

IDIA.618.185 - Dynamic Web Sites : Spring 2004

Programming PHP

Erich Spencer
Adjunct Professor
University of Baltimore
School of Information Arts and Technologies

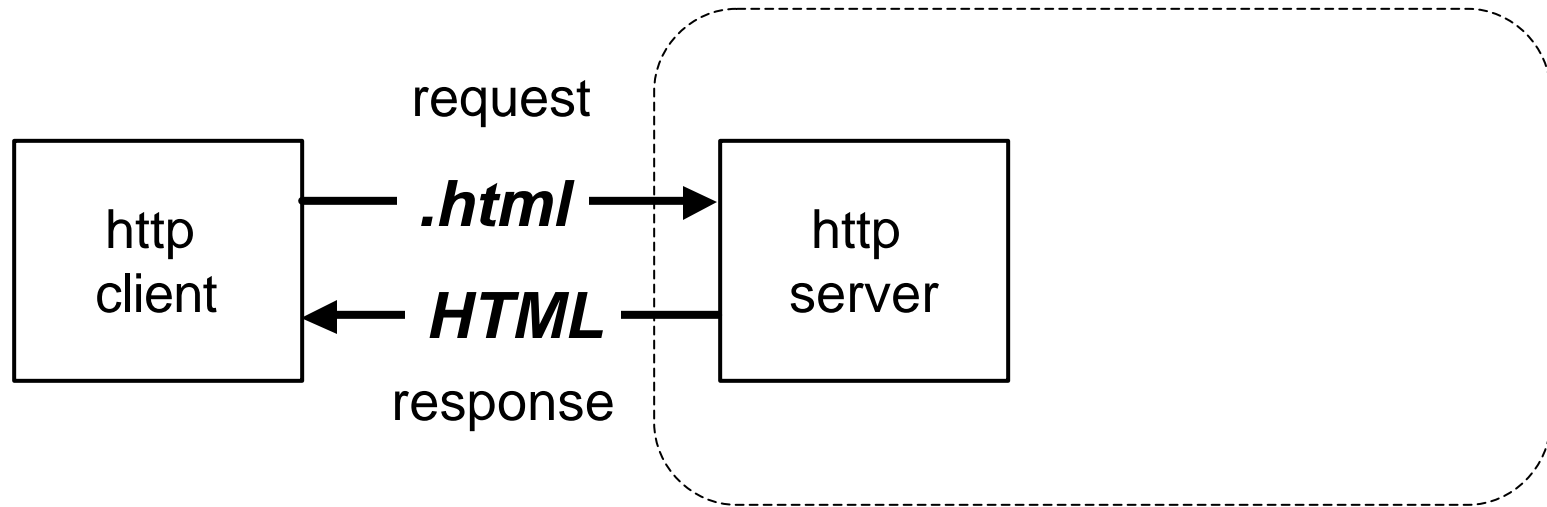
Programming vs. Scripting

- PHP is a scripting language
 - scripting makes it easier to write powerful programs
- PHP adheres to core programming principles
 - syntax
 - structure
 - style
- PHP code is *interpreted*
 - not pre-compiled, like programming language code

Types of Scripting

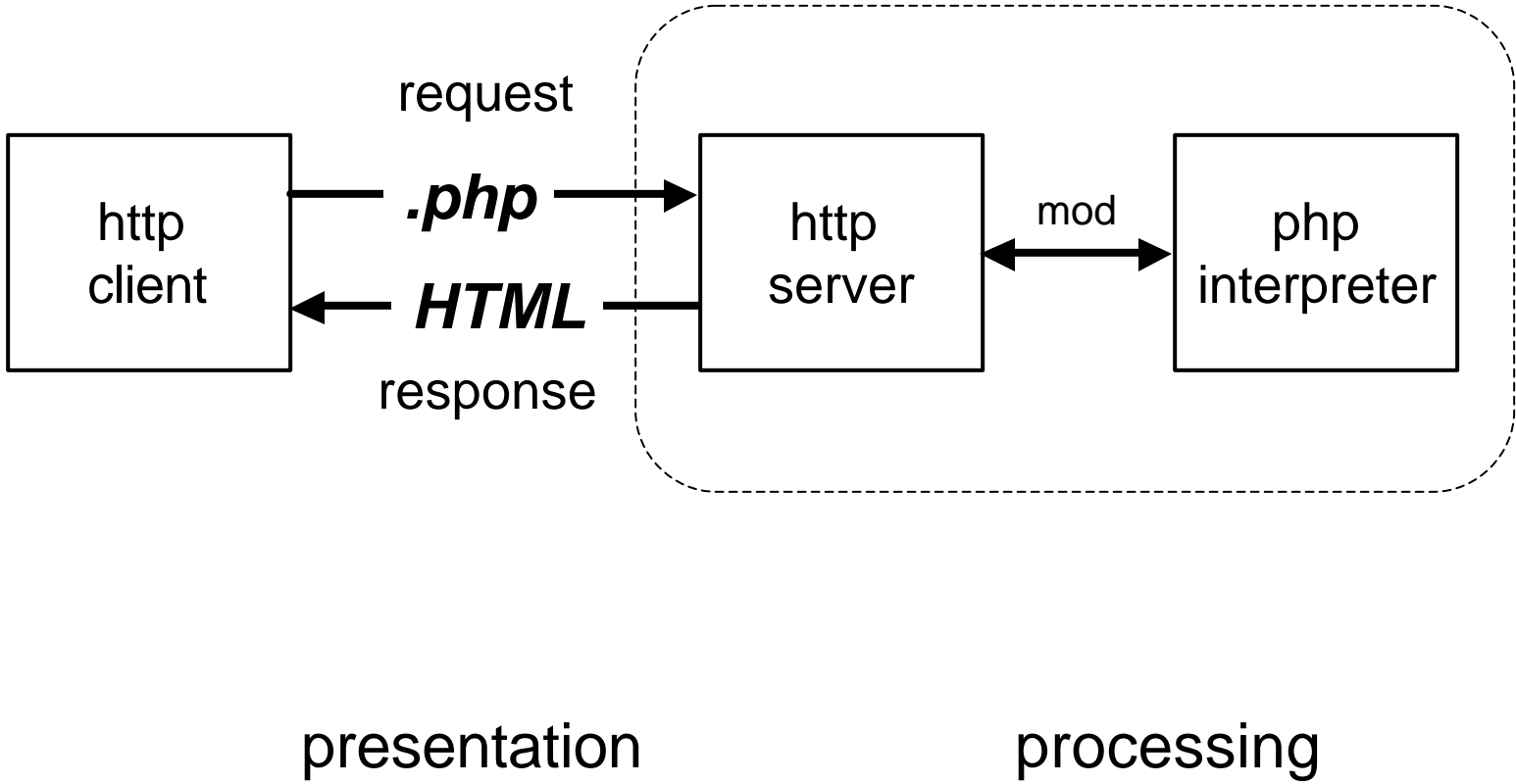
- Client-side scripting
 - Scripts interpreted and executed *in client*
 - JavaScript, VBScript
- Server-side scripting
 - Scripts interpreted, compiled, then executed *in server*
 - Includes Perl, JSP, CFM, ASP, and PHP

PHP + HTTP Request/Response



presentation

PHP + HTTP Request/Response



PHP: Hypertext Preprocessor

- A scripting language
- An interpreted language
- Syntax derives from
 - C (*programming*)
 - Perl (*scripting*)
- Used to develop server-side applications
 - Complete, standalone programs
 - May also embed scripts in HTML
- Tightly integrated with Apache and MySQL
 - Very fast execution

PHP: Hypertext Preprocessor

- HTTP server identifies PHP script file by “.*php*” file extension
- PHP interpreter identifies code blocks by special characters

<?php ... ?>

- Characters offset, or *escape*, code from HTML
- Enclosed contents evaluated as PHP code
- All PHP code must be in form of
 - Statements
 - Functions
 - Structures

Programming Basics

- Statements
 - A complete instruction or operation
 - Must be terminated
 - Terminator typically a semi-colon (*statement ;*)

```
print("Dynamic Web Sites");
```

Programming Basics

- Variables
 - A *human-friendly reference* to a location in memory
 - A place to temporarily *store a value*
 - PHP uses “\$” to identify variables

```
$spacer = " - ";
```

```
$course_id = "IDIA.628.185";
```

```
$course_title = "Dynamic Web Sites";
```

Programming Basics

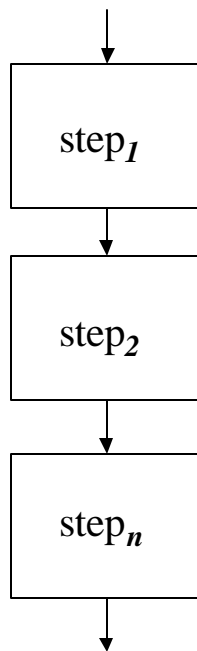
- Constants
 - A value that does not change for the life of the program
 - Usually written in capital letters (MY_NAME)

Programming basics

- Operators
 - Characters used to **assign** (= , .=)
 - Characters used to **calculate** (+ , - , * , / , %)
 - Characters used to **compare** (== , != , < , > , <= , >=)
 - Characters used to **assert logic** (&& , || , !)
 - Characters used to **concatenate** (.)
- Functions
 - 1 or more operations encapsulated as a logical unit
 - Takes a specific, predictable input
 - Performs a predictable, repeatable operation
 - Returns a predictable, single result
- Structures
 - Basic building block of every program

Structures

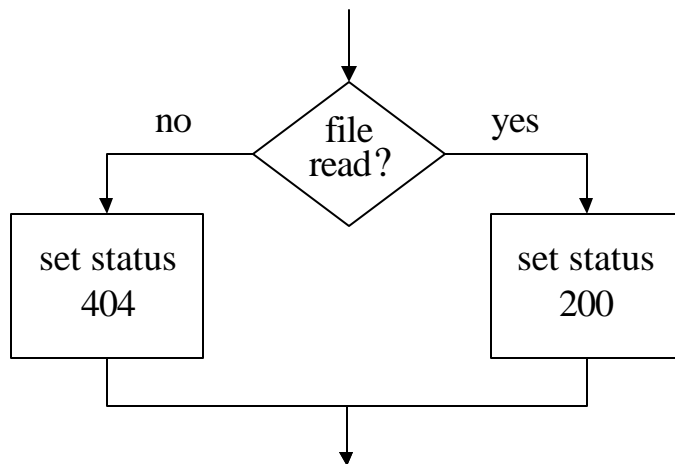
- Sequence
 - Serial execution of instructions
 - No branching



```
$spacer = " - ";  
$course_id = "IDIA.628.185";  
$course_title = "Dynamic Web Sites";  
print($course_id);  
print($spacer);  
print$course_title);
```

Structures

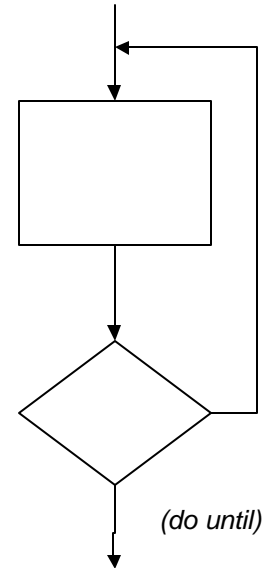
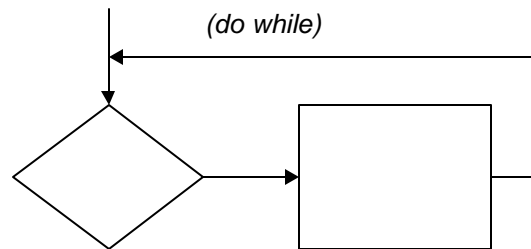
- Decision
 - Allow, deny, suspend, or branch program execution



```
if ( readfile($file) ) {  
    $status = 200;  
} else {  
    $status = 404;  
}
```

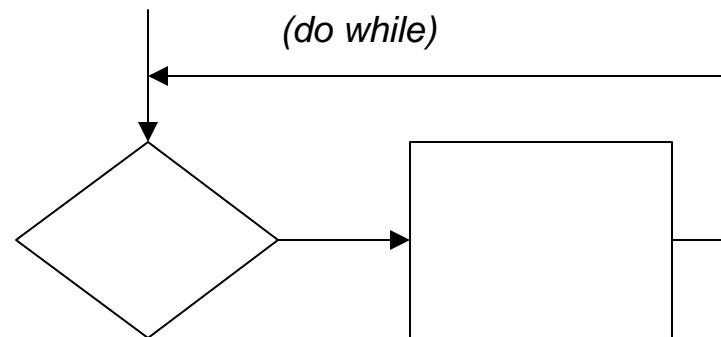
Structures

- Loop
 - Repetition, iteration
 - Evaluate condition(s)
 - Remain in loop / exit
 - DO WHILE and DO UNTIL



Loops

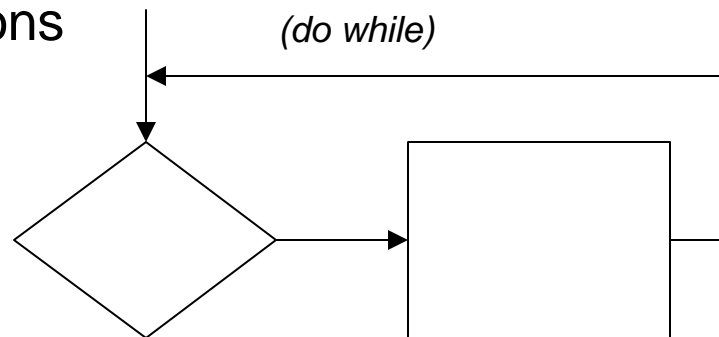
- DO WHILE
 - evaluate condition *first*
 - perform operation(s)
 - repeat



```
$yourAge = 18;  
$alcoholLaw = 'No way!";  
while ( $yourAge < 21 ) {  
    print( $alcoholLaw );  
    $yourAge = $yourAge + 1;  
}
```

Loops

- FOR
 - a type of DO WHILE
 - more control over iterations

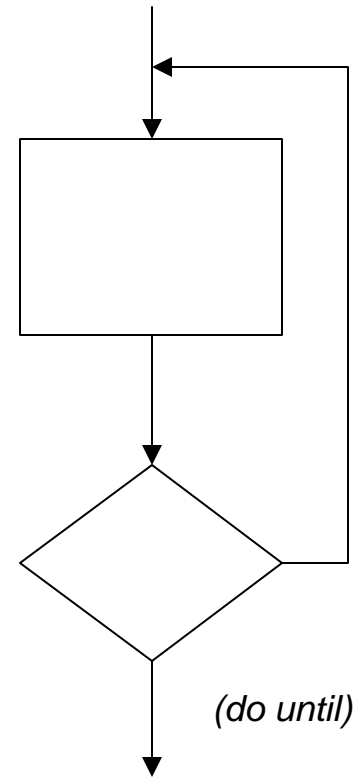


```
$contract = 10;  
for ( $i = 0; $i < $contract ; $i++) {  
    print("This is year " . $i . " of our contract. ");  
    print("We have " . ($contract - $i) . " years left.");  
    print("\n");  
}
```

Loops

- DO UNTIL
 - perform operation(s) *first*
 - evaluate condition
 - repeat

```
do {  
    print($yearsLeft);  
    print($gettingOut);  
    $yearsLeft = $yearsLeft - 1;  
} while ( $yearsLeft > 0 )
```



Commenting Your Programs

- Use comments to document your code

```
/**
 * A multi-line comment. Java style. Often used to start
 * a program file, describing the main operation and results
 * of the program. Also includes date, author, etc...
 */
$total = 4 + $amount;

// A single line comment, directly above, describes a step.
$amount = 3;

$name = "Erich"; // May also appear on same line
```

You've been *coding* using HTML!

- **Statements**
 - `<p>`This tagged sentence is an expression`</p>`
 - The end tag `</p>` **terminates** the statement
- Variables
 - A named container to store some value
 - Recognized by an identifier, such as the dollar sign (*\$variable*)
- Operators
 - Characters used to evaluate or calculate (+, =, !=, &&, ==, *...)
- **Functions**
 - `function p($textblock) { set margin around $textblock; ...; return $textblock; }`
- Conditions
 - Allow, deny, suspend, or branch program execution
- Loops
 - Program steps repeated until some condition changes

You've been *coding* using CSS!

- **Statements**
 - font-size: 12pt;
 - A semi-colon terminates the statement
- Variables
 - A named container to store some value
 - Recognized by an identifier, such as the dollar sign (*\$variable*)
- Operators
 - Characters used to evaluate or calculate (+, =, !=, &&, ==, *...)
- **Functions**
 - h2 { font-size: 12pt; }
- Conditions
 - Allow, deny, suspend, or branch program execution
- Loops
 - Program steps repeated until some condition changes