Date
08/02/2011

bool is_string()

Find whether the type of variable is string.

Example:

```
var_dump(is_string(false)); // bool(false)
                (true)      // bool(false)
                (null)      // bool(false)
                ('abc')     // bool (true)
                ('23')      // bool(true)
                (23)        // bool(false)
                ('23.5')    // bool(true)
                (23.5)      // bool(false)
                ('')        // bool(true)
                (' ')       // bool(true)
                ('0')       // bool(true)
                (0)         // bool(false)
```

Variables are case sensitive in PHP

Yes, either It is windows or Linux.

Example: $x & $X are two different variables

```
$x=25;
echo $X;
```

Output:

```
Notice: Undefined variable: X
```

Date
09/02/2017

bool isset()

Determine If a variable is set and is not NULL.

Example:

```
$a ='';
 $b = "test";
  $c = NULL;
var_dump(isset($a));      //TRUE
var_dump(isset($b));      // TRUE
var_dump(isset($c));      // FALSE
unset($b);
var_dump(isset($b));     // FALSE
```

If multiple parameters are supplied then isset() will return TRUE only if all of the parameters are set. Evaluation goes from left to right & stops as soon as an unset variable is encountered.

Example:

```
var_dump(isset($a, $b));           // bool(TRUE)
var_dump(isset($a, $b, $c));  // bool(FALSE)
```

Also note that a null character ("\0") is not equivalent to the PHP NULL constant.

Date
09/02/2017

void unset(mixed $var [, mixed $...])

It destroys the specified variables.

Example:

```
$x = 25;
unset($x);
echo $x;
```

Output:

```
Notice: Undefined variable: x
```

If a globalized variable is unset() inside of a function, only the local variable is destroyed. The variable in the calling environment will retain the same value as before unset() was called.

Example:

```
function destroy_foo()
{
    global $foo;
    unset($foo);
}
$foo = 'bar';
destroy_foo();
echo $foo;
```

Output:

```
bar
```

Example:

```
function foo()
{
    unset($GLOBALS['bar']);
}
$bar = "something";
foo();
echo $bar;
```

Output:

```
Undefined variable: bar
```

To unset a global variable inside of a function, use $GLOBALS array to do so.

If a variable that is PASSED BY REFERENCE is

Date
09/02/2017

unset() inside of a function, only the local variable is destroyed. The variable in the calling environment will retain the same value as before unset() was called.

Example:

```
function foo(&$bar)
{
    unset($bar);
    $bar = "blah";
}
$bar = 'something';
echo "$bar\n";
foo($bar);
echo "$bar\n";
```

Output:

```
something
something
```

Example:

```
function foo()
{
    static $bar;
    $bar++;
    echo "Before unset: $bar, ";
    unset($bar);
    $bar = 23;
    echo "after unset: $bar\n";
}
foo();
foo();
foo();
```

If a static variable is unset() inside of a function, unset() destroys the variable only in the context of ~~the~~ rest of a function. Following calls will restore ~~the~~ previous value of a variable.

Output:

```
Before unset: 1, after unset: 23
Before unset: 2, after unset: 23
Before unset: 3, after unset: 23
```

Date
10/02/2017

Some similar functions

htmlspecialchars()
htmlspecialchars_decode()
htmlentities()
html_entity_decode()
urlencode()
urldecode()
addslashes()
stripslashes()