

# PHP

- **Web site is <http://www.php.net>**
- **Recursive acronym for PHP: Hypertext Processor**
- **Apache Software Foundation project.**
- **C like language which may be embedded in HTML**
  - \* **Requires PHP processor installed**
- **Functionally similar to Perl CGI**
  - \* **Primarily a matter of taste**
- **Comments: C++, Perl style**

**`/* multiline`**

**`comment */`**

**`//comment to end of line`**

**`#same as above`**

# Variables

- **Variables start with \$**
- **Don't predeclare**
- **Types include boolean, int, float, string, array, object and NULL**
- **Boolean values are TRUE and FALSE**
  - \* **Expressions with convert to boolean: 0 is false, anything else is true.**
- **ints and floats are what you expect**
- **strings are Perl-like**
  - \* **single quotes, double quotes, and heredocs are supported**

//example

```
$total *= $i;
```

# Embedding PHP in HTML

- `<? code ?>`
  - \* simplest
- `<?php code ?>`
  - \* Compatible with XML, XHTML
- `<script language="php">code</script>`
  - \* Compatible with processors which don't like the others directives e.g. Front Page
- `<% code %>`, `<%= expression %>`
  - \* ASP / JSP style
  - \* `%=` prints expression to HTML page
  - \* must be enabled in config file

# Control structures

- Usual for, if, if-else, while, do-while, switch
  - \* { } optional for single line bodies
- Also supports *foreach* construct
- Alternative forms available
  - \* Use : instead of opening {
  - \* Use *endif; endfor;* et. al. to close loop
  - \* may be more readable when PHP and HTML mixed

- **Example**

- \* **Outputs a table of factorial values**

```

<table>
<tr><th>N</th><th>N!</th></tr>
<tr align="right"><td>0</td><td>1</td></tr>
<?php
    $total = 1;
    for($i = 1;$i<21;$i++):
        $total *= $i ?>
<tr align="right"><td><?php echo "$i" ?> </td>
    <td><?php echo "$total" ?></td></tr>
<?php endfor; ?>
</table>

```

# Arrays

- Use same \$ syntax for array names
- Both indexed and associative arrays supported
- Can just create on the fly or use *array* function

```
$arr[5] = 10;
```

```
$values = array(1, 3, 5)
```

```
$map=array("CT" => "Connecticut", "RI" =>"RHode Island");
```

```
echo $map["CT"];
```

\* Assigning to empty brackets adds to end of array

```
$arr[] = 20;
```

\* Use *unset* remove key/value pair from array

# Useful array functions

- **array\_push, array\_pop, array\_unshift, array\_shift**
- **sort, rsort**
  - \* **indexed type arrays**
- **asort, arsort, ksort, krsort**
  - \* **associative arrays. “a” values sorts by value, “k” by key**
- **usort, uasort, uksort**
  - \* **pass user defined function for comparing**
- **count**
- **array\_keys, array\_values**

- **current, next**

- \* **Used to iterate through array**

- \* **prev, reset also available**

```
$arr = array(2, 3, 5, 7, 11, 13, 17, 19);
```

```
do {
```

```
    $v = current($arr);
```

```
    echo "$v<br>";
```

```
} while (next($arr));
```

- **Can also use *foreach* control construct to loop through array**

```
foreach ($arr as $n) {
```

```
    echo "$n<br>";
```

```
}
```

# Parameters

- **PHP has built in support for accessing HTTP parameters and server environmental variables**
- **Each parameter automatically creates variable of same name**
  - \* e.g. parameter named ‘fish’ will create variable named “\$fish”
  - \* may be a security risk since web users can manually add parameters to url: *.../vars.php?fish=cod*
  - \* may be disabled in configuration file
- **Parameters also found in arrays named `$_GET` / `$HTTP_GET_VARS` and `$_POST` / `$HTTP_POST_VARS`**
  - \* underscore versions available after version 4.1.0

- **Server information in `$_SERVER`, `$_ENV` (`$HTTP_SERVER_VARS`, `$HTTP_ENV_VARS`)**
- **Information added by user available in `$_REQUEST` (no pre 4.1.0 equivalent)**

# Cookies

- **Cookies added by using the *setcookie* function**
  - \* **bool setcookie (string name , string value , int expire, string path, string domain , int secure)**
  - \* **all values except name optional**

- **Example**

```
<?php  
setcookie("demo","RH is great",time( ) + 120,"/","",0)?>
```

- **Cookie information is sent to browser as part of header**
  - \* **setcookie must be called *before* any other output**

- **To read cookie values, use `$_COOKIE` / `HTTP_COOKIE_VARS`**

```
<?php echo $HTTP_COOKIE_VARS["demo"]?>
```

# Sessions

- **PHP supports session information**
  - \* **Start session with *session\_start***
  - \* **Add a variable to session by using *session\_register***

```
<?php session_start( );  
if (!isset($count)) {  
    $count = 0;  
} else {  
    $count++;  
}  
session_register('count');  
?>
```

- **Variable can then be read into another page**

```
<?php session_register('count');?>  
You've visited the previous page <?php echo $count ?> times.</p>
```

# Functions

- **Functions are declared using function keyword**

```
function show_array($name, $array) {
```

- **Parameters passed as shown**

- \* **compile (“Parse”) error to not pass specified arguments**

- \* **extra arguments ignored**

- **May add default arguments, like C++**

```
function substring($string,$start,$length=-1)
```

- **Use *return* to return value**

# Classes supported

- **Similar to Java / C++**

```
<?php class Cart {  
    var $items; // Items in our shopping cart  
    // Add $num articles of $artnr to the cart  
    function add_item ($artnr, $num) {  
        $this->items[$artnr] += $num;  
    }  
    // Take $num articles of $artnr out of the cart  
    function remove_item ($artnr, $num) {  
        if ($this->items[$artnr] > $num) {  
            $this->items[$artnr] -= $num;  
            return true;  
        } else {  
            return false;  
        }  
    }  
}  
?>
```

- **Inheritance, constructors also supported**
- **Reflection concept of Java supported**
  - \* **Can query object to find what attributes and methods it has**

# References

- **PHP supports references**
  - \* **More C++ or Java like than C pointer like**
- **Use & to indicate something should be a reference**
  - \* **`$b = & $a;`**
    - space after & required**
  - \* **both variables refer to same value**

- **Example**

```
function double_it(&  
    $value *= 2;  
}
```

```
$a = 10;  
echo "a is $a<br>";  
double_it($a);  
echo "a is $a<br>";  
$b = & $a;  
$b = 57;  
echo "a is $a<br>";
```

- **Output**

a is 10

a is 20

a is 57

# MySQL registered trademark of SQL AB

- **Open Source Relational Database**
- **Developed and owned by SQL AB**
  - \* **Swedish company**
  - \* **Offers both commercial (~4,000 customers) and GPL (estimated 4,000,000 customers) licenses**
- **Wall Street Journal article <http://webreprints.djreprints.com/785490482991.html> notes Cox Communications (cable-tv) using**
  - \* **2.4 billion rows**
  - \* **600 gigabytes of data**
  - \* **Oracle estimate for system: \$300,000 plus service**
  - \* **MySQL \$1,000 + \$12,000 /yr support**

# MySQL features

- **Primary keys**
- **Foreign keys**
  - \* **enforcement depends on table type**
- **ACID transactions**
  - \* **for certain table types**
  - \* **others are autocommit**
- **Usual create, select, insert, update, delete commands**
- **Users, passwords, grants**
- **Typical database data types**

# Not currently supported

- *Select ... into ... table -- uses Insert into table ... select*
- Stored procedures, triggers
- Views

# Oracle compatibility

- Administrative commands are different
- *mysql* is console interface (equivalent to SQLplus)
- use *show* command instead of querying from catalog. e.g. *show tables, show variables*
- use *desc [table]* to get table description
- decent suite of utilities available; backing up, dumping, etc.

# MySQL table types

- **ISAM**
  - \* **Indexed Sequential Access Method (old IBM format)**
- **MyISAM**
  - \* **MySQL version of ISAM**
    - Data in \*.myd**
    - Indexes in \*.myi**
- **Merge**
  - \* **Experimental grouping of compatible tables**
- **Heap**
  - \* **hashed indexed tables stored in memory only**

- **InnoDB**
  - \* **Third party developed table format**
  - \* **Supports foreign keys and transactions**
  - \* **Uses tablespace concept, similar to Oracle**
- **BerkeleyDB**
  - \* **Another opensource backend**
  - \* **Supports transactions**

# Working without transactions

- **Multiple row commands may do partial inserts**
  - \* **stop on first primary key or unique index violation**
  - \* **can use IGNORE as part of insert or update to skip rows with violations and process remaining rows**
- **Can lock tables to get isolation similar to that available in transactions**

# MySQL and PHP

- **PHP supports MySQL commands**
  - \* **Use `mysql_connect` to attach to database**
  - \* **Use `mysql_select_db` to select database**
  - \* **Use `mysql_query` to performs queries**
  - \* **Use `mysql_fetch_row` to retrieve data from select**
  - \* **Use `mysql_close` to close connection**

# MySQL and Perl

- **Use DBI interface to use Perl to connect to database**
  - \* *connect* to connect to database
  - \* *prepare* to perform query
  - \* *fetchrow* to get rows
  - \* *finish* to close result
  - \* *disconnect* to detach from database