
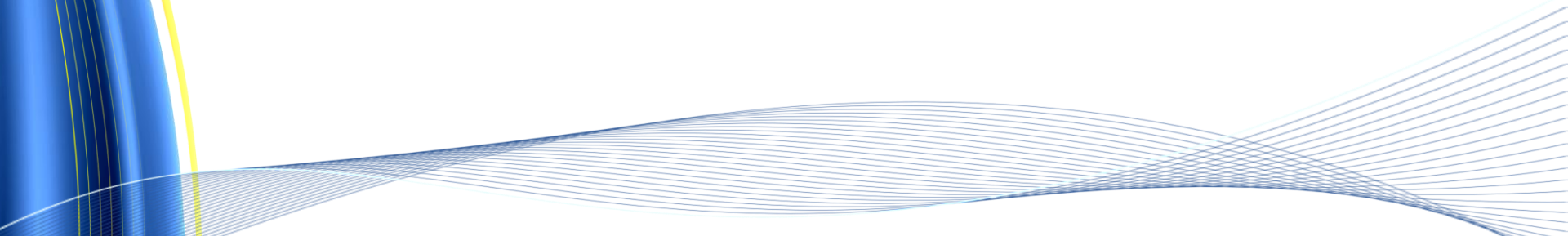


# CH-9 SESSION TRACKING

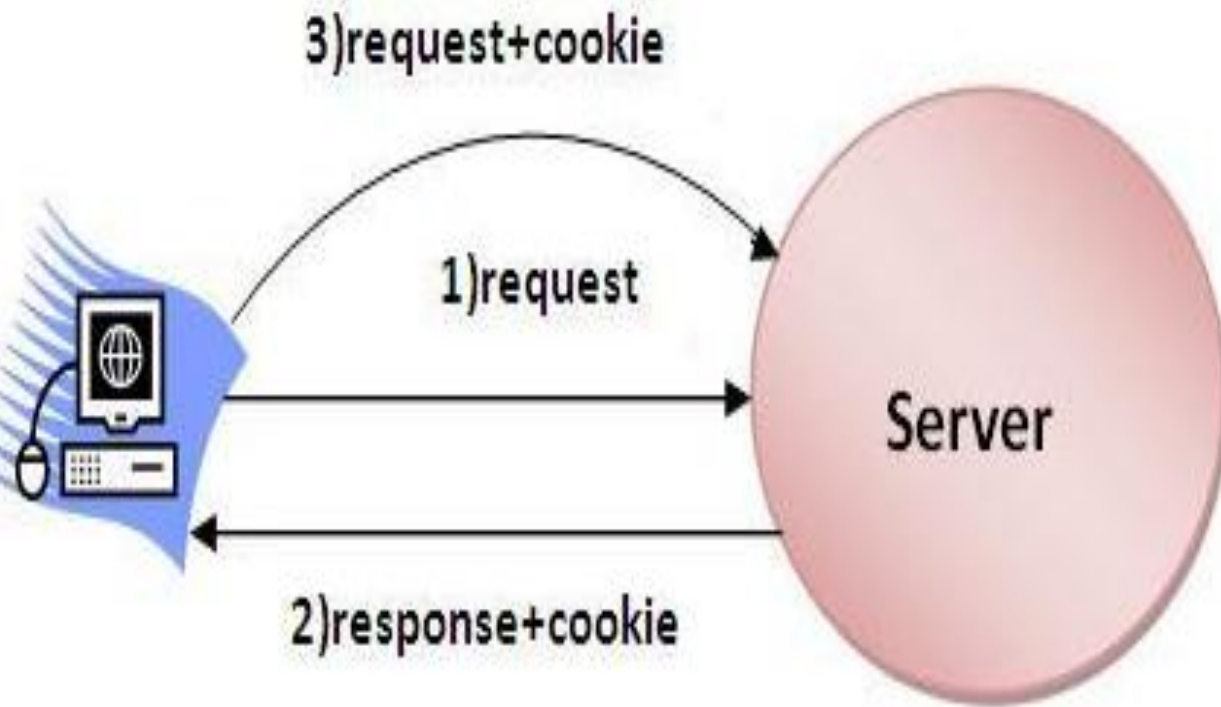
## WHAT IS THE NEED FOR SESSION TRACKING

# SESSION TRACKING

- **Session** simply means a particular interval of time.
- **Session Tracking** is a way to maintain state of an user.
- Http protocol is stateless protocol.
- Each time user request to the server, server treats the request as the new request.
- So we need to maintain the state of an user to recognize to particular user.

- 
- There are four typical solutions to this problem:
  - 1) Cookies
  - 2) URL rewriting
  - 3) Hidden Form Fields.
  - 4) Servlets provide an outstanding session-tracking solution: the HttpSession API.
- 

# COOKIE



# ADVANTAGES AND DISADVANTAGES OF COOKIES

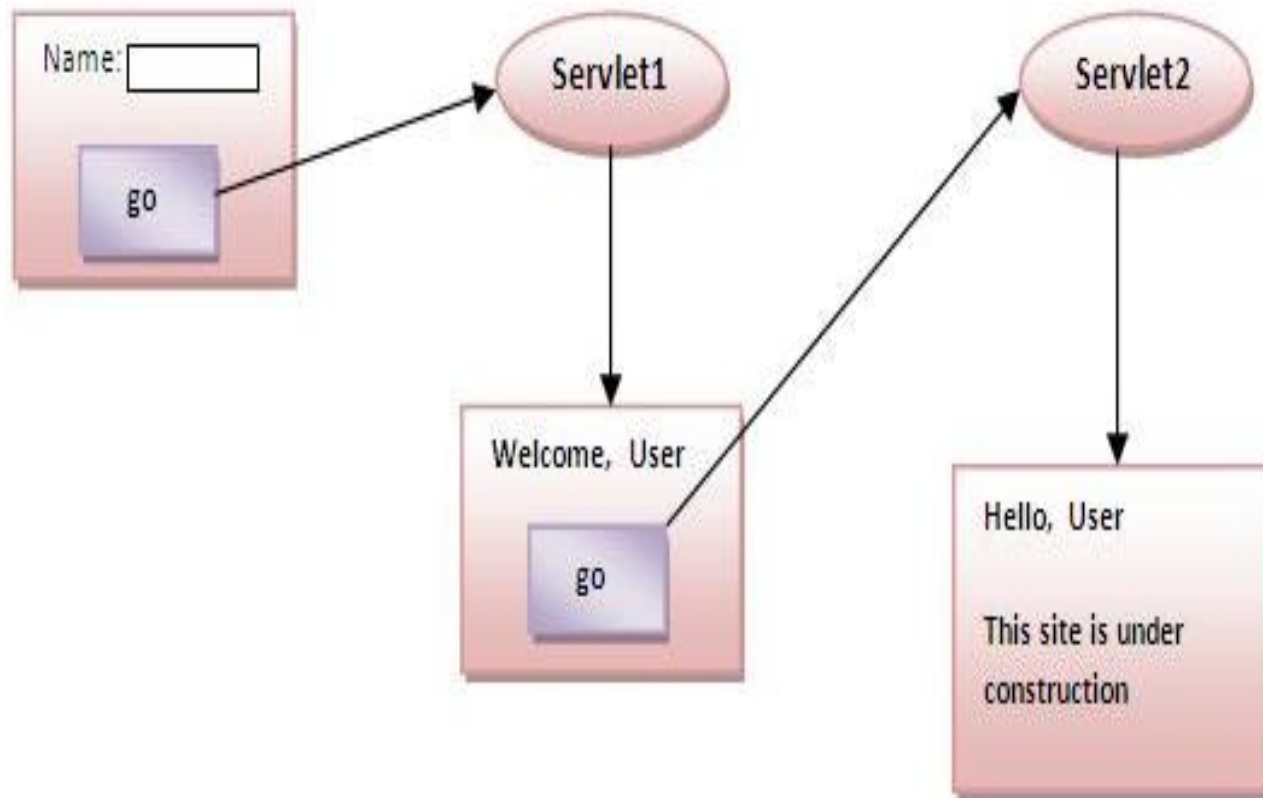
## ➤ **Advantage :**

- 1. Simplest technique of maintaining the state
- 2. Cookies are maintained at client side.

## ➤ **Disadvantage:**

- 1. It will not work if cookie is disabled from the browser.
- 2. Only textual information can be set in cookie object.

# EXAMPLE



# HIDDEN FORM FIELD

- In case of Hidden form field an invisible textfield is used for maintaining the state of an user.
- In such case, we store the information in the hidden field and get it from another servlet.
- This approach is better if we have to submit form in all the pages and we don't want to depend on the browser.
- `<INPUT TYPE="hidden" NAME="technology" VALUE="servlet">`

# ADVANTAGE AND DISADVANTAGE OF HIDDEN FORM FIELD

## ➤ **Advantage:**

- It will always work whether cookie is disabled or not.

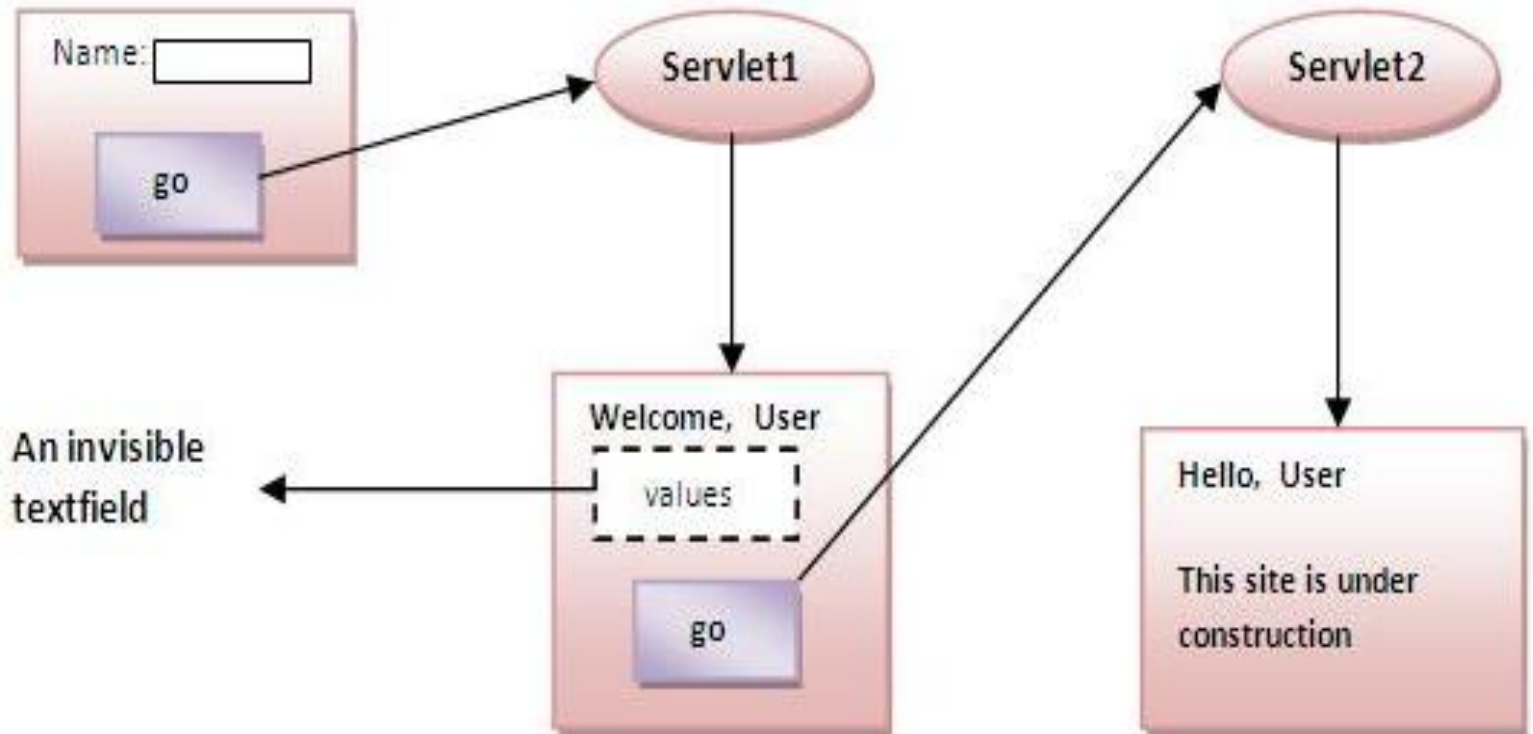
## ➤ **Disadvantage:**

- It is maintained at server side
- Extra form submission is required on each pages.
- Only textual information can be used.



# HIDDEN FORM FIELD

HIDDEN FORM FIELD



# URL REWRITING

- In URL rewriting, we append a token or identifier to the URL of the next Servlet or the next resource.
- We can send parameter name/value pairs using the following format:
- `url?name1=value1&name2=value2&??`
- Original  
URL: <http://server:port/servlet/ServletName>  
Rewritten  
URL: <http://server:port/servlet/ServletName?sessionId=7456>

# ADVANTAGE AND DISADVANTAGE OF URL REWRITING

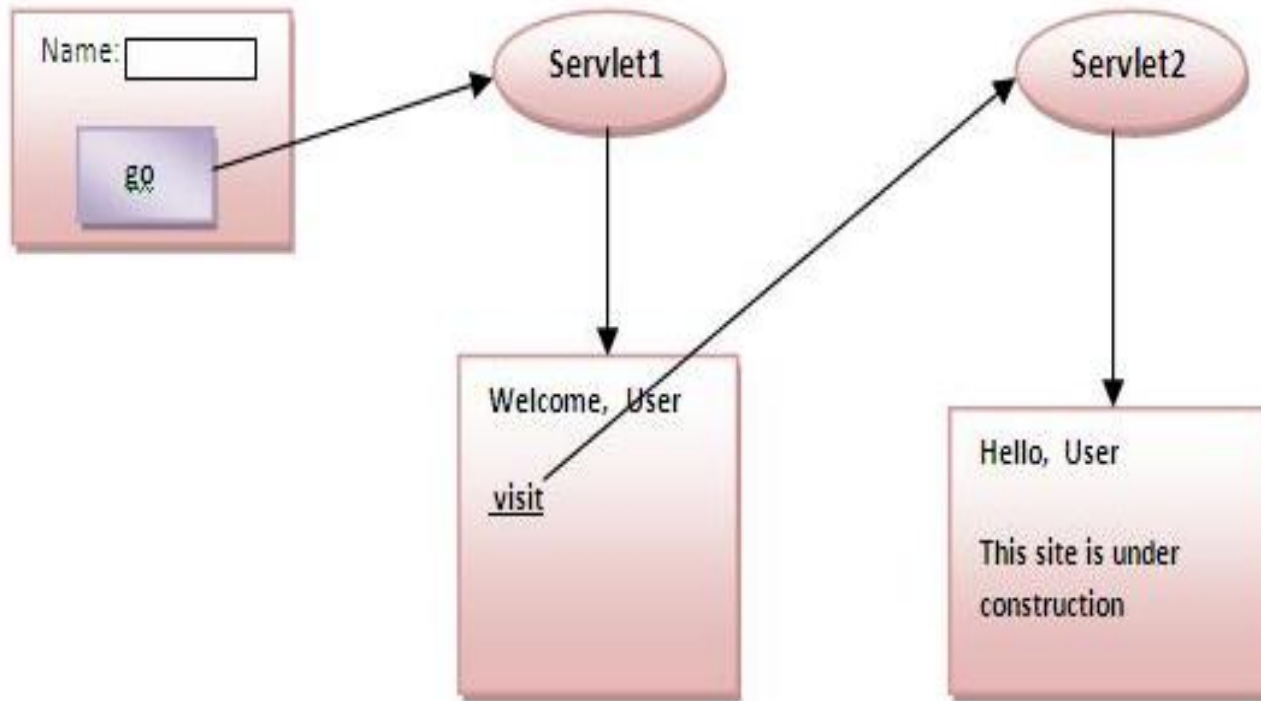
## ➤ **Advantage :**

- 1.It will always work whether cookie is disabled or not(browser independent).
- 2.Extra form submission is not required on each pages.

## ➤ **Disadvantage:**

- 1.It will work only with links.
- It can send only textual information.

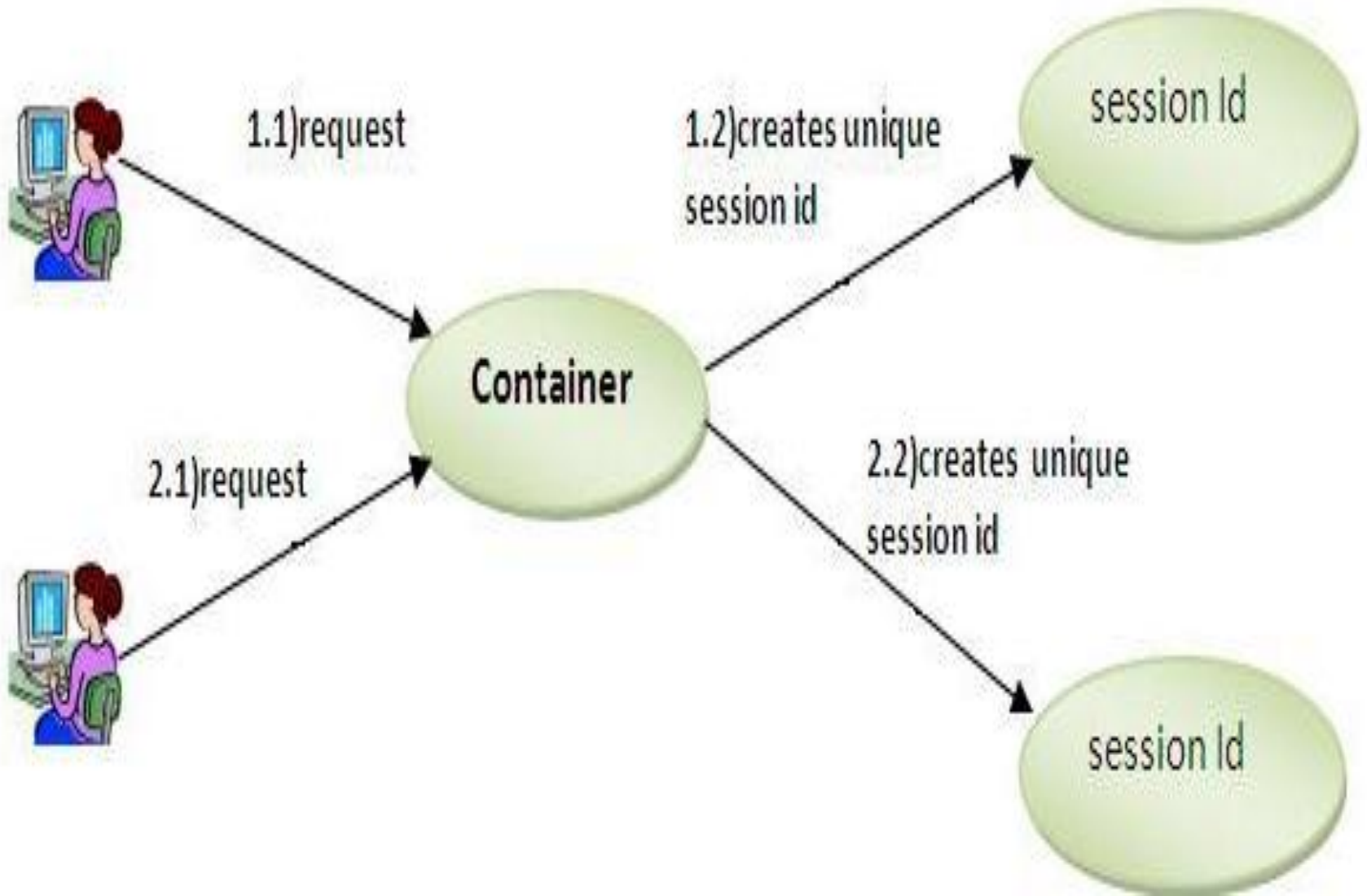
# EXAMPLE



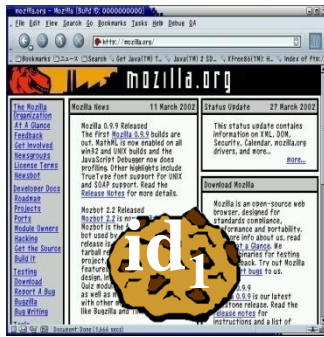
# HTTPSESSION INTERFACE

- In such case, container creates a session id for each user.
- The container uses this id to identify the particular user.
- An object of HttpSession can be used to perform two tasks:
  - 1. bind objects
  - 2. view and manipulate information about a session, such as the session identifier, creation time and last access time.

# EXAMPLE



# SESSION...



**Web browser 1**

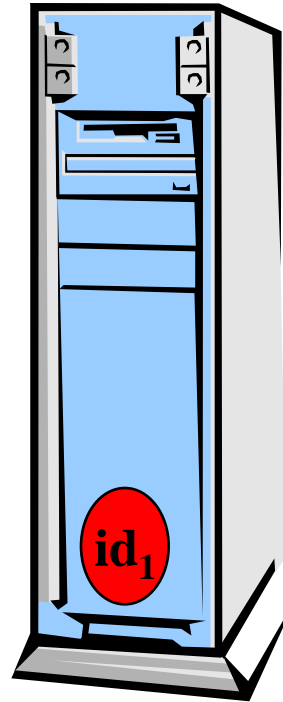
request



put id<sub>1</sub>



response



**Web server**

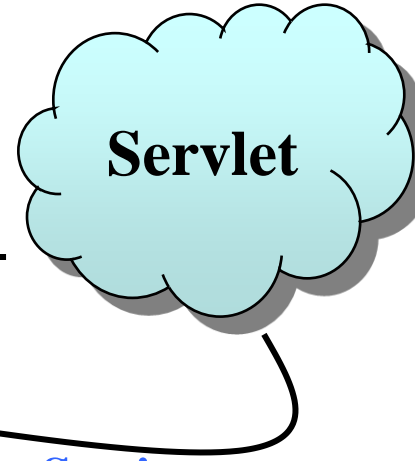
request



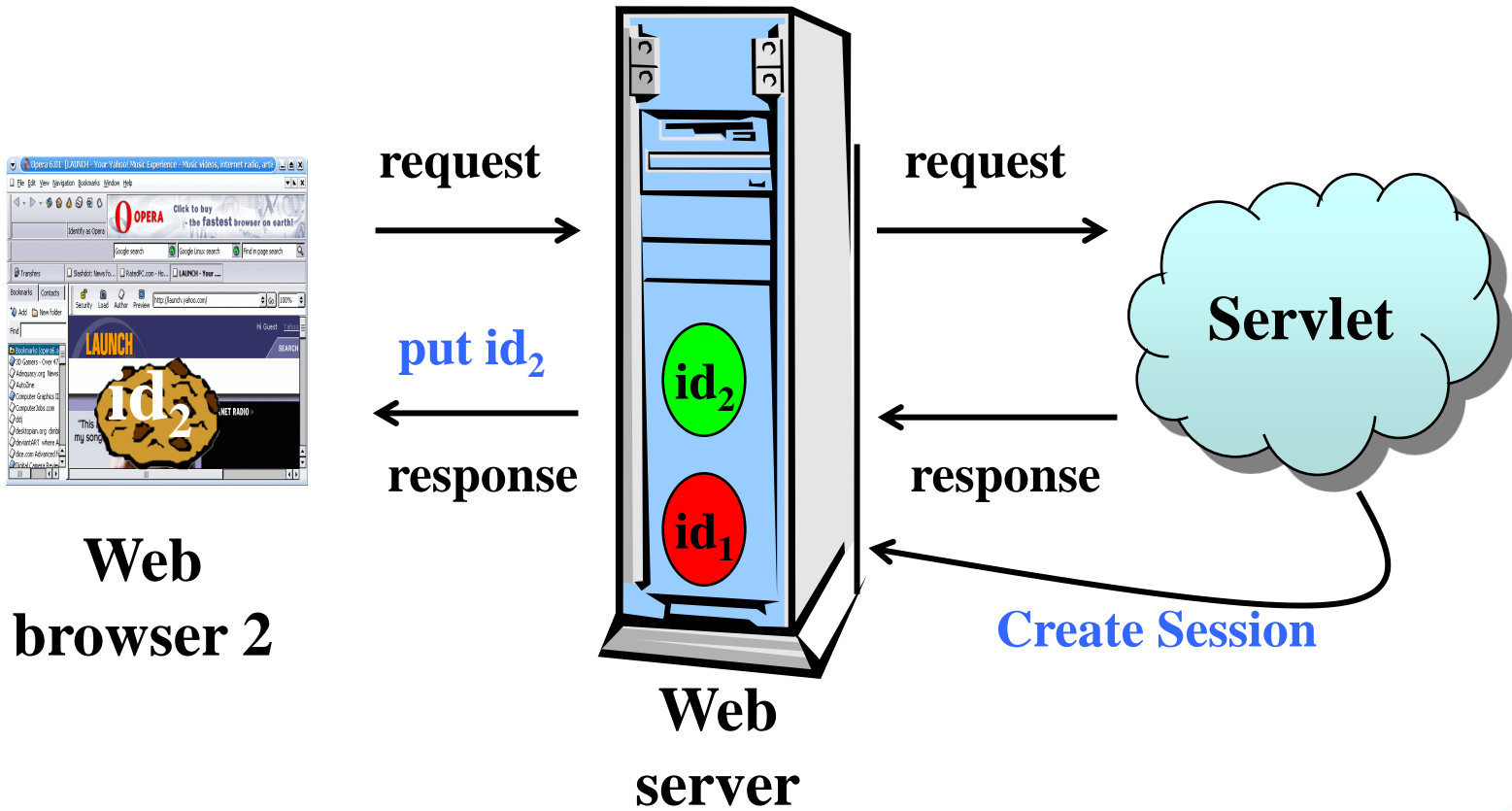
response



**Create Session**

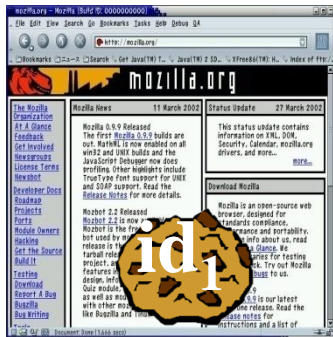


# SESSION.....

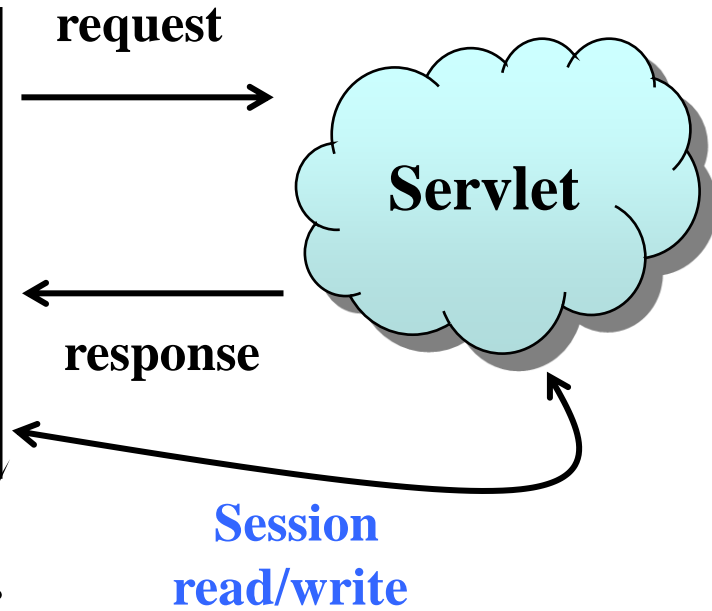
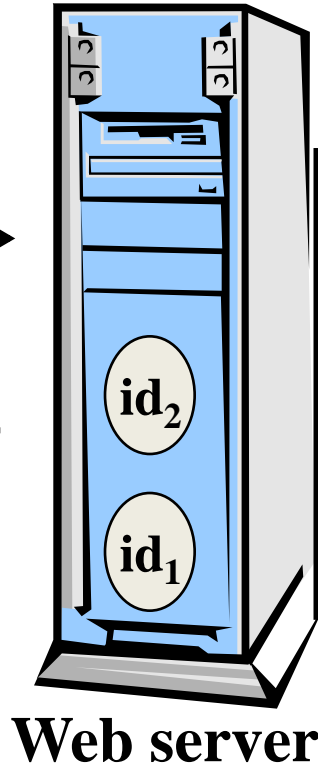
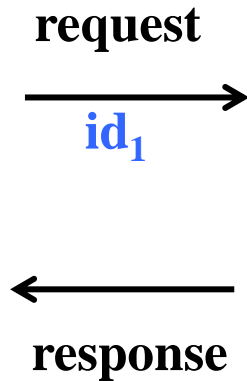




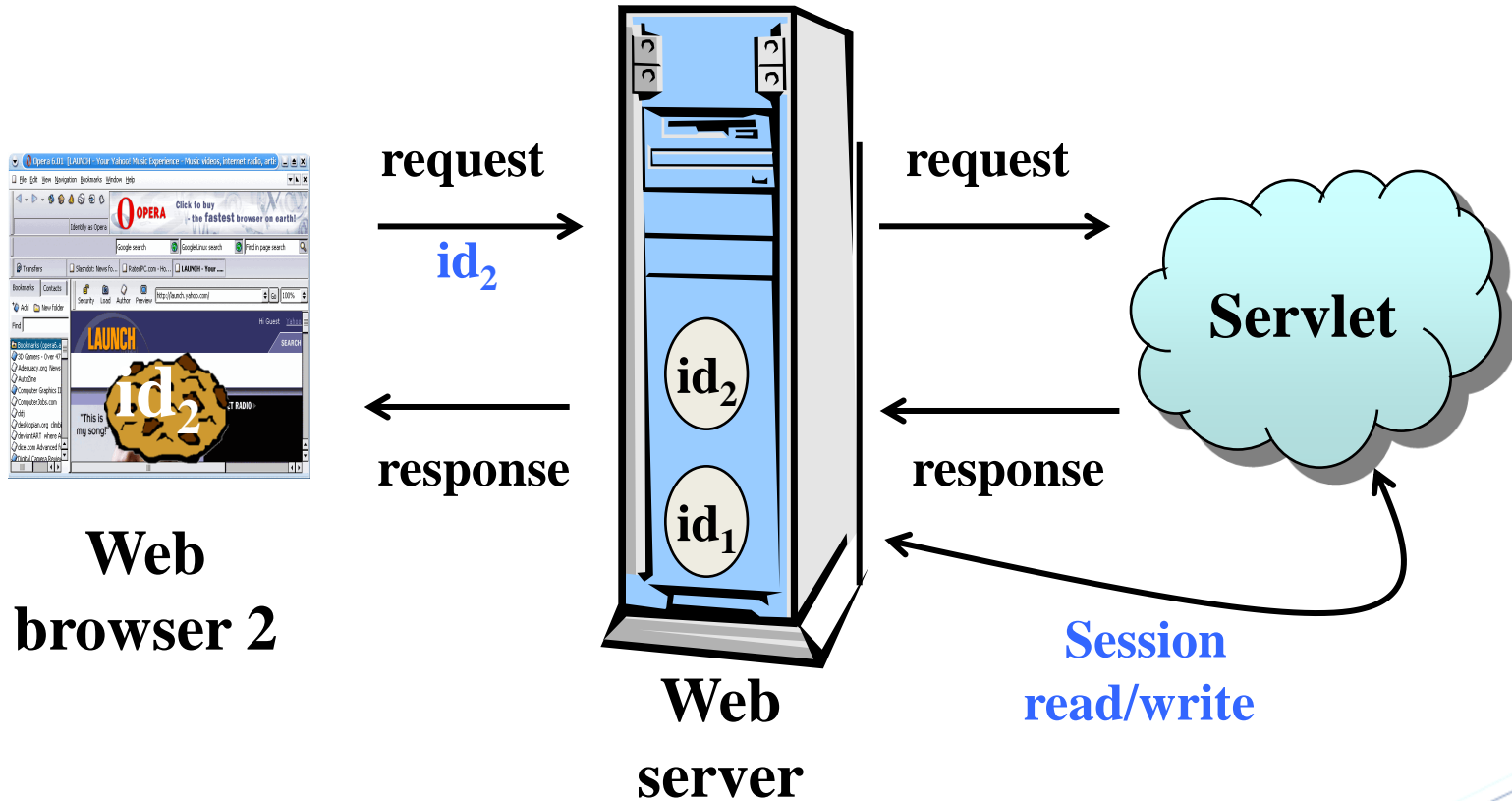
# SESSION...

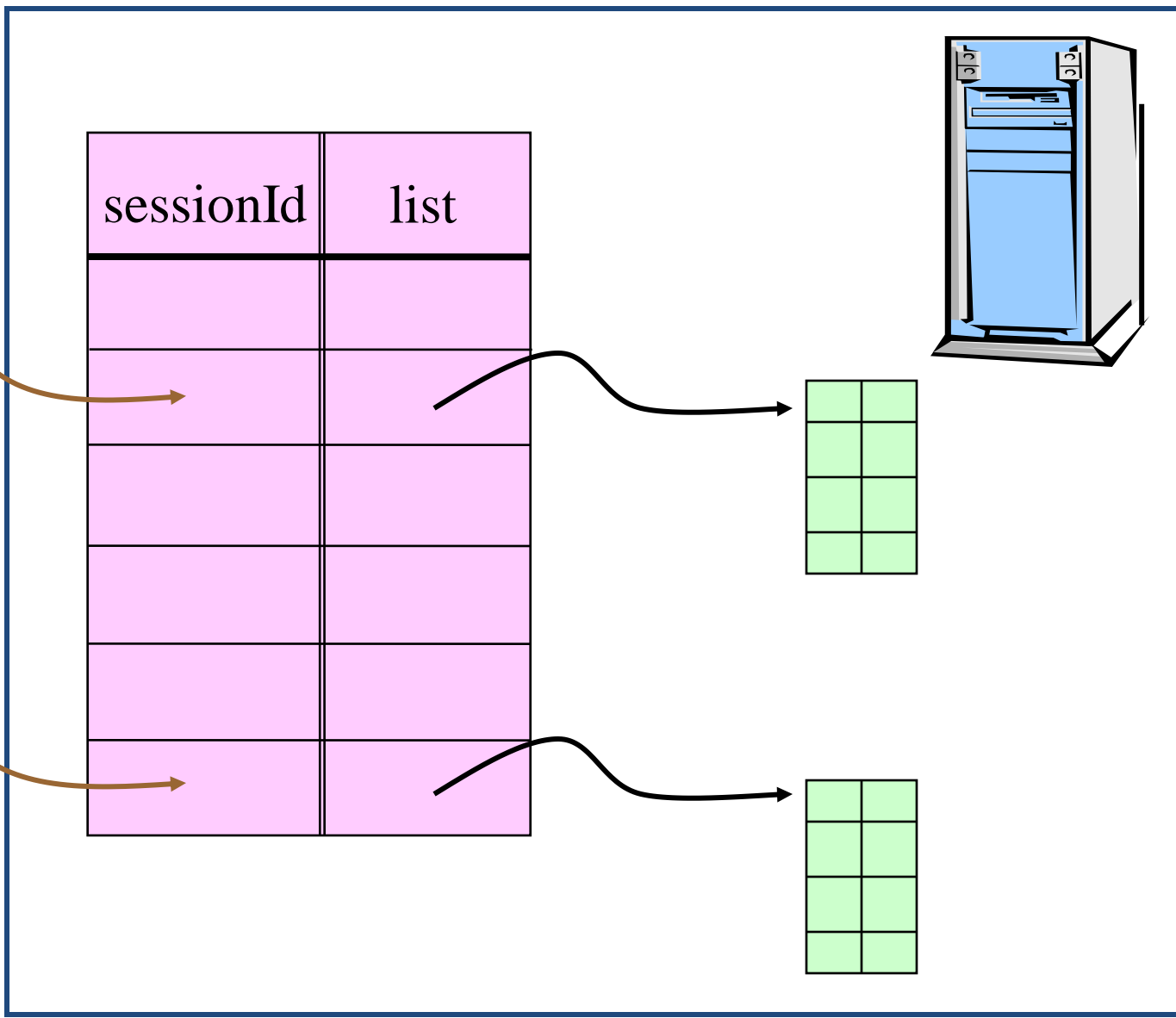
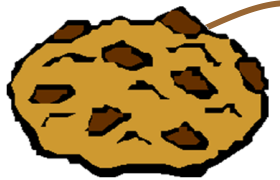


**Web browser 1**



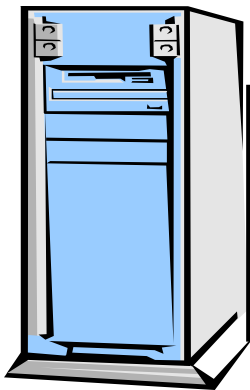
# SESSION...





sessionId	list





# HOW TO GET THE SESSION OBJECT

- The `HttpServletRequest` interface provides **two methods** to get the object of `HttpSession` :
- 1) **`HttpSession getSession()`** : returns the current session associated with this request, or if the request does not have a session, creates one.
- 2) **`HttpSession getSession(boolean create)`**  
: Return the current `HttpSession` associated with this request or, if there is no current session and `create` is true, returns a new session

# METHOD OF HTTPSESSION CLASS

- Object **getAttribute**(String name)
  - retrieves a previously stored value from a session, returns null if no value found.
- Enumeration **getAttributeNames**()
  - Returns names of all attributes in the session.
- void **setAttribute**(String name, Object value)
  - Stores a value in a session.
- void **removeAttribute**(String name)
  - Removes values associated with name.
- void **invalidate**()
  - Expires the session and unbinds all objects with it.

➤ **void logout()**

- This method logs the client out of the Web server and invalidates all sessions associated with that client.

➤ **String getId()**

- returns the unique identifier generated for each session.

➤ **boolean isNew()**

- This method returns true if the client (browser) has never seen the session, usually because the session was just created rather than being referenced by an incoming client request.
- It returns false for preexisting sessions.

- **long getCreationTime()**
  - returns the time in milliseconds since midnight, January 1, 1970 (GMT(*Greenwich Mean Time* )) at which the session was first built.
- **long getLastAccessedTime()**
  - returns the time in milliseconds at which the session was last accessed by the client.
- **int getMaxInactiveInterval()**
- **void setMaxInactiveInterval(int seconds):**
  - These methods get or set the length of time, in seconds, that a session should go without access before being automatically invalidated.
  - A negative value specifies that the session should never time out.
- **Example :ShowSession.java**

# EXAMPLE

- SessionServlet.java
- OrderForm.html, ShowItems.java