COMP4021
Internet Computing

Basic PHP

David Rossiter & Gibson Lam

注意不是“ Hypertext Preprocessor ”的缩写。这种将名称放到定义中的缩写方法被称作递归/嵌套缩写

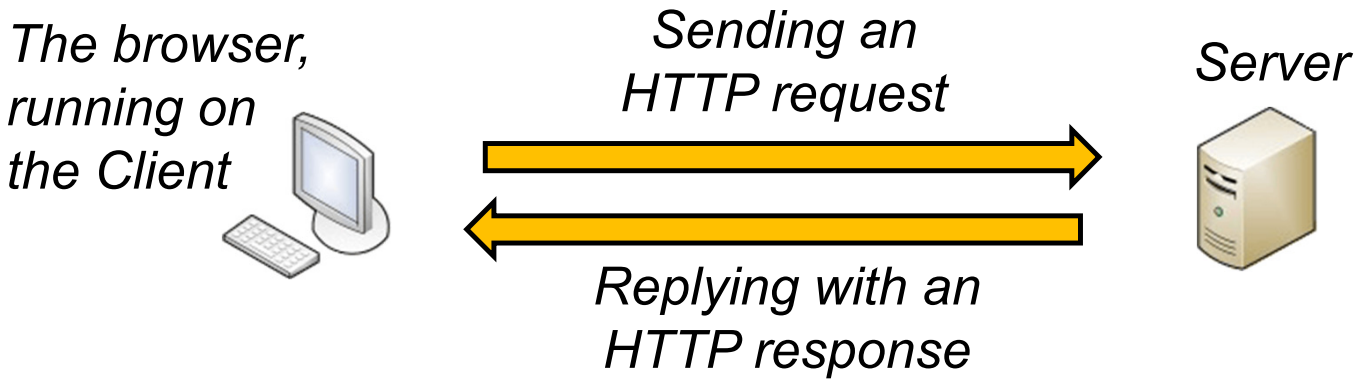
PHP

- PHP stands for
  PHP: **H**ypertext **P**reprocessor
- It is a very popular scripting language used in web servers
- PHP is used by 78% of all websites which use a server-side programming language (March 2020)
- PHP is not hard to learn

Data from https://w3techs.com/technologies/details/pl-php/all/all

Basic Web Server Operation

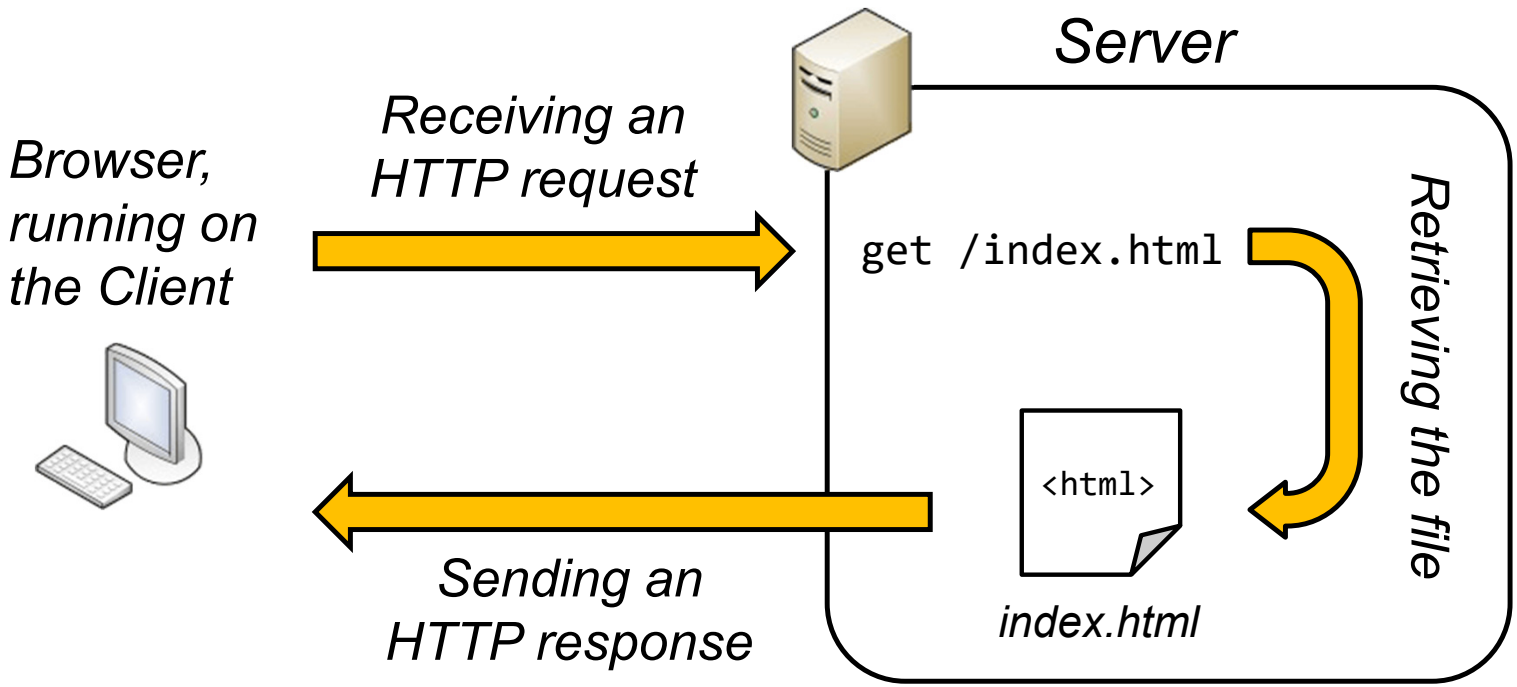
- When the browser wants to load a page, it sends a HTTP request to the server



- The file is given to the browser using a HTTP response

Requesting A Simple Document

- Here is an example of what happens when the browser requests a simple document (no PHP):



- User enters *localhost/index.html* in the browser
- Request for index.html goes to the server
- This is an example of what happens when a simple HTML page (with no PHP) is requested

HTML code on server

```
<!DOCTYPE html>
<html>
<head>
    <title>Simple Web Page</title>
</head>
<body>
    This is my web page
</body>
</html>
```

- *Exactly the same*

HTML code sent to browser

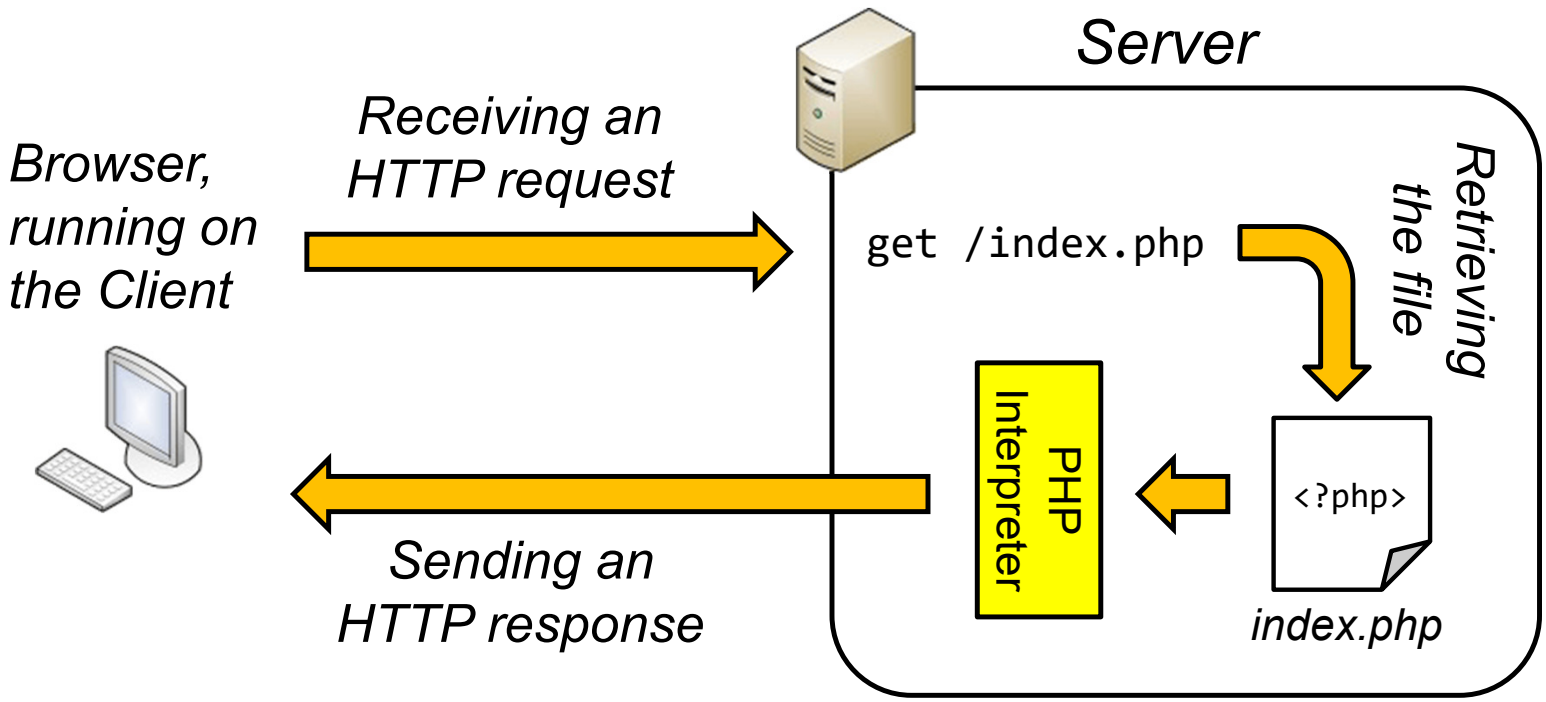
```
<!DOCTYPE html>
<html>
<head>
    <title>Simple Web Page</title>
</head>
<body>
    This is my web page
</body>
</html>
```

HTML code shown in browser

notice here: without PHP, it shows localhost! That means execute the html in the local computer.

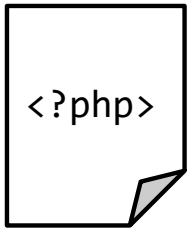
Requesting A PHP Document

- Here is an example of what happens when the browser requests a PHP document:



PHP Files and Web Servers

- PHP files are typically named with an extension of `.php`
- A web server has to be configured for PHP files so it knows when to use the PHP interpreter



index.php

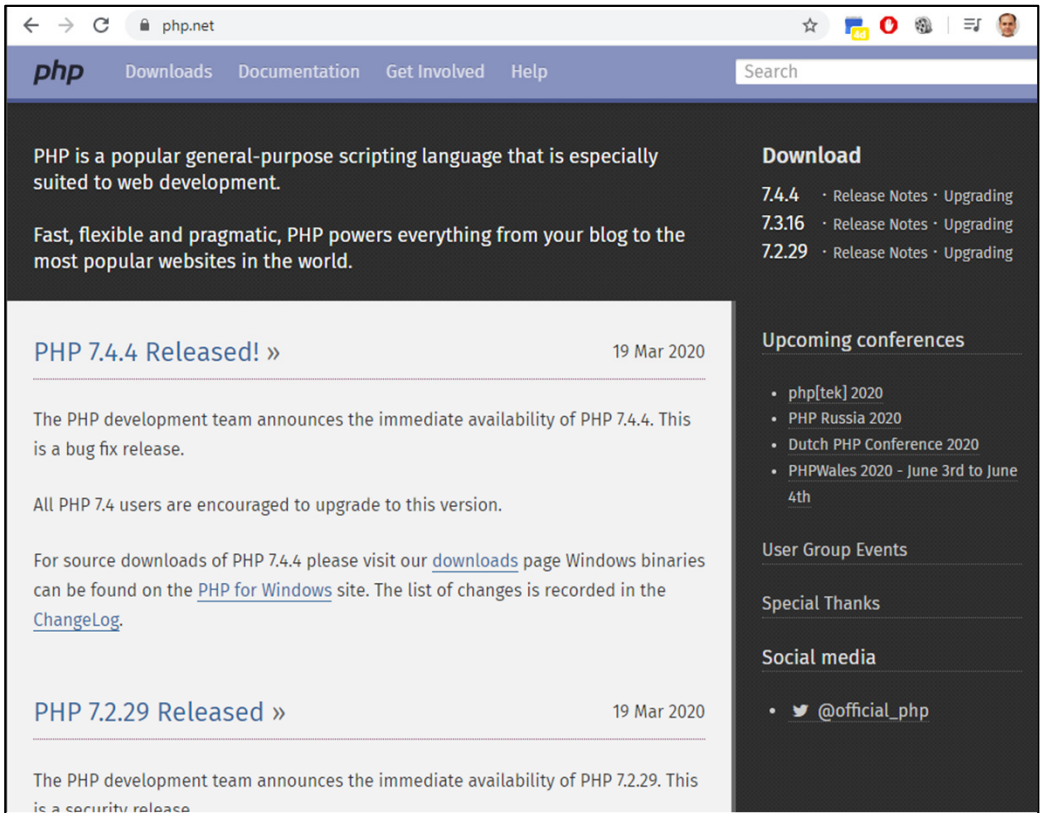
you can also put php code inside .html file

Downloading PHP

- You can download a PHP package from the official website:

  `https://php.net`

- The latest main version number is PHP 7



Setting Up PHP

- After extracting the PHP package, you then set up your web server to use PHP
- The configuration is different for different web servers
- If you are using a package (see next slide) you probably don’t have to do much configuration – possibly you don’t have to do any

Ready-made Packages

- If you don't have anything already installed, you can use special packages to get lots of things installed at the same time
- One popular package is XAMPP, which includes some main server components
  - an Apache web server
  - a database
  - PHP
  - sometimes other things as well

e.g. sql database, pearl language

It's a quick way to get a web server together with php

Two Quick Ways to Use PHP

- To quickly use PHP, we can:
  - Use the Personal Homepage service provided by ITSC:

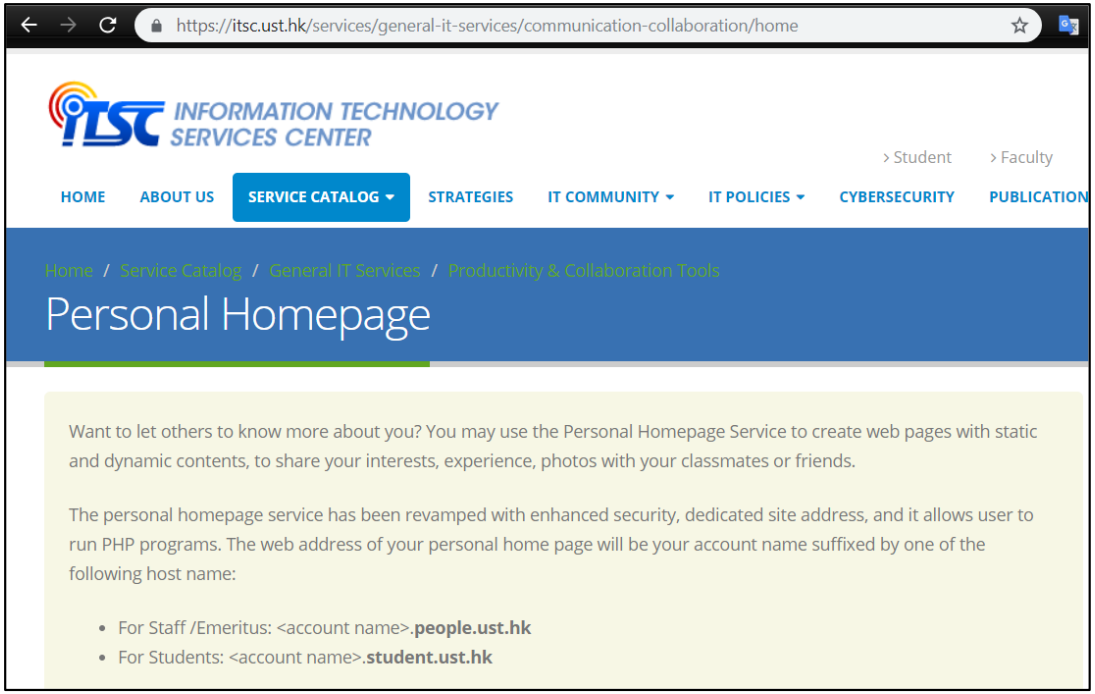
    `https://itsc.ust.hk/services/general-it-services/communication-collaboration/home`

    or

  - Use the built-in web server provided by PHP
    - It's OK for a temporary web server, but it can't do many things a proper web server like Apache can do

Using the ITSC Personal Homepage

- You need to enable your ITSC Personal Homepage from this ITSC page:



- Once enabled, you can access your homepage using `http://`*your ITSC account*`.student.ust.hk`

Testing the ITSC System

- For example, you can create a file `test.php` with the following line of code, and put it into the `public_html` folder of the ITSC server:

```
<?php echo "Hello"; ?>
```

  – If everything is set up correctly, you can access the file on the ITSC server like this:

  `http://`*your ITSC account*`.student.ust.hk/test.php`

  – This command `echo` simply sends text to the browser

Code on server

```
<?php echo "Hello"; ?>
```

Code sent to browser

```
Hello
```

source code

Hello

HTML code shown in browser

Using the PHP Web Server on Your Machine

- PHP has a built-in web server for quick testing of PHP code
- You can start the server using this command:

```
php -S localhost:80
```

- *localhost* basically means 'the computer you are in front of'

  - The above command runs a web server in the local machine, behind port 80
  - After starting the server, you can then use the URL `http://localhost` to see what your PHP code produces

Running the PHP Web Server on Your Machine

- Using the command on the last page, the *document root* folder of the server (the top level directory) is the folder in which you start the PHP web server i.e. `C:\php`

```
Command Prompt - php -S localhost:80
Microsoft Windows [Version 10.0.18362.720]
(c) 2019 Microsoft Corporation. All rights reserved.

C:\Users\dross>cd C:\php

C:\php>php -S localhost:80
[Sun Apr 12 16:58:02 2020] PHP 7.4.4 Development Server (http://localhost:80) started
```

- You can change it e.g. `php -S localhost:80 -t fun/`
- Then the document root will be `C:\php\fun`

PHP Scripts

- PHP scripts are enclosed within a special tag
  `<?php ... ?>`
- You can write as many lines of code as you want inside a tag
- When the PHP file is sent to the PHP interpreter, the interpreter runs the PHP code inside the tags and replaces the tags with the output of the code

Typical Use of PHP

- It's very common to 'mix' HTML code with pieces of PHP inside it, like this:

```
<!DOCTYPE html>
<html>
<head>
    <title>Simple PHP</title>
</head>
<body>
    <?php echo "This message is from PHP!"; ?>
</body>
</html>
```

PHP code

```
<!DOCTYPE html>
<html>
<head>
    <title>Simple PHP</title>
</head>
<body>
    <?php echo "This message is from PHP!"; ?>
</body>
</html>
```

Code on server

在server sending code to browser的时候PHP interpreter就会把php都变成html code

```
<!DOCTYPE html>
<html>
<head>
    <title>Simple PHP</title>
</head>
<body>
    This message is from PHP!
</body>
</html>
```

HTML code sent to browser

The browser never sees the PHP source code. The browser only sees the interpreted html code.

This message is from PHP!

HTML code shown in browser

Short Echo Tags

- It is very common to use the `echo` command to put things into the HTML page
- PHP gives you a quick way to do echo:

注意message依旧要用引号！

`<?=` ***message to show*** `?>`

then you don't have to type so much e.g.:

```
...
<body>
    <?= "This message is from PHP!" ?>
</body>
...
```

完整的是：<? php echo "This message is from PHP!" ; ? >

- This is equivalent to the code shown on the last slide

Easy to Write Code

- PHP code is fairly similar to JavaScript code
- PHP variables always start with a `$` sign e.g.:

```
$x = 10; // create variable x
```

- Semi-colons are more important in PHP than JavaScript:

```
$x = $start + 1;

myFunction();
```

不止是在create variable的时候，在使用variable的时候也要加 $

Dumping Variable Content

- The `var_dump` function outputs the content of a variable as well as its type
- This is useful when you need to debug something, e.g.:

Code on server

```
<?php
$age = 20;
var_dump($age);
?>
```

Sent to browser

```
int(20)
```

Shown in browser

int(20)

content as well as the type!

Variable Scope in PHP

- You have the typical variable scopes in PHP
  - Local variables,
    i.e. variables created inside any function
  - Global variables,
    i.e. variable created outside all functions
- To access global variables from within functions, you need to use the `global` keyword, as shown in the next slide

A Simple Example of Scope in PHP

```
<?php
$msg1 = "I am from outside!";
$msg2 = "I am from outside!";

function myFunction() {
    global $msg1;

    $msg1 = "I am from inside!";
    $msg2 = "I am from inside!";
}

myFunction();

echo $msg1;
echo "<br>";
echo $msg2;
?>
```

$msg1直接改了global variable

$msg2相当于在function里面新建了一个local variable，跟global那个无关

changed !

not changed !

I am from inside!
I am from outside!

Operators and Strings

- PHP has all the common operators that you can use in JavaScript e.g. `+ - / *` etc
- For strings, you use the period (`.`) operator to concatenate strings together, like this:

```
$message = "happy" . " birthday";
```

- To get the length of a string, you need to use the `strlen` function, i.e.:

```
strlen($message); // this returns 14
```

Flow Controls

- You can use all the common flow control statements in PHP such as `if…else…` , `switch…` , `while…` and `for…`
- You also have `foreach`
- `foreach` is used for traversing an array, like this:

```
$veg = ["apple", "banana", "coconut"];

foreach ($veg as $value) {
    echo $value . "<br>";
}
```

break: go to the next line

concatenate!

apple
banana
coconut

Functions

- Functions are created the same way as in JavaScript
- You can create named functions and anonymous functions, like you can in JavaScript

```
<?php
function showMessage() {
    echo "This is from PHP!";
}

showMessage();
?>
```

Arrays

- Creating indexed arrays and associative arrays is easy
- Here is an example indexed array:

```
$funny = ["OMG", "IMHO", "88"];
echo $funny[0]; // this will output OMG
```

- Here is an associative array:

Like a dictionary structure, where one item is mapped to another item.

```
$abbreviations = [ "OMG"  => "Oh My Goodness!",
                   "IMHO" => "In My Humble Opinion",
                   "88"   => "Bye bye"];
echo $abbreviations["OMG"];   // output Oh My Goodness!
```

Things That You Can Do With Arrays

- You can read the number of things in an array using the `count` function, e.g.

```
echo count($abbreviations); // will display 3
```

- A `foreach` loop can read both keys and values from associative arrays, like this:

OMG: **Oh My Goodness!**
IMHO: **In My Humble Opinion**
88: **Bye bye**

key value

```
foreach ($abbreviations as $shortForm => $fullMeaning) {
    echo "<code>{$shortForm}</code>:
          <b>{$fullMeaning}</b><br>";
}
```

Variables can be put inside a string too

without using " . "

```
<!DOCTYPE html>
<html>
<head>
    <title>Using Foreach Loop</title>
</head>
<body>
    <?php
    $abbreviations = ["OMG" => "Oh My Goodness!",
                      "IMHO" => "In My Humble Opinion",
                      "88" => "Bye bye"];

    foreach ($abbreviations as $shortForm => $fullMeaning) {
        echo "<code>{$shortForm}</code>:
              <b>{$fullMeaning}</b><br>";
    }
    ?>
</body>
</html>
```

Code on server

Code on server, see last slide

HTML code sent to browser

```
...
<code>OMG</code>:<b>Oh My Goodness!</b><br>
<code>IMHO</code>:<b>In My Humble Opinion</b><br>
<code>88</code>:<b>Bye bye</b><br>
...
```

OMG: **Oh My Goodness!**
IMHO: **In My Humble Opinion**
88: **Bye bye**

HTML code shown in browser