

CSE 485 - Công nghệ Web

Giới thiệu môn học

dungkt@tlu.edu.vn

Nội dung

- 1) Thông tin giảng viên
- 2) Mô tả khóa học
- 3) Mục tiêu và kết quả
- 4) Đánh giá môn học
- 5) Tài liệu tham khảo
- 6) Một số qui định của môn học

1. Thông tin giảng viên

- Kiều Tuấn Dũng
- Phòng làm việc: Bộ môn HTTT, Tầng 2, Nhà C1
- Email: dungkt@tlu.edu.vn
 - Đề nghị khi trao đổi qua email, gửi mail với cấu trúc TIÊU ĐỀ THƯ [CSE485] Vấn đề cài đặt Xampp (Ví dụ)
 - Thầy sẽ không check các email thực hiện không đúng yêu cầu.
- Mobile: 0868.600.513
 - Nếu gọi không được, vui lòng để lại tin nhắn, thầy sẽ gọi lại khi phù hợp.

2. Mô tả khóa học



- Trong khóa học này, chúng ta sẽ trải qua các kiến thức bao gồm:
 - Quản lý source code và làm việc nhóm với GIT.
 - Phát triển Web phía Front End: Thiết kế website với HTML, CSS, JavaScript , JQuery, Responsive, Bootstrap, dàn layout từ Photoshop
 - Phát triển Web phía Back End: Lập trình Web động với PHP & MySQL.
 - Xây dựng CSDL, triển khai một ứng dụng Web hoàn chỉnh (gồm Front End & Back End).
- Lớp học sẽ bao gồm các Buổi học Lý thuyết + Thực hành (Đề nghị sinh viên chủ động mang Laptop để quá trình học có thể diễn ra liên tục, cả trên phòng học Lý thuyết + Thực hành)
- Khóa học bao gồm 2 bài tập + 1 dự án cuối môn học [Nhóm 2 người].
- Khóa học sử dụng 2 kênh giao tiếp chính với sinh viên: Facebook Group: để thông báo nhanh và các trao đổi khác & Website: hoccongngghethongtin.com để lấy tài liệu và nộp bài tập.

3. Mục tiêu và kết quả môn học

- Mục tiêu chung: Giới thiệu cho sinh viên các khái niệm, kỹ thuật, hệ thống và thực hành hiệu quả để phát triển các ứng dụng trên nền web. Mục tiêu cụ thể bao gồm:
 - Hiểu biết về ngôn ngữ đánh dấu, ngôn ngữ định kiểu trang web, về mô hình lập trình phía máy khách và phía máy chủ cho các ứng dụng web.
 - Phát triển các kỹ năng cần thiết cần thiết cho lập trình web.
 - Trải nghiệm và tìm hiểu về sự phát triển của các ứng dụng web.
- Cuối khóa học, sinh viên có thể:
 - Giải thích các khái niệm và công nghệ cơ bản trong World Wide Web.
 - Thiết kế và triển khai các giao diện web bằng HTML, CSS và JavaScript.
 - Phát triển các ứng dụng web với các công nghệ phía máy chủ.
 - Sử dụng cơ sở dữ liệu trong các ứng dụng web.
 - Hiểu và đưa vào thực hành các nguyên tắc cơ bản của bảo mật web.



4. Đánh giá môn học

Hình thức	Số lần	Mô tả	Thời gian	Trọng số
1. ĐIỂM QUÁ TRÌNH				40%
Chuyên cần	1	Đi học đầy đủ Thái độ học tập tích cực	- Xuyên suốt môn học	20%
Bài tập #1	1	Bài tập nhỏ trên lớp (Phân Thiết kế giao diện sử dụng HTML và CSS, Javascript)	- Tuần 3	40%
Bài tập #2	1	Bài tập nhỏ trên lớp (Phân Thiết kế giao diện sử dụng jQuery và Bootstrap)	- Tuần 5	40%
2. ĐIỂM THI KẾT THÚC HỌC PHẦN				60%

Bài tập để đánh giá sẽ xem xét dựa trên sự kết hợp của 3 tiêu chí ASK:

1. Thái độ
2. Kỹ năng
3. Kiến thức

5. Tài liệu tham khảo

- Đề cương chi tiết, Slide Bài giảng & Bài tập: được cung cấp tại <http://hoccongnghethongtin.com>
- Video Bài giảng (Youtube)
- Giáo trình [**Khuyến khích đọc Giáo trình tiếng Anh**]:
 - Zak Ruvalcaba, Anne Boehm, **Murach's HTML5 and CSS3, 4th Edition**, Mike Murach & Associates, 2018
 - Randy Connolly, **Fundamentals of Web Development**, Pearson, 2018
 - Joel Murach and Ray Harris, **Murach's PHP and MySQL (3rd Edition)**, Mike Murach & Associates, 2017
 - Adrian W. West, **Practical PHP 7, MySQL 8, and MariaDB Website Databases: A Simplified Approach to Developing Database-Driven Websites 2nd ed. Edition**, Apress, 2018
- Website tham khảo:
 - <https://www.w3schools.com/>
 - <https://www.php.net/>
- **Copyright:** Nội dung được tham khảo từ devpro, có chỉnh sửa và cập nhật.

6. Một số qui định của môn học

- Sinh viên đi học muộn quá **5 phút** so với giờ vào tiết học, vui lòng chờ đến tiết sau vào lớp để không làm ảnh hưởng đến diễn tiến của lớp học.
- Sinh viên được nghỉ học theo qui định, qui chế đào tạo tối đa **20%** thời lượng môn học (dành cho các trường hợp ốm đau, gia đình có việc đột xuất)
 - Trừ trường hợp đặc biệt được xem xét cụ thể nếu nghỉ quá số buổi học (năm viện có giấy xác nhận, gia đình có việc hiếu ...)
 - Sinh viên không xin phép nghỉ với các lý do: em có việc bận, gia đình em có việc ...
- Sinh viên làm và nộp bài tập tự giác, đầy đủ đúng qui định.
- Sinh viên cài đặt đủ phần mềm theo yêu cầu của môn học và mang theo Laptop khi đến lớp.

CSE485 – Công nghệ Web

Bài 01: Giới thiệu

dungkt@tlu.edu.vn

Nội dung



- 1) Giới thiệu
- 2) Tại sao chọn PHP
- 3) Công cụ cần có
- 4) Kiến thức nền tảng
- 5) Cách học hiệu quả

1. Giới thiệu

Web Development Roadmap



- **Tham khảo chi tiết:**
 - <https://www.w3schools.com/whatis/>

Web tĩnh và Web động



- **Web tĩnh**, nghĩa là khi trang Web được tạo ra và lưu lại dưới định dạng HTML, nội dung và cấu trúc của trang sẽ không thay đổi cho tới khi mã nguồn được chỉnh sửa.

HTML	CSS	JavaScript
HTML	CSS	JavaScript
HTTP / XHR	CSS Responsive	ECMAScript 5

Web tĩnh và Web động



- **Web động**, nghĩa là khi trang Web được tạo ra và lưu lại dưới định dạng của ngôn ngữ lập trình Web động (ví dụ: **PHP**, ASP.net), nội dung và cấu trúc của trang sẽ thay đổi theo cách nó được lập trình và tương tác với người dùng + Cơ sở dữ liệu (**MySQL**, SQL Server...)

Fullstack

SQL

PHP

ASP

Python

Fullstack JS

SQL

Node.js

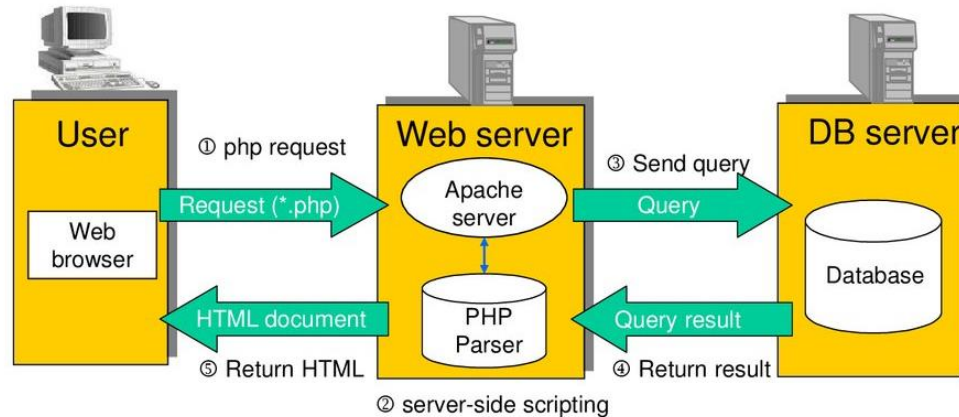
MySQL

Mongo.db

Web server

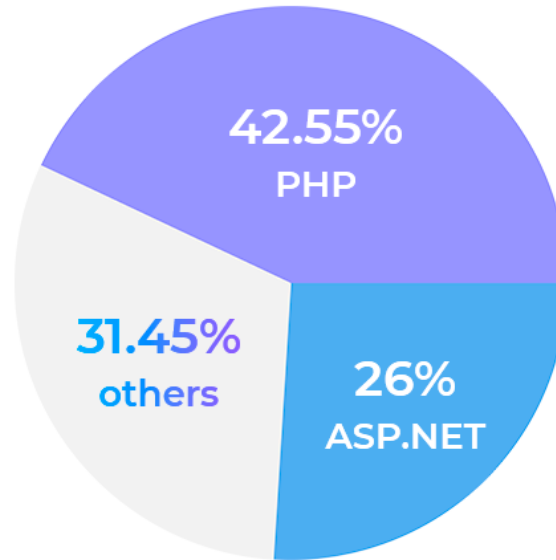


- Thường là máy tính có cấu hình cao được kết nối ra ngoài Internet, hoạt động 24/24
- Cài đặt hệ điều hành máy chủ (Ubuntu Server, CentOS, Windows Server)
- Cài đặt các ứng dụng máy chủ Web (Apache, Nginx, IIS ...)
- Chứa mã nguồn ứng dụng Web/Website



2. Tại sao chọn PHP?

Thị phần ngôn ngữ Web



source: datanyze.com

PHP vs ASP.net



Jan, 2002

26%

60.000+
developers

1.669.017

Microsoft


First release


Market share


Community


Number
of domains


Supported by



Nov, 1997

42.55%

5 million+
developers

2.731.340

Community &
Zend technologies



CMS



- Top 10 Web CMS

1. [WordPress](#) (27+ Million Live Websites)
2. [Wix](#) (3.8+ Million Live Websites)
3. [Squarespace](#) (1.9+ Million Live Websites)
4. [Joomla!](#) (1.8+ Million Live Websites)
5. [Shopify](#) (1.1+ Million Live Websites)
6. [Drupal](#) (630+ Thousand Live Websites)
7. [Blogger](#) (430+ Thousand Live Websites)
8. [Prestashop](#) (285+ Thousand Live Websites)
9. [Magento](#) (265+ Thousand Live Websites)
10. [Bitrix](#) (223+ Thousand Live Websites)

Đặc trưng của PHP



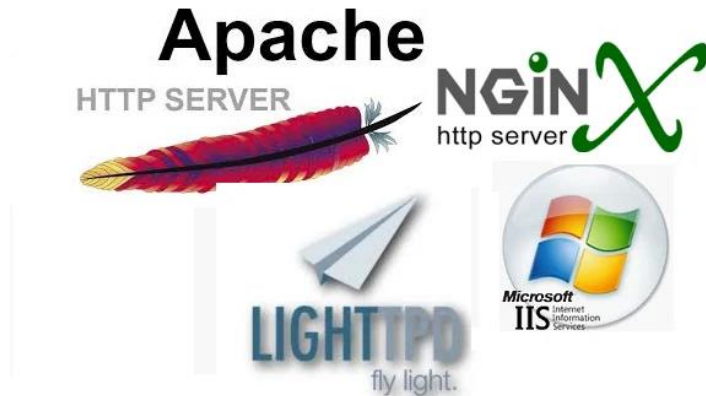
- Mã nguồn mở, chuyên dụng cho môi trường Web
- Tính cộng đồng, nhiều tài liệu hướng dẫn và hỗ trợ
- Thư viện phong phú
- Hỗ trợ nhiều Hệ CSDL
- Lập trình Hướng đối tượng
- Tính bảo mật
- Khả năng mở rộng không giới hạn

3. Công cụ cần có

Web Server



- Cài đặt thủ công: **A**pache, **P**HP, **M**ySQL + Tools hỗ trợ (MySQL workbench, phpMyAdmin ...)
- Cài đặt mì ăn liền:
 - Wamp
 - Xampp
 - Vertrigo
 - OpenServer
 - ...



Text Editor/IDE



- Text Editor: Visual Studio Code + extension/add-on
 - Sublime Text, Atom, Vim ...
- IDE: PHP Designer, PHPStorm, ZendStudio



Visual Studio Code

Trình duyệt Web








- **Firefox**

- Chrome

- Edge

- ...

BEST BROWSERS FOR WEB DEVELOPMENT		PRICE	DEFAULT SEARCH ENGINE	LICENSE	
92		Firefox Developer Edition	0	-	-
89		Mozilla Firefox	FREE	Google	OpenSource (MPL-2.0)
85		Google Chrome	FREE	Google	Proprietary
82		Chromium	-	Google	OpenSource (BSD)
--		Brave	-	Qwant	OpenSource (mainly MPL 2.0)

4. Kiến thức nền

Trình duyệt Web



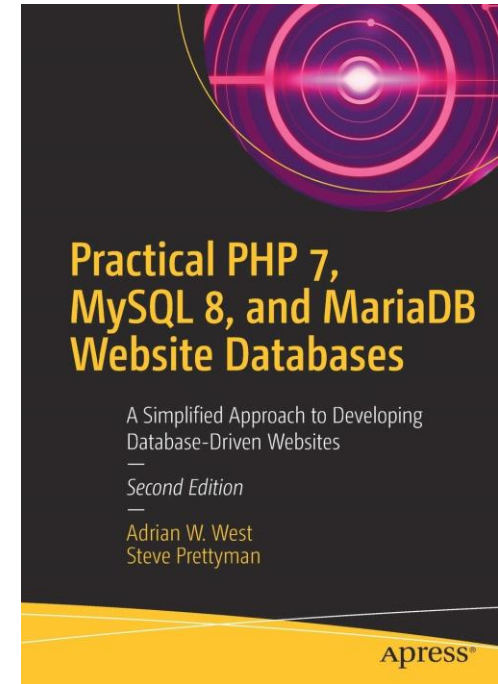
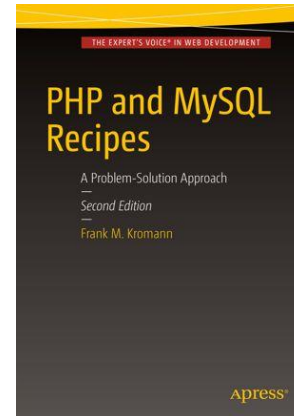
- **HTML**
- **CSS**
- **Javascript**
- -----
- **jQuery**
- **Angular**
- ...

5. Cách học hiệu quả

Học bằng sách



- Chọn 1 cuốn sách làm cẩm nang tra cứu lý thuyết (khi cần)
- Chọn 1 cuốn sách chứa Ví dụ thực hành theo một dự án thực tế
 - Có mã nguồn ví dụ kèm theo càng tốt (để học cách họ code)



Học bằng Video



- Xem Video tới đâu, thực hành Code ngay tại đó.
 - Tránh ngồi xem cả chục Video ... vừa uống trà, vừa gật gù ... hay quá, dễ hiểu quá.
 - Một bên mở video, một bên mở Trình gõ code



Tài liệu tham khảo



- <https://www.php.net>
- <https://www.w3schools.com/>

CSE485 – Công nghệ Web

Bài 02: PHP Căn bản 01

dungkt@tlu.edu.vn

Nội dung



- 1) Bắt đầu với PHP
- 2) Tạo trang PHP đầu tiên
- 3) Biến, Hằng, Hàm
- 4) Toán tử
- 5) Các cấu trúc điều khiển

1. Bắt đầu với PHP

.html vs .php



- **Code PHP** sẽ được đặt trong cặp thẻ: `<?php` [code php viết ở đây] `?>`
- Đoạn **code PHP** có thể được đặt xen kẽ với các thẻ HTML hoặc sử dụng tách biệt nhưng:
 - Cần được đặt trong một trang **.php** để server có thể tìm và thông dịch.
 - Nếu đặt trong một trang **.html**, các đoạn **code PHP** sẽ bị bỏ qua.

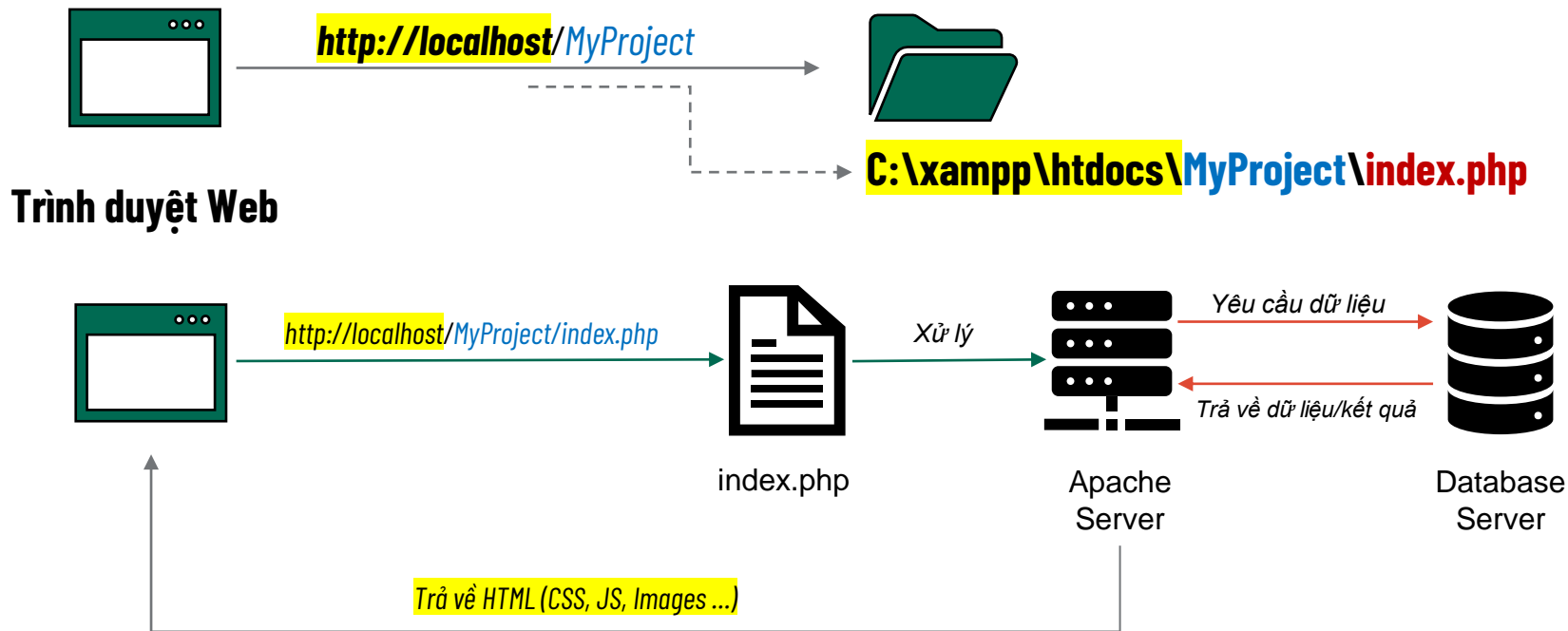
```
< > index.php x
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4     <meta charset="UTF-8">
5     <title>Nhúng PHP trong HTML</title>
6 </head>
7 <body>
8     <h1>
9         <?php echo "Hello world"; ?>
10    </h1>
11 </body>
12 </html>
```

Code php đặt ở đâu?



- Mỗi dự án website/ứng dụng web nên có **một thư mục riêng** chứa mã nguồn và các tài nguyên sử dụng cho trang Web, ví dụ: **MyProject**
 - Thư mục **MyProject** cần được đặt vào trong thư mục chỉ định của Web Server.
 - Tùy từng loại Web Server mà thư mục Web chỉ định mặc định là khác nhau:
 - **Xampp** sử dụng thư mục **htdocs**
 - **Wamp** sử dụng thư mục **www**
 - **Linux** sử dụng mặc định **/var/www/html**
 - Có thể chỉnh sửa thư mục Web mặc định trong tệp tin httpd.conf của Apache
- Truy cập trang Web thông qua địa chỉ: **http:localhost/MyProject** hoặc **http://localhost:Port/MyProject** (cần có tham số Port nếu đổi cổng Apache trong quá trình cài đặt)

Nguyên lý hoạt động của PHP



2. Tạo trang PHP đầu tiên

phpinfo()

- Xem thông tin Web Server

```
<?php

// Show all information, defaults to INFO_ALL
phpinfo();

// Show just the module information.
// phpinfo(8) yields identical results.
phpinfo(INFO_MODULES);

?>
```



phpinfo() options

Name (constant)	Value	Description
INFO_GENERAL	1	The configuration line, <i>php.ini</i> location, build date, Web Server, System and more.
INFO_CREDITS	2	PHP Credits. See also phpcredits() .
INFO_CONFIGURATION	4	Current Local and Master values for PHP directives. See also ini_get() .
INFO_MODULES	8	Loaded modules and their respective settings. See also get_loaded_extensions() .
INFO_ENVIRONMENT	16	Environment Variable information that's also available in \$__ENV .
INFO_VARIABLES	32	Shows all predefined variables from EGPCS (Environment, GET, POST, Cookie, Server).
INFO_LICENSE	64	PHP License information. See also the » license FAQ .
INFO_ALL	-1	Shows all of the above.

Thông tin Web Server



PHP Version 7.3.10

php

System	Linux uk-m247-web134.main-hosting.eu 3.10.0-962.3.2.lve1.5.25.6.el7.x86_64 #1 SMP Thu Apr 18 06:40:26 EDT 2019 x86_64
Build Date	Oct 8 2019 15:10:39
Configure Command	<code>./configure '--disable-dependency-tracking' '--prefix=/usr' '--exec-prefix=/usr' '--bindir=/usr/bin' '--sbindir=/usr/sbin' '--sysconfdir=/etc' '--datadir=/usr/share' '--includedir=/usr/include' '--libdir=/usr/lib64' '--libexecdir=/usr/libexec' '--sharedstatedir=/var/lib' '--mandir=/usr/share/man' '--infodir=/usr/share/info' '--build=x86_64-redhat-linux-gnu' '--host=x86_64-redhat-linux-gnu' '--target=x86_64-redhat-linux-gnu' '--program-prefix=' '--prefix=/opt/alt/php73' '--exec-prefix=/opt/alt/php73' '--bindir=/opt/alt/php73/usr/bin' '--sbindir=/opt/alt/php73/usr/sbin' '--sysconfdir=/opt/alt/php73/etc' '--datadir=/opt/alt/php73/usr/share' '--includedir=/opt/alt/php73/usr/include' '--libdir=/opt/alt/php73/usr/lib64' '--libexecdir=/opt/alt/php73/usr/libexec' '--localstatedir=/var' '--with-curl=/opt/alt/curlssl/usr' '--sharedstatedir=/usr/com' '--mandir=/opt/alt/php73/usr/share/man' '--infodir=/opt/alt/php73/usr/share/info' '--cache-file=/.config.cache' '--with-libdir=lib64' '--with-config-file-path=/opt/alt/php73/etc' '--with-config-file-scan-dir=/opt/alt/php73/ink/conf' '--disable-debug' '--enable-calendar' '--enable-exif' '--enable-ftp' '--enable-huge-code-pages' '--enable-opcache' '--enable-opcache-file' '--enable-shmop' '--enable-unicode' '--enable-xml' '--with-bz2' '--with-freetype-dir=/usr' '--with-gettext' '--with-gmp' '--with-iconv' '--with-jpeg-dir=/usr' '--with-kerberos' '--with-layout=GNU' '--with-libxml-dir=/opt/alt/libxml2/usr' '--with-mhash' '--with-openssl-dir=/opt/alt/openssl' '--with-openssl=/opt/alt/openssl' '--with-password-argon2=/usr' '--with-pcre-jit' '--with-pcre-regex' '--with-pic' '--with-png-dir=/usr' '--with-readline' '--with-t1lib=/opt/alt/t1lib/usr' '--with-webp-dir=/usr' '--with-xpm-dir=/usr' '--with-zlib' '--with-zlib-dir=/usr' '--without-gdbm' '--without-pear' '--with-litespeed' '--enable-pcntl' '--without-mysqli' '--disable-mbstring' '--disable-bcmath' '--disable-dba' '--disable-dom' '--disable-fileinfo' '--disable-json' '--disable-intl' '--disable-pdo' '--disable-phar' '--disable-posix' '--disable-soap' '--disable-sockets' '--disable-sysvsem' '--disable-sysvshm' '--disable-sysvmsg' '--disable-wddx' '--disable-xmlreader' '--disable-xmlwriter' '--disable-zip' '--without-gd' '--without-imap' '--without-xmlrpc' '--without-xsl' '--without-ldap' '--without-pgsql' '--without-snmpp' '--without-sodium' '--without-tidy' '--without-enchanted' '--without-recode' '--without-mssql' '--without-pspell' '--without-unixODBC' 'build_alias=x86_64-redhat-linux-gnu' 'host_alias=x86_64-redhat-linux-gnu' 'target_alias=x86_64-redhat-linux-gnu' 'CFLAGS=-O2 -g -pipe -Wall -Wp,-D_FORTIFY_SOURCE=2 -fexceptions -fstack-protector-strong -param=ssp-buffer-size=4 -grecord-gcc-switches -m64 -mtune=generic -fno-strict-aliasing -Wno-pointer-sign' 'CXXFLAGS=-O2 -g -pipe -Wall -Wp,-D_FORTIFY_SOURCE=2 -fexceptions -fstack-protector-strong -param=ssp-buffer-size=4 -grecord-gcc-switches -m64 -mtune=generic'</code>

helloWorld



```
<!DOCTYPE html>
<html>
<body>

<h1>My first PHP page</h1>

<?php
echo "Hello World!";
print "Hello World!";
echo "3 * 5". '<br/>';
echo 3 * 5;
echo 'Goodbye World!';
?>

</body>
</html>
```

My first PHP page

Hello World!Hello World!3 * 5
15Goodbye World!

3. Căn bản về PHP

Comment – Chú thích code



- Giải thích rõ ràng ý nghĩa của đoạn mã
 - Chỉ sử dụng khi cần thiết
- Có 3 cách comment trong PHP: **//**, **#** hoặc **/* ... */**

```
<!DOCTYPE html>
```

```
<html>
```

```
<body>
```

```
<?php
```

```
// This is a single-line comment
```

```
# This is also a single-line comment
```

```
?>
```

```
</body>
```

```
</html>
```

```
<!DOCTYPE html>
```

```
<html>
```

```
<body>
```

```
<?php
```

```
/*
```

```
This is a multiple-lines comment block  
that spans over multiple  
lines
```

```
*/
```

```
?>
```

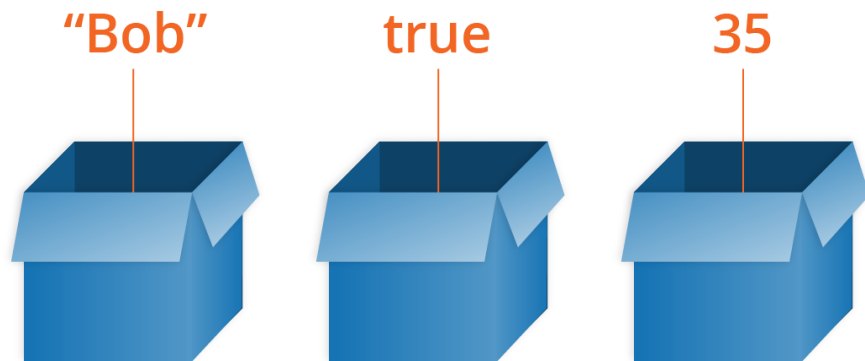
```
</body>
```

```
</html>
```

Biến là gì?



- Biến là một định danh, nó dùng để lưu trữ các giá trị có thể thay đổi
- Sử dụng phép gán (=) để thay đổi giá trị
- Tưởng tượng biến như “thùng chứa” dùng để lưu trữ thông tin.



Biến - Khai báo biến



- Một số quy định đặt tên biến:
 - Biến bắt đầu bằng kí tự **\$**: ví dụ **\$toiLaBien**
 - Biến có thể là chữ, số, kí tự **_**, kí tự **-** nhưng không sử dụng dấu cách:
 - Ví dụ: **\$toiLaBien**, **\$toi_la_bien**, **\$toi-la-bien** nhưng không được ~~**\$toi la bien**~~
 - Kí tự ngay sau \$ phải là chữ hoặc kí tự **_**
 - Ví dụ: **\$_99DoaHong** nhưng không được ~~**\$99DoaHong**~~
 - PHP phân biệt chữ in HOA và chữ in thường
 - Ví dụ: **\$toilabien** khác với **\$toiLaBien**
 - Biến phải nằm trong dấu **" ... "** khi muốn in:
 - Ví dụ: **echo "\$toiLaBien"** chữ không được ~~**echo '\$toiLaBien'**~~.

Biến – Các kiểu dữ liệu



- Integer //123, -123, 0123, 0x1A
- String //'Hello world!', "Hi"
- Float/Double //1.234, -1.23
- Boolean //true, false, TRUE, FALSE
- NULL //null
- Array //[1, 2, 3, 4], ['name' => 'Manh', 'age' => 12]
- Object //class - object
- Có thể sử dụng hàm var_dump() để kiểm tra kiểu dữ liệu của biến

Biến – Ép kiểu dữ liệu



- Dùng để thay đổi kiểu dữ liệu của biến.
- Cú pháp: dùng keyword ép kiểu trước biến.
- Các dạng ép kiểu được chấp nhận trong PHP: (int), (integer) / (bool), (boolean) / (float), (double), (real) / (string) / (array) / (object)
- Ví dụ
 - `$a=11.2; // $a là kiểu thực`
 - `$a=(int)$a; // Bây giờ, $ a là kiểu nguyên, giá trị = 11`
 - `$a= (double)$a; // Bây giờ $a lại trở về kiểu thực = 11.0`
 - `$b= (string)$a; // $b là giá trị kiểu chuỗi ="11"`

Biến – Phạm vi hoạt động



- Trong PHP có 3 kiểu phạm vi biến:
- Biến cục bộ
 - Biến được khai báo trong một hàm, biến đó chỉ có thể được truy cập trong bản thân hàm, khi ra ngoài hàm nó sẽ không tồn tại.
- Biến toàn cục
 - Khác với biến cục bộ, biến toàn cục có thể được truy cập từ bất kì đâu trong chương trình, sử dụng từ khóa GLOBAL trước tên biến
- Biến tĩnh
 - Biến được khai báo kiểu cục bộ trong hàm ko bị mất đi sau mỗi lần gọi hàm, sử dụng từ khóa static trước tên biến

Hằng số



- Hằng bản chất là một loại biến nhưng không thể thay đổi giá trị được, trong khi biến có thể thay đổi.
- Cách đặt tên hằng thì tương tự như biến, nhưng theo convention thì sẽ đặt tên hằng là chữ in hoa. VD: AGE, MAX_SIZE .v.v
- Khai báo hằng không cần có dấu \$ đằng trước, trong khi biến thì cần ký hiệu đó
- Hằng có thể truy cập từ mọi nơi trong khi biến chỉ có thể được truy cập từ nơi nó được khai báo.
- Hằng một khi đã được thiết lập giá trị thì chúng ta không thể thay đổi giá trị của nó

Hằng số - Khai báo



- Sử dụng hàm: `define('<tên-hằng>', '<giá-trị-hằng>');`
 - `define('MAX_FILE', 10);`
- Dùng từ khóa `const`
 - `const ABC = 5;`
- Để hiển thị dữ liệu của hằng có 2 cách:
 - `echo <tên-hằng>;` // cách thông dụng
 - `echo constant("<tên-hằng>");`

Hằng số - Các Hằng được định nghĩa sẵn



Tên hằng	Giải thích
<code>__LINE__</code>	Số dòng hiện tại khi gọi hằng này
<code>__FILE__</code>	Đường dẫn tuyệt đối tới file hiện tại gọi hằng này
<code>__DIR__</code>	Đường dẫn đến folder chứa file hiện tại
<code>__CLASS__</code>	Lớp được gọi dùng trong lập trình OOP
<code>__METHOD__</code>	Phương thức được gọi trong lập trình OOP
<code>__NAMESPACE__</code>	Tên NAMESPACE trong lập trình OOP

Hàm – Khái niệm



- Là 1 tập các dòng lệnh dùng để xử lý tác vụ nào đó
- Là khái niệm rất quan trọng trong lập trình
- Sử dụng hàm cho phép chia nhỏ chương trình thành các chức năng nhỏ hơn
- Tính chất quan trọng nhất của hàm là tính tái sử dụng

Hàm – Phân loại hàm



- Hàm xây dựng sẵn

Ví dụ: `echo`, `var_dump()`, `print_r()`

- Hàm tự định nghĩa

```
function function_name(parameter_list){  
    //code Javascript  
    return value;  
}
```

- `function_name`: tên hàm do bạn tự định nghĩa
- `parameter_list`: danh sách các tham số truyền vào hàm nếu có
- `return value`: câu lệnh `return` này là tùy chọn, để thể hiện hàm này có mang giá trị nào không

Hàm – Gọi hàm



- Khi định nghĩa ra 1 hàm, thì hàm sẽ chưa được sử dụng cho đến khi nó được gọi
- Cú pháp

```
function_name(parameter_list);
```

- Ví dụ

```
function sum(a, b) {  
    return a + b;  
}  
  
echo "Tổng của 2 vs 5 = " + sum(2, 5);
```

Hàm - Include và Require



- Việc bóc tách mã code thành nhiều file phù hợp cho việc quản lý và bảo trì code hơn là code tất cả trên cùng 1 file
- PHP cung cấp các hàm `include`, `require`, `include_once`, `require_once` để thực thi việc import file
- Sử dụng đường dẫn tương đối để import file
- Cú pháp
 - `require "<đường-dẫn-file>";`
 - `require_once "<đường-dẫn-file>";`
 - `include "<đường-dẫn-file>";`
 - `include_once "<đường-dẫn-file>";`

Hàm – Include và Require



- Hàm require về bản chất là hàm include, chỉ khác về cách xử lý lỗi khi import file không tồn tại
- Require tạo ra lỗi Fatal, dừng thực thi code phía sau nó
- Include tạo ra lỗi Warning, tiếp tục thực thi code phía sau nó
- Hàm require_once về bản chất là hàm require, chỉ khác về cách import file là kiểm tra file đã được import vào trước đó hay chưa, nếu chưa mới import.
- Tương tự với include_once

4. Toán tử

Toán tử số học



Toán tử	Mô tả	Ví dụ
+	Phép cộng	$A = 5 + 8$
-	Phép trừ	$A = 8 - 5$
*	Phép nhân	$A = 8 * 5$
/	Phép chia	$A = 20 / 5$
%	Phép chia lấy số dư	$10 \% 3 = 1$
++	Tăng lên một đơn vị	++x sẽ trả về giá trị của x sau khi tăng. x++ sẽ trả về giá trị của x trước khi tăng.
--	Giảm một đơn vị.	--x sẽ trả về giá trị của x sau khi giảm. x-- sẽ trả về giá trị của x trước khi giảm. .

Toán tử so sánh



Toán tử	Mô tả	Ví dụ
==	Bằng. Trả về giá trị true nếu các toán hạng bằng nhau.	<code>a == b</code>
!=	Không bằng. Trả về giá trị true nếu các toán hạng không bằng nhau.	<code>a != 5</code>
>	Lớn hơn. Trả về giá trị true nếu toán hạng trái lớn hơn toán hạng phải.	<code>a > b</code>
>=	Lớn hơn hoặc bằng. Trả về giá trị true nếu toán hạng trái lớn hơn hoặc bằng toán hạng phải.	<code>a >= b >= c</code>
<	Nhỏ hơn. Trả về giá trị true nếu toán hạng trái nhỏ hơn toán hạng phải.	<code>a < b</code>
<=	Nhỏ hơn hoặc bằng. Trả về giá trị true nếu toán hạng trái nhỏ hơn hoặc bằng toán hạng phải.	<code>a <= b <= c</code>

Toán tử logic



Toán tử	Giá trị	Mô tả
And (&&)	expr1 && expr2	Trả về true khi cả 2 biểu thức expr1 và expr2 trả về true
Or ()	expr1 expr2	Trả về true khi có ít nhất 1 trong 2 biểu thức expr1 và expr2 trả về true
Not (!)	!expr	Trả về giá trị false nếu biểu thức đúng và trả về giá trị true nếu biểu thức sai.

Toán tử gán



Toán tử	Ví dụ	Viết đầy đủ
=	<code>x = y;</code>	
+=	<code>x += y;</code>	<code>x = x + y;</code>
-=	<code>x -= y;</code>	<code>x = x - y;</code>
*=	<code>x *= y;</code>	<code>x = x * y;</code>
/=	<code>x /= y;</code>	<code>x = x / y;</code>
%=	<code>x %= y;</code>	<code>x = x % y;</code>

Toán tử điều kiện



Toán tử	Mô tả	Ví dụ
? :	Biểu thức điều kiện, nếu điều kiện là true ? gán giá trị là X : ngược lại là Y	<code>\$result = (\$a > \$b) ? \$a : \$b;</code>

5. Các cấu trúc điều khiển

Câu lệnh điều kiện IF



- Cú pháp

```
if (expression) {  
    Statement(s) to be executed if expression is true  
}
```
- Chỉ kiểm tra duy nhất 1 trường hợp khi biểu thức điều kiện expression là TRUE
- Cú pháp viết tắt khi lồng HTML:

```
<?php if(): ?>  
<?php endif; ?>
```

Câu lệnh điều kiện IF .. ELSE



- Cú pháp

```
if (expression) {  
    Statement(s) to be executed if expression is true  
} else {  
    Statement(s) to be executed if expression is false  
}
```
- Kiểm tra 2 trường hợp dựa vào tính đúng sai của biểu thức điều kiện expression
- Cú pháp viết tắt khi lồng HTML: `<?php if(): ?>`
`<?php else: ?>`
`<?php endif; ?>`

Câu lệnh điều kiện – IF .. ELSE IF



- Cú pháp

```
if (expression 1) {  
    Statement(s) to be executed if expression 1 is true  
} else if (expression 2) {  
    Statement(s) to be executed if expression 2 is true  
} else {  
    Statement(s) to be executed if no expression is true  
}
```
- Kiểm tra > 2 trường hợp dựa vào tính đúng sai của biểu thức điều kiện expression tương ứng
- Cú pháp viết tắt khi lồng HTML: `<?php if(): ?>`
`<?php elseif(): ?>`
`<?php endif; ?>`

Biểu thức SWITCH ... CASE



- Cú pháp

```
switch (expression){  
    case condition 1: statement(s)  
        break;  
    case condition 2: statement(s)  
        break;  
    ...  
    case condition n: statement(s)  
        break;  
    default: statement(s)  
}
```

- Có thể thay thế khi sử dụng quá nhiều if..elseif..else trong trường hợp so sánh bằng

Vòng lặp FOR



- Vòng lặp xác định vì biết trước được số lần lặp

- Cú pháp

```
for (giá-trị-khởi-tạo; biểu-thức-điều-kiện; biểu-thức-thay-đổi-giá-trị-khởi-tạo){  
    code-thực-thi;  
}
```

- Vòng lặp sẽ lặp khi biểu thức điều kiện đúng, khi biểu thức điều kiện sai thì sẽ thoát khỏi vòng lặp
- Cần chú ý về điều kiện dừng vòng lặp, tránh vòng lặp vô hạn
- Cú pháp viết tắt khi làm việc với HTML

```
<?php for() :?>
```

```
<?php endfor; ?>
```

Vòng lặp WHILE



- Vòng lặp dùng cho các bài toán không xác được được số lần lặp

- Cú pháp

```
while (biểu-thức-điều-kiện){  
    code-thực-thi;  
}
```

- biểu-thức-điều-kiện: điều kiện để dừng vòng lặp, nếu có giá trị false thì dừng vòng lặp, true tiếp tục lặp
- Để tránh vòng lặp vô hạn, luôn phải có hành động thay đổi biến trong biểu-thức-điều-kiện để tới lần lặp nào đó biểu-thức-điều-kiện sẽ thành false
- Cú pháp viết tắt khi làm việc với HTML

```
<?php while():?>
```

```
<?php endwhile; ?>
```

Vòng lặp DO ... WHILE



- Vòng lặp dùng cho các bài toán không xác được được số lần lặp
- Cú pháp

```
do {  
    code-thực-thi;  
}  
while (biểu-thức-điều-kiện );
```
- Khác với while, do...while luôn thực hiện ít nhất 1 lần lặp, cho dù điều kiện sai
- Để tránh vòng lặp vô hạn, luôn phải có hành động thay đổi biến trong biểu-thức-điều-kiện để tới lần lặp nào đó biểu-thức-điều-kiện sẽ thành false

Từ khóa Break, Continue



- Được sử dụng chủ yếu trong các vòng lặp
- **Break:**
 - Kết thúc vòng lặp mà không quan tâm đến điều kiện của vòng lặp đang TRUE hay FALSE
- **Continue:**
 - Nhảy tới lần lặp kế tiếp, đồng thời bỏ qua các dòng lệnh phía sau nó trong vòng lặp hiện tại

CSE485 – Công nghệ Web

Bài 03: PHP Căn bản 02

dungkt@tlu.edu.vn

Nội dung



- 1) Mảng
- 2) Các hàm dựng sẵn cho mảng
- 3) Các hàm dựng sẵn thao tác với String
- 4) Các hàm dựng sẵn thao tác với Số
- 5) Các hàm dựng sẵn thao tác với Time

1. Mạng

Mảng – Định nghĩa



- Mảng là cấu trúc dữ liệu có thể lưu trữ một hoặc nhiều kiểu giá trị tại 1 thời điểm
- Các phần tử trong mảng có thể là bất cứ kiểu dữ liệu nào.
- Bài toán: viết chương trình PHP lưu thông tin nhân viên của công ty
- Sử dụng biến:

```
$staff1 = "Nhân viên A";
```

```
$staff2 = "Nhân viên B";
```

```
...
```

- Sử dụng mảng:

```
$staffs = array('Nhân viên A', 'Nhân viên B', 'Nhân viên C');
```
- Xem cấu trúc mảng bằng các lệnh sau: `var_dump()`, `print_r()`

Mảng - Khai báo



- **\$arr** = array(<Danh sách các phần tử>);
- **\$arr** = [<Danh sách các phần tử>]; // sử dụng từ php version >= 5.4
- Danh sách các phần tử được ngăn cách nhau bởi dấu ,
- Ví dụ

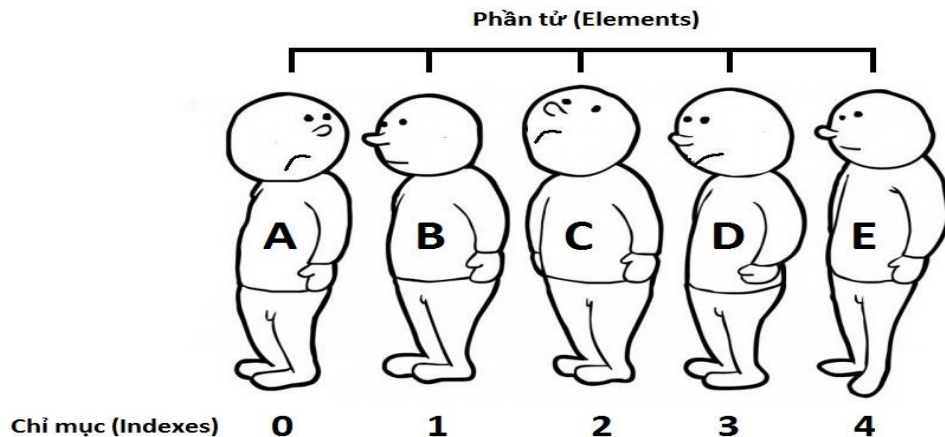
```
$staffs = array('Nhân viên A', 'Nhân viên B', 'Nhân viên C', 'Nhân viên D', 'Nhân viên E');  
$staffs = ['Nhân viên A', 'Nhân viên B', 'Nhân viên C', 'Nhân viên D', 'Nhân viên E'];  
$mangs = [0, 3, 6, 9];
```
- Cú pháp viết tắt khi viết lồng với HTML

```
<?php foreach(<biểu-thức-lặp>): ?>  
<?php endforeach;?>
```

Mảng - Key của mảng



- Là giá trị dùng để xác định vị trí của phần tử trong mảng
- Các giá trị của từng phần tử mảng được truy xuất thông qua key này
- Phần tử đầu tiên trong mảng luôn có key = 0



Mảng – Vòng lặp For ... Each



- Vòng lặp chuyên sử dụng để lặp các phần tử trong mảng
- Ngoài ra cũng có thể sử dụng các vòng lặp for, while, do...while để lặp mảng
- Cú pháp

```
foreach ($array as $key => $value){  
    //code-logic;  
}
```

```
foreach ($array as $value){  
    //code-logic;  
}
```

- Trong đó

\$array: mảng cần lặp

\$key: key của phần tử tại thời điểm lặp

\$value: giá trị tương ứng của phần tử với key là \$key

Mảng – Vòng lặp For ... Each



- Ví dụ: in ra danh sách các nhân viên

```
$staffs = array('Nhân viên A', 'Nhân viên B', 'Nhân viên C', 'Nhân viên D', 'Nhân viên E');  
echo 'Danh sách các nhân viên và vị trí tương ứng';  
echo '<br />';  
foreach($staffs as $key => $value){  
    echo "Vị trí $key : $value";  
    echo '<br />';  
}
```

- Ngoài ra có thể lấy giá trị của phần tử trực tiếp thông qua key tương ứng:

VD: \$staff[0]; \$staff[1]; \$staff[2];

Mảng số nguyên



- Key bắt buộc phải ở dạng số
- Phần tử đầu tiên luôn mặc định có key = 0
- Ví dụ

```
$arraysIndexes = ['phần tử thứ nhất', 2, '3'];
```

- Lấy giá trị của phần tử bằng vòng lặp foreach hoặc truy xuất trực tiếp thông qua key
- Ví dụ:

```
$element0 = $arraysIndexes[0];
```

```
$element1 = $arraysIndexes[1];
```

Mảng kết hợp



- Khác biệt duy nhất so với mảng số nguyên là key của mảng có thêm dạng chuỗi
- Ví dụ

```
$salaries = [  
    'A' => 1000,  
    'B' => 2000,  
    'C' => 3000,  
    'D' => 4000,  
];
```

- Lấy giá trị của phần tử bằng vòng lặp foreach hoặc truy xuất trực tiếp thông qua key
- Ví dụ:

```
$elementA = $salaries['A'];  
$elementB = $salaries['B'];
```


Mảng nhiều chiều



- Trong một mảng đa chiều, mỗi phần tử cũng có thể là một mảng. Và mỗi phần tử trong một mảng phụ cũng có thể là một mảng, và cứ tiếp tục như vậy.
- Trong mảng đa chiều, key có thể ở dạng số hoặc string
- Không nên tạo ra mảng lớn hơn 3 chiều vì rất khó quản lý

Ví dụ mảng 2 chiều:

```
$staffs = [  
    'group' => [  
        'staff1' => "Nhân viên A",  
        'staff2' => "Nhân viên B",  
    ],  
    'salary' => [  
        'staff1' => 1000,  
        'staff2' => 2000,  
    ]  
];
```

2. Các hàm dựng sẵn cho mảng

Các hàm có sẵn cho thao tác với mảng



Tên hàm	Mô tả
array_sum(\$array)	Trả về tổng các giá trị của mảng \$array
array_key_exists(\$key, \$value)	Kiểm tra xem chỉ mục \$key có tồn tại trong mảng \$array hay không, trả về TRUE nếu tồn tại, ngược lại là FALSE
array_merge(\$array1, \$array2)	Gộp mảng array2 vào array1, các phần tử trong array2 sẽ thêm vào cuối mảng array1, trả về 1 mảng mới
array_search(\$keyword, \$array)	Tìm kiếm theo giá trị của mảng, trả về chỉ mục(key) của phần tử đó nếu có.
array_slice(\$array, \$begin, \$length)	Lấy các phần tử từ vị trí \$begin của mảng \$array, lấy \$length phần tử, trả về mảng mới
array_unique(\$array)	Loại bỏ các phần tử có giá trị trùng lặp, và trả về 1 mảng mới

Các hàm có sẵn cho thao tác với mảng



Tên hàm	Mô tả
<code>array_values(\$array)</code>	Tạo ra mảng chỉ mục mới với giá trị là giá trị của mảng ban đầu, đồng thời các chỉ mục (\$key) cũng sẽ được sắp xếp lại theo thứ tự 0,1,2.v.v
<code>array_keys(\$array)</code>	Tạo ra mảng chỉ mục mới với giá trị là chỉ mục của mảng ban đầu
<code>array_pop(\$array)</code>	Bỏ đi phần tử cuối cùng của mảng ban đầu và trả về giá trị của phần tử cuối cùng đó
<code>array_push(\$array, \$var, \$var..)</code>	Thêm 1 hoặc nhiều phần tử vào cuối mảng ban đầu, trả về số lượng phần tử của mảng ban đầu sau khi thêm
<code>array_shift(\$array)</code>	Xóa phần tử đầu tiên của mảng ban đầu, trả về giá trị của phần tử vừa bị xóa
<code>array_unshift(\$array, \$var, \$var ...)</code>	Ngược lại với <code>array_shift</code> , hàm này sẽ thêm các giá trị \$var vào mảng ban đầu, và trả về số lượng phần tử của mảng ban đầu sau khi thêm

Các hàm có sẵn cho thao tác với mảng



Tên hàm	Mô tả
<code>array_flip(\$array)</code>	Trả về 1 mảng mới với từ việc đổi chỉ mục với giá trị của mảng ban đầu, lưu ý hàm này chỉ xử lý được key/value có kiểu dữ liệu là integer hoặc string
<code>count(\$array)</code>	Đếm số phần tử của 1 mảng
<code>explode(\$character, \$string)</code>	Chuyển chuỗi \$string thành mảng dựa vào ký tự phân tách \$character
<code>end(\$array)</code>	Trả về giá trị cuối cùng của mảng \$array
<code>is_array(\$array)</code>	Kiểm tra xem mảng \$array có phải là 1 mảng hay không, trả về TRUE nếu là mảng, ngược lại là FALSE
<code>sort(\$array)</code>	Sắp xếp mảng ban đầu theo chiều tăng dần các giá trị, trả về TRUE nếu thành công => mảng bị thay đổi, FALSE nếu thất bại

Các hàm có sẵn cho thao tác với mảng



Tên hàm	Mô tả
asort(\$array)	tương tự như sort nhưng sẽ giữ nguyên key của phần tử ở mảng ban đầu (với sort thì sẽ bị reset key), trả về TRUE nếu thành công, FALSE nếu thất bại
ksort(\$array)	sắp xếp theo chiều tăng dần theo key của mảng ban đầu, trả về TRUE nếu thành công => mảng bị thay đổi, FALSE nếu thất bại
rsort(\$array)	đảo ngược vị trí của các phần tử trong mảng ban đầu, trả về TRUE nếu thành công, FALSE nếu thất bại
implode(\$character, \$array)	convert các giá trị của mảng \$array thành string ngăn cách nhau bởi ký tự \$character
in_array(\$value, \$array)	kiểm tra xem giá trị \$value có tồn tại trong mảng \$array hay không, trả về TRUE nếu tồn tại, FALSE nếu không tồn tại
list(\$var1, \$var2, ...)	gán các biến tương ứng với các giá trị của 1 mảng

Các hàm có sẵn cho thao tác với mảng



Tên hàm	Mô tả
<code>min(\$array)</code>	Trả về giá trị nhỏ nhất của mảng \$array
<code>max(\$array)</code>	Trả về giá trị lớn nhất của mảng \$array
<code>print_r(\$array)</code>	Xem cấu trúc mảng
<code>reset(\$array)</code>	Trả về giá trị đầu tiên của mảng \$array
<code>unset(\$var1, \$var2,)</code>	Xóa phần tử \$var1, \$var2, ... khỏi mảng

3. Các hàm dựng sẵn thao tác với String

Các hàm có sẵn thao tác với String



Tên hàm	Mô tả
Nối chuỗi sử dụng ký tự .	<pre>\$string = "string1" . " , string2"; echo \$string; //string1, string2</pre>
<code>strlen(\$string)</code>	Trả về độ dài của chuỗi \$string
<code>str_word_count(\$string, \$format, \$charlist)</code>	<p>Đếm tổng số từ có trong chuỗi \$string</p> <p>\$format: chỉ định kiểu giá trị trả về của hàm <code>str_word_count()</code>. Các giá trị này có thể là:</p> <ul style="list-style-type: none">0 - Mặc định - trả về số lượng từ đếm được1 - Trả về một mảng chứa các từ trong chuỗi2 - Trả về một mảng với key là vị trí của từ trong chuỗi và value là từ trong chuỗi <p>\$charlist: chỉ định các ký tự đặc biệt sẽ được xem như một từ trong chuỗi</p>

Các hàm có sẵn thao tác với String



Tên hàm	Mô tả
strtoupper(\$string)	Chuyển tất cả các ký tự trong chuỗi \$string sang chữ hoa, trả về 1 chuỗi mới
strtolower(\$string)	Chuyển tất cả các ký tự trong chuỗi \$string sang chữ thường, trả về 1 chuỗi mới
ucfirst(\$string)	Đổi ký tự đầu tiên của chuỗi \$string sang chữ in hoa, trả về 1 chuỗi mới
ucwords(\$string)	Đổi các ký tự đầu tiên của các từ trong chuỗi \$string thành chữ in hoa

Các hàm có sẵn thao tác với String



Tên hàm	Mô tả
<code>trim(\$string, \$character)</code>	Xoá các ký tự <code>\$character</code> ở đầu và cuối của chuỗi <code>\$string</code> , nếu không nhập <code>\$character</code> thì mặc định là khoảng trắng
<code>ltrim(\$string, \$character)</code>	Tương tự <code>trim</code> nhưng chỉ xóa bên trái
<code>rtrim(\$string, \$character)</code>	Chuyển tất cả các ký tự trong chuỗi <code>\$string</code> sang chữ hoa, trả về 1 chuỗi mới
<code>str_replace(\$search, \$replace, \$string)</code>	Tìm kiếm chuỗi <code>\$search</code> trong chuỗi <code>\$string</code> , thay thế bằng chuỗi <code>\$replace</code> , trả về 1 chuỗi mới
<code>preg_replace(\$pattern, \$replace, \$string)</code>	Tìm kiếm theo khuôn mẫu <code>\$pattern</code> trong chuỗi <code>\$string</code> , thay thế bằng chuỗi <code>\$replace</code> , trả về 1 chuỗi mới

Các hàm có sẵn thao tác với String



Tên hàm	Mô tả
<code>substring(\$string, \$start, \$length)</code>	<p>Cắt chuỗi \$string, vị trí cắt từ \$start, và cắt \$length ký tự, trả về chuỗi mới</p> <p>\$start: nếu \$start là một số dương, chuỗi được trả về sẽ bắt đầu từ vị trí \$start(ký tự đầu tiên của chuỗi được tính là 0). Nếu \$start là một số âm chuỗi con trả về sẽ được cắt từ vị trí \$start tính từ cuối chuỗi</p> <p>\$length: độ dài của chuỗi trả về. Mặc định length được tính đến cuối chuỗi</p> <ul style="list-style-type: none">• \$length>0: độ dài của chuỗi trả về được tính từ đầu chuỗi• \$length<0: độ dài của chuỗi trả về được tính từ cuối chuỗi
<code>strstr(\$string, \$needle)</code>	<p>Tách chuỗi \$string bắt đầu từ \$needle đến hết chuỗi, trả về 1 chuỗi mới</p>

Các hàm có sẵn thao tác với String



Tên hàm	Mô tả
<code>strpos(\$string, \$needle)</code>	Tìm vị trí xuất hiện lần đầu tiên của <code>\$needle</code> trong chuỗi <code>\$string</code> , trả về vị trí xuất hiện nếu có, và ngược lại trả về <code>FALSE</code> nếu không thấy
<code>strrev(\$string)</code>	Đảo ngược chuỗi <code>\$string</code> , trả về 1 chuỗi mới
<code>strcmp(\$string1, \$string2)</code>	So sánh hai chuỗi phân biệt chữ hoa chữ thường, hàm trả về số âm nếu chuỗi thứ nhất nhỏ hơn chuỗi thứ 2, trả về 0 nếu hai chuỗi bằng nhau và trả về số dương nếu chuỗi thứ nhất lớn hơn chuỗi thứ 2.

4. Các hàm dựng sẵn thao tác với Số

Các hàm dựng sẵn thao tác với Số



Tên hàm	Mô tả
is_numeric(\$var)	Kiểm tra xem biến \$var có phải là kiểu Number hay không, trả về TRUE nếu phải, ngược lại là FALSE
is_int(\$var)	Kiểm tra xem biến \$var có phải kiểu integer hay không, trả về TRUE nếu phải, ngược lại là FALSE
is_float(\$var)	Kiểm tra xem biến \$var có phải kiểu float hay không, trả về TRUE nếu phải, ngược lại là FALSE
round(\$var)	Trả về số nguyên gần biến \$var nhất
ceil(\$var)	Trả về số nguyên được làm tròn lên số nguyên lớn nhất gần với \$var nếu là số dương, ngược lại trả về phần nguyên nếu là số âm
floor(\$var)	Trả về số nguyên được làm tròn xuống số nguyên nhỏ nhất gần với \$var nếu là số âm, ngược lại trả về phần nguyên nếu là số dương

Các hàm dựng sẵn thao tác với Số



Tên hàm	Mô tả
<code>max(\$var1, \$var2, ...)</code>	Trả về giá trị lớn nhất trong các biến <code>\$var1</code> , <code>\$var2</code> , ... truyền vào
<code>min(\$var1, \$var2, ...)</code>	Trả về giá trị nhỏ nhất trong các biến <code>\$var1</code> , <code>\$var2</code> , ... truyền vào
<code>rand(\$number1, \$number2)</code>	Trả về số ngẫu nhiên nằm trong khoảng <code>\$number1</code> và <code>\$number2</code>
<code>number_format(\$number)</code>	Định dạng số <code>\$number</code> theo hàng nghìn, trả về 1 string
<code>abs(\$number)</code>	Trả về giá trị tuyệt đối của số <code>\$number</code>
<code>pow(\$number, \$exponent)</code>	Trả về <code>\$number</code> mũ <code>\$exponent</code>
<code>sqrt(\$number)</code>	Trả về căn bậc 2 của <code>\$number</code>

5. Các hàm dựng sẵn cho Time

Các hàm dựng sẵn cho Time



Tên hàm	Mô tả
<code>getdate()</code>	trả về mảng chứa các thông tin về thời gian hiện tại của máy chủ
<code>date_default_timezone_get()</code>	trả về string của timezone(múi giờ) mặc định mà hệ thống máy chủ đang sử dụng
<code>date_default_timezone_set(\$timezone_identifier)</code>	thiết lập timezone mặc định cho hệ thống bằng string <code>\$timezone_identifier</code> , trả về TRUE nếu thành công, FALSE nếu thất bại.
<code>time()</code>	lấy giá trị unix timestamp tại thời điểm hiện tại (unix timestamp là số giây tại thời điểm hiện tại cho với 01/01/1970), trả về số int
<code>date(\$format, \$timestamp)</code>	định dạng các hiển thị thời gian theo format <code>\$format</code> và giá trị unix timestamp <code>\$timestamp</code>
<code>strtotime(\$format)</code>	chuyển <code>\$format</code> thành giá trị Unix timestamp

CSE485 – Công nghệ Web

Bài 04: PHP Form

dungkt@tlu.edu.vn

Nội dung



- 1) Cơ bản về Form
- 2) Form Upload File
- 3) Form – Các vấn đề bảo mật
- 4) Session và Cookie

1. Cơ bản về Form

Form: Khái niệm



- Form là thành phần không thể thiếu trong các ứng dụng Web vì form là nơi trao đổi và thu thập dữ liệu từ người dùng
- Form có 2 loại chính:
 - Form lấy thông tin dưới dạng căn bản
 - Form lấy thông tin dưới dạng các file được upload lên
- Có thể không sử dụng form để lấy thông tin từ người dùng không ?
 - Có thể, sử dụng kỹ thuật Ajax
- Hai phương thức thường được sử dụng khi gửi dữ liệu từ form là GET và POST
- Server phân biệt data gửi lên dựa vào thuộc tính name của các thẻ input form

Form: Khai báo



- Cú pháp khai báo HTML

`<form action='<url-xử-lý>' method='<tên-method>' >`

`<các-thẻ-input-html>`

`</form>`

- Trong đó:

form, action, method là các từ khóa

`<url-xử-lý>`: url dùng để xử lý các dữ liệu được gửi lên từ form

`<tên-method>`: phương thức truyền dữ liệu (POST/GET...)

Form: Ví dụ



facebook

Email or Phone

Password

☒ Keep me logged in

[Forgotten your password?](#)

[Log in](#)

Facebook helps you connect and share with the people in your life.

Sign Up

It's free and always will be.

First Name

Last Name

Your email address

Re-enter email address

New Password

Birthday

Day

Month

Year

[Why do I need to provide my date of birth?](#)

☐ Female ☐ Male

By clicking Sign Up, you agree to our [Terms](#) and that you have read our [Data Use Policy](#), including our [Cookie Use](#).

Sign Up

[Create a Page](#) for a celebrity, band or business.

Form: Danh sách Input



Component	Sample	HTML
Text Entry	<input type="text"/>	<code><input type="text"></code>
Password Entry	<input type="password"/>	<code><input type="password"></code>
Checkbox	<input type="checkbox"/>	<code><input type="checkbox"></code>
Radio Button	<input type="radio"/>	<code><input type="radio"></code>
Submit Button	<input type="submit" value="Submit Data"/>	<code><input type="submit" value="Submit Data"></code>
Reset Button	<input type="reset" value="Reset"/>	<code><input type="reset" value="Reset"></code>
File Selection Entry	<input type="file"/> No file chosen	<code><input type="file"></code>
Text Area (multiple lines)	<input type="textarea"/>	<code><textarea cols="20" rows="2"></code>
Select Menu (pop-up)	<input type="select" value="Red"/>	<code><select size="1"></code>
Select Menu (scrolling)	<input type="select" value="Red"/>	<code><select size="3"></code>

Form: Lưu ý đặt tên phần tử Input



- Với các input mà chỉ có 1 giá trị duy nhất tại 1 thời điểm, thì thuộc tính name của input sẽ ở dạng đơn, VD: input text, password, textarea, radio, select ở trạng thái đơn, file ở trạng thái đơn
- Với các input mà có thể có nhiều giá trị tại 1 thời điểm, thì thuộc tính name của input đó sẽ ở dạng mảng. VD: checkbox, select kiểu multi, select file kiểu multi.
 - VD: `<input type='checkbox' name='gender[]' />`
- Để input select và file có thể ở trạng thái chọn multi, cần thêm thuộc tính multiple vào cho thẻ
- Với riêng kiểu input file, sẽ có cách riêng để xử lý lấy dữ liệu từ file, được trình bày trong buổi học sau

Form: Phương thức GET



- Phương thức GET gửi dữ liệu thông qua URL trên trình duyệt
- Các biến truyền lên có format: <URL>?param1=value1¶m2=value2 ...
 - VD: `http://google.com.vn?name=Anh&age=18`
- Thường sử dụng cho truy vấn lấy dữ (GET) liệu từ server
- GET giới hạn độ dài chuỗi dữ liệu trên URL là 1024 ký tự
- PHP sử dụng biến `$_GET` lưu trữ toàn bộ dữ liệu gửi lên thông qua phương thức GET
- Không nên sử dụng GET để gửi các thông tin nhạy cảm như password...
- Dữ liệu gửi lên chỉ tồn tại khi submit form, vì vậy cần kiểm tra bằng hàm `isset()` để chắc chắn rằng đã submit form

Form: Phương thức POST



- Có tính bảo mật hơn GET, dữ liệu truyền đi không hiển thị trên trình duyệt
- Thường sử dụng cho các truy vấn thay đổi dữ liệu trong database (INSERT, UPDATE, DELETE)
- Không giới hạn độ dài dữ liệu như GET
- Tương tự như GET, PHP sử dụng biến `$_POST` được dùng để lưu trữ toàn bộ dữ liệu gửi lên thông qua phương thức POST

Form: Biến `$_REQUEST`



- Có thể được sử dụng để lấy dữ liệu gửi lên từ form của cả 2 phương thức GET và POST
- Chứa tất cả thông tin về các biến `$_GET`, `$_POST` và `$_COOKIE` (`$_COOKIE` là biến quản lý liên quan đến cookie trình duyệt)
- Thông thường trong form đã chỉ định cụ thể tên phương thức, nên sẽ ưu tiên sử dụng biến `$_GET` hoặc `$_POST` thay vì `$_REQUEST`

Form: Validate



- Là bước kiểm soát dữ liệu gửi lên từ user nhằm tăng tính bảo mật và tránh việc mất thời gian xử lý các dữ liệu rác.
- Là bước không thể thiếu khi xử lý form
- Các lỗi nhập từ người dùng phổ biến là dữ liệu trống và nhập sai định dạng
- Quá trình validate dữ liệu thường gồm 2 bước:
 1. Lấy dữ liệu từ user khi submit form và kiểm tra xem dữ liệu là hợp lệ hay chưa
 2. Nếu chưa hợp lệ, thông báo lỗi tới người dùng, đồng thời giữ nguyên các giá trị các trường mà user đã nhập đúng để tránh cho việc user phải nhập lại
 3. Có thể validate bằng Javascript, tuy nhiên trên server vẫn cần validate lại một lần nữa

Form: Submit



- Tùy thuộc vào phương thức gửi dữ liệu lên là gì (POST/GET) để gọi biến \$_POST/\$_GET tương ứng
- Server chỉ xử lý logic dữ liệu từ user sau khi đã hoàn thành bước validate dữ liệu
- Form có thể được submit bằng cách click input submit hoặc nhấn Enter

2. Form Upload File

Form Upload: Tổng quan



- Khai báo form cần thêm thuộc tính `enctype="multipart/form-data"` để cho phép form upload file
- Phương thức của form phải là POST
- Tương tự `$_POST`, `$_GET` hay `$_REQUEST`, biến toàn cục dùng lưu trữ thông tin file upload lên là `$_FILES`
- Thẻ HTML dùng để chọn file upload là `<input type=file />`
- Trong trường hợp upload nhiều file, cần đặt name dạng mảng, và thêm thuộc tính `multiple` trong input file. Xử lý upload file với trường hợp này sẽ là xử lý trên mảng các file.

Form Upload: Cấu trúc biến `$_FILES`



- Là mảng 2 chiều có dạng như sau **`$_FILES['nameInputFile']['properties']`**
 - **`nameInputFile`**: thuộc tính name của thẻ HTML input file
 - **`properties`**: đại diện cho 5 thuộc tính sau:
 - name: tên file được upload
 - type: kiểu dữ liệu của file như png, jpg, xlsx v.v
 - tmp_name: đường dẫn tạm thời mà file đang được lưu trên server
 - size: kích cỡ của file upload, đơn vị là Byte
 - error: trạng thái lỗi trong quá trình upload file, ở dạng số nguyên, 1 số mã lỗi thường gặp:
 - 0 - không có lỗi, hay upload thành công
 - 1 - Báo lỗi file upload vượt quá dung lượng cho phép
 - 2 - Báo lỗi số file upload vượt quá số lượng file cho phép của form
 - 3 - Báo lỗi file chỉ được upload 1 phần, không phải toàn bộ file
 - 4 - Báo lỗi không có file nào được upload

Form Upload: Hàm `move_uploaded_file()`



- Khi quá trình upload file không có lỗi (thuộc tính `error` của biến `$_FILES` = 0), sử dụng hàm `move_uploaded_file()` để lưu đường dẫn thật cho file
- Cú pháp: `move_uploaded_file(tmp_path, destination)`
 - `tmp_path`: đường dẫn tạm thời của file trên server
 - `destination`: đường dẫn thật sẽ lưu trữ file, phải là đường dẫn vật lý

3. Form – Các vấn đề bảo mật

Bảo mật Form: Khái niệm



- Bảo mật là vấn đề cực kỳ quan trọng đối với các hệ thống website
- Do form là thành phần chính của việc truyền dữ liệu từ client đến server nên thường là đích đến của các cuộc tấn công
- Hiện nay các framework đều hỗ trợ rất tốt việc chống các tấn công này
- Hai hình thức tấn công phổ biến trong form là
 - XSS (Cross-site scripting)
 - CSRF (Cross-site request forgery)

Bảo mật Form: Tấn công XSS



- Kỹ thuật này được thực hiện bằng cách chèn các mã script độc hại vào trong code của bạn
- Cách phòng chống
 - Không bao giờ tin tưởng dữ liệu mà user nhập, do vậy cần validate dữ liệu trước khi lưu vào database
 - Mã hóa các ký tự đặc biệt mà user nhập trong form thành các thực thể HTML, sử dụng hàm `htmlspecialchars()`

Bảo mật Form: Tấn công CSRF



- Kỹ thuật này giả mạo chính chủ sở hữu của hệ thống đó
 - Ví dụ: Hệ thống của bạn có link xóa user dạng `delete-user?id=xxx`, hacker biết được link này và gửi 1 mail cho bạn với 1 image có src là link trên, với id cụ thể, khi click vào ảnh đó đồng nghĩa bạn đã xóa user!
- Cách phòng chống: tạo key (token) cho form, với các bước xử lý như sau:
 - Thêm thẻ input ẩn với giá trị key và thuộc tính là hidden
 - Lưu key này vào session
 - Mỗi lần submit form sẽ kiểm tra key này có hợp lệ hay không, chỉ xử lý khi key hợp lệ

4. Session và Cookie

Session



- Được hiểu như 1 phiên làm việc của client, session lưu trữ thông tin giữa client và hệ thống
- Biến được lưu trong session có thể được truy cập từ mọi nơi trên hệ thống
- Session sẽ mất khi đóng trình ứng dụng
- Biến toàn cục trong PHP lưu trữ các thông tin về session là `$_SESSION`, có kiểu mảng
- Một số ứng dụng hay sử dụng session là đăng nhập, giỏ hàng .v.v

Session



- Khởi tạo: `session_start();`
- Thêm dữ liệu: `$_SESSION['name'] = value;`
- Lấy giá trị: `$_SESSION['name'];`
 - Lưu ý trước khi lấy giá trị cần kiểm tra session đã tồn tại hay chưa sử dụng lệnh `isset()`
- Xóa session
 - Xóa 1 phần tử cụ thể: `unset($_SESSION['name']);`
 - Xóa toàn bộ session trên hệ thống: `session_destroy();`

Cookie



- Thường được dùng để lưu các giá riêng của từng trang web cụ thể cho client
- Cookie không mất đi khi đóng ứng dụng, sự tồn tại của cookie phụ thuộc vào thời gian sống khi bạn set cho nó
- Cookie được lưu dưới trình duyệt của client
- Biến toàn cục trong PHP lưu trữ các thông tin về cookie là `$_COOKIE`, có kiểu mảng

Cookie



- Khởi tạo: `setcookie(name, value, expire, path, domain);`
 - name: tên cookie muốn tạo
 - value: giá trị cookie
 - expire: thời gian sống của cookie
 - path: đường dẫn lưu cookie, mặc định là '/'
 - domain: tên domain
- Lấy giá trị: `$_COOKIE['name'];`
 - Cần sử dụng lệnh `isset()` trước khi lấy giá trị
- Xóa cookie: Sử dụng lại phương thức `setcookie` như lúc khởi tạo, nhưng set thời gian sống nhỏ hơn thời gian hiện tại

Session vs Cookie



- Giống nhau:
 - đều dùng để lưu trữ thông tin giữa client và hệ thống
 - đều có thể được truy cập từ mọi nơi trên hệ thống
- Khác nhau:
 - Session có tính bảo mật hơn cookie
 - Cookie được lưu trên trình duyệt của client trong khi session được lưu trên server
 - Session sẽ bị mất khi đóng ứng dụng, trong khi cookie thì không

Bài tập: Session vs Cookie



- Demo chức năng login đơn giản, với mô tả như sau:
 - Màn hình login gồm 2 ô nhập username và password và 1 nút submit có tên Login
 - Trường hợp user nhập đúng username = nvanh, password = 123456 thì chuyển hướng sang màn hình khác - Gọi là màn hình login success, tại màn hình này sẽ hiển thị username của user và 1 nút Logout.
 - Tại màn hình Login success, khi click nút Logout thì sẽ chuyển hướng về trang login, tại trang login này sẽ hiển thị thông báo “Đăng xuất thành công”
 - Trong trường hợp login thành công rồi thì không thể truy cập vào trang login nữa, mà sẽ chuyển hướng sang màn hình Login success
 - Trong trường hợp chưa login, khi cố tình truy cập vào trang Login success (giả sử bạn đã biết trước url) thì báo “Cần đăng nhập để truy cập trang này” và chuyển hướng về trang login

Bài tập: Session vs Cookie



- Tạo form đăng nhập cho người dùng, gồm 2 trường username và password, và checkbox ghi nhớ đăng nhập
 - Nếu user/password là admin/123456 thì báo đăng nhập thành công, ngược lại là đăng nhập thất bại
 - Khi đăng nhập thành công, lưu session cho username, chuyển hướng tới 1 file khác , trong file này sẽ hiển thị tên username vừa đăng nhập và nút logout
 - Nếu user checkbox vào ô ghi nhớ đăng nhập, thì sẽ tự động đăng nhập vào lần sau
 - Khi click nút Logout, thì chuyển hướng người dùng về trang đăng nhập

CSE485 – Công nghệ Web

Bài 05: MySQL

dungkt@tlu.edu.vn

Nội dung



- 1) Tổng quan về MySQL
- 2) Các thao tác với MySQL

1. Tổng quan về MySQL

Tổng quan về MySQL

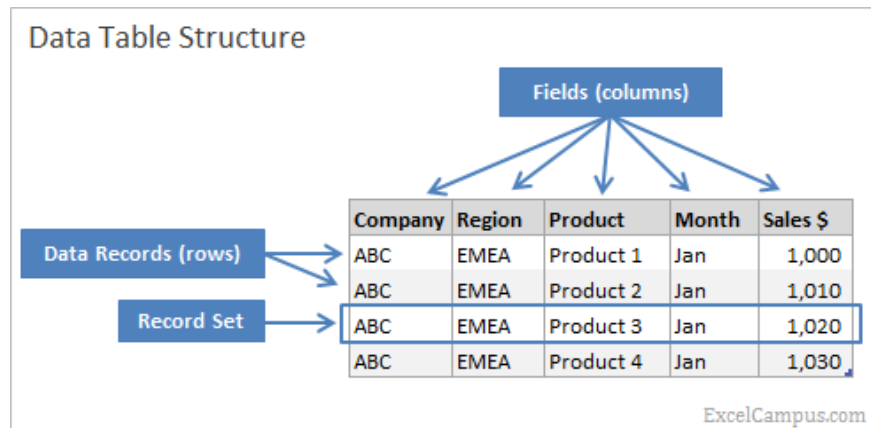


- MySQL là hệ quản trị cơ sở dữ liệu mã nguồn mở
- Hoàn toàn miễn phí
- Sử dụng cú pháp truy vấn SQL
- Tốc độ và dễ dàng sử dụng
- Hỗ trợ nhiều nền tảng khác nhau như Window, Linux, ...
- Mặc định khi cài XAMPP, đã install CSDL MySQL và tool PHPMyAdmin
- Có thể thao tác với MySQL thông qua giao diện đồ họa như PHPMyAdmin hoặc thông qua giao diện console của MySQL

Các thuật ngữ trong MySQL



- Trong CSDL MySQL bao gồm 1 hoặc nhiều các tables
- Mỗi table bao gồm các fields và các rows
- Cấu trúc của table được mô tả như hình:



- Ngoài ra trong table còn có thể có thêm primary key, foreign key

Bài tập



- Sử dụng phpMyadmin tạo CSDL có tên là demo_database, sau đó import file simple_db.sql trong thư mục mã nguồn tải về vào csdl vừa tạo sử dụng PhpMyadmin.
- Hãy liệt kê tất cả các bảng, trong từng bảng hãy liệt kê tất cả các fields, các thông tin về fields, các records, primary key, foreign key nếu có

2. Các thao tác với MySQL

Tạo CSDL



- **CREATE DATABASE** [IF NOT EXISTS] database_name CHARACTER SET utf8 COLLATE utf8_general_ci
 - database_name: tên CSDL muốn tạo
 - [IF NOT EXISTS]: tùy chọn cho phép không tạo nếu đã tồn tại CSDL cùng tên
- Ví dụ
 - CREATE DATABASE IF NOT EXISTS my_database

Xóa CSDL



- **DROP** DATABASE database_name
 - database_name: tên CSDL muốn xóa
- Ví dụ
 - DROP DATABASE my_database

Chọn CSDL



- **USE** database_name
 - database_name: tên CSDL muốn sử dụng
- Ví dụ
 - USE my_database

Các kiểu dữ liệu trong MySQL



- Kiểu Number**

Kiểu dữ liệu	Độ dài (số byte)	Giá trị nhỏ nhất (Có dấu)	Giá trị lớn nhất (Có dấu)	Giá trị nhỏ nhất (Không dấu)	Giá trị lớn nhất (Không dấu)
TINYINT	1	-128	127	0	255
SMALLINT	2	-32768	32767	0	65535
MEDIUMINT	3	-8388608	8388607 to	0	16777215
INT	4	-2147483648	2147483647	0	4294967295
BIGINT	8	-9223372036854775808	9223372036854775807	0	18446744073709551615
FLOAT	4	-3.402823466E+38	-1.175494351E-38	1.175494351E-38	3.402823466E+38
DOUBLE	8	-1.7976931348623157E+ 308	-2.2250738585072014E- 308	0, and 2.2250738585072014E- 308	1.7976931348623157E+ 308

Các kiểu dữ liệu trong MySQL



- Kiểu String**

Kiểu dữ liệu	Mô tả	Độ dài
CHAR	Dùng để lưu trữ một chuỗi ký tự có độ dài cố định (đã biết trước độ dài), tốc độ nhanh hơn varchar	1 - 255 ký tự
VARCHAR	Dùng để lưu trữ một chuỗi ký tự có độ dài thay đổi, Cần chỉ định trước độ dài khi tạo cột. Nếu đặt "size" lớn hơn 255 thì nó sẽ chuyển sang kiểu TEXT	1 - 255 ký tự
TINYTEXT	Dùng để lưu trữ một chuỗi ký tự, không cần chỉ định độ dài khi tạo cột	1 - 255 ký tự
TEXT	Dùng để lưu trữ một chuỗi ký tự, không cần chỉ định độ dài khi tạo cột	1- 65,535 ký tự
BLOB	Dùng để lưu trữ dữ liệu nhị phân	1 - 65,535 Byte
MEDIUMTEXT	Dùng để lưu trữ một chuỗi ký tự	1 - 16,777,215 ký tự
MEDIUMBLOB	Dùng để lưu trữ dữ liệu nhị phân	1 - 16,777,215 Byte
LONGTEXT	Dùng để lưu trữ một chuỗi ký tự	1 - 4,294,967,295 ký tự
LOBLOB	Dùng để lưu trữ dữ liệu nhị phân	1 - 4,294,967,295 Byte

Các kiểu dữ liệu trong MySQL



- Kiểu Date & Time**

Kiểu dữ liệu	Mô tả	Định dạng hiển thị
DATETIME	Được dùng để lưu trữ cả hai thông tin ngày tháng và thời gian. VD: 15h ngày 13 tháng 3 năm 2019 sẽ được lưu là 2019-03-13 15:00:00	YYYY-MM-DD HH:MM:SS
DATE	Được dùng để lưu trữ chỉ thông tin ngày tháng. VD: 15h ngày 13 tháng 3 năm 2019 sẽ được lưu là 2019-03-13	YYYY-MM-DD
TIMESTAMP	Lưu trữ cả hai thông tin ngày tháng và thời gian. Giá trị này sẽ được chuyển đổi từ múi giờ hiện tại sang UTC trong khi lưu trữ, và sẽ chuyển trở lại múi giờ hiện tại khi lấy dữ liệu ra.	YYYY-MM-DD HH:MM:SS
TIME	Được dùng để lưu trữ chỉ thông tin thời gian.	HH:MM:SS
YEAR	Được dùng để lưu trữ thông tin về năm.	YYYY

Tạo bảng



- **CREATE TABLE** table_name (field_name field_attribute)

- table_name : tên bảng muốn tạo
- field_name: tên cột muốn tạo
- field_attribute: danh sách các thuộc tính của cột muốn tạo

- Ví dụ tạo bảng books

```
CREATE TABLE books(  
id INT(11) NOT NULL AUTO_INCREMENT,  
title VARCHAR(100) NOT NULL,  
author VARCHAR(30) DEFAULT NULL,  
created_at TIMESTAMP,  
updated_at DATETIME,  
PRIMARY KEY (id)  
)
```

Xóa bảng



- **DROP TABLE** table_name
 - table_name: tên bảng muốn xóa
- Ví dụ:
 - DROP TABLE books

Lệnh INSERT



- Dùng để thêm dữ liệu vào bảng
- **INSERT INTO** table_name(field-1, field-2, ..., field-n) VALUES(value-1, value-2,..., value-n)
 - table_name: tên bảng
 - field-1, field-2, field-n: danh sách các cột
 - value-1, value-2, value-n: giá trị tương ứng với các cột
- Ví dụ:

```
INSERT INTO books(title, author, updated_at) VALUES('title1', 'author1', '2019-09-03')
```

Lệnh SELECT



- Dùng để lấy dữ liệu từ bảng
- **SELECT** field-1, field-2, ..., field-n FROM table_name [Where Clause] [OFFSET M] [LIMIT N]
 - table_name: tên bảng
 - field-1, field-2, field-n: danh sách các cột, lấy tất cả các cột sử dụng dấu *
 - Where Clause: điều kiện where (tùy chọn)
 - OFFSET M: trả về từ bản ghi thứ M (tùy chọn)
 - LIMIT N: giới hạn N bản ghi trả về (tùy chọn)
- Ví dụ: SELECT * FROM books

Lệnh UPDATE



- Dùng để cập nhật dữ liệu cho bảng
- **UPDATE** table_name SET field-1=value-1, field-2=value-2, ... [Where Clause]
 - table_name: tên bảng
 - field-1, field-2: danh sách các cột
 - value-1, value-2: giá trị tương ứng với các cột
- Ví dụ: UPDATE books SET author='new author' WHERE id = 1

Lệnh DELETE



- Dùng để xóa dữ liệu từ bảng
- **DELETE FROM** table_name [Where Clause]
 - table_name: tên bảng
- Ví dụ: DELETE FROM books WHERE id = 5

Mệnh đề LIKE



- Hoạt động theo kiểu so khớp, thường sử dụng trong mệnh đề Where
- **SELECT** field-1, field-2, ..., field-n **FROM** table_name **WHERE** field-1 **LIKE** condition-1 [AND [OR]] field-2=value-2 ...
 - table_name: tên bảng
 - field-1, field-2: danh sách các cột
 - condition-1: biểu thức so khớp của Like, thường dùng với ký tự % đại diện cho 1 hoặc nhiều ký tự bất kỳ
- Ví dụ: `SELECT * FROM `books` WHERE title LIKE '%2%'`

Từ khóa Order By



- Sắp xếp kết quả trả về theo quy luật nhất định
- **SELECT** field-1, field-2, ..., field-n **FROM** table_name **ORDER BY** field-1, [field-2] [ASC [DESC]]
 - ASC: ascending - sắp xếp theo chiều tăng dần theo cột field-1, field-2 .v.v
 - DESC: descending - sắp xếp theo chiều giảm dần theo cột field-1, field-2 .v.v
- Ví dụ: `sSELECT * FROM `books` ORDER BY id DESC`

Lệnh JOIN



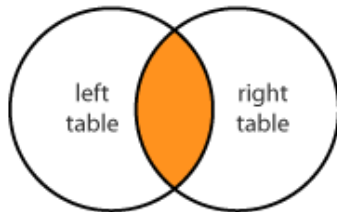
- Truy vấn nhiều bảng trong cơ sở dữ liệu
- **SELECT** table-1.field-1, table-1.field-2,..., table-2.field-3, table-2.field-4 **FROM** table-1
INNER JOIN table-2 ON table-1.field-5 = table-2.field-6
 - field-5 và field-6 là khóa liên kết giữa 2 bảng tương ứng
 - 3 cơ chế join chính bao gồm: inner join, left join, right join
- Ví dụ: `SELECT * FROM categories INNER JOIN books ON books.category_id = categories.id`

Lệnh JOIN

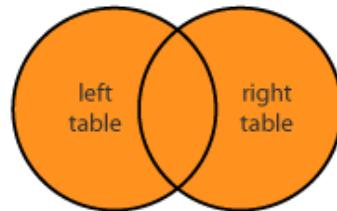


- Minh họa các cơ chế join trong MySQL

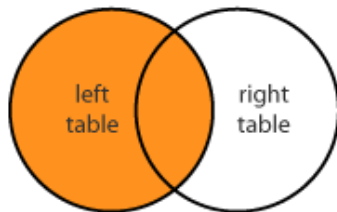
INNER JOIN



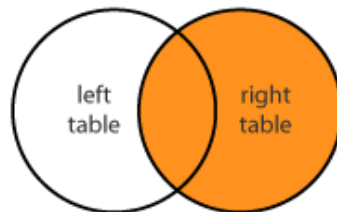
FULL JOIN



LEFT JOIN



RIGHT JOIN



Mệnh đề IN



- Được sử dụng để thay thế cho nhiều điều kiện AND
- Ví dụ:
 - Viết theo AND
 - `SELECT * FROM `books` WHERE id = 1 OR id = 2 OR id = 5`
 - Viết theo IN
 - `SELECT * FROM books WHERE id IN(1, 2, 5)`

Mệnh đề BETWEEN



- Dùng để thay thế cho câu điều kiện " \geq AND \leq "
- Ví dụ:
 - Sử dụng AND
 - `SELECT * FROM `books` WHERE id \geq 1 AND id \leq 3`
 - Sử dụng BETWEEN
 - `SELECT * FROM `books` WHERE id BETWEEN 1 AND 3`

Hàm COUNT



- Trả về tổng số bản ghi sau khi truy vấn
- Ví dụ:
 - `SELECT COUNT(category_id) AS count_category_id FROM books WHERE category_id = 6`

Hàm MAX



- Trả về giá trị lớn nhất của bản ghi theo cột
- Ví dụ: `SELECT MAX(category_id) FROM `books``

Hàm MIN



- Trả về giá trị nhỏ nhất của bản ghi theo cột
- Ví dụ: `SELECT MIN(category_id) FROM `books``

Hàm AVERAGE



- Trả về giá trị trung bình của bản ghi theo cột
- Ví dụ: `SELECT AVG(category_id) FROM `books``

Hàm SUM



- Trả về giá trị tổng của bản ghi theo cột
- Ví dụ: `SELECT SUM(category_id) FROM `books``

Mệnh đề GROUP BY



- Nhóm các giá trị theo cột chỉ định, thường kết hợp với các hàm trong MySQL trên các cột đã nhóm theo yêu cầu của bài toán
- Ví dụ : Tìm xem có bao nhiêu thí sinh đạt điểm 8 và 9 như bảng students dưới

```
SELECT diemthi, COUNT(diemthi) AS count_diem_thi  
FROM students  
GROUP BY diemthi
```

mssv	ho	ten	diemthi
1	Nguyen Thanh	Nam	9.00
2	Nguyen Van	Hoang	8.00
3	Tran Manh	Thang	8.00
4	Do Duy	The	9.00
5	Nguyen Thanh	Chung	9.50
6	Dinh Van	Cao	8.50
NULL	NULL	NULL	NULL

Bài tập



- Cho CSDL Quanlynhansu như hình dưới đây:

NHANVIEN	HONV	TENLOT	TENNV	PHAI	LUONG	MANV	NGSINH	DIACHI	PHG
	Đinh	Lê	Tiên	Nam	4000	123456789	01/09/1965	Nguyễn Trãi, Q5	1
	Nguyễn	Thị	Loan	Nữ	2500	333445555	12/08/1955	Nguyễn Huệ, Q1	5
	Nguyễn	Lan	Anh	Nữ	4300	666884444	15/09/1962	Lê Lợi, Q1	5
	Trần	Thanh	Tâm	Nam	3800	453453453	31/07/1972	Trần Nãi, Q2	2

DEAN	TENDA	MADA	DDIEM_DA	PHG
	Sản phẩm X	1	Quy Nhơn	5
	Sản phẩm Y	2	Nha Trang	5
	Sản phẩm Z	3	TP HCM	5
	Tin học hoá	10	Bình Dương	4

PHONGBAN	PHG	TENPHG
	1	Nhân sự
	2	Kế hoạch
	3	Kinh doanh
	4	Marketing
	5	Kế toán

PHANCONG	MANV	MADA	SOGIO
	123456789	1	32.0
	123456789	2	8.0
	666884444	3	40.0
	453453453	1	20.0

Bài tập



- Tạo cơ sở dữ liệu trên
- Truy vấn dữ liệu trong SQL
 - Hiển thị tất cả thông tin của bảng NHANVIEN
 - Hiển thị thông tin của những nhân viên ở phòng số 5
 - Hiển thị mã nhân viên, họ nhân viên, tên lót và tên nhân viên của những nhân viên ở phòng số 5 và có lương ≥ 3000
 - Hiển thị mã nhân viên, tên nhân viên của những nhân viên có lương từ 2000 đến 8000
 - Hiển thị thông tin của những nhân viên ở địa chỉ có tên đường là Nguyễn
 - Cho biết số lượng nhân viên
 - Cho biết số lượng nhân viên trong từng phòng ban
 - Hiển thị thông tin về mã nhân viên, tên nhân viên và tên phòng ban ở phòng kế toán

CSE485 – Công nghệ Web

Bài 06: PHP – MySQL

dungkt@tlu.edu.vn

Nội dung



- 1) PHP – MySQLi
- 2) PHP – Ajax – MySQL
- 3) Chống SQL Injection
- 4) PHP – PDO

1. PHP – MySQLi

MySQLi



- Là thư viện có sẵn trong MySQL, được phát triển dành cho ngôn ngữ PHP giúp lập trình viên dễ dàng kết nối tới CSDL MySQL
- MySQLi có thể viết theo cả 2 hướng lập trình là thủ tục và hướng đối tượng
- Ngoài MySQLi còn có 1 thư viện kết nối khác là PDO (PHP Data Objects), được viết hoàn toàn bằng hướng đối tượng, có thể kết nối tới nhiều CSDL khác nhau
- Chú ý rằng khi làm việc với CSDL thì quan trọng nhất vẫn là kỹ năng viết truy vấn

Các bước kết nối MySQLi



- Khởi tạo kết nối

`$connection = mysqli_connect($host, $username, $password, $database, $port);`

- `$host`: tên server của bạn, trong trường hợp là máy cá nhân thì sẽ là localhost
- `$username`: username đăng nhập database
- `$password`: password đăng nhập database
- `$database`: tên database muốn kết nối
- `$port`: cổng kết nối, cổng mặc định của MySQL là 3306

Các bước kết nối MySQLi



- Thực thi truy vấn **mysqli_query**(\$connection, \$query);
 - \$connection: biến kết nối đã khởi tạo ở bước 1
 - \$query: chuỗi truy vấn mysql
 - Với truy vấn Insert, Delete, Update hàm mysqli_query sẽ trả về Boolean
 - Với truy vấn Select hàm mysqli_query sẽ trả về object
- Đóng kết nối: **mysqli_close**(\$connection);

CRUD – Minh họa INSERT



- Insert
- Với CSDL demo, thêm sinh viên mới vào bảng students có name = Mạnh, age = 27

```
$connection = mysqli_connect('localhost', 'root', '', 'demo') or die('Kết nối thất bại : ' . mysqli_connect_error());
mysqli_query($connection, "set names 'utf8'");
$sqlInsert = "INSERT INTO students(name, age) VALUES ('Anh', 29)";
$isInsertTable = mysqli_query($connection, $sqlInsert);
if ($isInsertTable) {
    echo "Insert dữ liệu thành công";
}
else {
    echo "Insert dữ liệu thất bại";
}
mysqli_close($connection);
```

CRUD – Minh họa SELECT



- Select
- Với CSDL demo, lấy danh sách tất cả sinh viên trong bảng students

```
$connection = mysqli_connect('localhost', 'root', '', 'demo') or die('Kết nối thất bại : ' . mysqli_connect_error());
mysqli_query($connection, "set names 'utf8'");
$querySelect = "SELECT * FROM users";
$results = mysqli_query($connection, $querySelect);
if (mysqli_num_rows($results) > 0) {
    $users = mysqli_fetch_all($results, MYSQLI_ASSOC);
    foreach ($users as $user) {
        echo "Name: {$user['name']}";
        echo "Age: {$user['age']}";
    }
}
mysqli_close($connection);
```


CRUD – Minh họa UPDATE



- Update
- Với CSDL demo, cập nhật age = 50 cho sinh viên có mã id = 4 trong bảng students

```
$connection = mysqli_connect('localhost', 'root', '', 'demo') or die('Kết nối thất bại : ' . mysqli_connect_error());
mysqli_query($connection, "set names 'utf8'");
sqlUpdate = "UPDATE students SET age = 50 WHERE id = 4";
$isUpdateTable = mysqli_query($connection, $sqlUpdate);
if ($isUpdateTable){
    echo 'Update dữ liệu thành công';
}
else {
    echo 'Update dữ liệu thất bại';
}
mysqli_close($connection);
```

CRUD – Minh họa DELETE



- Delete
- Với CSDL demo, xóa sinh viên có mã id = 4 trong bảng students

```
$connection = mysqli_connect('localhost', 'root', '', 'demo') or die('Kết nối thất bại : ' . mysqli_connect_error());
mysqli_query($connection, "set names 'utf8'");
$sqlDelete = 'DELETE FROM students WHERE id = 4';
$isDelete = mysqli_query($connection, $sqlDelete);
if ($isDelete) {
    echo 'Xóa thành công';
}
else {
    echo 'Xóa thất bại';
}
mysqli_close($connection);
```

CRUD – Bài tập



- Demo chức năng CRUD với Category sử dụng Form

2. PHP – Ajax – MySQL

Khái niệm Ajax



- Ajax – Asynchronous Javascript and XML
- Tạo ra các website bất đồng bộ, load dữ liệu mà không cần load lại trang
- Giúp tăng tốc độ, sự linh hoạt, hướng đến trải nghiệm tốt nhất cho người dùng
- Nên sử dụng Ajax với thư viện jQuery hơn là Javascript thuần
- Các thư viện Javascript khác có thể xử lý tương tự như Ajax như Angular JS, Node JS, React JS...

Khai báo Ajax



```
$.ajax({  
  url: '<url>',  
  method: '<method>',  
  data: {  
    <name>: <value>  
  },  
  success: function (result) {  
    //xử lý kết quả trả về  
  }  
});
```

- url: url xử lý request ajax
- method: phương thức truyền dữ liệu (get/post)
- data: danh sách các biến gửi lên, có dạng <name>: <value>
- success: function (result): trường hợp trả về dữ liệu thành công được lưu ở biến result

Bài tập



- Ứng dụng lấy danh sách Book bằng jQuery Ajax

3. Chống SQL Injection

SQL Injection là gì?



- SQL Injection là kỹ thuật tấn công vào CSDL của ứng dụng thông qua việc khai thác các lỗ hổng bảo mật nhằm mục đích thay đổi hoặc thậm chí phá hủy hệ thống CSDL
- SQL Injection thường được thực hiện thông qua form nhập liệu để chèn vào câu truy vấn SQL

SQL Injection là gì?



- Cho form như sau: hãy tìm kiếm trong bảng students nhân viên có đúng hoặc gần đúng dựa vào name mà user nhập

Nhập tên:

- Nhập chuỗi sau vào ô tìm kiếm: `nothing' or name <> '`, kết quả là toàn bộ các bản ghi trong bảng students được show ra!

Cách phòng chống SQL Injection



- Luôn validate chặt chẽ dữ liệu từ client
- Với code thuần, luôn sử dụng hàm `mysqli_real_escape_string($connection, $query)` để lọc dữ liệu từ client
- Sử dụng parameter thay vì cộng chuỗi khi tạo câu truy vấn
- Không hiển thị exception, message thông báo lỗi
- Phân quyền rõ ràng trong Database
- Backup dữ liệu thường xuyên

4. PHP - PDO

PDO: Khái niệm



- PDO - PHP Data Objects - là lớp truy cập CSDL sử dụng các extension khác nhau để thao tác với từng hệ CSDL
- Hỗ trợ 12 loại CSDL khác nhau, khác với extension MySQLi chỉ có thể thao tác được với CSDL MySQL
- Sử dụng PDO với cơ chế bind param sẽ bảo mật hơn MySQLi sử dụng kiểu hướng thủ tục đã học

PDO: Khởi tạo kết nối



- `//CODE`

```
const DB_DSN = 'mysql:host=localhost;dbname=bt1';  
const DB_USERNAME = 'root';  
const DB_PASSWORD = "";  
try {  
    $connection = new PDO(DB_DSN, DB_USERNAME, DB_PASSWORD);  
}  
catch (PDOException $e) {  
    echo "Connection failed: " . $e->getMessage();  
}
```

PDO: Đóng kết nối



- `//CODE`
`$connection = null;`

PDO: Truy vấn Insert



- `//CODE`
`$queryInsert = $connection->prepare("INSERT INTO employees (`name`, `description`, `gender`) VALUES (?, ?, ?)");`
`$queryInsert->bindParam(1, $name);`
`$queryInsert->bindParam(2, $description);`
`$queryInsert->bindParam(3, $gender);`
`//gan gia tri`
`$name = "Anh1";`
`$description = "Desc Anh1";`
`$gender = 2;`
`//thuc thi`
`$isInsert = $queryInsert->execute();`

PDO: Truy vấn Update



- //CODE

```
$queryUpdate = $connection->prepare("UPDATE employees SET `name` = ? WHERE id = ?");
```

```
$queryUpdate->bindParam(1, $name);
```

```
$queryUpdate->bindParam(2, $id);
```

```
//gan gia tri
```

```
$name = "new name";
```

```
$id = 1;
```

```
//thuc thi
```

```
$isUpdate = $queryUpdate->execute();
```

PDO: Truy vấn Delete



- `//CODE`
`$queryDelete = $connection->prepare("DELETE FROM employees WHERE id = ?");`
`$queryDelete->bindParam(1, $id);`
`//gan gia tri`
`$id = 24;`
`//thuc thi`
`$isDelete = $queryDelete->execute();`

PDO: Truy vấn Select



- `//CODE`

```
$querySelect = $connection->prepare("SELECT * FROM employees WHERE id > ?");
```

```
$querySelect->bindParam(1, $id);
```

```
//gan gia tri
```

```
$id = 5;
```

```
//thuc thi
```

```
//gan kieu du lieu tra ve
```

```
$querySelect->execute();
```

```
$employees = $querySelect->fetchAll(PDO::FETCH_ASSOC);
```

CSE485 – Công nghệ Web

Bài 07: PHP – OOP

dungkt@tlu.edu.vn

Nội dung



- 1) Các phương pháp lập trình truyền thống
- 2) Tổng quan về OOP
- 3) Các tính chất của OOP

1. Các phương pháp lập trình truyền thống

Nội dung



- Lập trình tuyến tính (không cấu trúc)
- Lập trình thủ tục (có cấu trúc)
- Ưu điểm của lập trình hướng đối tượng

Lập trình tuyến tính



- Luồng viết code từ trên xuống dưới, không sử dụng hàm, các biến luôn là toàn cục
- Ưu điểm:
 - Code chạy nhanh vì không phải qua các bước gọi hàm hay tạo object
- Nhược điểm:
 - Code khó bảo trì, khó phát triển với các ứng dụng lớn
 - Không sử dụng hàm nên tốn công sức viết lại cho các chức năng tương tự
 - Code thiếu chuyên nghiệp, không có tính bảo mật

Lập trình tuyến tính

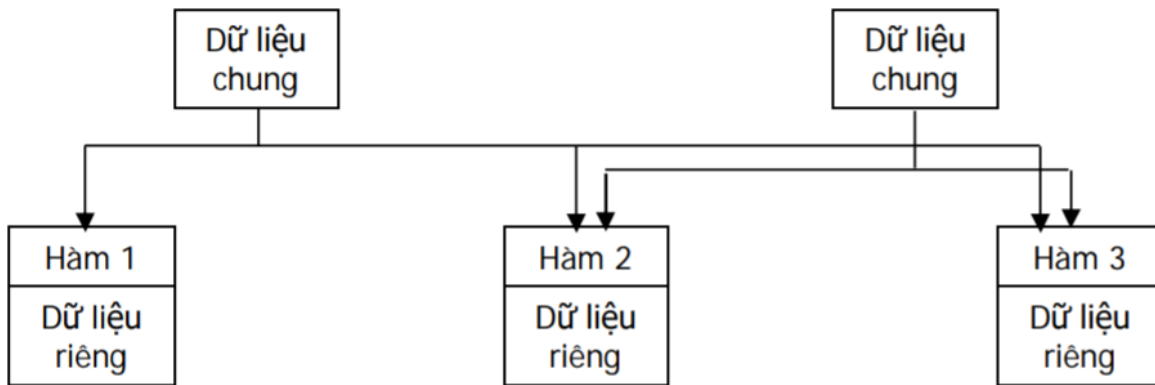


- Ví dụ: viết chương trình tính tổng 2 số nguyên
 - `$number1 = 1;`
 - `$number2 = 3;`
 - `$sum = $number1 + $number2;`
 - `echo "Tổng của 2 số nguyên $number1 + $number2 = " . $sum;`

Lập trình thủ tục [có cấu trúc]



- Chương trình được tổ chức thành các hàm tương ứng với các chức năng trên hệ thống
- Có các biến toàn cục để các hàm có thể sử dụng chung và thay đổi giá trị khi cần
- Dữ liệu được chuyển đổi qua lại thông qua các tham số gọi hàm
- Bản chất là các hàm sẽ gọi qua lại lẫn nhau



Lập trình thủ tục [có cấu trúc]



- Ưu điểm
 - Chương trình có cấu trúc rõ ràng
 - Phân chia thành các hàm độc lập nên có thể làm việc được theo nhóm
- Nhược điểm
 - Do chú trọng vào mặt chức năng mà không phải dữ liệu nên thiếu linh động trong việc xử lý dữ liệu
 - Các hàm sử dụng chung biến toàn cục dẫn tới khó kiểm soát sự thay đổi của biến

Lập trình thủ tục [có cấu trúc]



- Ví dụ

```
function connectDb(){  
function disconnectDb(){  
function addBook({  
    connectDb();  
    //code  
    disconnectDb();  
}
```

```
function editBook(){  
    connectDb();  
    //code  
    disconnectDb();  
}  
//gọi hàm
```

Ưu điểm của OOP



- Lấy đối tượng (object) làm trọng tâm thay cho chức năng (function) nên dễ dàng cài đặt và nâng cấp hơn
- Tính đóng gói và che giấu thông tin giúp dữ liệu được bảo mật
- Sử dụng đối tượng để mô hình hóa thế giới thực giúp cho việc mô tả giao diện hệ thống dễ dàng hơn
- Thông qua tính kế thừa, chương trình tránh được sự dư thừa, tăng khả năng mở rộng của hệ thống

2. Tổng quan về OOP

Nội dung



- Khái niệm
- Các thuật ngữ
- Đối tượng (Object)
- Lớp (Class)
- Phạm vi truy cập (Access Modifier)
- Thuộc tính của lớp (Member variable)
- Phương thức của lớp (Member function)
- Phương thức khởi tạo (Constructor Function)

Nội dung



- Từ khóa this
- Từ khóa static
- Từ khóa extends
- Từ khóa abstract
- Từ khóa implements

Đối tượng (Object)



- Trong thế giới thực, đối tượng là các thực thể có trạng thái và hành vi
- Trạng thái: là các thông tin, đặc điểm của đối tượng
- Hành vi: các thao tác, hành động mà đối tượng có thể thực hiện
- Trong PHP, đối tượng là 1 thể hiện của lớp (Class), có các thuộc tính (trạng thái) và phương thức (hành vi)



Đối tượng: Xe hơi Vinfast

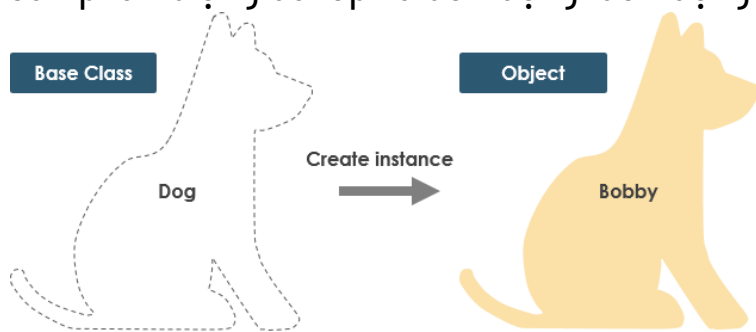
Thuộc tính: Thương hiệu, màu xe, biển số, năm sản xuất

Phương thức: Tăng ga, về số, phanh

Lớp (Class)



- Là kiểu dữ liệu trừu tượng bao gồm các phương thức và thuộc tính đã được định nghĩa từ trước
- Lớp giống như 1 khuôn mẫu của đối tượng
- Cần phân biệt giữa lớp và đối tượng: đối tượng được khởi tạo (instance) từ lớp



Properties	Methods
Color	Sit
Eye Color	Lay Down
Height	Shake
Length	Come
Weight	

Property Values	Methods
Color: Yellow	Sit
Eye Color: Brown	Lay Down
Height: 17 in	Shake
Length: 35 in	Come
Weight: 24 pounds	

object

Một con mèo có trạng thái: tên, màu lông, giống và hành vi: kêu, ăn uống



instance

Con mèo có
-Trạng thái: Tên: Mimi, màu lông: Vàng nâu, giống: Mèo thuần chủng
-Hành vi: kêu: meo meo, ăn uống: ăn thức ăn dành cho mèo

class

ConMeo
Ten
Mau_long
Giống
kêu()
an_uong()

Lớp (Class)



- Khai báo lớp sử dụng từ khóa class

```
class Cat {  
//Khai báo các thuộc tính  
    private $name;  
    private $color;  
    private $type;
```

```
//Khai báo các phương thức  
    public function eat() {  
        //code  
    }  
  
    public function run(){  
        //code  
    }  
}
```

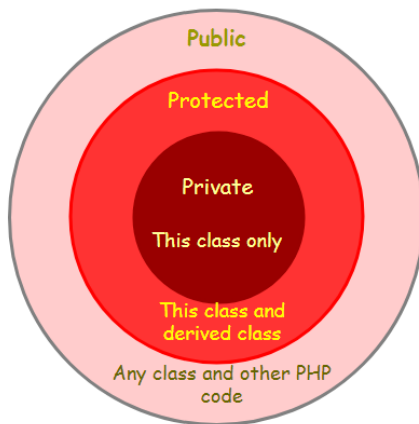
- Khởi tạo đối tượng của class bằng từ khóa new

```
$cat1 = new Cat();  
$cat2 = new Cat();
```

Phạm vi truy cập



- Từ khóa đặt trước tên thuộc tính hoặc phương thức lớp:
 - private: chỉ truy cập được bên trong class
 - protected: có thể truy cập được từ các lớp con (tính kế thừa trong OOP)
 - public: được truy cập từ mọi nơi



Thuộc tính của lớp



- Là các thuộc tính được khai báo bên trong class, có quyền truy cập tương ứng (private, protected, public)
- Khác với biến thông thường, biến lớp cần truy cập qua object để có thể sử dụng
- Cú pháp truy cập: `objectName->propertyName`

```
class Student {  
    public $name;  
    private $birthday;  
}  
$student1 = new Student();  
$student1->name = 'Anh';  
echo "Họ tên của sinh viên 1: " . $student1->name;  
//dòng sau báo lỗi  
$student1->birthday = '22/10/2000';
```

Phương thức của lớp



- Là các hàm được khai báo bên trong class, có quyền truy cập tương ứng
- Khác với hàm thông thường, phương thức lớp cần truy cập qua object để sử dụng
- Cú pháp truy cập: `objectName->methodName`

```
class Student {  
    public $name;  
    public function setName($name) {  
        $this->name = $name;  
    }  
    private function isPrivate() {}  
}
```

```
$student = new Student();  
$student->setName("Nguyễn Văn Anh");  
echo "Tên sinh viên: " . $student->name;  
//dòng sau báo lỗi  
$student->isPrivate();
```

Phương thức khởi tạo



- Phương thức tự động được gọi đầu tiên khi tạo object

```
class Student
```

```
{
```

```
    public function __construct()
```

```
{
```

```
    echo 'Hàm khởi tạo được chạy tự động khi khởi tạo đối tượng';
```

```
}
```

```
}
```

```
$student1 = new Student();
```

Toán tử this



- Tham chiếu đến đối tượng hiện tại
- Sử dụng \$this bên trong phương thức của lớp cũng tương tự như khi sử dụng tên đối tượng ở ngoài

```
lớp class Student {  
    private $birthday;  
  
    public function setBirthday($birthday){  
        //biến $this đại diện cho class hiện tại  
        $this->birthday = $birthday;  
    }  
    public function getBirthday() {  
        //biến $this đại diện cho class hiện tại  
        return $this->birthday;  
    }  
}  
  
$student = new Student();  
$student->setBirthday("22/10/2012");
```


Từ khóa static



- Hoạt động như 1 biến toàn cục trong PHP
- Thuộc tính/phương thức static không thể được truy xuất thông qua đối tượng, mà sử dụng toán tử :: với tên class hoặc từ khóa self, tùy thuộc vào phạm vi bên ngoài hay trong class
- Không thể sử dụng \$this trong phương thức static
- Cách khai báo: thêm từ khóa static vào sau phạm vi truy cập của thuộc tính/phương thức

Từ khóa static



```
class People
```

```
{
```

```
    public static $name;
```

```
    public static function setName()
```

```
    {
```

```
        self::$name++;
```

```
    }
```

```
    public static function getName()
```

```
    {
```

```
        return self::$name;
```

```
    }
```

```
}
```

```
echo People::$name;
```

```
People::setName();
```

```
echo People::getName();
```

Từ khóa **extends**



- Được sử dụng khi khai báo class, thể hiện cho tính kế thừa trong OOP
- PHP chỉ hỗ trợ đơn kế thừa
- Lớp con sẽ kế thừa toàn bộ phương thức và thuộc tính không private từ lớp cha
- Ngoài ra lớp con vẫn có phương thức và thuộc tính riêng của nó

Từ khóa abstract



- Được sử dụng khi khai báo class, thể hiện cho tính trừu tượng trong OOP
- Thường dùng với mục đích tạo 1 khung ban đầu cho các đối tượng
- Không thể khởi tạo đối tượng từ lớp trừu tượng
- Lớp trừu tượng có thể có các phương thức trừu tượng – phương thức không có nội dung, được khai báo cũng với từ khóa abstract
- Lớp extends từ lớp trừu tượng bắt buộc phải định nghĩa là các phương thức trừu tượng của nó

Từ khóa implements



- Đặc trưng cho việc thực thi các interface
- Interface được xây dựng như là 1 bộ khung các phương thức không có nội dung, ngoài ra không thể khai báo biến trong interface
- Một class có thể thực thi nhiều interface
- Class mà thực thi interface nào thì bắt buộc phải định nghĩa lại toàn bộ các phương thức của interface đó.

3. Các tính chất của OOP

Nội dung



- Tính trừu tượng (Abstraction)
- Tính đóng gói (Encapsulation)
- Tính kế thừa (Inheritance)
- Tính đa hình (Polymorphism)

Tính trừu tượng (Abstraction)



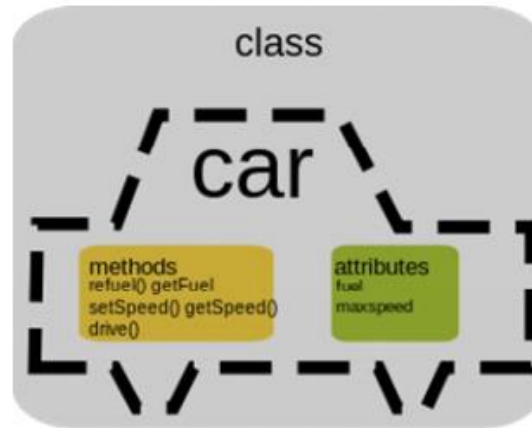
- Các đối tượng giống nhau về các thuộc tính và phương thức có thể được chọn lọc để trừu tượng hóa lên thành 1 lớp



Tính đóng gói (Encapsulation)



- Cho phép che giấu thông tin của lớp với các lớp bên ngoài khi cần thiết
- Phạm vi truy cập trong (private, protected, public) là một thể hiện của tính đóng gói trong OOP



Tính kế thừa (Inheritance)



- Cho phép xây dựng lớp mới dựa trên (kế thừa) từ lớp có sẵn
- Lớp có sẵn gọi là lớp cha, lớp kế thừa từ lớp cha gọi là lớp con
- Lớp con kế thừa tất cả phương thức/thuộc tính từ lớp cha, ngoài ra lớp con có thể có thêm các phương thức/thuộc tính của riêng nó
- Trong OOP từ khóa thể hiện tính kế thừa là extends

Tính đa hình (Polymorphism)



- Thể hiện ở việc các lớp chia sẻ chung (implements) từ 1 interface – bộ khung các phương thức để các lớp mà implements từ nó bắt buộc phải thực thi
- Hiểu đơn giản về tính đa hình là với cùng 1 phương thức được khai báo trong interface, các class implements sẽ sử dụng phương thức đó cho từng mục đích khác nhau.

CSE485 – Công nghệ Web

Bài 08: PHP – MVC

dungkt@tlu.edu.vn

Nội dung



- 1) Khái niệm
- 2) Thành phần
- 3) Luồng xử lý dữ liệu
- 4) Cấu trúc thư mục
- 5) Bài tập

Khái niệm



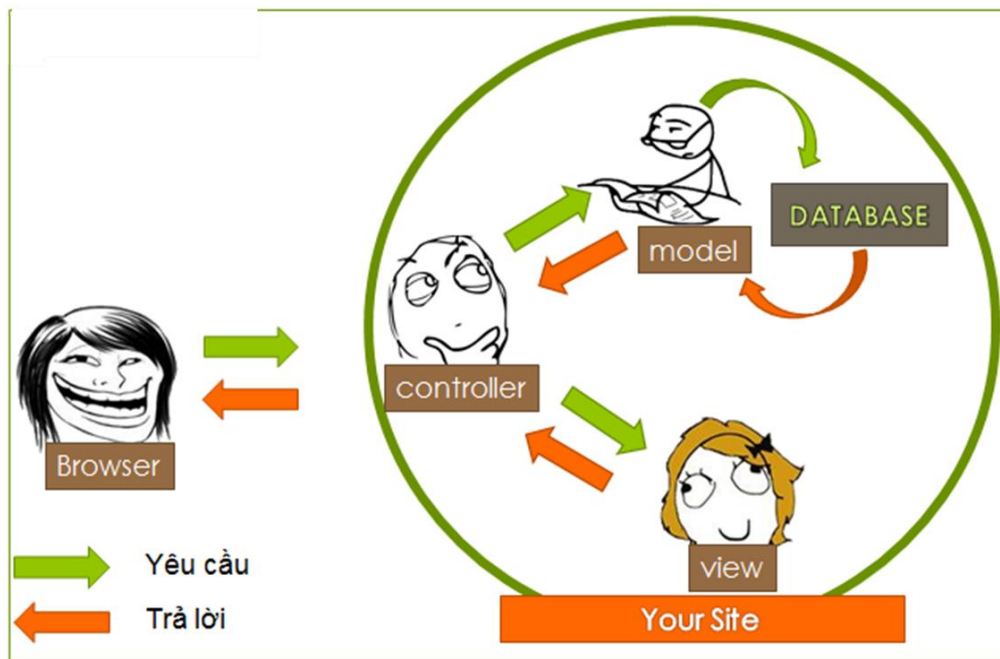
- Là mô hình kiến trúc phần mềm 3 lớp bao gồm Model, View, Controller
- Tách biệt ứng dụng web ra làm các thành phần riêng biệt, nên thuận lợi cho việc phát triển và bảo trì
- Phổ biến trong các framework hiện nay (Laravel, Zend, Cake, .v.v)
- Sử dụng OOP làm nền tảng để xây dựng và phát triển

Thành phần



- M – Model: nhận dữ liệu từ Controller gửi tới, thực hiện thao tác với database, gửi kết quả trả về Controller
- V – View: hiển thị dữ liệu từ Controller gửi về cho trình duyệt tại client
- C – Controller: tầng trung gian nhận request từ client, gọi Model thực thi, trả dữ liệu lại cho client thông qua View, View định dạng dữ liệu hiển thị cho client

Luồng xử lý dữ liệu

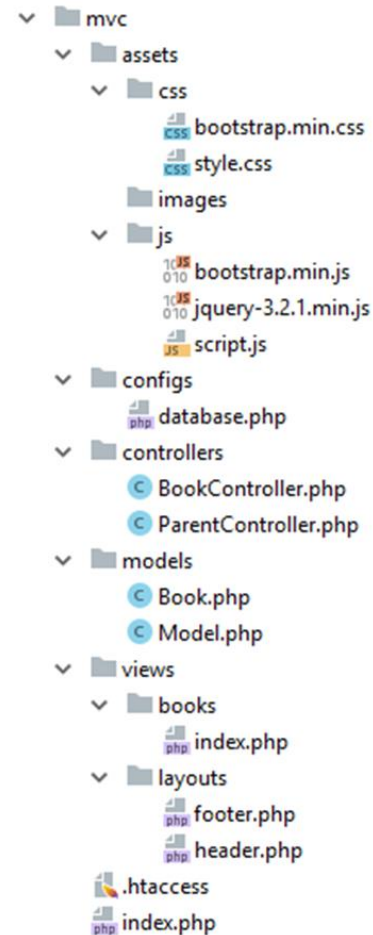


Luồng xử lý dữ liệu



- Ví dụ: Client truy cập link <http://cse.tlu.edu.vn/chi-tiet/3>
 - Controller nhận request chứa id của bài viết = 3
 - Controller gọi Model truy vấn database lấy ra nội dung bài viết có id = 3
 - Model truy vấn thành công, gửi nội dung bài viết lại cho Controller
 - Controller gửi nội dung bài viết đó ra View
 - View hiển thị nội dung bài viết tới client

Cấu trúc thư mục Code theo MVC



CSE485 – Công nghệ Web

Bài 09: PHP – Laravel

dungkt@tlu.edu.vn

Nội dung



- 1) Laravel là gì
- 2) Khái niệm Composer
- 3) Cài đặt Laravel sử dụng Composer
- 4) Cấu trúc thư mục Laravel
- 5) Luồng xử lý trong Laravel
- 6) File môi trường .env
- 7) Routing
- 8) Middleware
- 9) Namespace

Nội dung



- 10) Controller
- 11) Request
- 12) Cookie
- 13) Session
- 14) Views
- 15) Blade Template
- 16) Redirect
- 17) Làm việc với Database
- 18) Laravel Form
- 19) Validation Form

Laravel là gì?



- Là framework mã nguồn mở, hoạt động theo mô hình MVC
- Được sử dụng phổ biến nhất trong các framework hiện tại
- Trang chủ: <https://laravel.com>
- Document: <https://laravel.com/docs/7.x>

Composer là gì?



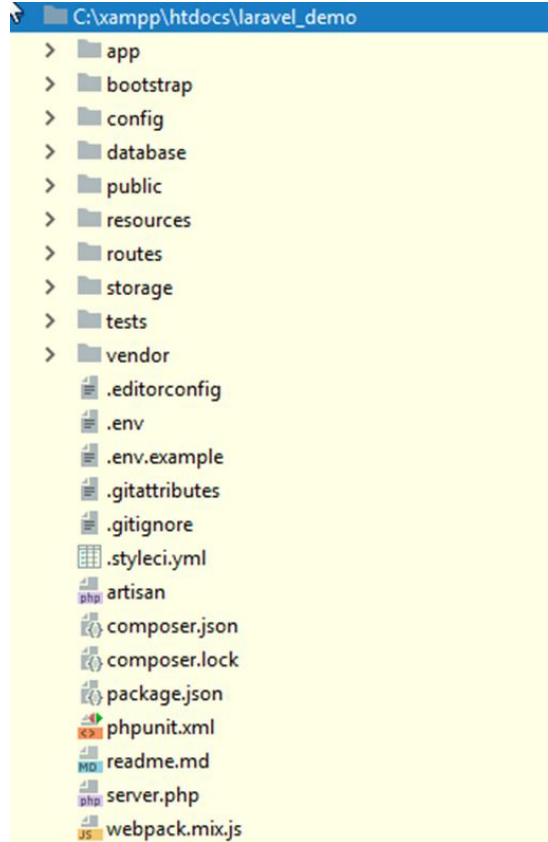
- Tải về và cài đặt tại <https://getcomposer.org/>
- Công cụ quản lý việc cài đặt, update tự động các thư viện code của bên thứ ba
- Là công cụ không thể thiếu đối với lập trình
- Laravel sẽ được cài đặt thông qua Composer

Cài đặt Laravel với Composer

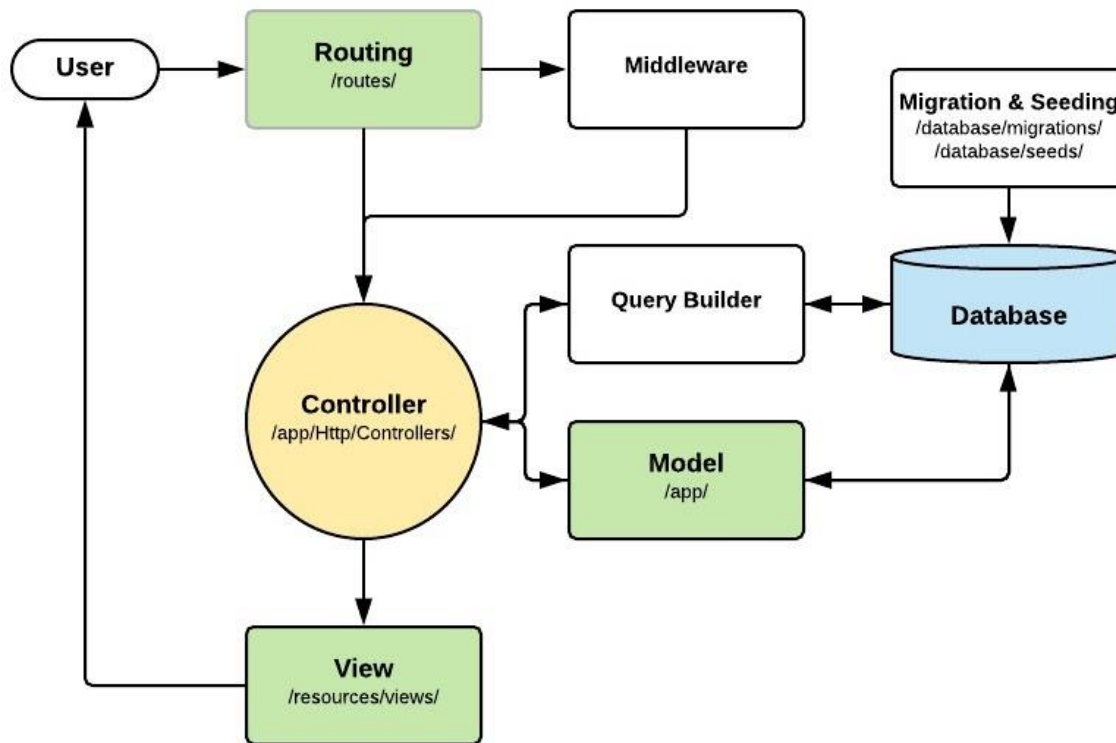


- Cách 1: chạy lần lượt 2 lệnh sau:
 - `composer global require laravel/installer`
 - `laravel new <tên-thư-mục-project>`
- Cách 2:
 - `composer create-project --prefer-dist laravel/laravel <tên-thư-mục-project>`

Cấu trúc thư mục Laravel



Luồng xử lý trong Laravel



Tệp tin môi trường .env



- File chứa các thông tin quan trọng liên quan đến cấu hình hệ thống và database
- Khi thay đổi thông tin cấu hình DB sẽ thay đổi tại file .env này mà không cần vào trong thư mục config/database.php
- Với CSDL MySql trên local sẽ thay đổi các thông số sau

DB_CONNECTION=mysql

DB_HOST=127.0.0.1

DB_PORT=3306

DB_DATABASE=tên-CSDL

DB_USERNAME=root

DB_PASSWORD=

Routing trong Laravel



- Trong Laravel, mọi url đều phải thông qua cơ chế Routing
- Cơ chế routing tương tự như khi làm việc với file .htaccess
- Các route nằm tại routes/web.php
- Route có thể có tham số hoặc không
- Ví dụ:

```
Route::get('demo-page', function () {  
    return view('demo-page');  
});  
Route::get('param2/{name}/{age}', function ($name, $age) {  
    echo "Name = $name <br />";  
    echo "Age = $age <br />";  
});  
Route::get('/home', 'HomeController@index');
```

Cơ chế Middleware



- Đóng vai trò như 1 firewall, lọc request gửi tới và response trả về
- Lệnh tạo với artisan: `php artisan make:middleware <tên-middleware>`
- Sau khi tạo, đăng ký middleware tại `app/Http/Kernel.php`
- Có thể sử dụng middleware khi tạo route hoặc tại hàm khởi tạo của controller
- Ví dụ đặt tại route của middleware FirstMiddleware

```
Route::get('demo-middleware/{id}', function ($id) {  
    echo "Đã pass qua middleware FirstMiddleware";  
})->middleware('first');
```

Namespace



- Được sử dụng như 1 định danh duy nhất cho 1 file, giải quyết trường hợp import các file trùng tên trong cùng 1 ứng dụng
- Laravel sử dụng từ khóa use để import class dựa theo namespace, thay vì phải import các file sử dụng các hàm include/require trong MVC thuần
- VD: import class Request có namespace là Illuminate\Http
use Illuminate\Http\Request;

Controller



- Ví trí tại app/http/Controllers
- Tạo với artisan:
 - `php artisan make:controller <tên-controller>`
- Tạo controller với các phương thức CRUD chuẩn của Laravel
 - `php artisan make:controller <tên-controller> --resource`

Class Request



- Chứa toàn bộ các thông tin của request gửi đến
- Sử dụng thông qua đối tượng của lớp Request hoặc phương thức request()
- Lấy thông tin liên quan đến url
 - `request()->path();` //lấy đường dẫn
 - `$request->path();` //lấy đường dẫn
 - `$request->url();` //lấy url, bỏ qua tham số truy vấn nếu có
- Lấy thông tin từ form, có các cách lấy sau
 - `$request->input('username');`
 - `$request->username;`
- Hai cách trên đều dùng để lấy giá trị input form có name là username

Session



- Nên sử dụng các hàm thao tác với session mà Laravel cung cấp, thay vì sử dụng biến `$_SESSION` của PHP
 - Set giá trị : `session()->put(key, value);`
 - Lấy giá trị : `session()->get(key);`
 - Hiển thị toàn bộ session trên hệ thống : `session()->all();`
 - Xóa session : `session()->forget(key);`
 - Xóa toàn bộ session trên hệ thống : `session()->flush();`

Cookie



- Nên sử dụng các hàm thao tác với cookie mà Laravel cung cấp, thay vì sử dụng biến `$_COOKIE` của PHP
- Set giá trị:
 - `$cookie = cookie('name', '123', '30');`
 - `Cookie::queue($cookie);`
- Lấy giá trị:
 - `Cookie::get('name');`

Views



- Vị trí nằm tại resources/views
- Laravel sử dụng engine blade cho view, với đuôi file là .blade.php
- Cú pháp trong view blade:
 - Hiển thị dữ liệu: `{{ <tên-biến> }}`
 - VD: `{{ $abc }}`
 - Các cấu trúc điều khiển
 - `@if - @else - @endif`, `@foreach - @endforeach`, `@for - @endfor`, `@while - @endwhile`
 - Viết code PHP trong file blade: `@php(<code-php>)`
 - VD: `@php($a = 5; $a++)`

Views - Layout



- Thường được khai báo tại resource/views/layouts
- Ví dụ tạo layouts resource/views/layouts/master.blade.php
- @yield: có thể hiểu như 1 tham số, khi views nào extends từ layouts này thì lúc đó mới set giá trị thực

```
<html>
  <head>
    <title>Laravel - @yield('title')</title>
  </head>
  <body>
    <div class="header">This is my header</div>
    <div class="main-content">
      @yield('content')
    </div>
    <div class="footer">This is my header</div>
  </body>
</html>
```

Views - Layout



- Tạo views kế thừa layout sử dụng từ khóa `@extends`
- Ví dụ: tạo view `child.blade.php` extends từ `layouts master.blade.php` từ ví dụ trước
- `@section`: cú pháp tại lớp con để set giá trị, hay có thể hiểu là nó thay thế cú pháp `@yield` từ bên layout

```
@extends('layouts.master')
@section('title', 'This is title')
@section('content')
    Content của tôi
@endsection
```

Redirect



- Chuyển hướng người dùng dựa theo route
- Có các phương thức sau
 - `redirect('<tên-route>');`
- VD: chuyển hướng về link /home
 - `redirect('home');`
 - `redirect()->route('<route_name>');`
- Với cách này cần set name cho route

Làm việc với Database



- Laravel cung cấp 2 cơ chế thao tác với Database
 - QueryBuilder: gần với truy vấn thô, tốc độ nhanh hơn, sử dụng class façade DB theo hướng PHP thuần
 - Eloquent ORM: cách viết đẹp và rõ ràng hơn, có tính bảo mật cao hơn, sử dụng Model theo hướng MVC
- Tùy mục đích mà có thể sử dụng cơ chế nào cho phù hợp
- Tạo model với artisan: `php artisan make:model <tên-model>`
 - VD: `php artisan make:model News`

Truy vấn QueryBuilder - Eloquent



QueryBuilder	Eloquent ORM
Lấy tất cả bản ghi	
<code>\$news = DB::table('news')->get();</code>	<code>\$news = News::all();</code>
Lấy 1 bản ghi theo id	
<code>\$new = DB::table('news')->where('id', 1)->first();</code>	<code>\$new = News::find(1);</code>
Lấy 1 trường của 1 bản ghi	
<code>\$new = DB::table('news')->where("id", 1)->select('title')->first();</code>	<code>News::find(1)->select('title')->first();</code>
Lấy 1 trường của tất cả bản ghi	
<code>\$new = DB::table('news')->select("title")->get();</code>	<code>\$news = News::all('title');</code>

Truy vấn QueryBuilder - Eloquent



QueryBuilder	Eloquent ORM
Insert bản ghi	
<pre>\$isInsert = DB::table('news')->insert(['title' => 'new']);</pre>	<pre>\$newModel = new News(); \$newModel->title = "Title insert"; \$newModel->save();</pre>
Update bản ghi	
<pre>\$isUpdate = DB::table('news')->where('id', 1)->update(['title' => 'title new']);</pre>	<pre>\$news = News::find(1); \$news->title = "Title update"; \$news->save();</pre>
Xóa bản ghi	
<pre>\$isDelete = DB::table('news')->where('id', 1)->delete();</pre>	<pre>\$news = News::find(1); \$news->delete();</pre>
Hàm count và max	
<pre>\$count = DB::table('news')->count(); \$max = DB::table('news')->max('id');</pre>	<pre>\$count = News::count(); \$max = News::max('id');</pre>

Validate Form



- Laravel xử lý validate dựa vào class Request hoặc phương thức request()
- Form trong Laravel bắt buộc phải có trường ẩn sau, để tránh lỗi bảo mật CSRF
 - `<input type="hidden" name="_token" value="{{ csrf_token() }}" />`
- Xử lý validate trong Laravel tại Controller
- Ví dụ sau validate username và password không được để trống, username chỉ cho phép độ dài lớn nhất là 8 ký tự

```
public function validateForm(Request $request) {  
    $this->validate($request, [  
        'username' => 'required|max:8',  
        'password' => 'required'  
    ]);  
}
```