0.2 - Server Side Scripting 101

During the first quarter of this year, the concept of *server side scripting* is discussed briefly. Understanding this concept is very important for this course, because the PHP platform is entirely based on this. Before we move on to the next episodes of the series, a short refresh might be in place.

Web pages

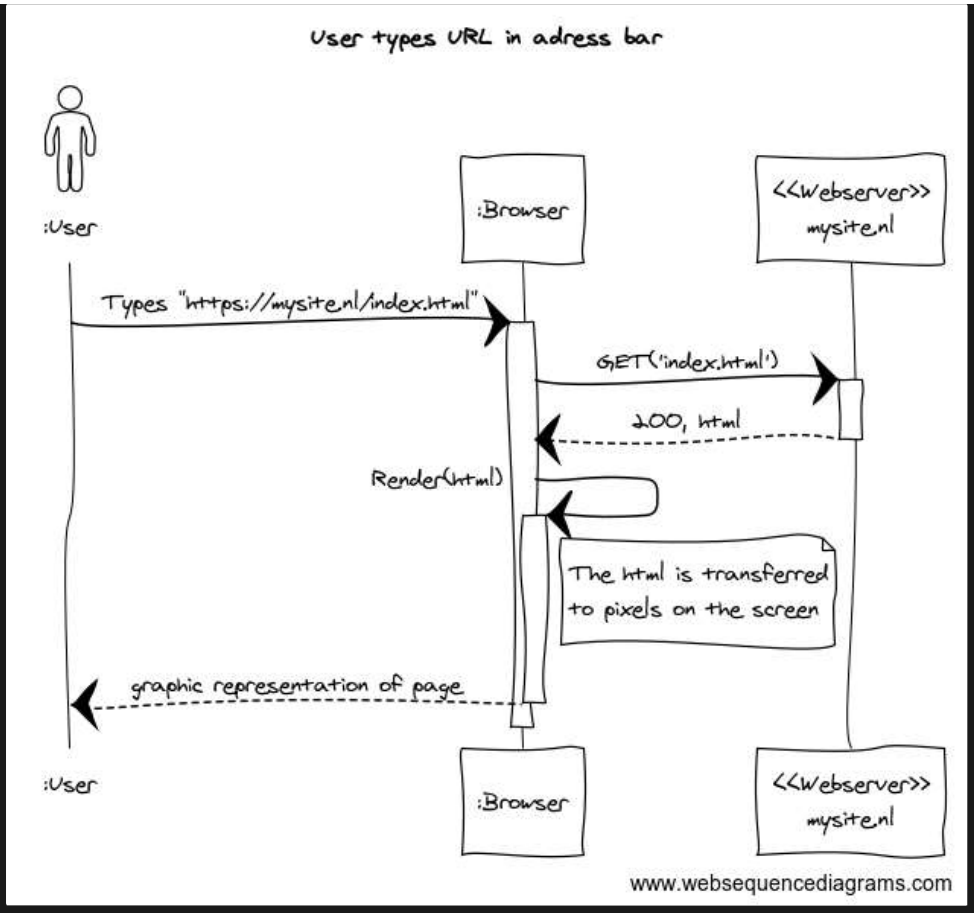
As you already know, a web site consists of one or more web pages. A page can be loaded by sending an HTTP request to a web server. Such a request might look like (most relevant parts are highlighted):

```
GET /index.html HTTP/1.1 User-Agent: Mozilla/4.0 (compatible; MSIE5.01; Windows NT) Host: mysite.nl Accept-Language: en-us Accept-Encoding: gzip, deflate Connection: Keep-Alive
```

This server tries to formulate an HTTP response for this request. If everything goes well, the response will have a status code of `200 OK` and will contain the HTML code of the requested page, like:

```
HTTP/1.1 200 OK Date: Mon, 27 Jul 2009 12:28:53 GMT Server: Apache/2.2.14 (Win32) Last-Modified: Wed, 22 Jul 2009 19:15:56 GMT Content-Length: 88 Content-Type: text/html Connection: Closed <html> <body> <h1>Hello, World! </h1> </body> </html>
```

The browser interprets the HTML code and renders it to pixels on the screen to produce a graphic representation of the page. The entire process can be modeled in a *sequence diagram* like so:

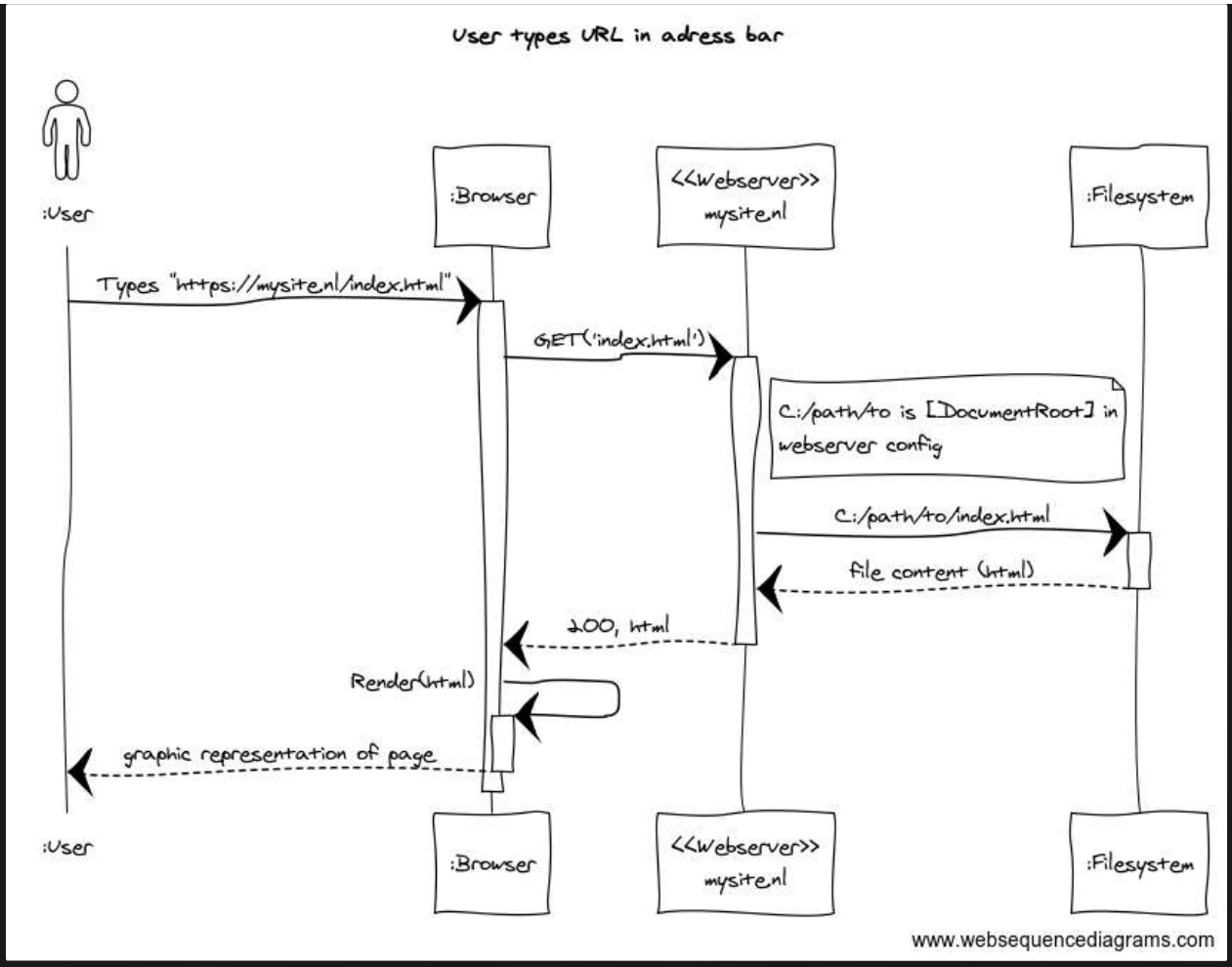


💡 Sequence diagrams will be explained during the coming weeks. However, they are fairly intuitive to read. The time axis runs from top to bottom, but is not to scale

Important to know here is that, because browser and server are usually on different computers located anywhere on earth (or even outside, as the ISS may also have webservers), **the only way to communicate between these objects are by means of HTTP messages.**

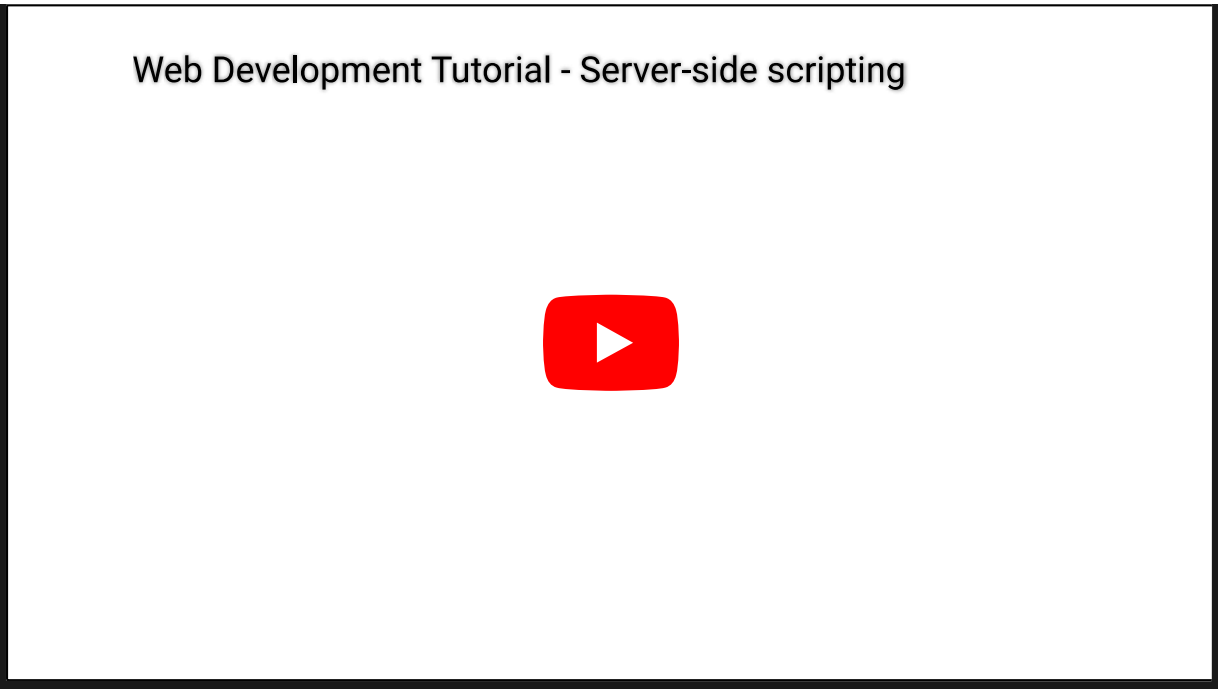
Static pages

In the first quarter you already have built your first website. This site consists of static pages. All the HTML is located in files in a folder called the *documentroot*. Users can request a specific file with an HTTP request. In a sequence diagram this can be modeled like this:



Dynamically generated HTML: server side scripting

In Server Side Scripting the HTML is not static, but will be dynamically generated for each HTTP response. The following video explains this:



The video shows that, when you want to apply server side scripting, you need special software applications: webservers. For the PHP platform, this is usually a module (like a plugin) in common web server applications like *Apache* or *Nginx*.

> 💡 Technically, PHP is a separate installation and you need to configure the web server to let it collaborate with that installation. However, your it is already built in your XAMPP installation.

A PHP example

We will discuss a simple example that demonstrates the very basics of dynamically generated HTML with PHP. Let's say we have a website with a file called `index.php` in the documentroot. The content of the file is like so:

```
<html> <head> <meta charset="utf-8"> <title>Welcome to PHP</title> </head>
<body> <h1><?php echo "Hello PHP world!"; ?></h1> </body> </html>
```

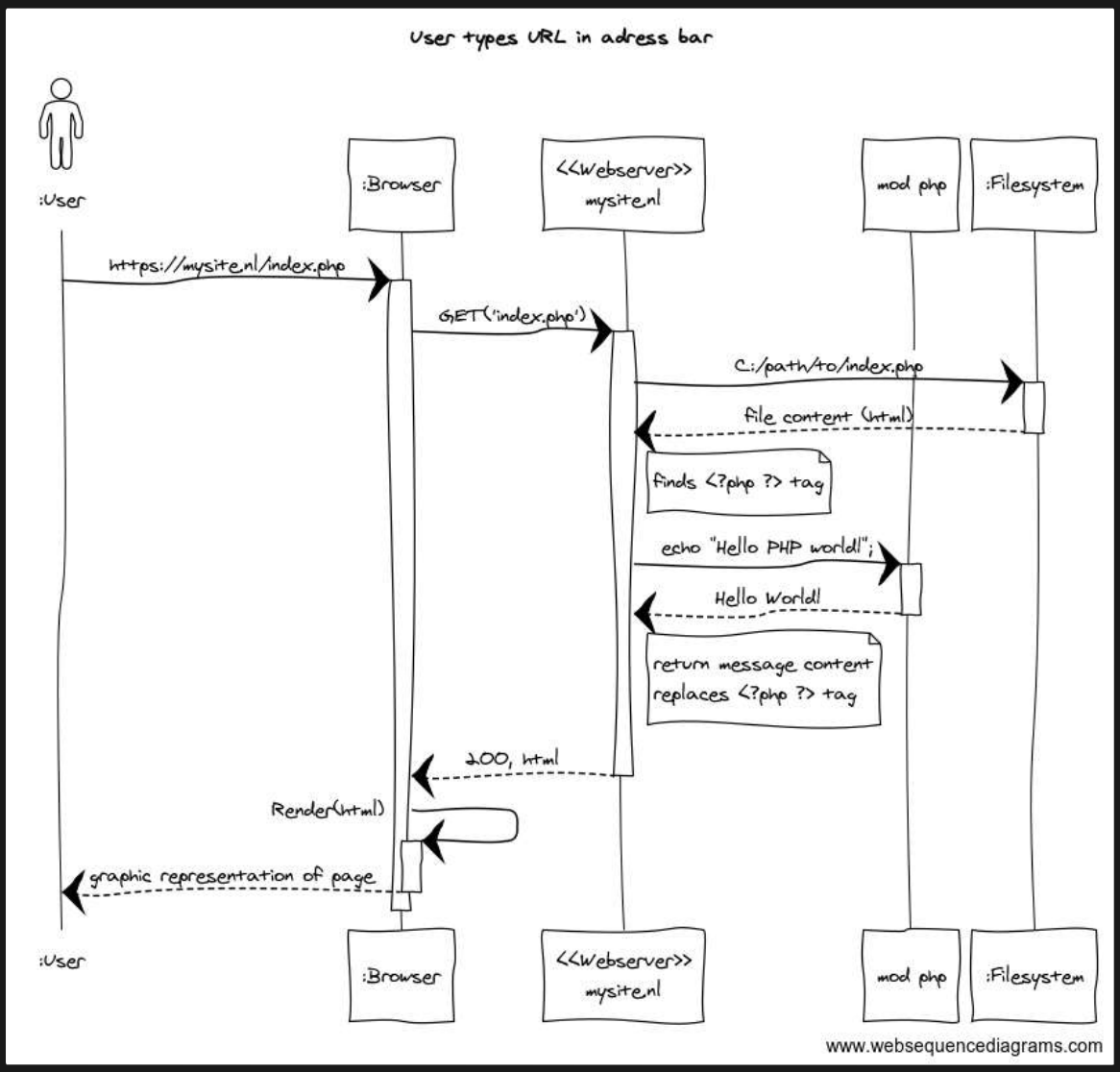
Content of `index.php`

When a user/browser requests this file, all the code between the `<?php` and `?>` tags are executed by the PHP module on the server. The result of this execution will be included in that specific location in the HTTP response. The HTML in the response will look like:

```
<html> <head> <meta charset="utf-8"> <title>Welcome to PHP</title> </head>
<body> <h1>Hello PHP world!</h1> </body> </html>
```

HTML response body

The following sequence diagram shows this:



Exercise 0.3 Reproduce this file in the documentroot of your XAMPP installation, and try to load it in your browser. Check the response in your browser. The page source must look like the example above, without the `<?php` tag.

Frequently made mistakes

If the result of the exercise is not as it should be, this might be your mistake:

- The file name MUST end with `.php`. If you name it anything else, the webserver doesn't know that it contains php code
- The file must be placed in (a subfolder of) the documentroot of your webserver. You may need to find out what the documentroot is in your webserver configuration
- You loaded the file directly from the file system, instead of via the webserver. Check the address bar of the browser. It should start with `http://` or `https://` instead of `file://`. Most browsers hide the prefix, but if you click or double click on the address bar, this prefix should be visible
- The URL must point to your webserver address. This should be something like