

PHP Programming on the Web

Mechanisms

Syntax

Libraries

Resources

This Lecture

- Discuss PHP in relation to apache
- Explain security and access issues
- Introduce basic syntax
- Programming environment
- Advanced usage
 - Cookies
 - HTTP Auth
 - Databases
- References

What IS PHP?



- Scripting language embedded in HTML
- Interpreted and executed by web server
- Highly functional
- Similar to C, C++, Perl, shell scripts

Using PHP

- Must tag file (like SSI)
- **.php3, .php1**
 - Indicate file has PHP code
 - Code is executed
- **.phps**
 - Indicates PHP source file
 - Source is delivered *as is*
- **index.php1** is valid

Using PHP in HTML

- 3 Syntaxes embedded in body
 1. Quick
 2. FrontPage-friendly
 3. XML-friendly
- Examples
 1. `<? echo "hello\n"; ?>`
 2. `<script language="php">
echo "hello\n";
</script>`
 3. `<?php echo "hello\n"; ?>`

Web Server HTML Parser

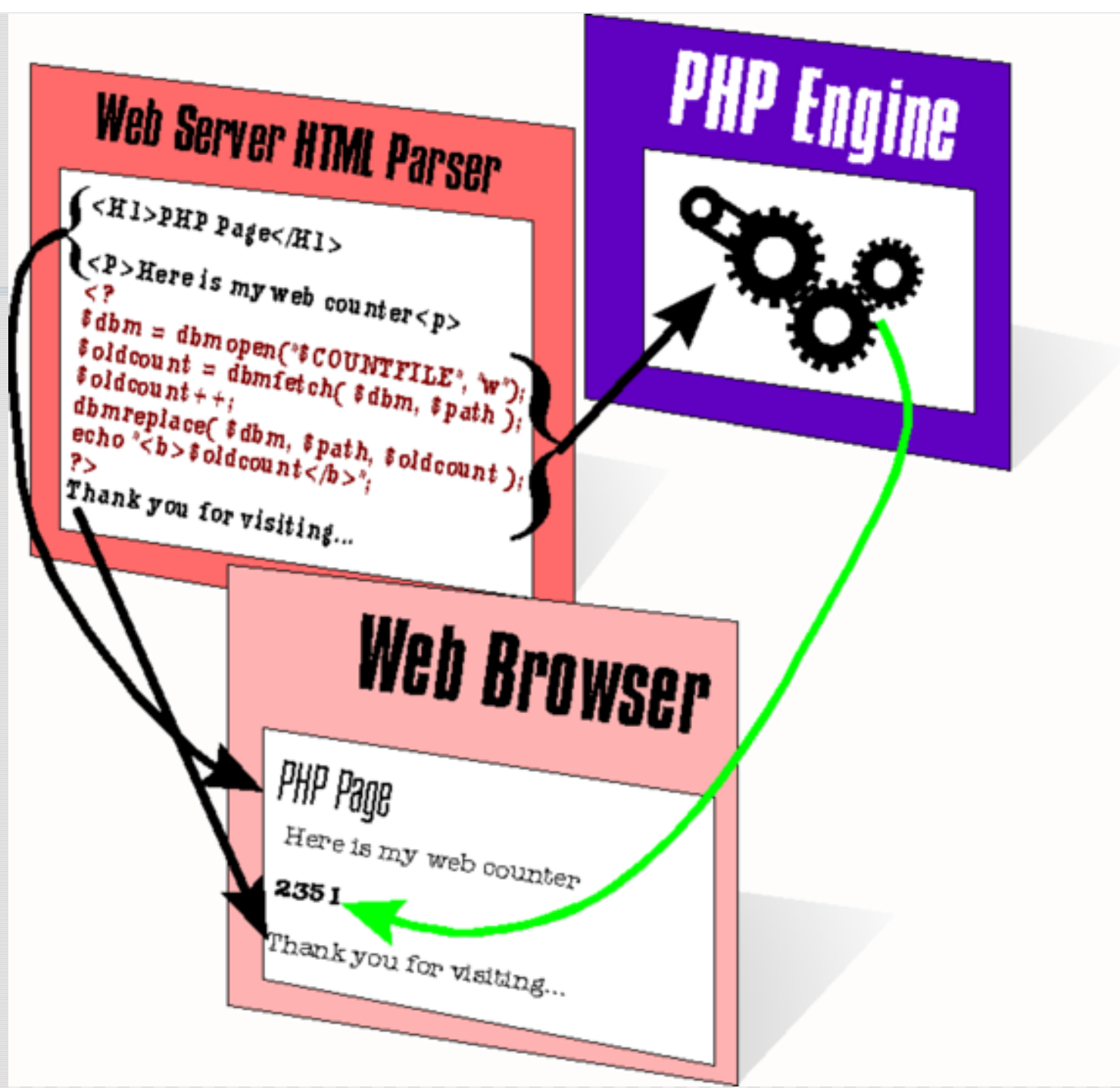
```
<H1>PHP Page</H1>
<P>Here is my web counter<p>
<?
$dbm = dbmopen('$COUNTFILE', 'w');
$oldcount = dbmfetch($dbm, $path);
$oldcount++;
dbmreplace($dbm, $path, $oldcount);
echo '<b>$oldcount</b>';
?>
Thank you for visiting...
```

PHP Engine

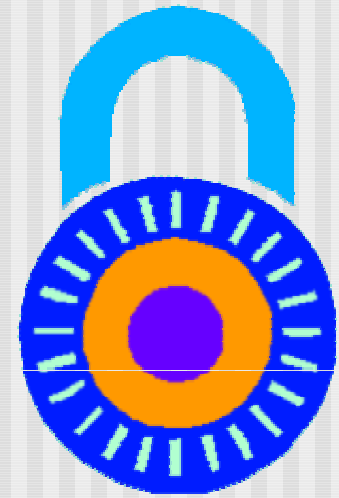


Web Browser

PHP Page
Here is my web counter
2351
Thank you for visiting...



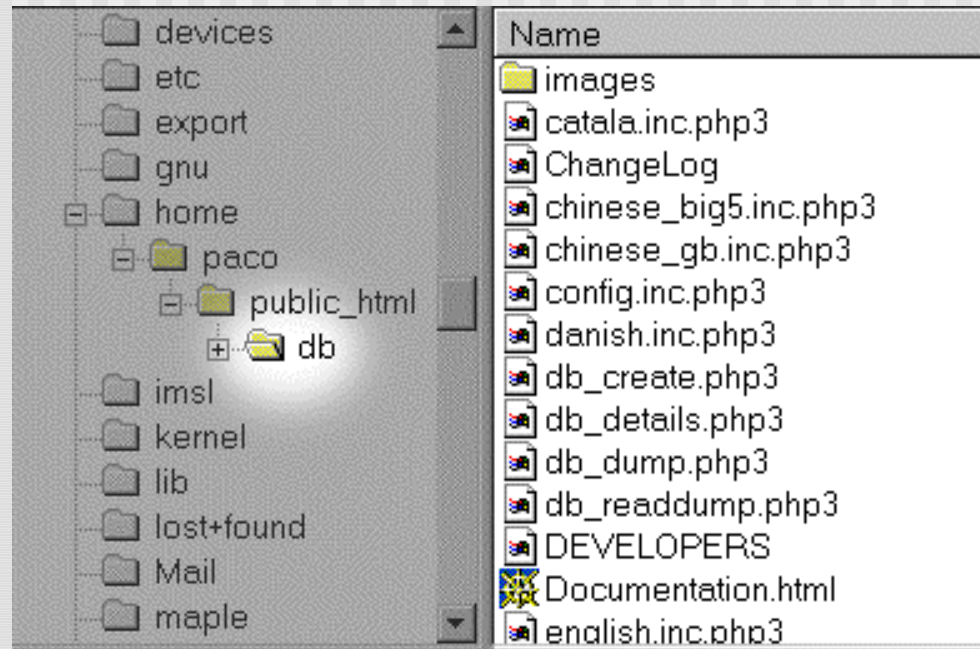
Security



- Scripts execute as `nobody`
- Can only do what "`nobody`" can do
 - Bad for **file access** (requires world-writable files/directories)
 - Ok for **databases** that use independent access methods (e.g. MySQL)
- PHP can be installed as CGI to fix security (c.f. cgiwrap)

File Access

- Current working directory is location of initial (web-requested) PHP file
- "require" or file open commands start here



Basic Syntax

- Variables always include leading \$
 - Like perl: never unadorned
- echo, print, and printf
 - Like sh, perl, and C. Take your pick.
- Comparators and assignments
 - =, >, >=, ==, etc like **c**
 - -> (method) like **C++**
 - . and .= (concatenation) like **perl**
 - Backticks (` `) for shell execution (like **sh**)

Scope and Variables

- Web-requested file is outer-most (global) scope
- Functions have scope, objects have scope, like OO languages
 - Must stipulate "global" within a function to gain access a global variable
- PHP3 (ours) uses variables **by value** only. (PHP4 includes **references**)

A Real Example: hit counter

```
<?
  $doc = getenv("REQUEST_URI");
  $COUNTFILE="/home/paco/public_html/counts";

  $dbm = dbmopen("$COUNTFILE", "w");
  $count=1;
  if( $dbm )
  {
    if( dbmexists( $dbm, $path ) )
    {
      $count += dbmfetch( $dbm, $path );
      dbmreplace( $dbm, $path, $oldcount );
    } else {
      dbminsert( $dbm, $path, $count);
    }
    echo "$count";
  } ?>
```

Some Available Libraries

General

- HTTP
- GD (GIF images)
- IMAP (e-mail)
- Cookies
- Math

Database

- ODBC (Microsoft)
- MS SQL
- Informix
- MySQL
- PostgreSQL

 No need to explicitly load or ask for libraries
All are available at all times

PHP Techniques

1. Cookies for tracking web visitors
2. HTTP Authentication against arbitrary user databases
3. Connecting to Databases

Trick #1: Cookies

- Often used to track visitors
- Stored on user's hard drive
- Web browser sends cookie each time it requests a page
- Can contain arbitrary data
- Must be issued before HTML

```
SetCookie("UserName",  
$UserName,  
time()+3600,  
"/~paco/cookies"  
,"www.cs.virginia.edu");
```

```
Set-Cookie: UserName=Paco; expires=29-Dec-99 13:52:23 GMT;  
path=/~paco/cookies; domain=www.cs.virginia.edu
```

Properties of Cookies

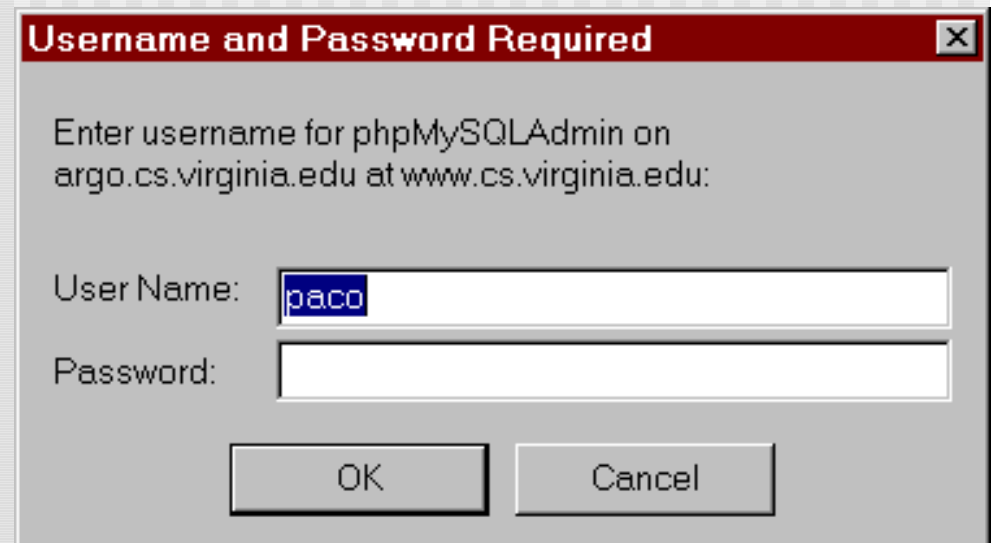
- Cookies have **expiration dates**
 - Set it far enough in future to be useful
 - Browser will stop sending it after exp date
- Cookies have **scope**
 1. **Server** (e.g. "www..." or ".cs.virginia.edu")
 2. **Path** (e.g. "/" or "/~paco/")
- Cookie will be **environment variable**
 - E.g. `SetCookie("ID", "Paco", ...)` yields variable `$ID` set to "Paco"
- Use **cookie as index** into a database
 - E.g. `"select from users where id = '$ID'"`

Another Way to Use Cookies

- Store the user data in the cookie:
 - `SetCookie("UserInfo", "Paco:UVA:paco@cs:password:143.00:...")`
 - Colon-delimited list of attributes
- Issues
 - Cookies are sent in plain text
 - All info is lost when cookie expires (renew it by sending a new one when visitor returns)
 - You should encrypt this kind of cookie to prevent sniffing of passwords

Trick #2 HTTP Auth

- PHP can cause browser to prompt for authentication
- PHP provides user's input in `$PHP_AUTH_USER` and `$PHP_AUTH_PW`
- Any scheme can be used to verify user
 - Database
 - Unix password
 - IMAP



Prompting for Authentication

- Send Auth headers to browser
- If auth fails, user sees error
- If they enter data, code proceeds

```
function auth()
{
    if(!isset($PHP_AUTH_USER)) {
        Header("WWW-Authenticate: Basic
            realm=MyRealm");
        Header("HTTP/1.0 401 Unauthorized");
        echo "User hit Cancel button\n";
        exit;
    } else {
        echo "Hello $PHP_AUTH_USER.<P>";
    }
}
```

Trick #3: Connecting to Databases

- Create a "handle" to the db
- Handles can persist (for efficiency)
- Once connected, issue raw SQL queries
- Results returned in data structures (like arrays, associative arrays, etc)

Connecting to a Database

- Connect to server with **server name**, **userid**, and **password** (if any)

```
$db = mysql_pconnect(  
    "server...",  
    "nobody",  
    "" );
```

- **\$db** is either a valid handle, or -1
- **pconnect** means "persistent." Future PHP calls to "**pconnect**" will use existing connection if possible

Fetching Rows

- Store results in special holding pen
- Move results to desired data structure
- Associative array is nice
- Can check for existance of data before printing or using it

```
$results = mysql_db_query(  
    "paco",  
    "select * from students  
    where grade = \"A\"",  
    $db );  
  
$row = mysql_fetch_array(  
    $results );  
  
echo "$row[\"FirstName\"]  
    $row[\"LastName\"]:  
    $row[\"Grade\"]\n";
```

Significant Examples

- IMP
 - get IMAP mail through the web
 - Maintains address book with db backend
 - <http://web.horde.org/imp/>
 - <https://www.cs.virginia.edu/mail/>
- phpMyAdmin
 - Full database management through the web
 - 3800 lines of PHP (153K)
 - <http://www.phpwizard.net/>
 - <https://www.cs.virginia.edu/~paco/db/>

Major Resources

- <http://www.php.net/>
- <http://www.phpwizard.net/>
- <http://www.devshed.com/>
- <http://www.mysql.net/>

phpWizard.net
Building Dynamic Websites with PHP

