Fundamentals of Web Development

Third Edition by Randy Connolly and Ricardo Hoar

Chapter 15

Managing State

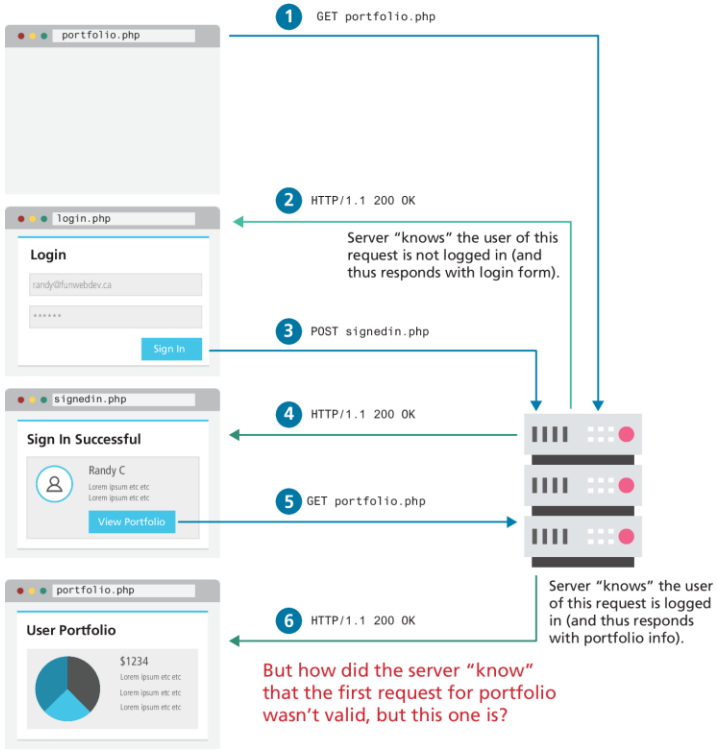
In this chapter you will learn . . .

- Why state is a problem in web application development
- What cookies are and how to use them
- What session state is and what are its typical uses and limitations
- What server cache is and why it is important in real-world websites

The Problem of State in Web Applications

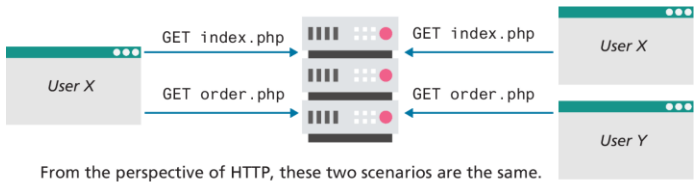
How can one request share information with another request?

The question is: how did the server "know" when the user was and was not logged in? →

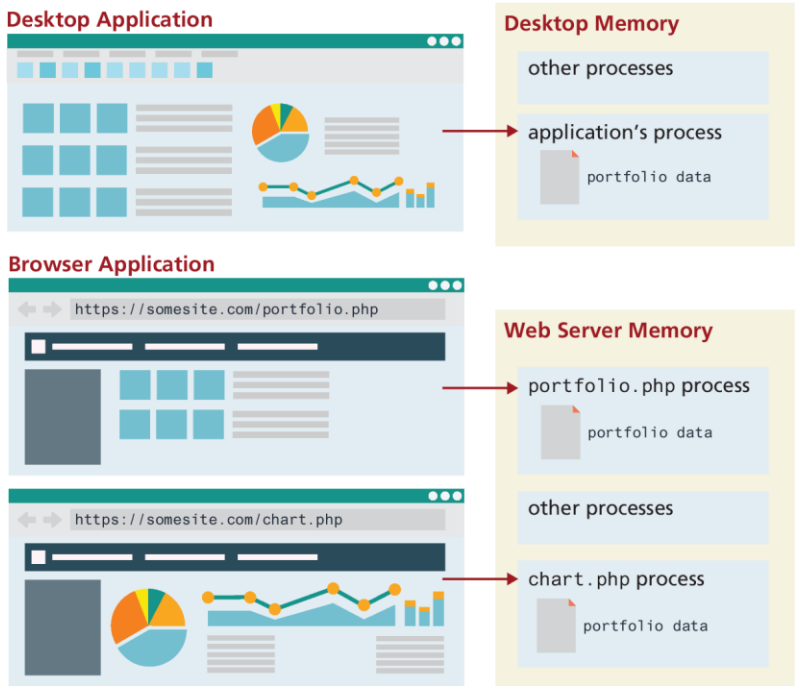


The Problem of State in Web Applications (ii)

Unlike a desktop application, A web application consists of a series of disconnected HTTP requests to a web server where each request for a server page is essentially a request to run a separate program



From the perspective of HTTP, these two scenarios are the same.



Passing Information in HTTP

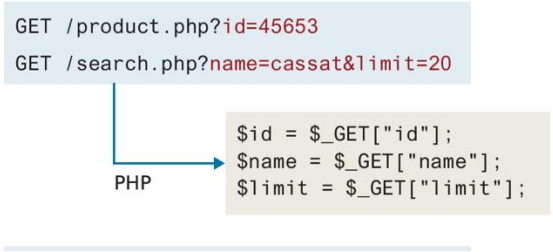
In HTTP, we can pass information using:

- URL
- HTTP header
- Cookies

Passing Information via the URL

Recall a web page can pass query string information from the browser to the server using one of the two methods:

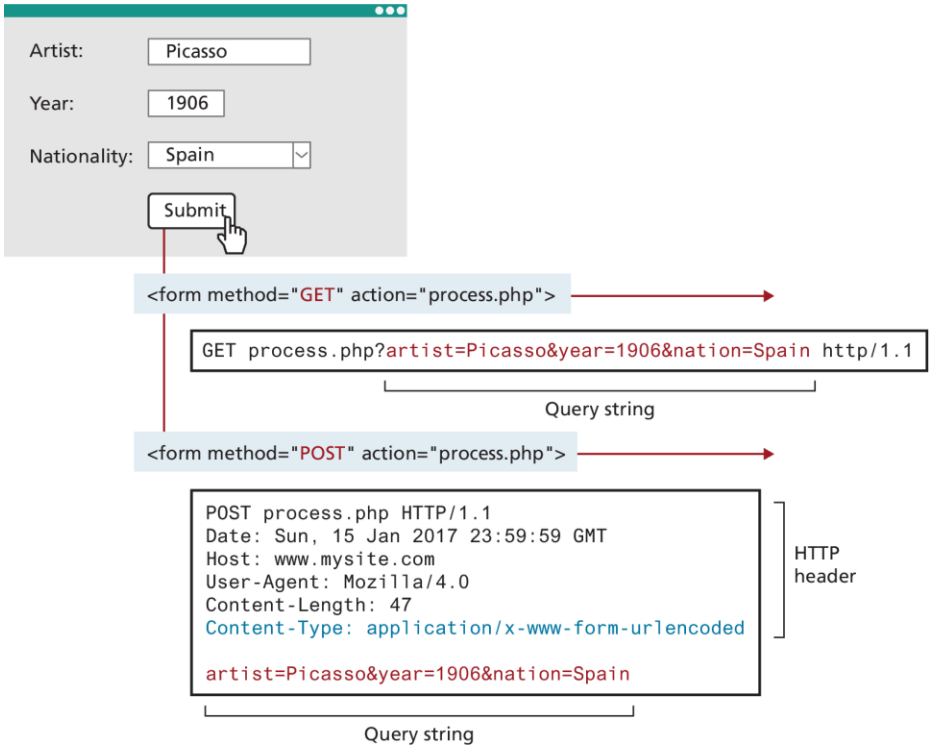
- a query string within the URL **(GET)** →
- a query string within the HTTP header (POST).



Passing Information via HTTP Header

You can see that the form data sent using the POST method is sent as a query string after the HTTP header.

Think of this data being passed via the HTTP header



Cookies

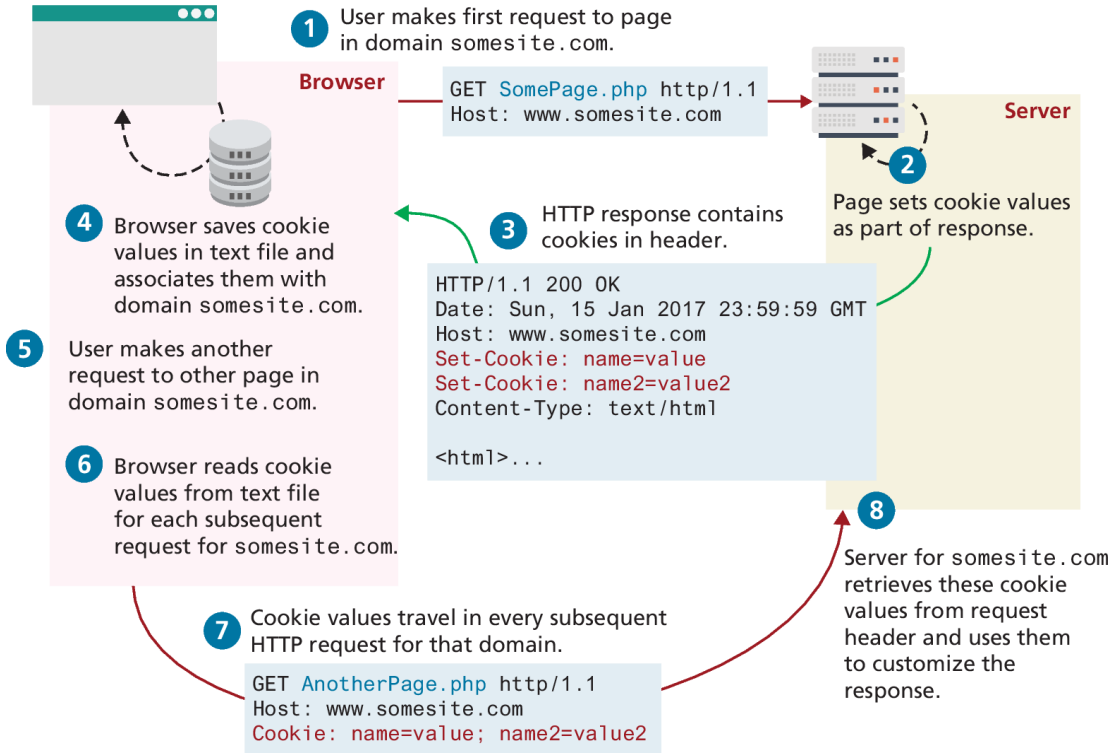
Cookies are a client-side approach for persisting state information.

They are name=value pairs that are saved within one or more text files that are managed by the browser.

While cookies can be used for any state-related purpose, they are principally used as a way of maintaining continuity over time in a web application. One typical use of cookies in a website is to “remember” the visitor so that the server can customize the site for the user.

Cookies are also frequently used to keep track of whether a user has logged into a site.

How Do Cookies Work?



Using Cookies in PHP

Cookies in PHP are *created* using the setcookie() function and are *retrieved* using the $_COOKIES superglobal associative array,

It is important to note that cookies must be written before any other page output.

```
$expiryTime = time()+60*60*24;
// create a persistent cookie
$name = "username";
$value = "Ricardo";
setcookie($name, $value, $expiryTime);
```

LISTING 15.2 Writing a cookie

```
if ( !isset($_COOKIE['username']) ) { echo "this cookie doesn't exist"; }
else {
  echo "The username retrieved from the cookie is:“. $_COOKIE['username’];
}

// loop through all cookies in request
foreach ($_COOKIE as $name => $value) {
                echo "Cookie: $name = $value";
}
```

LISTING 15.3 Reading a cookie

Persistent Cookie Best Practices

Due to the limitations of cookies your site's correct operation should not be dependent upon them.

Almost all login systems are dependent upon IDs sent in session cookies

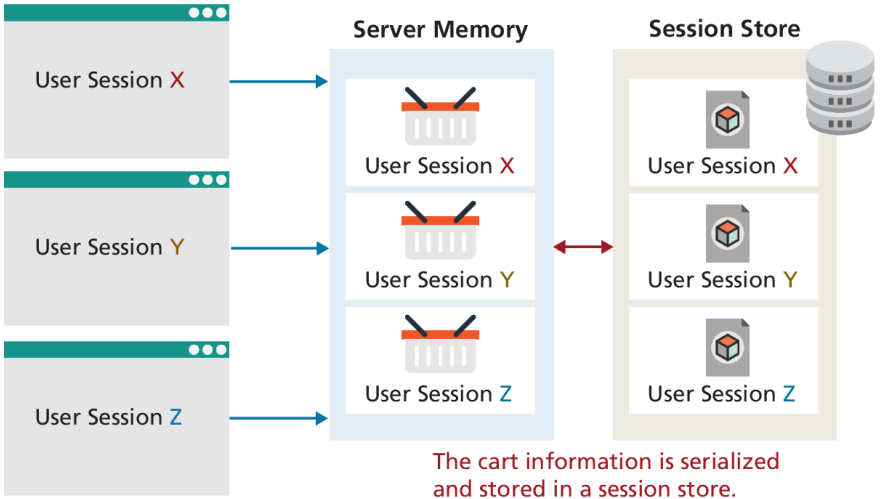
Cookies containing sensitive information should have a short lifetime

A login cookie might contain the username but not the password. Instead, the login cookie would contain a random token used by the site's back-end database. Every time the user logs in, a new token would be generated and stored in the database and cookie.

Session State

Session state is a server-based state mechanism that lets web applications store and retrieve objects of any type for each unique user session.

Session state is dependent upon some type of **session store**, that is, some type of storage area for session information.

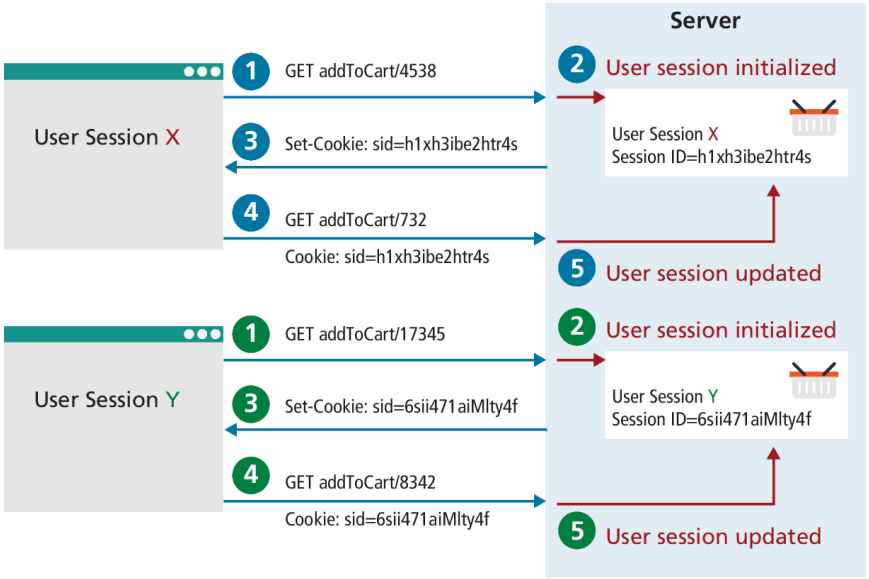


How Does Session State Work?

Since HTTP is stateless, some type of user/session identification system is needed.

Sessions in PHP and Express are identified with a unique session ID.

This session ID transmitted back and forth between the user and the server via a session cookie



Session Storage and Configuration

In the example shown in Figure 15.8, each user's session information is kept in serialized files

The decision to save sessions to files rather than in memory addresses the issue of memory usage that can occur on shared hosts as well as persistence between restarts.

One downside to storing the sessions in files is a degradation in performance compared to memory storage.

Session State in PHP

Session in PHP can be accessed via the $_SESSION variable, but unlike the other superglobals, you have to take additional steps first.

You must call the **session_start()** function at the beginning of the script

If the session object does not yet exist one might generate an error, redirect to another page, or create the required object

```
<?php
include_once("ShoppingCart.class.php");
session_start();
// check for existence of session object before accessing
if ( !isset($_SESSION["Cart"]) ) {
    $_SESSION["Cart"] = new ShoppingCart();
}
$cart = $_SESSION["Cart"];
?>
```

LISTING 15.8 Checking session existence