PHP - The "Final" Keyword

The "final" keyword in PHP is used in the definition of a class, a method inside a class, as well as with the definition of a constant property of a class.

A Class with "final" Keyword

Let's see how to create a class with the "final" keyword −

```
final class myclass {
   /*class members*/
}
```

The "final" keyword in class definition prevents such a class from being extended. In other words, you cannot use a final class as a parent. If you try, PHP parser throws an error

Open Compiler

```
<?php
   final class myclass {

      /* class body */
   }
   class newclass extends myclass {

      /* class body */
   }
?>
```

When you run this code, it will show an **error** −

```
PHP Fatal error:  Class newclass may not inherit from final class (myclass)
```

Method with "final" Keyword

Here is how you can create a method with the "final" keyword −

```
class myclass {
   final function myfunction() {

      /* function body */
   }
}
```

Prefixing a method definition with the **final** keyword prevents it from being overridden in a child class. A class with **final** method can be extended, but the child class cannot override it.

Example

Take a look at the following example –

Open Compiler

```
<?php
   class myclass {
      final public function hello() {
         echo "Hello World!";
      }
   }
   class newclass extends myclass {
      public function hello() {
         echo "Hello PHP!";
      }
   }
?>
```

When you run this code, it will show an **error** –

```
PHP Fatal error:  Cannot override final method myclass::hello() in hello.php
```

Constant with "final" Keyword

You can also declare a constant in a class with the **final** keyword, starting from PHP 8.1.0 onwards.

```
final public const NAME = "My Class";
```

If you try to override a final constant from parent class in a child class, an error is encountered.

</> Open Compiler

```
<?php
   class myclass {
      final public const NAME = "My Class";
      final public function hello() {
         echo "Hello World!";
      }
   }

   class newclass extends myclass {
      public const NAME = "New Class";
   }
?>
```

When you run this code, it will show an **error** −

```
Fatal error: newclass::NAME cannot override final constant myclass::NAME
```

Example

The following PHP script contains a parent class ellipse with a PI constant and area() method both declared as final. They are inherited by the circle class. The area() function calculates the area of circle.

</> Open Compiler

```
<?php
   class ellipse {
      final public const PI=22/7;
      private float $a, $b;
      public function __construct($x, $y) {
         $this->a = $x;
         $this->b = $y;
      }
      final public function area() : float {
         return self::PI*$this->a*$this->b;
      }
   }
   class circle extends ellipse {
      public function __construct(float $x) {
         parent::__construct($x, $x);
      }
   }
   $c1 = new circle(5);
   echo "Area: " . $c1->area() . PHP_EOL;
?>
```

It will produce the following **output** −

```
Area: 78.571428571429
```

Note that the instance variables or properties of a class cannot be declared as final.