LÀM QUEN VỚI MÔI TRƯỜNG

PHẦN 1

Nội dung bài học

- Giới thiệu về lập trình web với PHP
- Hướng dẫn viết mã cho ứng dụng PHP
- Hướng dẫn kiểm thử và gỡ lỗi cho ứng dụng PHP

1. Giới thiệu về lập trình web với PHP

Trong phần này có các nội dung:

1.1. Kiến trúc của một ứng dụng web

1.2. Cách thức xử lý trang web tĩnh

1.3. Cách thức xử lý trang web động

1.4. Tổng quan về các phần mềm web

1.5. Giới thiệu môi trường làm việc PHP

1.6. Giới thiệu phần mềm NetBeans

1.7. Quy trình xây dựng một ứng dụng PHP đơn giản

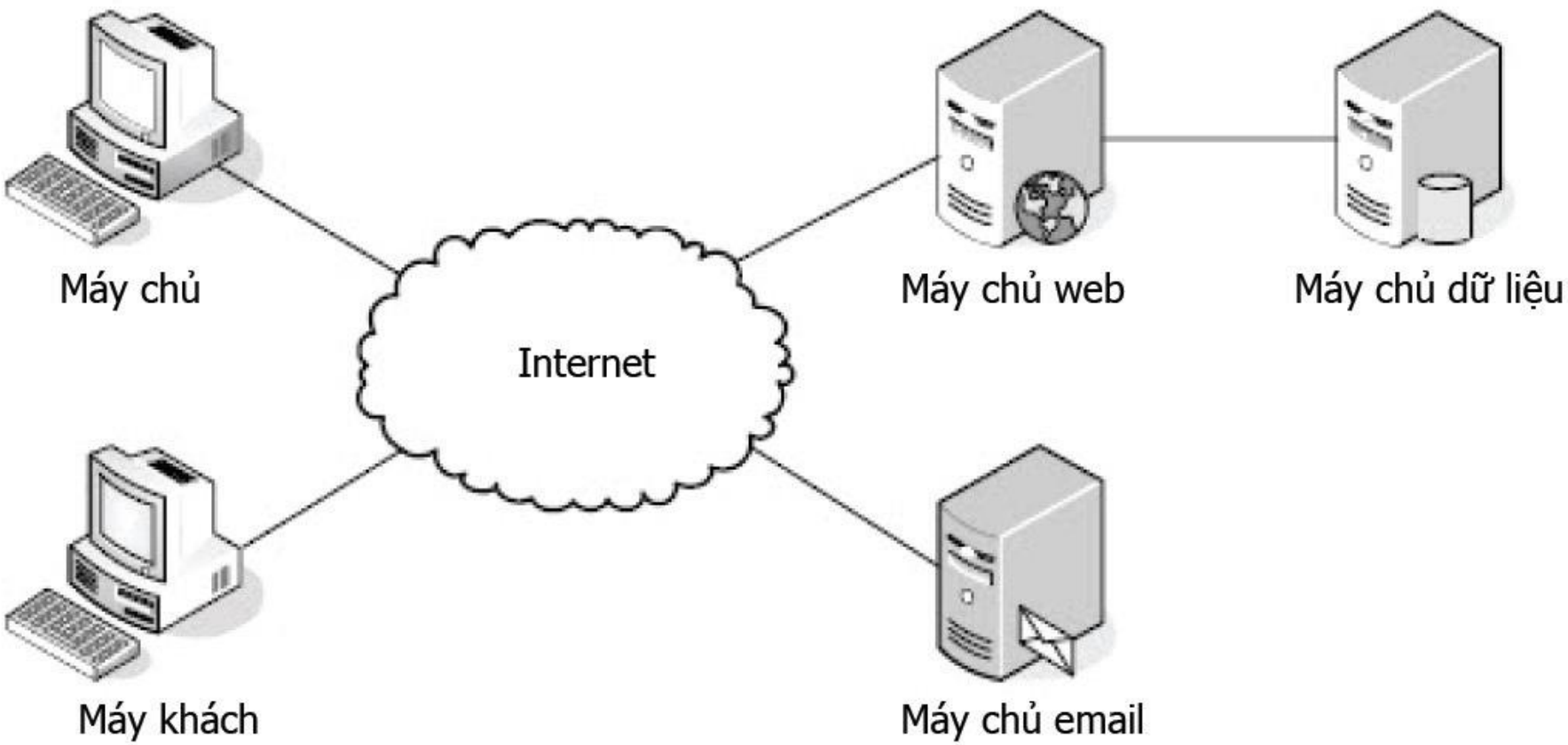
1.1. Kiến trúc của ứng dụng web

- Ứng dụng web sử dụng mô hình client – server (khách – chủ). Máy chủ có thể chia sẻ file, máy in, cơ sở dữ liệu hoặc e-mail cho các máy khách

- Một số khái niệm cơ bản:
  - **Máy chủ web** (web server): máy chủ chuyên chia sẻ trang web
  - **Trình duyệt web** (web browser): ứng dụng phía người dùng để kết nối với máy chủ web, lấy thông tin từ máy chủ và hiển thị thông tin trên cửa sổ trình duyệt
  - **Mạng** (network): hệ thống trao đổi thông tin cho phép máy khách và máy chủ có thể giao tiếp với nhau

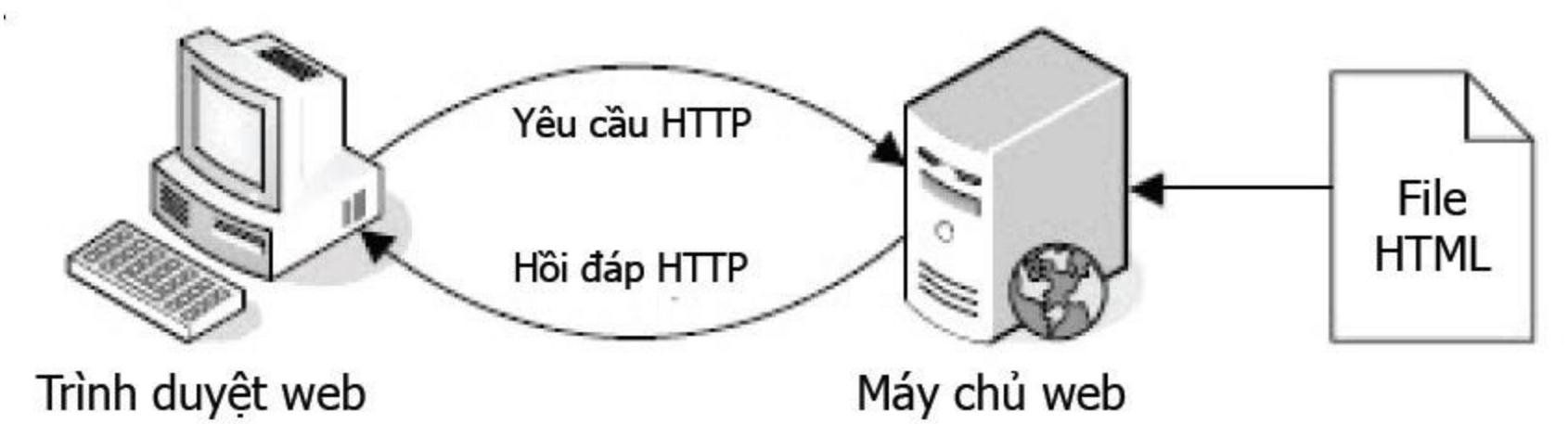
Kiến trúc của ứng dụng web

Minh họa kiến trúc ứng dụng web:



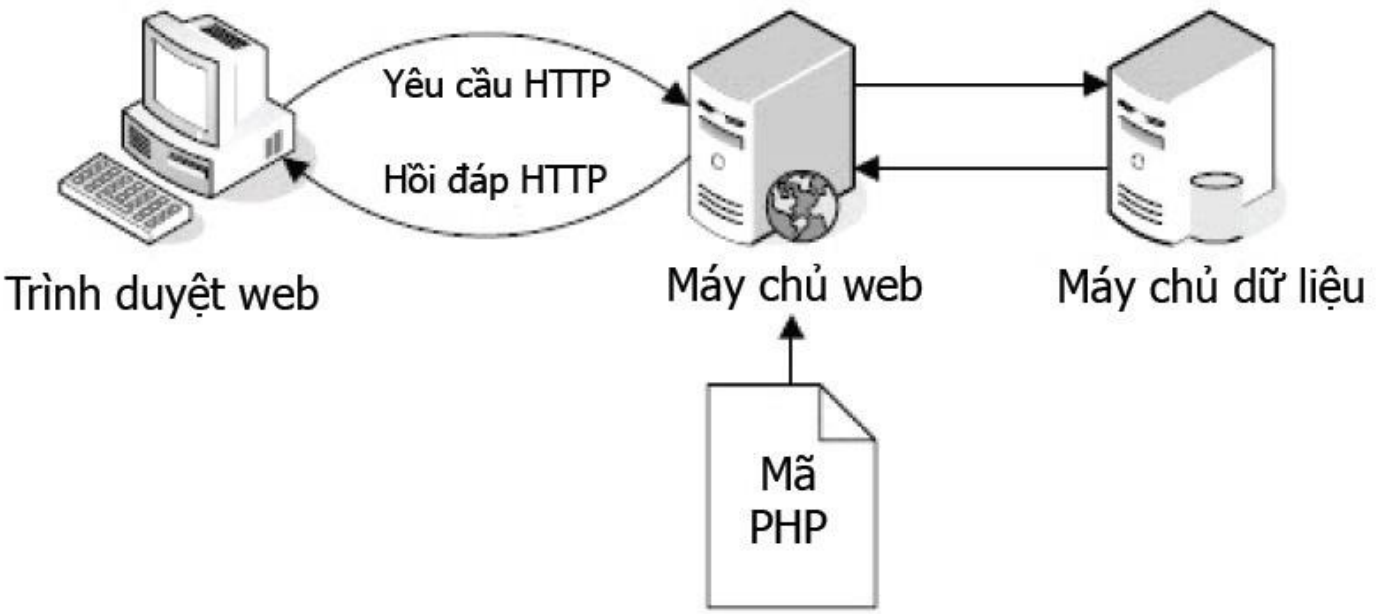
1.2. Cách thức xử lý trang web tĩnh

- Trang web tĩnh (static web page): trang web chỉ thay đổi nội dung khi có sự tác động của người tạo ra nó
- Quy trình xử lý trang web tĩnh:



1.3. Cách thức xử lý trang web động

- Trang web động (dynamic web page) là trang web được tạo bởi chương trình hoặc mã kịch bản (script) chạy trên máy chủ
- Nội dung của trang web động có thể thay đổi mỗi lần được yêu cầu
- Cách thức xử lý trang web động (giả sử script là PHP):



1.4. Tổng quan về các phần mềm web

Trình duyệt web	
Internet Explorer	Phát hành bởi Microsoft. Hiện chỉ có phiên bản trên Windows.
Firefox	Phát hành bởi Mozilla Corporation. Có tất cả các phiên bản trên các hệ điều hành chính như Windows, MacOS, Linux
Safari	Phát hành bởi Apple. Có phiên bản trên OSX và Windows.
Opera	Phát hành bởi Opera Software. Có tất cả các phiên bản trên các hệ điều hành phổ biến và có một phiên bản rút gọn rất được ưa dùng trên điện thoại di động và PDA.
Chrome	Phát hành bởi Google. Hiện chỉ có phiên bản trên Windows

Tổng quan về các phần mềm web

Máy chủ web	
Apache	Là máy chủ web mã nguồn mở có thể được vận hành trên bất cứ hệ điều hành phổ dụng nào hiện nay. Apache hỗ trợ nhiều ngôn ngữ kịch bản phía máy chủ và có thể tương tác với nhiều máy chủ dữ liệu khác nhau. Bộ tứ hoàn hảo nhất được biết đến là LAMP, nghĩa là Linux, Apache, MySQL và PHP
IIS	Là máy chủ web do Microsoft phát hành và chỉ chạy trên Windows. Mục đích chính là hỗ trợ ASP.NET và MS SQL Server

Tổng quan về các phần mềm web

Ngôn ngữ phía server	
PHP	Thường được sử dụng với Apache nhưng hiện cũng được IIS hỗ trợ, xử lý các file có đuôi .php.
ASP.NET	Sử dụng bởi Microsoft IIS. ASP.NET sử dụng tên đuôi .aspx và làm việc chủ yếu với các ứng dụng được viết trên C# hoặc Visual Basic
Pearl	Được phát triển cho mục đích xử lý văn bản bằng dòng lệnh của UNIX và viết các ứng dụng web. Perl sử dụng đuôi .pl
Python	Được dùng để phát triển nhiều loại ứng dụng trong đó có ứng dụng web. Python được sử dụng chủ yếu với Apache và có đuôi là .py.
JSP	Cần thêm một máy chủ ứng dụng như Tomcat server, được phát triển miễn phí bởi Công ty Apache Software Foundation. Các trang JSP sử dụng đuôi .jsp và chủ yếu làm việc với servlet được viết trên ngôn ngữ Java

Tổng quan về các phần mềm web

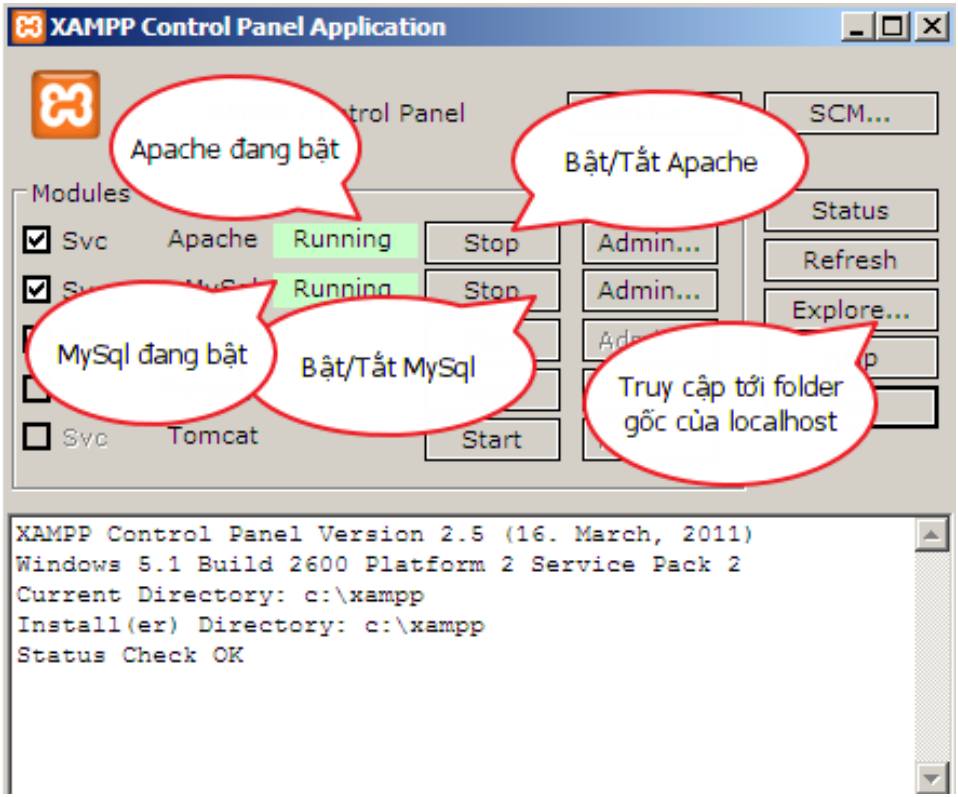
Máy chủ cơ sở dữ liệu	
MySQL	Cơ sở dữ liệu mã nguồn mở, sẵn dùng cho tất cả các hệ điều hành phổ biến
Oracle	Máy chủ cơ sở dữ liệu của Oracle, sẵn dùng cho tất cả các hệ điều hành phổ biến
DB2	Máy chủ cơ sở dữ liệu của IBM, sẵn dùng cho tất cả các hệ điều hành phổ biến
MS SQL	Máy chủ cơ sở dữ liệu của Microsoft, chỉ chạy trên HĐH Windows

1.5. Môi trường lập trình PHP

- Môi trường lập trình PHP thường gồm các phần mềm:
  - PHP: để xử lý mã lệnh PHP
  - MySQL: để quản trị cơ sở dữ liệu
  - Apache: máy chủ web để chạy các ứng dụng PHP sau khi lập trình trên máy tính cá nhân
- Tất cả các phần mềm trên được gói gọn trong gói phần mềm XAMPP
- Ngoài ra còn có:
  - Phần mềm soạn thảo: để viết mã cho ứng dụng PHP
  - Phần mềm dò lỗi: để dò lỗi PHP trong quá trình phát triển ứng dụng

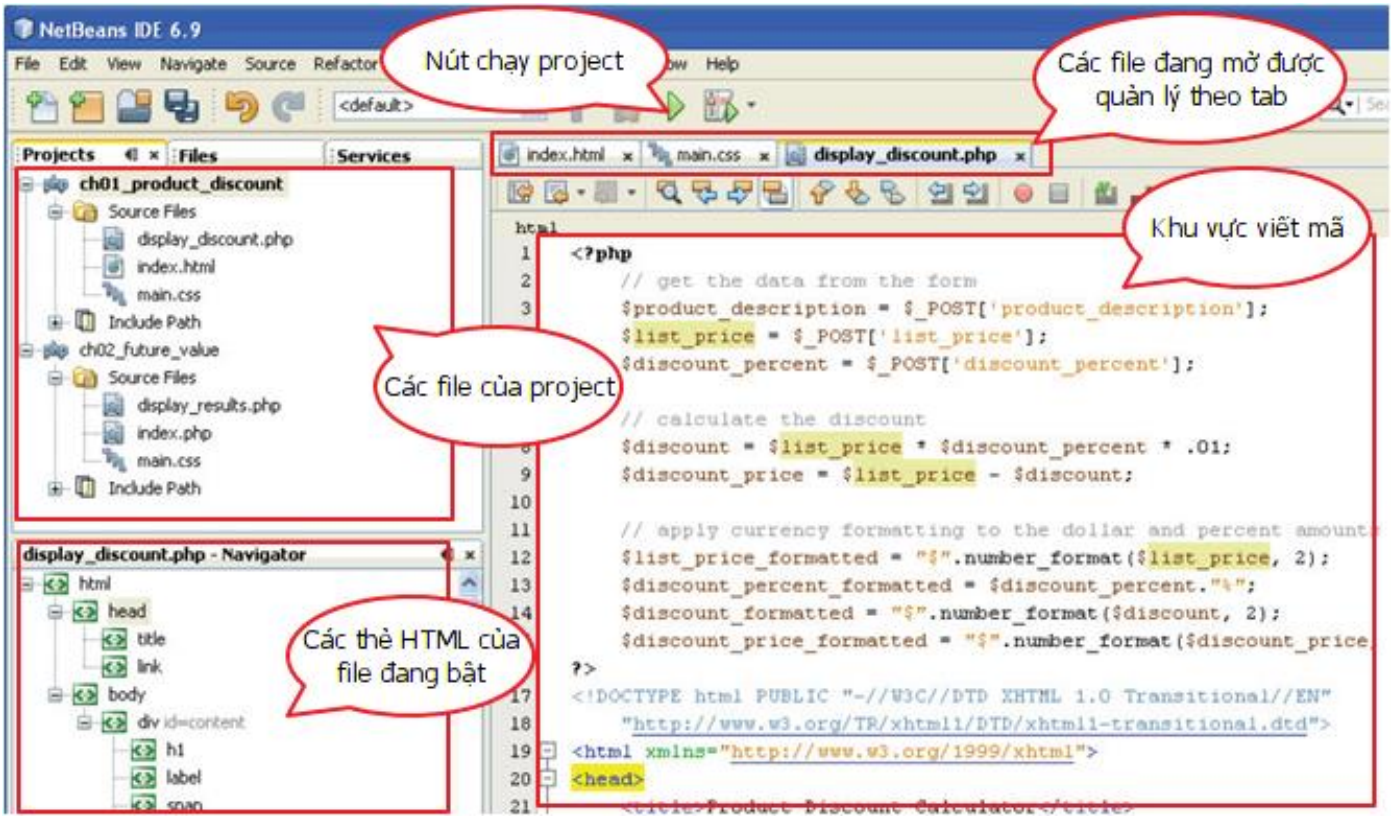
Môi trường lập trình PHP

- Giao diện điều khiển của XAMPP:
  - Bật/Tắt Apache
  - Bật/Tắt MySql
  - Truy cập tới folder gốc của localhost, là nơi bạn upload website lên đó



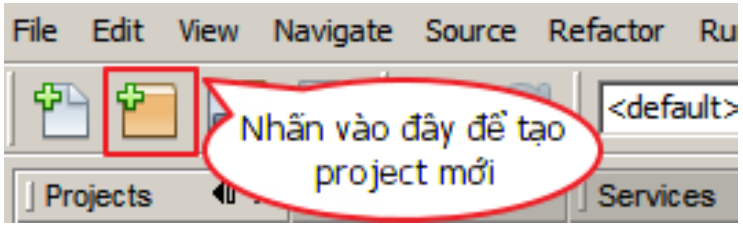
1.6. Giới thiệu phần mềm NetBeans

- Là phần mềm cung cấp môi trường phát triển ứng dụng PHP chuyên nghiệp: quản lý mã nguồn, soạn thảo mã, dò lỗi, …
- Mỗi ứng dụng PHP được coi là một project (dự án) trong NetBeans
- Giao diện chính của NetBeans:

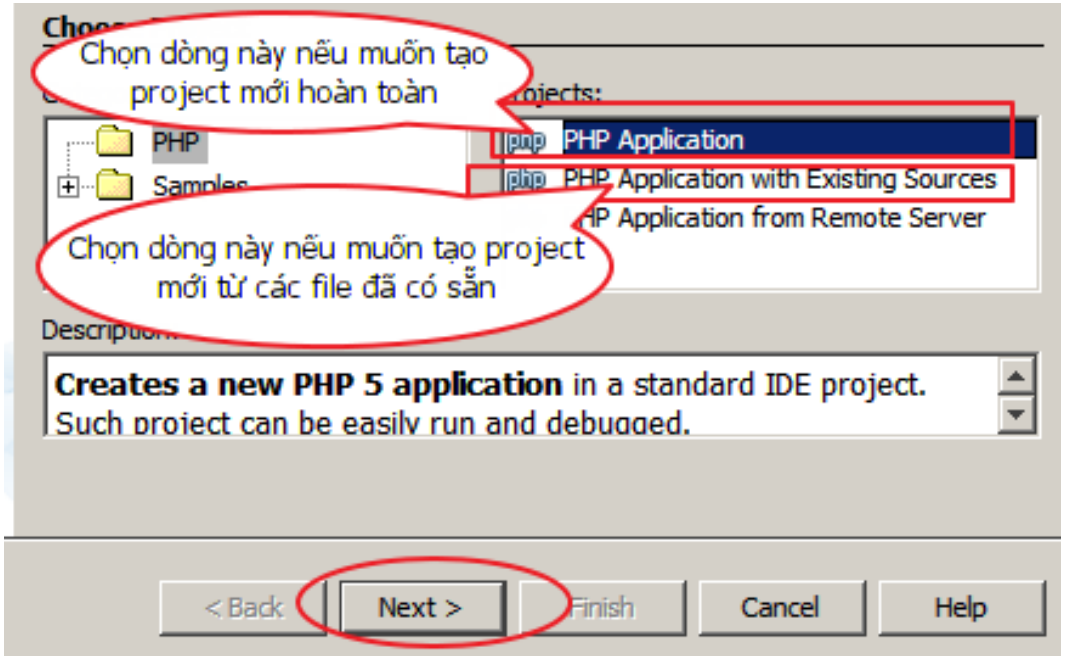


Giới thiệu phần mềm NetBeans

- Tạo một project mới:
  - Bước 1: Nhấn nút New Project

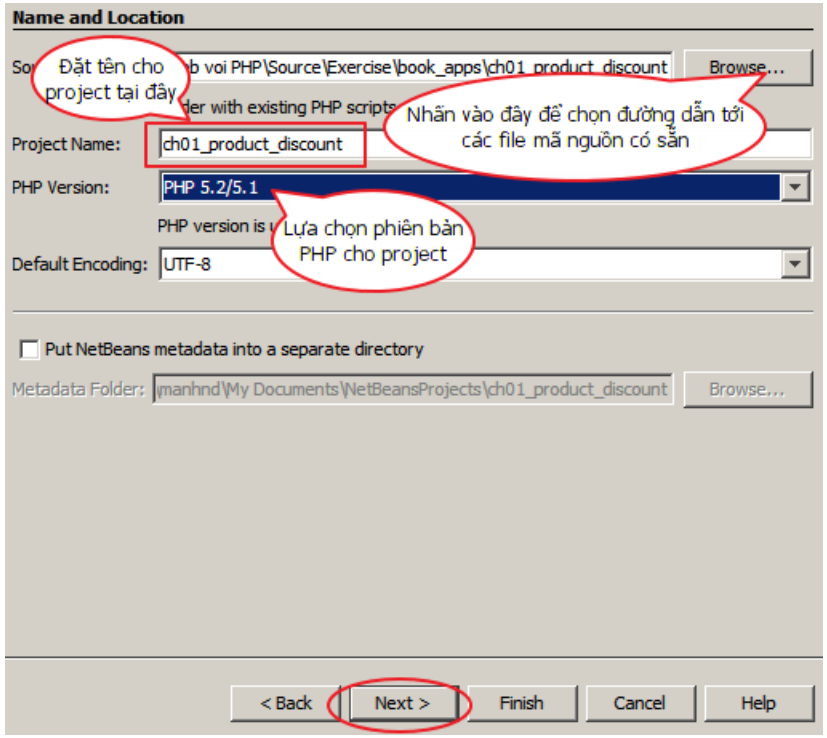


  - Bước 2: Lựa chọn tạo project mới hoàn toàn (chưa có file nào) hoặc tạo project mới từ các file có sẵn



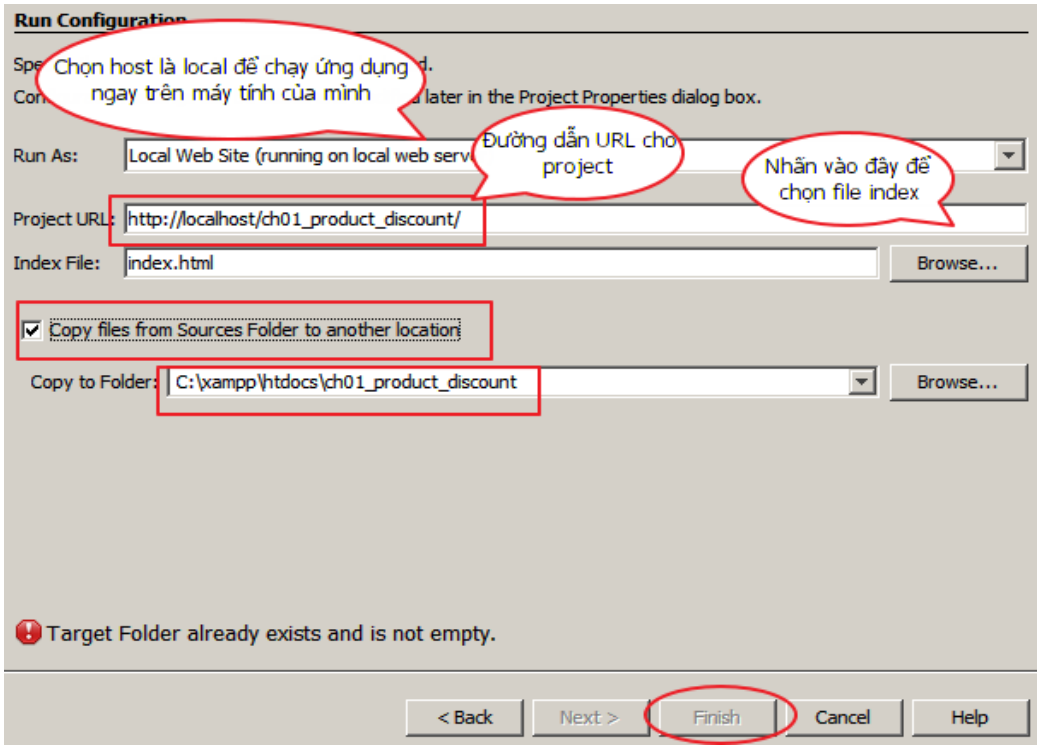
Giới thiệu phần mềm NetBeans

- Bước 3: Chọn các thông số cần thiết
  - Nếu project mới tạo từ các file có sẵn thì phải chọn đường dẫn tới đó
  - Điền tên project cần tạo
  - Chọn phiên bản PHP phù hợp



Giới thiệu phần mềm NetBeans

- Bước 4: Điền các thông tin cần thiết
  - Chọn đường dẫn tới file index (khi chạy ứng dụng sẽ tìm đọc file này đầu tiên)
  - Đánh dấu chọn như hình minh họa để project hoạt động được trên XAMPP (các file sẽ được copy đến folder **htdocs** của XAMPP)



1.7. Quy trình xây dựng một ứng dụng PHP đơn giản

- Bước 1: Cài đặt và chạy các phần mềm cần thiết trong gói XAMPP
- Bước 2: Cài đặt NetBeans và tạo một project mới
- Bước 3: Tạo cấu trúc file, folder cần thiết của project
- Bước 4: Viết mã PHP cần thiết cho ứng dụng vào các file của project
- Bước 5: Cài đặt xDebug và tiến hành kiểm thử project
- Bước 6: Sửa những lỗi phát hiện ở bước 5 và hoàn thiện ứng dụng

2. Hướng dẫn viết mã cho ứng dụng PHP

Trong phần này có các nội dung:

2.1. Các kỹ năng lập trình PHP cơ bản

2.2. Truyền dữ liệu từ yêu cầu

2.3. Làm việc với dữ liệu

2.4. Các câu lệnh điều khiển

2.1. Các kỹ năng lập trình PHP cơ bản

- Nhúng mã PHP vào mã HTML:
  - Mở thẻ PHP bắt đầu bằng <?php và kết thúc bằng ?> rồi viết mã PHP vào giữa cặp thẻ này
  - Nếu muốn thực hiện thao tác xử lý trước khi hiển thị HTML, nhúng mã PHP trước khi bắt đầu tài liệu HTML
  - Nếu muốn sử dụng PHP để hiển thị dữ liệu động trong tài liệu HTML, nhúng mã PHP trực tiếp vào vị trí cần hiển thị
  - Ví dụ: `<p>First name: <?php echo $first_name;?></p>`

Các kỹ năng lập trình PHP cơ bản

- Viết chú thích cho câu lệnh:
  - Để viết chú thích một dòng, sử dụng hai ký tự xổ xuống (//) hoặc dấu thăng (#) và viết chú thích cho tới khi hết
  - Để viết chú thích nhiều dòng (hay còn được gọi là khối chú thích), đầu tiên mở dấu /* rồi viết chú thích và kết thúc với dấu đóng */
  - Ví dụ:

```
/*******************************************
* Author:    Joel Murach
* Purpose:   This program calculates the discount for
*            a price that's entered by the user
*******************************************/

//lấy dữ liệu từ form
$list_price = $_GET['list_price'];
```

Các kỹ năng lập trình PHP cơ bản

- Khai báo và gán giá trị cho biến:
  - Để khai báo biến, trước tiên viết ký hiệu $, sau đó là tên biến
  - Để gán giá trị cho biến, sử dụng toán tử gán (=), theo sau là biểu thức trả về giá trị cho biến
  - Các biến trong PHP có phân biệt chữ hoa và chữ thường
  - PHP tự chọn kiểu dữ liệu cho biến tùy theo giá trị được gán mà không cần khai báo kiểu như các ngôn ngữ khác
  - Ví dụ: $list_price = 9.50

- Khai báo hằng số:
  - Sử dụng cú pháp: define('<tên hằng số>', <giá trị của hằng số>)
  - Theo quy tắc đặt tên, hầu hết các lập trình viên sử dụng chữ viết hoa cho tên hằng số
  - Ví dụ: define('MALE', 'm')

2.2. Truyền và lấy dữ liệu từ yêu cầu HTTP

Cách 1: **Sử dụng mảng tích hợp $_GET**

- Truyền dữ liệu: các thuộc tính và giá trị của chúng được lưu vào URL với quy tắc:

  …<tên file PHP>?<Thuộc tính 1>=<Giá trị 1>&<Thuộc Tính 2>=<Giá trị 2>

  Ví dụ: http://abc.com/index.php?Ten=Nam&Tuoi=20 (Truyền hai thuộc tính là 'Ten' và 'Tuoi' có giá trị tương ứng là 'Nam' và '20')

- Lấy dữ liệu: các thuộc tính cùng giá trị của chúng đều nằm trong mảng $_GET. Giá trị của các thuộc tính được lấy theo quy tắc $_GET['<Tên thuộc tính>']

  Thông thường người ta hay sử dụng một biến PHP để lưu giá trị của thuộc tính vừa lấy từ mảng $_GET

  Ví dụ: $Bien = $_GET['Ten'] (Lấy giá trị của thuộc tính 'Ten' và lưu vào biến $Bien)

Truyền và lấy dữ liệu từ yêu cầu HTTP

Cách 2: Sử dụng mảng $_POST

- Truyền dữ liệu: sử dụng form HTML

```
<form action="<tên file PHP>" method="post">
  <input type="text" name="<thuộc tính 1>">
  <input type="text" name="<thuộc tính 2>">
</form>
```

(form này còn có thể sử dụng cho phương thức GET ở cách 1, chỉ cần thay method là "get" thay vì "post")

- Lấy dữ liệu: các thuộc tính cùng giá trị của chúng đều nằm trong mảng $_POST. Giá trị của các thuộc tính được lấy theo quy tắc $_POST['<Tên thuộc tính>']

Truyền và lấy dữ liệu từ yêu cầu (hồi đáp) HTTP

- Nên sử dụng phương thức GET khi:
  - Có yêu cầu xem trang nhận dữ liệu từ máy chủ cơ sở dữ liệu
  - Muốn thực thi yêu cầu nhiều lần mà không gây lỗi
- Nên sử dụng phương thức POST khi:
  - Có yêu cầu xem trang viết dữ liệu lên máy chủ cơ sở dữ liệu
  - Việc thực hiện nhiều yêu cầu gây ảnh hưởng tới trang
  - Không muốn truyền tham số
  - Cần truyền hơn 4KB dữ liệu.

2.3. Làm việc với dữ liệu

- Viết mã cho biểu thức chuỗi:
  - Sử dụng dấu nháy đơn cho các chuỗi đơn giản sẽ tăng tính hiệu quả của PHP

```
$first_name = 'Bob';
```

  - Gán giá trị NULL và chuỗi rỗng

```
$address2 = '';                    //chuỗi rỗng
$address2 = null;                  //giá trị NULL
```

  - Sử dụng dấu nháy kép để chèn biến vào chuỗi

```
$name = "Name: $first_name"; //Tên: Bob
```

  - Sử dụng dấu nháy đơn và nháy kép cho các trường hợp đặc biệt

```
$last_name = "O'Brien";        //O'Brien
$line = 'She said, "Hi."'      //Cô ấy nói "Xin chào"
```

Làm việc với dữ liệu

- Nối chuỗi: sử dụng toán tử nối (.)
  - Nối chuỗi với biến chuỗi

```
$name = 'Name: ' . $first_name;       //Tên: Bob
```

  - Nối số với chuỗi

```
$price = 19.99;                        //Giá: 19.99
$price_string = 'Price: ' . $price;
```

- Viết câu lệnh echo:
  - Cú pháp cho câu lệnh echo

```
echo <biểu thức chuỗi>;
```

  - Sử dụng echo trong câu lệnh HTML

```
<p>Name: <?php echo $name; ?></p>
```

  - Sử dụng echo để tạo thẻ HTML và dữ liệu

```
<?php echo '<p>Name:' .$name . '</p>';
?>
```

Làm việc với dữ liệu

- Viết biểu thức số:
  - Sử dụng các toán tử số học thông dụng

Toán tử	Thứ tự ưu tiên	Mô tả	Ví dụ	Kết quả
+	4	Cộng	5+7	12
-	4	Trừ	5-12	-7
*	3	Nhân	6*7	42
/	3	Chia	13/4	3.25
%	3	Lấy dư	13%4	1
++	2	Tăng 1	$counter++	Giá trị biến tăng 1
--	1	Giảm 1	$counter--	Giá trị biến giảm 1

  - Ví dụ:

```
$x = 14;
$result = $x + 8;    //22
```

2.4. Viết các câu lệnh điều khiển

- Câu lệnh If:
  - Thực thi các câu lệnh dựa trên kết quả của biểu thức điều kiện
  - Cú pháp:

    If (<*biểu thức điều kiện*>) {<*Khối lệnh 1*>} else {<*Khối lệnh 2*>}
  - Ví dụ:

```
if (empty($first_name)) {          //Nếu $first_name rỗng thì yêu
    cầu nhập tên vào
$message =‘Yeu cau nhap ten';
}
else {                             //Nếu $first_name không rỗng thì đưa ra
    lời chào
$message = ‘Xin chao ' . $first_name;
}
```

Viết các câu lệnh điều khiển

- Câu lệnh while:
  - Viết mã vòng lặp để thực hiện lặp lại một khối lệnh cho đến khi biểu thúc điều kiện không còn đúng
  - Cú pháp:

    while (<biểu thức điều kiện>) {<Khối lệnh>}
  - Ví dụ:

```
$counter = 1;
while ($counter <= 5) {          //lặp lại khối lệnh sau cho đến khi
    $counter > 5
$message = $counter + 2;
$counter++;                      //tăng 1 cho biến đếm
}
//sau khi thoát khỏi vòng lặp, $counter = 6
```

Viết các câu lệnh điều khiển

- Câu lệnh for:
  - Viết mã vòng lặp để thực hiện lặp lại một khối lệnh theo một số lần nhất định
  - Cú pháp:

    for (<*biến đếm*>; <*biểu thức điều kiện*>; <*tăng giá trị biến đếm*>)

        {<*khối lệnh*>}
  - Ví dụ:

```
for ($i = 1; $i <= 5; $i++)
```

//Khởi tạo biến đếm $i giá trị 1, lặp lại khối lệnh sau đến khi $i > 5, mỗi lần lặp lại tăng giá trị $i thêm 1

```
    {
      echo 'Xin chao';
    }
```

//Sau khi thoát khỏi vòng lặp, $i có giá trị 6

Viết các câu lệnh điều khiển

- Truyền điều khiển sang trang khác: sử dụng các hàm chuyển

Tên hàm	Mô tả
include($path)	Chèn và chạy file được chèn theo đường dẫn $path
include_once($path)	Giống như hàm include, nhưng chỉ thực hiện một lần
require($path)	Hoạt động tương tự hàm include. Tuy nhiên, nếu xảy ra lỗi (không có file), nó đưa ra cảnh báo và dừng đoạn mã
require_once($path)	Giống hàm require, nhưng đảm bảo file này chỉ được yêu cầu đúng một lần
exit([$status])	Thoát khỏi đoạn mã PHP hiện thời. Nếu muốn hiển thị thông báo trạng thái trước khi thoát thì truyền chuỗi $status
die([$status])	Hoạt động tương tự hàm exit

Ví dụ: require '/home/index.php'; //Chạy file index.php

3. Hướng dẫn kiểm thử và gỡ lỗi cho ứng dụng PHP

Trong phần này có các nội dung:

3.1. Giới thiệu về kiểm thử và sửa lỗi

3.2. Hướng dẫn sửa lỗi với xDebug & NetBeans

3.1. Giới thiệu về kiểm thử và sửa lỗi

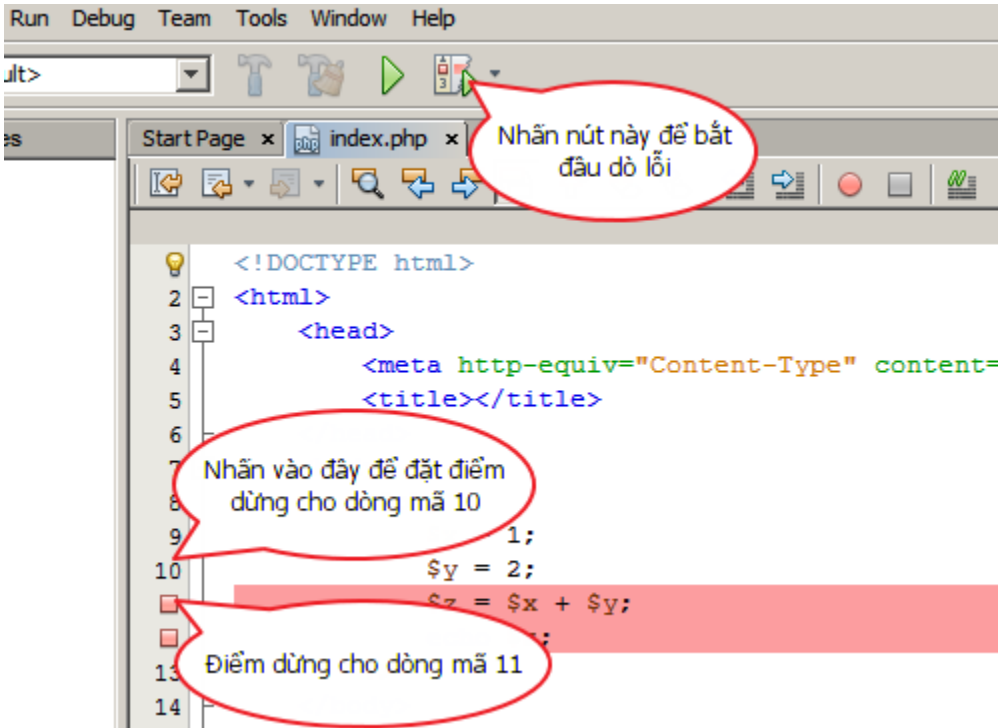
- Mục tiêu kiểm thử: tìm ra tất cả các lỗi trước khi ứng dụng được đưa vào sử dụng
- Mục tiêu sửa lỗi: sửa tất cả các lỗi trước khi ứng dụng được đưa vào sử dụng
- Các bước kiểm thử:
  - Kiểm tra giao diện người dùng để đảm bảo mọi thứ hoạt động đúng
  - Kiểm thử ứng dụng với các dữ liệu nhập hợp lệ để đảm bảo kết quả chuẩn xác
  - Kiểm thử ứng dụng với các dữ liệu không hợp lệ hoặc hành động người dùng không mong muốn. Thử mọi khả năng mà bạn cho là sẽ khiến phần mềm bị lỗi

Giới thiệu về kiểm thử và sửa lỗi

- Các loại lỗi thường gặp:
  - Lỗi cú pháp: vi phạm các quy tắc viết câu lệnh PHP, những lỗi này sẽ khiến trình thông dịch hiển thị lỗi và dừng thực thi mã
  - Lỗi thực thi: khi chạy không vi phạm các quy tắc cú pháp nhưng vẫn khiến trình thông dịch PHP hiển thị lỗi và có thể dừng thực thi mã
  - Lỗi lôgíc: các câu lệnh không gây ra lỗi cú pháp hay lỗi khi chạy nhưng cho ra kết quả sai

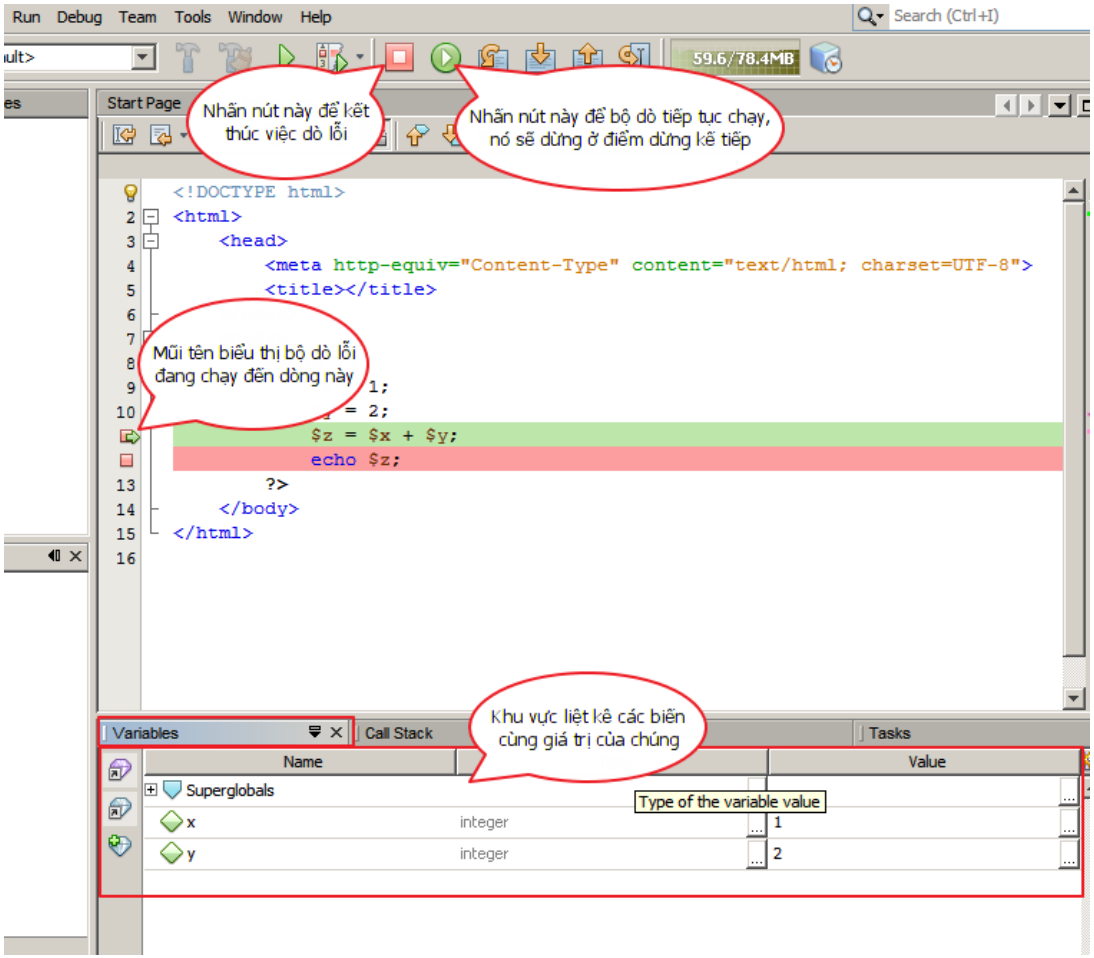
Hướng dẫn sửa lỗi với xDebug & NetBeans

- Chạy từng dòng mã và đặt điểm dừng:
  - Nhấn vào thanh đếm dòng mã tại vị trí tương ứng với dòng mã muốn đặt điểm dừng
  - Sau khi đặt các dòng mã, nhấn nút **Debug Project** để bắt đầu chạy từng dòng mã. Đến dòng mã có điểm dừng thì bộ dò lỗi sẽ dừng lại



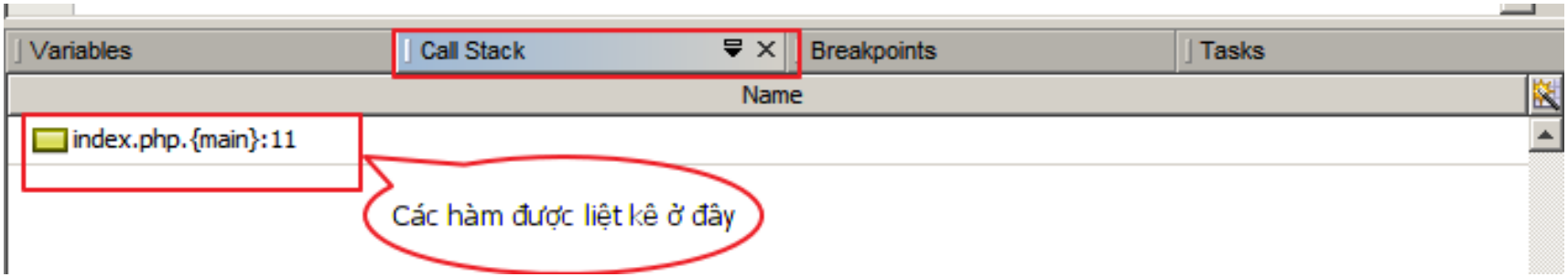
Hướng dẫn sửa lỗi với xDebug & NetBeans

- Kiểm tra giá trị biến tại vị trí đặt điểm dừng: các biến cùng giá trị của chúng được liệt kê trong tab Variables



Hướng dẫn sửa lỗi với xDebug & NetBeans

- Kiểm tra dấu ngăn xếp:
  - Dấu ngăn xếp là một danh sách các hàm theo thứ tự ngược với thứ tự được gọi
  - Hữu dụng khi project có số lượng hàm lớn
  - Thực hiện bằng cách nhấn vào tab Call Stack, danh sách các hàm sẽ được liệt kê
  - Nhấn vào bất cứ hàm nào trong ngăn xếp gọi để hiển thị hàm và tô đậm dòng mã gọi hàm tiếp theo



Tổng kết bài học

- Website chạy theo mô hình máy chủ - máy khách, giao tiếp với nhau bằng các yêu cầu (hồi đáp) HTTP

- Môi trường lập trình PHP thường bao gồm: XAMPP (PHP, MySQL, Apache), NetBeans, xDebug

- Các kỹ năng lập trình PHP cơ bản
  - Nhúng mã PHP vào mã HTML
  - Viết chú thích cho các câu lệnh
  - Khai báo và gán giá trị cho biến
  - Khai báo hằng số

- Làm việc với yêu cầu HTTP: truyền và nhận dữ liệu

Tổng kết bài học

- Làm việc với dữ liệu:
  - Viết mã cho biểu thức chuỗi
  - Nối chuỗi
  - Viết câu lệnh echo
  - Viết biểu thức số
- Viết các câu lệnh điều khiển: if, while, for, chuyển trang
- Sửa lỗi với xDebug và NetBeans

XIN CẢM ƠN!