The Hypertext Blender

July ‘99
Eugene Khoo

A long, long time ago, in a place faraway from here, something was created that was by itself a revolutionary concept. This concept would soon herald a new form of interactivity and power on the web! The time is 1993, the place is Seattle, Washington, USA and the concept was code-named Denali.

In case you’re still unsure what I am talking about, here is the scoop. Denali was conceived as a way to deliver web applications that was platform independent and is based on fat server-thin client principles. All you would need on a client end was a web browser. Today, that concept is Active Server Pages and that Seattle Company is none other than Microsoft.

For a while, there is nothing like ASP on UNIX with the exceptions of ChiliSoft’s ChiliASP and most Unix web developers had to contend with CGI and Java as the primary methods to build a web application. Java however was extremely slow , there were also many types of Java engines that were not all necessarily compatible and JavaScript was too lightweight to be of much use. Today, we have something called PHP. PHP was originally known as Personal Home Page Tools (PHP) but it’s known today generally as “PHP Hypertext Pre-processor.”

Before I actually start talking about PHP and what you could do with it, note that this is based on my experience, which probably doesn’t count as expert advice or comments in any way. I’ll start with Microsoft’s Active Server Pages because that was my first experience with a Hypertext processor.

Like all Hypertext processors, ASP compiles and executes all codes and scripts on the server-end before sending the HTTP stream to the client. Thus, all the client sees is HTML. I found this an extremely powerful method to build web applications, as you do not need anything special on the client, which made deployment easy. How easy? It’s as easy as telling everyone you want to roll the application to fire up the browser and go to the URL. No special Java classes to download, no ActiveX applets or Plug-ins required! In addition, ASP came with built-in ODBC support that enable it to access any database that it has the ODBC drivers for. Everyone should know that a REAL website always has a database behind it and this was a key requirement. Of course if you had any experience with Microsoft’s Visual Basic or even Basic, ASP was a snap to learn. Of course, like all new things, documentation was sparse and methods of doing things were not very efficient but heck, it opened a whole new world of doing things! PHP opened similar worlds on non-Microsoft platforms!

I could build a website that allows users to come in, configure their settings, themes (look-and-feel), preferences, interests, etc and generate a fresh HTML page dynamically just for the user when he or she enters my URL the next time. How is this done? A customized site is easy to build. With the following three ingredients: cookies, sessions and databases.

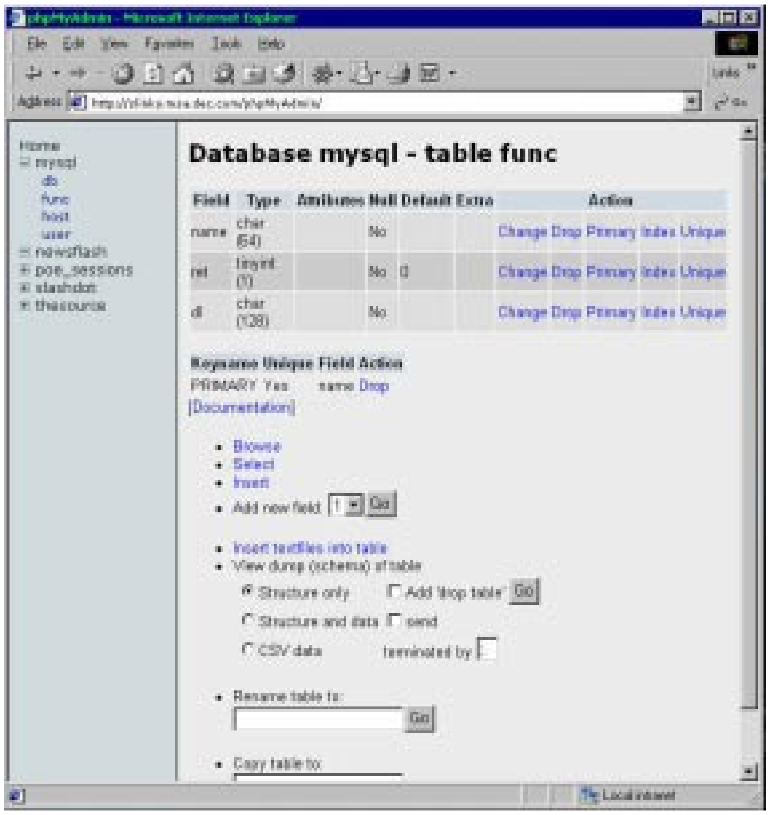
Cookies are small bits of information typically used to store some information about you and is saved on your local hard disk. A session is a variable that is stored on the web server. Sessions can be global, available to all users or more likely, unique to a particular visitor. The web server tracks this by assigning cookies. For example, -if you have cookie A, your session variable is A. Each user may have more than one session variable, however the more session variables are used, the more server memory is required to support it’s usage. In most cases, it is good practice to set session variables to ‘expire’ after a set time, enabling the server to reclaim the memory for reuse. If you don’t know what a database is, I guess you are not really running a website, and hence it can be deduced that PHP will have very little relevance to you.I don’t know what to say if you don’t know what a database is.

In short, this is how it would work. User enters and the system (or web server) checks the session variable titled “username” but it is blank so it checks if the user possesses a cookie from the website. Unfortunately, this is the visitor’s first time so there’s no cookie. Obviously we cannot have a situation like that so the web server then generates a unique identifier and stores it in the session variable and the cookie. I’m assuming the visitor would have cookies enabled on the browser. You might want to put some code in to check if the client accepted the cookies on a production site. You could redirect the user to a page that explains the usage of cookies before they accept it. The visitor then browses the site and it is also expected that you’d be putting some link that tells the user that your site is configurable. So the visitor goes in, configures the fonts, colors, style sheets and even the interests’ section. Upon update, the system would save this in a database with the primary key being the user's unique id (UID). The next time this visitor surfs by, immediately the system knows who it is, loads the settings from the database and uses it throughout the site! CSS (Cascading Style Sheets) is ideal for the look-and-feel of the pages because with one line, you can change the entire look of a page as illustrated by the following code snippet:

```
<?php
// © EK 'Dolphin'
// $userinfo is the result of a db query and is an array with the various row info
// this is still in the header so the body tag has not been written yet
//
if ($userinfo[6] <> 0) {    //checks if user visits is greater then zero
     echo "Welcome back ".$userinfo[2]."</title>"; //has been here before
 } else {
     echo "Welcome and thanks for visiting us!";  /first time visitors
 }
//load stylesheet depending on user
// $userinfo[7] is stylesheet filename
echo "<link rel=stylesheet type=text/css href=/stylesheets/".$userinfo[7].">";
?>
```

After I started to get into Linux and Apache, I have been wondering if I could do all this in Linux. And after a long wait, I'm glad to say that PHP is ASP for Linux. For me, PHP is the best way to build web apps on Linux and there's no turning back. PHP is constantly evolving and I am sure it will get more powerful, faster and flexible (One thing ASP has over PHP is the support for multiple languages: ASP can run Java, JavaScript, PERL, VBScript. This is because there's no such thing as an 'ASP' language.)

However, PHP has two unbeatable attributes. It's multi-platform and it is free. I could develop PHP pages on my notebook or desktop that might be running Windows but the code can be easily ported over to a Linux server! This is a core advantage in the corporate environment when the corporate platform might be Windows but you need to deliver a project on a Linux box. I've tried setting up both the Linux and the Windows installations and they were equally simple. So basically, if you need cross-platform, multi-OS HTTP pre-processor, PHP is the solution. This bodes well for its future as I can see that with the rapidly changing and evolving web technologies, who in this world can keep up! Just imagine keeping up with all the new web technologies such as XML/XSL, DSSSL/SGML. Just learn PHP and you'd be able to produce web apps on both Windows and Linux.



Database mysql - table func

Field	Type	Attributes	Null	Default	Extra	Action
name	char (64)		No			Change Drop Primary Index Unique
ret	tinyint (1)		No	0		Change Drop Primary Index Unique
dl	char (128)		No			Change Drop Primary Index Unique

Keyname	Unique	Field	Action
PRIMARY	Yes	name	Drop

[Documentation]

- Browse
- Select
- Insert
- Add new field: 1 Go
- Insert textfiles into table
- View dump (schema) of table
  - (●) Structure only   ☐ Add 'drop table' Go
  - ( ) Structure and data   ☐ send
  - ( ) CSV data   terminated by
- Rename table to: Go
- Copy table to:

PHP allows you to build dynamic database driven sites like this MySQL Web Administrator Application.

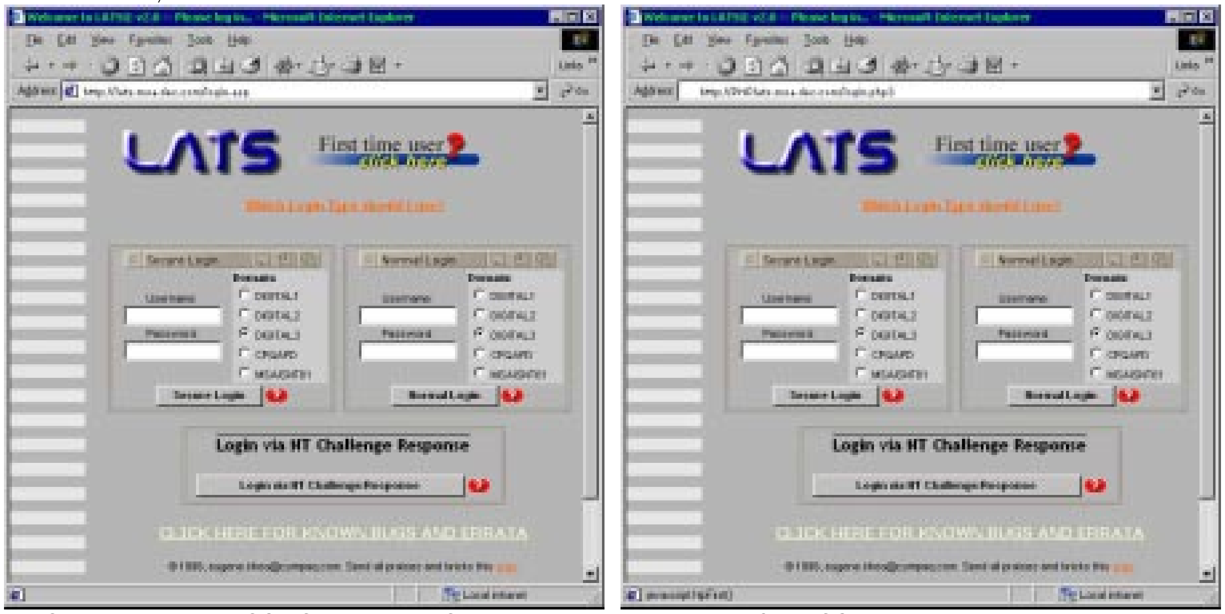
A hypertext blender is just the tool to deliver all kinds of interactivity, dynamism, and personalization to an audience as diverse as the web. As research has shown, a web interface is inherently more usable and user-friendly that a customized client application. Of course poorly designed web pages abound, and if I talked about web interface design here, I will digress. The Human-machine interface will be a subject in the near future since it is bound to come up. So not only can you build a powerful application, you can deliver it across all mediums and all platforms. That means any device that can read HTML. I am talking web browsers on PCs, Macs, Linux boxes as well as cell phones, PDAs, and even WebTV boxes. Of course you'd have to code for the diverse screen sizes but that's inherently do-able, especially if you could query the user on which screen size they would like to view the pages on or have some form of platform detection code.

But all this talk about delivering apps will probably get you wondering about performance. You know that PHP like ASP is executed at the server end so I bet you're thinking that it might be a big hit on the server. Surprise! It is not such a heavy load that a decent modern server cannot handle. From what I have tested, PHP on Win32 runs about as fast as ASP. This says a lot about PHP considering that it was not even the latest version I tested. PHP is as fast if not faster in some aspects and matches ASP feature for feature and then some! So, using that as a relative comparison, I had 600 simultaneous users, five thousand sessions (including entire rows from databases) on both ASP and PHP and they both ran fine. This test was done on a Windows NT Server with dual 200MHz Pentium Pro processors, 256Mb RAM and 26Gb RAID 5 disk. So performance is definitely a non-issue here. I will be elaborating on this in the articles to come.

And if you need any more prove that PHP has come of age, did you know that the following sites are powered by PHP? US Army Publishing Agency, Games Domain, First USA Bank, Audi, Wall Street online and Xoom? Well, you can find a listing on this URL: http://www.php.net/sites.php3

You can see that I'm excited about this technology as it obviously holds promise. But which would you use? For one, I'd vouch for PHP over ASP. ASP does have a gentler learning curve then PHP and may be easier for those with Visual Basic background. However, PHP has got portability, and its features are growing each day. Obviously PHP is not the answer to every problem and neither is ASP, but until Zend[1], PHP is my Hypertext blender of choice!

NT and Linux, ASP and PHP...



NT Server, Microsoft SQL Server and ASP compared to Red Hat 5.2, MySQL and PHP!

[1] Zend is a critical component of PHP 4. More info can be found here: http://www.zend.com