Lecture No: 1
Topic: Getting Started with PHP

Objectives:
In this chapter you will:
- Create PHP scripts
- Create PHP code blocks
- Work with variables and constants
- Study data types
- Use expressions and operators

History of PHP
- PHP as it's known today is actually the successor to a product named PHP/FI.
- Created in 1994 by Rasmus Lerdorf, the very first incarnation of PHP was a simple set of Common Gateway Interface (CGI) binaries written in the C programming language.
- Originally used for tracking visits to his online resume, he named the suite of scripts "Personal Home Page Tools," more frequently referenced as "PHP Tools."
- Over time, more functionality was desired, and Rasmus rewrote PHP Tools, producing a much larger and richer implementation. This new model was capable of database interaction and more, providing a framework upon which users could develop simple dynamic web applications such as guestbooks.
- In June of 1995, Rasmus released the source code for PHP Tools to the public, which allowed developers to use it as they saw fit. This also permitted - and encouraged - users to provide fixes for bugs in the code, and to generally improve upon it.

PHP: Hypertext Preprocessor
- PHP is a recursive acronym for "PHP: Hypertext Preprocessor".
- PHP is a server side scripting language that is embedded in HTML. It is used to manage dynamic content, databases, session tracking, even build entire e-commerce sites.
- It is integrated with a number of popular databases, including MySQL, PostgreSQL, Oracle, Sybase, Informix, and Microsoft SQL Server.

Common uses of PHP
- PHP performs system functions, i.e. from files on a system it can create, open, read, write, and close them.
- PHP can handle forms, i.e. gather data from files, save data to a file, through email you can send data, return data to the user.
- You add, delete, modify elements within your database through PHP.
- Access cookies variables and set cookies.
- Using PHP, you can restrict users to access some pages of your website.
- It can encrypt data.

Reasons to love PHP

REFERENCE: PHP Programming with MySQL 2nd Edition

- Simplicity
- Portability
- Speed
- Open Source
- Extensible

What's new in PHP 7
- PHP 7 is much faster than the previous popular stable release (PHP 5.6)
- PHP 7 has improved Error Handling
- PHP 7 supports stricter Type Declarations for function arguments
- PHP 7 supports new operators (like the spaceship operator: <=>)

Tools You need:
- PHP
  - ■ PHP is a server-side scripting language that allows your Web site to be truly dynamic.
- Apache
  - ■ Apache acts as your Web server. Its main job is to parse any file requested by a browser and display the correct results according to the code within that file.
- MySQL
  - ■ MySQL is the database construct that enables PHP and Apache to work together to access and display data in a readable format to a browser. It is a Structured Query Language server designed for heavy loads and processing of complex queries.

Creating Basic PHP Scripts:
- **Embedded language** refers to code that is embedded within a Web page (XHTML document)
- PHP code is typed directly into a Web page as
  a separate section
- A Web page containing PHP code must be saved with an extension of .php to be processed by the scripting engine
- PHP code is never sent to a client's Web browser; only the output of the processing is sent to the browser
- The Web page generated from the PHP code, and XHTML elements found within the PHP file, is returned to the client
- A PHP file that does not contain any PHP code should be saved with an **.html** extension
- .php is the default extension that most Web servers use to process PHP scripts

Creating PHP Code Blocks
- **Code declaration blocks** are separate sections on a Web page that are interpreted by the scripting engine

Standard PHP Script Delimiters

- A **delimiter** is a character or sequence of characters used to mark the beginning and end of a code segment
- The standard method of writing PHP code declaration blocks is to use the <?php and ?> script delimiters
- The individual lines of code that make up a PHP script are called **statements**

Understanding Functions

- A **function** is a subroutine (or individual statements grouped into a logical unit) that performs a specific task
  - To execute a function, you must invoke, or **call**, it from somewhere in the script
- A **function call** is the function name followed by any data that the function needs
- The data (in parentheses following the function name) are called **arguments** or **actual parameters**
- Sending data to a called function is called **passing arguments**

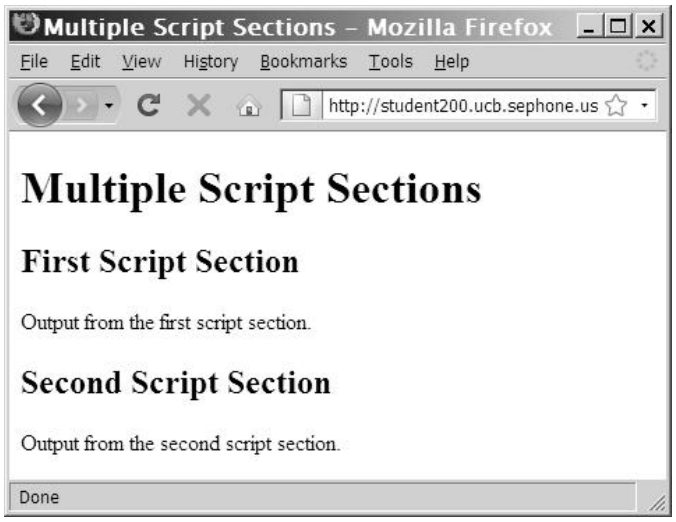
Displaying Script Results

- The echo and print statements are language constructs (built-in features of a programming language) that create new text on a Web page that is returned as a response to a client
- The text passed to the echo statement is called a “literal string” and must be enclosed in either single or double quotation marks
- To pass multiple arguments to the echo statement, separate the statements with commas
- Use the echo and print statements to return the results of a PHP script within a Web page that is returned to a client
- The print statement returns a value of 1 if successful or a value of 0 if not successful, while the echo statement does not return a value

Creating Multiple Code Declaration Blocks

- For multiple script sections in a document, include a separate code declaration block for each section

```
...
</head>
<body>
<h1>Multiple Script Sections</h1>
<h2>First Script Section</h2>
<?php echo "<p>Output from the first script section.</p>";
?>
<h2>Second Script Section</h2>
<?php echo "<p>Output from the second script
section.</p>";?>
</body>
</html>
```


Multiple Script Sections – Mozilla Firefox
File Edit View History Bookmarks Tools Help
http://student200.ucb.sephone.us
Done

Multiple Script Sections

First Script Section

Output from the first script section.

Second Script Section

Output from the second script section.

- PHP code declaration blocks execute on a Web

REFERENCE: PHP Programming with MySQL 2nd Edition

server before a Web page is sent to a client

```
...
</head>
<body>
<h1>Multiple Script Sections</h1>
<h2>First Script Section</h2>
<p>Output from the first script section.</p>
<h2>Second Script Section</h2>
<p>Output from the second script section.</p>
</body>
</html>
```

Case Sensitivity in PHP

- Programming language constructs in PHP are mostly case **insensitive**

```
<?php
echo "<p>Explore <strong>Africa</strong>, <br />";
Echo "<strong>South America</strong>, <br />";
ECHO " and <strong>Australia</strong>!</p>";
?>
```

Adding Comments to a PHP Script

- **Comments** are nonprinting lines placed in code that do not get executed, but provide helpful information, such as:
  - The name of the script
  - Your name and the date you created the program
  - Notes to yourself
  - Instructions to future programmers who might need to modify your work
- **Line comments** hide a single line of code
  - Add // or # before the text
- **Block comments** hide multiple lines of code
  - Add /* to the first line of code
  - And */ after the last character in the code

```
<?php
/*
This line is part of the block comment.
This line is also part of the block comment.
*/
echo "<h1>Comments Example</h1>"; // Line comments can follow
code statements
// This line comment takes up an entire line.
# This is another way of creating a line comment.
/* This is another way of creating
a block comment. */
?>
```

Using Variables and Constants

- The values stored in computer memory are called **variables**

- The values, or data, contained in variables are classified into categories known as **data types**
- The name you assign to a variable is called an **identifier**
- An identifier must begin with a dollar sign ($), may not include a number or underscore as the first character, cannot include spaces, and is case sensitive

Displaying Variables

- To display a variable with the echo statement, pass the variable name to the echo statement without enclosing it in quotation marks:

```
$VotingAge = 18;
echo $VotingAge;
```

- To display both text strings and variables, send them to the echo statement as individual arguments, separated by commas:

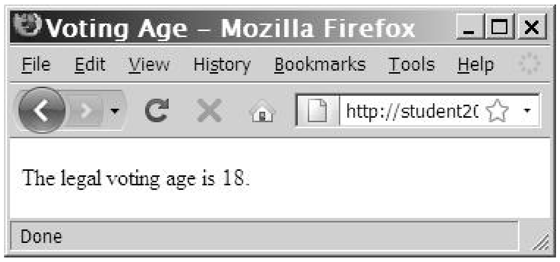
```
echo "<p>The legal voting age is ", $VotingAge, ".</p>";
```

Naming Variables

- The name you assign to a variable is called an identifier
- The following rules and conventions must be followed when naming a variable:
  - Identifiers must begin with a dollar sign ($)
  - Identifiers may contain uppercase and lowercase letters, numbers, or underscores (_). The first character after the dollar sign must be a letter.
  - Identifiers cannot contain spaces
  - Identifiers are case sensitive

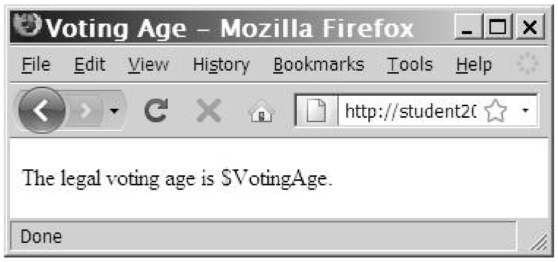
Declaring and Initializing Variables

- Specifying and creating a variable name is called declaring the variable
- Assigning a first value to a variable is called initializing the variable
- In PHP, you must declare and initialize a variable in the same statement:
  $*variable_name* = *value*;


Voting Age – Mozilla Firefox
File Edit View History Bookmarks Tools Help
http://student2(
Done

The legal voting age is 18.

- The output of variable names inside a text string depends on whether the string is surrounded by double or single quotation marks


Voting Age – Mozilla Firefox
File Edit View History Bookmarks Tools Help
http://student2(
Done

The legal voting age is $VotingAge.

Modifying Variables

- You can modify a variable’s value at any point in a script

```
$SalesTotal = 40;
echo "<p>Your sales total is $$SalesTotal</p>";
$SalesTotal = 50;
echo "<p>Your new sales total is $$SalesTotal</p>";
```

REFERENCE: PHP Programming with MySQL 2nd Edition

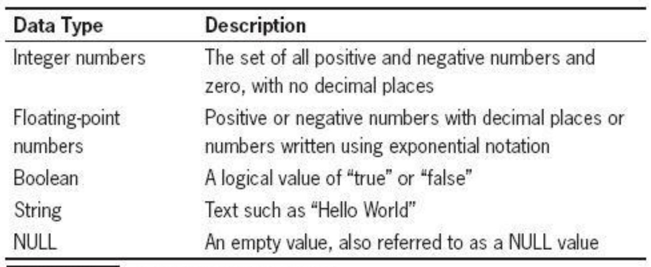
Defining Constants

- A **constant** contains information that does not change during the course of program execution
- Constant names do not begin with a dollar sign ($)
- Constant names use all uppercase letters
- Use the **define()** function to create a constant

  define("CONSTANT_NAME", value);
- The value you pass to the define() function can be a text string, number, or Boolean value

Working with Data Types

- A **data type** is the specific category of information that a variable contains
- Data types that can be assigned only a single value are called **primitive types**



Data Type	Description
Integer numbers	The set of all positive and negative numbers and zero, with no decimal places
Floating-point numbers	Positive or negative numbers with decimal places or numbers written using exponential notation
Boolean	A logical value of "true" or "false"
String	Text such as "Hello World"
NULL	An empty value, also referred to as a NULL value

Table 1-1 Primitive PHP data types

- The PHP language supports:
  - A **resource** data type – a special variable that holds a reference to an external resource such
    as a database or XML file
  - **Reference** or **composite** data types, which contain multiple values or complex types of information
  - Two reference data types: **arrays** and **objects**
- **Strongly typed programming languages** require you to declare the data types of variables
- **Static or strong typing** refers to data types that do not change after they have been declared
- **Loosely typed programming languages** do not require you to declare the data types of variables
- **Dynamic or loose typing** refers to data types that can change after they have been declared

Numeric Data Types

- PHP supports two numeric data types:
  - An **integer** is a positive or negative number and 0 with no decimal places (-250, 2, 100, 10,000)
  - A **floating-point number** is a number that contains decimal places or that is written in exponential notation (-6.16, 3.17, 2.7541)
    - **Exponential notation**, or **scientific notation**, is a shortened format for writing very large numbers or numbers with many decimal places (2.0e11)

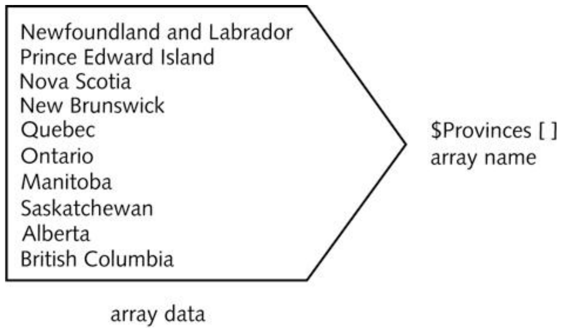
Boolean Values

- A **Boolean value** is a value of TRUE or FALSE
- It decides which part of a program should execute and which part should compare data

- In PHP programming, you can only use TRUE or FALSE Boolean values
- In other programming languages, you can use integers such as 1 = TRUE, 0 = FALSE

Arrays

- An **array** contains a set of data represented by a single variable name



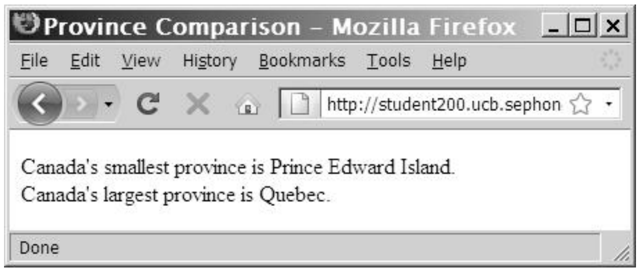
Declaring and Initializing Indexed Arrays

- An **element** refers to each piece of data that is stored within an array
- An **index** is an element's numeric position within the array
  - By default, indexes begin with the number zero (0)
  - An element is referenced by enclosing its index in brackets at the end of the array name: $Provinces[1]
- The array() construct syntax is:
  ***$array_name* = array(*values*);**

```
$Provinces = array(
   "Newfoundland and Labrador",
   "Prince Edward Island",
   "Nova Scotia",
   "New Brunswick",
   "Quebec",
   "Ontario",
   "Manitoba",
   "Saskatchewan",
   "Alberta",
   "British Columbia"
   );
```

- Array name and brackets syntax is:
  ***$array_name*[ ]**

```
$Provinces[] = "Newfoundland and Labrador";
$Provinces[] = "Prince Edward Island";
$Provinces[] = "Nova Scotia";
$Provinces[] = "New Brunswick";
$Provinces[] = "Quebec";
$Provinces[] = "Ontario";
$Provinces[] = "Manitoba";
$Provinces[] = "Saskatchewan";
$Provinces[] = "Alberta";
$Provinces[] = "British Columbia";
```


Province Comparison – Mozilla Firefox
File Edit View History Bookmarks Tools Help
http://student200.ucb.sephon
Done

```
Canada's smallest province is Prince Edward Island.
Canada's largest province is Quebec.
```

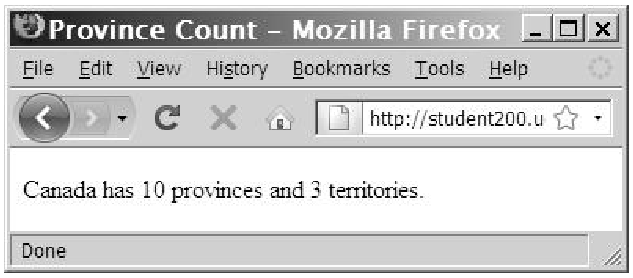
Accessing Element Information

```
echo "<p>Canada's smallest province is $Provinces[1].<br />";
echo "Canada's largest province is $Provinces[4].</p>";
```

REFERENCE: PHP Programming with MySQL 2nd Edition

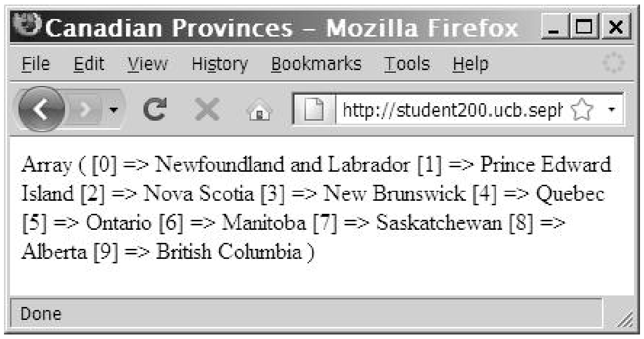
- Use the **count()** function to find the total number of elements in an array

```
        $Provinces = array("Newfoundland and Labrador", "Prince Edward
Island", "Nova Scotia", "New Brunswick", "Quebec", "Ontario", " Manitoba", "Saskatchewan", "Alberta", "British Columbia");
        $Territories = array("Nunavut", "Northwest Territories", "Yukon
        Territory");
        echo "<p>Canada has ", count($Provinces), " provinces and ",
  count($Territories), " territories.</p>";
```


Province Count – Mozilla Firefox
File Edit View History Bookmarks Tools Help
http://student200.u
Done

```
Canada has 10 provinces and 3 territories.
```

- Use the print_r(), var_dump() or var_export() functions to display or return information about variables
  - The print_r() function displays the index and value of each element in an array
  - The var_dump() function displays the index, value, data type and number of characters in the value
  - The var_export() function is similar to var_dump() function except it returns valid PHP code


Canadian Provinces – Mozilla Firefox
File Edit View History Bookmarks Tools Help
http://student200.ucb.seph
Done

```
Array ( [0] => Newfoundland and Labrador [1] => Prince Edward
Island [2] => Nova Scotia [3] => New Brunswick [4] => Quebec
[5] => Ontario [6] => Manitoba [7] => Saskatchewan [8] =>
Alberta [9] => British Columbia )
```

Modifying Elements

- To modify an array element. include the index for an individual element of the array:

```
        $HospitalDepts = array(
                "Anesthesia",           // first element(0)
                "Molecular Biology",    // second element (1)
                "Neurology");           // third element (2)
```

To change the first array element in the $HospitalDepts[] array from “Anesthesia” to “Anesthesiology” use:

```
        $HospitalDepts[0] = "Anesthesiology";
```

Avoiding Assignment Notation Pitfalls

- Assigns the string “Hello” to a variable named $list

REFERENCE: PHP Programming with MySQL 2nd Edition

$list = "Hello";

- Assigns the string “Hello” to a new element appended to the end of the $list array
  $list[] = "Hello";
- Replaces the value stored in the first element (index 0) of the $list array with the string “Hello”
  $list[0] = "Hello";

Building Expressions

- An **expression** is a literal value or variable that can be evaluated by the PHP scripting engine to produce a result
- **Operands** are variables and literals contained in an expression
- A **literal** is a static value such as a literal string or a number
- **Operators** are symbols (+) (*) that are used in expressions to manipulate operands

Type	Description
Array	Performs operations on arrays
Arithmetic	Performs mathematical calculations
Assignment	Assigns values to variables
Comparison	Compares operands and returns a Boolean value
Logical	Performs Boolean operations on Boolean operands
Special	Performs various tasks; these operators do not fit within other operator categories
String	Performs operations on strings

Table 1-2 PHP operator types

- A **binary operator** requires an operand before and after the operator
  - $MyNumber = 100;
- A **unary operator** requires a single operand either before or after the operator

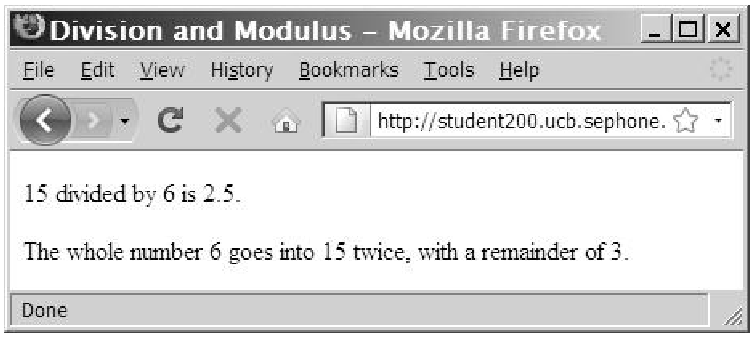
Arithmetic Operators

- **Arithmetic operators** are used in PHP to perform mathematical calculations (+ - x ÷)

Symbol	Operation	Description
+	Addition	Adds two operands
-	Subtraction	Subtracts the right operand from the left operand
*	Multiplication	Multiplies two operands
/	Division	Divides the left operand by the right operand
%	Modulus	Divides the left operand by the right operand and returns the remainder

Table 1-3 PHP arithmetic binary operators

$DivisionResult = 15 / 6;
$ModulusResult = 15 % 6;
echo "<p>15 divided by 6 is $DivisionResult.</p>"; // prints '2.5'
echo "The whole number 6 goes into 15 twice, with a remainder of $ModulusResult.</p>"; // prints '3'


Division and Modulus – Mozilla Firefox
File Edit View History Bookmarks Tools Help
http://student200.ucb.sephone.
Done

15 divided by 6 is 2.5.

The whole number 6 goes into 15 twice, with a remainder of 3.

Arithmetic Unary Operators

- The increment (++) and decrement (--

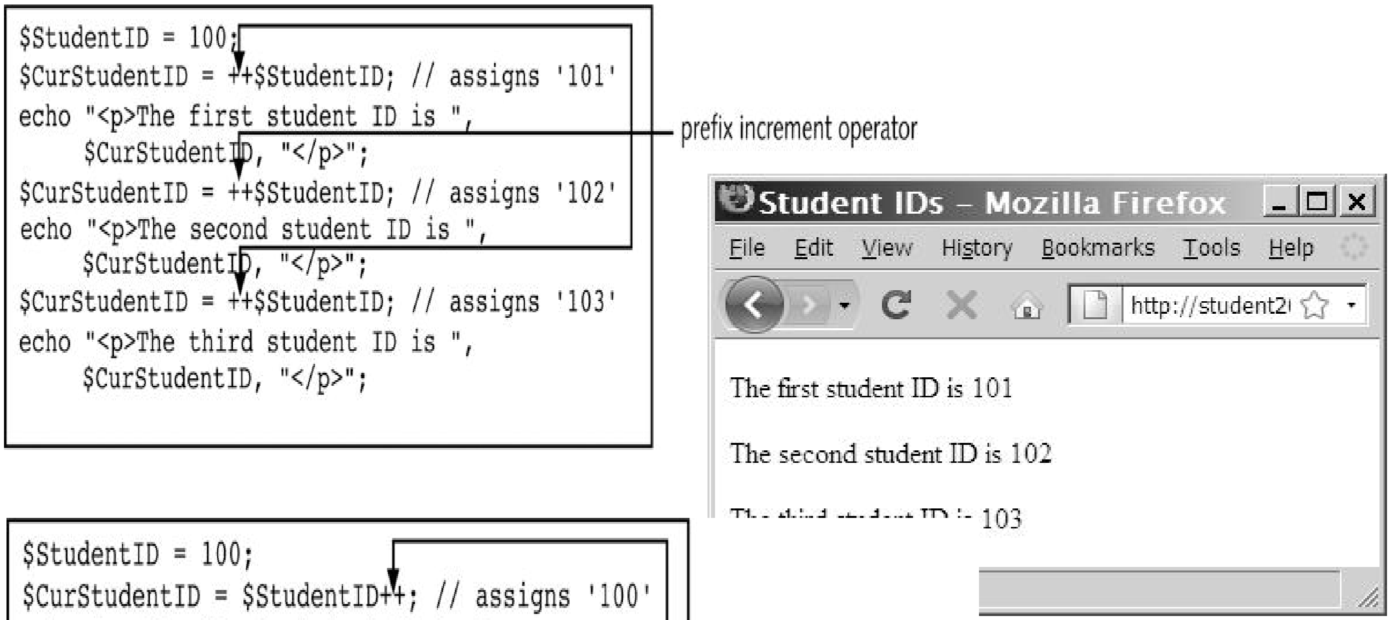
Symbol	Operation	Description
++	Increment	Increases an operand by a value of 1
--	Decrement	Decreases an operand by a value of 1

Table 1-4 PHP arithmetic unary operators

REFERENCE: PHP Programming with MySQL 2nd

unary operators can be used as prefix or postfix operators

- A **prefix operator** is placed before a variable
- A **postfix operator** is placed after a variable



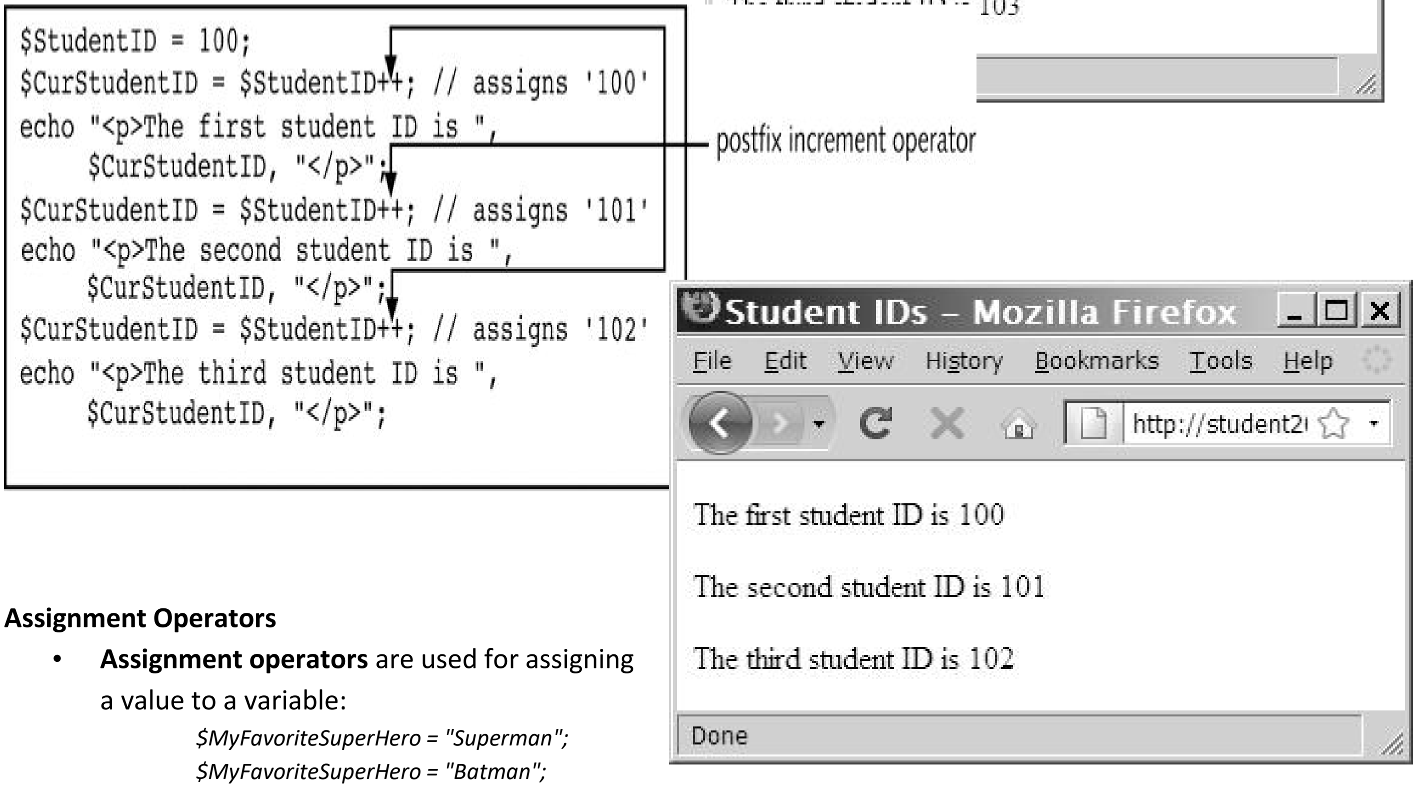
```
$StudentID = 100;
$CurStudentID = ++$StudentID; // assigns '101'
echo "<p>The first student ID is ",
    $CurStudentID, "</p>";
$CurStudentID = ++$StudentID; // assigns '102'
echo "<p>The second student ID is ",
    $CurStudentID, "</p>";
$CurStudentID = ++$StudentID; // assigns '103'
echo "<p>The third student ID is ",
    $CurStudentID, "</p>";
```

prefix increment operator
Student IDs – Mozilla Firefox
File Edit View History Bookmarks Tools Help
http://student2

```
The first student ID is 101

The second student ID is 102

The third student ID is 103
```



```
$StudentID = 100;
$CurStudentID = $StudentID++; // assigns '100'
echo "<p>The first student ID is ",
    $CurStudentID, "</p>";
$CurStudentID = $StudentID++; // assigns '101'
echo "<p>The second student ID is ",
    $CurStudentID, "</p>";
$CurStudentID = $StudentID++; // assigns '102'
echo "<p>The third student ID is ",
    $CurStudentID, "</p>";
```

postfix increment operator
Student IDs – Mozilla Firefox
File Edit View History Bookmarks Tools Help
http://student2
Done

```
The first student ID is 100

The second student ID is 101

The third student ID is 102
```

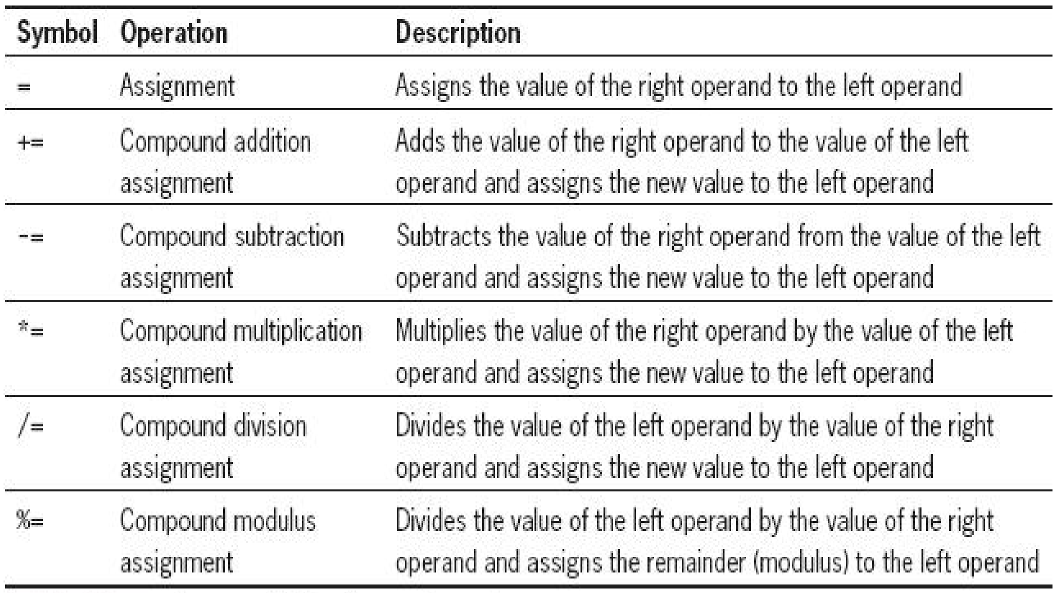
Assignment Operators

- **Assignment operators** are used for assigning a value to a variable:

  $MyFavoriteSuperHero = "Superman";
  $MyFavoriteSuperHero = "Batman";

- **Compound assignment operators** perform mathematical calculations on variables and literal values in an expression, and then assign a new value to the left operand

REFERENCE: PHP Programming with MySQL 2nd Edition

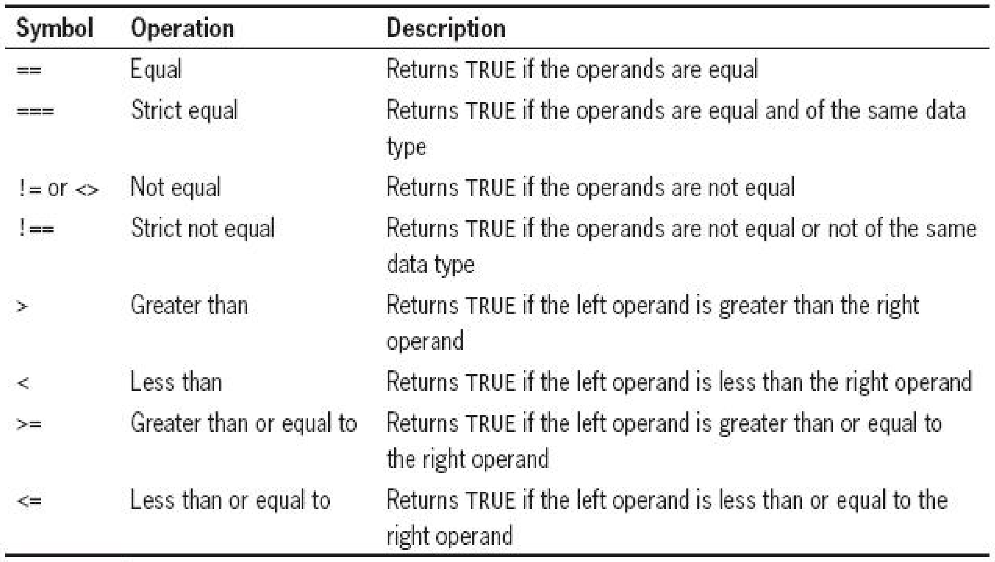


Symbol	Operation	Description
=	Assignment	Assigns the value of the right operand to the left operand
+=	Compound addition assignment	Adds the value of the right operand to the value of the left operand and assigns the new value to the left operand
-=	Compound subtraction assignment	Subtracts the value of the right operand from the value of the left operand and assigns the new value to the left operand
*=	Compound multiplication assignment	Multiplies the value of the right operand by the value of the left operand and assigns the new value to the left operand
/=	Compound division assignment	Divides the value of the left operand by the value of the right operand and assigns the new value to the left operand
%=	Compound modulus assignment	Divides the value of the left operand by the value of the right operand and assigns the remainder (modulus) to the left operand

Table 1-5 Common PHP assignment operators

Comparison and Conditional Operators

- **Comparison operators** are used to compare two operands and determine how one operand compares to another
- A Boolean value of TRUE or FALSE is returned after two operands are compared
- The comparison operator *compares* values, whereas the assignment operator *assigns* values
- Comparison operators are used with **conditional statements** and **looping statements**



Symbol	Operation	Description
==	Equal	Returns TRUE if the operands are equal
===	Strict equal	Returns TRUE if the operands are equal and of the same data type
!= or <>	Not equal	Returns TRUE if the operands are not equal
!==	Strict not equal	Returns TRUE if the operands are not equal or not of the same data type
>	Greater than	Returns TRUE if the left operand is greater than the right operand
<	Less than	Returns TRUE if the left operand is less than the right operand
>=	Greater than or equal to	Returns TRUE if the left operand is greater than or equal to the right operand
<=	Less than or equal to	Returns TRUE if the left operand is less than or equal to the right operand

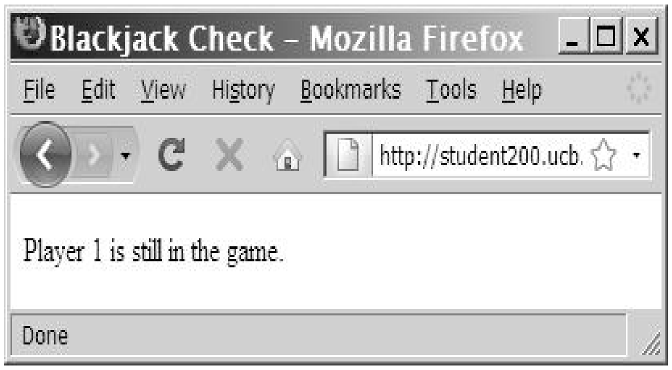
Table 1-6 PHP comparison operators

- The **conditional operator** executes one of two expressions, based on the results of a conditional expression
- The syntax for the conditional operator is:

conditional expression ? expression1 : expression2;

- If the conditional expression evaluates to TRUE, *expression1* executes
- If the conditional expression evaluates to FALSE, *expression2* executes

```
$BlackjackPlayer1 = 20;
($BlackjackPlayer1 <= 21) ? $Result =
   "Player 1 is still in the game. " : $Result =
   "Player 1 is out of the action.";
echo "<p>", $Result, "</p>";
```


Blackjack Check - Mozilla Firefox
File Edit View History Bookmarks Tools Help
http://student200.ucb.
Done

Player 1 is still in the game.

Logical Operators

- **Logical operators** are used for comparing two Boolean operands for equality
- A Boolean value of TRUE or FALSE is returned after two operands are compared

Symbol	Operation	Description
&& or AND	Logical And	Returns TRUE if both the left operand and right operand return a value of TRUE; otherwise, it returns a value of FALSE
|| or OR	Logical Or	Returns TRUE if either the left operand or right operand returns a value of TRUE; otherwise (neither operand returns a value of TRUE), it returns a value of FALSE
XOR	Logical Exclusive Or	Returns TRUE if only one of the left operand or right operand returns a value of TRUE; otherwise (neither operand returns a value of TRUE or both operands return a value of TRUE), it returns a value of FALSE
!	Logical Not	Returns TRUE if an expression is FALSE and returns FALSE if an expression is TRUE

Table 1-7 PHP logical operators

Symbol	Operation
[and]	Accesses an element of an array
=>	Specifies the index or key of an array element
,	Separates arguments in a list
? and :	Executes one of two expressions based on the results of a conditional expression
instanceof	Returns TRUE if an object is of a specified object type
@	Suppresses any errors that might be generated by an expression to which it is prepended (or placed before)
(int), (integer), (bool), (boolean), (double), (string), (array), (object)	Casts (or transforms) a variable of one data type into a variable of another data type

Table 1-8 PHP special operators

Type Casting

- **Casting** or **type casting** copies the value contained in a variable of one data type into a variable of another data type
- The PHP syntax for casting variables is:
  $NewVariable = (new_type) $OldVariable;
- (*new_type*) refers to the type-casting operator representing the type to which you want to cast the variable
- Returns one of the following strings, depending on the data type:
  - Boolean
  - Integer
  - Double
  - String
  - Array
  - Object
  - Resource
  - NULL

Symbol	Operator	Associativity
new clone	New object—highest precedence	None
[]	Array elements	Right to left
++ --	Increment/Decrement	Right to left
(int) (double) (string) (array) (object)	Cast	Right to left
@	Suppress errors	Right to left
instanceof	Types	None
!	Logical Not	Right to left
* / %	Multiplication/division/modulus	Left to right
+ - .	Addition/subtraction/string concatenation	Left to right
< <= > >= <>	Comparison	None
== != === !==	Equality	None
&&	Logical And	Left to right
||	Logical Or	Left to right
?:	Conditional	Left to right
= += -= *= /= %= .=	Assignment	Right to left
AND	Logical And	Left to right
XOR	Logical Exclusive Or	Left to right
OR	Logical Or	Left to right
,	List separator—lowest precedence	Left to right

Table 1-9 Operator precedence in PHP

- Unknown type

Understanding Operator Precedence

- **Operator precedence** refers to the order in which operations in an expression are evaluated
- **Associativity** is the order in which operators of equal precedence execute
- Associativity is evaluated on a left-to-right or a right-to-left basis

Summary

- JavaScript and PHP are both referred to as embedded languages because code for both languages is embedded within a Web page (either an HTML or XHTML document)
- You write PHP scripts within code declaration blocks, which are separate sections within a Web page that are interpreted by the scripting engine
- The individual lines of code that make up a PHP script are called statements
- The term, **function**, refers to a procedure (or individual statements grouped into a logical unit) that performs a specific task
- Comments are lines that you place in code to contain various types of remarks, including the name of the script, your name and the date you created the program, notes to yourself, or instructions to future programmers who might need to modify your work
    - Comments do not display in the browser
- The values a program stores in computer memory are commonly called **variables**
- The name you assign to a variable is called an **identifier**
- A **constant** contains information that cannot change during the course of program execution
- A **data type** is the specific category of information that a variable contains
- PHP is a *loosely-typed* programming language
- An **integer** is a positive or negative number or zero, with no decimal places
- A floating-point number contains decimal places or is written in exponential notation
- A **Boolean** value is a logical value of TRUE or FALSE
- An **array** contains a set of data represented by a single variable name
- An **expression** is a single literal value or variable or a combination of literal values, variables, operators, and other expressions that can be evaluated by the PHP scripting engine to produce a result
- **Operands** are variables and literals contained in an expression. A literal is a value such as a string or a number.
- **Operators** are symbols used in expressions to manipulate operands, such as the addition operator (+) and multiplication operator (*)
- A **binary operator** requires an operand before and after the operator
- A **unary operator** requires a single operand either before or after the operator
- **Arithmetic operators** are used in the PHP scripting engine to perform mathematical calculations, such as addition, subtraction, multiplication, and division
- **Assignment operators** are used for assigning a value to a variable
- **Comparison operators** are used to determine how one operand compares with another

- The **conditional operator** executes one of two expressions, based on the results of a conditional expression
- **Logical operators** are used to perform operations on Boolean operands
- **Casting** or type casting creates an equivalent value in a specific data type for a given value
- **Operator precedence** is the order in which operations in an expression are evaluated

REFERENCE: PHP Programming with MySQL 2nd Edition