

# LẬP TRÌNH WEB ĐỘNG VỚI **PHP / MySQL**

- ❖ GUESTBOOK
- ❖ CATALOG
- ❖ FORUM
- ❖ SHOPPING CART

## PHẦN 1

**Tổng Phước Khải (tổng hợp & biên dịch)**

# Giới thiệu

Chúng ta hãy thực hiện một chuyến đi thần thoại, trong chuyến đi này chúng ta sẽ khám phá ngoại hình cũng như nội tại của MySQL và PHP một cách thật tỉ mỉ. Đây là một cuộc hành trình đầy những thú vị và bất ngờ.

Okie, có lẽ tôi có vẻ hơi lạc quan phải không các bạn. Nếu như bạn đồng quan điểm với tôi trên một phương diện nào đó, trong cuộc hành trình này bạn sẽ có ngay sự giúp đỡ mỗi khi gặp phải những sự nhàm chán. Hãy đổi mặt sự thật ngay nhé: Trò chơi lập trình ứng dụng không phải lúc nào cũng dễ nuốt đầu. Trong bất kỳ cuộc thám hiểm nào thì chắc chắn các bạn sẽ phải có những giây phút nản lòng, đó là lúc gặp phải sự cố lỗi cú pháp hoặc đôi khi là những đoạn mã không cho kết quả như mong muốn. Nhưng ngoài những việc đó ra, tôi nghĩ là có một lý do thật chính đáng đến các bạn đến với chúng tôi ở đây. Lập trình Web đang là một cuộc chơi đầy hứa hẹn hiện nay cũng như tương lai. Bất kể bạn có kiến thức cơ sở lập trình cho bất kỳ loại ngôn ngữ nào như Visual Basic, Cobol, hay bạn chỉ biết về HTML và JavaScript, thì hôm nay bạn vẫn có cơ hội để nắm bắt các kinh nghiệm mới mẻ về lập trình ứng dụng Web. Tôi nghĩ là không có sự kết hợp nào tốt hơn giữa PHP và MySQL. Số lượng người sử

dụng ngôn ngữ này càng gia tăng, PHP và MySQL đã trở thành rất thông dụng, những đòi hỏi lượng người biết các công cụ lập trình này cũng tăng theo. Một chút xíu nữa tôi sẽ nói rõ cho bạn biết tại sao lại phải sử dụng PHP và MySQL. Nhưng trước hết tôi muốn bạn hãy khảo sát qua kiến trúc sơ bộ của ứng dụng Web. Vì chỉ khi bạn nắm bắt được điều này thì tôi mới có thể tiếp tục trình bày chi tiết rằng tại sao PHP và MySQL là trung tâm của môi trường phát triển ứng dụng Web.

*Trước khi tiếp tục, tôi nghĩ rằng bạn đã đọc những gì tôi đã giới thiệu và hiểu nó. Chúng ta tiếp tục đi thôi!*

## Kiến trúc cơ bản

Kiến trúc căn bản nhất để trang Dynamic Web hoạt động được là nó phải làm việc trên mô hình client/server. Nôm na là mỗi thứ **client** hay **server** đều đảm đương một chức năng riêng để hoàn thành công việc chung đó là cho ra một trang Web động. Các bạn có lẽ đã quen thuộc với chương trình WinWord để soạn văn bản, nó có thể hoạt động độc lập trên bất kỳ máy tính nào chẳng cần quan tâm tới cái gì là client hay cái gì là server. Ứng dụng Web thì khác hẳn, phải có một mô hình server có thể là

một máy tính làm server thôi, nhằm tập trung hoá việc xử lý dữ liệu. Còn các client, còn được hiểu nôm na là máy tính của người sử dụng phải được nối mạng với server, giả sử các máy này truy cập vào một Website chẳng hạn, thì có nghĩa họ đã truy cập vào server, sau đó lấy dữ liệu từ server về thể hiện lên máy mình. Cùng một lúc có thể có hàng trăm người (client) truy cập vào cùng một Website được xử lý tập trung trên server, tương tự như một đám trẻ xúm nhau giành phần của mình từ một cái bánh.

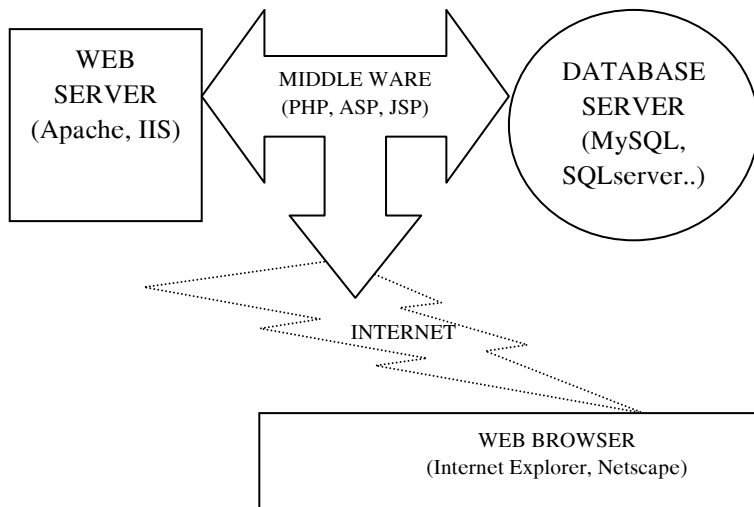
## **Client (người Việt tạm đọc là klai-ờn)**

Các ứng dụng mà bạn phát triển trên nền MySQL và PHP sử dụng tính năng single client đó là trình duyệt Web. Tuy nhiên, không phải đây chỉ là ngôn ngữ duy nhất để phát triển ứng dụng Web. Đối với những ứng dụng phức tạp đòi hỏi multi-client hoặc cần các tính năng bảo trì (chúng ta sẽ bàn tính năng này sau), thì ứng dụng Java applet sẽ hữu dụng cho việc này. Chỉ trừ trường hợp bạn cần sử dụng ứng dụng thời gian thực như ứng dụng chat chẳng hạn, thì bạn Java Applet mới cần thiết. Ở đây chúng ta không bàn tới lập ứng dụng cho chuyện tán gẫu mà chỉ tập trung vào ứng dụng duyệt Web nên không đụng chạm gì tới Java Applet cả.

Như bạn đã biết ngôn ngữ khởi thủy cho việc duyệt Web là HTML. HTML cung cấp hàng tá những thẻ lệnh (Tag) cho phép thể hiện trang Web theo nhiều kiểu cách khác nhau. Nếu bạn chưa có kiến thức cơ sở về HTML thì có thể chạy ra ngoài mua ngay một quyển sách hoặc download trên internet xuống các bài học hướng dẫn. Bạn không nên bỏ ra quá nhiều thời gian để học về HTML. Ngoài HTML ra các trình duyệt Web còn cho phép các add-in hỗ trợ nhiều thứ khác như RealPlayer, Flash, Shockwave, hoặc hỗ trợ về Javascript hoặc XML. Nhưng ở đây chúng tôi chỉ tập trung trên những gì cần thiết cho sự hội nhập của bạn – đó là HTML.

## Server (người Việt tạm đọc là sơ-vơ)

Hầu hết các ứng dụng Web đều hoạt động tập trung trên Server. Một ứng dụng đặc trưng gọi là Web Server sẽ đảm trách việc giao tiếp với các trình duyệt. Một Cơ sở dữ liệu (CSDL) trên Server sẽ lưu trữ tất cả những thông tin đáp ứng yêu cầu cho công việc của ứng dụng Web. Kế tiếp, bạn cần phải có một ngôn ngữ làm vai trò "**chú bé liên lạc**" giữa **Web Server** và **CSDL** trên server. Ngôn ngữ này cũng thực hiện các công việc xử lý thông tin đến và đi từ Web Server.



Và dĩ nhiên là các thứ này sẽ chẳng hoạt động được nếu như không chạy trên một Hệ Điều Hành (HĐH). Các thứ như Web Server, Ngôn ngữ lập trình, CSDL phải hoạt động tốt trên một HĐH nào đó.

## Hệ điều hành

Có rất nhiều chủng loại HĐH. Windows 98/XP và Linux có lẽ rất phổ biến với tất cả mọi người. Có trường hợp bạn làm việc trên HĐH mà ít ai biết tới và bạn chỉ có ấn tượng thích sử dụng nó mà thôi. Hãy gác qua những ý tưởng đó nếu như bạn thật sự muốn đi trên con đường thiết kế web. Hãy trang bị cho mình kiến thức về HĐH WinNT /2000 /2003 và Unix đi. Việc làm này sẽ rất có ích hơn là chuyện bảo mọi người nên đi học một khoá về AS/400.

Bạn sẽ sử dụng loại nào trong các thứ nói trên đây? Okie, đây là một câu hỏi hơi rắc rối đấy. Câu trả lời ở đây là tùy thuộc bạn là "tín đồ" của HĐH nào. Nếu như bạn vẫn chưa rõ ràng về điều này, hãy để tôi nói cho bạn nghe về "chiến tranh giáo phái HĐH".

Nếu bạn chưa hiểu được tôi đang nói gì, thì đây là các kiến thức cơ bản: PHP và MySQL thuộc nhóm phần mềm ứng dụng có tên gọi là **open source** (nguồn mở). Việc này có nghĩa là người dùng sẽ xem được mã nguồn của các ứng dụng sử dụng PHP/MySQL. Chúng tận dụng được mô hình phát triển dựa vào nguồn mở, cho phép người nào cảm thấy thích nó đều có thể góp phần vào việc phát triển các dự án.

Trong trường hợp của PHP, các lập trình viên trên toàn thế giới tham gia vào việc phát triển ngôn ngữ và không trông chờ một khoản lợi nhuận nào. Phần lớn những người tham gia công việc đều có niềm đam mê việc tạo ra một sản phẩm phần mềm tốt, họ sẽ cảm thấy thích thú khi thấy người khác sử dụng các công cụ của họ như tôi và bạn chẳng hạn.

Phương pháp nguồn mở này ban đầu chỉ còn là những vòng lẩn quẩn mà thôi, nhưng về sau đã trở thành đầy tiềm lực khi có sự ra đời và trở nên phổ biến của bộ nguồn mở Linux. Hầu như các nguồn mở đều miễn phí, bạn có thể download, cài đặt và sử dụng chúng mà không cần phải đợi sự cho phép hay phải trả tiền cho bất kỳ ai. Phương thức này thì Microsoft, Oracle hay một số các công ty lập trình nào khác không thể đáp ứng được.

Nếu bạn không phải là tín đồ của phái nguồn mở, thì hãy chọn công cụ được coi là béo bở: NT/2000/2003. Nếu công ty của bạn đã sử dụng sản phẩm của Microsoft nhiều năm rồi thì mọi việc sẽ trở nên dễ dàng nếu bạn muốn duy trì làm việc với môi trường này. Nếu bạn là thành viên của nhóm lập trình Visual Basic, có lẽ bạn sẽ gắn bó với NT/2000/2003. Ngay cả trong trường hợp này, không có trở lực nào ngăn cản



bạn trong công việc phát triển với công cụ PHP và MySQL. Bạn cũng có thể thử nghiệm **PHP/MySQL** trên nền HĐH **Windows 95, 98, XP**.

## Web Server

Chức năng của Web Server có vẻ không phức tạp mấy. Nó chỉ ở tại chỗ, chạy trên nền của HĐH, lắng nghe các yêu cầu ai đó trên Web gửi đến, sau đó trả lời những yêu cầu này, và cấp phát những trang Web thích ứng. Thực tế thì nó không quá đơn giản như vậy, bởi vì nhiệm vụ của Web Server là phải cung cấp tính ổn định cho môi trường Web cho nên đòi hỏi này phải được đáp ứng một cách rất nghiêm túc.

Có nhiều loại Web Server khác nhau, nhưng chủ yếu trên thị trường chỉ thường sử dụng Apache và IIS (Internet Information Server của Microsoft).

INTERNET INFORMATION SERVER (IIS) được gắn liền với môi trường Windows và nó là thành phần không thể thiếu của Active Server Pages (ASP). Nếu bạn chọn con đường của Microsoft thì có lẽ bạn đã hiểu rõ về IIS.

Có một sự tích hợp nhất định giữa một ngôn ngữ lập trình và một Web Server. Cũng vậy, PHP4 được tích hợp rất tối đối với IIS. Trước đây, có một số vấn đề cần phải bàn

về tính ổn định của PHP/IIS với việc truyền tải lớn, nhưng PHP và IIS cũng đã được cải thiện liên tục nên việc này không còn đáng phải bận tâm.

APACHE là một kiểu mẫu Web Server rất phổ biến. Giống như Linux, PHP, MySQL nó là một dự án nguồn mở. Không có gì ngạc nhiên khi người ta thấy Apache được hỗ trợ **rất tốt trên môi trường Unix**, nhưng chỉ **khá tốt trong Windows**.

Apache tận dụng được tính năng của third-party. Bởi vì đây là nguồn mở nên bất kỳ ai có khả năng đều có thể viết chương trình mở rộng tính năng của Apache. PHP hoạt động với tư cách là một phần mở rộng của Apache, và người ta gọi là một module của Apache.

Apache có tính ổn định và tốc độ đáng phải nói. Tuy nhiên, cũng có một số sự phàn nàn về nó là không hỗ trợ công cụ đồ họa trực quan, điều có thể giúp người ta làm việc một cách dễ dàng hơn. Bạn phải thực hiện các thay đổi đối với Apache bằng cách sử dụng dòng lệnh, hoặc sử dụng các tập tin text trong folder chương trình Apache. Nếu lần đầu đến với Apache thì bạn sẽ gặp một chút lạ lẫm.

Mặc dù Apache chỉ làm việc tốt trên Unix, nhưng cũng có những phiên bản chạy tốt trên hệ Windows. Không một ai, kể cả các nhà phát triển Apache đề nghị rằng

Apache nên được chạy trên một server Windows bạn rợn. Nếu bạn quyết định chọn HĐH Windows cho server thì bạn nên sử dụng IIS. Nếu bạn thử nghiệm ứng dụng trên Windows và sau đó đem upload và chạy trên Unix/Apache của nhà cung cấp host thì cũng không hề hấn gì, ứng dụng của bạn vẫn chạy ngon lành.

## Middleware

PHP thuộc lớp ngôn ngữ lập trình gọi là middleware. Các ngôn ngữ này hoạt động cận kề với Web Server để thông dịch các yêu cầu từ trên World Wide Web, sau đó nhận các trả lời từ Web Server chuyển tải đến trình duyệt Web nhằm đáp ứng các yêu cầu đó.

Middleware là nơi mà bạn sẽ thực hiện các khối lượng rất lớn công việc chính yếu của bạn. Với hỗ trợ này Web Server của bạn sẽ không phải cán đáng quá nhiều khối lượng công việc. Nhưng khi bạn phát triển ứng dụng của bạn, bạn sẽ tốn nhiều thời gian viết mã chương trình để cho chương trình của bạn có thể hoạt động được. Ngoài PHP ra có một số ngôn ngữ khác có chức năng tương đương như ASP, Perl, ColdFusion.

## Hệ CSDL quan hệ

Relational Database Management Systems (Hệ Quản trị Cơ Sở Dữ Liệu Quan hệ - RDBMSs) cung cấp phương thức tuyệt vời để lưu trữ và truy xuất lượng thông tin lớn và phức tạp. Nó đã ra đời khá lâu. Thực tế, nó có trước Web, Linux và WindowsNT, cho nên không có gì ngạc nhiên khi có quá nhiều hệ CSDL để chọn lựa. Tất cả các CSDL này đều dựa trên cơ sở SQL (Structure Query Language).

Một số hệ phổ biến như **Oracle, Sysbase, Informix, Ms SQL Server, IBM's DB2**. Hệ nguồn mở thông dụng hiện nay là **MySQL** mà quyển sách này đề cập đến, ngoài ra còn có hai hệ nguồn mở khác là **PostgreSQL** đã một thời thay thế MySQL và Interbase là bộ nguồn mở của Borland giới thiệu vào tháng 8/1999.

## Tại sao sử dụng PHP và MySQL

Tại sao có quá nhiều chọn lựa như vậy mà chúng ta lại phải chỉ lấy ra cặp bài trùng PHP/MySQL mà thôi? Tôi sẽ giải thích điều này ở phần sau.

# Nói về PHP

Các ngôn ngữ lập trình xem ra giống như các loại giày dép. Có loại có vẻ bắt mắt với một số người này, nhưng lại khó ưa với người khác và ngược lại. Một số người chỉ thích sử dụng một hiệu giày nào đó đã quen thuộc và ngôn ngữ lập trình cũng tương tự như vậy.

Ở đây tôi muốn ngụ ý với các bạn là khi lập trình Web, các ngôn ngữ lập trình đều cho kết quả gần giống nhau. Câu hỏi ngôn ngữ nào tốt nhất không phải là vấn đề nó không có khả năng thực hiện một số chức năng nào đó mà thường là nó có làm cho bạn thực hiện công việc một cách nhanh chóng và đỡ nhọc công hay không?

## Tốc độ nhanh, dễ sử dụng

Chúng ta hãy bàn về tốc độ. Có 3 thứ mà tôi chắc chắn khi bàn về việc so sánh tốc độ giữa các ngôn ngữ lập trình Web. Thứ nhất, ứng dụng viết bằng C chạy nhanh nhất. Thứ hai, công việc lập trình C khá là phức tạp, và sẽ tốn nhiều thời gian hơn. Thứ ba, việc so sánh giữa các ngôn ngữ là một điều khó khăn. Tất cả những gì tôi biết là tôi cảm thấy yên tâm khi nói rằng PHP cũng nhanh như các ngôn ngữ khác. Trở lại ví

dự so sánh với các loại giày dép: Vina, Đông Hải, Kiến Hoa, Hồng Thanh, Italy v.v., chắc chắn bạn sẽ chọn loại tiện dụng nhất? Nếu bạn giống như tôi, bạn sẽ cảm thấy rằng PHP có đầy đủ các đặc tính như khả năng, cấu trúc và dễ sử dụng. Xin nói thêm, đây chỉ là cách nhìn riêng của tôi thì tôi tin rằng cú pháp PHP tuyệt hơn ASP hay JSP. Và theo tôi thì việc gõ lệnh PHP nhanh hơn ColdFusion và nó không khó học như Perl. Tóm lại, tôi cho rằng PHP cung cấp các tính năng mạnh mẽ để thực hiện ứng dụng Web một cách nhanh chóng.

## **Chạy trên nhiều hệ điều hành**

Như đã trình bày ở phần kiến trúc web, tôi có nói là PHP có thể chạy trên WindowsNT/2000/2003 và Unix với sự hỗ trợ của IIS và Apache. Nhưng ngoài ra nó có thể chạy trên một số các platform khác như Netscape, Roxen, hay một vài thứ khác. Như chúng ta biết ASP có thể chạy trên Unix, ColdFusion có thể chạy trên Solaris và Linux, JSP có thể chạy trên khá nhiều loại platform. Đối với PHP, nó có thể chạy tốt trên những platform hỗ trợ các chủng loại trên.

## **Truy cập bất kỳ loại CSDL nào**

Ứng dụng của bạn dự định sẽ truy cập những loại dữ liệu dịch vụ nào? LDAP, IMAP mail server, DB2, hay XML parser hay WDDX.

Bất kể bạn cần đến thứ gì thì PHP cũng sẵn sàng hỗ trợ thông qua các hàm được xây dựng sẵn nó sẽ làm công việc của bạn trở nên rất dễ dàng và tiện lợi. Nhưng nếu như có một số thứ chưa được xây dựng sẵn thì sao? Ta tiếp tục sang phần sau sẽ rõ.

## **Luôn được cải tiến & cập nhật**

Nếu như bạn cảm thấy bỡ ngỡ đối với việc phát triển nguồn mở, bạn có lẽ sẽ ngạc nhiên đối với chất lượng của loại phần mềm này. Có hàng ngàn những chuyên gia lập trình xuất sắc đợi sẵn và họ sẵn sàng bỏ thời gian ra để tạo những phần mềm tuyệt vời và hầu như miễn phí. Đối với ngôn ngữ thịnh hành như PHP thì ắt hẳn là các rất nhiều các nhà lập trình đang thực hiện phát triển nó hằng ngày.

Sự thật có một việc rất ấn tượng là nếu như bạn có một sự cố kỹ thuật, bạn có thể gửi email đến một nhà phát triển PHP các chi tiết sự cố đó. Chỉ trong vòng vài giờ bạn sẽ nhận được sự trả lời thỏa đáng.

Khi PHP4 được phổ biến, nó đã trở thành một hiện tượng của ngôn ngữ lập trình. Nó giúp cho việc bổ sung số lượng lớn các hàm chức năng một cách dễ dàng. Nếu như ngôn ngữ đã có sẵn nhiều hàm đặc thù cho công việc thì bạn sẽ đỡ tốn công hơn cho việc lập trình của mình.

## **Được hướng dẫn kỹ thuật bất cứ lúc nào**

Hầu hết các ngôn ngữ đều hỗ trợ active mailing list (hiểu nôm na là danh sách mail những thành viên trực chiến hỗ trợ kỹ thuật) và các development site (trang web hỗ trợ giải quyết kỹ thuật). PHP cũng không ngoại lệ. Nếu bạn gặp phải sự cố - gặp những lỗi trong chương trình và không tìm ra cách khắc phục - sẽ có hàng trăm người có tên trong danh sách mail luôn sẵn lòng kiểm tra và khắc phục sự cố cho bạn.

Bộ nguồn mở PHP thật sự đã tạo ra một tình cảm của cả cộng đồng. Khi bạn gặp phải khó khăn đối với nó thì lúc nào cũng có những đồng môn chia sẻ nỗi lòng đó và giúp bạn khắc phục nhằm đem lại niềm vui cho bạn.



## **Hoàn toàn miễn phí\$**

Bạn không ngại gì về vấn đề bản quyền khi bạn sắm một máy vi tính và cài lên đó những phần mềm như Linux, Apache, PHP vì tất cả đều miễn phí.



# Nói về MySQL

Mặc dù MySQL được phổ biến rất nhiều nhưng nó vẫn có những đối thủ đáng gờm đang cạnh tranh với nó. Những đối thủ của nó có thể trội hơn về một phương diện đặc thù nào đó.

Trong phần trên, chúng ta đã bàn sơ qua MySQL. Trong phần này, bạn sẽ được biết về những đặc điểm của những Hệ quản trị CSDL khác mà MySQL không hỗ trợ.

Với những hạn chế đó đã làm cho MySQL không được chọn để chạy trên một số các môi trường. Nếu bạn đang có kế hoạch bắt đầu cho một ngân hàng chẳng hạn, thì tôi khuyên bạn là MySQL không thích hợp cho bạn.

Nhưng đối với phần đông mọi người và phần lớn các ứng dụng, MySQL là sự chọn lựa của họ bởi nó rất thích hợp cho những ứng dụng Web.

## Vừa túi tiền

Hãy nghĩ bạn cần cài đặt Oracle. Hãy chuẩn bị hấu bao của mình khoảng 30.000 đến 100.000 USD hoặc thậm chí còn hơn thế nữa. Điều hiển nhiên là Oracle, Sysbase và

Informix là những Hệ Quản trị CSDL tuyệt vời, nhưng giá thành quá cao, không hợp với túi tiền của phần đông mọi người.

MySQL hoàn toàn miễn phí. Bạn có thể sử dụng mà không cần chuẩn bị bất kỳ khoản tiền nào.

## **Nhanh và mạnh**

MySQL không có đầy đủ những cơ sở vật chất cho một Hệ Quản trị CSDL chính tông, nhưng đối với công việc thường nhật của phần đông mọi người thì nó cung cấp cũng khá nhiều thứ. Nếu công việc của bạn là lưu trữ dữ liệu trên Web hoặc làm một trang Thương mại Điện tử cỡ vừa, thì MySQL có đủ những thứ bạn cần.

Đối với những CSDL cỡ trung bình thì MySQL hỗ trợ tuyệt vời về tốc độ. Các nhà phát triển MySQL rất tự hào về tốc độ sản phẩm của họ. Với các ứng dụng mà tôi giới thiệu trong phần III và IV của quyển sách này, thì bạn khó có thể kiếm được một Hệ Quản trị CSDL nào đạt được tốc độ nhanh hơn nó.

## **Cải tiến liên tục**

MySQL được cải thiện liên tục với một tần số không ngờ. Các nhà phát triển cập nhật nó thường xuyên, ngoài ra còn bổ sung các tính năng rất ấn tượng cho nó mọi lúc mọi nơi.

Hiện tại, MySQL đã được bổ sung thêm hỗ trợ transaction. Như vậy là MySQL đã thực thụ trở thành một Hệ Quản trị CSDL chuyên nghiệp.



# Thực hành ứng dụng đầu tiên

Phần mở đầu như vậy là tạm đủ. Bây giờ chúng ta hãy tiếp tục sang phần viết một ứng dụng thử nghiệm để biết được cách thức hoạt động của ngôn ngữ này như thế nào. Có lẽ đọc qua phần giới thiệu bạn cũng đã có một số khái niệm nhất định về sự hoạt động của chúng.

## Công cụ cần thiết

Có một số thành phần cần thiết mà bạn phải có trước hết. Tôi sẽ giới thiệu ngay sau đây và bạn sẽ biết mình cần đến những gì.

### PHP Webserver

Đây là ứng dụng chạy trên Web, cho nên bạn điều hiển nhiên là bạn cần phải có một Web Server. Bạn sử dụng Apache, bạn cài lên Winserver2000/ 2003 hoặc 98, XP thông dụng của bạn. Có một số phiên bản Apache có sẵn bộ cài đặt PHP trong đó.

Nếu chưa có bạn phải cài đặt thêm PHP. Còn nữa, bạn phải cài MySQL. Như vậy bộ ba **Apache, PHP và MySQL** luôn đồng hành với nhau.

Bạn xem thêm phần cài Apache server trên các CD thực hành PHP hoặc xem trên các Diễn đàn Tin học . Sau khi cài đặt xong bạn khởi động Apache. Nếu từ trình duyệt gõ vào <http://localhost> trang web thông tin của Apache hiển thị thì coi như thành công.

**Lưu ý:** Bạn cần phải xác định thư mục gốc của localhost để chứa các file **.php** của bạn sau này (xem trong hướng dẫn cài đặt Apache).

## Text Editor

Để soạn thảo các dòng lệnh PHP bạn cần có một chương trình soạn thảo text đơn giản thôi, như Notepad trong Windows chẳng hạn.

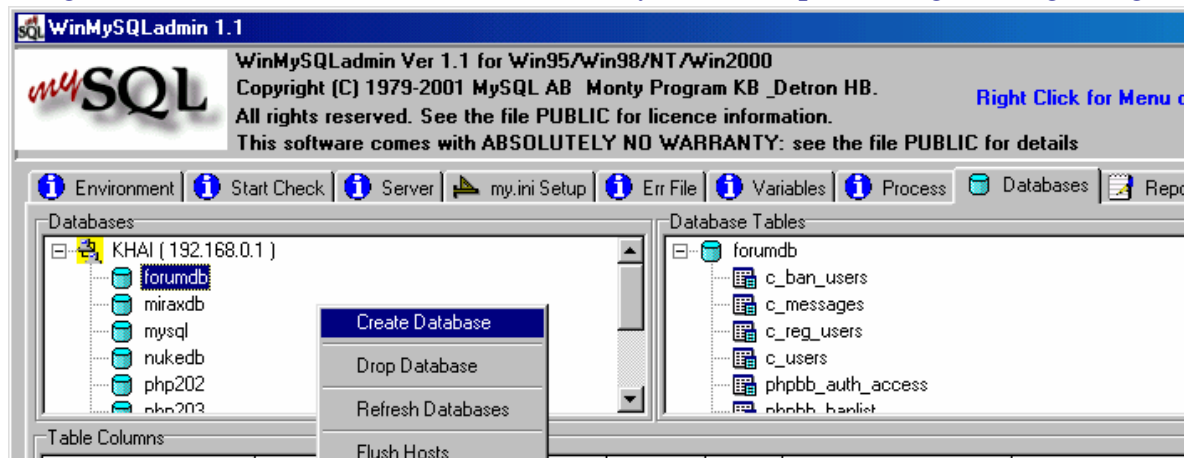
Có một số chương trình soạn thảo hỗ trợ PHP, các dòng lệnh được ngời sáng bằng nhiều màu khác nhau nhằm làm cho việc lập trình của bạn dễ dàng hơn. Bạn có thể vào các website của Allaire ([www.allaire.com](http://www.allaire.com)) hoặc Editplus ([www.editplus.com](http://www.editplus.com)). Hiện nay, chương trình Dreamweaver MX của Macromedia cũng hỗ trợ soạn thảo mã lệnh PHP rất tốt.

# BẮT ĐẦU LÀM

Tôi nghĩ là tôi đã khởi đầu quyển sách với những mở lý thuyết suông. Bây giờ chúng ta hãy bắt tay vào thực đi thôi. Như bạn đã biết khi truy cập vào một trang Web có thể bạn sẽ được yêu cầu hay chính bạn muốn ghi lại ý kiến cùng với một số các thông tin nhận dạng về mình như họ tên, địa chỉ website, email v.v. Tất cả các thông tin này sẽ được lưu trữ vào một CSDL trên Web. Nhờ vậy, người quản trị Web hoặc những người truy cập khác sẽ biết thông tin cá nhân cùng những ý kiến của bạn. Người ta gọi thông tin này là GuestBook (hiểu nôm na là Sổ vàng để khách viếng thăm ghi chép). Bây giờ chúng ta bắt tay vào việc tạo một GuestBook.

## Tạo một Database (quan trọng!)

Bây giờ bạn cần biết phải làm gì rồi. Chuyện đầu tiên là phải tạo một CSDL lưu trữ thông tin của khách. Để làm được điều này bạn cần phải dùng đến ngôn ngữ SQL



(thực tế bạn có thể làm với vài động tác nhấp chuột và vài ngón gõ phím, nhưng hãy tập làm quen với SQL vì nó sẽ hữu dụng về sau). Bạn sẽ được học kỹ về SQL trong các chương sau. Do đó bạn đừng lo lắng khi chưa hiểu gì về nó.



Bây giờ bạn hãy khởi động MySQL. Nếu bạn đã cài đặt MySQL trong Windows thì nó sẽ có biểu tượng để khởi động hoặc nó sẽ được tự động khởi động khi mở Windows lên. Đối với MySQLAdmin version 1.1 cho phép bạn làm việc trong 2 giao diện: Windows và Dos. Đối với giao diện Windows thì biểu tượng MySQL (biểu tượng đèn giao thông) nằm ở SystemTray, bạn chỉ việc click chuột phải lên nó và chọn **Show me**. Cửa sổ làm việc của MySQL hiện lên, tuy nhiên trong cửa sổ này chỉ cho phép bạn thực hiện một số thao tác có hạn đối với CSDL. Hình trên là cách tạo Database mới trong MySQL theo giao diện Windows.

Tuy nhiên, tôi khuyên các bạn nên dùng tiện ích **PhpMyAdmin**, chương trình này hỗ trợ **các thao tác đối với CSDL** trong MySQL với giao diện dễ sử dụng.

Trong phần này tôi hướng dẫn thêm bạn thực hiện thao tác với Database trong giao diện **dòng lệnh MSDOS** bởi vì các giao diện khác tôi nghĩ tự bạn có thể làm được. Bật màn hình dòng lệnh DOS lên, chuyển sang thư mục cài đặt MySQL có chứa tập tin **mysql.exe** (/mysql/bin) và gõ vào mysql <Enter>.

Tại dấu nhắc lệnh hãy gõ lệnh để tạo ra một database mới:

```
mysql> create database guestbook;  
Query OK, 1 row affected (0.00 sec)
```

```
mysql>
```

Bây giờ trong CSDL guestbook bạn cần phải có table chứa thông tin của khách. Bạn hãy dùng lệnh **create table** từ dấu nhắc lệnh:

```
mysql> use guestbook  
Database changed  
mysql> create table guestbook  
-> (  
-> name varchar(40) null,  
-> location varchar(40) null,  
-> email varchar(40) null,  
-> url varchar(40) null,  
-> comments text null  
-> )  
-> ;  
Query OK, 0 rows affected (0.00 sec)
```

```
mysql>
```

Bạn đã có một CSDL tên là **guestbook** và một table ở trong CSDL này cũng tên là guestbook. Bây giờ đã đến lúc chúng ta viết một ứng dụng bằng PHP để thực hiện các thao tác: *xem, chèn, sửa, xoá* các thông tin trong CSDL **guestbook**.

## Viết lệnh PHP

Bạn hãy dùng một chương trình soạn thảo văn bản đơn giản như Notepad chẳng hạn. Đặt tên cho các tập tin là **.php**, các tập tin này được lưu trữ trong thư mục gốc của **web local** trên máy bạn (**nên tham khảo CD cài Apache để rõ hơn**).

## Cú pháp cơ bản

Điều thú vị đối với PHP là cho phép bạn xen kẽ giữa lệnh HTML và lệnh PHP. Do đó, PHP được xem như là một script giống như Javascript hay Vbscript. Các lệnh của PHP được gói trong thẻ mở: `<?` và thẻ đóng: `?>`

Bây giờ bạn hãy thử chạy tập tin **hi.php** sau:

```
<?
echo "Hi, ";
?>
mom.
```

Khi chạy bạn sẽ gõ vào: **localhost/hi.php**

Kết quả cho ra là "Hi, mom". Ta thấy chữ "Hi," nằm trong tag lệnh PHP còn chữ "mom" thuộc về HTML.

Tuy nhiên, PHP còn làm được nhiều điều khác nữa, cũng giống như các ngôn ngữ lập trình khác, nó có thể làm việc với các loại biến, kiểu dữ liệu, chứa rất nhiều hàm chức năng. Hãy tìm hiểu ví dụ sau:

```
<?
echo "hi, mom." ;
$var = date("H");
if ($var <= 11)
{
echo "good morning";
}
elseif ($var > 11 and $var < 18)
{
echo "good afternoon";
}
else
{
echo "good evening";
}
?>
```

Nếu như bạn thấy khó hiểu thì cũng không sao. Chúng ta sẽ biết tường tận hơn ở phần sau.

Trang kết quả sẽ hiển thị các lời chào tùy thuộc vào giờ giấc hiện tại. Tôi đã dùng hàm date của PHP để lấy ra được giờ giấc hiện tại. Giá trị giờ được đem gán cho biến \$var. Kế đến là các chọn lựa được sử dụng để đưa ra lời chào thích hợp.

Các bạn hãy để ý một chút, các lệnh của PHP đều được kết thúc với dấu chấm phẩy (;). Trong phát biểu IF chúng ta thấy dấu ngoặc nhọn { } chứa các lệnh tùy sẽ được thi hành tùy thuộc vào điều kiện. Các điều kiện thì được bao trong dấu ngoặc đơn ( ). Hàm date() và lệnh echo chỉ là 2 trong hàng trăm các hàm và lệnh có trong PHP mà các bạn sẽ được học một số cần thiết của chúng trong các chương sau. Bây giờ bạn hãy tìm hiểu thêm một số lệnh về CSDL.

## **Lệnh PHP để kết nối Database (quan trọng)**

Bởi vì PHP và MySQL hiện tại trên máy của bạn vẫn còn là hai thế giới tách biệt nhau. Do đó, muốn dùng PHP để làm việc được với CSDL bạn cần phải tạo ra sợi dây liên kết giữa hai chiến hữu này.

Vì có thể có rất nhiều database trong MySQL, do đó bạn cần phải chỉ ra bạn muốn sử dụng database nào trong MySQL. Chúng ta hãy thực hiện như sau:

```
<?  
mysql_connect("localhost", "khai", "kkk") or  
die ("Could not connect to database");  
mysql_select_db("guestbook") or  
die ("Could not select database");  
?>
```

Dòng đầu tiên thực hiện việc kết nối với Database Server đang nằm trên máy **localhost**, có username là **khai**, password là **kkk**.

Nếu kết nối thành công, nó sẽ thực hiện bước kế tiếp là kết nối với database nằm trong đó là **guestbook** bằng lệnh `mysql_select_db()`

Các bạn nên lưu ý là các lệnh trên bạn sử dụng thường xuyên cho mọi kết nối CSDL của bạn, do đó tôi khuyên bạn nên lưu nó vào một tập tin (`dbconnect.php` chẳng hạn), sau này cần thì chỉ việc dùng lệnh `include('dbconnect.php');`

## Nhập dữ liệu vào Database

Bởi vì hiện tại database của bạn vẫn chưa có user nào, cho nên tôi sẽ hướng dẫn bạn viết các lệnh để thực hiện việc này. Nhưng trước tiên, bạn cần phải biết thêm một chút ít về biến trong PHP. Ở phần trước bạn đã xem qua một ví dụ trong đó có chứa biến, tuy nhiên đối với môi trường client/server, bạn cần phải làm việc với biến data từ client. Bạn sẽ thường xuyên làm việc với form HTML (bạn có thể tìm hiểu kỹ hơn ở phần Phục lục A. Chúng ta nên biết là mỗi phần tử của form đều có một cái tên, và khi bạn submit một form nào đó thì các tên của các phần tử trong đó trở thành một biến trong script PHP được form submit đến. Với form như sau, khi được submit, các biến \$surname và \$submit sẽ được tạo ra trong *myscript.php*. Giá trị \$surname sẽ mang giá trị mà user đã nhập vào. Giá trị của \$submit sẽ là chuỗi "submit".

```
<form action="myscript.php">  
<input type="text" name="surnmae">  
<input type="submit" name="submit" value="submit">  
</form>
```

Tôi xin lưu ý với các bạn là lập trình Web không giống như các dạng lập trình khác ở chỗ nó không ở trạng thái tĩnh. Để thể hiện một trang, Web Server phải trước hết nhận một thỉnh cầu từ trình duyệt. Giao thức sử dụng của chúng là HTTP, Hypertext Transfer Protocol. Các yêu cầu sẽ bao gồm: trang web mà trình duyệt sẽ thấy, form data, loại trình duyệt đang được sử dụng, địa chỉ IP mà trình duyệt sử dụng. Dựa vào thông tin này mà Web Server sẽ quyết định phục vụ những gì. Một khi server phục vụ yêu cầu trang web, nó sẽ duy trì sự kết nối với trình duyệt. Thông thường, bạn cần biết cách thức để chuyển các biến từ trang này sang trang khác. Bạn sẽ tìm thấy thao tác này trong ứng dụng tiếp theo. Ứng dụng của chúng ta sẽ giải quyết vấn đề dựa theo 1 trong 3 cách thức: *chuyển giao theo phần tử form ẩn, sử dụng cookies, sử dụng session*.

Bây giờ trở lại script sau:

```
<form action="myscript.php">
<input type="text" name="surnmae">
<input type="submit" name="submit" value="submit">
</form>
```



Bạn có thể quyết định cho hiển thị trên site những gì dựa vào các biến thông tin từ form HTML. Thông thường, bạn có thể kiểm tra nếu form đã được submit hay chưa bằng cách kiểm tra biến \$submit có chứa giá trị "submit" hay không.

Hãy bắt tay vào công việc đi thôi. Trang đầu tiên trong ứng dụng được gọi là **sign.php** có chứa một form HTML. Action của nó là *create\_entry.php*. Sau đây là chi tiết dòng lệnh:

```
<h2>Sign my Guest Book!!!</h2>
<form method=post action="create_entry.php">
<b>Name:</b>
<input type=text size=40 name=name>
<br>
<b>Location:</b>
<input type=text size=40 name=location>
<br>
<b>Email:</b>
<input type=text size=40 name=email>
<br>
<b>Home Page URL:</b>
<input type=text size=40 name=url>
<br>
<b>Comments:</b>
```

```
<textarea name=comments cols=40 rows=4 wrap=virtual></textarea>
<br>
<input type=submit name=submit value="Sign!">
<input type=reset name=reset value="Start Over">
</form>
```

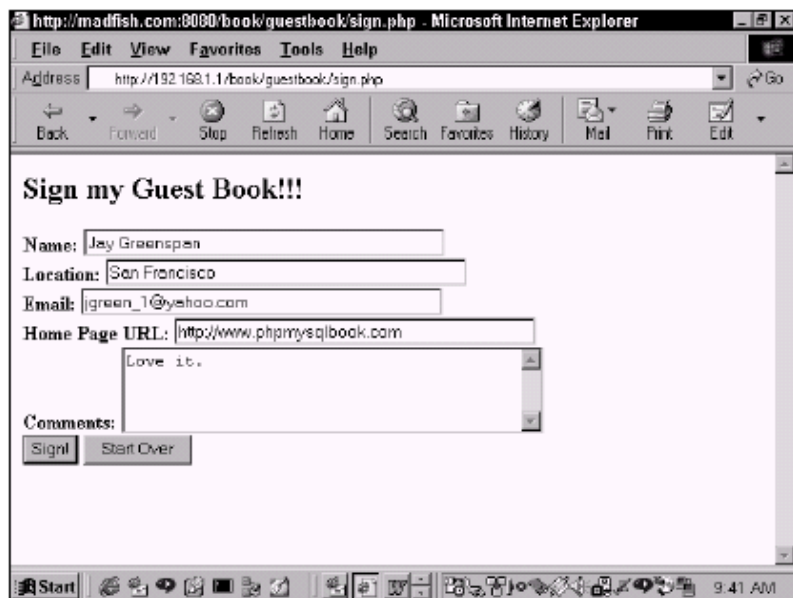
Khi bạn điền đầy đủ thông tin ở trong form, thì các thông tin sẽ được chuyển đổi tới *create\_entry.php*. Chuyện đầu tiên phải làm trên trang này là kiểm tra xem form đã được submit chưa. Nếu rồi, nhận lấy giá trị đã nhập vào trong form và sử dụng chúng để tạo một query đồng thời gửi đến MySQL. Bạn đừng lo lắng là không biết các lệnh SQL, điều trước tiên là chỉ cần biết là nó sẽ thực hiện việc chèn dữ liệu vào table của guestbook. Tập tin **create\_entry.php** như sau:

```
<?php
include("dbconnect.php");
if ($submit == "Sign!")
{
$query = "insert into guestbook
(name,location,email,url,comments) values
('$name', '$location', '$email', '$url', '$comments')";
mysql_query($query) or
```

```
die (mysql_error());  
?>  
<h2>Thanks!!</h2>  
<h2><a href="view.php">View My Guest Book!!!</a></h2>  
<?php  
}  
else  
{  
include("sign.php");  
}  
?>
```

Trong lần đầu tiên *create\_entry.php* được gọi, form *sign.php* sẽ được hiển thị. Kế tiếp, dữ liệu sẽ được cập nhật vào CSDL.

Hình sau minh họa các form được hiển thị:

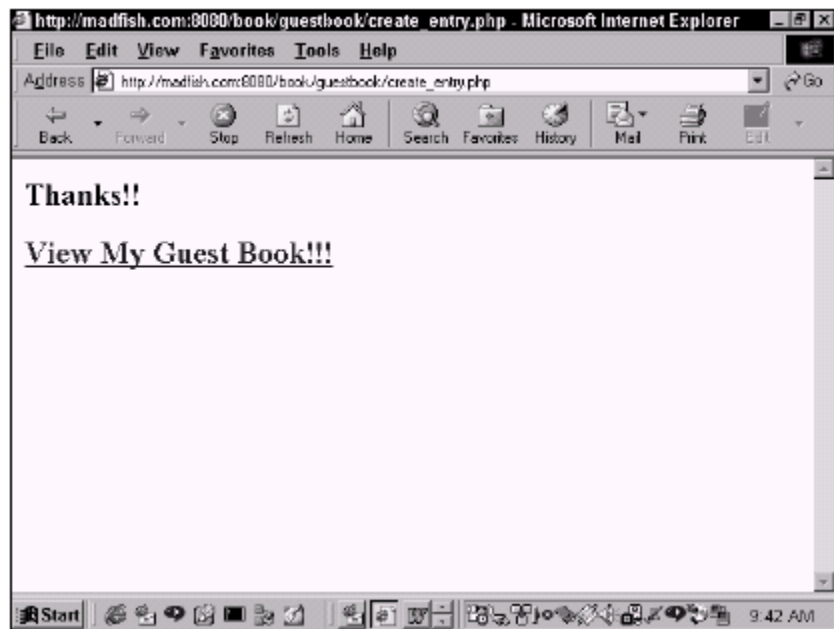


The screenshot shows a Microsoft Internet Explorer window with the address bar displaying `http://madfish.com:8080/book/guestbook/sign.php`. The page title is "Sign my Guest Book!!!". The form contains the following fields and controls:

- Name:** Text input field containing "Jay Greenspan".
- Location:** Text input field containing "San Francisco".
- Email:** Text input field containing "jgreen\_1@yahoo.com".
- Home Page URL:** Text input field containing "http://www.phpmysqlbook.com".
- Comments:** A large text area containing the text "Love it.".
- Buttons:** Two buttons at the bottom: "Sign" and "Start Over".

The Windows taskbar at the bottom shows the Start button, several application icons, and the system clock displaying "9:41 AM".

*sign.php*



`create_entry.php`

## Hiển thị dữ liệu trong Database lên màn hình

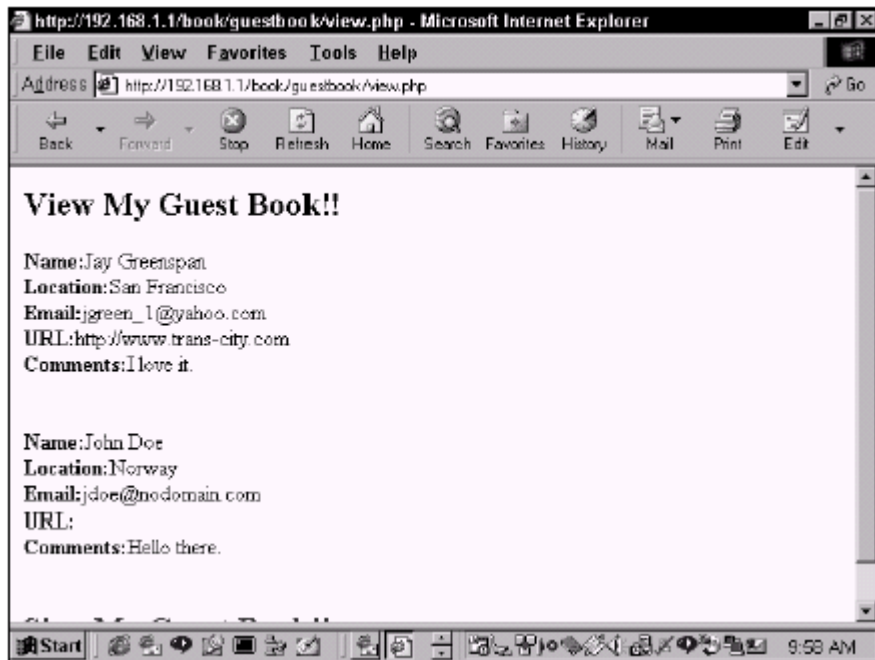
Bây giờ dữ liệu đã được ghi vào CSDL guestbook . Bạn cần thực hiện việc xem các dữ liệu đó. Nên nhớ là chúng ta lại phải sử dụng *dbconnect.php* như tôi đã nói với bạn trước đây. Bạn cần phải cho hiển thị tất cả các record trong table thông tin của khách viếng thăm đã nhập vào. Chúng ta thực hiện script sau và đặt tên là **view.php**:

```
<?php include("dbconnect.php"); ?>
<h2>View My Guest Book!!</h2>
<?php
$result = mysql_query("select * from guestbook") or
die (mysql_error());
while ($row = mysql_fetch_array($result))
{
echo "<b>Name:</b>";
echo $row["name"];
echo "<br>\n";
echo "<b>Location:</b>";
echo $row["location"];
echo "<br>\n";
echo "<b>Email:</b>";
echo $row["email"];
echo "<br>\n";
```

```
echo "<b>URL:</b>";  
echo $row["url"];  
echo "<br>\n";  
echo "<b>Comments:</b>";  
echo $row["comments"];  
echo "<br>\n";  
echo "<br>\n";  
echo "<br>\n";  
}  
mysql_free_result($result);  
?>  
<h2><a href="sign.php">Sign My Guest Book!!</a></h2>
```

Như chúng ta thấy query trong MySQL truy cập tất cả các hàng trong database. Script thực hiện việc này bằng cách sử dụng vòng lặp thông qua biến \$row.

Trong mỗi vòng lặp thì mỗi field trong từng record được hiển thị. Vd: print \$row["email"] sẽ ghi ra màn hình đối với record đang truy cập. Khi chạy chương trình, tất cả các field của từng record sẽ được hiển thị:



[view.php](#)



# Chương trình này upload lên internet được chưa?

Bây giờ thì ứng dụng đầu tiên của bạn đã hoàn tất. Nếu bạn muốn upload web vừa tạo lên trên Server để thử nghiệm thì cũng được thôi. Nhưng muốn để nó trở thành một site guestbook đúng nghĩa thì chưa được đâu! Bạn còn cần phải làm nhiều thứ để dữ liệu bạn không bị hacker quấy phá.... **(Còn tiếp)**

Lưu ý:

Bạn có thể tìm thấy các CD thiết kế web PHP tại các cửa hàng CD (Tôn Thất Tùng, tp.HCM chẳng hạn). Trên CD có trình cài đặt Apache, PHP, MySQL thông dụng là FOX. Ngoài ra còn có các Website PHP mẫu.

**CÁC LỆNH Ở BÀI HỌC TRÊN TỐT NHẤT BẠN NÊN GÕ LẠI, KHÔNG NÊN COPY!**

[tongphuockhai@mail15.com](mailto:tongphuockhai@mail15.com)

*- Kiến thức là kho báu không phải của riêng ai. Vì vậy bạn đừng ngần ngại khi chia sẻ cho mọi người.*

# LẬP TRÌNH WEB ĐỘNG VỚI **PHP / MySQL**

- ❖ GUESTBOOK
- ❖ CATALOG
- ❖ FORUM
- ❖ SHOPPING CART

## PHẦN 2

**Tống Phước Khải (tổng hợp & biên dịch)**

# **Phương pháp truy xuất CSDL**

## **MySQL**

- 1- Từ Command Prompt**
- 2- Bằng lệnh PHP**
- 3- Dùng phpMyadmin**

Để hiểu được chương này một cách rõ ràng, trước tiên bạn cần phải có một số kiến thức cơ bản về **Cơ Sở Dữ Liệu quan hệ**. Nếu bạn đã học qua một khoá căn bản về MS Access trong chương trình đào tạo chứng chỉ B chẳng hạn thì hẳn nhiên bạn có thể tiếp tục. Còn nếu như bạn chưa biết gì về nó thì tôi sẽ bàn đến nó trong phần Phụ Lục của giáo trình này hoặc bạn có thể tìm ngay một tài liệu tham khảo về CSDL, dễ nhất là tài liệu về MS Access ...

... Tôi chắc rằng bây giờ bạn đã có kiến thức về CSDL và hiểu biết Table là gì rồi! Có hàng khối công việc bạn sẽ phải làm việc đối với các Table và bạn sẽ được hướng dẫn cặn kẽ trong quyển sách này. Bạn sẽ phải vượt qua một số kiến thức về nó để mới có thể thành thạo trong thao tác với Table. Như bạn biết đấy: *Con đường đi đến thành công không có trải thảm sẵn đâu!*

Nếu bạn đã từng làm việc với *MS SQL Server* hay *Access* chúng đều có hỗ trợ việc tạo CSDL rất là dễ dàng với giao diện trực quan. Đối với MySQL bạn cũng có thể sử dụng công cụ trực quan đó là *phpMyadmin*.

Tuy nhiên, bạn phải học cách thao tác với CSDL bằng dòng lệnh, tôi chắc rằng điều này sẽ rất có ích cho bạn. Muốn chương trình của bạn trong lúc chạy thao tác tự động với CSDL thì bạn cần hàng tá lệnh PHP/SQL để thực hiện các yêu cầu của chương trình.

Trước khi chúng ta tạo các table trong CSDL của MySQL, có một vài thứ bạn cần phải hiểu rõ. Những khái niệm cơ bản mà tôi sắp giới thiệu sau đây rất quan trọng. Bạn hãy chắc rằng mình đã nắm kỹ về chúng trước khi thực hiện việc thiết kế dữ liệu.

## Null

Việc đầu tiên bạn phải làm trong việc thiết kế một table là quyết định xem một field có cho phép giá trị NULL hay không.

Trong CSDL quan hệ, giá trị NULL của một field đồng nghĩa với nó có thể chấp nhận không có dữ liệu trong đó. Nên nhớ rằng giá trị NULL khác với giá trị của một chuỗi không có ký tự trong đó hoặc số có giá trị 0.

Đôi khi trong chương trình, bạn sẽ thực hiện một số động tác so sánh xem một chuỗi nào đó có chứa giá trị hay không, nó có thể là một câu lệnh IF. Xét một ví dụ PHP như sau:

```
$var //this is a variable used in the test
if ($var == "")
{
echo "Var is an empty string";
} else {
echo $var;
}
```

Nếu bạn thực hiện việc so sánh xem giá trị số có phải 0 hay không thì cũng thực hiện tương tự.

Động tác so sánh trên sẽ không làm việc được đối với giá trị NULL. Bạn nên hiểu rằng NULL là không chứa bất kỳ giá trị gì trong đó, cho nên việc so sánh trị không mang ý nghĩa gì cả. Trong chương 3 bạn sẽ thấy rằng đối với giá trị NULL đòi hỏi lập trình viên phải rất cẩn nhắc khi viết lệnh liên kết table.

*Trong lệnh SELECT của SQL, có một số cách để bạn có thể kiểm tra nếu như một field chứa giá trị NULL. Trước hết bạn hãy sử dụng hàm Isnull(). Giả sử tìm một record trong table mà giá trị middle\_name là NULL, bạn có thể sử dụng query sau:*

**select \* from names where isnull(middle\_name);**

*Hoặc lấy các record mà middle\_name khác NULL:*

**select \* from names where !isnull(middle\_name);**

*Bạn cũng có thể sử dụng **is null** và **is not null**:*

**select \* from users where addr2 is null;**

**select \* from users where addr2 is not null;**

Để rõ hơn, bạn hãy xem chuyện gì xảy ra khi tôi cố gắng liên kết hai table sau:

### Khach\_hang

ten	ho_lot	ba_xa
Trung	Le Anh	1321
Khai	Tong Phuoc	Null

### Hon\_nhan

ba_xa	ten	ho_lot
1321	Diem	Nguyen Thuy

Nếu bạn muốn tìm tên các **khách hàng** và tên những **bà xã** của họ, bạn sẽ phải liên kết 2 table này thông qua field ba\_xa. (Xin bạn chờ lo lắng khi chưa hiểu về cú pháp, bạn sẽ học ngay ở phần tiếp theo thôi).

```
SELECT * FROM khach_hang, hon_nhan  
WHERE khach_hang.ba_xa = hon_nhan.ba_xa
```

Việc thực hiện này chỉ đúng đối với Trung, nhưng sẽ có vấn đề đối với Khai bởi vì anh ta hãy còn độc thân và ba\_xa của anh ta là NULL.

Trong chương 3 bạn sẽ khảo sát kỹ hơn về vấn đề này.

## Index

Người ta nói rằng ưu điểm vượt trội của Hệ quản trị CSDL quan hệ là nó thực hiện các việc tìm kiếm hay sắp xếp những khối lượng dữ liệu khổng lồ một cách rất nhanh chóng. Sở dĩ nó thực hiện được việc này là do nó có chứa một cơ cấu lưu trữ dữ liệu gọi là INDEX.

INDEX cho phép database server tạo được một field đặc trưng tìm kiếm với tốc độ khó ngờ. Các INDEX đặc biệt hỗ trợ một hoặc một nhóm các record trong một table chứa số lượng lớn các record. Chúng cũng hỗ trợ tốc độ cho các hàm liên kết hoặc tách nhóm dữ liệu như min(), max(), bạn sẽ tìm hiểu ở Chương 3.

Với các tính năng vượt trội này, tại sao người ta lại không tạo index trong tất cả các field của một table? Có một số điều trở ngại như sau: Thứ nhất, index có sẽ làm chậm một số tiến trình trong CSDL. Mỗi lần bảo trì các index Database Server phải mất khá nhiều thời gian. Có một vài trường hợp chính các index làm cho chúng chậm hẳn. Nếu như trên table của bạn tất cả các record đều giống y như nhau thì không có lý do gì để bạn tạo index. Các index dư thừa chỉ làm cho tốn thêm không gian đĩa của bạn mà thôi.



Đối với một table có gắn khoá chính (primary key) thì thông thường field có khoá này được dùng vào việc tìm kiếm cho nên index sẽ được gán tự động trên field này. Bạn sẽ gặp lại rất nhiều công việc tạo index ở các phần tiếp theo sau.

## Lệnh CREATE Database

Trước khi tạo được một Table thì điều tất yếu là bạn phải tạo được một Database cái đã. Việc này dễ dàng và nhanh chóng thôi. Lệnh CREATE được sử dụng như sau:

```
mysql> create database database_name;
```

*Nếu như bạn thắc mắc rằng sau khi tạo database nó sẽ nằm ở trong thư mục nào trên ổ đĩa của bạn thì bạn hãy tìm trong `..\mysql\data` xem có không.*

Khi đặt tên cho database, hay đặt tên cho field và index gì đấy tránh trường hợp đặt những cái tên khó nhớ hoặc dễ bị lẫn lộn. Đối với một số hệ thống Unix chẳng hạn có sự phân biệt chữ HOA/thường thì CSDL chạy trên nó cũng ảnh hưởng theo.

Bạn hãy chọn một quy ước cho riêng mình trong việc đặt tên để khỏi nhầm lẫn về sau. Chẳng hạn tên của table và field đều đặt chữ thường chẳng hạn. Nên nhớ là không được sử dụng khoảng trắng.

Bây giờ bạn tìm hiểu cả hai cách tạo database: Cách thứ nhất tạo thủ công từ dấu nhắc dòng lệnh DOS, cách thứ hai sử dụng các lệnh trong PHP.

Cách thứ nhất tôi đã có trình bày ở chương giới thiệu và bạn đã tạo một database tên là guestbook. Cú pháp tạo như sau:

```
mysql> create database guestbook;
```

Cách thứ hai là sử dụng lệnh trong PHP, bạn có thể dùng hàm `mysql_create_db()` hoặc `mysql_query()`. Nhưng nên nhớ trước khi tạo bạn phải thực hiện được kết nối với database server.

```
$conn = mysql_connect("localhost", "username", "password")  
or die ("Could not connect to localhost");  
mysql_create_db("my_database") or  
die ("Could not create database");  
$string = "create database my_other_db";  
mysql_query($string) or  
die(mysql_error());
```

# Lệnh USE Database

Sau khi đã tạo được một database mới trong database server bạn sẽ bắt đầu chọn nó để sử dụng cho công việc của mình. Cách thực hiện như sau:

## 1. Command Prompt:

```
mysql> use database_name;
```

## 2. Trong PHP:

```
$conn = mysql_connect("localhost", "username", "password")  
or die ("Could not connect to localhost");  
mysql_select_db("test", $conn) or  
die ("Could not select database");
```

# Lệnh CREATE Table

*Lưu ý: Lệnh này thực hiện sau khi đã có lệnh **CREATE Database**.*

Một khi bạn đã tạo và chọn database, việc tiếp theo là tạo một table. Bạn sẽ sử dụng lệnh Create Table như sau:

```
create table table_name  
(  
column_1 column_type column_attributes,  
column_2 column_type column_attributes,  
primary key (column_name),  
index index_name(column_name)  
)
```

Đối với thuộc tính các field (cột) chúng ta cần bàn về:

- **null** hoặc **not null**
- **default**

Nếu bạn không định nghĩa NULL hay NOT NULL thì NULL sẽ được chọn làm giá trị mặc định. Hãy xét ví dụ sau:

```
create table topics2 (  
  topic_id integer not null auto_increment,  
  parent_id integer default 0 not null,  
  root_id integer default 0,  
  name varchar(255),  
  description text null,  
  create_dt timestamp,  
  modify_dt timestamp,  
  author varchar(255) null,  
  author_host varchar(255) null,  
  primary key(topic_id),  
  index my_index(parent_id))
```

Trong ví dụ trên bạn tạo ra một table có tên topics2, có tất cả 8 field và có 2 index, một index cho khoá chính và một cho parent\_id. Type của các field trên lần lượt là integer, varchar, text, timestamp. Giá trị đứng sau default là giá trị mặc định bạn gán cho một ô trong record khi không nhập liệu vào.

Bây giờ chúng ta áp dụng các lệnh này vào một chương trình PHP để tạo table, hàm `mysql_query ()` được sử dụng:

```
$conn = mysql_connect("localhost", "username", "password") or  
die ("Could not connect to localhost");  
mysql_select_db("test", $conn) or  
die("could not select database");  
$query = "create table my_table (col_1 int not null primary key,  
col_2 text)";  
mysql_query($query) or  
die(mysql_error());
```

\* *Lưu ý: username và password tùy thuộc vào MySQL của bạn. Ví dụ:*

```
$conn = mysql_connect("localhost", "minhtrung", "zadfdfaked") or
```

*Thông thường các nhà cung cấp host PHP sử dụng localhost, tuy nhiên một số hosting không sử dụng localhost như Yahoo chẳng hạn. Do đó bạn cần xem hướng dẫn của nhà cung cấp host.*

# Kiểu dữ liệu

Bây giờ chúng ta hãy bàn về các kiểu dữ liệu (type) gán cho field trong table. Có rất nhiều kiểu khác nhau chúng ta sẽ lần lượt khảo sát từng kiểu một.

## Kiểu chuỗi văn bản

MySQL có 7 kiểu dành cho dữ liệu kiểu chuỗi văn bản:

- ☐ char
- ☐ varchar
- ☐ tinytext
- ☐ text
- ☐ mediumtext
- ☐ longtext
- ☐ enum

## CHAR

Cách sử dụng: `char(length)`

Length có giá trị tối đa là 255. Giả sử bạn sử dụng khai báo `char(10)` thì bạn chỉ được phép nhập vào tối đa 10 ký tự mà thôi.

## VARCHAR

Cách sử dụng: `varchar(length)`

Kiểu này cũng gần giống như kiểu CHAR có độ dài tối đa cũng là 255. Điểm khác biệt của varchar là nó chỉ là biến lưu trữ độ dài, cho nên nó sẽ không thay đổi khi giá trị của ô dữ liệu dài hay ngắn. MySQL sẽ sinh ra một ký dùng làm biến chứa độ dài của field kiểu varchar. Đồng thời MySQL sẽ thực hiện chức năng loại bỏ các khoảng trống trong mỗi ô dữ liệu nếu như không được sử dụng hết.

## USING CHAR OR VARCHAR

Có sự khác nhau trong việc sử dụng CHAR và VARCHAR. Sau đây là phương hướng lựa chọn của bạn.

Giả dụ bạn tạo một field là ĐỊA CHỈ và bạn dự tính độ dài tối đa là 150. Có những trường hợp địa chỉ rất ngắn ví dụ: *1 Lê Lợi, Q.1, TPHCM*. Bạn chỉ sử dụng có 20 ký tự, như vậy còn



trống rất nhiều ký tự không dùng đến. Trong trường hợp này bạn nên sử dụng kiểu VARCHAR (150).

Trường hợp field của bạn là MÃ SỐ chẳng hạn, và bạn cho độ dài tối đa là 6 theo quy ước tạo mã của bạn ví dụ: KH0001 . Trong trường hợp này các ô khác đều được nhập theo chuẩn định sẵn luôn luôn là 6 ký tự cho nên không việc gì bạn phải sử dụng VARCHAR để MySQL phải nhọc công theo dõi độ dài của các ô mỗi khi nhập vào. Bạn chỉ sử dụng VAR(6) là được.

Trong trường hợp bạn chọn kiểu dữ liệu là varchar(4) thì MySQL sẽ tự động đổi lại là kiểu char.

## TINYTEXT

Cách sử dụng: tinytext

Đây là một trong bốn kiểu text nhị phân. Tất cả 4 kiểu này (tinytext, text, mediumtext, largertext) đều là kiểu biến tương tự như varchar. Chúng khác nhau về độ dài của cho phép của ký tự mà thôi. Tuy nhiên, đối với TINYTEXT thì cho phép độ dài tối đa là 255, giống như varchar(255). Cho phép tạo index trên toàn bộ các ký tự của field này.

## **TEXT**

Cách sử dụng: `text`

Cho phép độ dài tối đa là 65,535 ký tự. Có thể tạo index trên 255 ký tự đầu.

## **MEDIUMTEXT**

Cách sử dụng: `mediumtext`

Cho phép độ dài tối đa là 16,777,215 ký tự. Có thể tạo index trên 255 ký tự đầu.

## **LONGTEXT**

Cách sử dụng: `longtext`

Cho phép độ dài tối đa là 4,294,967,295 ký tự. Có thể tạo index trên 255 ký tự đầu. Tuy nhiên loại này không thông dụng bởi vì MySQL chỉ hỗ trợ chuỗi 16 triệu bytes.

## **ENUM**

Cách sử dụng: `enum ('value1', 'value2', 'value3' ...) [default 'value']`

Với enum bạn có thể giới hạn các giá trị được định sẵn cho một field. Cho phép bạn định trước tối đa 65.535 giá trị.

Thông thường người ta dùng kiểu này cho field chứa giá trị Yes hoặc No. Ví dụ:

```
create table my_table (  
id int auto_increment primary key,  
answer enum ('yes', 'no') default 'no'  
);
```

## SET

Cách sử dụng: set ('value1', 'value2', 'value3' ...) [default 'value']

Kiểu này định nghĩa một tập hợp hàng loạt các giá trị định trước. Tuy nhiên, cách này ít được dùng bởi vì nó phá vỡ cấu trúc thiết kế CSDL (một field có quá nhiều kiểu) và các bạn sẽ không thấy tôi sử dụng trong quyển sách này.

## Kiểu dữ liệu số

MySQL có tất cả 7 kiểu số. Lưu ý rằng các kiểu sau đây là giống nhau: int/ integer, double/double precision/real, decimal/numeric

- ❑ int/integer
- ❑ tinyint
- ❑ mediumint
- ❑ bigint

- ❑ float
- ❑ double/double precision/real
- ❑ decimal/numeric

Đối với tất cả kiểu số, giá trị lớn nhất cho phép là 255. Đối với một số kiểu cho phép bạn thể hiện các số 0 đứng đầu. Giả sử bạn có một field có độ dài là 10 thì số 25 trong ô dữ liệu sẽ được thể hiện 0000000025. Field số còn được định nghĩa là có dấu (signed) hoặc không dấu (unsigned). Mặc định là có.

## INT/INTEGER

Cách sử dụng: `int(display size) [unsigned] [zerofill]`

Nếu bạn dùng không dấu thì giá trị của field cho phép là từ 0 đến 4.294.967.295. Nếu dùng có dấu thì giá trị từ -2.147.483.648 đến 2.147.483.647. Kiểu Int sẽ sử dụng `auto_increment` (tự động theo chiều tăng) để định nghĩa khoá chính của table.

```
create table my_table (  
table_id int unsigned auto_increment primary key,  
next_column text  
);
```

Để ý rằng bạn sử dụng không dấu (unsigned) bởi vì auto\_increment không dùng cho các giá trị âm.

## **TINYINT**

Cách sử dụng: `tinyint(display size) [unsigned] [zerofill]`

Nếu không dấu, tinyint sẽ chứa các giá trị nguyên từ 0 đến 255. Nếu có dấu thì từ -128 đến 127.

## **MEDIUMINT**

Cách sử dụng: `mediumint(display size) [unsigned] [zerofill]`

Có dấu: có giá trị từ -8.388.608 đến 8.388.607

Không dấu: có giá trị từ 0 đến 1677215

## **BIGINT**

Cách sử dụng: `bigint(display size) [unsigned] [zerofill]`

Có dấu: -9.223.372.036.854.775.808 đến 9.223.372.036.854.775.807

Không dấu: from 0 to 18.446.744.073.709.551.615

## **FLOAT**

Cách sử dụng 1: `FLOAT(precision) [zerofill]`

Với cách sử dụng này, cho phép chứa các số thập phân không dấu. Số lượng phần thập phân có thể là  $\leq 24$  đối với loại single và 25 đến 53 đối với loại double. Các version trước đây của MySQL, luôn chia làm 2 loại:

Cách sử dụng 2: `FLOAT[(M,D)] [ZEROFILL]`

Đây là loại single và giá trị có thể là từ  $-3,402823466E+38$  đến  $-1,175494351E-38$ , số 0, và từ  $1,175494351E-38$  đến  $3,402823466E+38$ . M là phần nguyên, D là phần thập phân. ????

## **DOUBLE/DOUBLE PRECISION/REAL**

Cách sử dụng 1: `DOUBLE[(M,D)] [zerofill]`

Cho phép giá trị từ  $-1,7976931348623157E+308$  đến  $-2,2250738585072014E-308$ , số 0 và  $2,2250738585072014E-308$  đến  $1,7976931348623157E+308$ . M là phần nguyên, D là phần thập phân.

Cách sử dụng 2: `DECIMAL[(M[,D])] [ZEROFILL]`

Các số trong phần thập phân được lưu trữ như ký tự. Mỗi số được xem như một ký tự chuỗi. Nếu  $D = 0$  thì sẽ không có phần thập phân. Giá trị thập phân giống như dạng Double. ????

## Kiểu dữ liệu ngày, giờ

MySQL có 5 dạng ngày giờ:

- ❑ date \_ time
- ❑ datetime \_ year
- ❑ timestamp
- ❑ time
- ❑ year

Ngày và giờ trong MySQL rất uyển chuyển, nó có thể chấp nhận kiểu chuỗi hoặc số hãy xét ví dụ sau:

```
create table date_test(  
id int unsigned auto_increment,  
a_date date  
);
```

Sau đó dùng insert để đưa giá trị ngày vào a\_date:

```
insert into date_test (a_date) values ('00-06-01');  
insert into date_test (a_date) values ('2000-06-01');
```

```
insert into date_test (a_date) values ('20000601');  
insert into test6 (a_date) values (000601);
```

MySQL tương thích với việc nhận giá trị ngày là kiểu chuỗi hơn. Cho nên "000501 là chọn lựa thích hợp hơn là việc nhập một số nguyên. Sử dụng giá trị chuỗi cho ngày giúp bạn sẽ tránh được một số sự cố về sau.

MySQL có hỗ trợ một số hàm giúp bạn trong việc rút trích dữ liệu dạng ngày.

## **DATE**

Cách sử dụng: date

Định dạng của ngày như sau: YYYY-MM-DD. Cho phép bạn các giá trị từ 1000-01-01 đến 9999-12-31.

## **DATETIME**

Cách sử dụng: datetime [null | not null] [default]

Định dạng của ngày giờ như sau: YYYY-MM-DD HH:MM:SS. Cho phép bạn các giá trị từ 1000-01-01 00:00:00 đến 9999-12-31 23:59:59.



## **TIMESTAMP**

Cách sử dụng: timestamp(size)

Đây là kiểu dữ liệu ghi nhận tự động giờ giấc sửa đổi gần nhất đối với một record, bất khi khi nào nó được tạo ra, hoặc cập nhật đều xảy ra việc ghi nhận này. Size của nó có thể định nghĩa trong khoảng từ 2 đến 14. Bảng sau trình bày các size. Giá trị mặc định là 14.

<b>Size</b>	<b>Định dạng</b>
<b>2</b>	<b>YY</b>
<b>4</b>	<b>YYMM</b>
<b>6</b>	<b>YYMMDD</b>
<b>8</b>	<b>YYYYMMDD</b>
<b>10</b>	<b>YYMMDDHHMM</b>
<b>12</b>	<b>YYMMDDHHMMSS</b>
<b>14</b>	<b>YYYYMMDDHHMMSS</b>

## TIME

Cách sử dụng: time

Lưu trữ dạng giờ theo định dạng HH:MM:SS và có giá trị từ -838:59:59 đến 838:59:59. Lý do mà giá trị này lớn như vậy là để nó có thể chứa được các kết quả tính toán giờ giấc.

## YEAR

Cách sử dụng: year[(2|4)]

Chứa dữ liệu dạng năm. Nếu sử dụng hai ký tự để biểu thị năm thì biểu diễn được từ Có giá trị từ 1970 cho đến 2069, nên nhớ: 70 đến 99 biểu thị từ 1970 đến 1999, còn 01 đến 69 biểu thị từ 2001 đến 2069.

Dùng 4 ký tự thì biểu diễn được từ 1901 đến 2155.

# Tạo chỉ mục INDEX

Bắt đầu từ phiên bản 3.23.6 của MySQL bạn có thể tạo index trên bất kỳ field nào. Cho phép 1 table có 16 field chứa index. Cú pháp như sau:

```
index index_name (indexed_column)
```

Mặc dù index\_name là tùy chọn, nhưng bạn nên luôn luôn cho nó một cái tên. Sau này nó rất cần thiết khi bạn muốn xoá bỏ index của một field nào đó trong lệnh SQL của bạn. Nếu bạn không cho tên thì MySQL sẽ chọn tên index của field đầu tiên.

Còn một cách nữa để tạo index là khai báo khoá chính trên field đó. Chú ý rằng bất kỳ field auto\_increment (sort tự động) cũng phải được index, và bạn nên khai báo nó là khoá chính. Trong ví dụ sau id\_col được index:

```
create table my_table (  
id_col int unsigned auto_increment primary key,  
another_col text  
);
```

Khoá chính có thể cũng được khai báo giống như các index khác ngay sau khi định nghĩa một field:

```
create table my_table (  
id_col int unsigned not null auto_increment,  
another_col text,  
primary key(id_col)  
);
```

Index có thể trải rộng ra hơn một cột. Nếu như query sử dụng 2 hàng phối hợp với nhau trong khi thực hiện việc tìm kiếm, bạn có thể tạo một index bao gồm luôn cả 2 với các lệnh sau:

```
create table mytable(  
id_col int unsigned not null,  
another_col char(200) not null,  
index dual_col_index(id_col, another_col)  
);
```

Index này sẽ được sử dụng cho việc tìm kiếm vừa trên id\_col vừa trên another\_col. Các index này làm việc từ trái sang phải. Do đó index này sẽ được sử dụng cho việc tìm kiếm exclusive trên id\_col. Tuy nhiên, nó sẽ không exclusive cho việc tìm kiếm trên another\_col.

Còn một điều về index nữa là bạn có thể tạo nó chỉ trên một phần của field. Bắt đầu từ phiên bản 3.23 của MySQL bạn có thể tạo index các field kiểu index tinytext, text, mediumtext và longtext trên 255 ký tự đầu. Đối với char và varchar, bạn có thể tạo index trên một số ký tự đầu của field. Cú pháp của nó như sau:

**index index\_name (column\_name(column\_length))**

Ví dụ:

```
create table my_table(  
char_column char (255) not null,  
text_column text not null,  
index index_on_char (char_column(20)),  
index index_on_text (text_column(200))  
);
```

Một index có thể đảm bảo giá trị duy nhất tồn tại trong mọi record của table bằng cách sử dụng lệnh *unique*.

```
create table my_table(  
char_column char (255) not null,  
text_column text not null,  
unique index index_on_char (char_column));
```

# Các loại Table

MySQL hỗ trợ các dạng table sau: ISAM, MyISAM, BDB và Heap. ISAM là dạng table đã xưa và trong các ứng dụng mới không được sử dụng. Dạng table mặc định là MyISAM. Cú pháp để khai báo loại table này là:

```
create table table_name type=table_type(  
col_name column attribute  
);
```

Table dạng MyISAM có tốc độ tốt và tính ổn định cao. Không cần thiết phải định nghĩa một dạng table mới nào khác trừ phi bạn cần dùng loại khác cho vì một lý do đặc biệt nào đó.

Heap là dạng table thường trú trong bộ nhớ. Chúng không được lưu trữ trong đĩa cứng hay các thiết bị dùng để trữ tin. Cho nên nếu bị mất điện heap sẽ mất theo. Vì được trữ trong bộ nhớ nên heap có tốc độ truy cập cao. Tuy nhiên bạn chỉ dùng cho các table tạm thời trong lúc chạy chương trình.

## Lệnh Alter table

Nếu như bạn muốn thay đổi các thành phần của table bạn sẽ sử dụng lệnh *alter table*. Bạn có thể thực hiện các thay đổi như: đổi tên table, field, index; thêm hoặc xoá field và index; định nghĩa lại các field và index. Cú pháp cơ bản của lệnh này là:

```
alter table table_name
```

Các lệnh còn lại tùy thuộc vào thao tác mà chúng ta sẽ bàn tiếp theo đây:

## Đổi tên Table

Cú pháp như sau:

```
alter table table_name rename new_table_name
```

If you have MySQL version 3.23.27 or higher you can make use of the rename statement. The basic syntax is

```
rename table_name to new_table_name
```

## Thêm và xoá cột trong Table

Khi thêm field vào trong table, bạn sẽ cần phải có những định nghĩa cần thiết cho field đó. Từ phiên bản 3.22 của MySQL cho phép bạn chọn vị trí để đặt field mới vào trong table. Chức năng này không bắt buộc.

```
alter table table_name add column column_name column attributes
```

Ví dụ:

```
alter table my_table add column my_column text not null
```

Cách sử dụng định vị một field trong table: Sử dụng lệnh first để chèn field mới vào vị trí đầu của table. After để chèn vào vị trí sau cùng trên table:

```
alter table my_table add column my_next_col text not null first  
alter table my_table add column my_next_col text not null after  
my_other_column
```

Để xoá một field, bạn thực hiện lệnh sau:



`alter table table_name drop column column name`

Khi thực hiện lệnh alter đối với một table, bạn chỉ nên thực hiện một thao tác đối với table mà thôi. Ví dụ: Bạn thực hiện việc xoá một index, sau đó tạo một index mới, thì không nên gộp chung vào một lệnh alter mà nên thực hiện 2 lần.

## Thêm và xoá Index

Bạn có thể thêm index bằng cách sử dụng lệnh index, unique và primary key, tương tự như việc sử dụng lệnh create vậy.

```
alter table my_table add index index_name (column_name1,  
column_name2, ...)
```

```
alter table my_table add unique index_name(column_name)
```

```
alter table my_table add primary key(my_column)
```

Bạn cũng có thể bỏ index bằng cách sử dụng lệnh drop:

```
alter table table_name drop index index_name
```

```
alter table_name test10 drop primary key
```

## Đổi thuộc tính của cột (field)

Thay đổi cách thành tố của field bằng lệnh change hoặc modify:

```
alter table table_name change original_column_name new_column_name  
int not null
```

Lệnh sau là sai:

```
alter table table_name change my_col2 my_col3;
```

Nếu bạn muốn chỉ đổi thuộc tính của field thì dùng lệnh change và tạo field mới cùng tên nhưng thay đổi thuộc tính. Giả dụ bạn đổi field col\_1 từ kiểu char(200) sang varchar(200):

```
alter table table_name change col_1 col_1 varchar(200)
```

Với MySQL phiên bản 2.22.16 trở đi bạn có thể dùng lệnh modify:

```
alter table table_name modify 1 col_1 varchar(200)
```

## Lệnh Insert

Bây giờ bạn đã biết tất cả những kiến thức cần thiết để tạo và thực hiện các thay đổi trên định dạng table, bây giờ bạn sẽ tìm hiểu cách thức để nhập thông tin vào table. Bạn sẽ thực hiện việc này bằng lệnh INSERT:

```
insert into table_name (column_1, column2, column3,...) values  
(value1, value2, value3 ...)
```

Nếu trong một field cho phép giá trị NULL bạn có thể không cần đưa vào trong lệnh INSERT.

Như bạn thấy các chuỗi tên field và giá trị đều được đặt trong dấu ngoặc đơn (). Ngoài ra giá trị kiểu chuỗi phải được bao bọc bởi dấu nháy đơn. Như vậy nếu như trong chuỗi có chứa dấu nháy đơn và dấu ngoặc đơn thì sao? Để tránh trường hợp lẫn lộn giữa dấu chuỗi và dấu phân định nghĩa lệnh SQL có các qui định rằng muốn đưa ký hiệu đặc biệt vào chuỗi thì phải đặt chúng sau dấu \

‘ (single quote)

“ (double quote)

\ (backslash)

% (percent sign)

\_ (underscore)

Bạn có thể thoát ra khỏi dấu nháy đơn bằng cách sử dụng 2 dấu ngoặc đơn đóng mở ().  
Bạn sử dụng các ký tự đặc biệt sau để thực hiện một thao tác đặc thù trong câu lệnh:

\n (newline)

\t (tab)

\r (carriage return)

\b (back space)

Nên lưu ý một điều là, bạn không cần phải lo lắng về các ký tự thoát ở đây trong khi lập trình PHP. Bạn sẽ gặp được những hàm và xác lập trong PHP dùng để thực hiện việc này một cách tự động. Hàm addslashes() và các xác lập trong php.ini sẽ hỗ trợ việc này.

## Lệnh Update

Lệnh UPDATE có một chút khác biệt so với các lệnh mà chúng ta đã khảo sát, nó thực hiện thông qua lệnh WHERE. Cú pháp thông thường là:

```
update table_name set col_1=value1, col_2=value_2 where col=value
```

Xin nhắc lại lần nữa, nếu bạn muốn thêm vào một chuỗi, bạn cần phải bao nó trong các dấu nháy đơn và dấu thoát. Nên nhớ rằng lệnh WHERE trong câu lệnh UPDATE có thể thực hiện bất kỳ phép so sánh nào ở phía sau nó. Thông thường nó hay được dùng để định xác định một record đơn với khoá chính. Trong table **folks** sau ID là khoá chính.

id	fname	lname	salary
1	Don	Liu	25,000
2	Don	Corleone	800,000
3	Don	Juan	32,000
4	Don	Johnson	44,500

Câu lệnh sau sẽ tác động đến Don Corleone:

```
update folks set fname='Vito' where id=2
```

Như bạn thấy, nếu như bạn dùng lệnh UPDATE dựa trên field fname thì thật không nên chút nào, vì bạn có thể cập nhật tất cả các field trong table này.

```
update folks set fname='Vito' where fname='don'
```

Bạn cũng có thể dùng UPDATE để thực hiện việc điều chỉnh tăng lương đối với nhân viên chẳng hạn:

```
update folks set salary=50000 where salary<50,000
```

## Lệnh drop table/drop database

Lệnh DROP dùng để xoá table hoặc cả database. Nên nhớ một điều là một khi bạn thực hiện lệnh này rồi thì bạn không thể khôi phục lại dữ liệu của bạn. Hãy cẩn trọng!

```
drop table table_name
```

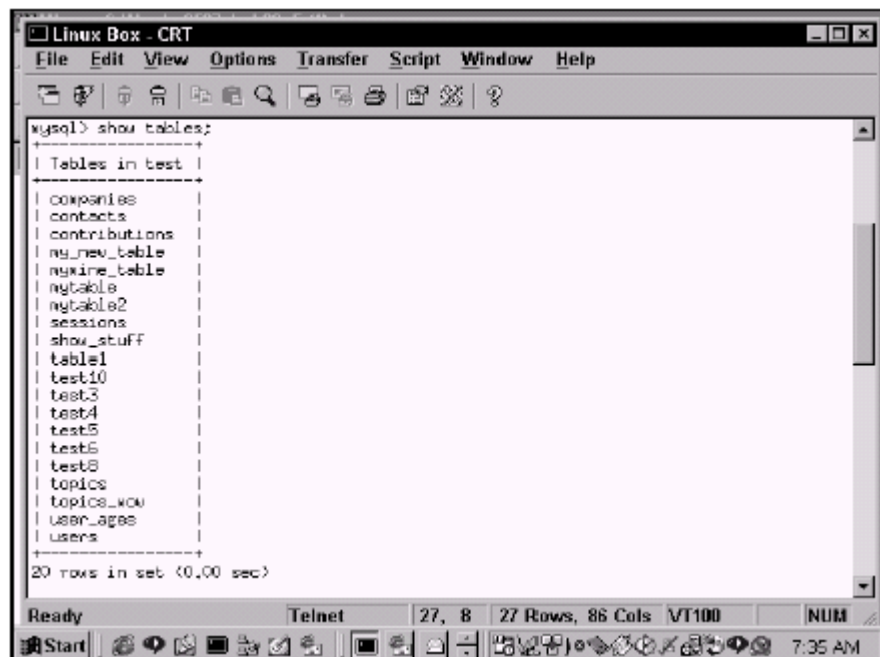
```
drop database database_name
```

Lệnh DROP TABLE được chuyển đổi sang PHP thông qua hàm mysqlquery(). Nếu bạn muốn xoá database trong PHP, bạn cần sử dụng hàm mysql\_drop\_db(). Sẽ được trình bày căn kẽ ở các phần sau!

## Lệnh show tables

Để trình bày một danh sách các table có trong database, bạn sử dụng lệnh SHOW TABLES. Để thực hiện được lệnh này bạn nên lưu ý là phải chọn database trước đã bằng lệnh USE DATABASE

Hình sau là kết quả của việc thực hiện lệnh SHOW TABLES từ dấu nhắc lệnh.



The screenshot shows a terminal window titled "Linux Box - CRT" with a menu bar (File, Edit, View, Options, Transfer, Script, Window, Help) and a toolbar. The terminal displays the MySQL prompt "mysql>" followed by the command "show tables;". The output is a list of tables in the "test" database, enclosed in a dashed box. The tables are: companies, contacts, contributions, my\_new\_table, myxine\_table, mytable, mytable2, sessions, show\_stuff, table1, test10, test3, test4, test5, test6, test8, topics, topics\_xcuv, user\_agas, and users. Below the list, it says "20 rows in set (0.00 sec)". The terminal status bar at the bottom shows "Ready", "Telnet", "27, 8", "27 Rows, 86 Cols", "VT100", and "NUM". The system tray at the very bottom includes a "Start" button, various icons, and the time "7:35 AM".

```
mysql> show tables;
+-----+
| Tables in test |
+-----+
| companies      |
| contacts       |
| contributions   |
| my_new_table    |
| myxine_table    |
| mytable         |
| mytable2        |
| sessions       |
| show_stuff      |
| table1          |
| test10          |
| test3           |
| test4           |
| test5           |
| test6           |
| test8           |
| topics          |
| topics_xcuv     |
| user_agas       |
| users           |
+-----+
20 rows in set (0.00 sec)
```

Lệnh Show table



Trong PHP, bạn có thể cho hiển thị một danh sách table bằng cách sử dụng hàm `MYSQL_LIST_TABLES()`:

```
<?
mysql_connect("localhost", "root", "");
$result = mysql_list_tables("test");
while($row = mysql_fetch_array($result))
{
    echo $row[0] . "<br>\n";
}
?>
```

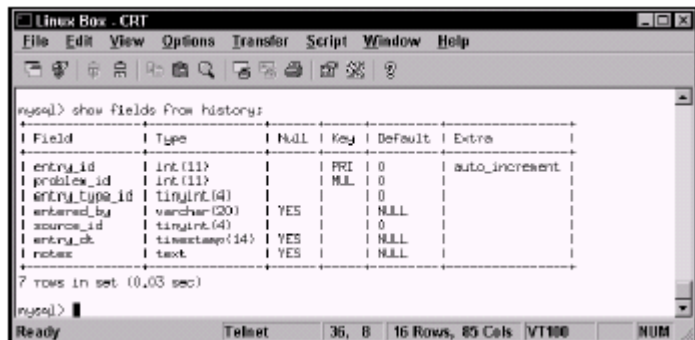
## Lệnh show columns /show fields

Các lệnh cho kết quả giống nhau. Bây giờ bạn hãy dùng lệnh `CREATE` đã học để tạo một table sau đó dùng một trong hai lệnh trên để cho hiển thị các field của nó:

```
create table topics (
    topic_id integer not null auto_increment primary key,
    parent_id integer default 0 not null,
    root_id integer default 0,
    name varchar(255),
```

description text null,  
create\_dt timestamp,  
modify\_dt timestamp,  
author varchar(255) null,  
author\_host varchar(255) null,  
index my\_index(parent\_id)  
)

Hình sau trình bày kết quả của lệnh SHOW FIELDS thực hiện từ dấu nhắc lệnh:



```
mysql> show fields from history;
```

Field	Type	Null	Key	Default	Extra
entry_id	int(11)		PRI	0	auto_increment
problem_id	int(11)		MUL	0	
entry_type_id	tinyint(4)			0	
entered_by	varchar(20)	YES		NULL	
source_id	tinyint(4)			0	
entry_dt	timestamp(14)	YES		NULL	
notes	text	YES		NULL	

7 rows in set (0.03 sec)

```
mysql>
```

Ready Telnet 36, 8 16 Rows, 85 Cols VT100 NUM

**Lệnh Show Fields**

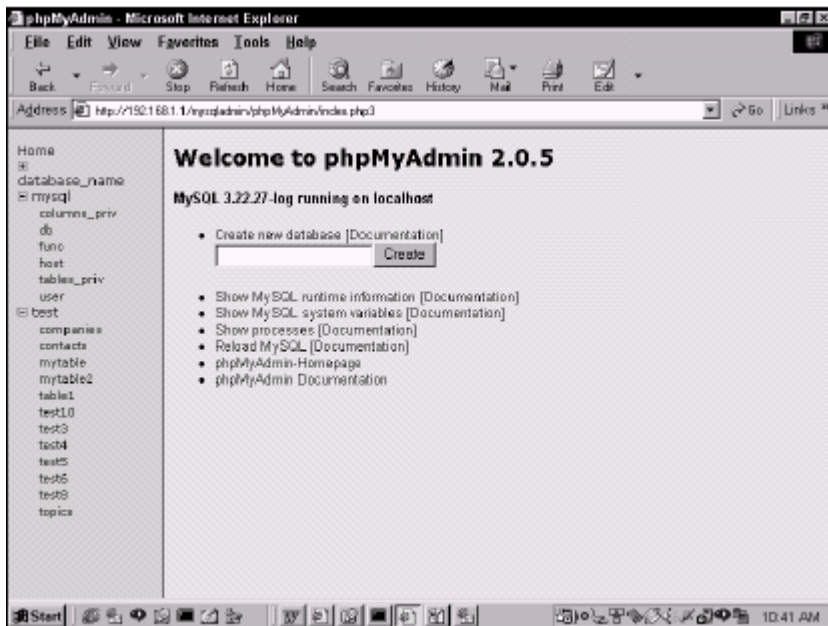
Bạn có thể cho ra kết quả tương tự trong PHP bằng cách sử dụng hàm MYSQL\_FIELD\_NAME(), MYSQL\_FIELD\_TYPE(), và MYSQL\_FIELD\_LEN(). Tất cả các hàm này được trình bày cặn kẽ ở các phần sau.

```
$db = mysql_connect("localhost","root", "")
or die ("Could not connect to localhost");
mysql_select_db("test", $db)
or die ("Could not find test");
$db_name = "topics";
$query = "select * from $db_name";
$result = mysql_query($query);
$num_fields = mysql_num_fields($result);
//create table header
echo "<table border = 1>";
echo "<tr>";
for ($i=0; $i<$num_fields; $i++)
{
echo "<th>";
echo mysql_field_name ($result, $i);
echo "</th>";
```

```
}  
echo "</tr>";  
//end table header  
//create table body  
echo "<tr>";  
for ($i=0; $i<$num_fields; $i++)  
{  
echo "<td valign = top>";  
echo mysql_field_type ($result, $i) . "<br> \n";  
echo "(" . mysql_field_len ($result, $i) . "<br> \n";  
echo mysql_field_flags ($result, $i) . "<br> \n";  
echo "</td>";  
}  
echo "</tr>";  
//end table body  
echo "</table>";
```

## Sử dụng phpMyAdmin

Tôi không loại trừ trường hợp bạn là tín đầu của DẤU NHẮC LỆNH. Có khả năng là bạn không thích GIAO DIỆN ĐỒ HOẠ vì có thể bạn cho rằng giao diện này dành cho những tay mơ mà thôi. Nhưng tôi khuyên bạn rằng bạn hãy tập làm việc với GIAO DIỆN ĐỒ HOẠ. Bản thân tôi xuất thân từ thời dấu nhắc cổ lỗ, và có thể nói nếu như nhắm mắt tôi vẫn gõ được tất cả các ký tự trên phím nhanh hơn cả việc nhìn để gõ chúng. Tuy nhiên, trời phú cho bạn đôi mắt bạn hãy biết hưởng thụ khả năng của nó đừng nên bỏ qua! Và tôi cũng vậy! Được vậy thì chúng ta hãy bắt tay vào việc sử dụng chương trình phpMyAdmin. Đây là chương trình có giao diện đồ hoạ hỗ trợ mọi truy cập trên CSDL MySQL. Nếu như bạn chưa có thì hãy chạy ra các cửa hàng ngoài Bùi Thị Xuân hay Tôn Thất Tùng kiếm ngay một đĩa đi. Không thôi thì download trên các website phpMyAdmin. Chương trình này rất dễ sử dụng. Nếu như bạn cảm thấy khó thì hãy xem phần Phụ Lục của quyển sách, tôi có hướng dẫn chi tiết trong đó.



phpMyAdmin

phpMyAdmin - Microsoft Internet Explorer

File Edit View Favorites Tools Help

Back Forward Stop Refresh Home Search Favorites History Mail Print Edit

Address http://192.168.1.1/nginx/admin/phpMyAdmin/index.php3

Home  
database\_name  
mysql  
columns\_priv  
func  
host  
tables\_priv  
user  
test  
companies  
contacts  
mytable  
mytable2  
table1  
test10  
test3  
test4  
test5  
test6  
test8  
topics

## Database test - table companies

Select fields (at least one):

company\_id  
company\_name  
company\_address

• Add search conditions (body of the "where" clause):  
[ ] [Documentation]

• Do a "query by example" (wildcard: "%")

Field	Type	Value
company_id	int	
company_name	string	
company_address	string	

Start

10:40 AM

# Tóm tắt

Trong chương này bạn đã tìm hiểu tất cả các lệnh cần thiết để tạo và thao tác với CSDL trong môi trường MySQL. Bạn đã biết mục đích của index. Biết các kiểu dữ liệu và mục đích sử dụng theo tùy trường hợp. Ngoài ra tôi cũng giới thiệu một chương trình quản lý CSDL có giao diện đồ họa phpMyAdmin giúp bạn thao tác thủ công trên CSDL một cách dễ dàng. Trong phần III và IV bạn sẽ tìm hiểu kỹ hơn và thực hiện những ứng dụng quan trọng của CSDL trong chương trình. **(Còn tiếp)**

Lưu ý:

Bạn có thể tìm thấy các CD thiết kế web PHP tại các cửa hàng CD (Tôn Thất Tùng, tp.HCM chẳng hạn). Trên CD có trình cài đặt Apache, PHP, MySQL thông dụng là FOX. Ngoài ra còn có các Website PHP mẫu.

**CÁC LỆNH Ở BÀI HỌC TRÊN TỐT NHẤT BẠN NÊN GÕ LẠI, KHÔNG NÊN COPY!**

[tongphuockhai@mail15.com](mailto:tongphuockhai@mail15.com)

*- Kiến thức là kho báu không phải của riêng ai. Vì vậy bạn đừng ngần ngại khi chia sẻ cho mọi người.*



# LẬP TRÌNH WEB ĐỘNG VỚI **PHP / MySQL**

- ❖ GUESTBOOK
- ❖ CATALOG
- ❖ FORUM
- ❖ SHOPPING CART

## PHẦN 3

**Tổng Phước Khải (tổng hợp & biên dịch)**

# **BIẾN (variables)**

## **và các phép xử lý trên biến**

# **PHP**

- 1- Biến và cách sử dụng Biến**
- 2- Xử lý dữ liệu từ FORM**
- 3- Tìm hiểu sâu hơn về Biến**

Bạn đọc thân mến,

Vừa qua tôi đã soạn xong phần 1 và phần 2 của giáo trình tự học PHP/MySQL. Tôi đã nhận được email của những bạn quan tâm, chờ đợi phần 3 của giáo trình này. Đáng lẽ phần 3 đã cho ra đời sớm nhưng vì bận rộn quá nhiều công việc (hiện tôi đang phụ trách và có rất nhiều công việc trong nhóm Hanosoft - software Hán Nôm) nên việc biên soạn sách tự học này ít nhiều bị trì hoãn.

Do hoàn cảnh trên, chắc chắn công việc biên soạn này không tránh được những sai sót. Nhưng dù sao đi nữa, biết được các bạn có thể áp dụng giáo trình này vào thực tế thì tôi phần nào cũng lấy đó làm sự khích lệ cho riêng mình.

Đúng lý ra phần 3 này là dành trọn cho việc nói về ngôn ngữ SQL nhưng phần 2 đã bàn về CSDL rồi, nếu phần này nếu cũng bàn về nó thì có vẻ hơi nhàm phải không các bạn? Với ý nghĩ này tôi đã dành trọn phần 3 để nói về biến trong PHP.

Hẳn nhiên tôi biết một số bạn mong mỗi những gì trong đây có thể áp dụng liền thì đỡ chán hơn. Nhưng theo tôi nghĩ trước tiên hết bạn cần phải nắm rõ mọi ngóc ngách của PHP và MySQL thì mới có thể thiết kế được những chương trình đạt tiêu chuẩn. Do vậy mong các bạn hãy kiên nhẫn khi đọc những chương hướng dẫn suông như thế này! Đừng nản lòng và nên ghi nhớ đây là cội rễ cho các ứng dụng thực tế của các bạn.

Để bắt đầu chương này ít nhất bạn cũng phải có chút đỉnh kiến thức về **Cơ Sở Lập Trình**. Tôi nghĩ nếu bạn đã học qua một khoá lập trình căn bản thì bạn có thể hiểu được. Nếu không, đòi hỏi bạn cần phải động não hoặc tìm tòi hơi nhiều. Nào, chúng ta bắt đầu đi thôi!

PHP xử lý các biến rất linh động. Nó có thể nhận biết được kiểu của biến và làm cho cú pháp câu lệnh đơn giản hơn. Ai đã từng lập trình với C, Java hoặc Perl sẽ cảm thấy rất dễ dàng khi sử dụng PHP. Tuy nhiên việc dễ dãi này cũng gây ra một số trở ngại nhất định.

**Tất cả những biến khai báo trong PHP đều được bắt đầu với dấu đô la (\$).** Dù cho biến của bạn kiểu chuỗi, nguyên hay thập phân hoặc thậm chí là mảng thì chúng không có gì khác biệt nhau. PHP chỉ theo dõi dữ liệu chứa trong biến thay đổi như thế nào thôi.

Nói chung, khi làm việc với PHP bạn sẽ quan tâm đến 3 vị trí khác nhau của biến đó là: (1) *khai báo ngay trong mã lệnh PHP*, (2) *chuyển tiếp từ một trang HTML* hoặc (3) *là biến sẵn có trong của hệ thống PHP*.

Chúng ta sẽ tìm hiểu về từng loại trên ở phần tiếp theo. Nên lưu ý rằng biến cũng có thể được chuyển tiếp từ các nơi khác như từ các URL hoặc từ các SESSION.

# Gán biến trong một Script

Bạn không cần phải khai báo EXPLICIT cho biến như trong một số ngôn ngữ khác. Chỉ cần khai báo tên biến là nó sẽ sẵn sàng làm việc. Bạn hãy xét các ví dụ sau để hiểu cách khai báo biến trong PHP **uyên chuyển** như thế nào:

```
$a = "Toi thich hoc PHP"; //day la bien chuoi  
$b = 4; //day la bien so  
$c = 4.837; //day la bien so thuc  
$d = "2"; //day cung la bien chuoi
```

Để ý rằng dấu = là dấu dùng để **gán**. Còn khi thực hiện **phép so sánh** bằng thì bạn dùng hai dấu bằng (= =). Ví dụ: IF (\$x==1)

PHP rất thông minh trong việc biến đổi kiểu. Ví dụ, bạn thực hiện phép cộng một số nguyên với một chuỗi chứa ký tự số (trong ví dụ trên là \$b và \$d).

```
$a = "Toi thich hoc PHP"; //day la bien chuoi  
$b = 4; //day la bien so  
$c = 4.837; //day la bien so thuc  
$d = "2"; //day cung la bien chuoi
```

```
$e = $b + $d;  
echo $e;
```

PHP sẽ nhận ra rằng bạn muốn xem chuỗi trong `$d` (chuỗi "2") như là một số nguyên. Thế là nó sẽ hoán chuyển sang trị nguyên và thực hiện phép toán cộng cho ra kết quả là `$e = 6`. Ngoài ra, PHP còn có thể hiểu được các chuỗi vừa số vừa chữ như ví dụ sau:

```
$a = 2;  
$b = "2 con heo con";  
$c = $a + $b;
```

Kết quả cho ra là `$c = 4`. Nếu một số nguyên hay thập phân đứng ở vị trí đầu một chuỗi thì PHP có thể hiểu được như ví dụ trên. Tương tự, PHP thực hiện tương tự đối với các kiểu số khác nhau:

```
$f = 2; // $f là một số nguyên  
$g = 1.444; // $g là một số thực  
$f = $f + $g; // $f tự biến đổi thành số thực
```

Việc xử lý này thật là hay, nhưng nó có thể dẫn đến một số rắc rối đó là sẽ có những lúc bạn không biết ở thời điểm nào bạn sẽ làm việc với kiểu của biến là kiểu gì. Tôi sẽ trình bày trong phần Kiểm Tra Biến.

## Qui định về chuỗi

Trong các ví dụ trên, bạn thấy tất cả các chuỗi đều được bao trong dấu nháy đôi. Có hai cách khác để bạn thể hiện một cho PHP hiểu đó là chuỗi.

Trong một chuỗi mà bạn đã bao lại bằng cặp nháy đôi "...", xong bạn chèn một biến vào giữa, thì PHP vẫn hiểu được biến đó. Ví dụ:

```
$my_name = "Jay";  
$phrase = "Hello, my name is, $my_name";  
echo $phrase;
```

Kết quả cho ra là: Hello, my name is, Jay. Thật khác thường phải không các bạn?! (Đáng lẽ ra dấu nháy " thứ hai phải sau chữ *is* rồi đặt một dấu cộng chuỗi với biến *\$my\_name*)

Trong trường hợp sau đây, tôi muốn xuất ra một chuỗi: *Tôi đăng ký hosting hết \$20* thì phải làm sao? Bởi vì trong chuỗi này có chứa \$, điều này sẽ làm cho PHP hiểu đó là một biến mới. Chúng ta xem cách giải quyết như sau:

Nếu như trong chuỗi bạn muốn có chứa các ký tự đặc biệt như: dấu nháy đôi "", dấu slash \, dấu đô la \$, bạn phải sử dụng đến ký tự chuyển (gọi là **dấu escape**) đó là dấu slash (\).

*Tôi quen đọc dấu / là "dấu suyệt trái" và \ là "dấu suyệt phải".*

Giả sử, để xuất ra màn hình một dòng chữ: *<form action="mypage.php" method="get">*, như bạn thấy trong đó chứa tới 4 dấu nháy đôi - thuộc dạng ký tự đặc biệt. Ta phải sử dụng tới 4 dấu suyệt phải như sau:

```
echo "<form action=\"mypage.php\" method=\"get\">";
```

Thì đến khi chạy chương trình mới mong cho ra kết quả như mong muốn.

### ***Tác dụng của dấu nháy đơn đối với PHP:***

Bạn sẽ thấy dấu nháy đơn trong PHP có tác dụng hơn dấu nháy đôi như thế nào! Nếu chuỗi của bạn có chứa các biến (bắt đầu bằng \$), bạn bao lại bằng dấu nháy đơn thì biến đó sẽ bị biến thành chuỗi luôn, chớ không được hiểu là một biến như cách bao bằng dấu nháy đôi:



```
$my_name = "Jay";  
echo 'Hello, my name is, $my_name';
```

Kết quả cho ra là *Hello, my name is, \$my\_name* chứ không phải Hello, my name is Jay.

Cuối cùng, trong PHP4 bạn có thể sử dụng dấu **Here Documents**. Đây là một loại ký hiệu tương tự hai loại nháy đơn và nháy đôi. Trong một số trường hợp khi sử dụng nó bạn sẽ cảm thấy rất tiện lợi. **Here Docs** xác định giới hạn ở đầu chuỗi với 3 dấu nhỏ hơn <<< và ký hiệu nhận dạng (trong sách này tôi sử dụng ký hiệu nhận dạng **EOQ**) Chuỗi được kết thúc cũng với ký hiệu nhận dạng như vậy và kèm theo là dấu chấm phẩy (;). Sau đây là ví dụ chuỗi *Toi thích học PHP* được gán cho biến **\$mystring** được xác định bằng cách sử dụng **Here Doc**.

```
$my_string = <<<EOQ  
Toi thích học PHP.  
EOQ;
```

Sử dụng **Here Doc**, các biến sẽ chỉ ảnh hưởng trong chuỗi cho nên khi thể hiện dấu nháy đôi trong chuỗi thì không cần sử dụng dấu escape.

```
$element = <<<EOQ
<textarea name="$name" cols="$cols" rows="$rows"
wrap="$wrap">$value</textarea>
EOQ;
```

Như ví dụ trên các bạn thấy không cần phải hao phí nhiều dấu suyệt (\), chúng ta vẫn có thể có được một chuỗi chứa các ký hiệu dạng biến không có tầm ảnh hưởng ra bên ngoài.

Các phần tử mảng sử dụng khoá liên hợp (bạn sẽ tìm hiểu ở phần tiếp theo) không thể sử dụng Here Doc được. Ví dụ sau đây sẽ xuất hiện lỗi:

```
$array = array ("fname"=>"jay", "lname"=>"greenspan");
$str = <<<EOQ
print my string $array["fname"]
EOQ;
```

# Mảng (array) trong PHP

Mảng là một dạng của biến trong đó có chứa nhiều giá trị. Ví dụ một dạng đơn giản của mảng là tháng:

```
$thang = array("Gieng", "Hai", "Ba", "Bon", "Nam",  
"Sau", "Bay", "Tam", "Chin", "Muoi", "Muoi Mot", "Muoi Hai");
```

Mảng này có chứa 12 phần tử, và bạn có thể định vị chúng bằng thứ tự ở trong mảng, bắt đầu bằng vị trí 0. Do đó lệnh `echo $thang[0]` sẽ cho ra là **Gieng** và `echo $thang[11]` sẽ cho ra **Muoi Hai**. Để truy xuất được tất cả các phần tử trong mảng, bạn có thể tính ra chiều dài của mảng và thực hiện vòng lặp:

```
for ($i=0; $i<count($months); $i++)  
{  
    echo $thang[$i] . "<br>\n" ;  
}
```

Chi tiết về vòng lặp sẽ được trình bày ở các phần sau. Bạn có thể gán giá trị vào mảng với một phép toán đơn giản như sau:

```
$dogs = array();  
$dogs[0] = "kiki";  
$dogs[1] = "lulu";
```

Nếu bạn không xác định chỉ số bên trong ngoặc vuông thì giá trị sẽ được gán cho phần tử cuối mảng. Trong ví dụ sau "nana" sẽ được gán vào \$dogs[2]:

```
$dogs[] = "nana";
```

## Mảng liên hợp

Cũng giống như các ngôn ngữ khác, PHP tận dụng khả năng của **mảng liên hợp** (associative array). Có thể bạn cảm thấy mới mẻ với khái niệm này. Để tôi nói sơ qua một chút: Mỗi phần tử trong mảng liên hợp mang **khóa(key)** riêng. Các phần tử của mảng sẽ được truy cập thông qua khóa. Điều này giống như cách thức truy xuất trong các query khi làm việc với Database. Trong ví dụ sau, bạn sẽ thấy các phần tử `first_name`, `last_name`, `e-mail` sử dụng các **key**:

```
$person = array (  
    "first_name" => "Jay",
```

```
"last_name" => "Greenspan",  
"e-mail" => "jgreen_1@yahoo.com"  
);
```

Nếu như bạn muốn thêm phần tử vào mảng, bạn có thể gán tiếp một giá trị khác. Dòng lệnh sau sẽ thêm một số nguyên vào trong mảng, do đó mảng này sẽ chứa tất cả 4 phần tử.

```
$person["age"] = 32;
```

Nếu bạn muốn truy cập cả **khoá** và **giá trị** của một mảng liên hợp, bạn sẽ dùng **list() = each()** như sau:

```
while (list($key, $value) = each($person))  
{  
echo "<b>key :</b> $key, value = $value <br>\n";  
}
```

Các chương sau này tôi sẽ nói kỹ về `list() = each()` một cách chi tiết hơn. Trên cơ bản `each()` truy xuất được cả khoá và giá trị của phần tử trong mảng. `List()` giữ các giá trị

và gán vào **\$key** và **\$value**. Tiến trình này tiếp tục cho đến khi mỗi phần tử trong mảng được truy cập. Nếu bạn muốn duyệt qua hết mảng bạn cần phải sử dụng `reset($person)`.

Nếu bạn chỉ muốn sử dụng giá trị của phần tử trong mảng mà thôi hoặc bạn muốn sử dụng mảng không liên lệp và vẫn muốn sử dụng cấu trúc `list()=each()` bạn phải thực hiện như sau:

```
while (list( , $value) = each($person))  
{  
echo "value = $value <br>\n";  
}
```

Hoặc bạn chỉ muốn truy xuất khoá, bạn sẽ làm như sau:

```
while (list($key) = each($person))  
{  
echo "key = $key <br>\n";  
}
```

*Hãy nhận định về mảng trong PHP như sau:*

*- Tất cả các mảng trong PHP đều là mảng liên hợp. Tại vì sao? Bởi vì những mảng không phải là liên hợp thì PHP cũng sẽ tự động gán cho chúng các key. Ví dụ: \$x= array ("pug", "poodle"), PHP sẽ tự gán cho \$x các khoá là các con số nguyên theo thứ tự bắt đầu từ số 0. Bạn sẽ được tìm hiểu kỹ ở chương 6.*

## Mảng đa chiều

PHP cũng hỗ trợ mảng đa chiều. Mảng đa chiều thường sử dụng nhất đó là mảng hai chiều. Chúng chứa thông tin dựa trên hai khoá. Giả sử, nếu chúng ta chứa thông tin hai người trở lên thì mảng hai chiều sẽ hỗ trợ việc này rất tốt. Chúng ta sẽ xác lập một mảng \$people. Trong mảng \$people lại chứa mảng cho từng cá nhân:

```
$people = array (  
    "khai" => array ("ho_lot" => "tongphuoc", "tuoi" => 30) ,  
    "minh" => array ("ho_lot" => "leanh" , "tuoi" => 52)  
);
```

Ta thấy \$people chứa các thông tin của 2 người, Khai và Minh. Để truy cập một trị trong bất kỳ thông tin của cá nhân nào bạn sẽ phải dùng cả hai khoá. Ví dụ để truy xuất tuổi của Minh bạn sẽ thực hiện lệnh như sau:

```
echo $people["minh"]["tuoi"];
```

Bạn có thể truy cập tất cả các phần tử trong mảng hai chiều bằng cách sử dụng vòng lặp trên cả hai chiều của mảng:

```
while(list($person, $person_array) = each($people))  
{  
    echo "<b>Ban biet gi ve $person</b><br>\n";  
    while(list($person_attribute, $value) = each($person_array))  
    {  
        echo "$person_attribute = $value<br>\n";  
    }  
}
```

## Biến gán từ trình duyệt (web browser)

Quan điểm chung của việc sử dụng PHP cũng như các ngôn ngữ khác là cung cấp khả năng nhập thông tin theo ý muốn của khách. Thông thường các thông tin này được nhập vào thông qua một form HTML. Nhưng cũng có thể chúng xuất phát từ các nguồn khác như: HTML, cookie, session.



## Biến từ Form của HTML

Dạng thông thường nhất để khách có thể nhập thông tin riêng là thông qua một form HTML. Trong phần phụ lục A có trình bày chi tiết về các tạo một form HTML. Nếu bạn chưa biết gì về cách tạo form này thì hãy đọc phần phụ lục. Bạn hãy tạo trang **sign.php** chỉ chứa 100% mã lệnh HTML như sau (có thể đặt là **sign.htm** cũng được):

```
<form action=myspage.php action=post>
<input type=text name=email>
<input type=text name=first_name>
<input type=submit name=submit value=OK>
<input type=submit name=reset value=Cancel> </form>
```

Một khi khách nhấp chuột vào nút SUBMIT (chấp nhận) thì các biến như **\$email**, **\$first\_name**, và **\$submit** sẽ được chuyển giao sang trang action là **myspage.php**. Sau đó, trong trang **myspage.php** bạn sẽ xử lý các biến này tùy thuộc vào mục đích chương trình. Để ý rằng phần lớn các ứng dụng trong sách này đều sử dụng giá trị của nút lệnh **SUBMIT**.

Trong trang **myspage.php** bạn phải viết các lệnh để xử các thao tác của người truy cập. Bạn hãy xem cách xử lý trong trang **myspage.php** mẫu như sau:

```
<?php
if (isset($submit) && $submit=="OK")
{
echo "Cam on ban da gui thong tin cho chung toi.";
} else {
?>      <form action=mypage.php action=post>
        <input type=text name=email>
        <input type=text name=first_name>
        <input type=submit name=submit value=OK>
        <input type=submit name=reset value=Cancel> </form>

        <?php
        }
?>
```

Bạn hãy xem kỹ ví dụ trên, nếu như người truy cập nhập đủ thông tin và nhấn nút **OK** từ trang sign.php (chứa toàn mã lệnh html), thì nó chuyển sang trang **mypage.php** và xuất ra dòng thông báo: **Cam on ban da gui thong tin cho chung toi.** Ngược lại, nếu như nhấn nút **Cancel** thì nó sẽ thực hiện mã lệnh trong lệnh **Else** và sẽ hiển thị form để buộc nhập lại.

**Chú ý:** Bạn hãy xem lại cách thức submit trong ví dụ GuestBook ở tập một. Trong tập 1, nếu bạn không chọn Submit thì chương trình sẽ gọi lại trang **sign.php** là trang chứa Form nhập liệu

bằng lệnh `include`. Còn ở đây không gọi lại trang `sign.php` nữa, bởi vì chúng ta làm theo kiểu khác là gắn Form nhập liệu ngay trong file Action là `mypage.php`.

Các biến cũng có thể được truy xuất thông qua mảng `$HTTP_POST_VARS` hoặc `$HTTP_GET_VARS`, dựa vào method sử dụng trong form của bạn. Việc này rất thuận tiện, nếu các biến từ các forms có thể mang cùng tên với biến trong script của bạn, hoặc nếu bạn có các biến chưa định nghĩa được chuyển giao thì bạn sẽ tìm được ở đó.

Bạn có thể truy cập bất kỳ phần tử riêng biệt nào như đã làm trong mảng liên hợp (`$HTTP_POST_VARS["e-mail"]`). Hoặc bạn có thể tạo vòng lặp duyệt qua tất cả các phần tử của mảng:

```
while (list($key, $value) = each($HTTP_POST_VARS))  
{  
    echo "variable = $key value = $value <br>";  
}
```

# Truyền mảng

Có những trường hợp khi việc chuyển giao biến không thể thực hiện được. Ví dụ như khi bạn chọn cả hai giá trị cho cùng một biến. Việc này thường xảy ra khi làm việc với form có chứa listbox và có thể là bạn sẽ giữ phím Ctrl để chọn phần tử thứ 2 trong list. Ta giải quyết bằng cách sử dụng phép truyền mảng.

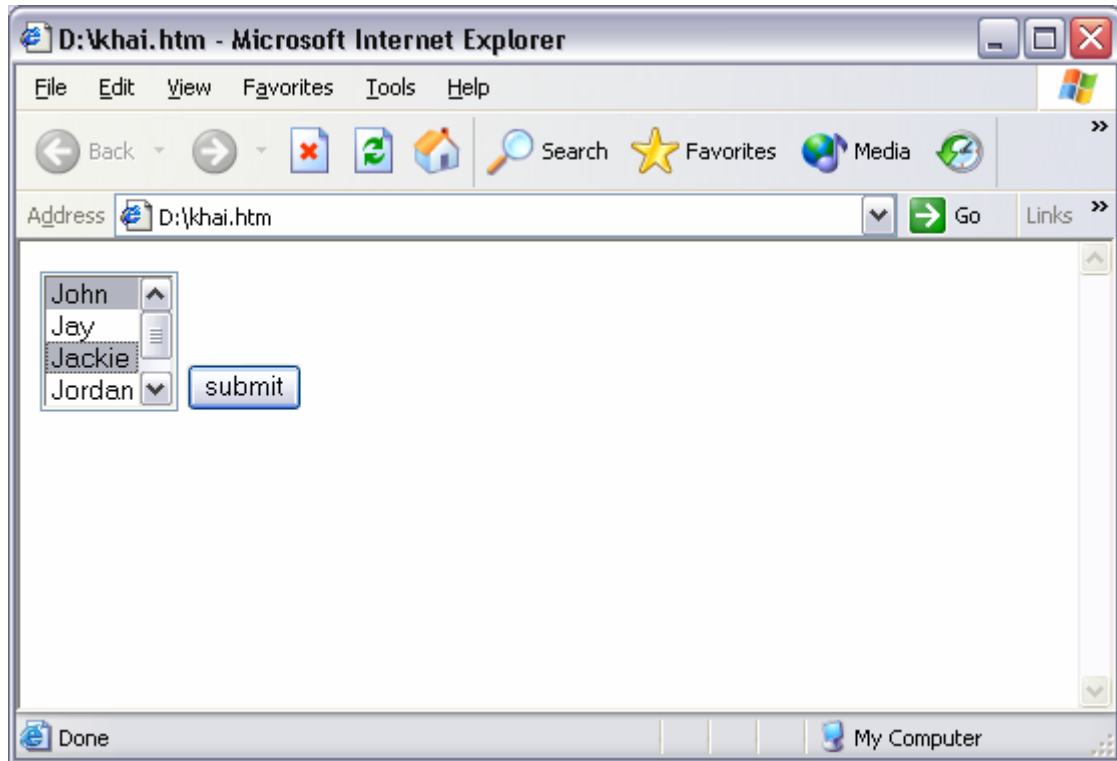
Các lệnh được sử dụng như sau:

```
<form action ="mypage.php" method="post">
<select name="j_names[]" size=4 multiple>
<option value="2">John
<option value="3">Jay
<option value="4">Jackie
<option value="5">Jordan
<option value="6">Julia
</select>
<input type="submit" value="submit">
</form>
```

Để ý rằng trong trong lệnh câu lệnh `select name`, tôi sử dụng dấu ngoặc vuông để bảo PHP biết rằng đây là một mảng. Nếu không sử dụng dấu `[ ]` thì sẽ có tới 2 giá trị gán cho cùng một biến.

Một khi được SUBMIT bạn có thể truy cập mảng như sau:

```
if (is_array($j_names))
{
echo "<b>the select values are:<br> <br>";
while(list($key, $value) = each($j_names))
{
echo $value . "<br>\n";
}
}
```



Việc truyền mảng rất thông dụng khi bạn Submit Form có một loạt các checkbox (tên các mặt hàng chẳng hạn). Khách truy cập có thể sẽ nhấp chuột vào nhiều checkbox hoặc không có checkbox nào. Trong chương 8, có ví dụ cho phép người quản trị có thể sử dụng checkbox để chọn và xoá các phần tử.

Mảng được chuyển giao từ form có thể có khoá liên hợp, ngay cả đối với mảng nhiều chiều. Tên của phần tử form thường có tên là `name = "array_name[element_name]"`. Hoặc đối với mảng nhiều chiều là `array_name[element_name] [subelement_name]"`.

## Cookies

Cookie là những **file nhỏ** chứa một số các thông tin truy cập Web. Các cookie do Webserver phát sinh, lưu giữ lại, sau đó sẽ được đọc ở những lần truy cập về sau.

Cookie đơn thuần chỉ là *thông tin ghi nhận lại những động tác truy cập web của khách*. Khi tồn tại trên đĩa cứng, cookie trở thành các thỉnh cầu của giao thức HTTP, được gửi đến Webserver.

Để có thể phát sinh một cookie bạn cần phải sử dụng hàm `setcookie()` như sau:

```
setcookie(name, value, time_to_expire, path, domain, security  
setting);
```

Chúng ta sẽ tìm hiểu chi tiết về cookie ở chương 6. Còn bây giờ bạn tìm hiểu sơ lược các chức năng thông qua một ví dụ:

```
setcookie("my_cookie",  
"my_id",time()+ (60*60*24*30) , "/", ".mydomain.com", 0)
```

Lệnh trên sẽ phát sinh một cookie với các chức năng sau:

- Chứa một biến tên là my\_cookie
- Giá trị của mycookie my\_id
- Cookie tồn tại trong vòng 30 ngày kể từ ngày nó phát sinh (`time() + (30*24*60*60)` ngày giờ hiện tại + 30 ngày được quy ra giây).
- Cookie có tác dụng đến tất cả các trang trong domain. Bạn có thể hạn chế lại bằng các chỉ ra đường dẫn đến một số trang nào đó trong domain.
- Nó sẽ hiện diện trong tất cả các website có địa chỉ <http://mydomain.com>
- Không có xác lập đặc biệt nào về bảo mật.

Một khi cookie được xác lập, các biến phát sinh từ cookie có tác dụng giống như biến phát sinh từ form mà chúng ta đã bàn trước đây. Chúng sẽ hiện diện với chức năng là biến global.



Sau khi script PHP xác lập cookie, các script khác trong domain có thể truy cập cookie một cách tự động.

Nếu như bạn muốn cẩn thận hơn để `$mycookie` không xung đột với một biến nào khác cũng có tên `$mycookie`, bạn có thể truy xuất nó thông qua mảng `HTTP_COOKIE_VARS` và sử dụng lệnh: `HTTP_COOKIE_VARS["mycookie"]`.

Bạn có thể xác lập cookie cung cấp khả năng truy xuất như là một mảng:

```
setcookie("mycookie[first]","dddd",time()+2592000,"/","192.168.1.1", 0);
```

```
setcookie("mycookie[second]","my_second_id",time()+2592000,"/","192.168.1.1", 0);
```

Cả hai biến trên đều có thể truy cập đến như là một mảng liên hợp.

## Sessions

PHP4 cũng giống như ASP và ColdFusion đều có hỗ trợ session, việc này giúp ích rất nhiều cho việc truy cập web. Vậy session là gì?

Đơn giản nó chỉ là **một cách thức để duy trì và truyền biến** trong khi chuyển tiếp giữa các trang web. Chương trình của bạn khai báo một session được bắt đầu với hàm `start_session()`. PHP đăng ký một SessionID duy nhất, và thường thì ID này được gửi đến user thông qua một cookie. PHP sau đó tạo một tập tin trên server để theo dõi sự thay đổi của biến. Tập tin này có tên giống như tên của SessionID.

Một khi session được tạo, bạn có thể đăng ký bất kỳ số lượng biến. Các giá trị của những biến này được lưu giữ trong tập tin trên server. Cũng như sự tồn tại của cookie, các biến trong session sẽ hiện diện trên bất kỳ trang nào được truy cập đến trong phạm vi một domain. Việc xác lập này rất thuận tiện hơn là chuyển tiếp các biến từ trang này sang trang khác thông qua các phần tử ẩn trong form hay cookie.

Session nói chung là khá đơn giản. Hãy xem script sau sẽ đăng ký một biến session tên là `$my_var`, và sẽ gán cho nó một giá trị là `"hello world"`.

```
<?
session_start();
session_register("my_var");
$my_var = "hello world";
?>
```

Trên những trang tiếp theo biến `$my_var` sẽ hiện diện, nhưng chỉ sau khi bạn chạy hàm `session_start()`. Hàm này bảo PHP tìm kiếm một session xem có tồn tại hay không, rồi làm cho các biến session trở thành global. Nó có thể sử dụng câu lệnh IF để làm cho các biến session hoàn toàn có thể truy cập được. Hãy xem xét ví dụ sau:

```
<?php
session_start();
session_register("your_name");
//check to see if $your name contains anything
if(!empty($your_name))
{
echo "I already know your name, $your_name";
}
//this portion will probaby run the first time to
//this page.
elseif(empty($your_name) && !isset($submit))
{
echo "<form name=myform method=post action=$PHP_SELF>
<input type=text name=first_name> first name<br>
<input type=text name=last_name> last name<br>
<input type=submit name=submit value=submit>
</form>";
//if the form has been submitted, this portion will
```

```
//run and make an assignment to $your_name.  
} elseif (isset($submit) && empty($your_name))  
{  
$your_name = $first_name . " " . $last_name;  
echo "Thank you, $your_name";  
}
```

Sau khi chạy chương trình này, chọn **refresh** trên trình duyệt. Bạn sẽ thấy script sẽ nhớ được rằng bạn là ai.

Các hàm `setcookie()` và `session_start()` nên ở vị trí gần đầu tập tin. Nếu bạn thử chuyển đến trình duyệt trước để xác lập một cookie bạn sẽ nhận được một thông báo lỗi.

# Biến sẵn có

Có rất nhiều biến sẵn có của PHP và Server. Bạn có thể liệt kê một danh sách đầy đủ bằng cách sử dụng lệnh `phpinfo()` để xem. Bạn hãy tạo một file php và cho chạy thử xem:

```
<?php  
phpinfo() ;  
?>
```

Bạn có thể sử dụng các biến này bằng nhiều cách thức khác nhau. Tôi sẽ trình bày một sau ngay sau đây, và sẽ chỉ ra bạn nên dùng vào trường hợp nào. Một số biến đến từ PHP engine, một số khác bắt nguồn từ Webserver.

# Biến sẵn có của PHP

## PHP\_SELF

Biến này nhận giá trị là địa chỉ hiện tại của tập tin .php đang được duyệt. Địa chỉ này sẽ là địa chỉ đầy đủ từ gốc (bắt đầu từ http://) . Bạn sẽ sử dụng nó khi muốn truy cập lại chính trang web đang thi thành.

Xét ví dụ sau, đây là một form tương tự như form sign.php mà các bạn đã có dịp xét qua. Nếu khách thực hiện thao tác khác với submit thì chính form này sẽ được thi hành lại:

```
<?
if(isset($submit))
{
//Xuat ra thông báo tại đây
echo "Cam on ban da submit";
} else {
?>
<form name=myform method=post action=<?=$PHP_SELF?>>
<input type=text name=first_name> first name<br>
<input type=text name=last_name> last name<br>
<input type=submit name=submit value=submit>
```

```
</form>  
<?  
}  
?>
```

## HTTP\_POST\_VARS

Đây là một mảng chứa tất cả các biến được chuyển tiếp thông qua POST method từ một form. Bạn có thể truy cập từng biến riêng rẽ như là một phần tử của mảng liên hợp (ví dụ: `$PHP_POST_VARS["myname"]`).

## HTTP\_GET\_VARS

Đây là một mảng chứa tất cả các biến được chuyển tiếp thông qua GET method. Bạn có thể truy cập từng biến riêng rẽ như là một phần tử của mảng liên hợp (ví dụ: `$PHP_GET_VARS["myname"]`).

## HTTP\_COOKIE\_VARS

Tất cả các cookie chuyển đến trình duyệt đều có thể được truy xuất trong mảng liên hợp này. Nó bao gồm cả session cookie. Nếu bạn còn thắc mắc cookie sẽ thi hành như thế nào thì hãy xem hàm `phpinfo()` để biết được trình duyệt của bạn đang chuyển đến server những gì.

# BIẾN CỦA APACHE

Apache có sẵn rất nhiều biến. Tôi không trình bày đầy đủ tất cả các biến ra đây. Các biến bạn sử dụng, chúng tùy thuộc vào xác lập hiện tại của bạn như thế nào. Sau đây là một số biến mà có lẽ bạn sẽ sử dụng thường xuyên trong chương trình của bạn.

## DOCUMENT\_ROOT

Biến này trả về đường dẫn của Webserver. Biến này được tôi sử dụng trong xuyên suốt quyển sách này. Hãy xét ví dụ sau:

```
include"$DOCUMENT_ROOT/book/functions/charset.php";
```

Bằng cách sử dụng biến DOCUMENT\_ROOT thay vì dùng đường dẫn tuyệt đối, chúng ta có thể di chuyển toàn bộ một thư mục sang một Apache Server khác mà không lo lắng rằng đường dẫn sẽ bị sai lệch trong include path. Nên nhớ rằng nếu như bạn không sử dụng Apache Server thì biến này không sử dụng được. Nếu bạn sử dụng include\_path trong tập tin php.ini,



bạn không cần phải lo lắng phải xác định đường dẫn như thế nào bởi vì PHP sẽ duyệt hết tất cả các thư mục và tìm ra tập tin bạn đã chỉ định.

## HTTP\_USER\_AGENT

Bất kỳ ai đã từng thiết kế Web site đều hiểu rằng tầm quan trọng của việc nhận dạng được trình duyệt của người sử dụng là gì. Một số trình duyệt thì không sử dụng được JavaScript, một số khác thì đòi hỏi dạng HTML đơn giản. Biến `user_agent` cung cấp cho bạn khả năng uyển chuyển đối với từng trình duyệt khác nhau. Một `user_agent` chuẩn có dạng như thế này:

```
Mozilla/4.0 (compatible; MSIE 5.01; Windows 98)
```

Nếu bạn phân tích chuỗi này ra bạn sẽ biết được những gì bạn cần tìm. Có thể bạn chỉ thích hàm `get_browser()` của PHP. Về lý thuyết mà nói, hàm này định nghĩa khả năng cho phép của trình duyệt của user đang sử dụng. Cho nên bạn có thể biết được là chương trình của bạn đang phục vụ tốt hay không. Các sách PHP có những hướng dẫn về cách cài đặt và sử dụng `get_browser()`, nhưng tôi khuyên bạn không nên sử dụng nó. Bởi vì sử dụng `get_browser` bạn sẽ được bảo rằng IE 5 dùng cho PC và Netscape 4.01 dùng cho Mac có hỗ trợ CSS (cascading stylesheets) và JavaScript. Nhưng bất kỳ người sử dụng nào cũng biết rằng: viết lệnh DHTML

để chạy trên cả hai môi trường trình duyệt này là một công việc phức tạp. Thông tin bạn nhận được từ `get_browser()` có thể dẫn đến những tính năng giả trong bảo mật. Cách tốt nhất là bạn sử dụng `HTTP_USER_AGENT` và thực hiện quyết định của mình dựa trên trình duyệt hoặc platform xác định nào đó.

## **REMOTE\_ADDR**

Dùng để lấy địa chỉ IP của user. Tuy nhiên có những user am hiểu chuyện này và có thể họ thay đổi IP của máy mình. Cho nên không lấy gì để đảm bảo rằng: một địa chỉ IP chắc chắn là của một user nào đó. Bạn sử dụng biến này để theo dõi sự truy nhập của một user nhưng nó chỉ mang tính tương đối thôi.

## **REQUEST\_URI**

Biến này cũng giống như biến `PHP_SELF`. Ngoài ra nó còn chứa thêm tham số trong địa chỉ truy vấn. Nếu bạn truy cập vào địa chỉ:

<http://www.mydomain.com/info/products/index.php?id=6>

Thì biến `REQUEST_URI` của bạn có giá trị là: `info/products/index.php?id=6`

## **SCRIPT\_FILENAME**

Biến này chứa toàn bộ đường dẫn của tập tin.

# Kiểm tra biến

Ở trên chúng ta đã nói nhiều về Biến. Như các bạn biết đó, tên của một biến không quan trọng bằng giá trị mà nó chứa trong đó. Như tôi đã nói Biến trong PHP rất uyển chuyển. Điều này phát sinh sự bất lợi là bạn sẽ không biết ở tại một thời điểm nào đó thì biến này sẽ mang giá trị gì. Do đó bạn cần phải thực hiện thao tác kiểm tra biến.

## isset( )

Hàm này thực hiện việc kiểm tra biến có chứa giá trị hay không. Nó sẽ trả về giá trị TRUE hoặc FALSE. Nếu biến chưa được xác lập thì `isset()` sẽ là false.

Bạn hãy xem xét ví dụ sau, nó thi hành một query MySQL. Bạn đã biết rằng một field trong database có thể chứa trị null hoặc chuỗi rỗng. Với việc sử dụng hàm `isset()` bạn sẽ kiểm tra và phân biệt được hai giá trị trên. Trong đoạn lệnh PHP bên dưới. Trong đó biến `$query` là một phát biểu SELECT lấy dữ liệu submit từ form của user.

```
$result = mysql_query($query) or  
die (mysql_error());  
$number_cols = mysql_num_fields($result);
```

```
echo "<b>query: $query</b><br>\n";  
//layout table header  
echo "<table border = 1>\n";  
echo "<tr align=center>\n";  
for ($i=0; $i<$number_cols; $i++)  
{  
echo "<th>", mysql_field_name($result, $i), "</th>\n";  
}  
echo "</tr>\n";//end table header  
//layout table body  
while ($row = mysql_fetch_row($result))  
{  
echo "<tr align=left>\n";  
for ($i=0; $i<$number_cols; $i++)  
{  
echo "<td>";  
if (!isset($row[$i])) //test for null value  
{echo "NULL";}   
else  
{echo $row[$i];}  
echo "</td>\n";  
}
```

```
}  
echo "</tr>\n";  
}  
echo "</table>";
```

Lưu ý rằng dấu chấm than (!) có nghĩa là phủ định.

Tức là nếu **\$var** có giá trị **null** thì:

**isset(\$var)** cho ra giá trị **False**

**!isset(\$var)** cho ra giá trị **True**

## **empty()**

Hàm **empty()** có vẻ ngược ngạo so với hàm **isset()**. Nó sẽ cho ra trị **True** nếu **\$var** có trị **null**, chuỗi rỗng hoặc số 0. Hàm này thường được sử dụng để kiểm tra xem user có nhập trị vào trong form hay không:

```
if(empty($first_name))  
{  
echo "Ban can phai nhap ten cua minh";  
exit;
```

```
}
```

## **is\_int( )**

Hàm này để kiểm tra biến có phải là số nguyên hay không. Có 2 cú pháp khác cho cùng kết quả như nó là: `is_integer` và `is_long()`. Bạn sử dụng hàm này khi không chắc rằng biến là một trị nguyên hay chuỗi. Ví dụ:

```
$a = "222";  
$b = 22;
```

`is_int($a)` cho ra trị False

`is_int($b)` cho ra trị True

Tương tự bạn sẽ có một loạt hàm kiểm tra kiểu của biến sau đây:

## **is\_double()**

Kiểm tra số kiểu double (dấu phẩy động). Hàm thay thế: `is_float( )` và `is_real( )`.

## **is\_string( )**

Kiểm tra kiểu chuỗi.

## **is\_array( )**

Kiểm tra kiểu mảng.

## **is\_bool( )**

Kiểm tra kiểu boolean (TRUE và FALSE)

## **is\_object( )**

Kiểm tra biến kiểu object. Bạn sẽ tìm hiểu kiểu object trong các phần sau.

## **gettype( )**

Hàm này sẽ cho bạn biết kiểu của biến như: string, double, integer, array, hoặc boolean. Ngoài ra nó có trả về các kiểu như object, class. Bạn sẽ khảo sát kỹ về việc lập trình hướng đối tượng trong các phần sau để biết thêm về object và class.

Lưu ý trị của hàm `gettype()` trả về luôn là một chuỗi: "string", "integer", "double" v.v.  
Bạn hãy xem ví dụ sau:

```
$str = "Day la mot chuoi";  
$type = gettype($str);  
if ($type == "string")  
{  
echo "Dung vay";  
}
```



# Đổi kiểu của biến

Bạn sẽ sử dụng 3 cách để đổi kiểu của biến.

## Phương pháp type casting

Phương pháp này rất đơn giản: Bạn chỉ cần ghi tên kiểu ra, đóng ngoặc đơn lại, rồi đặt trước biến. Tức khắc biến sẽ bị đổi theo kiểu mà bạn muốn.

Cách thức: (kiểu) \$biến

Ví dụ:

```
$a = 1;  
$b = (string)$a; //số 1 sẽ biến thành chuỗi 1  
echo gettype($a), "<br>\n";  
echo gettype($b), "<br>\n";
```

Kết quả cho ra là:

```
integer  
string
```

## Sử dụng hàm `settype()`

Hàm này có 2 đối số. Thứ nhất là tên biến, thứ nhì là kiểu. Ưu điểm của nó là nó có thể cho ra kết quả FALSE nếu như việc hoán đổi không được.

Cách thức: `settype($biến, "kiểu")`

Ví dụ:

```
$a = 1;  
settype($a, "string");
```

## Sử dụng hàm `intval()`, `doubleval()`, và `stringval()`

Phương pháp này thường để bạn áp dụng nhanh trong khi tính toán. Có lẽ nhìn tên hàm bạn cũng biết được chức năng của nó rồi. Hãy xét ví dụ sau:

```
$a = "43" ; // 43 là kiểu chuỗi  
$b = (intval($a) * 2);
```

# Biến của biến

Nghe qua có vẻ lạ lạ, nhưng đây là một "độc chiêu" của PHP. Với cách thức này bạn sẽ lấy giá trị của một biến để hình thành tên của một biến mới.

Cách thức: `$$biến`

Ví dụ:

```
$a = 'khai';  
$$a = 'Chao moi nguoi';
```

Bạn sẽ thấy trong ví dụ trên một biến mới được hình thành đó là `$khai` chứa giá trị là "Chao moi nguoi"

Xét thêm ví dụ sau, trong đó `$tacgia` là một mảng liên hợp.

```
<?  
$tacgia = array ("ho"=>"Tong", "ten"=>"Khai");  
while (list($field,$value) = each($tacgia))
```

```
{  
$field = "bien_{$field}";  
$$field = $value;  
}  
echo $bien_ho, " ", $bien_ten;  
?>
```

Khi chạy chương trình, các biến sau sẽ được tạo `$bien_ho`, `$bien_ten` và ghi ra màn hình: **Tong Khai**

## Tóm tắt

Bạn đã tìm hiểu các biến trong PHP. Bạn thấy PHP xử lý các biến linh hoạt hơn nhiều so với các ngôn ngữ khác. Còn một vấn đề khá quan trọng đối với biến đó là scope bạn cũng sẽ biết kỹ về nó ở trong các phần sau của giáo trình này.

**(Còn tiếp)**