jesuitas
Logroño

SERVER-SIDE WEB PROGRAMMING UNIT2: SERVER CODE GENERATION

CFGS - DAW 2

Index

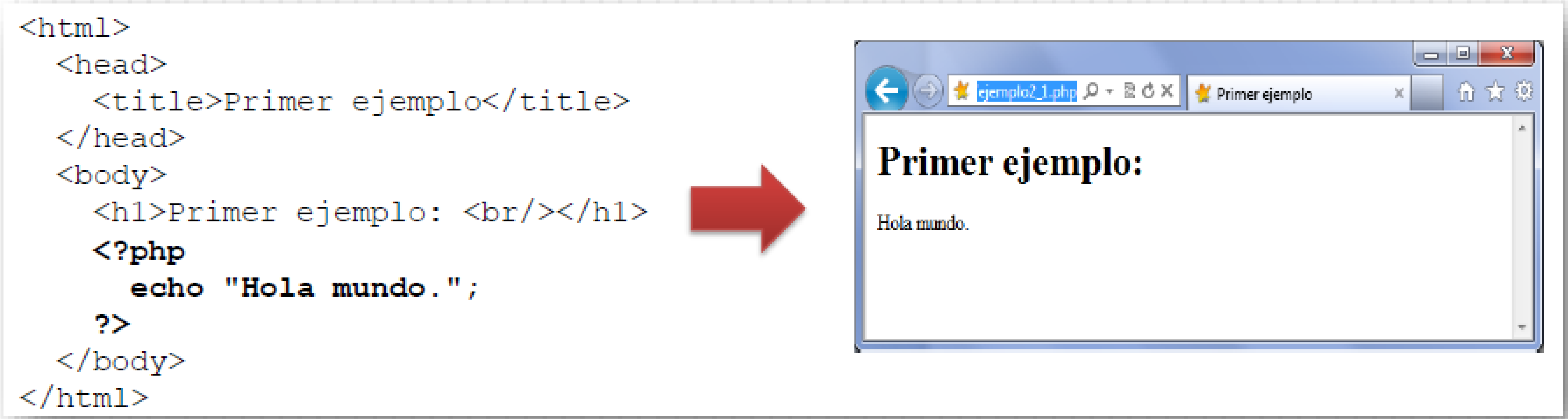
1. Obtaining the code sent to the client

- PHP (HypertextPreprocessor):
  - Scripting language.
  - General purpose and open source
  - Specially designed for the web development.
  - Embedded in HTML code.

2. PHP-Code-Tags

- Each language that can be used to insert code into a web page uses a series of **tags** to delimit fragments of code that must be processed by the web server.
- The component of the server responsible for processing will ignore the HTML code that is outside those tags.

2. PHP-Code-Tags



```
<html>
  <head>
    <title>Primer ejemplo</title>
  </head>
  <body>
    <h1>Primer ejemplo: <br/></h1>
    <?php
      echo "Hola mundo.";
    ?>
  </body>
</html>
```

Primer ejemplo:

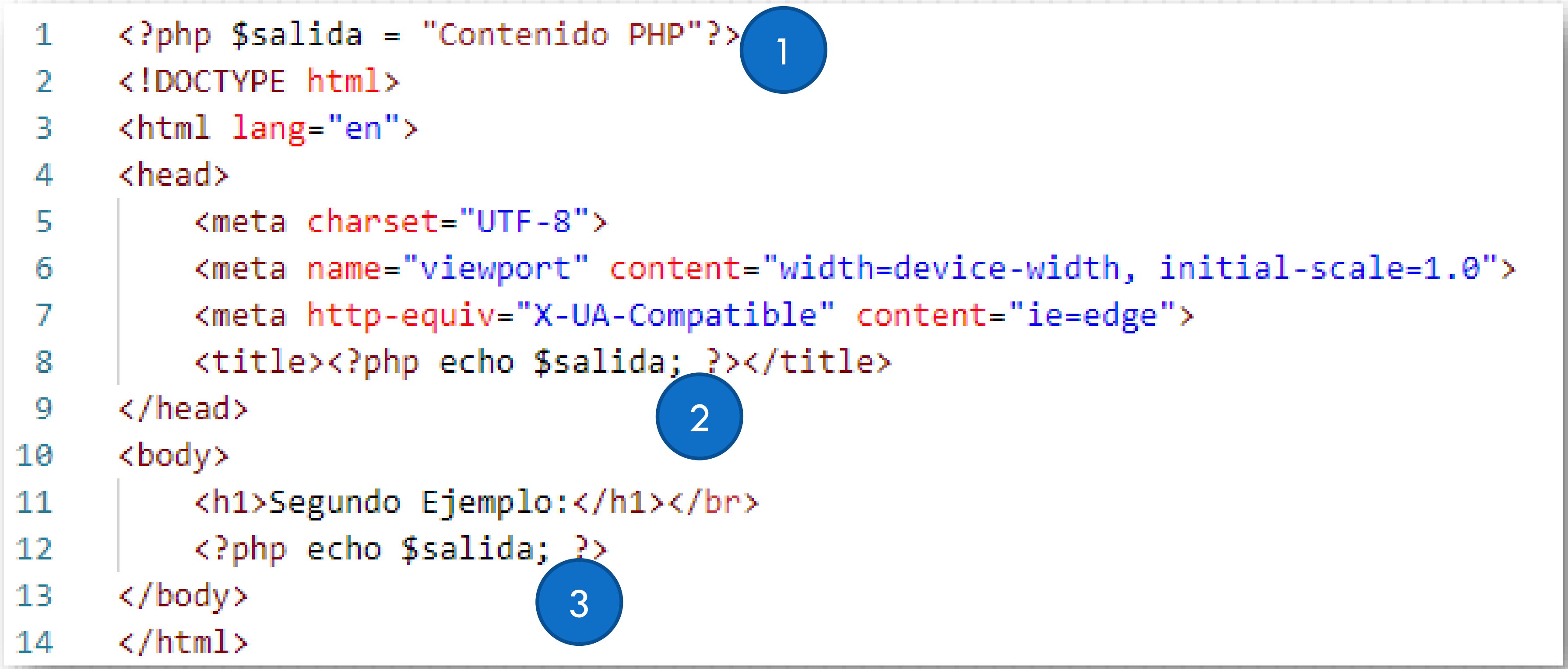
Hola mundo.

2. PHP-Code-Tags

- <u>Basic features of scripting languages:</u>

1. All script begins (and ends) with a start tag and an ending.
2. Blanks written within the embedded code have no effect.
3. The server code embedded into HTML consists of a set of statements that should be clearly separated.
4. Embedded scripts can be located anywhere on the web resource executed.
5. The number of scripts that we can have within an HTML file is undefined.
6. When an embedded code is executed, the entire script is replaced by the result of the execution, including start and end tags.

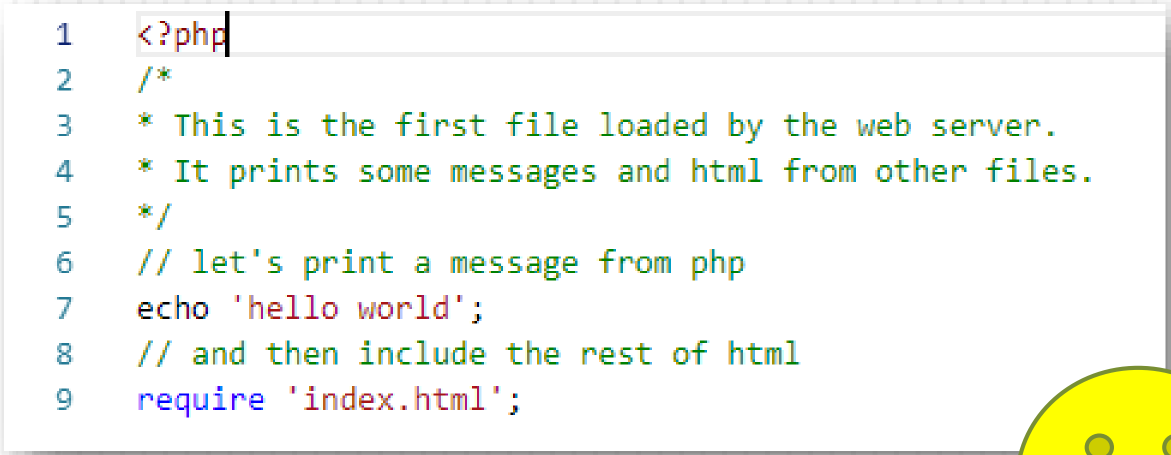
2. PHP-Code-Tags



```
<?php $salida = "Contenido PHP"?>
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <meta http-equiv="X-UA-Compatible" content="ie=edge">
    <title><?php echo $salida; ?></title>
</head>
<body>
    <h1>Segundo Ejemplo:</h1></br>
    <?php echo $salida; ?>
</body>
</html>
```

2. PHP-Code-Tags

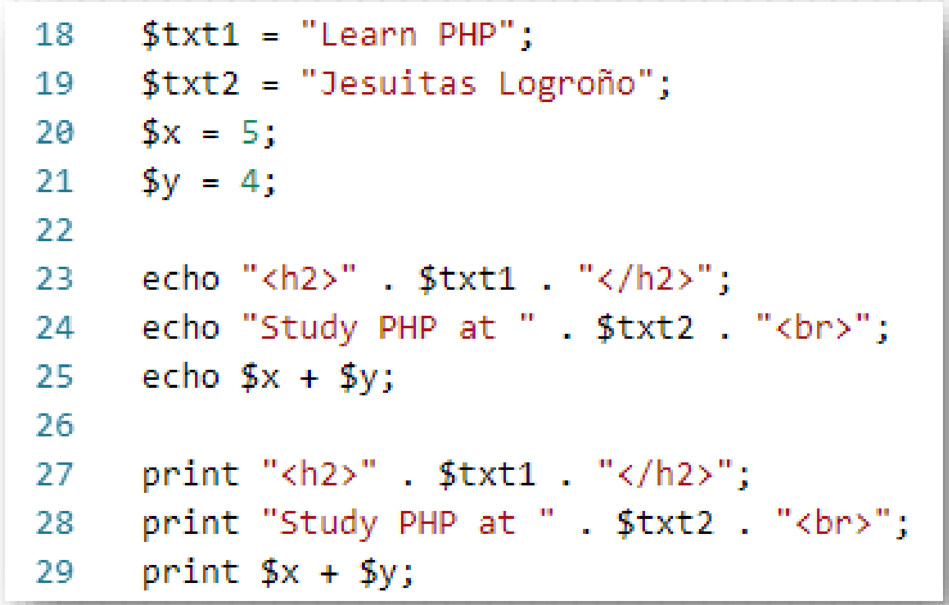
- <u>**Comments and more:**</u>



```php
<?php
/*
* This is the first file loaded by the web server.
* It prints some messages and html from other files.
*/
// let's print a message from php
echo 'hello world';
// and then include the rest of html
require 'index.html';
```

2. PHP-Code-Tags

- **Echo & Print:**



```php
$txt1 = "Learn PHP";
$txt2 = "Jesuitas Logroño";
$x = 5;
$y = 4;

echo "<h2>" . $txt1 . "</h2>";
echo "Study PHP at " . $txt2 . "<br>";
echo $x + $y;

print "<h2>" . $txt1 . "</h2>";
print "Study PHP at " . $txt2 . "<br>";
print $x + $y;
```

2. PHP-Code-Tags

- Case sensitivity:

1. The names of **user-defined classes and functions**, as well as **built-in constructs and keywords** such as echo, while, class, etc., are case-insensitive. Thus, these three lines are equivalent:
   - echo("hello, world");
   - ECHO("hello, world");
   - EcHo("hello, world");
2. **Variables**, on the other hand, are case-sensitive. That is, $name, $NAME, and $NaME are three different variables.

3. Variables

- Variables hold the data that your program manipulates while it runs, such as user information that you've loaded from a database or entries that have been typed into an HTML form.
- In PHP, variables are denoted by a $ followed by the variable's name.

3. Variables

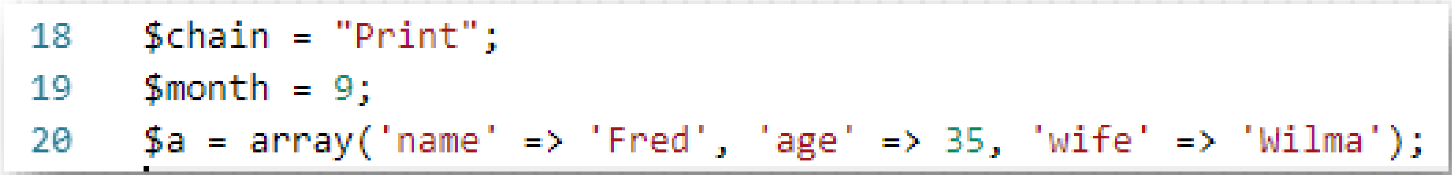
- Variable names: Variable names always begin with a dollar sign ($) and are case-sensitive
    - $bill
    - $head_count
    - $MaximumForce
    - $I_HEART_PHP
    - $_underscore
    - $_int
- Here are some illegal variable names:
    - $not valid
    - $|
    - $3wa
- These variables are all different due to case sensitivity:
    - $hot_stuff $Hot_stuff $hot_Stuff $HOT_STUFF

3. Variables

Find a list of PHP core language Keywords.

3. Variables

- A variable may hold a value of any type. PHP provides eight types of values, or data types:
  - Four are scalar (single-value) types: <u>integers, floating-point numbers, strings, and Booleans</u>.
  - Two are compound (collection) types: <u>arrays</u> and <u>objects</u>.
  - NULL
- There is no explicit syntax for declaring variables in PHP:



```php
$chain = "Print";
$month = 9;
$a = array('name' => 'Fred', 'age' => 35, 'wife' => 'Wilma');
```

4. Variables

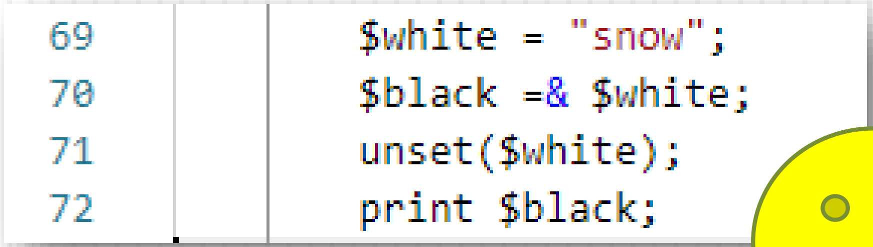
- <u>Variable References</u>:
  - In PHP, references are how you create variable aliases. To make $black an alias for the variable $white, use:

    $black =& $white;

  - The old value of $black, if any, is lost. Instead, $black is now another name for the value that is stored in $white.

4. Variables

- Variable References:
  - After the assignment, the two variables are alternate names for the same value. Unsetting a variable that is aliased does not affect other names for that variable's value, however:

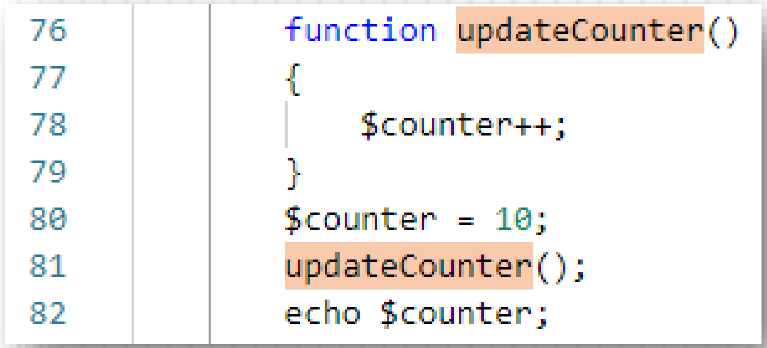


```
$white = "snow";
$black =& $white;
unset($white);
print $black;
```

3. Variables

- <u>Variable Scope</u>:
  - The scope of a variable, which is controlled by the location of the variable's declaration, determines those parts of the program that can access it.
  - 4 types:

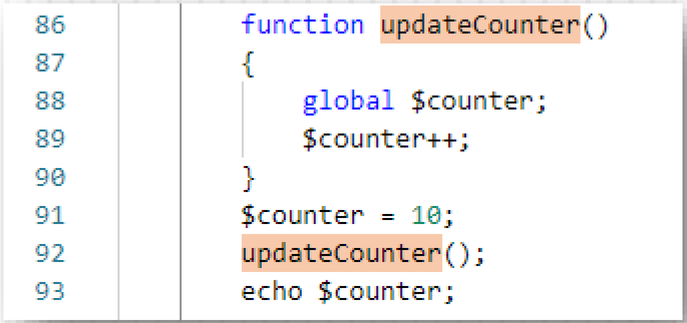
1. Local:



```php
function updateCounter()
{
    $counter++;
}
$counter = 10;
updateCounter();
echo $counter;
```

3. Variables

2. Global: Variables declared outside a function are global. That is, they can be accessed from any part of the program. However, by default, they are not available inside functions. To allow a function to access a global variable, you can use the global keyword inside the function to declare the variable within the function.

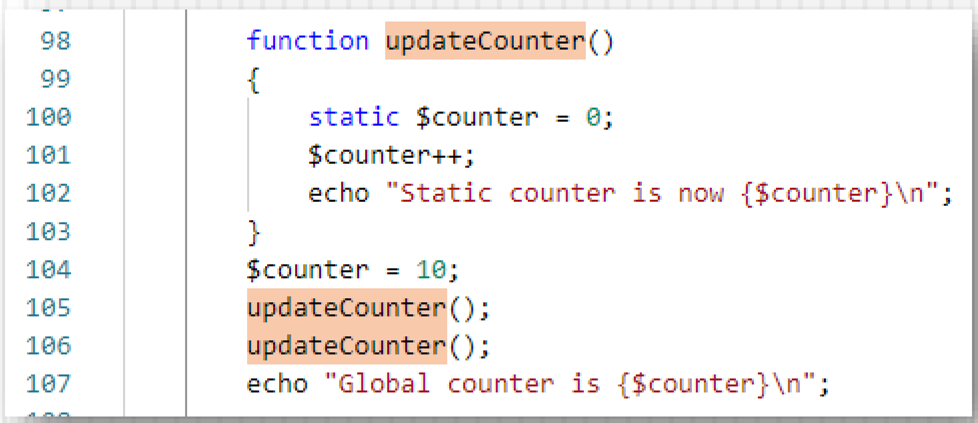


```
function updateCounter()
{
    global $counter;
    $counter++;
}
$counter = 10;
updateCounter();
echo $counter;
```

There is anoher way of accessing global variables… do you know how?

3. Variables

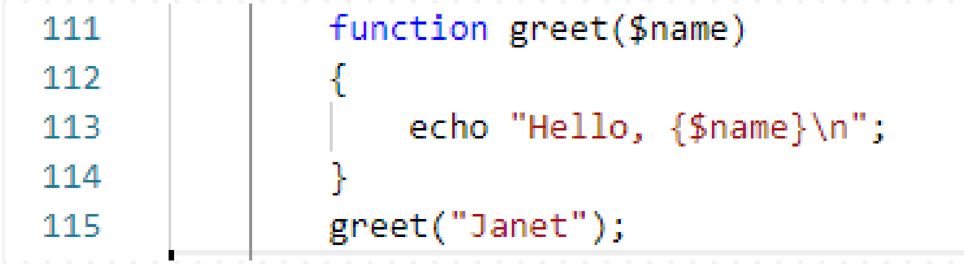
3. Static variable: A static variable retains its value between calls to a function but is visible only within that function.



```php
function updateCounter()
{
    static $counter = 0;
    $counter++;
    echo "Static counter is now {$counter}\n";
}
$counter = 10;
updateCounter();
updateCounter();
echo "Global counter is {$counter}\n";
```

3. Variables

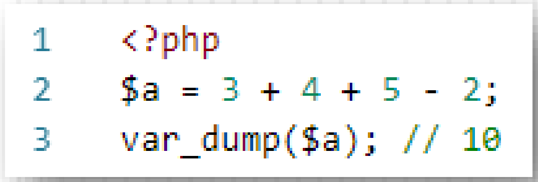
4. Function parameter:



```
function greet($name)
{
    echo "Hello, {$name}\n";
}
greet("Janet");
```

3. Variables

- Operators: The ones you already know from other languages (See manual when needed):



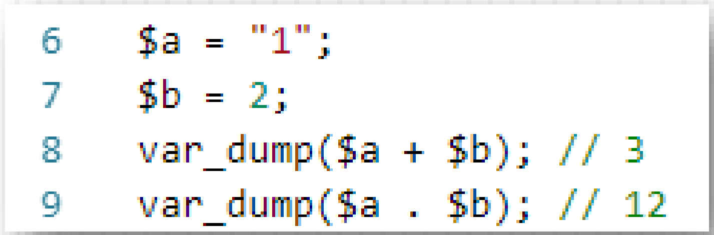
```php
<?php
$a = 3 + 4 + 5 - 2;
var_dump($a); // 10
```

  - Precedence: The order in which operators in an expression are evaluated depends on their relative precedence.

Check this link.

3. Variables

- <u>String chaining:</u>



```
$a = "1";
$b = 2;
var_dump($a + $b); // 3
var_dump($a . $b); // 12
```

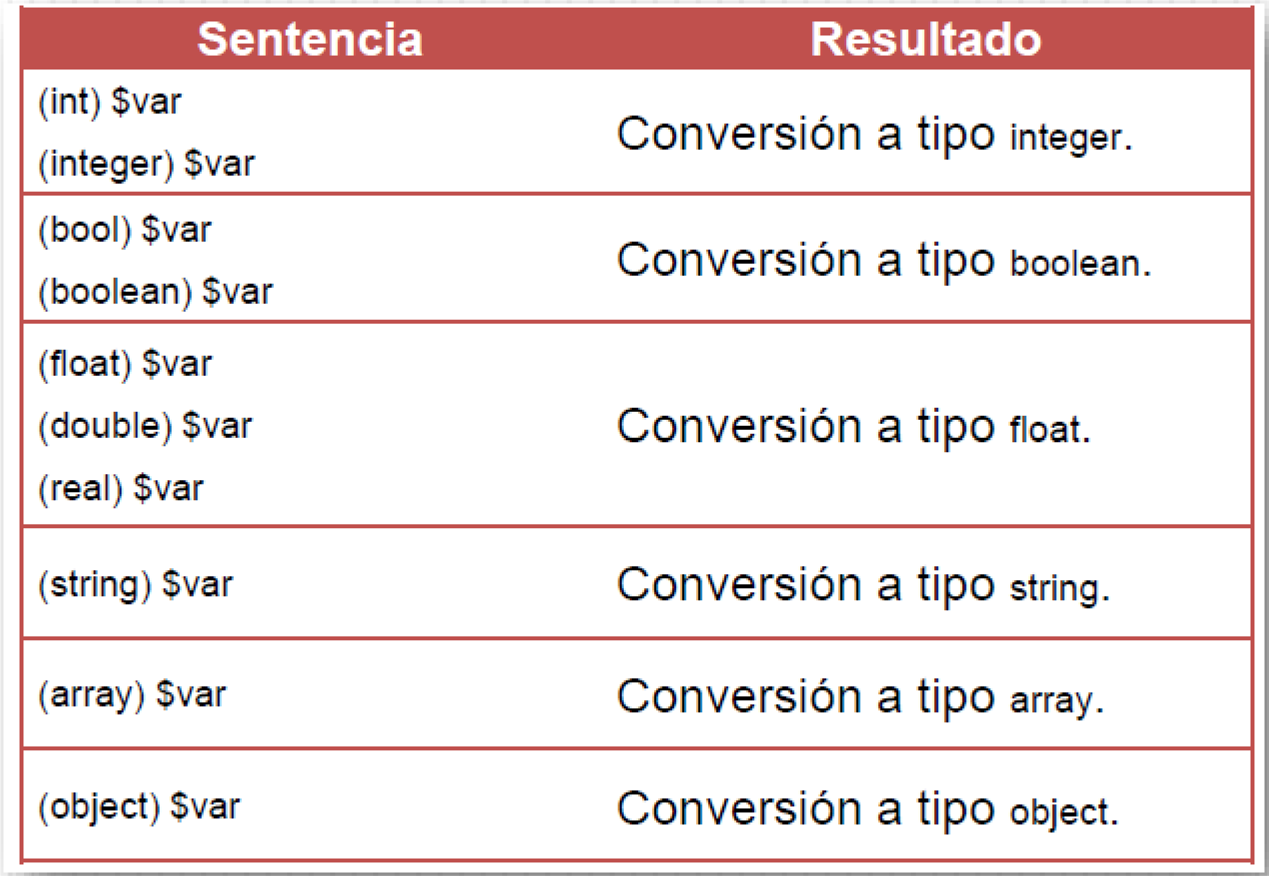
3. Variables

- <u>Conversions between data types:</u>
  - Some interesting functions:
    - *string* **strval(mixed variable)**
    - *integer* **intval(mixed variable)**
    - *float* **floatval(mixed variable)**

3. Variables

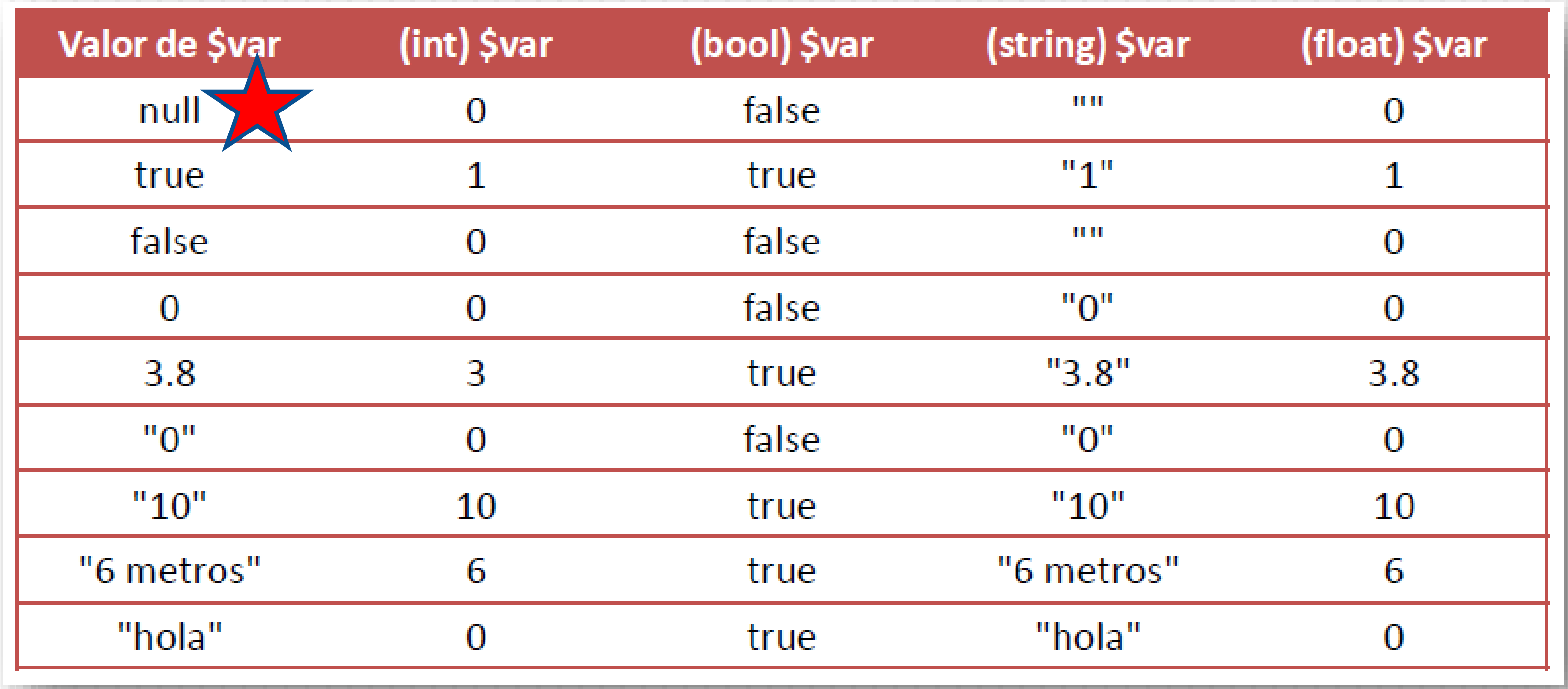
- <u>Conversions between data types:</u>
  1. Implicit conversions:
     - Binary arithmetic operators:
       - If one operand is an integer and the other is a floating-point number, then the first is also evaluated as a float. If one is a string and the other an integer, PHP converts the string to an integer before evaluating both operands as integers.
     - Boolean expressions and expression operators:
       - For those places where an expression must be evaluated as a Boolean, PHP converts the result of the expression to a Boolean before continuing.
     - Certain methods that expect strings:
       - Certain methods and operators echo, print, or the string concatenation operator (.) expect their arguments or operands to be strings. In these cases, PHP tries its best to convert non-string variables to strings.

3. Variables



Sentencia	Resultado
(int) $var (integer) $var	Conversión a tipo integer.
(bool) $var (boolean) $var	Conversión a tipo boolean.
(float) $var (double) $var (real) $var	Conversión a tipo float.
(string) $var	Conversión a tipo string.
(array) $var	Conversión a tipo array.
(object) $var	Conversión a tipo object.

3. Variables



Valor de $var	(int) $var	(bool) $var	(string) $var	(float) $var
null	0	false	""	0
true	1	true	"1"	1
false	0	false	""	0
0	0	false	"0"	0
3.8	3	true	"3.8"	3.8
"0"	0	false	"0"	0
"10"	10	true	"10"	10
"6 metros"	6	true	"6 metros"	6
"hola"	0	true	"hola"	0

3. Variables

2. Automatic conversions: When they are combined in the same expression two variables that initially have different types or when a variable is passed as an argument to a function that expects a different data type.

What would happen in this cases:

- $var= "20" + 15;
- $var= 20 . " years";
- $var= 20 + " years";
- $var= 40 + "2 reasons";

3. Variables

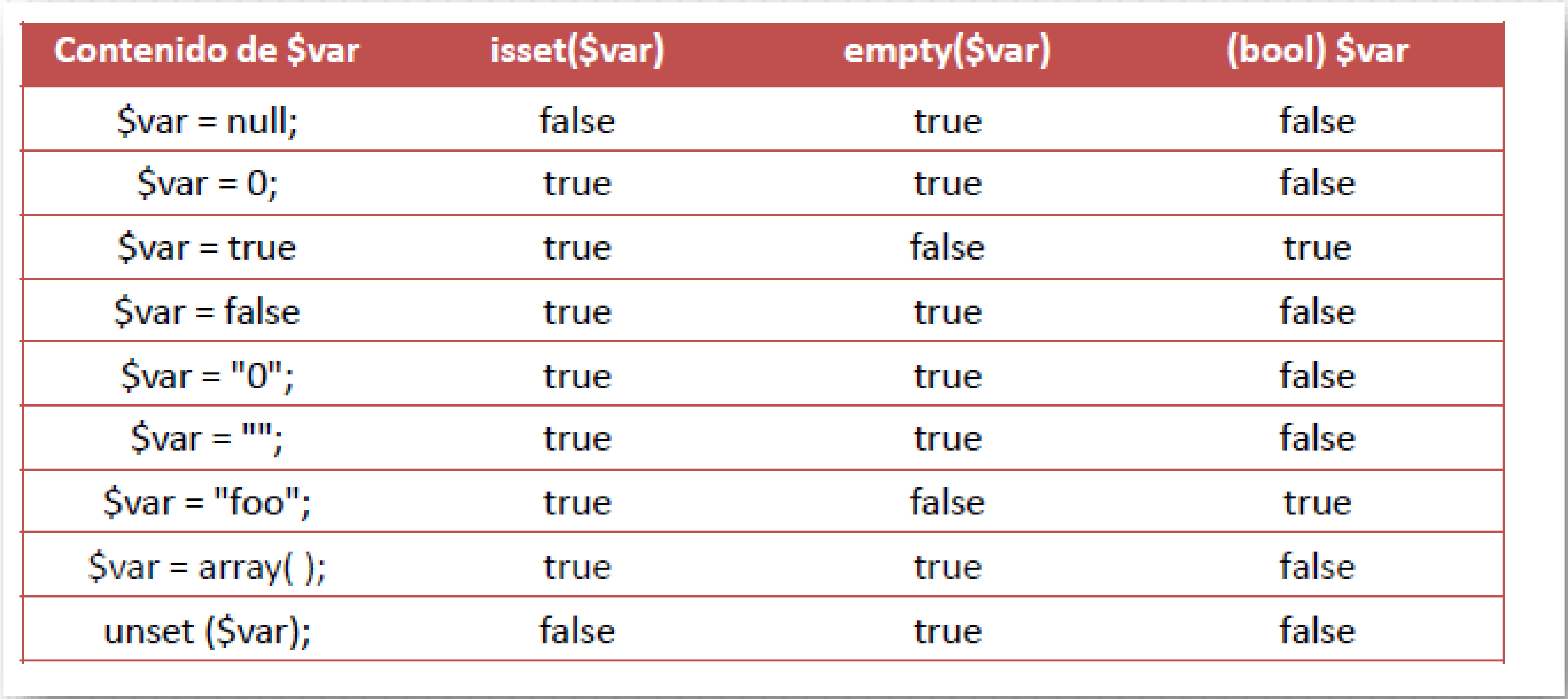
- Checking the type of a variable:
  - *boolean is_int(mixed variable)*.
  - *boolean is_float(mixed variable)*.
  - *boolean is_bool(mixed variable)*.
  - *boolean is_string(mixed variable)*.
  - *boolean is_array(mixed variable)*.
  - *boolean is_object(mixed variable)*.

There are also two functions that are very useful for any web developer: ***print_r()* and *var_dump()***. What are they used for and explain an example.

3. Variables

- <u>State of a variable:</u>
  - Functions to check the status of a variable:
    - boolean **isset**(mixed var): checks if a variable has been defined and is not null.
    - boolean **empty**(mixed var): checks whether that variable has a value.
  - Function to pass a variable undefined state:
    - unset().

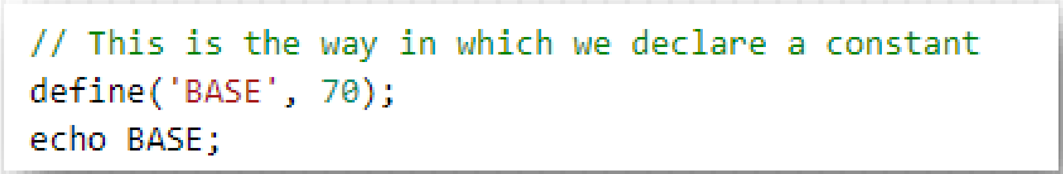
3. Variables



Contenido de $var	isset($var)	empty($var)	(bool) $var
$var = null;	false	true	false
$var = 0;	true	true	false
$var = true	true	false	true
$var = false	true	true	false
$var = "0";	true	true	false
$var = "";	true	true	false
$var = "foo";	true	false	true
$var = array( );	true	true	false
unset ($var);	false	true	false

4. Constants

- A constant is an identifier for a simple value; only scalar values (boolean, integer, double, and string) can be constants.
- Once set, the value of a constant cannot change.
- Constants are referred to by their identifiers and are set using the define() function:



```
// This is the way in which we declare a constant
define('BASE', 70);
echo BASE;
```