CT4009

Sessions and Cookies

Andy Bell

What we’ll cover

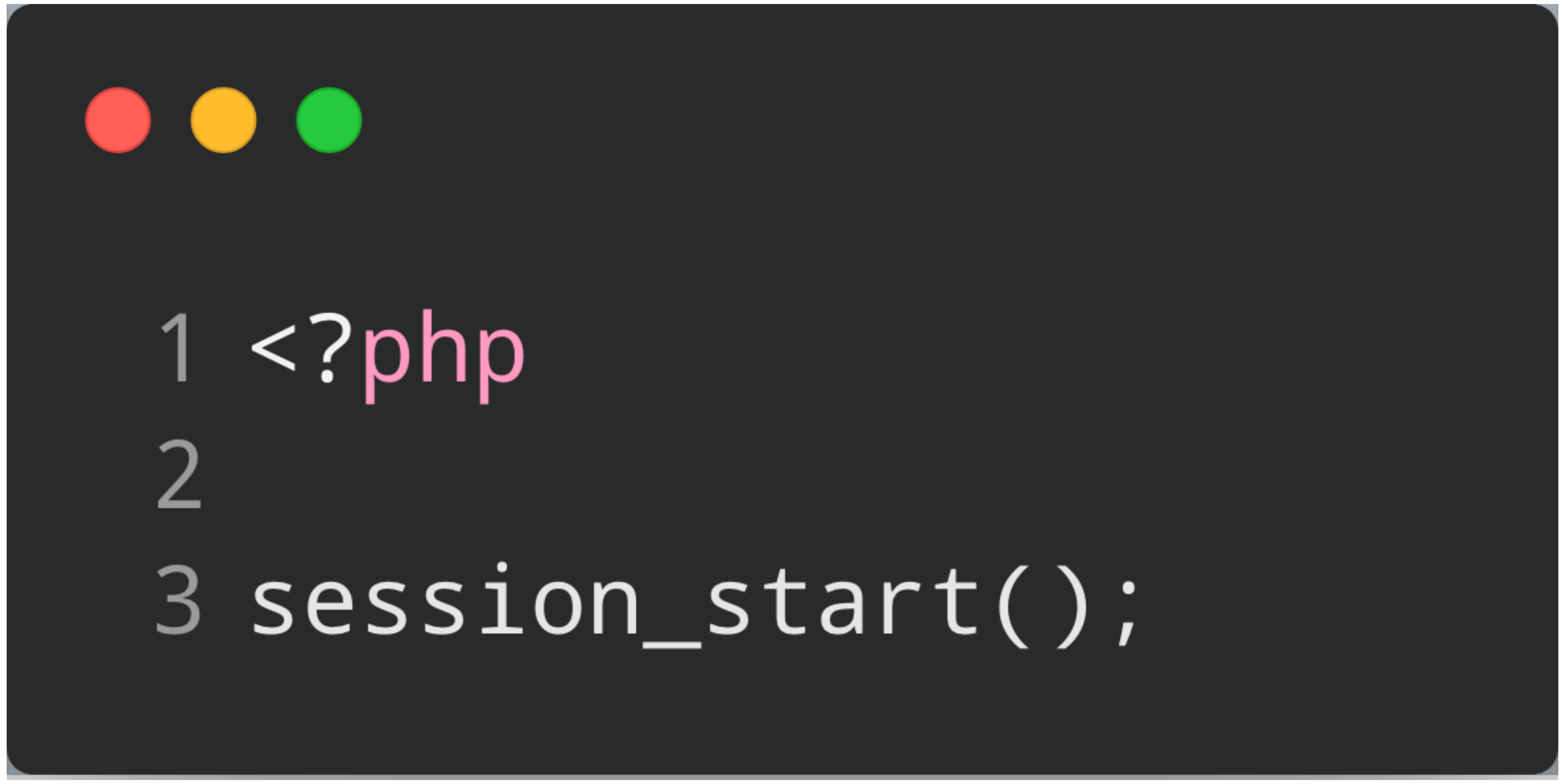
- What a PHP session is
- What a Cookie is
- What the difference between them is
- Best practice
- Practical demos

What is a PHP Session?

- A facility to store small amounts of data against a token in memory on your server
- Often, it’s used to store information about the current user (logged in status)
- It does expire though, so shouldn’t be used for persistent storage
- They’re useful for storing incredibly sensitive information as the info never leaves the server and can only be accessed with a unique token

Session example: Step 1

You’ve got to start/enable sessions first

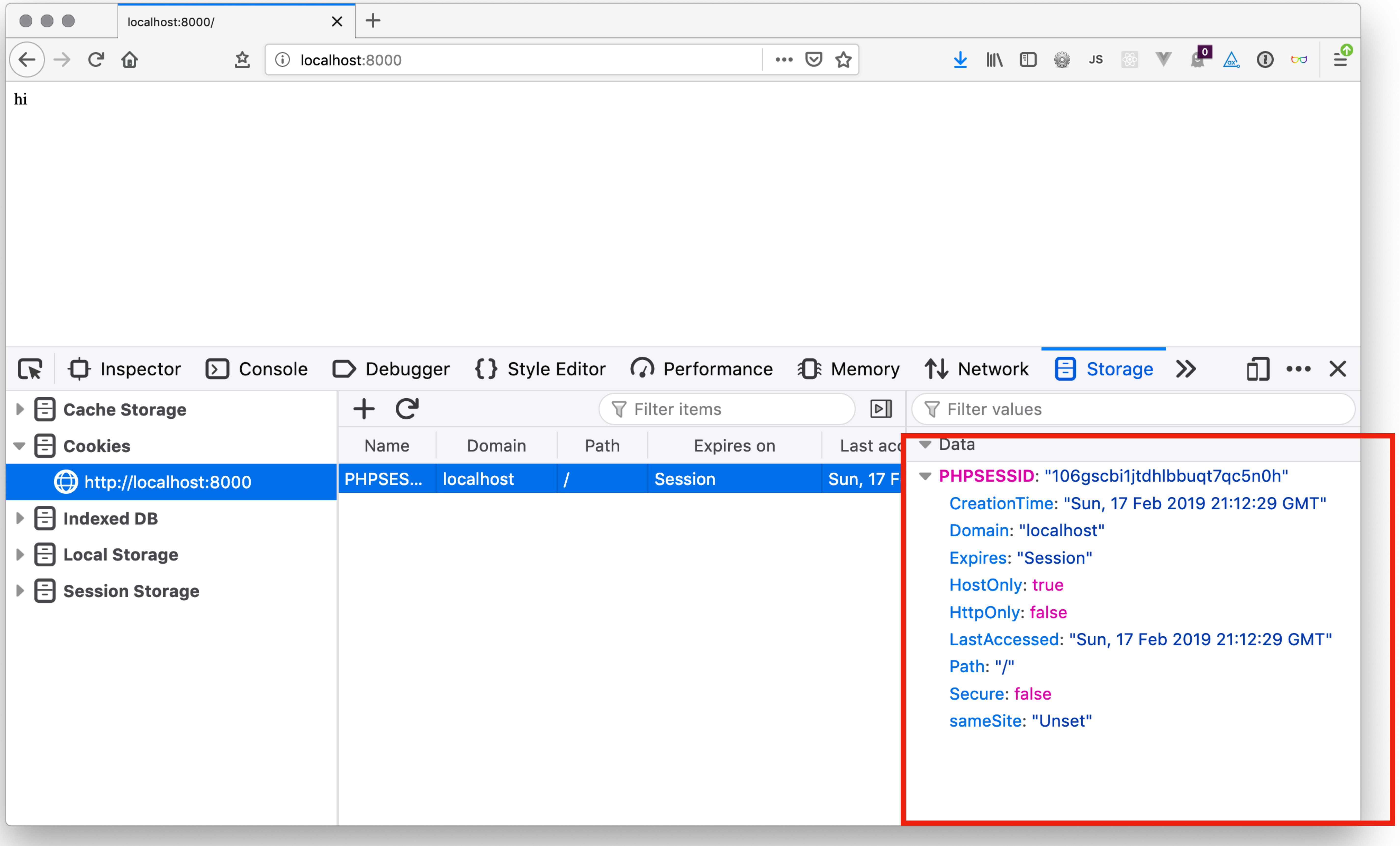


```php
<?php

session_start();
```

Session example: Step 2

You can see in your browser's cookie area if the sessions are enabled



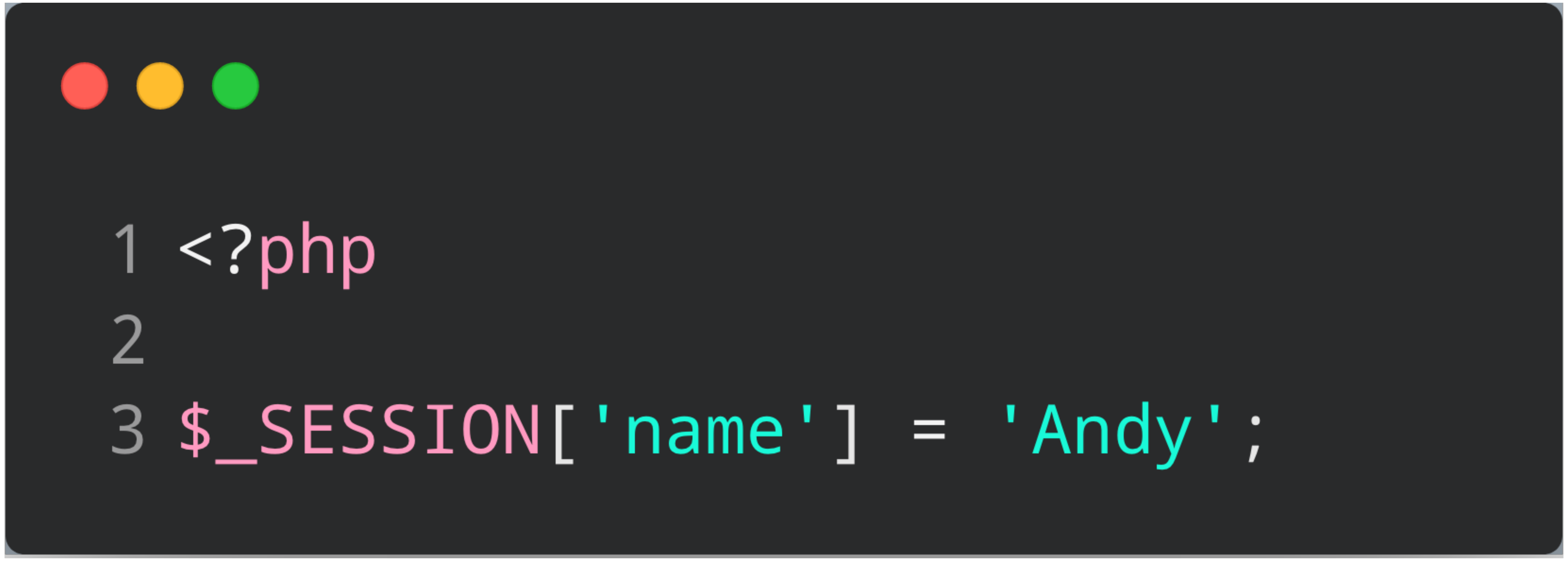
hi

Name	Domain	Path	Expires on	Last acc
PHPSES...	localhost	/	Session	Sun, 17 F

```
▼ Data
▼ PHPSESSID: "106gscbi1jtdhlbbuqt7qc5n0h"
    CreationTime: "Sun, 17 Feb 2019 21:12:29 GMT"
    Domain: "localhost"
    Expires: "Session"
    HostOnly: true
    HttpOnly: false
    LastAccessed: "Sun, 17 Feb 2019 21:12:29 GMT"
    Path: "/"
    Secure: false
    sameSite: "Unset"
```

Session example: Step 3

You interact with the session with the **$_SESSION** super global (just like **$_GET** and **$_POST**)

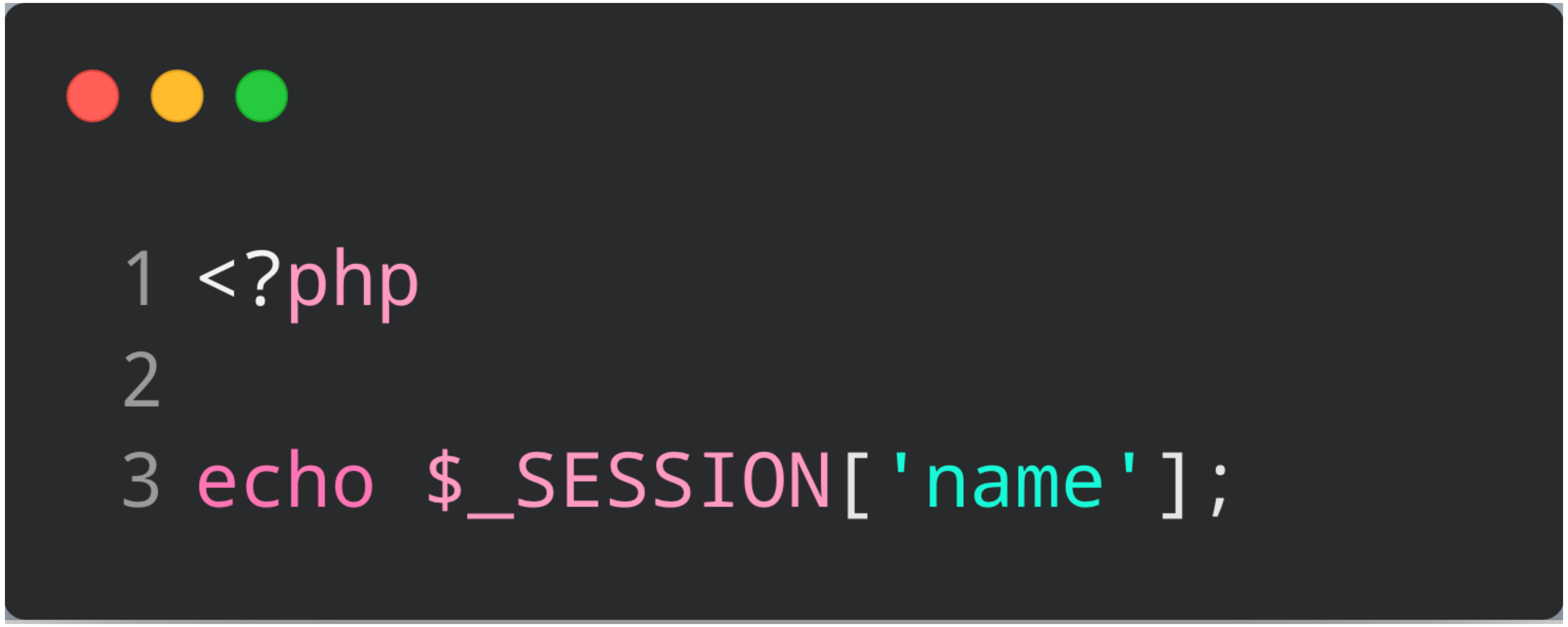


```php
<?php

$_SESSION['name'] = 'Andy';
```

Session example: Step 4

Getting the data is the same as getting **$_POST** data.



```php
<?php

echo $_SESSION['name'];
```

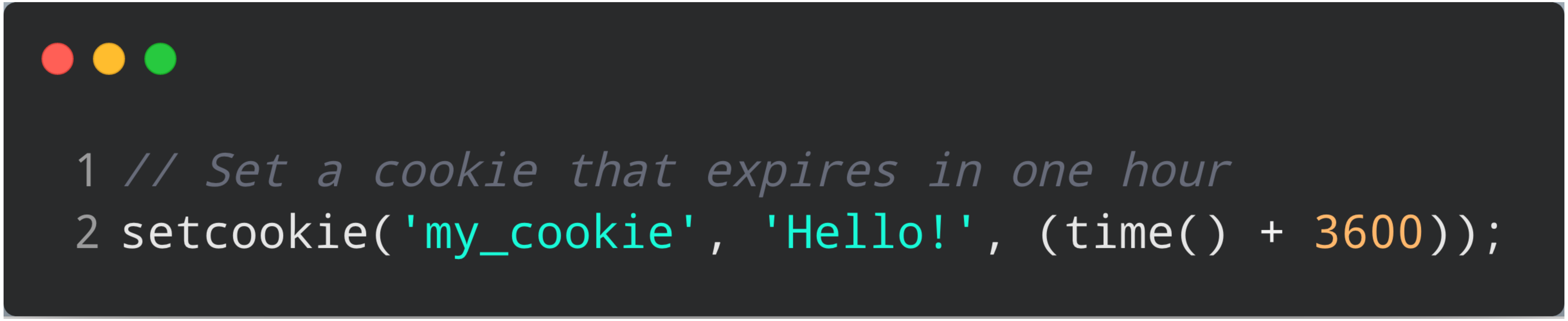
Sessions best practice

- Only use for data that is supposed to only be temporary
- Use it for sensitive information that only the server should know about
- Use it if cookies are disabled by the user

What is a Cookie?

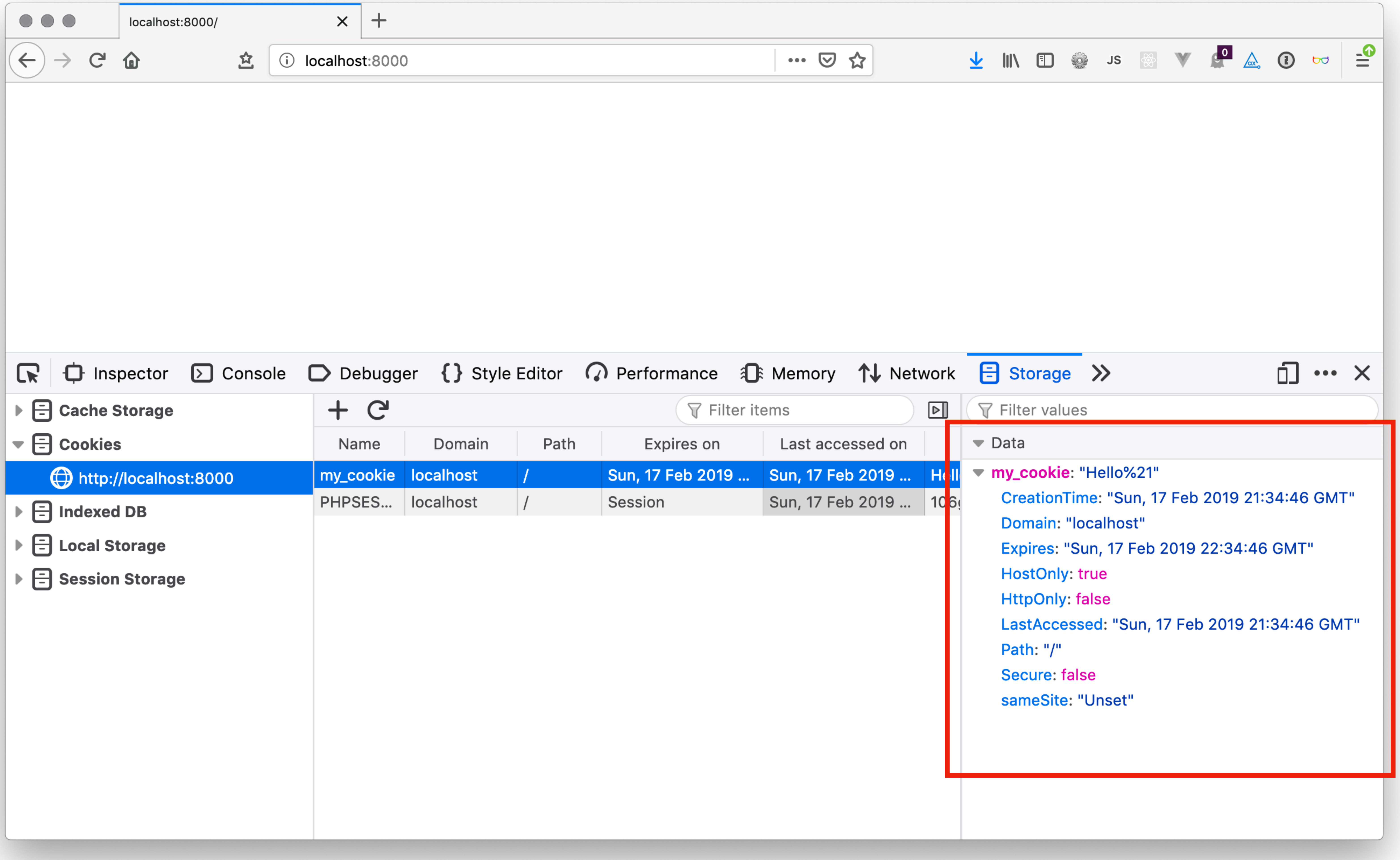
- A small piece of text data that is **persistent**
- You can set an expiration for it
- It lives on a user's computer, rather than the server
- It is sent back and forth between every single request to the server
- You can also send them with your ajax / fetch requests
- Very handy for storing user login state

Cookie example: Step 1



```
// Set a cookie that expires in one hour
setcookie('my_cookie', 'Hello!', (time() + 3600));
```

Cookie example: Step 2

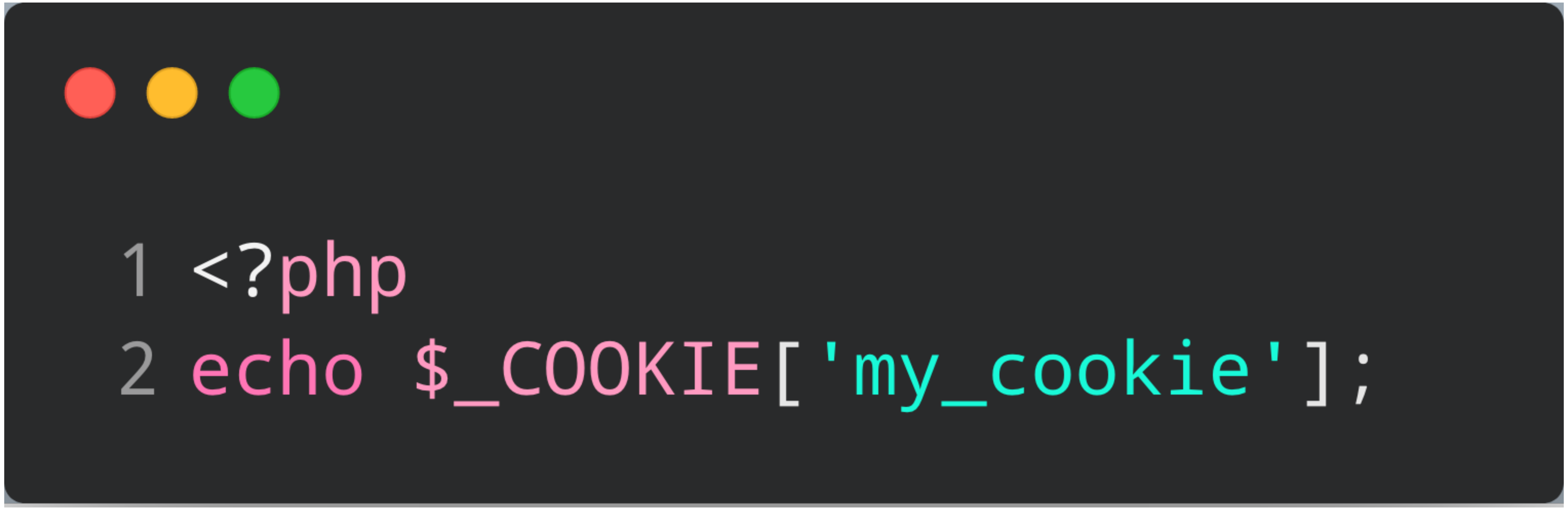


Name	Domain	Path	Expires on	Last accessed on	
my_cookie	localhost	/	Sun, 17 Feb 2019 ...	Sun, 17 Feb 2019 ...	Hell
PHPSES...	localhost	/	Session	Sun, 17 Feb 2019 ...	106

```
Data
my_cookie: "Hello%21"
  CreationTime: "Sun, 17 Feb 2019 21:34:46 GMT"
  Domain: "localhost"
  Expires: "Sun, 17 Feb 2019 22:34:46 GMT"
  HostOnly: true
  HttpOnly: false
  LastAccessed: "Sun, 17 Feb 2019 21:34:46 GMT"
  Path: "/"
  Secure: false
  sameSite: "Unset"
```

Cookie example: Step 3

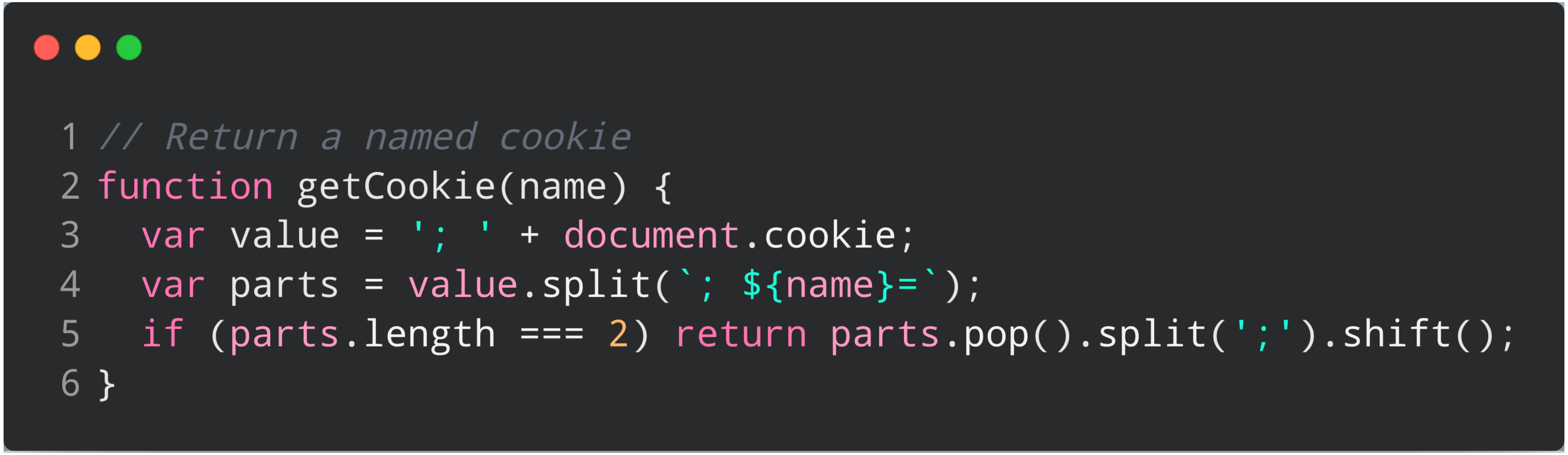
Getting the cookie value in PHP



```php
<?php
echo $_COOKIE['my_cookie'];
```

Cookie example: Step 4

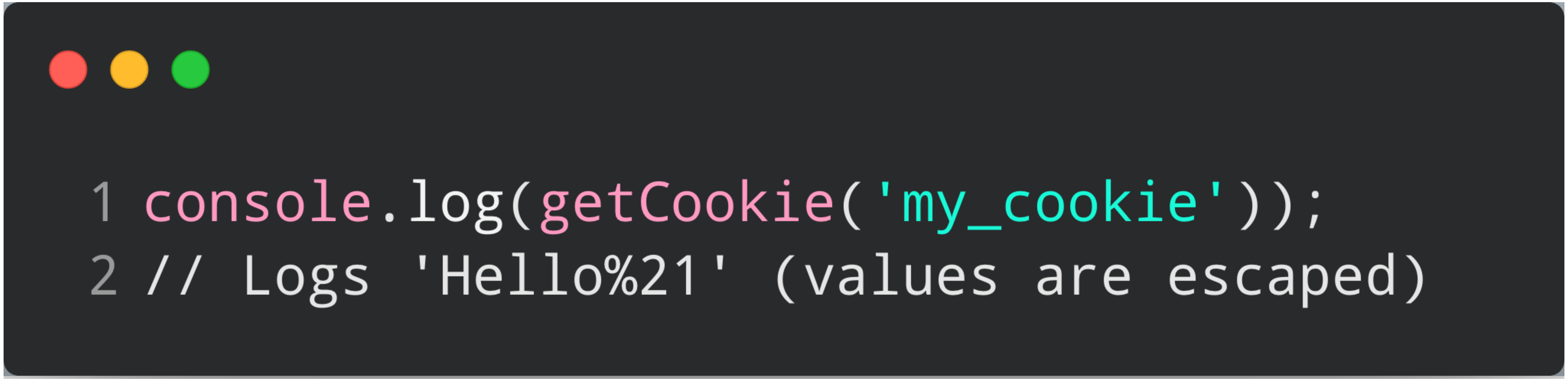
For JavaScript, we have to create a function to get our cookie by name



```javascript
// Return a named cookie
function getCookie(name) {
  var value = '; ' + document.cookie;
  var parts = value.split(`; ${name}=`);
  if (parts.length === 2) return parts.pop().split(';').shift();
}
```

Cookie example: Step 4

Then we can use it like this



```
console.log(getCookie('my_cookie'));
// Logs 'Hello%21' (values are escaped)
```

Cookies best practice

- Remember, they go back and forth with **every** request, so don’t store loads of data in them
- They can last as long as you like, so very good for persistent storage
- Anyone can access cookies, so make sure you don’t store sensitive data in them!

Demo

I'll show you a basic example of sessions and cookies that:

- Presents a login form **if** the user isn't logged in
- Present a logout link if they **are** logged in
- Show the logged in user's name from the cookie data

Source code: https://github.com/andybelldesign/CT4009-sessions-and-cookies

Recap

- We learned what a PHP session is
- We learned what a Cookie is
- We learned what the difference between them is
- We covered some best practice