

PHP Details

In This Lecture

- Constants, Expressions, Operators
- Control Structures
 - Conditional statements
 - Looping statements

Defining Constants

- Example:

- `define("_PI_", 3.1415);`
- `define("NAME_LABEL", "Name");`

- Usage:

- `echo _PI_;`
- `echo NAME_LABEL . " : $name ";`

Constants

- No \$ required.
- Must be defined with `define()` function.
- Can be accessed anywhere in script
- Cannot be changed once they have been set.
- Only scalar values allowed.

Operators

- **&**

- **^**

- **|**

- **&&**

- **||**

- **? :**

- **= += - =**

- *= /= . = % =**

- &= |= ^= << =**

- >> =**

- **and**

- **xor**

- **or**

Some selected Math functions

- <http://www.php.net/manual/en/ref.math.php>
 - Trigonometric functions, commonly used functions such as round, ceiling functions, etc.
 - Square root, logarithms etc.

Control structures: if, if-else,

```
<?php
```

```
    if ($p > $q) echo "Whatever!";
```

```
    if ( afunctionthatreturnsbool() )
```

```
    {
```

```
        doThingsHere() ;
```

```
    } else {
```

```
        doAnother() ;
```

```
    } ?>
```

Control Structures: if else if

```
<?php
```

```
    if ( cond ) {
```

```
        //do stuff
```

```
    } else if ( another_cond ) {
```

```
        //do other stuff
```

```
    } else {
```

```
        //do other other stuff
```

```
    }
```

```
?>
```


Control Structures: if-elseif-endif

```
<?php
```

```
    if ( cond ) :
```

```
        echo "Hello";
```

```
    elseif ( another_cond ) :
```

```
        echo "World";
```

```
    else:
```

```
        echo "!";
```

```
endif;      ?>
```

If statement with template text

```
<?php $a = 4; ?>
```

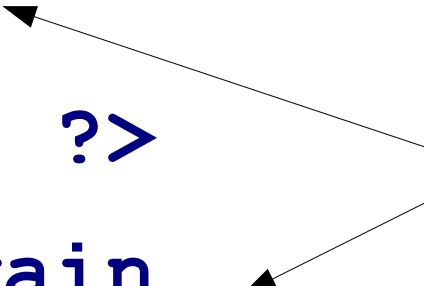
```
<?php if ($a==4) { ?>
```

```
Hello World
```

```
<?php } else { ?>
```

```
World, Hello Again.
```

```
<?php } ?>
```



Template text inside conditional Statements. No echo needed.

Control structures: while

```
while (expr)
    statement;
```

```
while (expr) {
    statement1;
    statement2;
}
```

```
while (expr):
    statement1;
    statement2;

endwhile;
```

Control Structures: do-while

```
do {                                //singlestatement
    statement1;
    Statement2;
}
while (cond) ;

do statement;
while (cond) ;
```

Control structures: for

```
<?php
    for ( $i =0; $i<10; $i++ ) {
        //iterated statements here
    }
?>
```

Control Structures: foreach

```
$arr = array(1,2,3,4);  
$sum = 0;  
foreach ($arr as $item) {  
    $sum += $item;  
}  
  
$assoc = array( 'fname'=>'Juan',  
    'lname'=>'Dela Cruz' );  
  
foreach ($assoc as $key=>$value) {  
    echo $key . ":" . $value;  
}
```

Control structures; break and continue, return

- break;
 - Breaks the execution of a for, foreach, while and switch
- continue;
 - Used in a looping structure to skip an iteration
- return;
 - Used in exiting functions or “returning” values in functions.

Control Structures: switch

```
switch ($var) {  
    case 0: dosomething();  
        break;  
    case 1: doAnything();  
        break;  
    default: defaultAction();  
}
```

```
switch ($var) {  
    case "apple":  
        dosomething();  
        break;  
    case "orange":  
        doAnything();  
        break;  
    default: defaultAction();  
}
```


Control structures: require and include

- Used to include and evaluate a specified file.
- Useful when you have 'modularized' script files or other kinds of files to combine

```
include 'config.php' ;
```

```
include 'http://myserver.com/file.txt' ;
```

include and require

- `require()` is similar to `include` but will produce an error and halt execution if it fails to find the file specified.
 - `include()` will only generate a warning.

include_once, require_once

- Similar to include and require except that PHP will check if the file has already been included (or required).
 - Will include/require specified file only once

Control structures: goto

- Goto is a jump statement. (PHP 5.3 only)

<?

```
goto thispart;
```

```
echo "Hello";
```

```
thispart:
```

```
echo "World";
```

?>