Visibility (public, private, protected)

The visibility of a property, a method or (as of PHP 7.1.0) a constant can be defined by prefixing the declaration with the keywords public, protected or private. Class members declared public can be accessed everywhere. Members declared protected can be accessed only within the class itself and by inheriting and parent classes. Members declared as private may only be accessed by the class that defines the member.

***Public:***

When you declare a method (function) or a property (variable) as public, those methods and properties can be accessed by:

The same class that declared it.

The classes that inherit the above declared class.

Any foreign elements outside this class can also access those things.

class GrandPa

{

    public $name='Mark Henry';  // A public variable

}

class Daddy extends GrandPa // Inherited class

{

    function displayGrandPaName()

    {

        return $this->name; // The public

// variable will be available to the inherited class

    }

}

// Inherited class Daddy wants to know Grandpas Name

$daddy = new Daddy;

echo $daddy->displayGrandPaName(); // Prints 'Mark Henry'

// Public variables can also be accessed outside of the class!

$outsiderWantstoKnowGrandpasName = new GrandPa;

echo $outsiderWantstoKnowGrandpasName->name; // Prints 'Mark Henry'

\*public properties/methods/constants can be used by: the originating class, any of its instances and the extending classes or any of their instances (they can be accessed directly without having to be returned from a method)

***Protected:***

Protected:

When you declare a method (function) or a property (variable) as protected, those methods and properties can be accessed by:

The same class that declared it.

The classes that inherit the above declared class.

Outsider members cannot access those variables. "Outsiders" in the sense that they are not object instances of the inheriting class itself (so only the instances of the parent class).

class GrandPa

{

    protected $name = 'Mark Henry';

}

class Daddy extends GrandPa

{

    function displayGrandPaName()

    {

        return $this->name;

    }

}

$daddy = new Daddy;

echo $daddy->displayGrandPaName(); // Prints 'Mark Henry'

$outsiderWantstoKnowGrandpasName = new GrandPa;

echo $outsiderWantstoKnowGrandpasName->name; // Results in a Fatal Error

\*protected properties/methods/constants can only be used by the originating class and the classes that extend it + the inheriting classes’ instances, they CANNOT be used by instances of the originating class. They have to be returned by a class method in order to be used by the class instances;

\*for the class instance (for inheriting classes), the protected properties/methods/constants cannot be accessed directly, they have to be returned through a class metod:

class GrandPa

{

    protected $name = 'Mark Henry';

}

class Daddy extends GrandPa

{

    protected $name = 'Mark Henry2';

    function displayGrandPaName()

    {

        return $this->name;

    }

}

$daddy = new Daddy;

echo $daddy->name; // throws error

/\*

even though we have a protected $name defined inside of Daddy class,

we have to return it through a method of the class to access

it from the Daddy class instances)

\*/

\*if we return the protected name through a class method, we can access it from the class instance:

class GrandPa

{

    protected $name = 'Mark Henry';

}

class Daddy extends GrandPa

{

    protected $name = 'Mark Henry2';

    function displaydaddy()

    {

        return $this->name;

    }

}

$daddy = new Daddy;

echo $daddy->displaydaddy(); // prints Mark henry2

***Private:***

When you declare a method (function) or a property (variable) as private, those methods and properties can be accessed by:

The same class that declared it.

class GrandPa

{

    private $name = 'Mark Henry';

}

class Daddy extends GrandPa

{

    function displayGrandPaName()

    {

        return $this->name;

    }

}

$daddy = new Daddy;

echo $daddy->displayGrandPaName(); // Results in a Notice

$outsiderWantstoKnowGrandpasName = new GrandPa;

echo $outsiderWantstoKnowGrandpasName->name; // Results in a Fatal Error

\*we can access GrandPa’s private $name only by returning it through a public method in a class instance:

class GrandPa

{

    private $name = 'Mark Henry';

    public function returnName(){

        return $this->name;

    }

}

class Daddy extends GrandPa

{

    function displayGrandPaName()

    {

        return $this->name;

    }

}

// $daddy = new Daddy;

// echo $daddy->displayGrandPaName(); // Results in a Notice

$outsiderWantstoKnowGrandpasName = new GrandPa;

echo $outsiderWantstoKnowGrandpasName->returnName(); // prints GrandPa's name