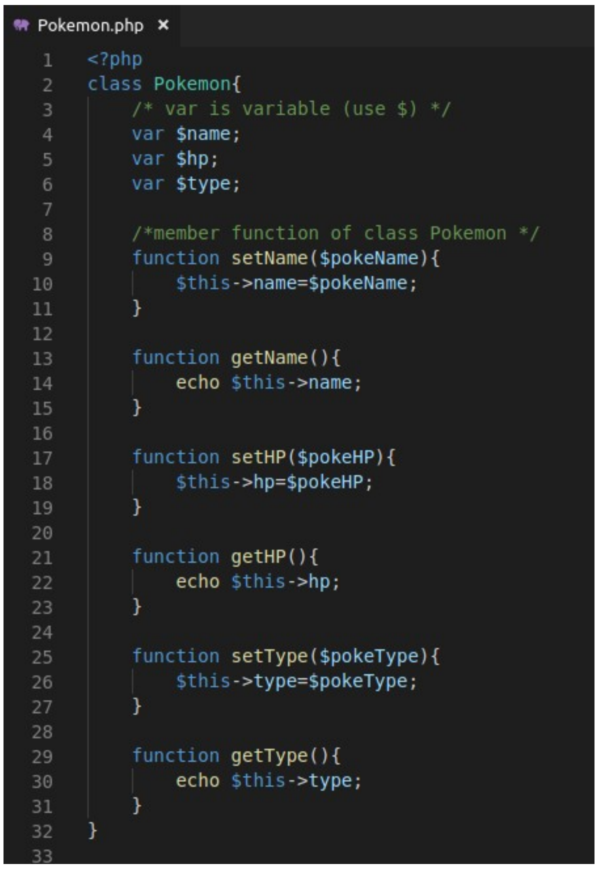
{OOP} in PHP

Try OOP in PHP
(Pokemon case)

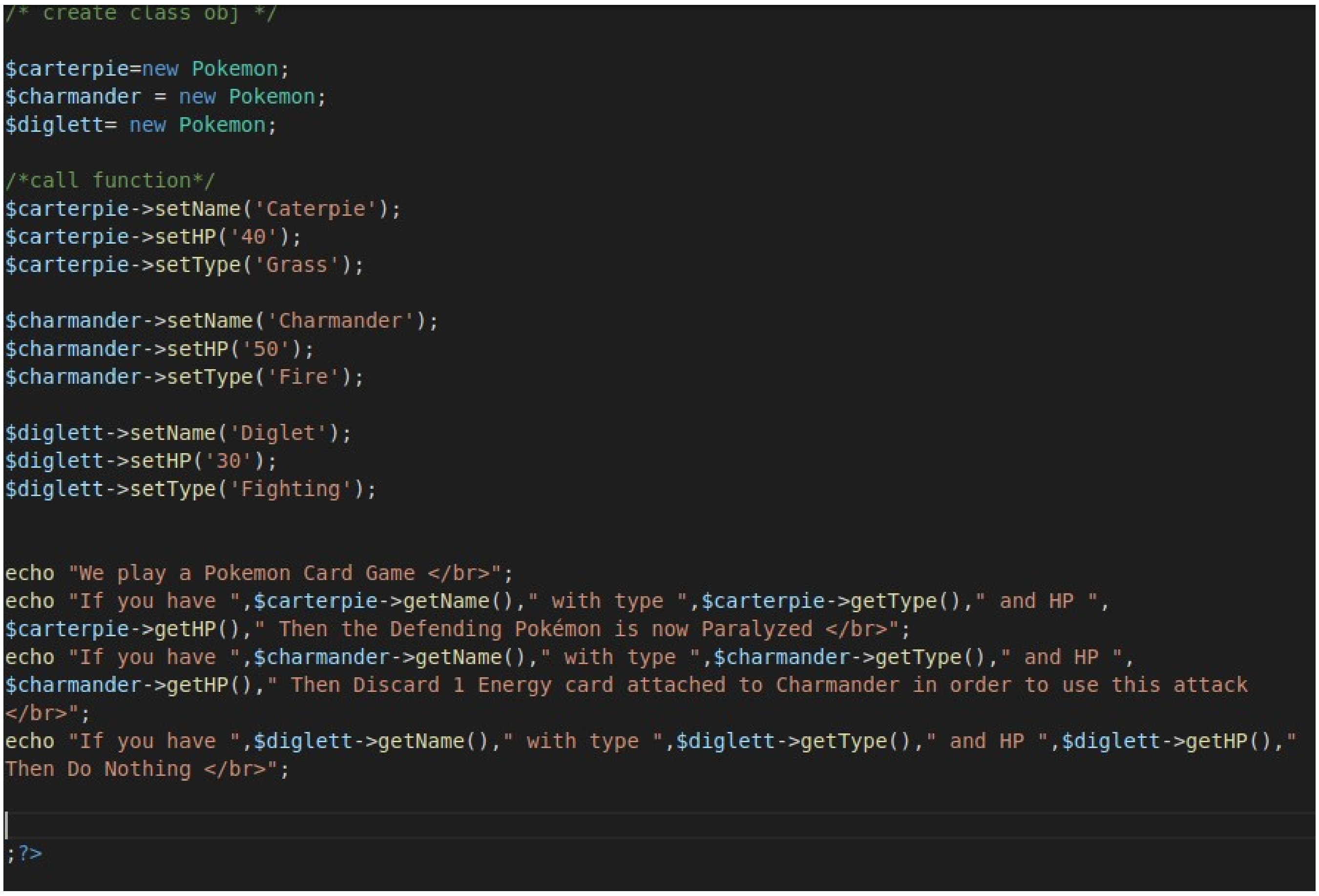
Basic OOP Concepts

1. Class (variable member, function member, object, call fuction)
2. Constructor Function
3. Inheritance (extends)
4. Function Overriding
5. Public Members
6. Private Members
7. Protected Members
8. Interfaces
9. Constant
10. Abstract Classes
11. Static Keyword
12. Final Keyword

1. Class

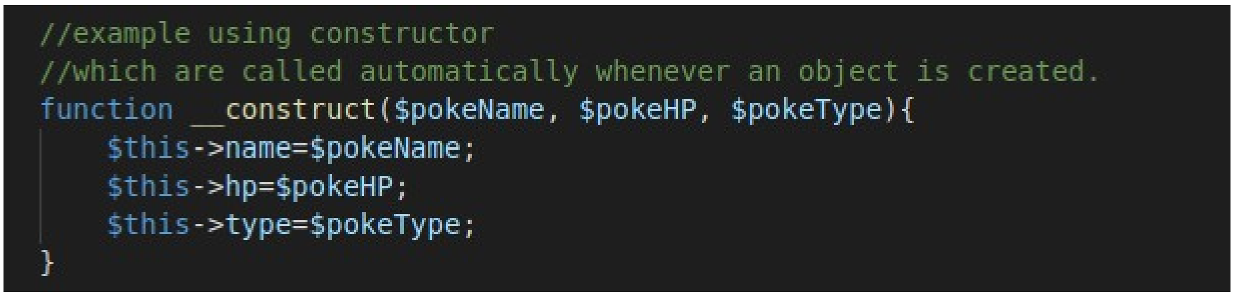


```php
<?php
class Pokemon{
    /* var is variable (use $) */
    var $name;
    var $hp;
    var $type;

    /*member function of class Pokemon */
    function setName($pokeName){
        $this->name=$pokeName;
    }

    function getName(){
        echo $this->name;
    }

    function setHP($pokeHP){
        $this->hp=$pokeHP;
    }

    function getHP(){
        echo $this->hp;
    }

    function setType($pokeType){
        $this->type=$pokeType;
    }

    function getType(){
        echo $this->type;
    }
}

```



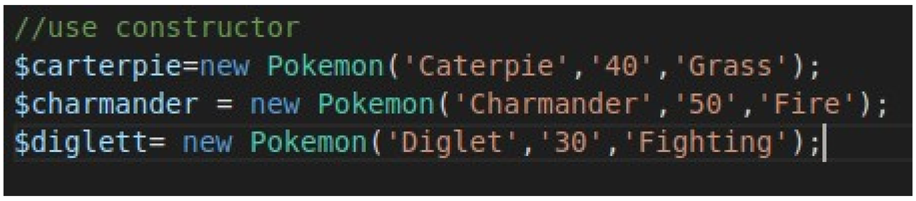
```php
/* create class obj */

$carterpie=new Pokemon;
$charmander = new Pokemon;
$diglett= new Pokemon;

/*call function*/
$carterpie->setName('Caterpie');
$carterpie->setHP('40');
$carterpie->setType('Grass');

$charmander->setName('Charmander');
$charmander->setHP('50');
$charmander->setType('Fire');

$diglett->setName('Diglet');
$diglett->setHP('30');
$diglett->setType('Fighting');


echo "We play a Pokemon Card Game </br>";
echo "If you have ",$carterpie->getName()," with type ",$carterpie->getType()," and HP ",
$carterpie->getHP()," Then the Defending Pokémon is now Paralyzed </br>";
echo "If you have ",$charmander->getName()," with type ",$charmander->getType()," and HP ",
$charmander->getHP()," Then Discard 1 Energy card attached to Charmander in order to use this attack
</br>";
echo "If you have ",$diglett->getName()," with type ",$diglett->getType()," and HP ",$diglett->getHP(),"
Then Do Nothing </br>";


;?>
```

2. Constructor Function



```
//example using constructor
//which are called automatically whenever an object is created.
function __construct($pokeName, $pokeHP, $pokeType){
    $this->name=$pokeName;
    $this->hp=$pokeHP;
    $this->type=$pokeType;
}
```

Comment /**/ all setter then create object

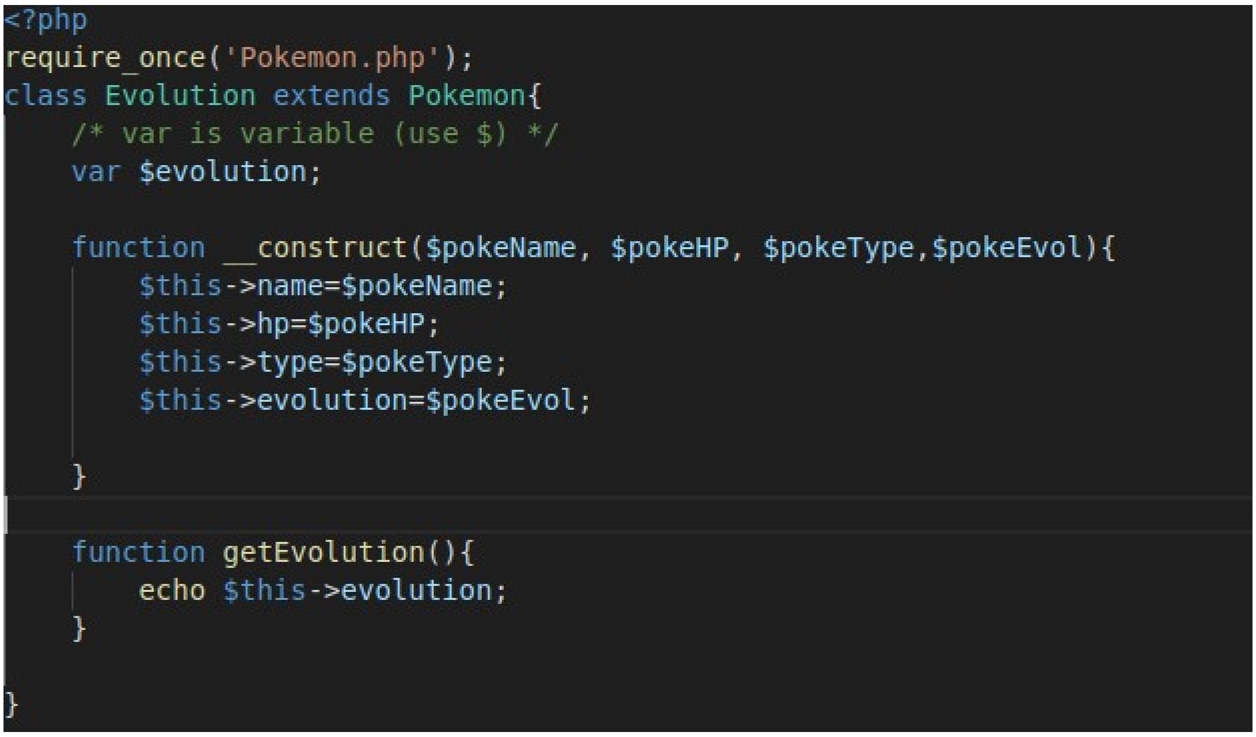


```
//use constructor
$carterpie=new Pokemon('Caterpie','40','Grass');
$charmander = new Pokemon('Charmander','50','Fire');
$diglett= new Pokemon('Diglet','30','Fighting');
```

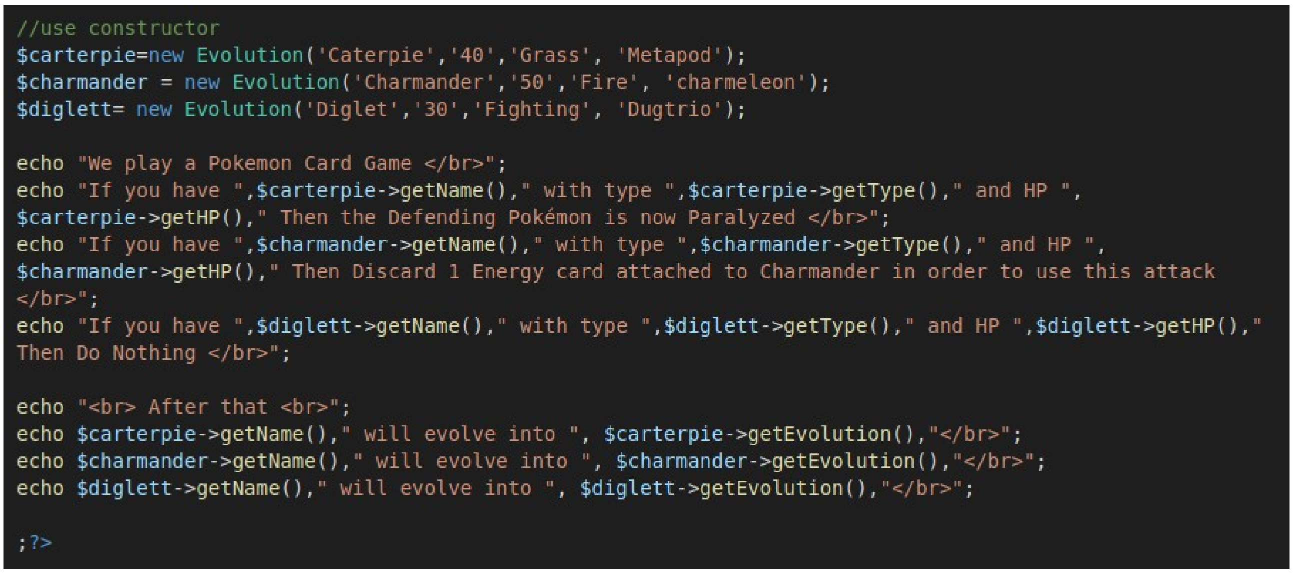
Output:

```
We play a Pokemon Card Game
If you have Caterpie with type Grass and HP 40 Then the Defending Pokémon is now Paralyzed
If you have Charmander with type Fire and HP 50 Then Discard 1 Energy card attached to Charmander in order to use this attack
If you have Diglet with type Fighting and HP 30 Then Do Nothing
```

3. Inheritance



```php
<?php
require_once('Pokemon.php');
class Evolution extends Pokemon{
    /* var is variable (use $) */
    var $evolution;

    function __construct($pokeName, $pokeHP, $pokeType,$pokeEvol){
        $this->name=$pokeName;
        $this->hp=$pokeHP;
        $this->type=$pokeType;
        $this->evolution=$pokeEvol;

    }

    function getEvolution(){
        echo $this->evolution;
    }

}
```



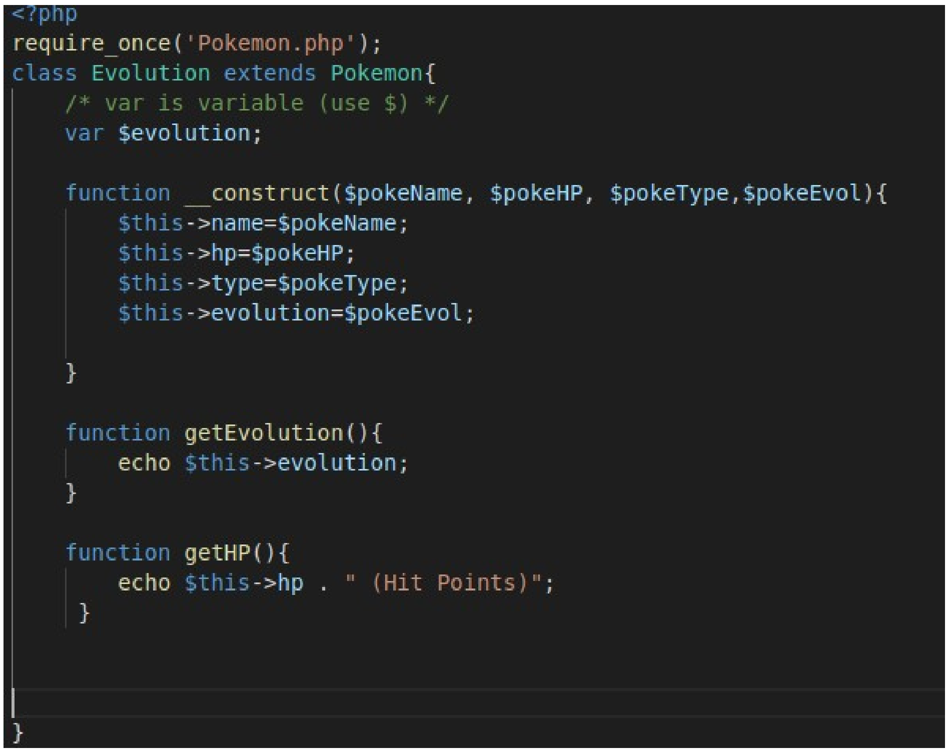
```php
//use constructor
$carterpie=new Evolution('Caterpie','40','Grass', 'Metapod');
$charmander = new Evolution('Charmander','50','Fire', 'charmeleon');
$diglett= new Evolution('Diglet','30','Fighting', 'Dugtrio');

echo "We play a Pokemon Card Game </br>";
echo "If you have ",$carterpie->getName()," with type ",$carterpie->getType()," and HP ",
$carterpie->getHP()," Then the Defending Pokémon is now Paralyzed </br>";
echo "If you have ",$charmander->getName()," with type ",$charmander->getType()," and HP ",
$charmander->getHP()," Then Discard 1 Energy card attached to Charmander in order to use this attack
</br>";
echo "If you have ",$diglett->getName()," with type ",$diglett->getType()," and HP ",$diglett->getHP(),"
Then Do Nothing </br>";

echo "<br> After that <br>";
echo $carterpie->getName()," will evolve into ", $carterpie->getEvolution(),"</br>";
echo $charmander->getName()," will evolve into ", $charmander->getEvolution(),"</br>";
echo $diglett->getName()," will evolve into ", $diglett->getEvolution(),"</br>";

;?>
```

output :

```
We play a Pokemon Card Game
If you have Caterpie with type Grass and HP 40 Then the Defending Pokémon is now Paralyzed
If you have Charmander with type Fire and HP 50 Then Discard 1 Energy card attached to Charmander in order to use this attack
If you have Diglet with type Fighting and HP 30 Then Do Nothing

After that
Caterpie will evolve into Metapod
Charmander will evolve into charmeleon
Diglet will evolve into Dugtrio
```

4. Function Overriding

Function getHP in child override function in parent with same name.



```php
<?php
require_once('Pokemon.php');
class Evolution extends Pokemon{
    /* var is variable (use $) */
    var $evolution;

    function __construct($pokeName, $pokeHP, $pokeType,$pokeEvol){
        $this->name=$pokeName;
        $this->hp=$pokeHP;
        $this->type=$pokeType;
        $this->evolution=$pokeEvol;

    }

    function getEvolution(){
        echo $this->evolution;
    }

    function getHP(){
        echo $this->hp . " (Hit Points)";
     }


}
```

Output :

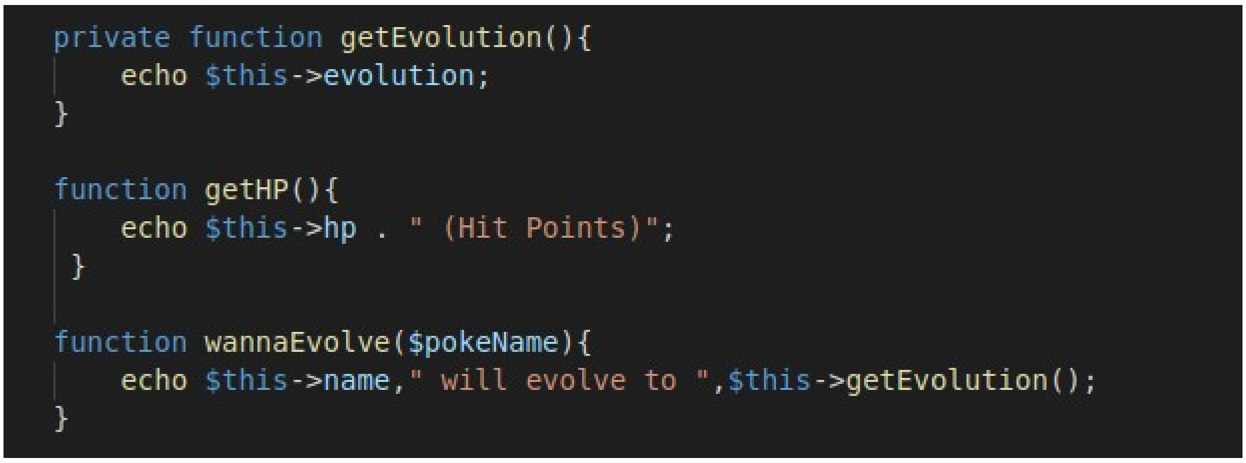
```
We play a Pokemon Card Game
If you have Caterpie with type Grass and HP 40 (Hit Points) Then the Defending Pokémon is now Paralyzed
If you have Charmander with type Fire and HP 50 (Hit Points) Then Discard 1 Energy card attached to Charmander in order to use this attack
If you have Diglet with type Fighting and HP 30 (Hit Points) Then Do Nothing
```

5. Public Members

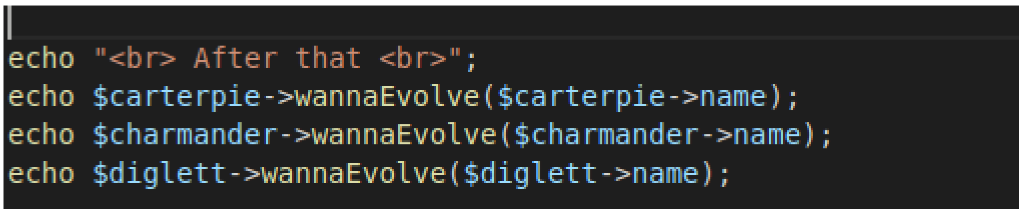
By default var and functions is public, can be accessed outside class, within class, or another class.

6. Private Members

Example 1



```php
private function getEvolution(){
    echo $this->evolution;
}

function getHP(){
    echo $this->hp . " (Hit Points)";
 }

function wannaEvolve($pokeName){
    echo $this->name," will evolve to ",$this->getEvolution();
}
```

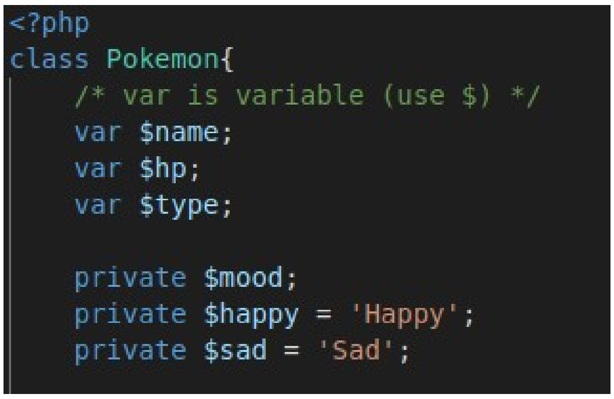


```php
echo "<br> After that <br>";
echo $carterpie->wannaEvolve($carterpie->name);
echo $charmander->wannaEvolve($charmander->name);
echo $diglett->wannaEvolve($diglett->name);
```

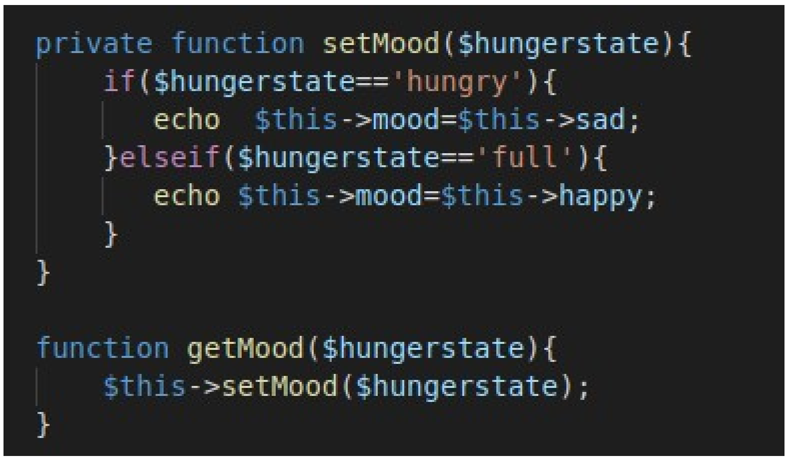
Output

```
After that
Caterpie will evolve to Metapod
Charmander will evolve to charmeleon
Diglet will evolve to Dugtrio
```

Example 2



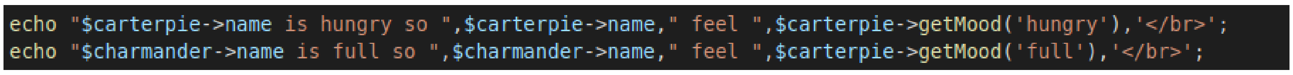
```php
<?php
class Pokemon{
    /* var is variable (use $) */
    var $name;
    var $hp;
    var $type;

    private $mood;
    private $happy = 'Happy';
    private $sad = 'Sad';
```



```php
private function setMood($hungerstate){
    if($hungerstate=='hungry'){
        echo  $this->mood=$this->sad;
    }elseif($hungerstate=='full'){
        echo $this->mood=$this->happy;
    }
}

function getMood($hungerstate){
    $this->setMood($hungerstate);
}
```

call the function in evolve class *).

*) I’m not allowed to set directly the mood of the pokemon.



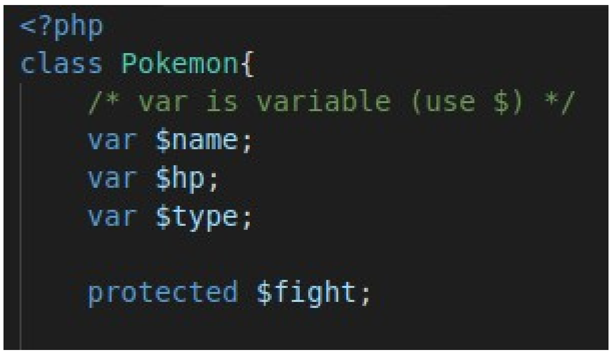
```php
echo "$carterpie->name is hungry so ",$carterpie->name," feel ",$carterpie->getMood('hungry'),'</br>';
echo "$charmander->name is full so ",$charmander->name," feel ",$carterpie->getMood('full'),'</br>';
```

Output :

```
Caterpie is hungry so Caterpie feel Sad
Charmander is full so Charmander feel Happy
```

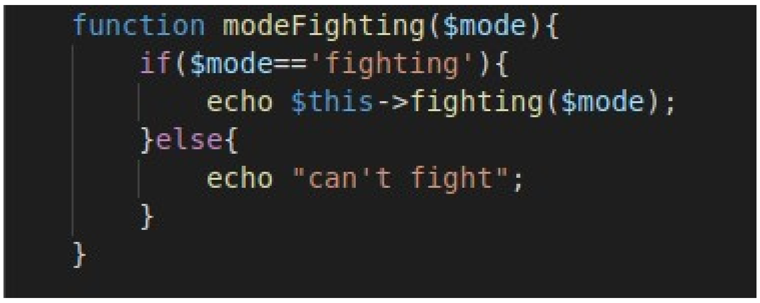
7. Protected members

Pokemon Class

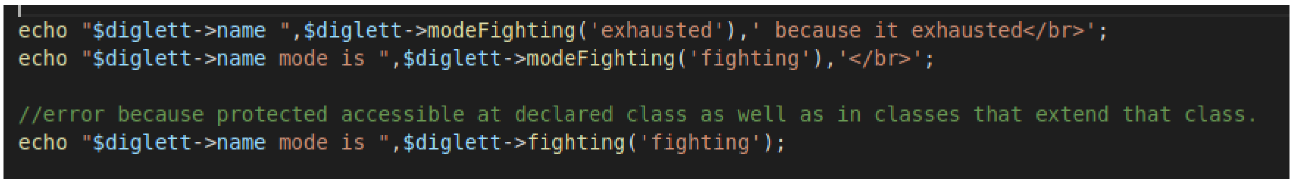


```php
<?php
class Pokemon{
    /* var is variable (use $) */
    var $name;
    var $hp;
    var $type;

    protected $fight;
```

Evolution Class extend Pokemon



```php
    function modeFighting($mode){
        if($mode=='fighting'){
            echo $this->fighting($mode);
        }else{
            echo "can't fight";
        }
    }
```



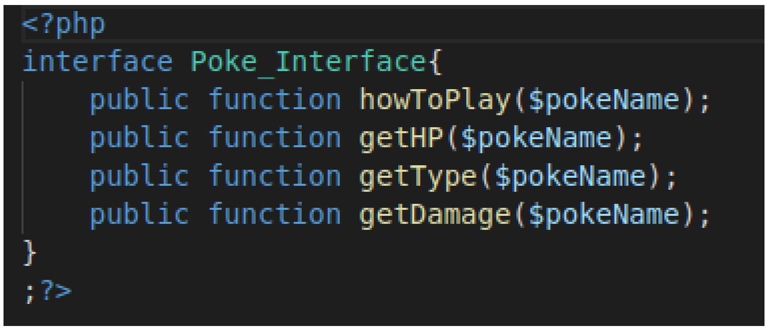
```php
echo "$diglett->name ",$diglett->modeFighting('exhausted'),' because it exhausted</br>';
echo "$diglett->name mode is ",$diglett->modeFighting('fighting'),'</br>';

//error because protected accessible at declared class as well as in classes that extend that class.
echo "$diglett->name mode is ",$diglett->fighting('fighting');
```

output :

```
Diglet can't fight because it exhausted
Diglet mode is fighting
Diglet mode is
Fatal error: Uncaught Error: Call to protected method Pokemon::fighting() from context '' in
/opt/lampp/htdocs/belajaroop/Evolution.php:68 Stack trace: #0 {main} thrown in
/opt/lampp/htdocs/belajaroop/Evolution.php on line 68
```

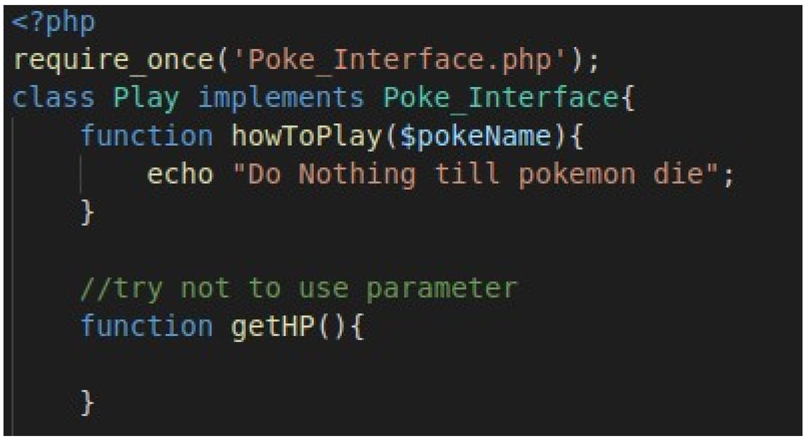
8. Interfaces

Poke_Interface.php



```php
<?php
interface Poke_Interface{
    public function howToPlay($pokeName);
    public function getHP($pokeName);
    public function getType($pokeName);
    public function getDamage($pokeName);
}
;?>
```

Example 1 :

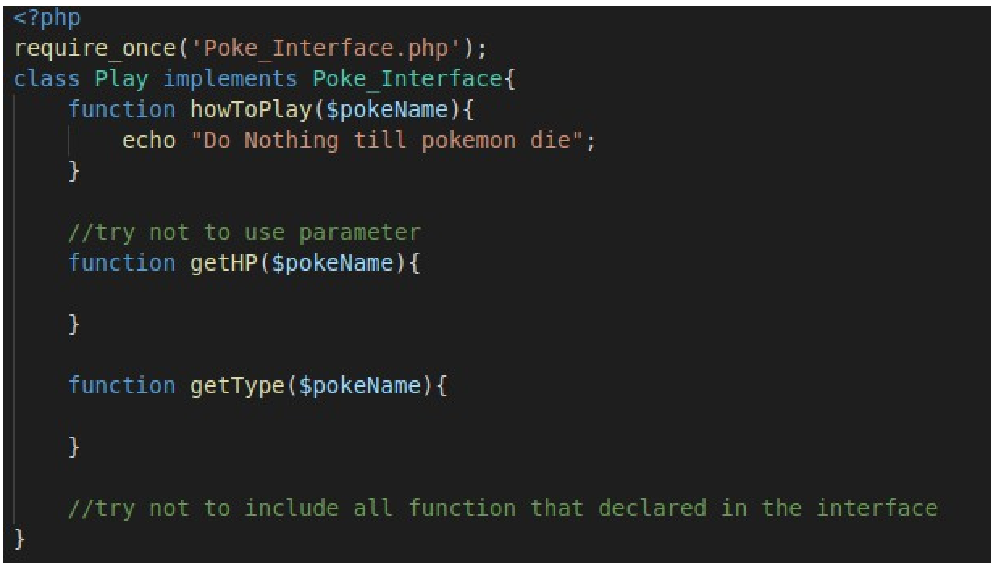


```php
<?php
require_once('Poke_Interface.php');
class Play implements Poke_Interface{
    function howToPlay($pokeName){
        echo "Do Nothing till pokemon die";
    }

    //try not to use parameter
    function getHP(){

    }
```

Output :

Fatal error: Declaration of Play::getHP() must be compatible with Poke_Interface::getHP($pokeName) in **/opt/lampp/htdocs/belajaroop/Play.php** on line **3**

Example 2:

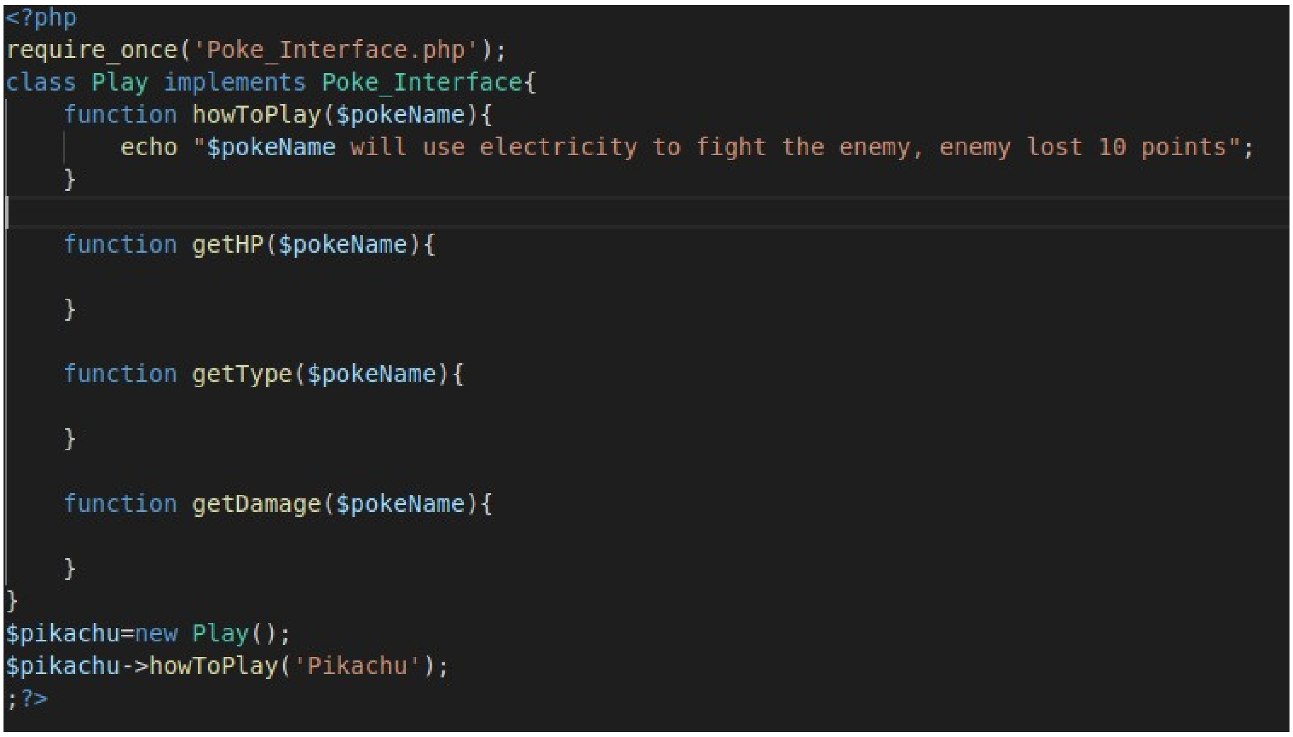


```php
<?php
require_once('Poke_Interface.php');
class Play implements Poke_Interface{
    function howToPlay($pokeName){
        echo "Do Nothing till pokemon die";
    }

    //try not to use parameter
    function getHP($pokeName){

    }

    function getType($pokeName){

    }

    //try not to include all function that declared in the interface
}
```

Output :

Fatal error: Class Play contains 1 abstract method and must therefore be declared abstract or implement the remaining methods (Poke_Interface::getDamage) in **/opt/lampp/htdocs/belajaroop/Play.php** on line **3**

Example 3 :



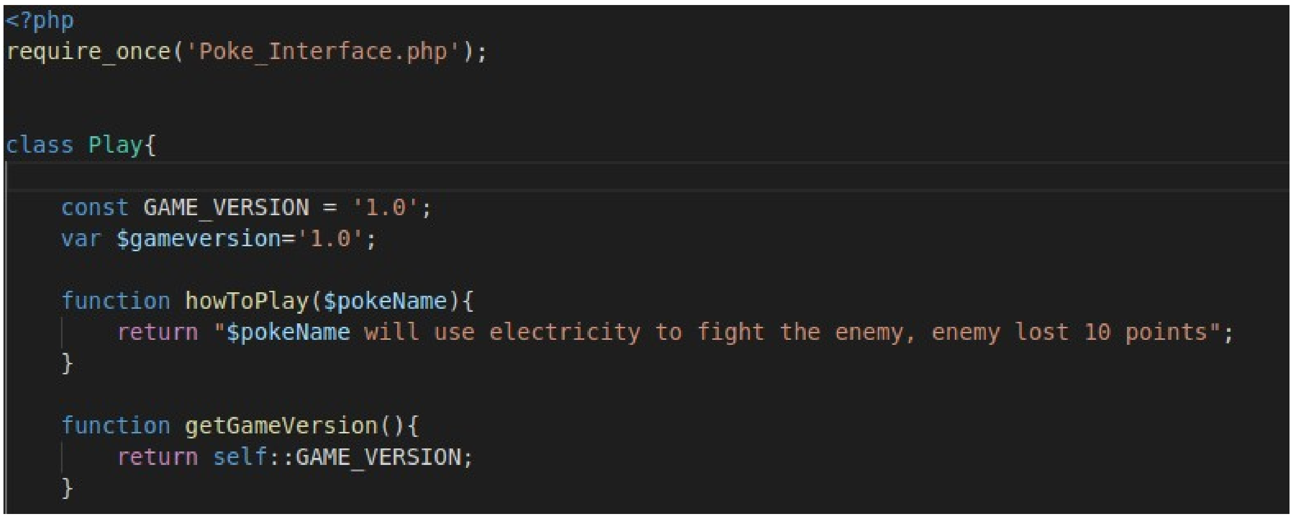
```php
<?php
require_once('Poke_Interface.php');
class Play implements Poke_Interface{
    function howToPlay($pokeName){
        echo "$pokeName will use electricity to fight the enemy, enemy lost 10 points";
    }

    function getHP($pokeName){

    }

    function getType($pokeName){

    }

    function getDamage($pokeName){

    }
}
$pikachu=new Play();
$pikachu->howToPlay('Pikachu');
;?>
```

Ouput :

```
Pikachu will use electricity to fight the enemy, enemy lost 10 points
```

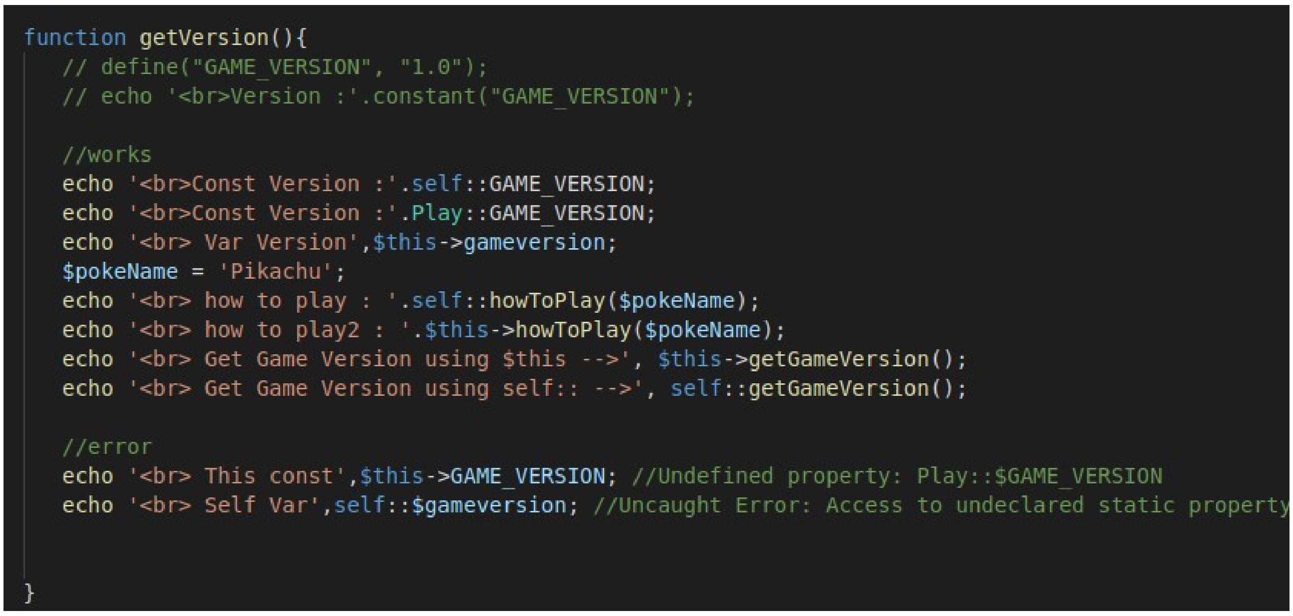
9. Constant

Const vs Var

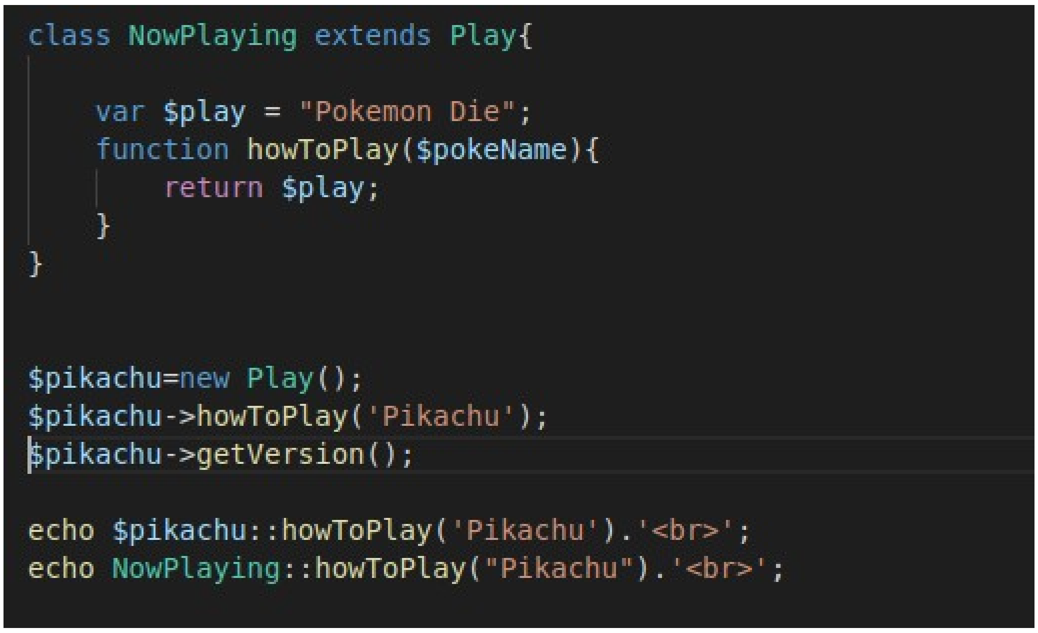


```php
<?php
require_once('Poke_Interface.php');


class Play{

    const GAME_VERSION = '1.0';
    var $gameversion='1.0';

    function howToPlay($pokeName){
        return "$pokeName will use electricity to fight the enemy, enemy lost 10 points";
    }

    function getGameVersion(){
        return self::GAME_VERSION;
    }
```

Example



```php
function getVersion(){
    // define("GAME_VERSION", "1.0");
    // echo '<br>Version :'.constant("GAME_VERSION");

    //works
    echo '<br>Const Version :'.self::GAME_VERSION;
    echo '<br>Const Version :'.Play::GAME_VERSION;
    echo '<br> Var Version',$this->gameversion;
    $pokeName = 'Pikachu';
    echo '<br> how to play : '.self::howToPlay($pokeName);
    echo '<br> how to play2 : '.$this->howToPlay($pokeName);
    echo '<br> Get Game Version using $this -->', $this->getGameVersion();
    echo '<br> Get Game Version using self:: -->', self::getGameVersion();

    //error
    echo '<br> This const',$this->GAME_VERSION; //Undefined property: Play::$GAME_VERSION
    echo '<br> Self Var',self::$gameversion; //Uncaught Error: Access to undeclared static property


}
```

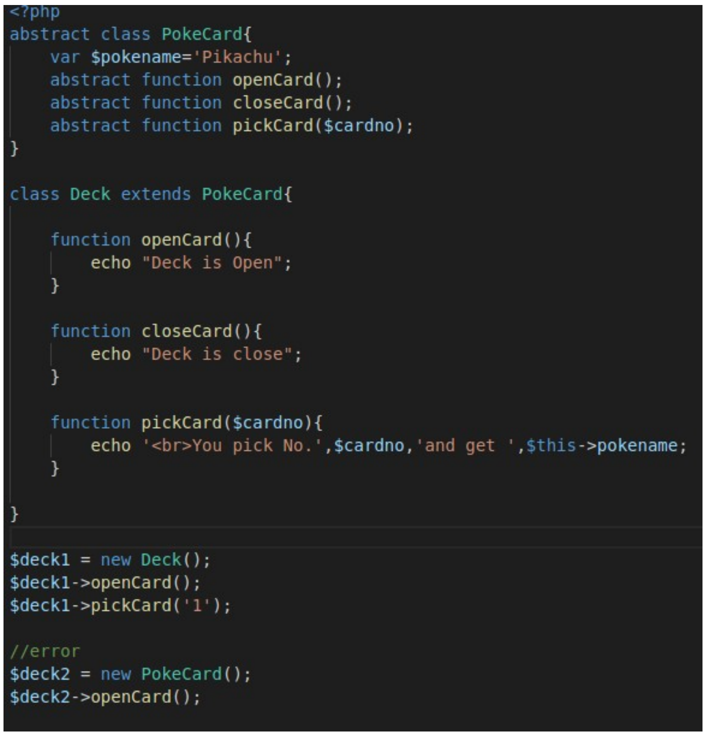


```php
class NowPlaying extends Play{

    var $play = "Pokemon Die";
    function howToPlay($pokeName){
        return $play;
    }
}


$pikachu=new Play();
$pikachu->howToPlay('Pikachu');
$pikachu->getVersion();

echo $pikachu::howToPlay('Pikachu').'<br>';
echo NowPlaying::howToPlay("Pikachu").'<br>';
```

Output :

```
Const Version :1.0
Const Version :1.0
Var Version1.0
how to play : Pikachu will use electricity to fight the enemy, enemy lost 10 points
how to play2 : Pikachu will use electricity to fight the enemy, enemy lost 10 points
Get Game Version using $this -->1.0
Get Game Version using self:: -->1.0
This const
Notice: Undefined property: Play::$GAME_VERSION in /opt/lampp/htdocs/belajaroop/Play.php on line 33

Self Var
Fatal error: Uncaught Error: Access to undeclared static property: Play::$gameversion in /opt/lampp/htdocs/belajaroop/Play.ph
>getVersion() #1 {main} thrown in /opt/lampp/htdocs/belajaroop/Play.php on line 34


 Notice: Undefined variable: play in /opt/lampp/htdocs/belajaroop/Play.php on line 57
```

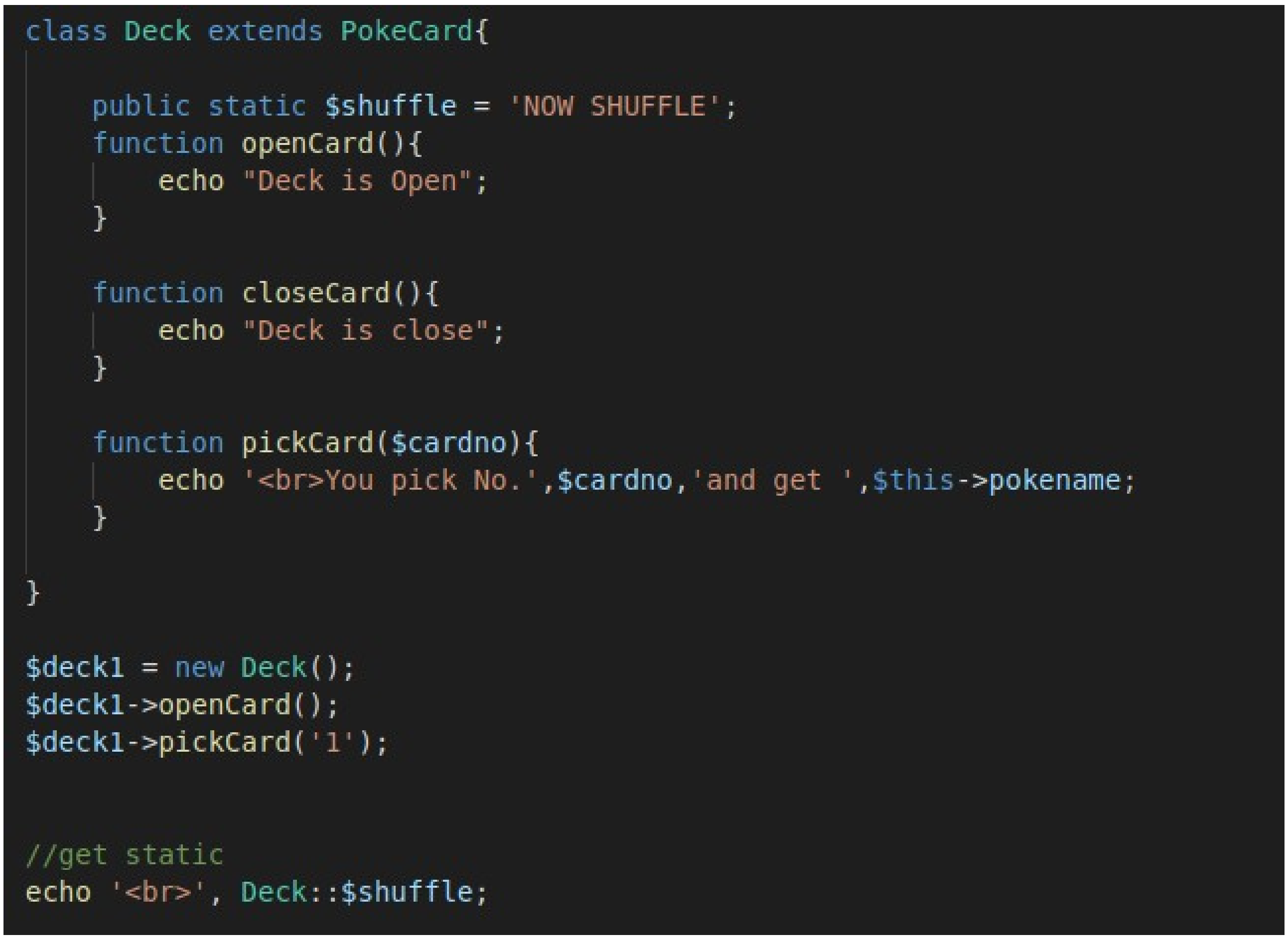
10. Abstract Classes


```php
<?php
abstract class PokeCard{
    var $pokename='Pikachu';
    abstract function openCard();
    abstract function closeCard();
    abstract function pickCard($cardno);
}

class Deck extends PokeCard{

    function openCard(){
        echo "Deck is Open";
    }

    function closeCard(){
        echo "Deck is close";
    }

    function pickCard($cardno){
        echo '<br>You pick No.',$cardno,'and get ',$this->pokename;
    }

}

$deck1 = new Deck();
$deck1->openCard();
$deck1->pickCard('1');

//error
$deck2 = new PokeCard();
$deck2->openCard();
```

Output :

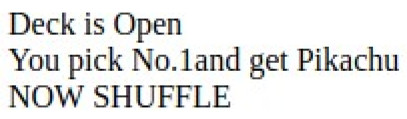
Deck is Open
You pick No.1and get Pikachu
Fatal error: Uncaught Error: Cannot instantiate abstract class PokeCard in /opt/lampp/htdocs/belajaroop/Abstract.php:29 Stack trace: #0 {main} thrown in **/opt/lampp/htdocs/belajaroop/Abstract.php** on line **29**

11. Static Keyword



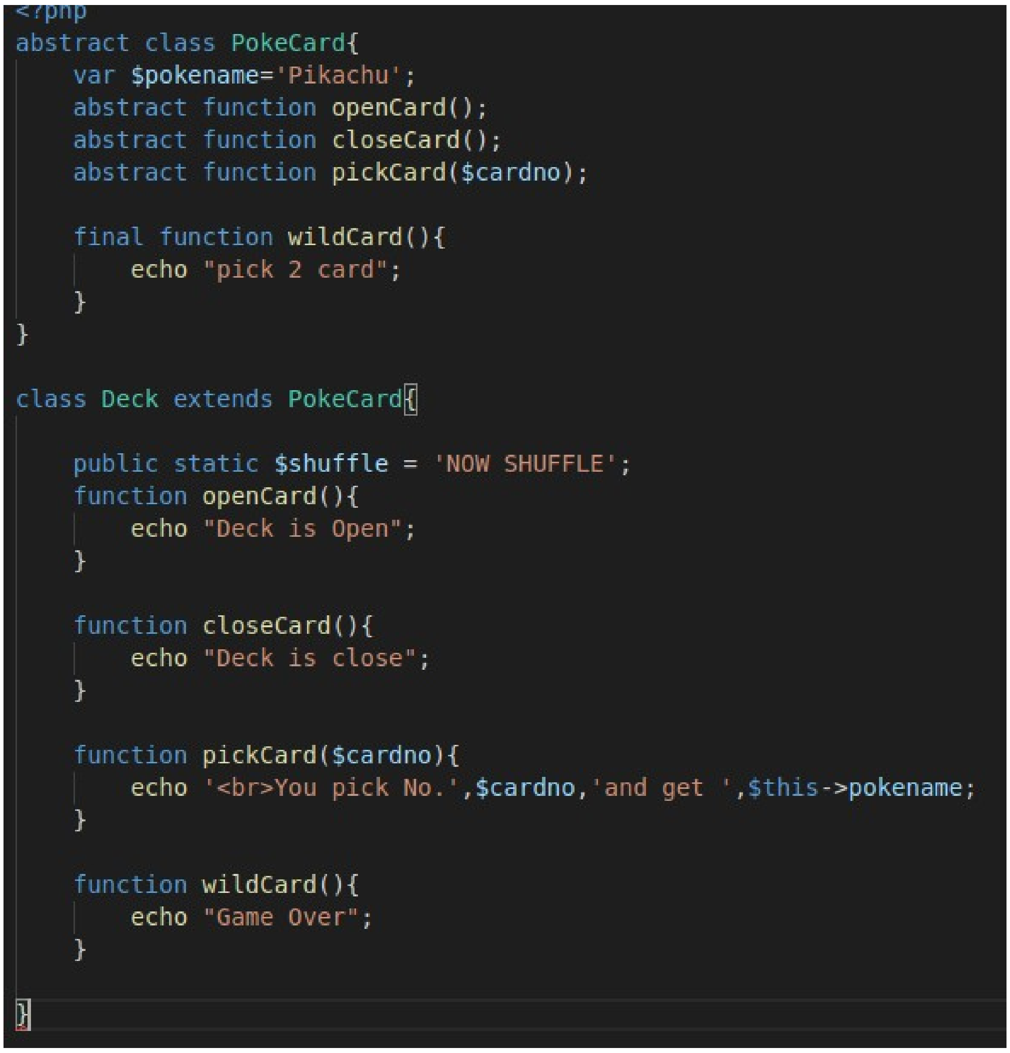
```php
class Deck extends PokeCard{

    public static $shuffle = 'NOW SHUFFLE';
    function openCard(){
        echo "Deck is Open";
    }

    function closeCard(){
        echo "Deck is close";
    }

    function pickCard($cardno){
        echo '<br>You pick No.',$cardno,'and get ',$this->pokename;
    }

}

$deck1 = new Deck();
$deck1->openCard();
$deck1->pickCard('1');


//get static
echo '<br>', Deck::$shuffle;
```

Output :

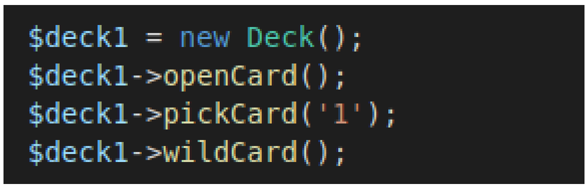


```
Deck is Open
You pick No.1and get Pikachu
NOW SHUFFLE
```

12. Final Keyword



```php
<?php
abstract class PokeCard{
    var $pokename='Pikachu';
    abstract function openCard();
    abstract function closeCard();
    abstract function pickCard($cardno);

    final function wildCard(){
        echo "pick 2 card";
    }
}

class Deck extends PokeCard{

    public static $shuffle = 'NOW SHUFFLE';
    function openCard(){
        echo "Deck is Open";
    }

    function closeCard(){
        echo "Deck is close";
    }

    function pickCard($cardno){
        echo '<br>You pick No.',$cardno,'and get ',$this->pokename;
    }

    function wildCard(){
        echo "Game Over";
    }

}
```

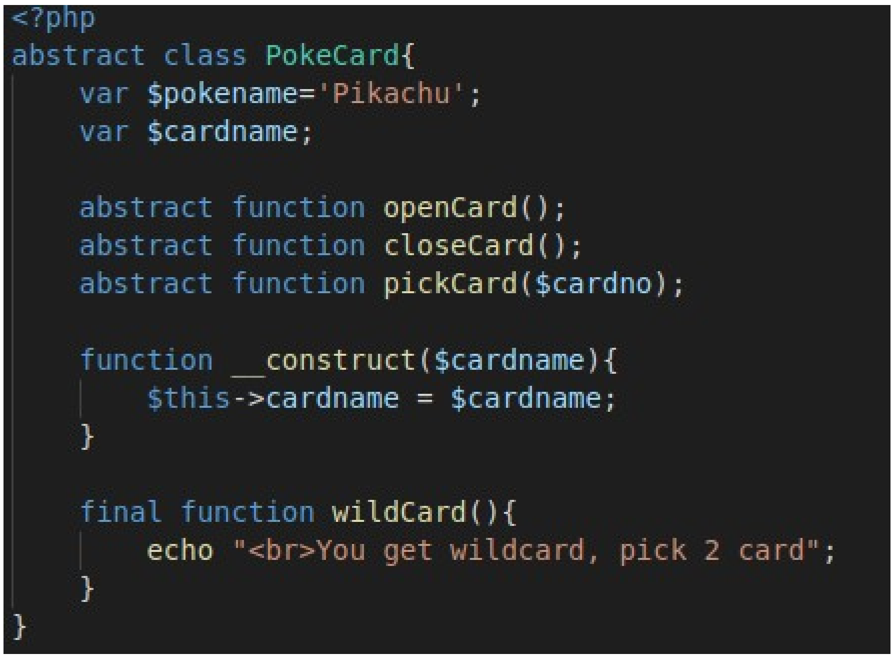


```php
$deck1 = new Deck();
$deck1->openCard();
$deck1->pickCard('1');
$deck1->wildCard();
```

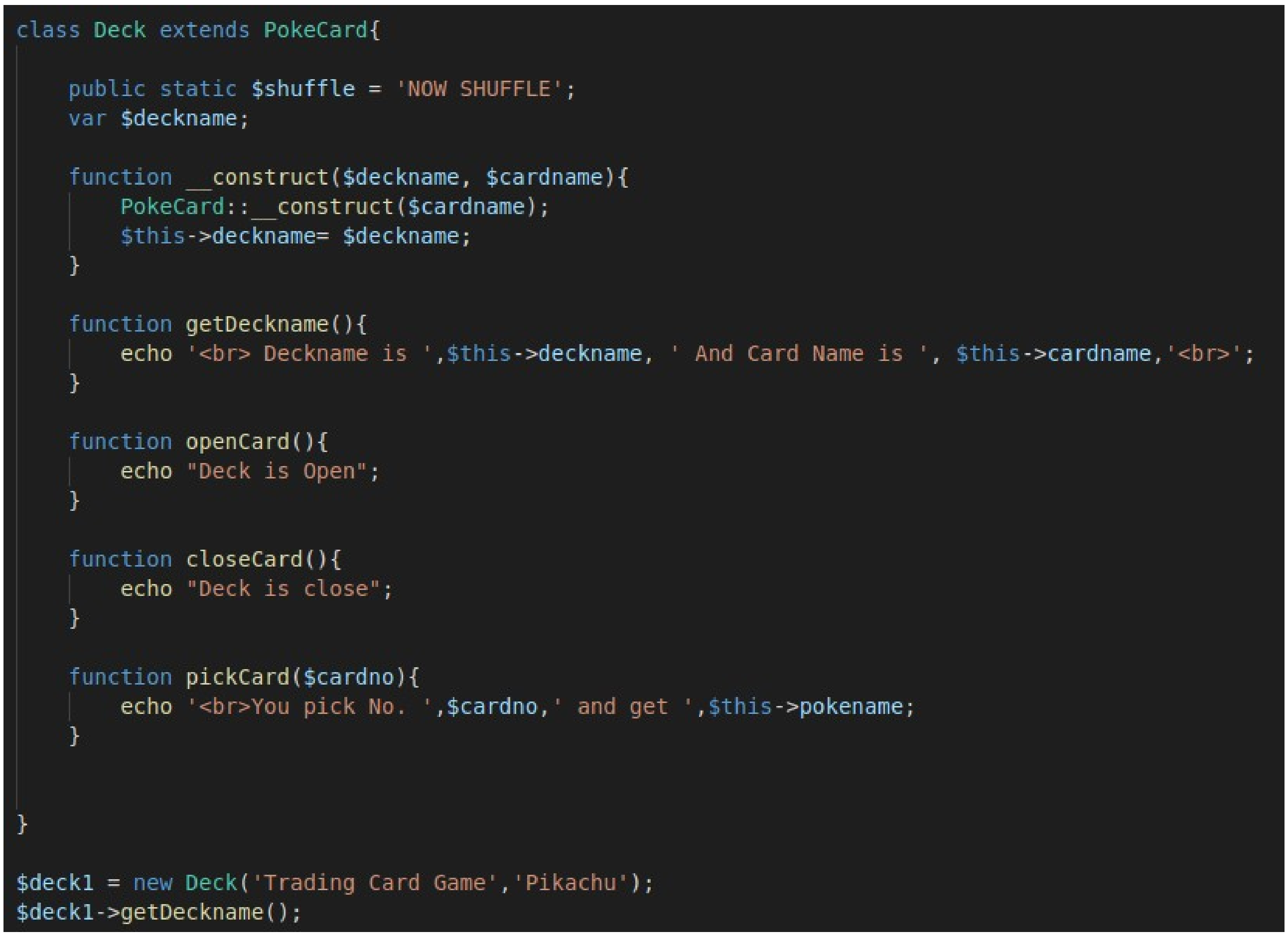
Ouput :

```
Fatal error: Cannot override final method PokeCard::wildCard() in
/opt/lampp/htdocs/belajaroop/Abstract.php on line 32
```

13. Calling Parent constructor



```php
<?php
abstract class PokeCard{
    var $pokename='Pikachu';
    var $cardname;

    abstract function openCard();
    abstract function closeCard();
    abstract function pickCard($cardno);

    function __construct($cardname){
        $this->cardname = $cardname;
    }

    final function wildCard(){
        echo "<br>You get wildcard, pick 2 card";
    }
}
```



```php
class Deck extends PokeCard{

    public static $shuffle = 'NOW SHUFFLE';
    var $deckname;

    function __construct($deckname, $cardname){
        PokeCard::__construct($cardname);
        $this->deckname= $deckname;
    }

    function getDeckname(){
        echo '<br> Deckname is ',$this->deckname, ' And Card Name is ', $this->cardname,'<br>';
    }

    function openCard(){
        echo "Deck is Open";
    }

    function closeCard(){
        echo "Deck is close";
    }

    function pickCard($cardno){
        echo '<br>You pick No. ',$cardno,' and get ',$this->pokename;
    }


}

$deck1 = new Deck('Trading Card Game','Pikachu');
$deck1->getDeckname();
```

Output:

```
Deckname is Trading Card Game And Card Name is Pikachu
```