PHP

PHP is mostly used for building websites. It typically runs on the web server and is accessed via a web browser. When developing with PHP on your computer environment, it is important to launch a local server environment such as WAMP or MAMP.

Basic Syntax:

<?php ... ?> Indicates the start and end tag of a php statement
$<var> All variable names are prefaced with '$'
`` Single quotes denote strings
. The concatenation symbol
% Begin Format string rule:
%[0].[0-9]+[f,d,s]
[0] - pad with spaces or 0
. - indicates a period
[0-9]+ - specify number of decimals
f = floating point
d = decimal
s = string

Variable Declarations:

```
$stringx = 'hello' | "hello";  (double or single quotes)
$inty = [0-9]+;
$floatz = [0-9]+\.[0-9]+;
$boolean1 = true | false;
$array2 = array('x', 'y', 'z', ...);
$array3 = array('key' => 'value'); ... $array3['key'] denotes 'value'
class X {
        function X(){
        ...
        }
}
$nullx = null;
```

Logic:

```
if( condition ){ ... }
elseif( condition ) { ... }
else { ... }
while(){ ... }
for($i = 1; $i <10; $i++){ ... }
```

Special String Characters:

\n Newline (ASCII 10)
\t Tab
\\ \
\$ $
\" "
\0 .. \777 Octal (base 8) number
/x0 .. /xFF Hexadecimal (base 16) number

Array Functions:

array_diff(arr1, arr2, ...);
returns values in array one that are not present in other arrays
array_flip(arr);
keys from array become values and vice versa
array_intersect(arr1, arr2);
returns array of intersecting elements
array_merge(arr1, arr2);
returns second array appended to first
array_pop(arr);
array_push(arr, var1, var2);
array_search(var, arr);
array_count(arr, var1, var2);
in_array(var, arr); - returns bool
sort(arr);
count(arr);

String Functions:

echo (str);
Outputs all parameters
bin2hex(str);
converts binary string to hexadecimal
explode(delimiter, str [, int limit]);
returns array of substrings split by delimiters
implode([glue,] str);
joins array elements into string (glue is optional delimiter)
hex2bin(str);
trim(str);
strcmp(str);
strlen(str);
substr(str, start [, length]);
strtolower(str);
strtoupper(str);

Reg-Ex Functions:

preg_match(pattern, str);
Returns # of matches for pattern
preg_replace(pattern, replacement, str [, limit [, count]]);
Returns string with replacements
preg_split(pattern, str [, limit [, flag]]);
Returns array split at pattern
preg_grep(pattern, str [, flag]);
Returns array of matches from the input string