

# Sessions

# How to Make an Application out of a Web Page?

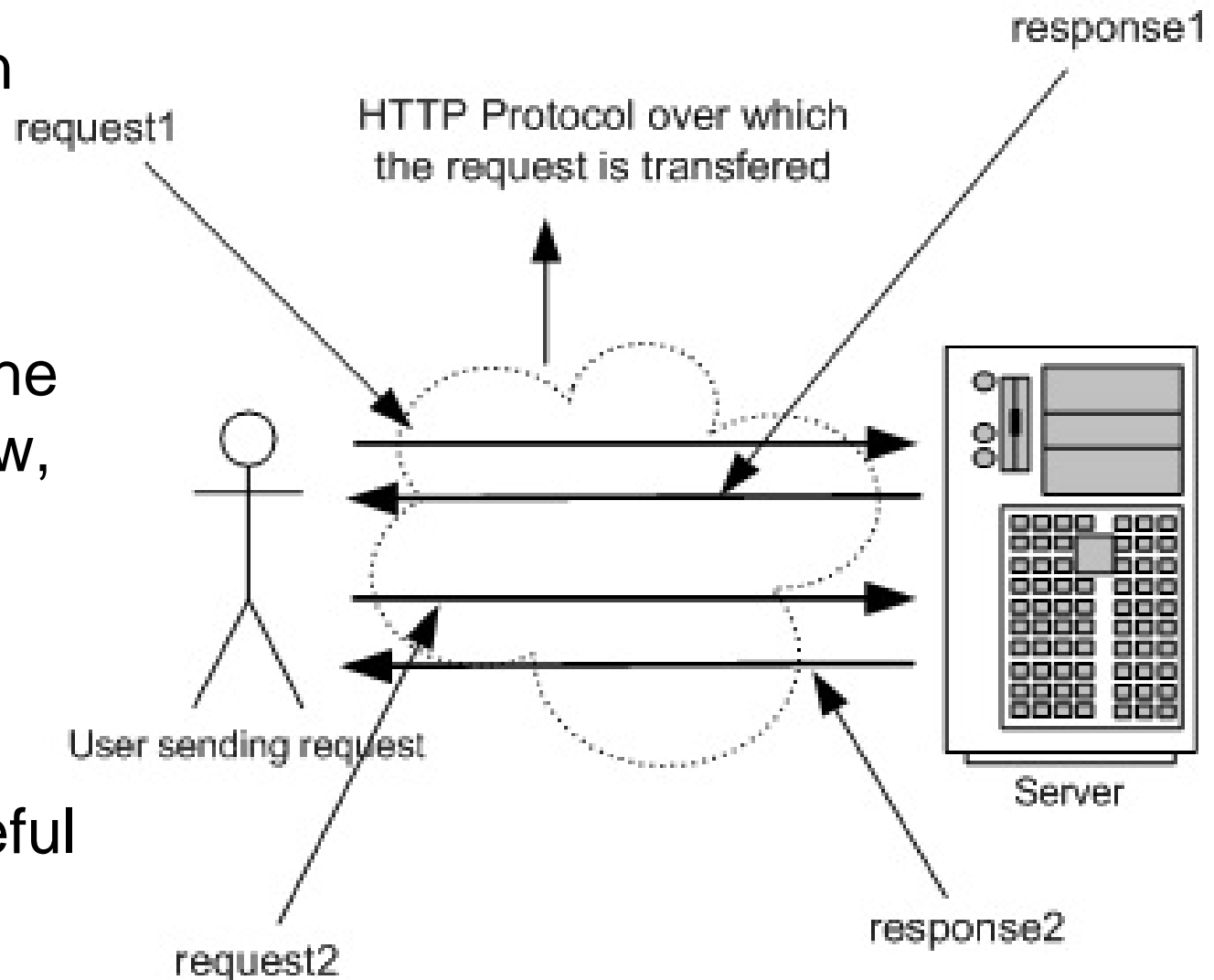
---

- On the internet, a web page is a web page is a web page...
  - If you surf from ./page1.html to ./page2.html these are two unique requests.
  - The server doesn't know anything about the fact that both pages are visited by the same user.
- Sessions are the technique used to logically group several requests into a “group” (called a session)
  - If you start a session, the server will know that it's still the same user who surfed from ./page1.html to ./page2.html



# Sessions

- HTTP itself is “stateless”
  - no state stored on the server between requests from the same client
- but many web apps are stateful
  - necessary to connect requests from the same user / browser / browser-window, e.g. shopping cart, appointments calendar etc...
- *Session*
  - multiple requests performed in a stateful context
- *Session tracking*
  - technique that allows sessions in stateless environments



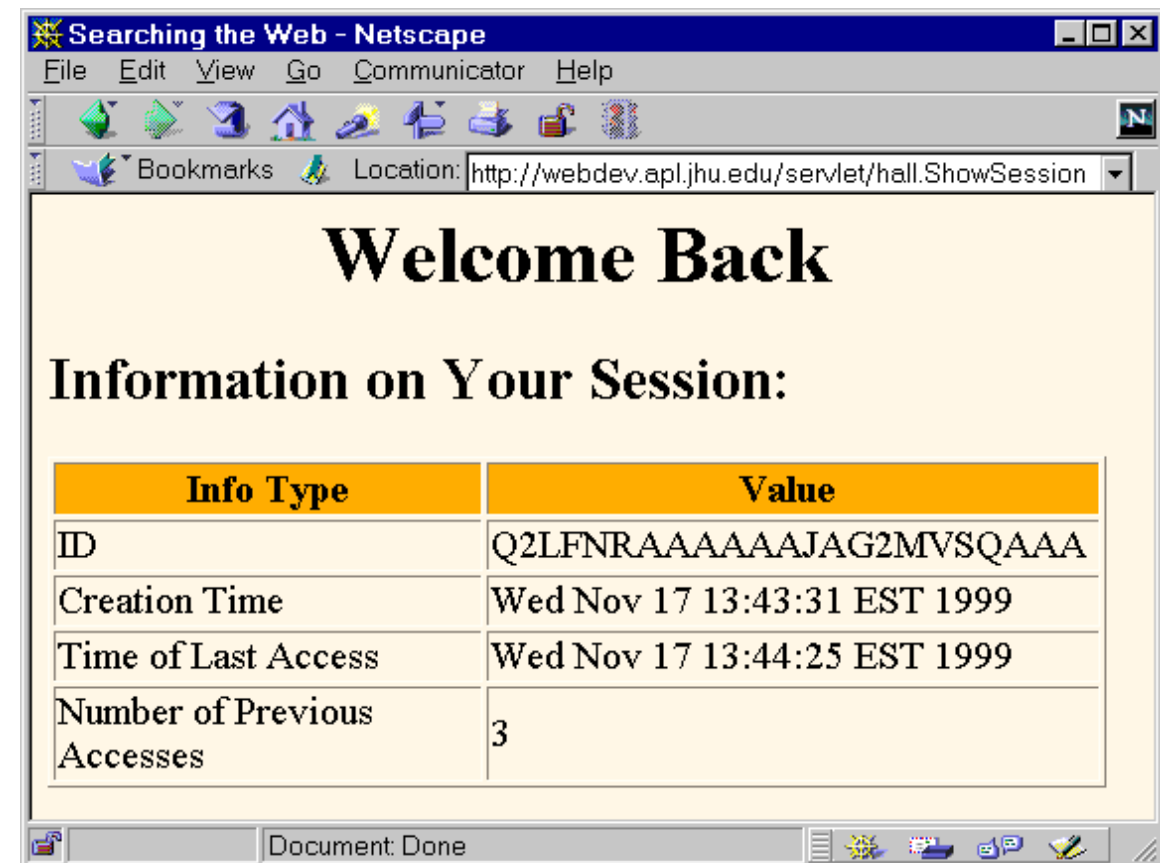
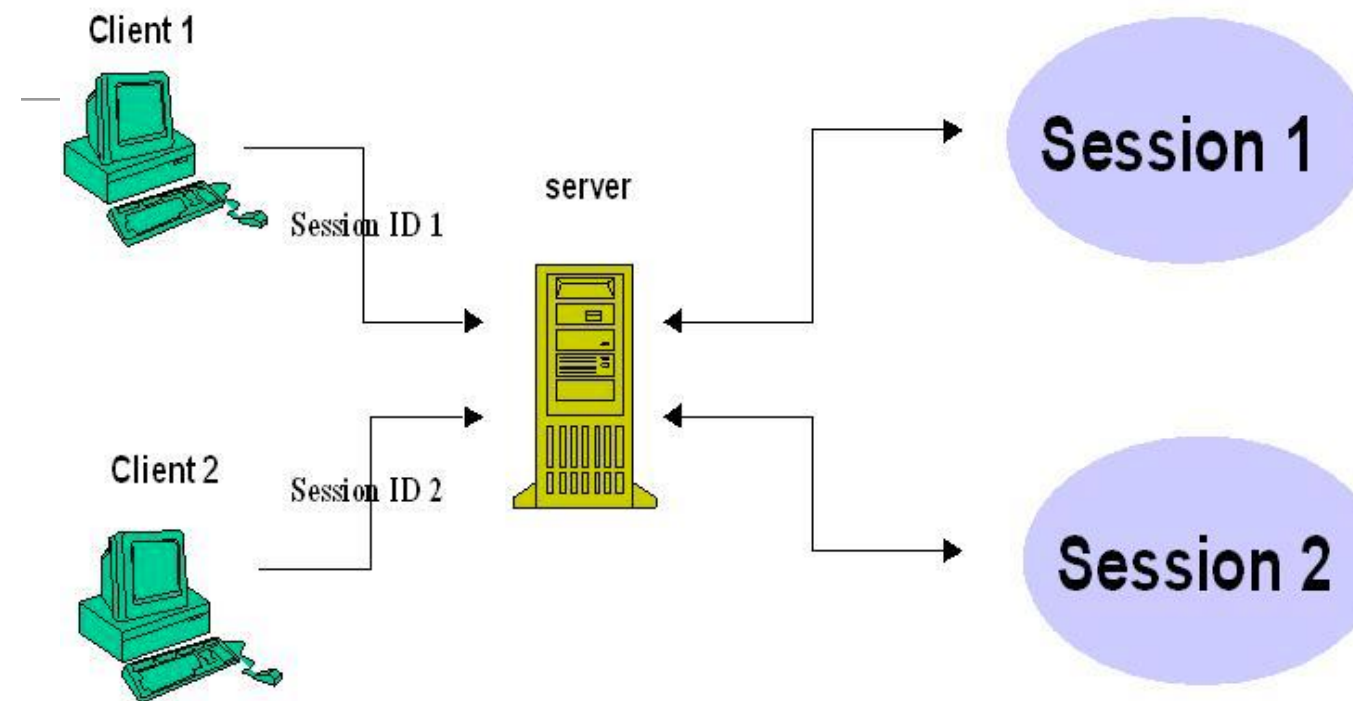
# Sessions

---

- HTTP protocol is stateless by design, each request is done separately and is executed in a separate context.
- The idea behind session management is to put requests from the same client in the same context.
- This is done by issuing an identifier by the server and sending it to the client, then the client would save this identifier and resend it in subsequent requests so the server can identify it.

- User surfs to <http://demo.com>
  - Server (on 1st request / if no sessionID stored on client)
    - generates unique session id, which is mapped to ...
    - ... a session-object
      - stored in memory (lost on shutdown), in a file or in database
      - can contain anything (list of articles, game state, counters, ...)
  - Session id is added to the response
- from now on:
  - each subsequent request from the same user (browser) must contain the session id ...
  - ... which is used by the server to map to the session-object
- No data gets stored on the client, except SessionID

# Session Tracking



# Session Management

---

- **Session management** refers to the way web servers and web applications track a given user's interactions throughout their site pages while preserving their state between navigation.
- Due to the nature of HTTP, the protocol includes no built-in facility to uniquely identify or track a particular customer (or session) within an application without transmitting some data between the client and the server.
- Typically, the process of managing the state of a web-based client is through the use of session IDs. Session IDs are used by the application to uniquely identify a client browser, while background (server-side) processes are used to associate the session ID with the existing state of the user.

# Session Tracking Techniques

---

- **HttpSession**

- **Cookies**

- **Hidden Field**

- **URL-Rewriting**

# HttpSession

---

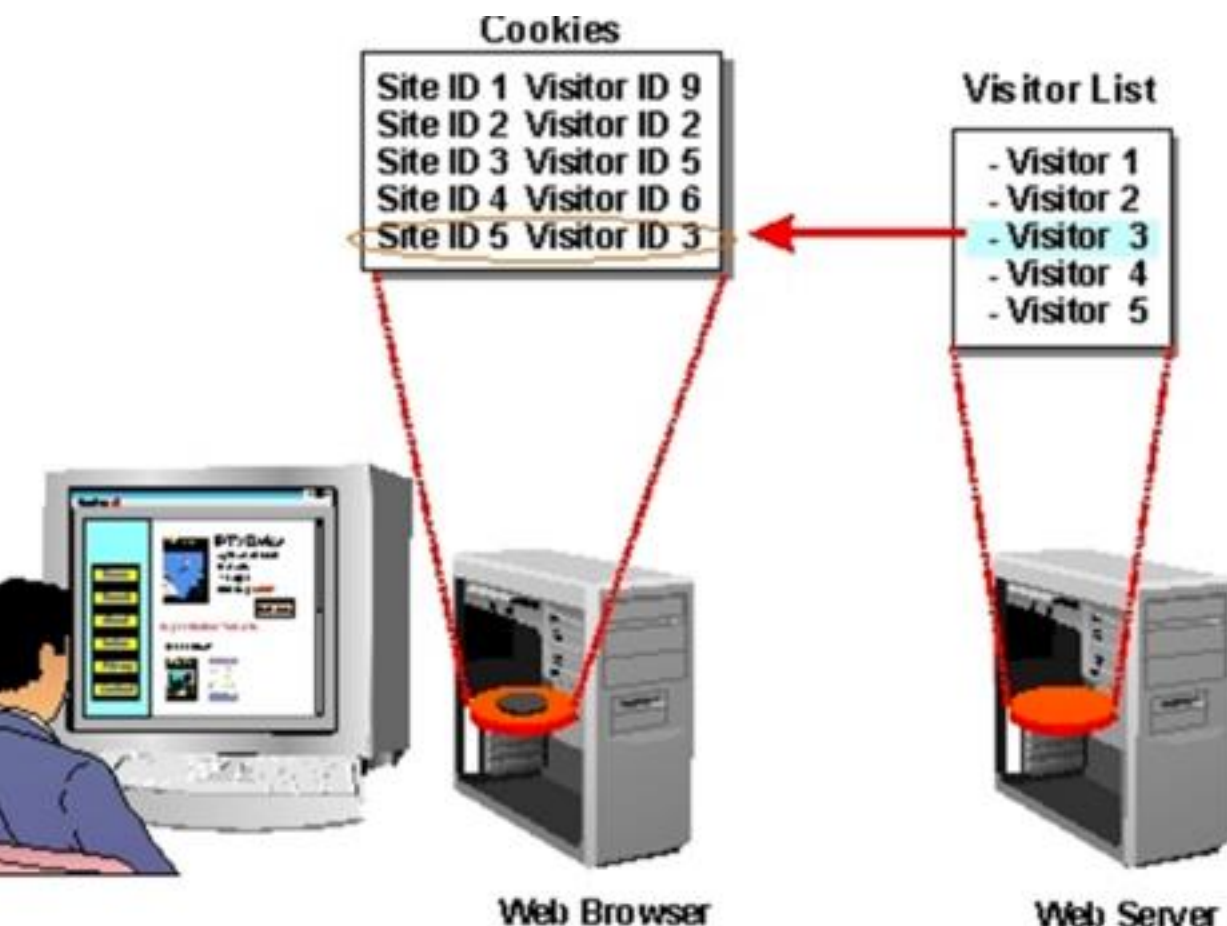
- Provides a way to identify a user across more than one page request or visit to a Web site and to store information about that user.
- The container uses this interface to create a session between an HTTP client and an HTTP server. The session persists for a specified time period, across more than one connection or page request from the user. A session usually corresponds to one user, who may visit a site many times. The server can maintain a session in many ways such as using cookies or rewriting URLs.



# Cookies

A cookie is a small piece of data stored on the client-side which servers use when communicating with clients.

They're used to identify a client when sending a subsequent request..



1. Server creates a cookie with session-id on first request
2. Server maps id to a new user-specific session object
3. The session-id is sent to the client with the first response
4. ..and automatically added by the browser on each further request (to the same address/domain/...)
5. Server receives request + cookie with session-id
6. Server maps session-id to session-object

# URL Rewrite

---

- Basic idea:
  - Server adds the session-id to all links the user can follow
    - <http://server/myhome>
  - is changed to
    - <http://server/myhome?sessionid=123>
  - session-id must be dynamically added
    - functionality usually offered by scripting frameworks

# Hidden Form Fields

---

- In HTML, we can define "hidden" fields in a form
  - `<input type="hidden" name="sessionid" value="123">`
- These fields are not visible and cannot be changed by the client
- Usage:
  - server creates a session-object for each client and generates a unique ID
  - When HTML documents are created and sent back, the hidden form field is automatically generated containing the actual ID
  - Upon form submit, the session ID is automatically sent back to the server
  - The server can associate this call with an already existing session

# Web Frameworks

---

- Cookies generally preferred.
- However, framework will try to ‘abstract away’ specific session management technology, and deliver simpler abstraction to the programmer
- Framework may in fact be able to switch between different techniques depending on circumstances.