PHP

By the legend, Adam Prax, of the Internet Programming/Web group
History

- PHP 1 (Personal Home Page Tools) - 1994
  - Created by Rasmus Lerdorf
  - Access logging
  - IP tracking
  - Form creation

- PHP 2 - 1996
  - Arrays, variables
  - MSQL and Postgres95 support
  - Conditions
  - Regular Expressions
  - Access restriction

- PHP 3-1998
  - Bigger, Faster, Stronger
  - Support for Windows

- PHP 4-2000
  - Technical overhaul
  - New Zend Engine

- PHP 5-2004
  - Support for XML
  - Object Orientation
  - SQLite
What is PHP?

- Dynamic web page design (forms, widgets, hit counters)
- Integrated into HTML files
- Imperative, object-oriented language
- Used for online database queries
PHP installation and usage

- Installation easy on most distributions, excruciatingly painful on others
  - Fedora 8 rules
    - `yum install php; //wow, so easy!`
  
- PHP usually runs with Apache
  - PHP module in apache allows automatic runtime compilation and execution of php files
Portability

- Code typically executed and debugged on server
  - Most typically the Apache server
    - Can be installed on Windows, Linux, and Unix
    - PHP is only installable on Windows, Linux, and Unix
- There are standalone compilers for Windows and Linux (*phc* being the most popular)
- A client of any platform can execute php code on the server
  - Code is executed by server, not in browser
Key uses

- **Server-side scripting**
  - Does not need source files in “cgi-bin” folder
  - Code is executed on the server

- **Interactive website design**
  - Forms
  - Guestbooks
  - Counters

- **Database management**
  - Queries
  - Input

- **Desktop programs**
  - Using GTK 2
Influences on PHP

• Ideas from C++
  – Block Structure
  – Boolean Expressions: if-else
  – Object-Oriented
  – Control Structures: for, do

• Ideas from Python, Perl
  – Regular expressions
  – Dynamic type binding
    • That's right, PERL
      – A typeless, script-kiddie “language”
        • Not a real language
  – Arrays, Hashes
  – Interactive Command line
    • Exactly like Python
Case Sensitivity

- Case insensitive: keywords, functions, classes
  - `echo("hey guys!!!!!!!!!")`;
  - `eChO("same thing")`;
  - `myawesomefunc(); //will call same function`
  - `MYAWSOMEFUNC(); //will call same function`

- NOT Case insensitive: variables
  - `$abc = "something"`;
  - `$ABC = "something else"`;
Type Conversion

- Implicit type conversion
  - $x = 5$
  - $y = 4.5$
  - $z = x + y; // x is cast to float$
  - $word = \text{"Frank"} $
  - $z = x + word // word is cast to int$

- Any combination of types for assignment works, including unintended assignments, leading to increased chance of run-time errors
Conditional and Control Expressions

- Conditionals are syntactically the same as in C, nothing to see here
  - Even the ternary operator
- Control statements are exactly the same syntax too
  - One exception: foreach
    - `foreach ($array as $element) { <do stuff>}`
- The scoping is different from C
  - Variables outside a control expression cannot be seen

- What is false?
  - `false`
  - `0`
  - `""`
  - `"0"`
  - `NULL`

- What is true?
  - Any nonzero number that is not null or an empty string
References and Pointers

- $a = &b;  \text{//a points to the memory cell referenced by b}$
- `unset($name); //sets $name to "nothing"`
  - Sets the value of name to NULL
  - Will not free memory if another pointer points to same location
    - No dangling pointer problem
- Setting a pointer to another location will also free the previous memory location
  - If no other pointer was pointing at that previous location as well

Function double(& $number) {
    $number *= 2;
}
Interactive shell

```php
php > $wasabi = "snowflake";
php > $blargh = & $wasabi;
php > $nom = & $wasabi;
php > unset($blargh);
php > echo $blargh;
PHP Notice: Undefined variable: blargh in php shell code on line 1
php > echo $nom;
snowflake
php > unset($wasabi);
php > echo $nom;
snowflake
php > echo $wasabi;
PHP Notice: Undefined variable: wasabi in php shell code on line 1
php >
```
Statements

- Statements are separated by semicolons
- Compound statements are delimited by brackets

```php
if ($this == $that) {
    echo "So it is";
    $a = 5; //hey, a variable assignment!
} //bracket indicates end of "if" block
```
Scope

• Variables are visible everywhere except in functions and control statements
  – Variables local to functions are only visible inside the function
  – Variables outside a function are also not visible

• No scope inside loop, conditional branch, or any other block statements

• `global` keyword makes variables outside of control or function scope accessible
$aNumber = 5;
for ($x = 0; $x <=5; $x++) {
    echo $aNumber++; //I can't see you!
}

echo "<br />";

for ($x = 0; $x <= 5; $x++) {
    global $aNumber;
    echo $aNumber++; //There you are!
}

12345
678910
Object-Oriented Support

- Supports superclasses, subclasses, function inheritance, encapsulation

- Accessibility levels: same as in C
  - Public
  - Protected
  - Private

- NOT purely object-oriented, similar to C++ and not like Java
  - Functions and variables can be defined without the need for classes
class Adam {
    protected $strength = 10;
    public function killOpponent ($opponent) {
        $opponent->life=0;
        echo "$opponent dies!";  
    }
    public function getStrength() {return $this->strength;}
    public function taunt() { return “Abandon all hope!”; }
}

class psuedoAdam extends adam {
    protected $strength = 1000;
    public function killOpponent ($opponent) {
        $opponent->life=-10;
        echo "$opponent dies horribly!";
    }
}
$difficult_monster = new Adam;
$impossible_monster = new pseudoAdam;

echo "Adam is ".$difficult_monster->getStrength().' strong <br />

echo "pseudoAdam is ".$impossible_monster->getStrength().' strong <br />

echo "pseudoAdam says: Abandon all hope! " <br />

Adam is 10 strong
pseudoAdam is 1000 strong
todo: Some text
PHP and HTML

```html
<html>
  <head>
    <title>"Loop example"</title>
  </head>
  <body>
    <?php
      for ($x = 1; $x <= 7; ++$x) {
        echo "<font size=".$x.""> This is font size ".$x."</font><br />
      }
    ?>
  </body>
</html>
```
This is font size 1
This is font size 2
This is font size 3
This is font size 4
This is font size 5
This is font size 6
This is font size 7

Done
PHP and html forms

<html>
<head>
    <title> "Form submission "</title>
</head>
<body>
    <form method = "post" action ="database.php">
        Your Name: <input type="text" name = "name"> <br />
        Password: <input type = "password" name = "password"> <br />
        The greatest Linux distribution: <input type="text" name = "distribution"> <br />
        A failure at life: <input type="text" name = "failure">
<br />
        <input type = "submit"/>
    </form>
</body>
</html>
Your Name: Adam Prax
Password: ************
The greatest Linux distribution: Fedora
A failure at life: George Bush
Submit Query
<?php
$name = $_POST['name'];
$password = $_POST['password'];
$failure = $_POST['failure'];
$database = "database.txt";
$dataStream = fopen($database, 'w');
fwrite ($database, $failure);
fwrite($database, "\n");
fwrite($database, $distribution);
fclose($dataStream);
?>
References

  - Written by the God himself
  - Instructions on how to process HTML forms with php
  - Download and install php
- http://httpd.apache.org/download.cgi
  - Download and install Apache