Persistent

Session management

Keeping track of who moved my cheese !!!

Agenda

- Session handling

  Hidden form fields

  Cookies

  Server-side http session

  URL rewriting

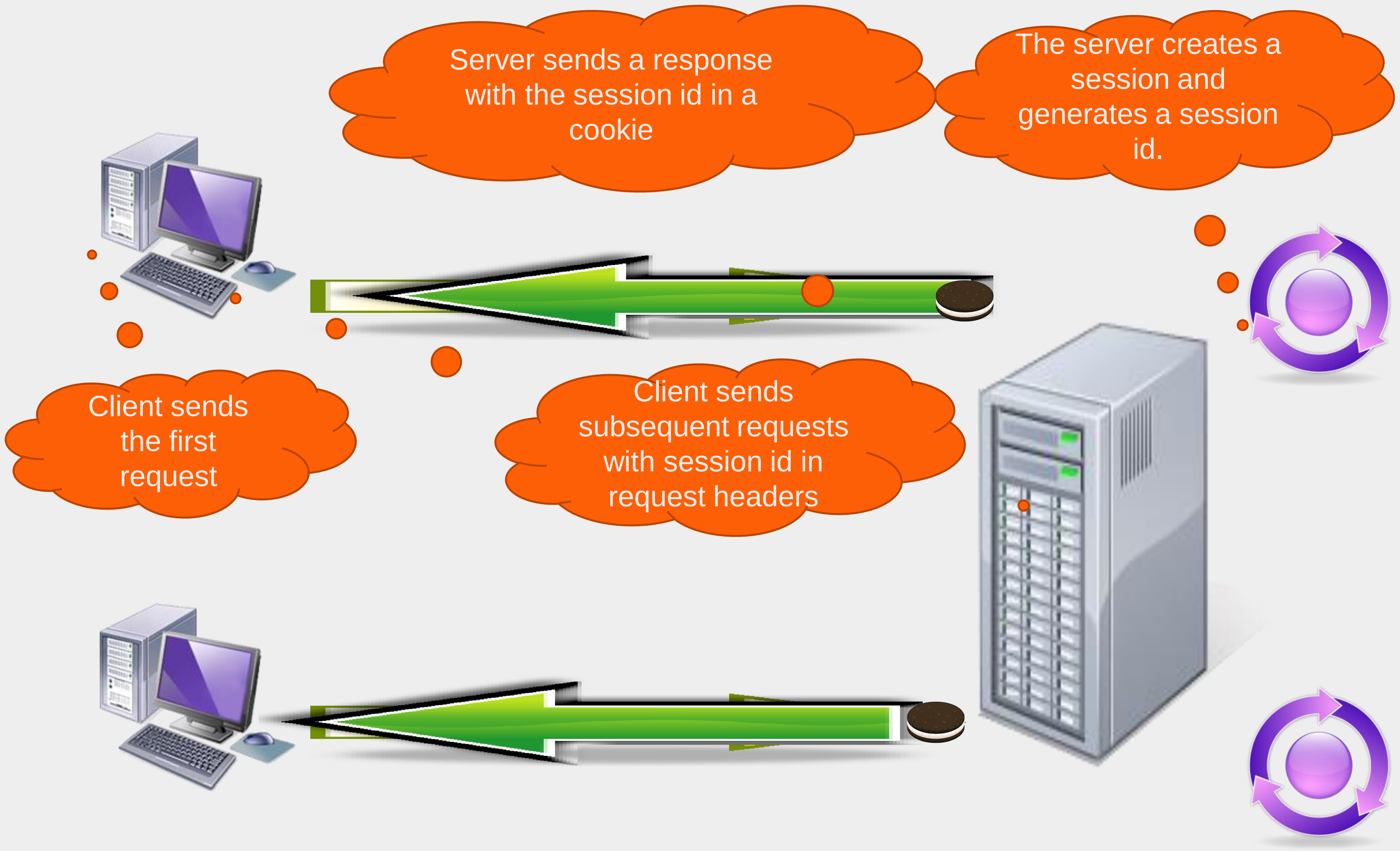
Understanding a session

- HTTP is a stateless protocol.
  - Each time a request is to be sent a new connection is opened
  - The web server receives the request, sends a response and closes the connection.
  - In a chain of such request-response cycles, the server has no “memory” of any preceding requests.

- Session management is all about keeping track of this “conversational state” between a client and server.
  - This is useful for the server to serve a more relevant response to the current request based on information tracked through the preceding requests.

Confidential © 2020 Persistent Systems

How stuff works ?


Server sends a response with the session id in a cookie
The server creates a session and generates a session id.
Client sends the first request
Client sends subsequent requests with session id in request headers

Confidential © 2020 Persistent Systems

Cookies

- A cookie is a bit of information sent by a web server in the response headers to a browser that can later be read back from that browser.
- When a browser receives a cookie, it saves the cookie and thereafter sends the cookie back to the server in subsequent requests each time it accesses a page on that server.
- A cookie has a name, value and optional attributes.
- A cookie whose max age attribute is not set (i.e. is negative) lasts as long as the browser is open. These are known as session cookies. The default value for this attribute is -1.
- A cookie whose max age attribute is set (i.e. > 0) is stored as a file on the user's computer. These are known as persistent cookies.

Cookies as a session tracking mechanism

- Advantages
  - Ease of implementation
  - Can be made to persist across browser shut-downs

- Disadvantages
  - Users can turn-off cookies for privacy and security concerns.

Hidden form fields

- Advantages
  - Supported by all browsers

- Disadvantages
  - Works only for a sequence of dynamically generated forms
  - Breaks down with static, emailed and bookmarked pages

Confidential © 2020 Persistent Systems

Persistent

URL rewriting

- URL’s can be encoded to include the session id.

- If cookies are enabled and url encoding is used then the url’s are returned unchanged.

- If cookies are disabled and url encoding is used the url’s are changed to contain session information.

Cookie notes

- A cookie in a JavaEE web application is an instance of javax.servlet.http.Cookie.
- Programmers can add more cookies to the response in addition to the session cookie which contains the session id added by the server.
- A cookie is added by invoking the addCookie(Cookie) method on the response object.
- A cookie can be retrieved by invoking the getCookies() method on the request object. Returns an array of Cookie objects.
- Key javax.servlet.http.Cookie methods

- getName()	Returns the name of the cookie as a string
- getValue()	Returns the value of a cookie as a string
- setValue(String)	Set a value to this cookie
- getMaxAge()	Returns the max. age of a cookie in seconds
- setMaxAge(int)	Sets the maximum age in seconds for a cookie

Confidential © 2020 Persistent Systems

Features of session tracking in JavaEE web applications

- Provides a high level API for session management
  - Built on top of cookies and URL encoding
- The container maintains
  - Internal hash table of session id’s
  - Session information in javax.servlet.http.HttpSession objects
- Generates and maintains session id
- Provides a simple API for adding and removing session information (attributes) to javax.servlet.http.HttpSession.
- Can automatically switch to URL encoding if cookies are unsupported or disabled.
  - Works only if the programmer explicitly asks url’s to be encoded

Http session

- To retrieve a client’s existing session object or create a new session object
  - request.getSession()
  - request.getSession(true)

- To retrieve a client’s existing session object if any but not create a new one
  - request.getSession(false)

javax.servlet.http.HttpSession interface

- Contains methods to
  - Retrieve and manipulate information about a session like creation time, last accessed time, session id, inactive interval etc.

  - Bind objects to a session allowing client information to persist across request-response cycles.
    - session.setAttribute(<attribute name as string>, <some object>);
    - session.getAttribute(<attribute name as string>);

Confidential © 2020 Persistent Systems

Session invalidation and timeout

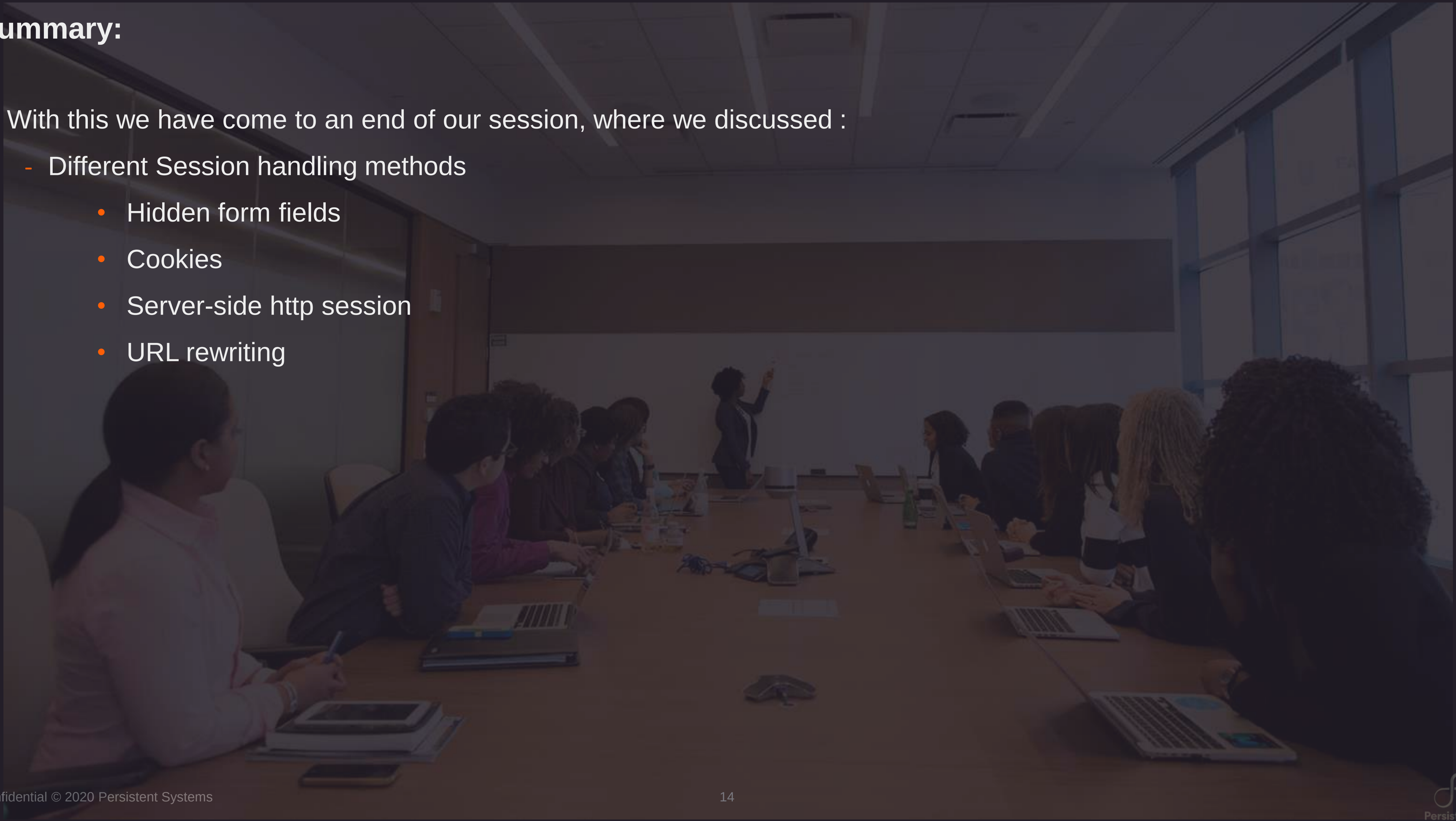
- **Session timeout**
  - Used when an end-user can leave the browser without actively closing the session

  - Session usually times out after 30 minutes of inactivity
    - Product specific
    - Can be configured in the web.xml

```
<session-config>
            <session-timeout>30</session-timeout>
</session-config>
```

- **Session invalidation**
  - Can be used by the programmer to end a session proactively; for example when the logout link is clicked.
    - session.invalidate()
    - Expire the session and unbinds all objects with it

Summary:



- With this we have come to an end of our session, where we discussed :
  - Different Session handling methods
    - Hidden form fields
    - Cookies
    - Server-side http session
    - URL rewriting

Appendix

Thank You

Persistent

Thank you