PHP - Null Coalescing Operator

The Null Coalescing operator is one of the many new features introduced in PHP 7. The word "coalescing" means uniting many things into one. This operator is used to replace the ternary operation in conjunction with the **isset()** function.

Ternary Operator in PHP

PHP has a ternary operator represented by the "**?**" symbol. The ternary operator compares a Boolean expression and executes the first operand if it is true, otherwise it executes the second operand.

```
expr ? statement1 : statement2;
```

Example

Let us use the ternary operator to check if a certain variable is set or not with the help of the isset() function, which returns true if declared and false if not.

</> Open Compiler

```
<?php
   $x = 1;
   $var = isset($x) ? $x : "not set";
   echo "The value of x is $var";
?>
```

It will produce the following **output** –

```
The value of x is 1
```

Now, let's remove the declaration of "x" and rerun the code –

</> Open Compiler

```
<?php
   # $x = 1;
   $var = isset($x) ? $x : "not set";
```

```
   echo "The value of x is $var";
?>
```

The code will now produce the following **output** –

```
The value of x is not set
```

The Null Coalescing Operator

The Null Coalescing Operator is represented by the "??" symbol. It acts as a convenient shortcut to use a ternary in conjunction with isset(). It returns its first operand if it exists and is not null; otherwise it returns its second operand.

```
$Var = $operand1 ?? $operand2;
```

The first operand checks whether a certain variable is null or not (or is set or not). If it is not null, the first operand is returned, else the second operand is returned.

Example

Take a look at the following example –

</> Open Compiler

```
<?php
   # $num = 10;
   $val = $num ?? 0;
   echo "The number is $val";
?>
```

It will produce the following **output** –

```
The number is 0
```

Now uncomment the first statement that sets **$num** to 10 and rerun the code –

</> Open Compiler

```
<?php
   $num = 10;
```

```
   $val = $num ?? 0;
   echo "The number is $val";
?>
```

It will now produce the following **output** –

```
The number is 10
```

A useful application of Null Coalescing operator is while checking whether a username has been provided by the client browser.

Example

The following code reads the name variable from the URL. If indeed there is a value for the name parameter in the URL, a Welcome message for him is displayed. However, if not, the user is called Guest.

</> Open Compiler

```
<?php
   $username = $_GET['name'] ?? 'Guest';
   echo "Welcome $username";
?>
```

Assuming that this script "hello.php" is in the htdocs folder of the PHP server, enter **http://localhost/hello.php?name=Amar** in the URL, the browser will show the following message –

```
Welcome Amar
```

If **http://localhost/hello.php** is the URL, the browser will show the following message –

```
Welcome Guest
```

The Null coalescing operator is used as a replacement for the ternary operator's specific case of checking isset() function. Hence, the following statements give similar results –

</> Open Compiler

```php
<?php
   $username = isset($_GET['name']) ? $_GET['name'] : 'Guest';
   echo "Welcome $username";
?>
```

It will now produce the following **output** –

```
Welcome Guest
```

You can chain the "??" operators as shown below –

</> Open Compiler

```php
<?php
   $username = $_GET['name'] ?? $_POST['name'] ?? 'Guest';
   echo "Welcome $username";
?>
```

It will now produce the following **output** –

```
Welcome Guest
```

This will set the username to Guest if the variable $name is not set either by GET or by POST method.