

MAINTAINING "STATE"

**PERSISTENT STATE USING A "STATELESS"
PROTOCOL**

HTTP IS "STATELESS"



A "stateful"
protocol: FTP

Interactive sessions

User "authenticated"

Variables set on the server:
working directory; transfer
mode



Stateless protocol

No
"persistence"



Hypertext Transfer
Protocol (HTTP)

A message is
sent.

Data is
received.

WHY STATELESS?



MORE RELIABLE



CRASH-PROOF

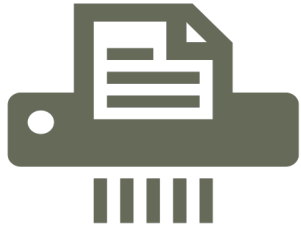


FASTER



EASIER TO WRITE
(AND MAINTAIN)

WAYS TO MAINTAIN STATE IN A STATELESS PROTOCOL



Cookies

Client-side state

Small file → user's web browser

Write user information to the file

Send the file back to the server



PHP session

Server-side state

PHP function: `session_start()` → PHP (on the server) remembers

Session variables: `$_SESSION`

Until: `session_unset()` and `session_destroy()` or timeout

USER SESSIONS USING PHP AND MYSQL

1. User → HTML form
 - username/password
2. Form captured → PHP script; compared (MySQL)
3. Match? PHP session is started; session variables set
4. Page to page, each asks: session variable?
 - If yes, show this webpage
 - If not, redirect

LOGIN SYSTEM REQUIREMENTS

Three pages, minimum:

- Login page
- Registration page
- Index page (and other pages that require authentication)

Features LOTS of error handling

- Login: no or incorrect username and/or password; account does not exist
- Registration: no or incorrect username and/or password; passwords don't match; account already exists

1. Go to a protected page -- should redirect to the login page
2. Click to go to the registration page
3. Create a new account -- should redirect to the login page (or direct to the first protected page)
4. Login -- should redirect to the first protected page
5. Logout -- should redirect to the login page (or an exit page)