Session Tracking mechanism

As a part of webapplication,it is essential to manage the client previous request data at the time of processing later request.

Solution-1

usage of request object

request object gets created, when we send the request to the application and it gets destroyed at the time of sending the response, so we can't keep track of the client previous request data using "request object".

Solution-2

usage of context object

If we use context object, then the data will be shared to all components of the application and to all the users of the application where we can't differentiate the data between the user, so it is not a good approach.

In webapplication to manage the client previous request data at the time of processing later request , we need to have a clear cut seperation b/w the clients data.

To manage this, we need to go for Server side mechanism called "SessionTracking".

Session

It refers to the amount of time the client spend with the server.

State of the Session

The data which is transferred b/w client and server through multiple no of requests during a particular session then that data is called "State of the Session".

In webapplication as soon as we start the server ,ServletContext object is created, when we send the request to the application HttpServletRequest object is created, similarly w.r.t we need to write a code to create "HttpSession" object.

If multiple users uses the application, then we need to write a code to create seperate "HttpSession" object and we need to manage the object, to manage these session objects we need to learn "SessionTracking mechanism".

There are 4 SessionTracking mechanism

1. HttpSessionTracking mechanism.
2. Cookie Session Tracking mechanism.
3. URL-ReWriting Session Tracking mechanism.
4. Hidden Form Field Session Tracking mechanism.(developer specification)

As per SUN specification for ServletApi, we have only 3 mechanism for SessionTracking.

HttpSession Tracking mechanism

What is the difference b/w getSession() and getSession(false)?

Answer: Both the method will return HttpSession object only.

getSession() => container will check whether session object existed w.r.t the particular user or not.

if HttpSession object existed then it would return the same object, otherwise

it will create a new HttpSession object w.r.t the

user and that object will be returned.

getSession(false) => container will check whether session object existed w.r.t the particular user or not.
if HttpSession object existed then it would return the same object, otherwise then container
will return null.

note: getSession() and getSession(true) both are same.

Q> If we allow mulitple users to access webapplication then to container we need to instruct the creation of HttpSession
objects,In this case how container will identify user specific HttpSession object in order to put the user specific attributes
and get the attributes?
Answer: HttpSession objects are created manually, and for these objects container will maintain unique ID in the form of
HexaDecimal value called SessionID.
Container will prepare Session id value in the form of key called "JSessionID".
Container will create a cookie and attach the session id to send it as a response to the browser every time when
the interaction happens b/w client and server.
when we use request.getSession(), container will get the sessionid and checks whether that user specific object is
available or not, if avaiable it will identify that object to process the data.

Note: we can destroy the HttpSession object manually using the method called
public void invalidate()

Limitations of HttpSession Tracking mechanism
=====
1. Creating HttpSession object w.r.t client at the server side is too costly which would decrease the performance of
the application(managing the session object at the server side).
2. HttpSession object is exchanged b/w the client and server through cookie file,if client disables cookie at the client side
then this mechanism won't work effectively.
To resolve this issue, we have a new mechanism called "Cookie Session Tracking".
refer: HttpSessionTrackingApp

Cookie Session Tracking
=====
In case of Cookie Session Tracking mechanism, we need to create a cookie for every request data w.r.t every user.
After creating a cookie, we need to send this cookie along with response object.
These cookies will be exchanged b/w client and server in the form "request-response" object.
To Create a cookie we use
public Cookie(String key,String value)
To retreive the cookies we use request object
public Cookies[] getCookies()
To add the cookies to response object we use
public void addCookie(Cookie c)
refer: CookieSessionTrackingApp

Drawback

========
1. In case of Cookie Session tracking,cookies are maintained at the client side. if browser disables the cookie then this
    mechanism wont' work effectively.
2. Since cookies are stored in the client machine(browser), there is every possibility that client can misuse the data sent
     from the server.(security breach can happen)

      To overcome this limiation we need to use "URLReWriting Mechanism".