

# Server-Side Scripting Intro

Web, Mobile and Security Frédéric Vlummens

### Agenda

- What is server-side scripting?
- HTTP
  - Request Response
  - Request methods
  - Headers
  - Status codes
- Network ports
- Webservers
- Introduction to PHP



2

### Let's start with an exercise

- We want to write a Facebook wall-like application...
- Let's try using what we've learned so far







### **Facebook wall: solution**

- Build a website using HTML, JavaScript and Local Storage
- Problem: the site is confined to one computer and one browser
- We need something that is available on all computers connected to the internet and all browsers
- Solution: server-side scripting ullet



### What is server-side scripting?

- Common technique used in web development
- Scripts (programs) run on the web server
- The scripts generate custom HTML (or even other files, such as CSS or images) that is sent to the browser
- The browser interprets the HTML it receives as if it were a static page
- This raises quite a few questions:
  - How does this transmission of data work?
  - What is a web server?
  - . . .



### Server-side scripting





### Server-side scripting





### Web server

- What is it? A machine? Software?
- Answer: both
- Machine
  - Hosts the files on its hard disks
  - Is connected to the internet
- Software
  - HTTP server
  - Processes incoming request
  - Sends response to client
- In this course: web server = software



### **Overview of popular web server software**

- Several popular products:
  - IIS
  - NGINX  $\bullet$
  - Apache ullet
  - Tomcat lacksquare
  - Glassfish  $\bullet$
  - ${\color{black}\bullet}$ ...
- Some are general purpose web servers, others more specific
- In this course: Apache running on Debian Linux
- A virtual machine with Debian Linux, Apache and PHP (incl. Laravel) has been prepared
- Will be installed and used in various labs for this module  $\bullet$





#### HTTP

### HTTP \_

# <u>HyperText</u> Transfer Protocol



### HTTP

- HyperText Transfer Protocol
- Hypertext: structured text using hyperlinks between nodes
- HTTP is an application protocol that transfers said text
- It is the standard used to access the web





#### **Request – Response**





### Viewing the Request and Response

• Using Mozilla Firefox Developer Tools

• Network tab

webtech.local/	× +					- 0	×
← → ♂ ଢ	(i) webtech.local		· 🛡 🏠	<b>Q</b> Search		III\ 🗉	D≡
2018-09-20 08:23:29							
□ Inspector S Console	🗅 Debugger 🛛 { } Style Editor 🖉 Performan	ce 🕼 Memory 🗦 No	etwork E	Storage		6	×
🗑 🗑 Filter URLs	II AII HTML CSS JS XHR	Fonts Images Media W	/S Other	Persist Logs Dis	able cache No	o throttling	♦ HAR ♦
Status Method F 200 GET /	Doi Cause Type Transferred	▶ Headers Cookies Perparent URL: http://weining Request method: GET Remote address: 192.168 Status code: 200 ⑦ Economic HTTP/1.1 ♥ Filter headers ♥ Response headers (204 B) ⑦ Connection: Keep-Alian	Params btech.loc 3.62.130: dit and Rese	Response Timings cal/ 80 end Raw headers	Stack Trace		
One request     19 B / 223 B f	transferred Finish: 19 ms DOMContentLoaded: 35	<ul> <li>Content-Length: 19</li> <li>Content-Type: text/htm</li> <li>Date: Thu, 20 Sep 201</li> </ul>	ml; charset= 8 06:23:29 G	UTF-8 SMT			
₩ ¥ Filter output							Logs X



### Viewing the Request and Response

- Using Mozilla Firefox Developer Tools
- Network tab

webtech.loca	1/	× +									_	· [		×
← → C' 6	i) (1)	webtech.local				•••	◙	☆	Q Se	earch		111	•	≡
2018-09-20 08:23	3:29													
🕞 🗘 Inspector	▷ Console	Debugger {}	Style Editor	r @ Performan	ce 🕼 Memo	ory 🔄 Ne	twork	8	Storage				•••	·×
💼 🛛 🗑 Filter URLs		11 4	AII HTML	CSS JS XHR	Fonts Images	Media W	S Otł	her	Persis	st Logs 🗌 Disable	cache No t	hrottling	а 🕈 Н	IAR 🖨
Status Meth	od F De	oi Cause	Туре	Transferred	Headers	Cookies	Para	ams	Respon	se Timings S	Stack Trace			
200 GET	/ 💋	ve document	html	223 B	Remote addre	ss: 192.168	.62.1	30:8	0					^
					Status code:	00 ? Ed	it and	Resen	d Raw	headers				
					Version: HTTP Request heade	/1.1 ers:				Response header	s			
					Host: webtech User-Agent: M Win64; x64; rv: Firefox/62.0 Accept: text/h /xhtml+xml,ap Accept-Langu Accept-Encod	local ozilla/5.0 (Wi 62.0) Gecko/2 ml,applicatio plication/xml age: en-US,en ing: gzip, defl	indows 201001 on I;q=0.9 i;q=0.5 late	s NT 10 01 9,*/*;q=	0.0; <sup>∧</sup>	HTTP/1.1 200 OK Date: Thu, 20 Sep Server: Apache/2. Content-Length: 1 Keep-Alive: timeo Connection: Keep Content-Type: tex	2018 06:23:29 4.25 (Debian) 19 put=5, max=10 p-Alive ct/html; chars <mark>e</mark> t	GMT 00 t=UTF-8		
C One request	19 B / 223 B transfe	rred Finish: 19	ms DOM	ContentLoaded: 35	Connection: ke	en-alive			~					<b>_</b>
🛍 💡 Filter out	put											Persi	st Logs	×
$\gg$														



#### **Request and Response**

#### Requests



#### Responses

#### HTTP/1.1 403 Forbidden

```
Server: Apache
Content-Type: text/html; charset=iso-8859-1
Date: Wed, 10 Aug 2016 09:23:25 GMT
Keep-Alive: timeout=5, max=1000
Connection: Keep-Alive
Age: 3464
Date: Wed, 10 Aug 2016 09:46:25 GMT
X-Cache-Info: caching
Content-Length: 220
```

<!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML 2.0//EN"> (more data)



### Using PuTTY or telnet to issue a request





### Using Postman to issue a request

🤣 Postman		
File Edit View Help		
+ New - Import R	tunner 🗜 🔹 🚼 My W	orkspace 🔻 🛃 Invite
Q Filter	http://webtech.local/  • •••	
History Collections	GET <ul> <li>http://webtech.local/</li> </ul>	
Postman Echo	Authorization Headers Body Pre-re-	equest Script Tests
23 requests	KEY	VALUE
	Key	Value
	Body Cookies Headers (6) Test Re	esults
	Pretty Raw Preview HTML <b>v</b>	
	<b>i</b> 1 2018-09-20 08:38:50	

			-	- 🗆	×
O	<b>K F</b>	<b>4</b> •		Upgrade	e 🔻
	No Environm	ent		•	*
	Params	Sen	d 🔻	Save	•
				Cookies	Code
	DESCRIPTION	•••	Bulk Edit	Presets	•
	Description				
	Status: 20	0 OK 1	Time: 23 ms	Size: 22	23 B
					Q
E	Build B	rowse	Q	<b></b>	. ?



17

### **Request method**

- Indicates the desired action for a given resource
- Made up of one noun or verb
- Also called *HTTP Verbs*
- Non-exhaustive list
  - GET
  - POST
  - PUT
  - DELETE
  - HEAD
  - OPTIONS



### GET

- The HTTP GET method requests a representation of the specified resource.  $\bullet$
- Requests using GET should only retrieve data.

GET /index.php HTTP/1.1 Host: webtech.local **User-Agent: Mozilla Firefox** 



### POST

- The HTTP POST method sends data to the server.  $\bullet$
- The type of the request body is indicated by header Content-Type.
- Usually used to submit form data.  $\bullet$

**POST /process.php HTTP/1.1** Host: webtech.local **Content-Type:** application/x-www-form-urlencoded **Content-Length: 21** 

name=Frederic&team=TI



### PUT

- PUT creates a new resource or replaces a representation of target resource with request  $\bullet$ payload.
- PUT is **idempotent**: calling it multiple times has no side effect (such as an order being submitted)  $\bullet$ twice, which can happen with POST)

PUT /aboutme.html HTTP/1.1 Host: webtech.local **Content-type: text/html Content-length: 17** 

<h1>About me</h1>



### DELETE

• Deletes the specified resource

DELETE /old.html HTTP/1.1



22



- Used to request headers returned should resource be requested using GET.
- Example use case: request the content length (size) and the decide whether to go ahead with download or not.

HEAD /largefile.zip HTTP/1.1

HEAD • http://webted	ch.local/largefile.zip		
Body Cookies Headers (9	) Test Results		
Accept-Ranges → bytes			
Connection $\rightarrow$ Keep-Alive			
<b>Content-Length</b> $\rightarrow$ 20000000			
<b>Content-Type</b> → application/zip			
Date → Thu, 20 Sep 2018 13:13:53 GMT			
ETag → "bebc200-5764d4a886a2d"			
<b>Keep-Alive</b> $\rightarrow$ timeout=5, max=100			
Last-Modified → Thu, 20 Sep 2018	8 13:13:38 GMT		
Server → Apache/2.4.25 (Debian)			



### **OPTIONS**

Used to determine which request methods are supported by server or specific resource.  $\bullet$ 

**OPTIONS /largefile.zip HTTP/1.1** 

<b>OPTIONS</b> • http://webtech.local/largefile.zip				
Body Cookies Headers (7) Test Results				
Allow → GET,HEAD,POST,OPTIONS,HEAD,HEAD				
Connection $\rightarrow$ Keep-Alive				
Content-Length $\rightarrow 0$				
<b>Content-Type</b> → application/zip				
Date → Thu, 20 Sep 2018 13:15:34 GMT				
<b>Keep-Alive</b> $\rightarrow$ timeout=5, max=100				
Server → Apache/2.4.25 (Debian)				



### **HTTP** headers

- Allow client and server to pass additional information with request or response.
- Request header format:
   header-name: value
- Example:



```
-12656974
(more data)
```

#### request or response. (case insensitive)



### **HTTP** headers

- Allow client and server to pass additional information with request or response.
- Response header format:

header-name: value

Example:



(body)

12/02/2020 Web, Mobile and Security – Server-Side Scripting: Intro

# (case insensitive)

Response headers



### **HTTP status codes**

- Sent back to client  $\bullet$
- Indicates whether request was completed successfully or not
- 3 digit codes
- Divided into 5 categories:
  - Informational (1##) ullet
  - Successful (2##) lacksquare
  - Redirects (3##) lacksquare
  - Client errors (4##) lacksquare
  - Server errors (5##) lacksquare



27

#### Which status codes do you know?





### **Common HTTP status codes**

- 200 OK  $\bullet$
- 201 CREATED
- **400 BAD REQUEST**  $\bullet$
- 401 UNAUTHORIZED  $\bullet$
- **403 FORBIDDEN**
- 404 NOT FOUND  $\bullet$
- **500 INTERNAL SERVER ERROR**  $\bullet$

For more information, see <a href="https://developer.mozilla.org/en-US/docs/Web/HTTP/Status">https://developer.mozilla.org/en-US/docs/Web/HTTP/Status</a>  $\bullet$ 



#### **HTTP status code**

GET /index.php HTTP/1.1 Host: webtech.local User-Agent: MyBrowser

> HTTP/1.1 200 OK Date: Thu, 20 Sep 2018 13:34:01 GMT Server: Apache/2.4.25 (Debian) Content-Length: 19 Keep-Alive: timeout=5, max=98 Connection: Keep-Alive Content-Type: text/html; charset=UTF-8



### **Network ports**

- An **IP address** identifies a **computer**  $\bullet$
- A **network port** identifies the **application/service** running on the computer.  $\bullet$





#### Some common network ports

Port	Description
21	File Transfer Protocol (FTP)
22	Secure Shell (SSH)
25	Simple Mail Transfer Protocol (SMT
80	HyperText Transfer Protocol (HTTP)
110	Post Office Protocol (POP3)
443	HTTP Secure (HTTPS)





#### **Back to server-side scripting...**

- Now we know what a web server is and how a browser and server communicate.
- Server side scripting = common technique used in web development.
- Scripts (programs) run on the web server.
- The scripts generate custom HTML (or even other files, such as CSS or images) that gets sent to the browser.
- The browser interprets the received HTML as if it where a static HTML page.



# **Introducing PHP**

- PHP = PHP HyperText Processor (recursive acronym)  ${\color{black}\bullet}$
- Development started in 1994
- Various versions have since been introduced
- Today we are at PHP 7.3
- Interested in the history? <u>https://en.wikipedia.org/wiki/PHP#History</u>





# A first look at a plain PHP script: index.php





### Requesting the page and executing the script

Hello world		$\times$ +	
← → ⊂	i debian-wm	ns.local/hello-world/	•••
Hello world	5	S De Passer E ×   S Pizzeria ×   S Hello world × → C ③ view-source:localhost	S view-so
	2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17	<pre><!DOCTYPE html>     <html lang="en"> <head></head></html></pre>	∙width, ini 'ie=edge">





### Requesting the page and executing the script

Hello world	× +
$\leftrightarrow \rightarrow \mathbf{C}$ (i) debian	-wms.local/hello-world/
Hello world	S De Passer $\mathbb{B} \times  $ S Pizzeria $\times  $ Hello world $\times$ S view-so $\leftarrow \rightarrow \mathbb{C}$ View-source:localhost
The sum is 8	<pre> 2 <!DOCTYPE html>     3 <html lang="en"> 4 <head> 5</head></html></pre>
	The PHP code stays on All that gets sent to th





#### Requesting the page and executing the script





## **About PHP syntax**

- Different from Python, more or less comparable to Java and JavaScript
- Excellent reference material: <u>http://php.net/manual/en/langref.php</u>
- Basis control structures:

...

```
if – else – elseif
for
while
```

http://php.net/manual/en/language.control-structures.php





### Key aspects of PHP syntax

- Variables start with a \$ sign  $\Leftrightarrow$  Java, JavaScript and Python
- Functions are defined using the **function** keyword (idem as JavaScript)

```
<?php
    function calc_avg($a, $b) {
        return ($a + $b) / 2;
    }
?>
The average of <?php echo $x; ?> and <?php echo $y;</p>
?> is: <?php echo calc_avg($x, $y); ?>
```





40

### Key aspects of PHP syntax

- Arrays are initialized using square brackets []
- You can loop over an array using **foreach**
- Here, array\_push is an example of one of the many built-in functions of PHP

```
<?php
   $fruits = [];
   array_push($fruits, "apple");
   array_push($fruits, "pear");
   array_push($fruits, "banana");
?>
<?php foreach ($fruits as $fruit) { ?>
      <?php echo $fruit; ?>
  <?php } ?>
```



#### **Questions?**



