PHP oops - Access Modifiers

Topics : PHP
Written on February 11, 2021

What is Access Modifiers in php oops?

PHP access modifiers are used to provide access rights for PHP classes and their members that are the functions and variables defined within the class scope.

Properties and methods can have access modifiers which control where they can be accessed.

There are 3 types of Access Specifiers available in PHP like Public, Private and Protected.

1. **Public :** class members with this access modifier will be publicly accessible from anywhere, even from outside the scope of the class.
2. **Private :** the property or method can only be accessed within the class
3. **Protected :** Protected class properties are a bit like private properties in that they can't be accessed by code outside the class, but there's one little difference in any class that inherits from the class i.e. base class can also access the properties.

You can define scope of class, method and variable using keyword like public, private and protected.

PHP Example :

```php
<?php
class Mobile {
  public $name;
  protected $color;
  private $model;
}

$mobile = new Mobile();
$mobile->name = 'Samsung'; // OK
$mobile->color = 'Blue'; // Error: Cannot access protected property Mobile::$color
$mobile->model = 'M31s'; // Error: Cannot access private property Mobile::$model
?>
```

You can see in above example that protected and private variable can not accessed outside class Mobile.

PHP Example :

```php
<?php
class Mobile {
```

```php
  public $name;
  public $color;
  public $model;
  public function set_name($name)
  {
     $this->name = $name;
  }

  private function set_color($name)
  {
     $this->color = $name;
  }

  private function set_model($name)
  {
     $this->model = $name;
  }
}

$mobile = new Mobile();
$mobile->set_name('Samsung');
$mobile->set_color('Blue'); //PHP Fatal error:  Uncaught Error: Call to private method
Mobile::set_color()
$mobile->set_model('M31s'); //PHP Fatal error:  Uncaught Error: Call to private method
Mobile::set_model() from global scope

?>
```

You can see in above example that protected and private function can not accessed outside class Mobile.

Output :