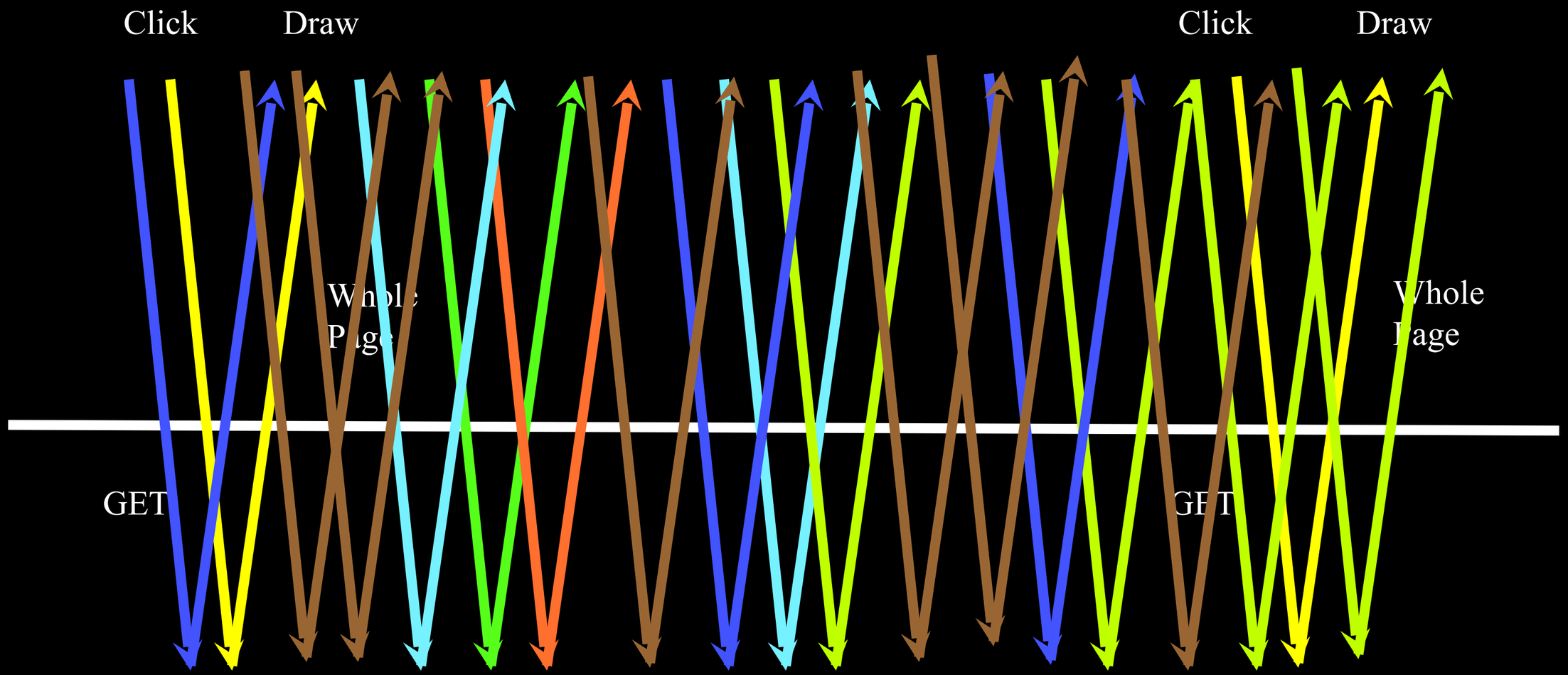
GET

GET

Server

How you see YouTube...

Browser



Server

How YouTube sees you...

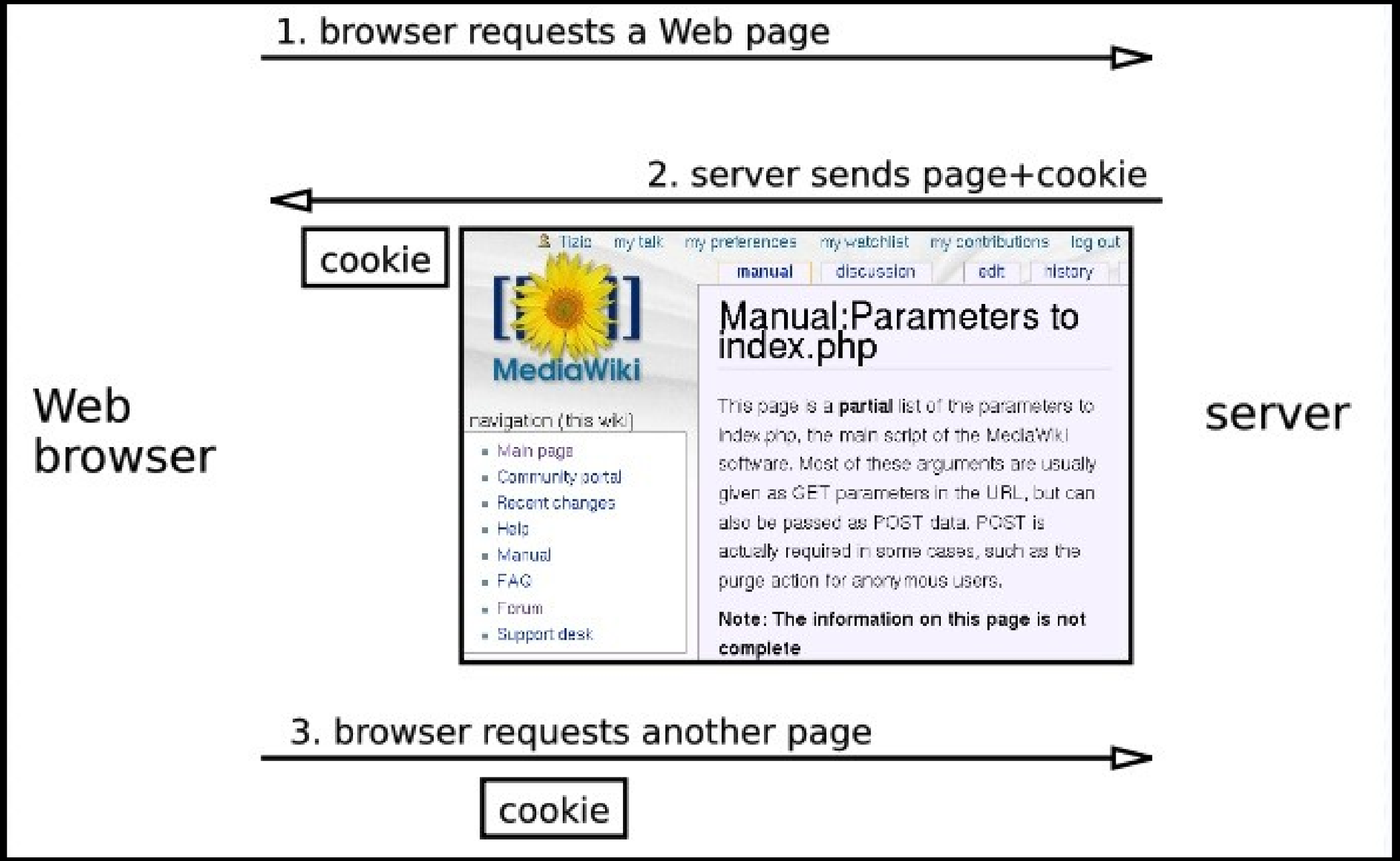
Multi-User

- When a server is interacting with many different browsers at the same time, the server needs to know *which* browser a particular request came from
- Request / Response initially was stateless - all browsers looked identical - this was really really bad and did not last very long at all.

Web Cookies to the Rescue

Technically, cookies are arbitrary pieces of data chosen by the Web server and sent to the browser. The browser returns them unchanged to the server, introducing a state (memory of previous events) into otherwise stateless HTTP transactions. Without cookies, each retrieval of a Web page or component of a Web page is an isolated event, mostly unrelated to all other views of the pages of the same site.

http://en.wikipedia.org/wiki/HTTP_cookie



http://en.wikipedia.org/wiki/HTTP_cookie

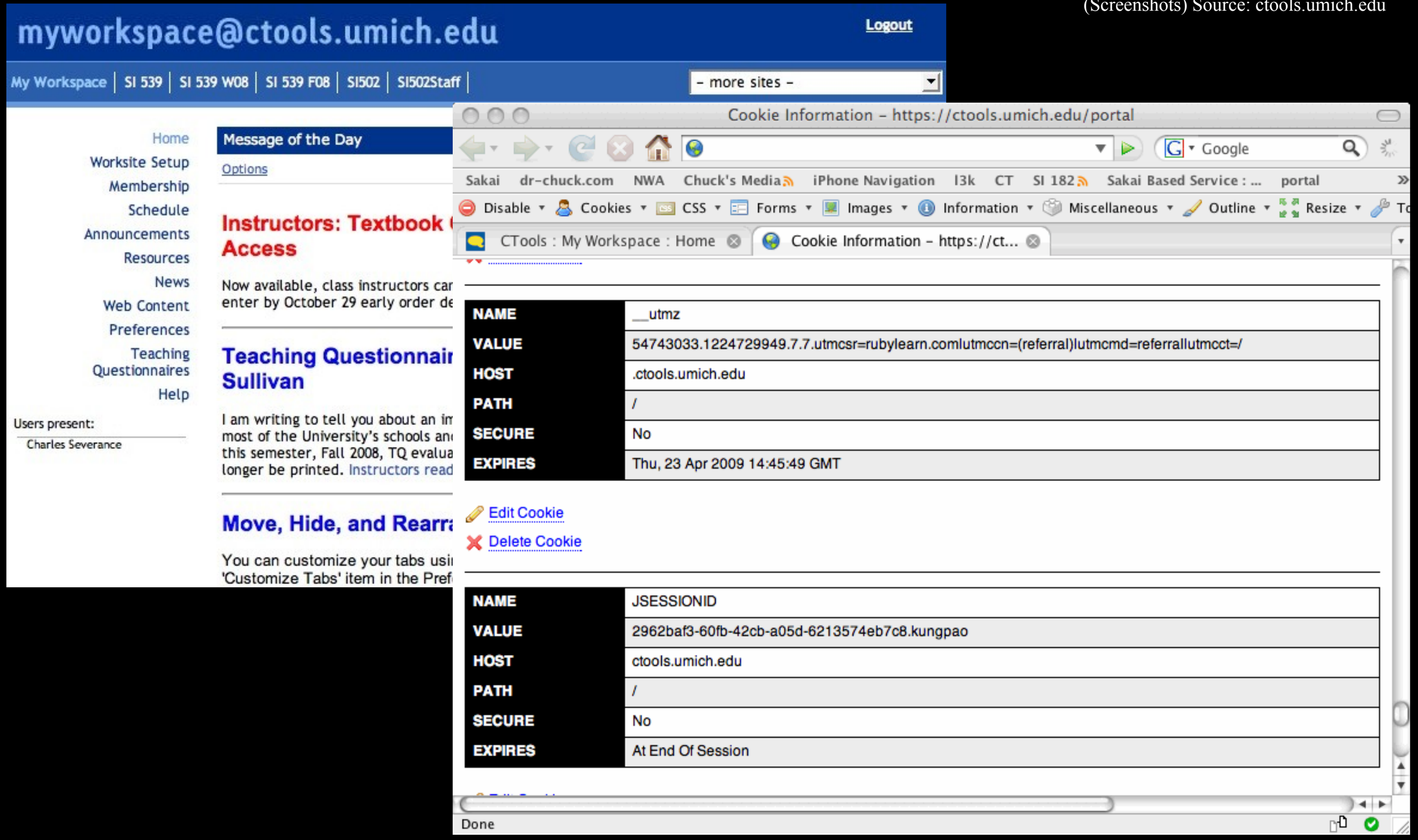
Cookies In the Browser

- Cookies are marked as to the web addresses they come from - the browser only sends back cookies that were originally set by the same web server
- Cookies have an expiration date - some last for years - others are short-term and go away as soon as the browser is closed

Playing with Cookies

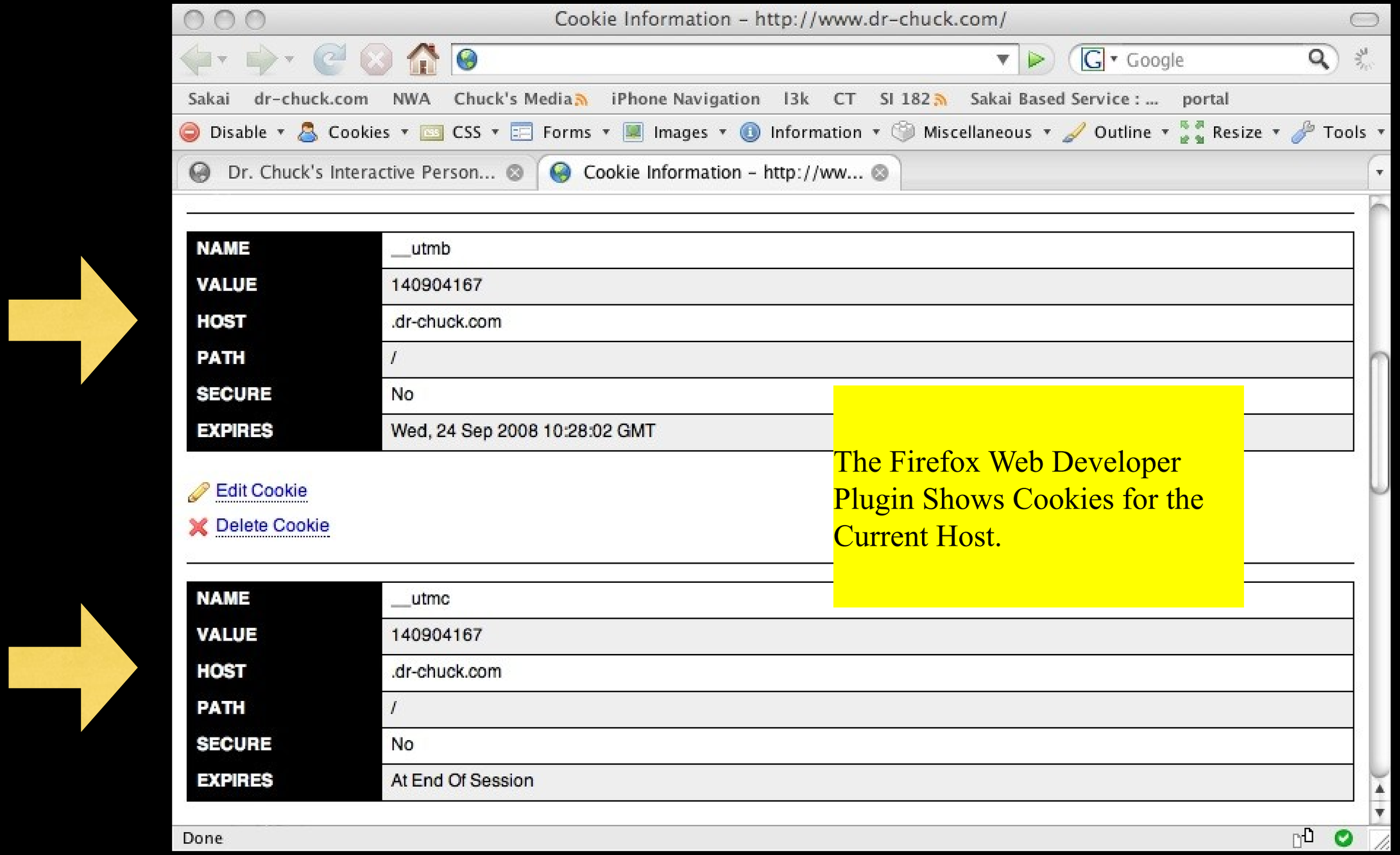
- Firefox Developer Plugin has a set of cookie features
- Other browsers have a way to view or change cookies

(Screenshots) Source: ctools.umich.edu

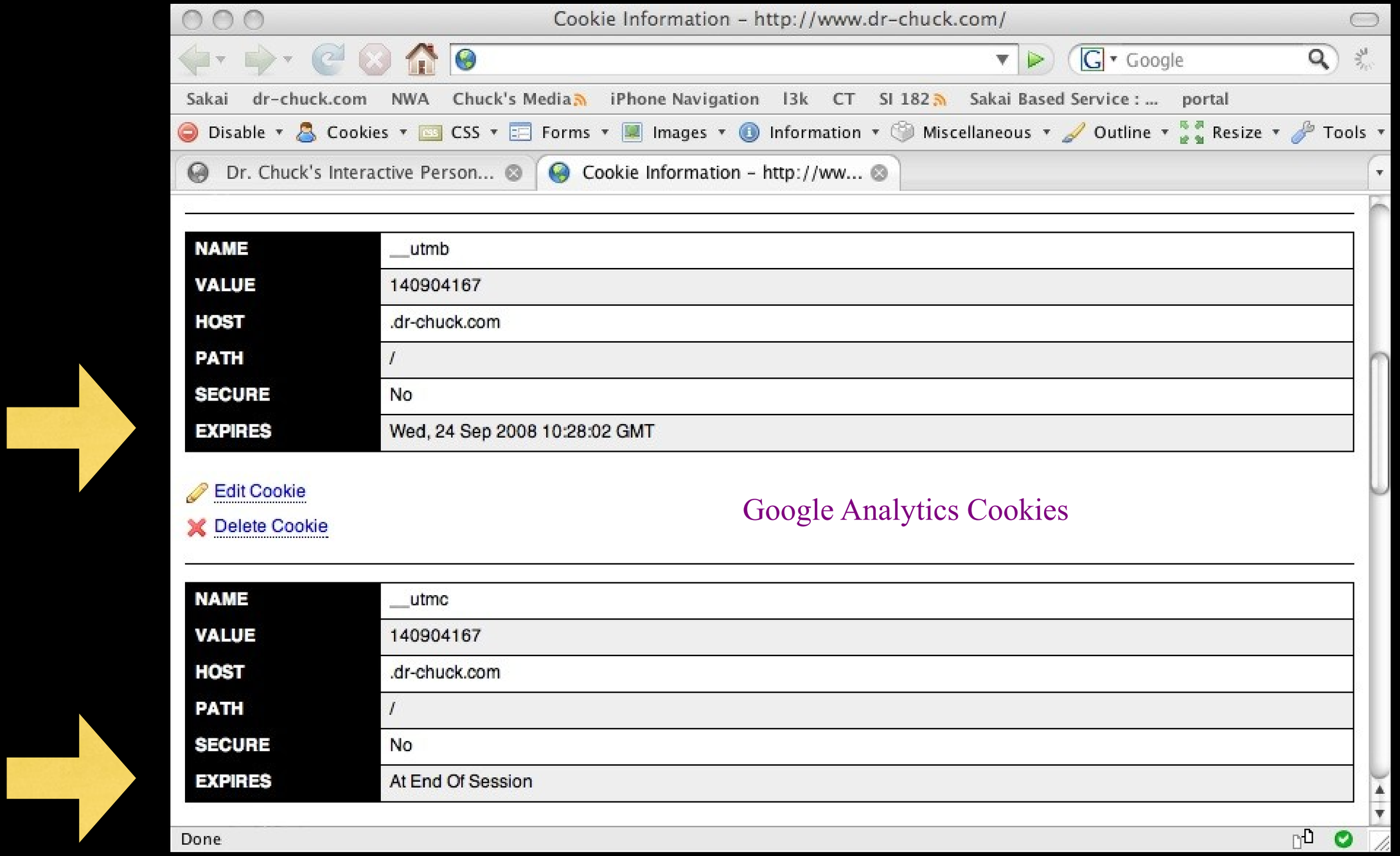


Two Kinds of Cookies

- Two kinds of cookies
  - Long-lived - who you are - account name last access time - you can close and reopen your browser and it is still there
  - Temporary - used to identify your session - it goes away when you close the browser



The Firefox Web Developer Plugin Shows Cookies for the Current Host.

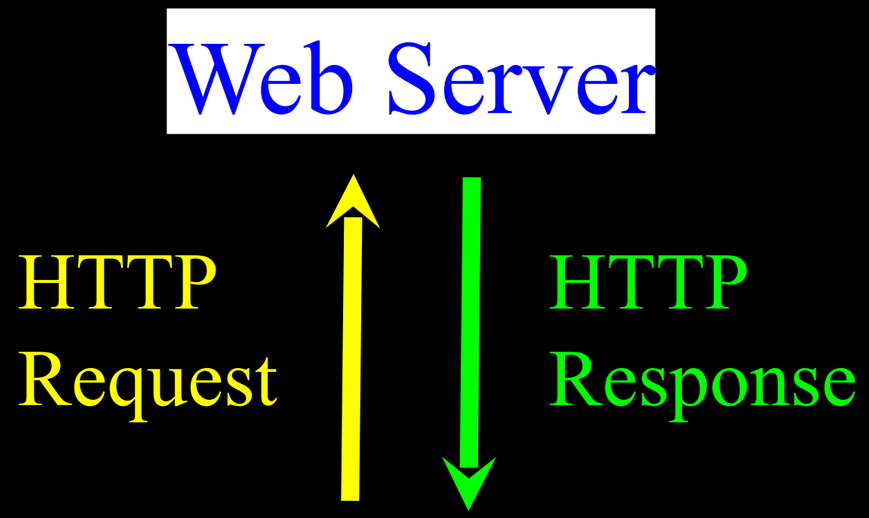

Cookie Information - http://www.dr-chuck.com/
Google
Sakai
dr-chuck.com
NWA
Chuck's Media
iPhone Navigation
l3k
CT
SI 182
Sakai Based Service : ...
portal
Disable
Cookies
CSS
Forms
Images
Information
Miscellaneous
Outline
Resize
Tools
Dr. Chuck's Interactive Person...
Cookie Information - http://ww...
NAME
__utmb
VALUE
140904167
HOST
.dr-chuck.com
PATH
/
SECURE
No
EXPIRES
Wed, 24 Sep 2008 10:28:02 GMT
Edit Cookie
Delete Cookie
Google Analytics Cookies
NAME
__utmc
VALUE
140904167
HOST
.dr-chuck.com
PATH
/
SECURE
No
EXPIRES
At End Of Session
Done

Request Response Again!
This time with cookies...

(Screenshot) Source: www.dr-chuck.com

HTTP Request / Response Cycle

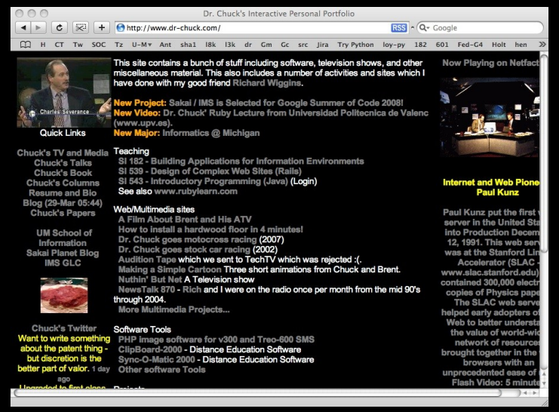
(Review)



Hello there my name is Chuck

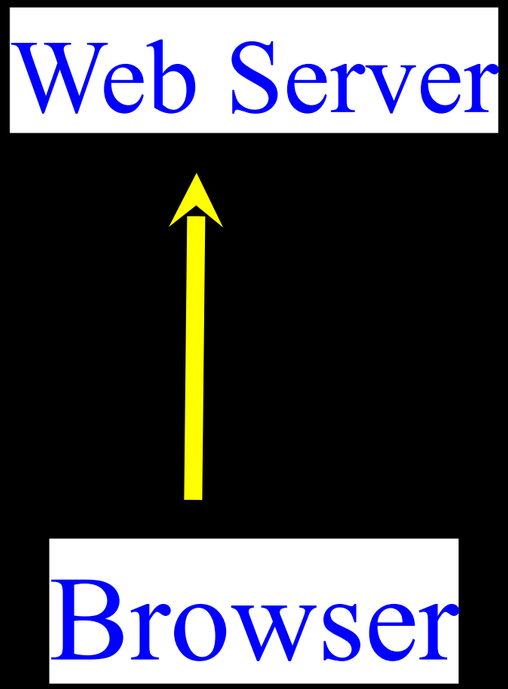
Go ahead and click on here.

Internet Explorer, FireFox, Safari, etc.



http://www.oreilly.com/openbook/cgi/ch04_02.html

HTTP Request / Response Cycle



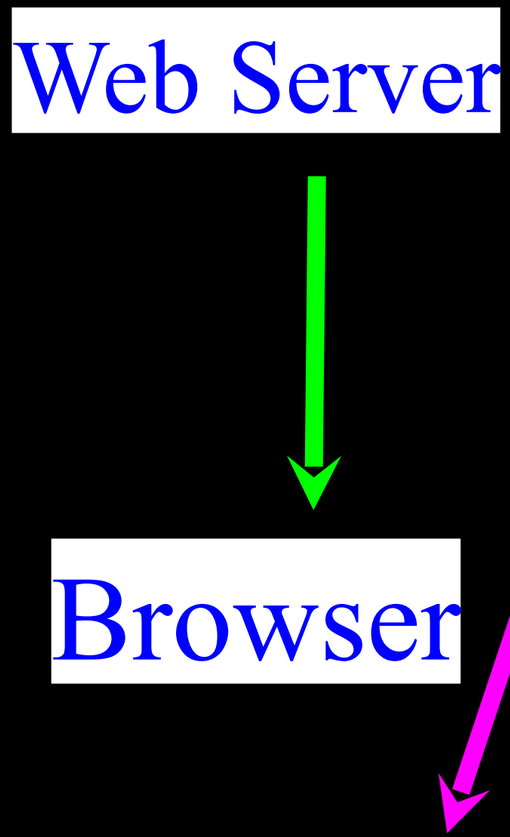
GET /index.html HTTP/1.1
Accept: www/source
Accept: text/html
User-Agent: Lynx/2.4

HTTP Request

We do or initial GET to a server. The server checks to see if we have a cookie with a particular name set. Since this our first interaction, we do not have cookies set for this host.

http://www.oreilly.com/openbook/cgi/ch04_02.html

HTTP Request / Response Cycle



HTTP/1.1 200 OK
Content-type: text/html
Set-Cookie: sessid=123

<head> .. </head>
<body>
<h1>Welcome ....

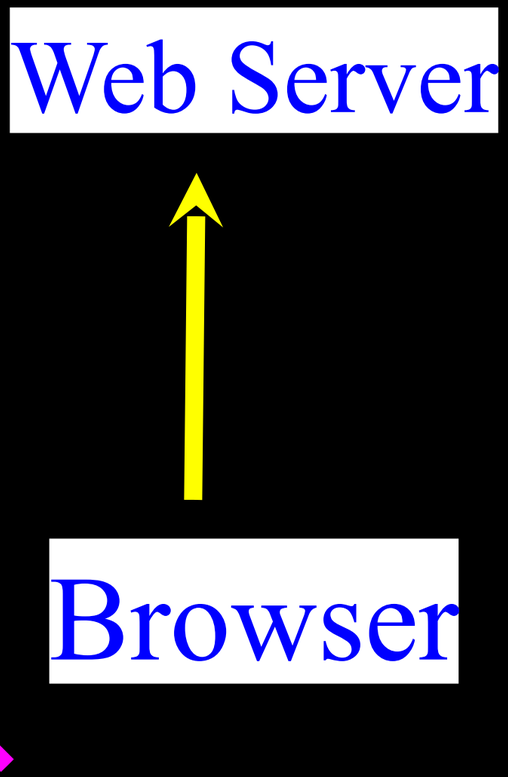
HTTP Response

Along with the rest of the response, the server sets a cookie with some name (sessid) and sends it back along with the rest of the response.

host: sessid=123

http://www.oreilly.com/openbook/cgi/ch04_02.html

HTTP Request / Response Cycle



GET /index.html HTTP/1.1
Accept: www/source
Accept: text/html
Cookie: sessid=123
User-Agent: Lynx/2.4

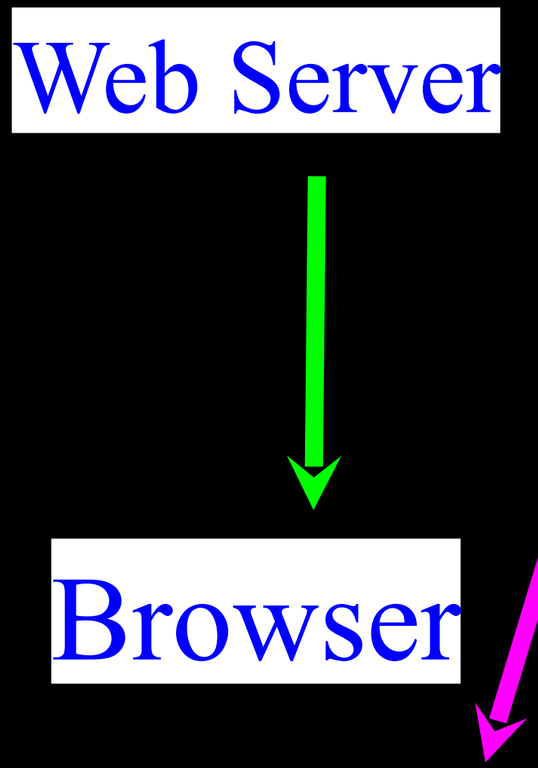
HTTP Request

host: sessid=123

From that point forward, each time we send a GET or POST to the server, we include any cookies which were set by that host.

http://www.oreilly.com/openbook/cgi/ch04_02.html

HTTP Request / Response Cycle



HTTP/1.1 200 OK
Content-type: text/html
Set-Cookie: name=chuck

<head> .. </head>
<body>
<h1>Welcome ....

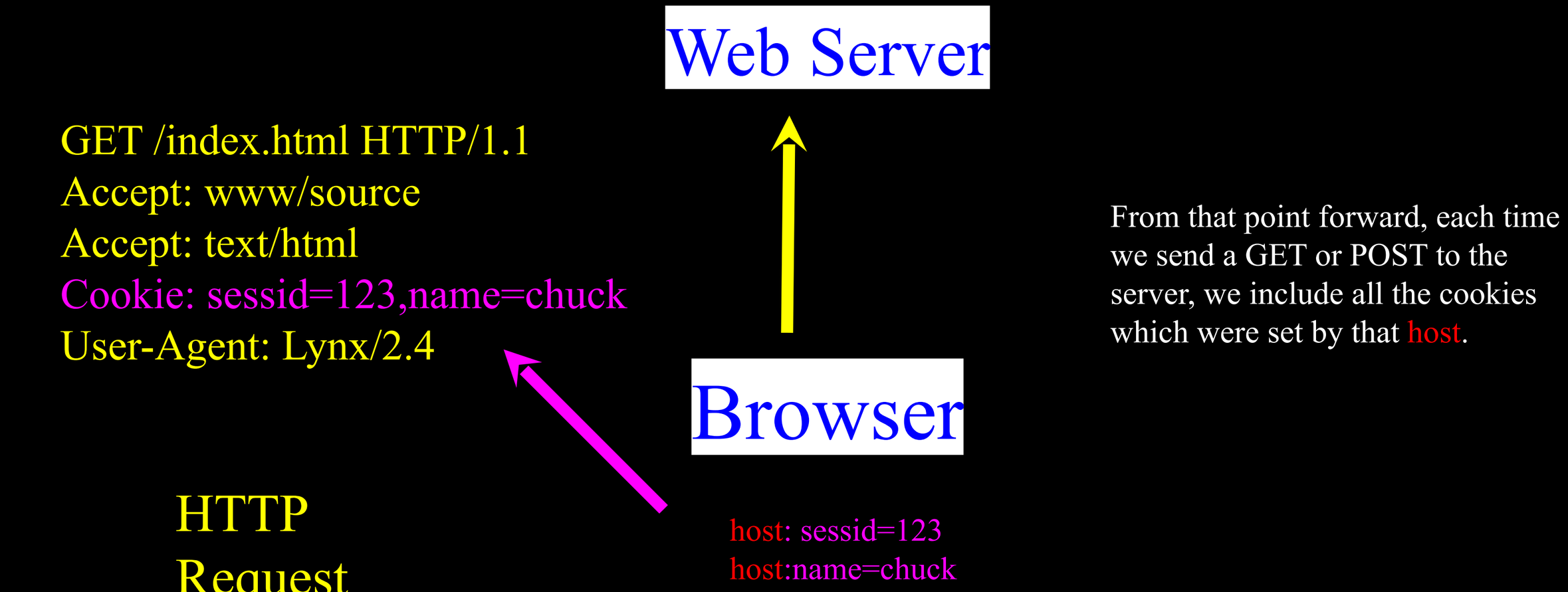
HTTP
Response

On each response, the server can change a cookie value or add another cookie.

host: sessid=123
host:name=chuck

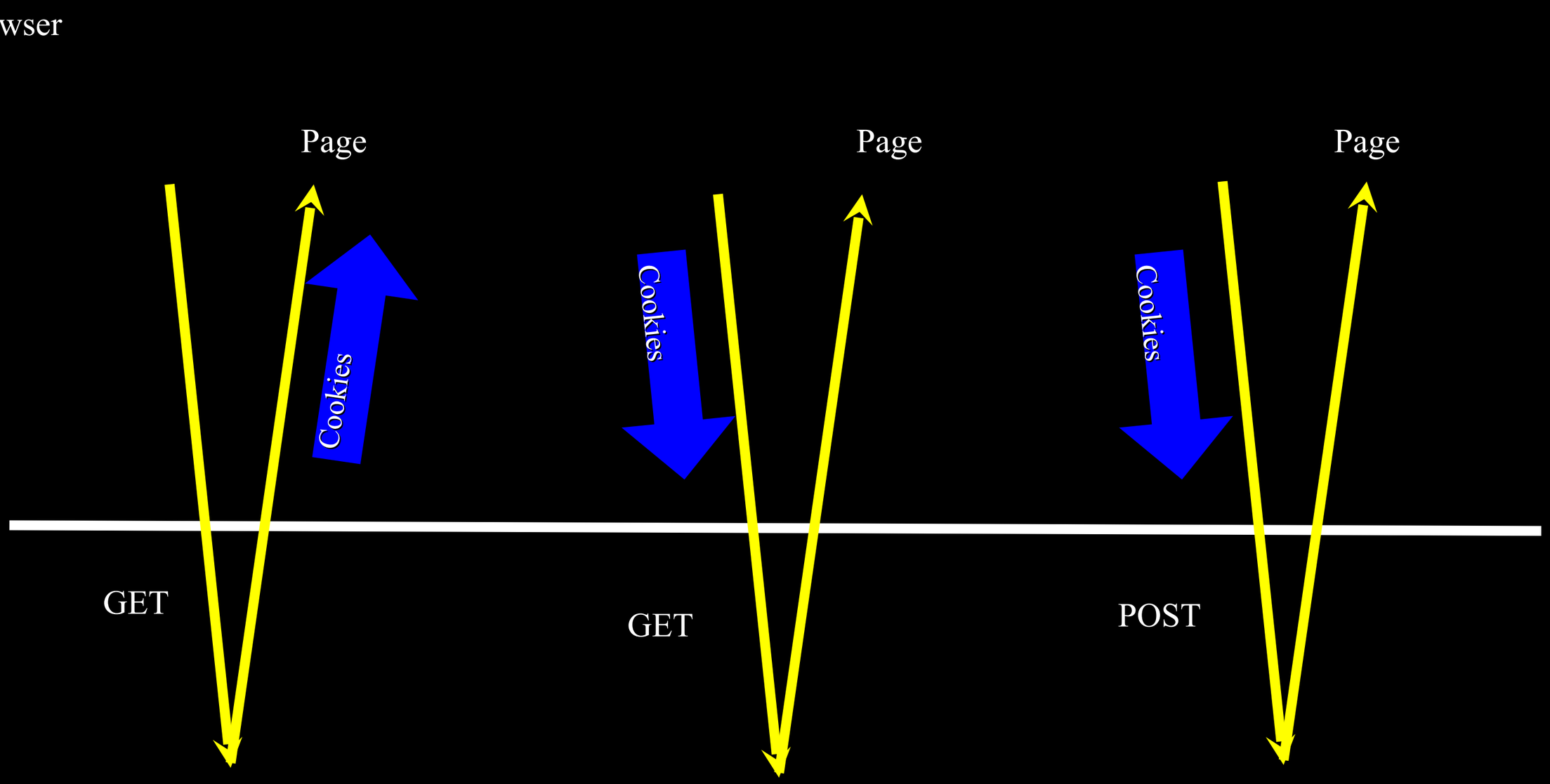
http://www.oreilly.com/openbook/cgi/ch04_02.html

HTTP Request / Response Cycle



From that point forward, each time we send a GET or POST to the server, we include all the cookies which were set by that host.

http://www.oreilly.com/openbook/cgi/ch04_02.html



Remember that cookies are only sent back to the host that set the cookie.

Security

- We ony send cookies back to the host that originally set the cookie
- The browser has *lots* of cookies for lots of hosts
- To ses all Cookies: Firefox -> Preferences -> Privacy -> Show Cookies

Using Cookies to Support Sessions and Login / Logout

Source: https://weblogin.umich.edu/

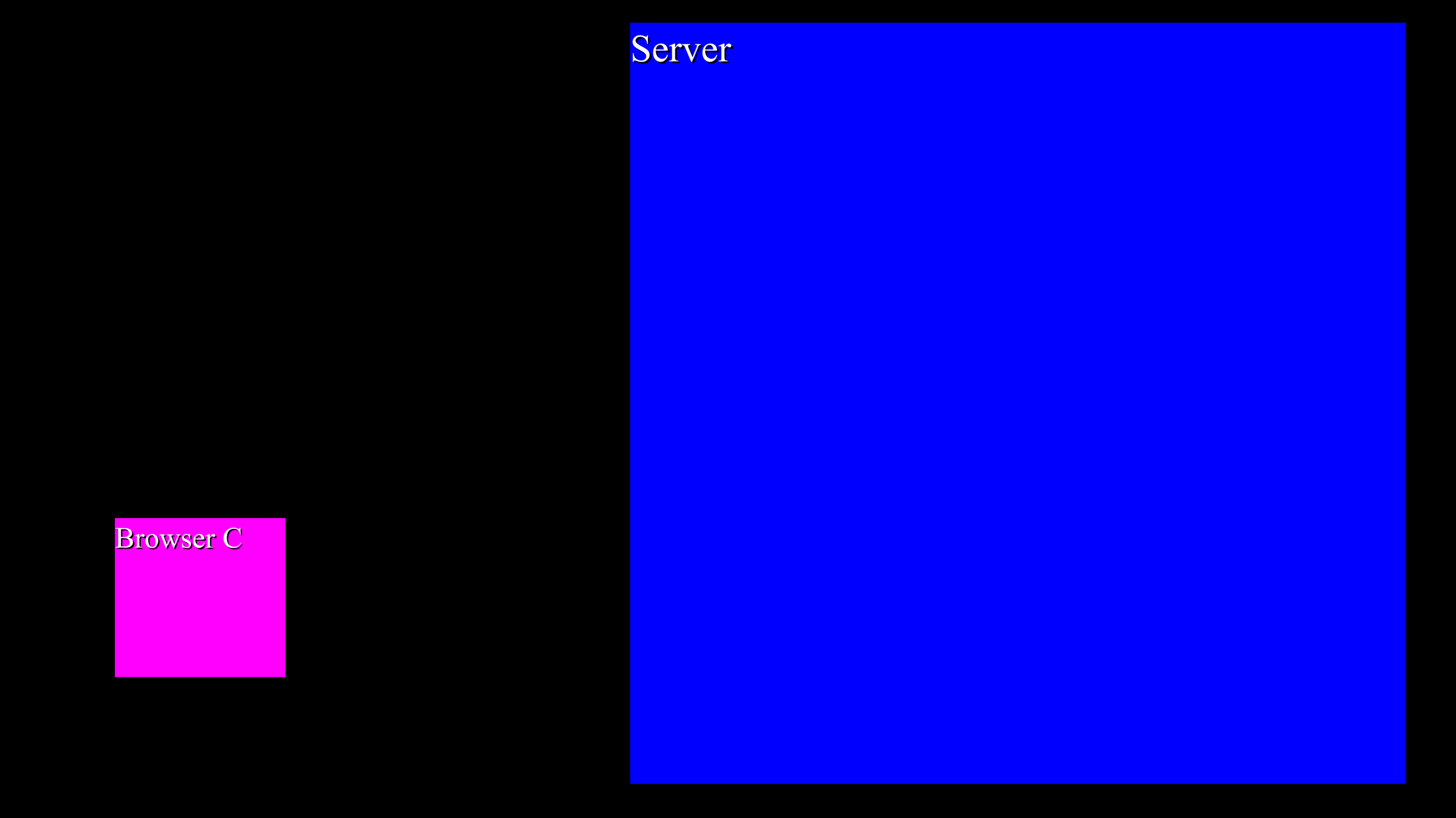
Some Web sites always seem to want to know who you are!

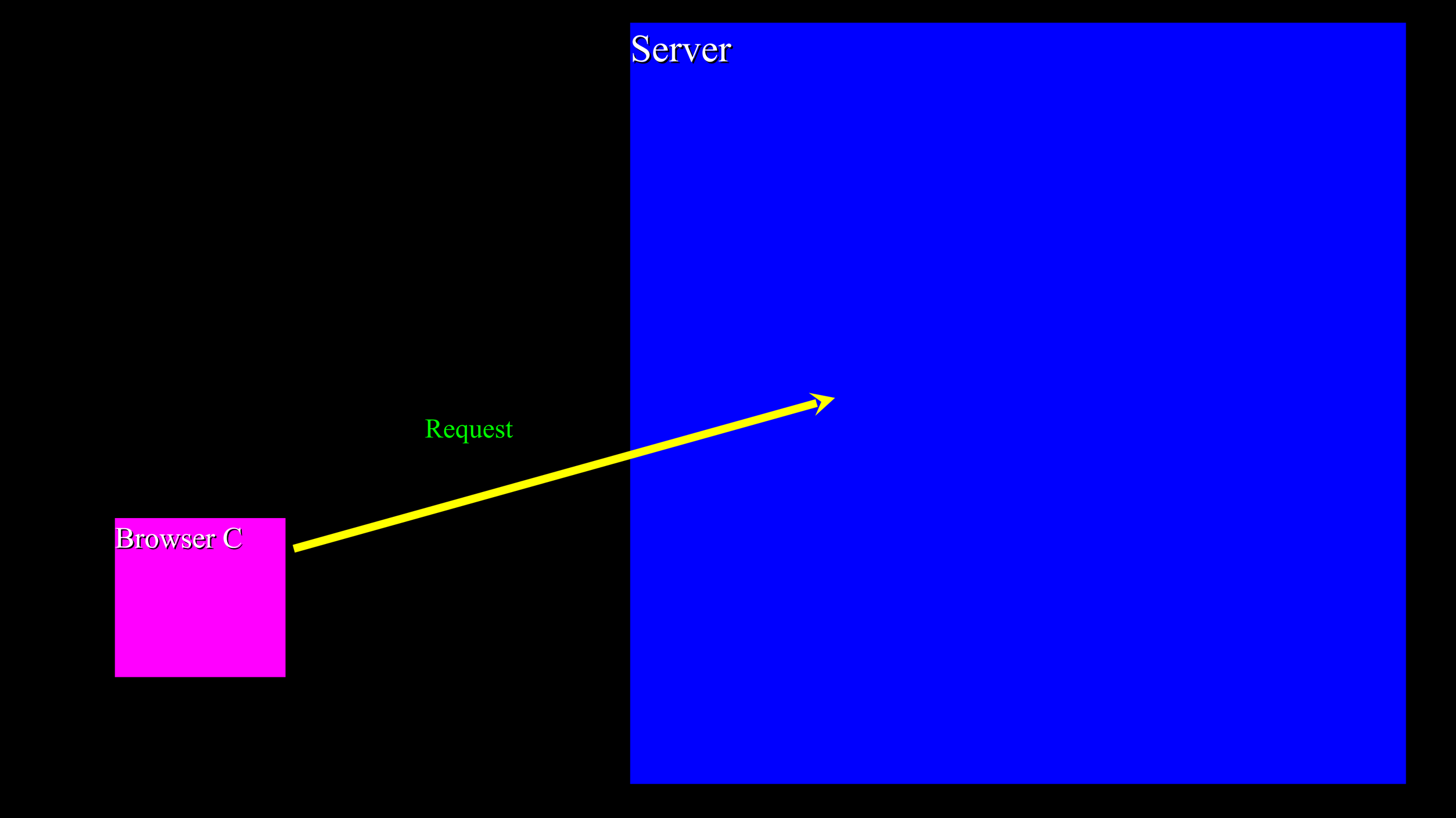
In The Server - Sessions

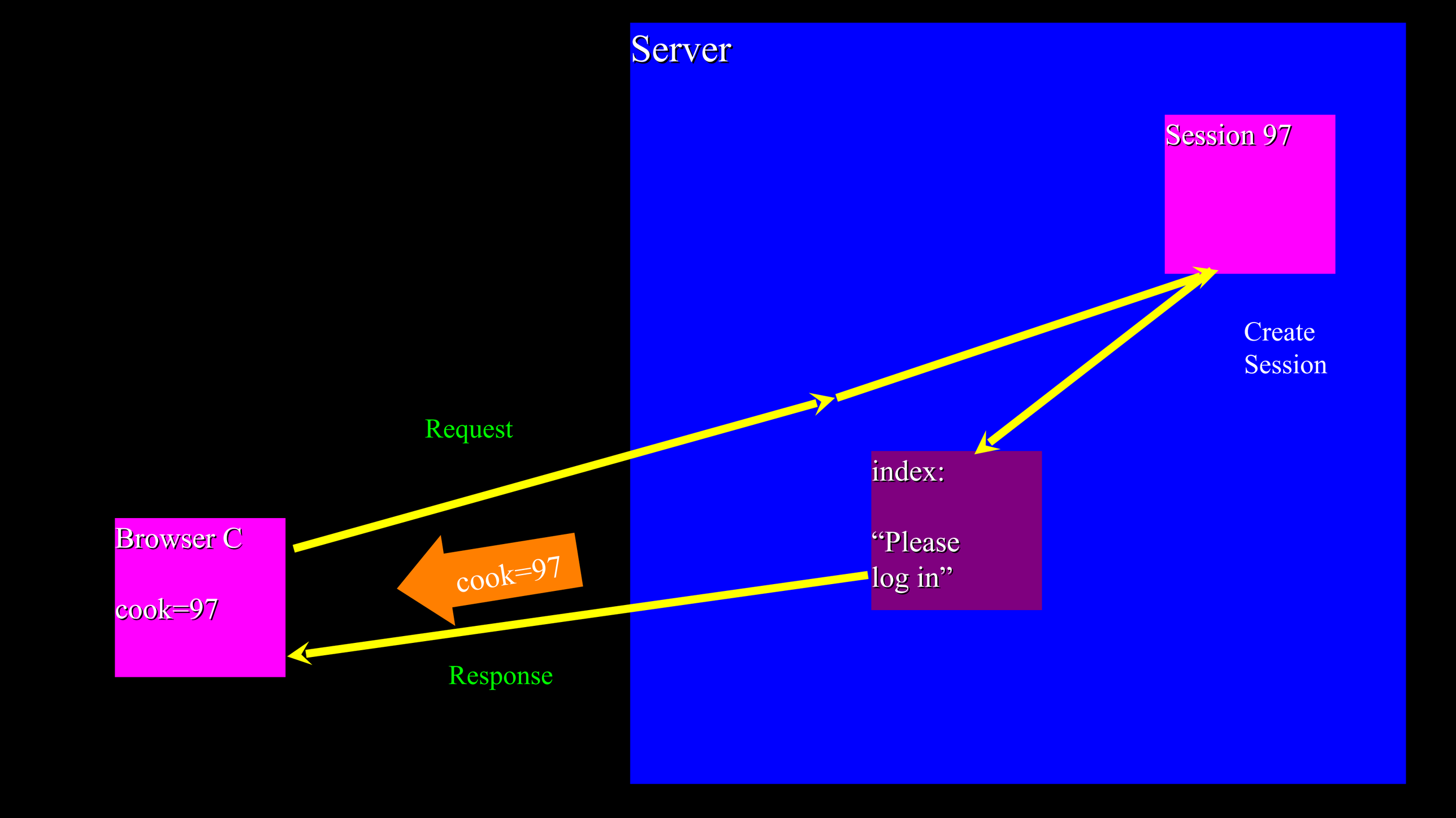
- In most server applications, as soon as we meet a new browser - we create a session
- We set a session cookie to be stored in the browser which indicates the session id in use
- The creation and destruction of sessions is generally handled by a web framework or some utility code that we just use to manage the sessions

Session Identifier

- A large, random number that we place in a browser cookie the first time we encounter a browser.
- This number is used to pick from the many sessions that the server has active at any one time.
- Server software stores data in the session which it wants to have from one request to another from the same browser.
  - Shopping cart or login information is stored in the session in the server


Server
Browser C


Server
Request
Browser C


Server
Session 97
Create
Session
Request
index:
“Please
log in”
Browser C
cook=97
cook=97
Response

Source: https://weblogin.umich.edu/

Typing

Browser C

cook=97

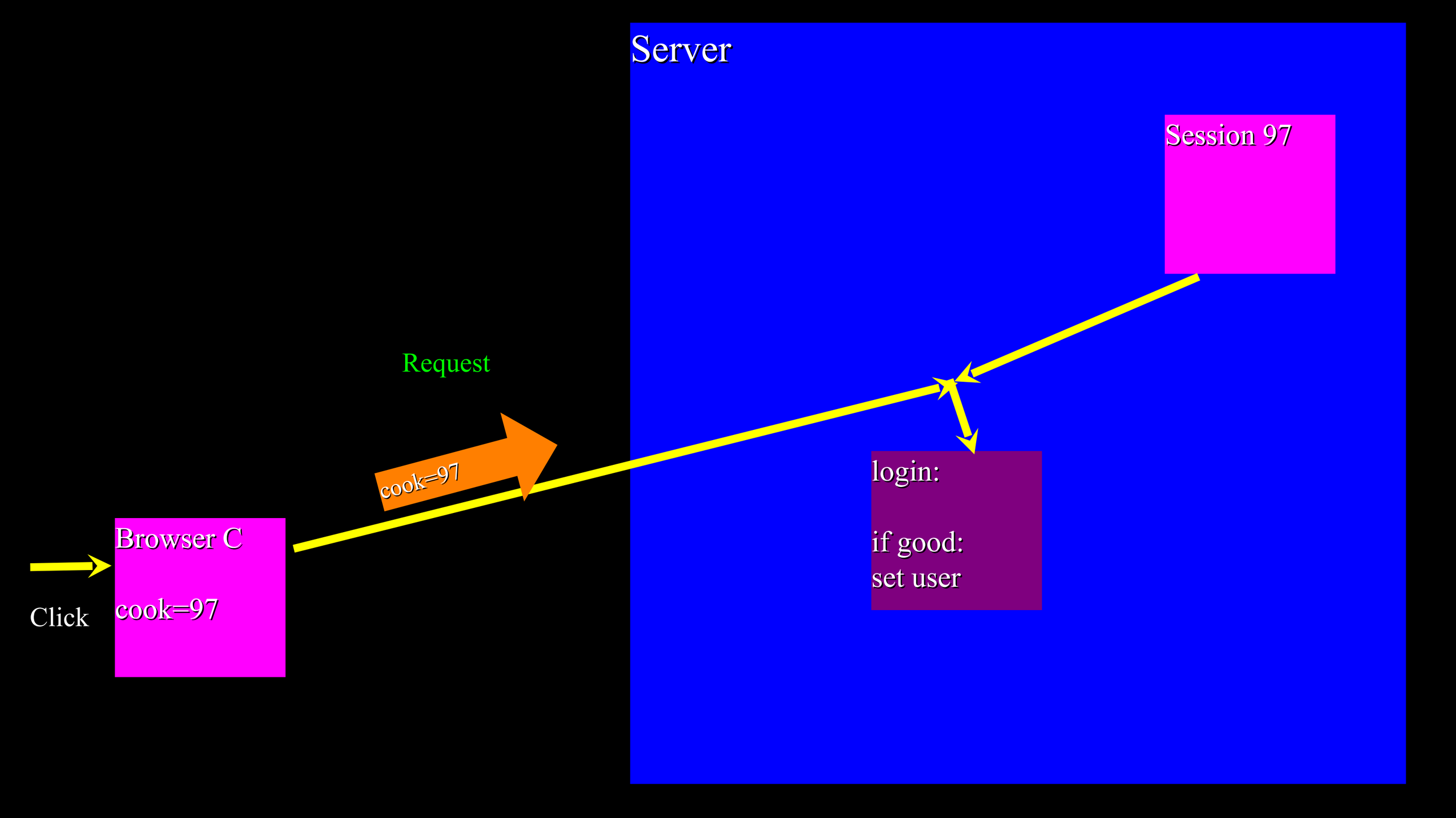
Server

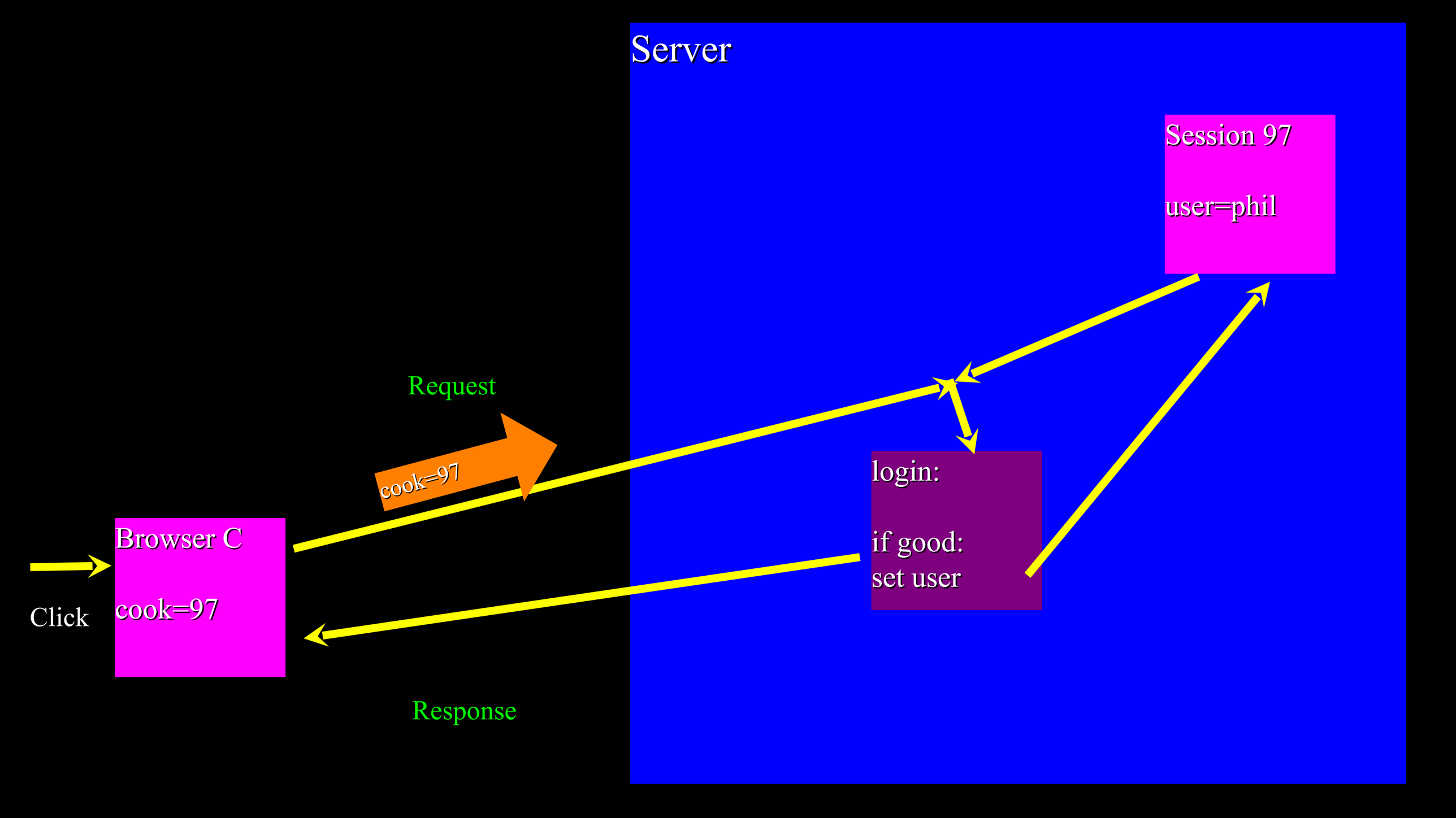
Session 97

We now have a session established
but are not yet logged in.

Login / Logout

- Having a session is not the same as being logged in.
- Generally you have a session the instant you connect to a web site
- The Session ID cookie is set when the first page is delivered
- Login puts user information in the session (stored in the server)
- Logout removes user information from the session


Server
Session 97
Request
cook=97
login:
if good:
set user
Browser C
cook=97
Click


Server
Session 97
user=phil
Request
cook=97
login:
if good:
set user
Browser C
cook=97
Click
Response

Server

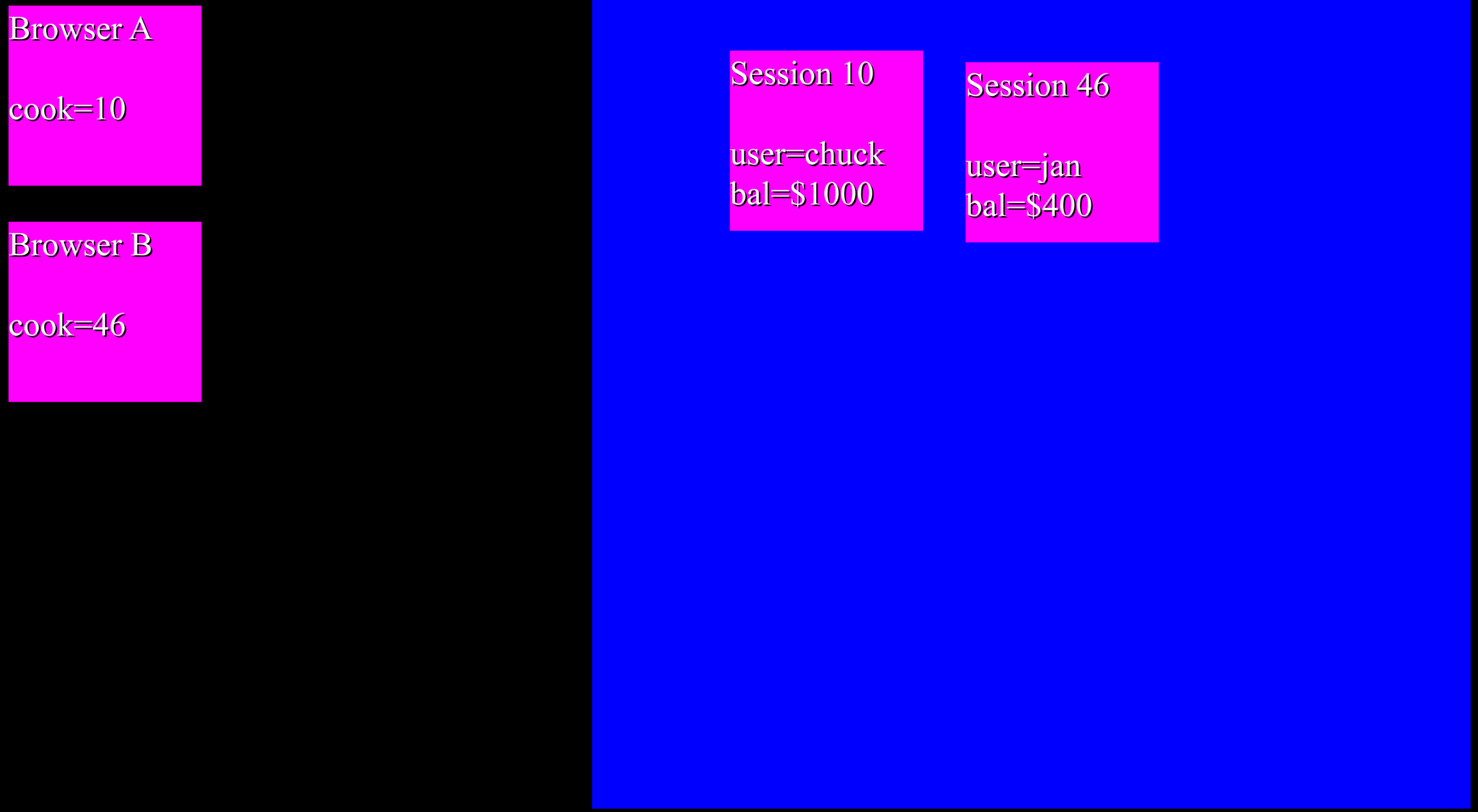
Session 97

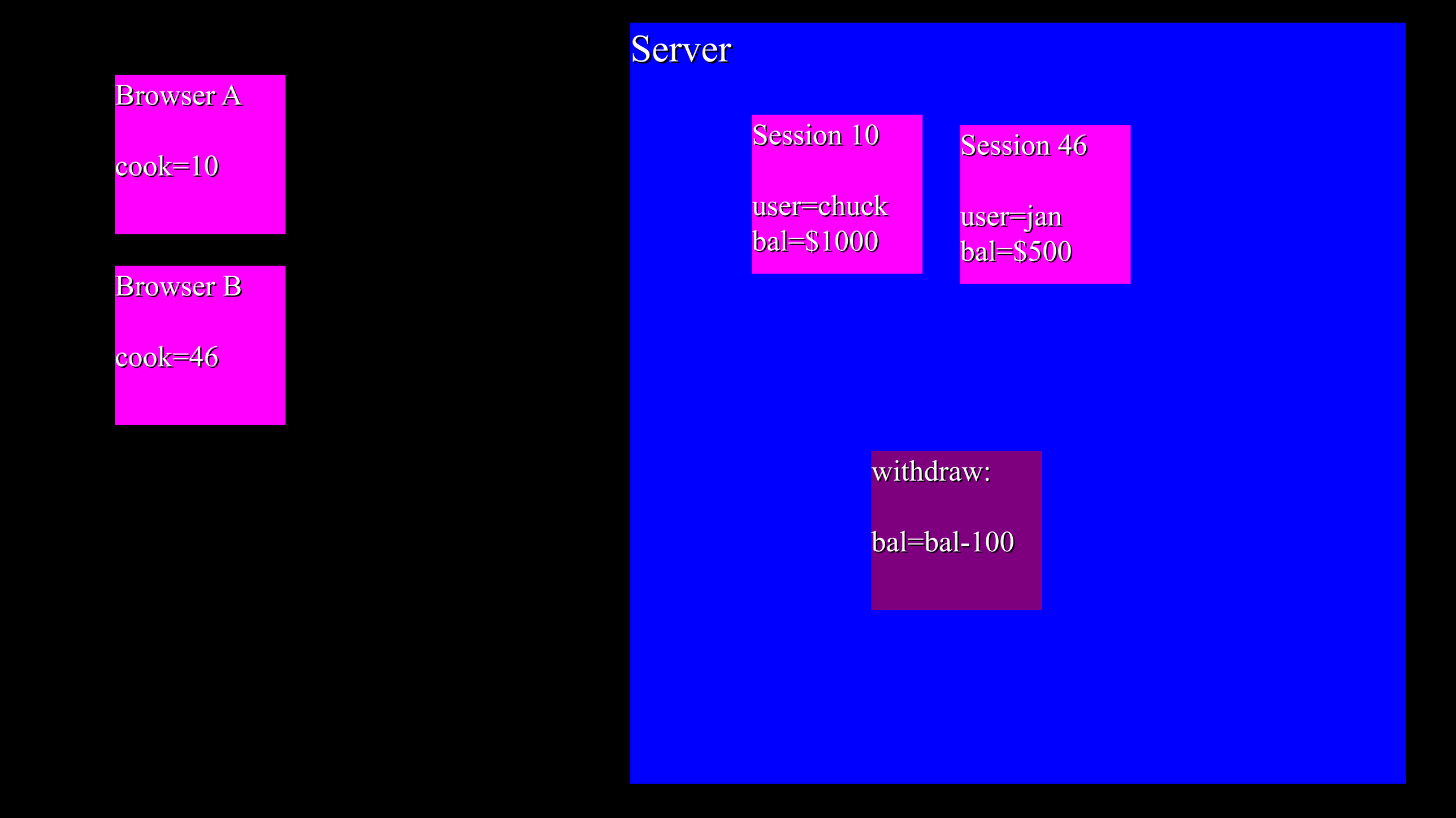
user=phil

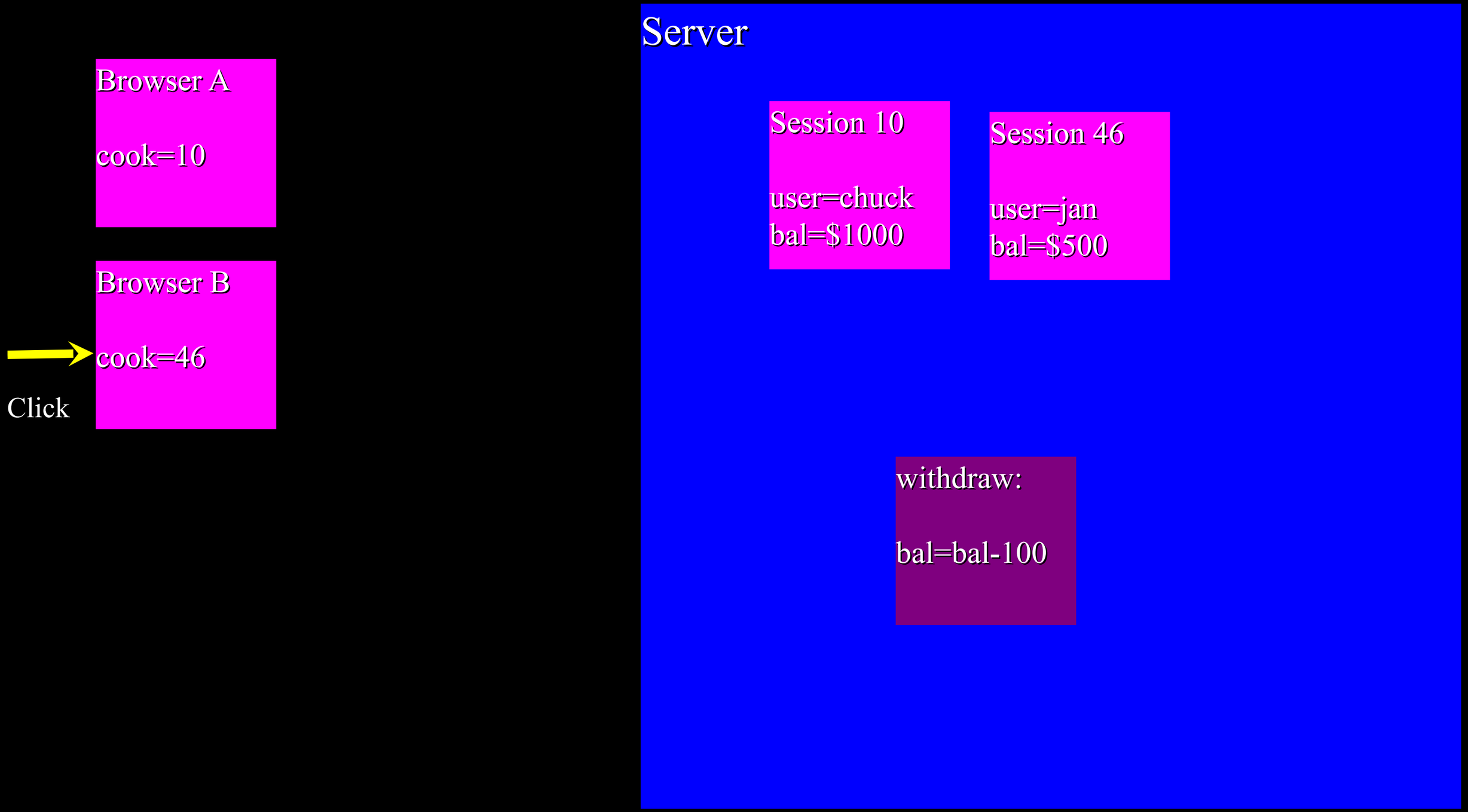
Browser C

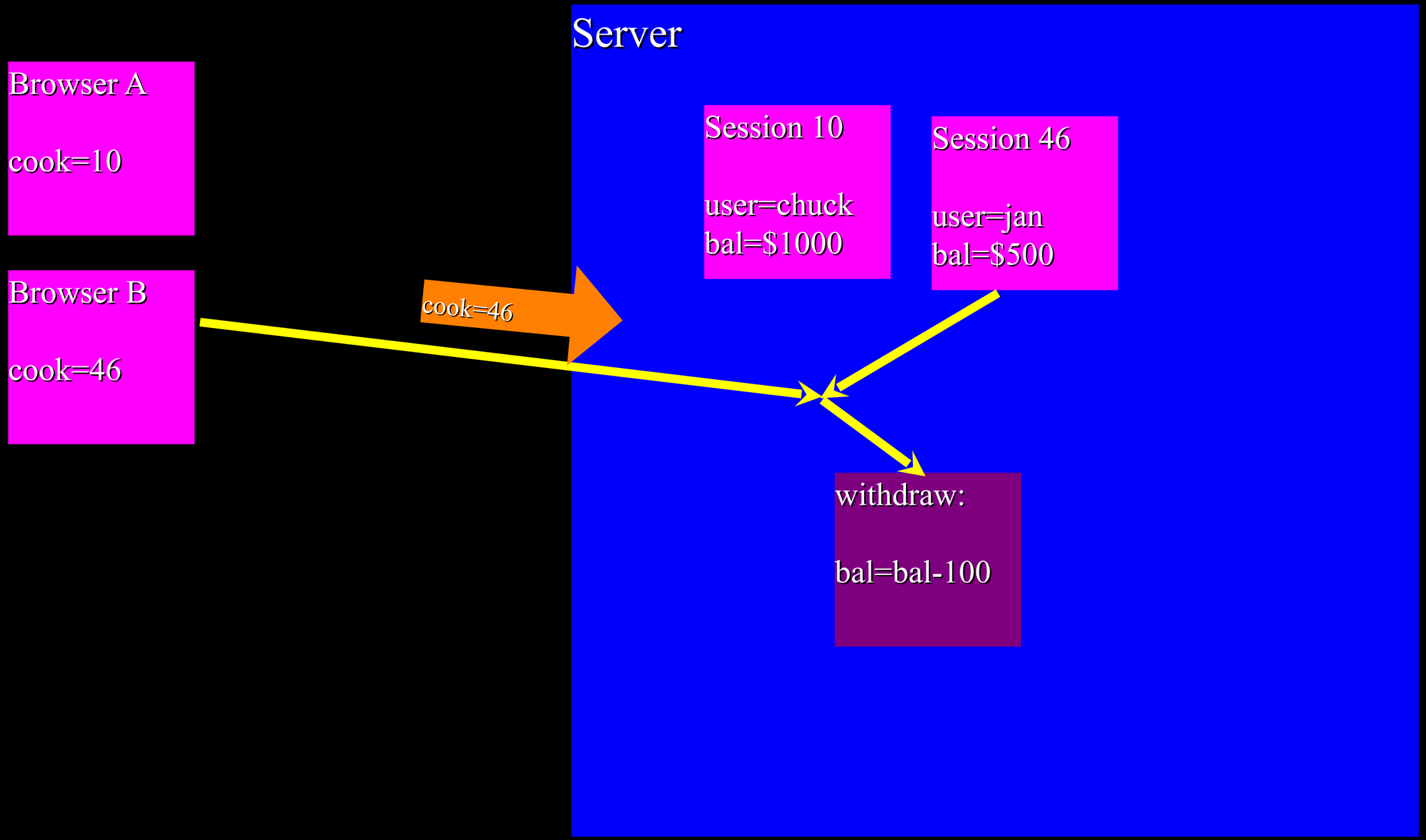
cook=97

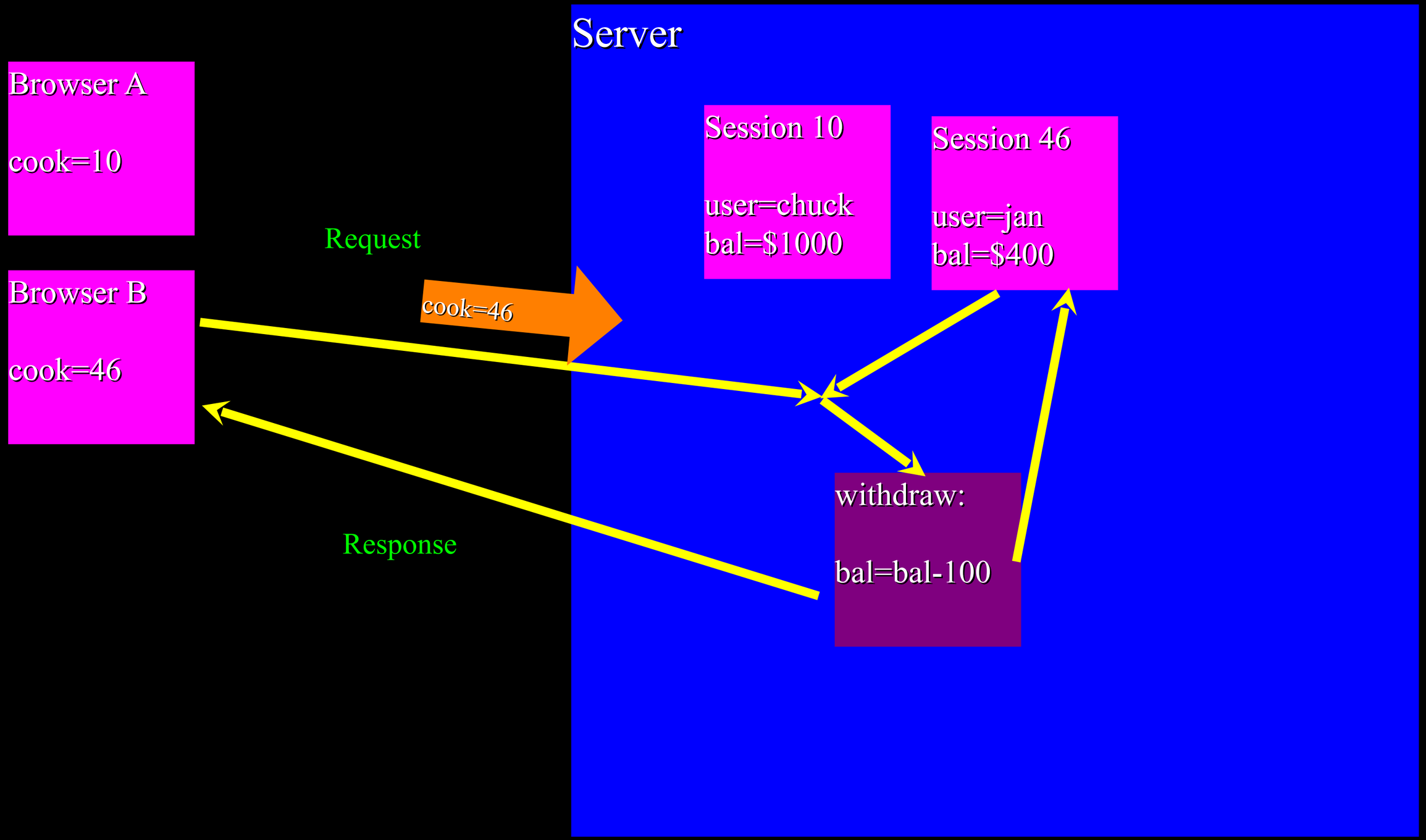
Using Sessions for Other Stuff


Browser A
cook=10
Browser B
cook=46
Server
Session 10
user=chuck
bal=$1000
Session 46
user=jan
bal=$400


Server
Browser A
cook=10
Browser B
cook=46
Session 10
user=chuck
bal=$1000
Session 46
user=jan
bal=$500
withdraw:
bal=bal-100


Server
Browser A
cook=10
Browser B
cook=46
Click
Session 10
user=chuck
bal=$1000
Session 46
user=jan
bal=$500
withdraw:
bal=bal-100


Server
Browser A
cook=10
Browser B
cook=46
cook=46
Session 10
user=chuck
bal=$1000
Session 46
user=jan
bal=$500
withdraw:
bal=bal-100


Server
Browser A
cook=10
Browser B
cook=46
Request
cook=46
Session 10
user=chuck
bal=$1000
Session 46
user=jan
bal=$400
withdraw:
bal=bal-100
Response

Review...

High Level Summary

- The web is "stateless" - the browser does not maintain a connection to the server while you are looking at a page. You may never come back to the same server - or it may be a long time - or it may be one second later
- So we need a way for servers to know "which browser is this?"
  - In the browser state is stored in "Cookies"
  - In the server state is stored in "Sessions"

Source: http://www.youtube.com/watch?v=f90ysF9BenI

Browser

Click

Draw

You watch the YouTube video for an 30 seconds

Click

Draw

Whole Page

Whole Page

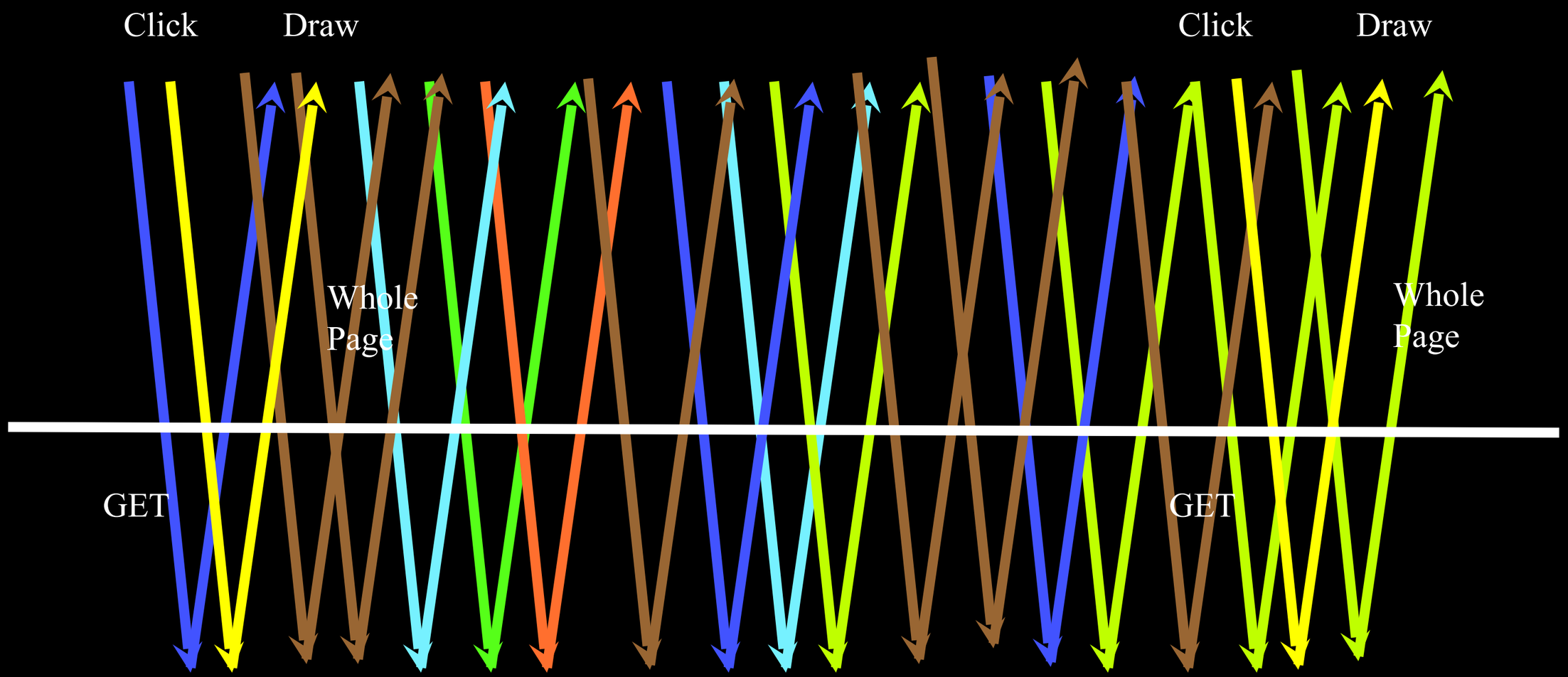
GET

GET

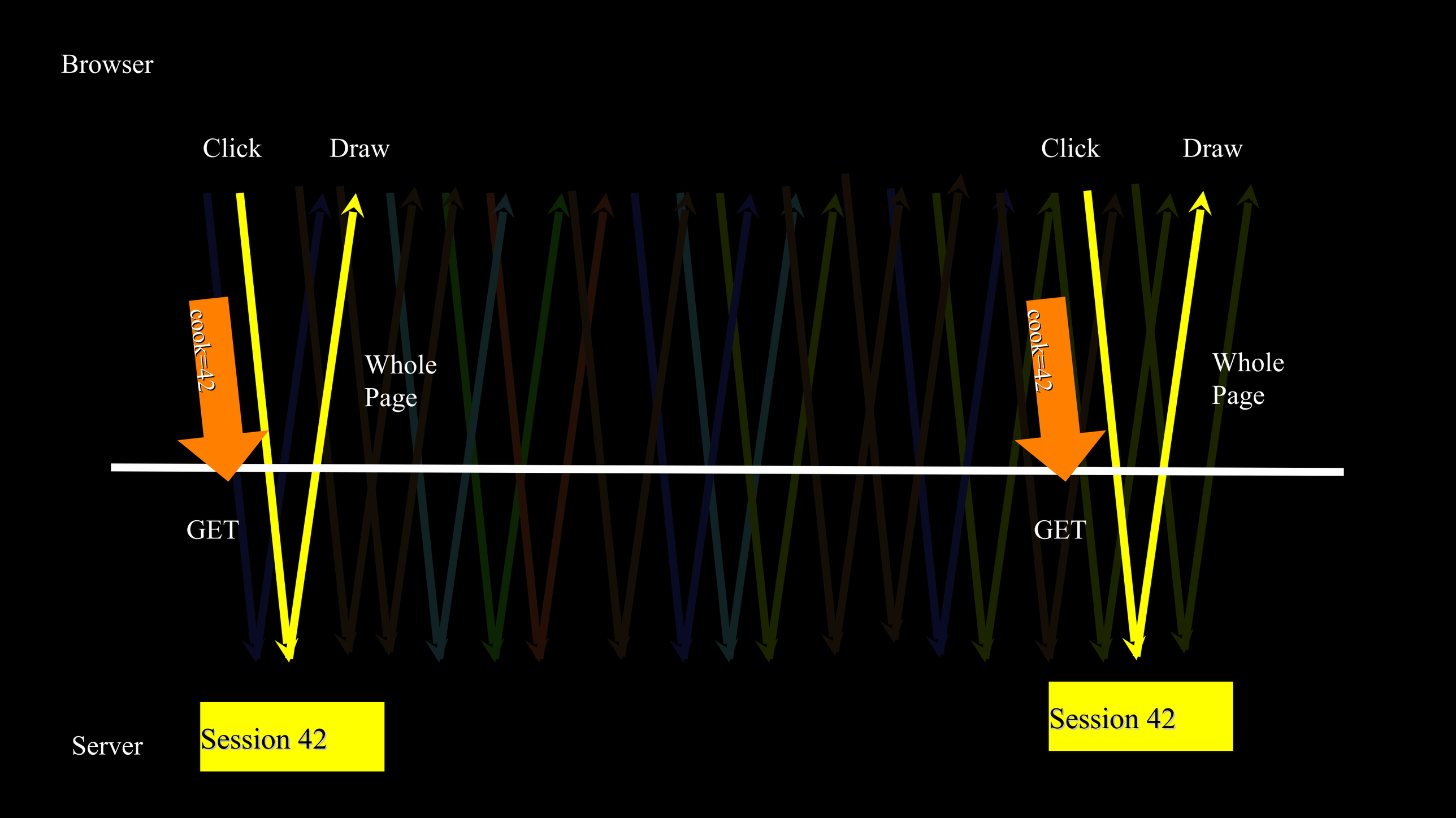
Server

How you see YouTube...

Browser



Server


Browser
Click
Draw
Click
Draw
cook=42
Whole Page
cook=42
Whole Page
GET
GET
Server
Session 42
Session 42

??? Server Questions ???

- Who is this user?
- Are they logged in yet?
- What screen did they come from?
- What button did they push?
- Where do we store this data?
- What screen do they want next?

Cookie/Session Summary

- Cookies take the stateless web and allow servers to store small “breadcrumbs” in each browser.
- Session IDs are large random numbers stored in a cookie and used to maintain a session on the server for each of the browsers connecting to the server
- Server software stores sessions *somewhere* - each time a request comes back in, the right session is retrieved based on the cookie
- Server uses the session as a scratch space for little things