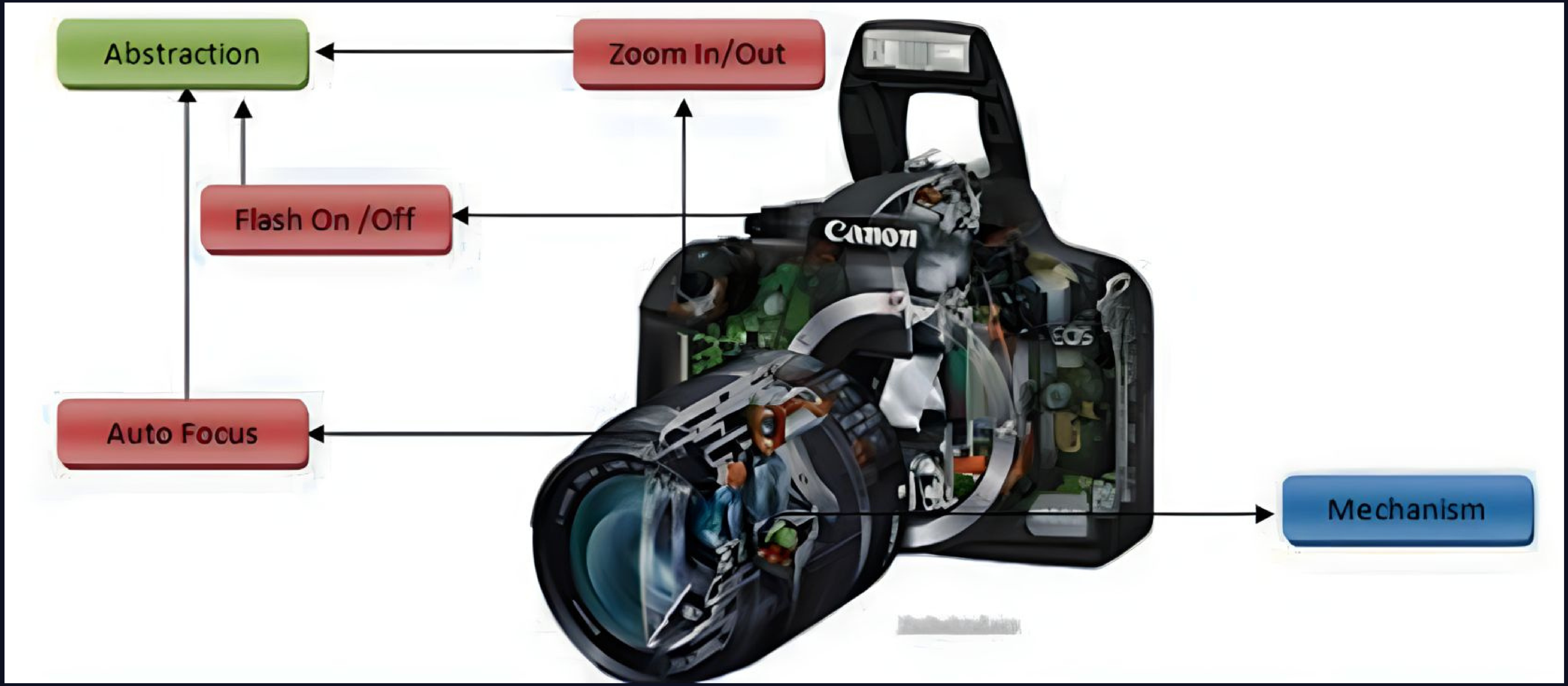
OBJECT ORIENTED PROGRAMMING

Abstraction & Polymorphism

What is an Abstraction ?

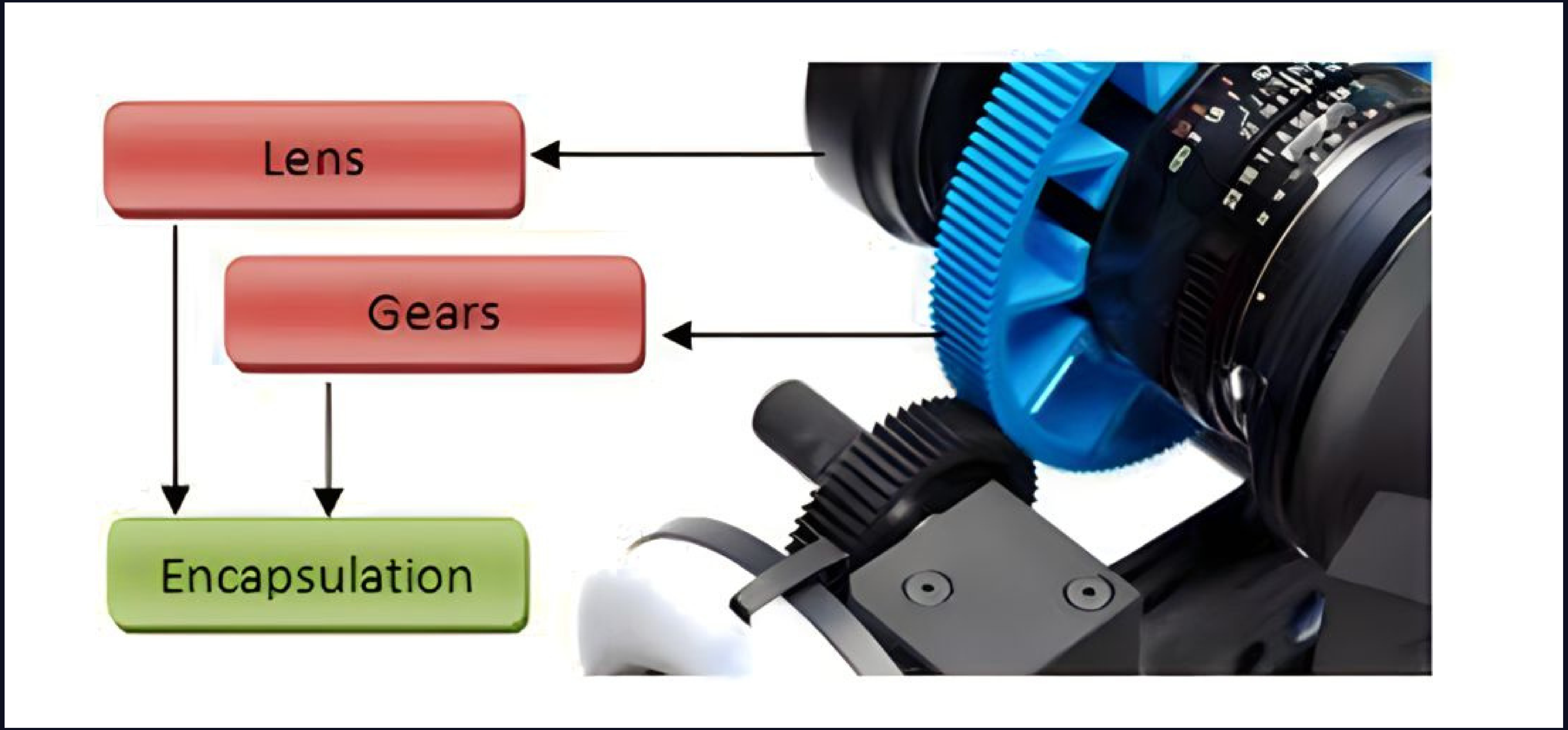
- Abstraction is the any representation of data in which the implementation details are hidden (abstracted).
- It's An OOP Concept That Allow Us To hide Implementation
- It's Focus On WHAT Not On HOW
- The main purpose of abstraction is hiding the unnecessary details from the users
- solves the issues at the design level.
- We can Achieve abstraction By : Interface and Abstract Class

Example ...


Abstraction
Zoom In/Out
Flash On /Off
Auto Focus
Canon
EOS
Mechanism

It's Like Encapsulation Hamm...

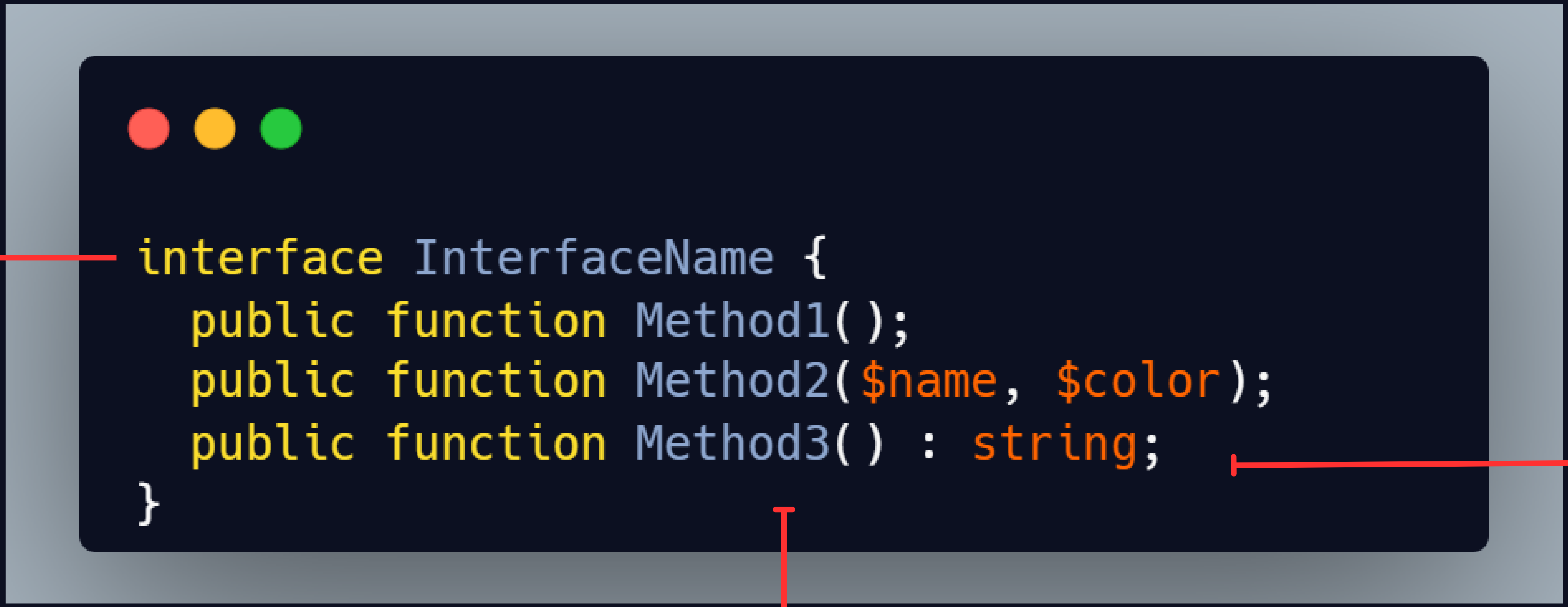
Encapsulation is simply combining the data members and functions into a single entity called an object.



Interface

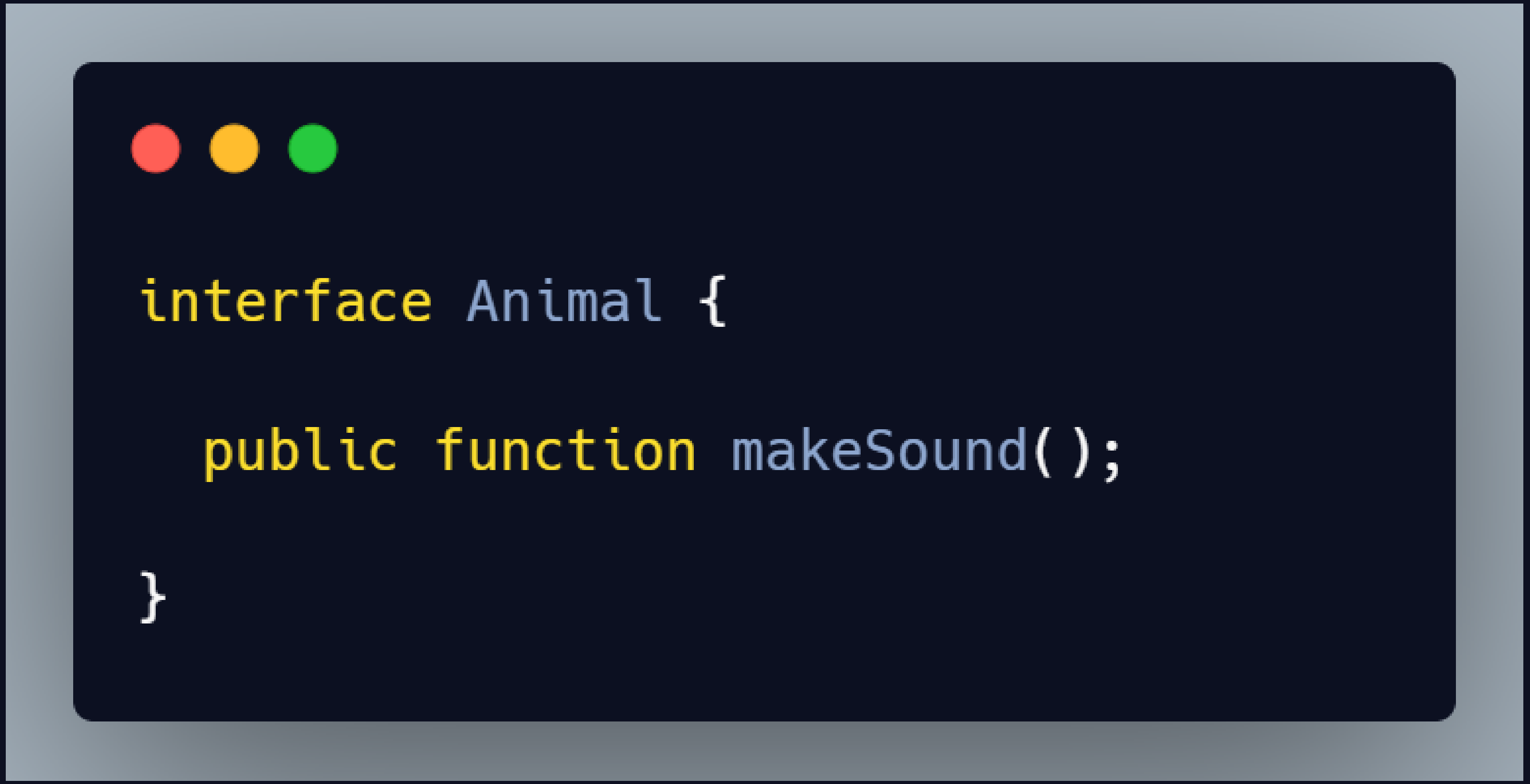
- All Method in Interface are Abstracted (No Body)
- Can't Define Properties In it in PHP But java Can
- To declare an interface we use interface keyword
- To make a class inherit from an interface we use implements keyword
- Can't create an object from the interface
- When one or more classes use the same interface, it is referred to as "**Polymorphism**"
- It's a relation called " Anything can do can use "

Code ...



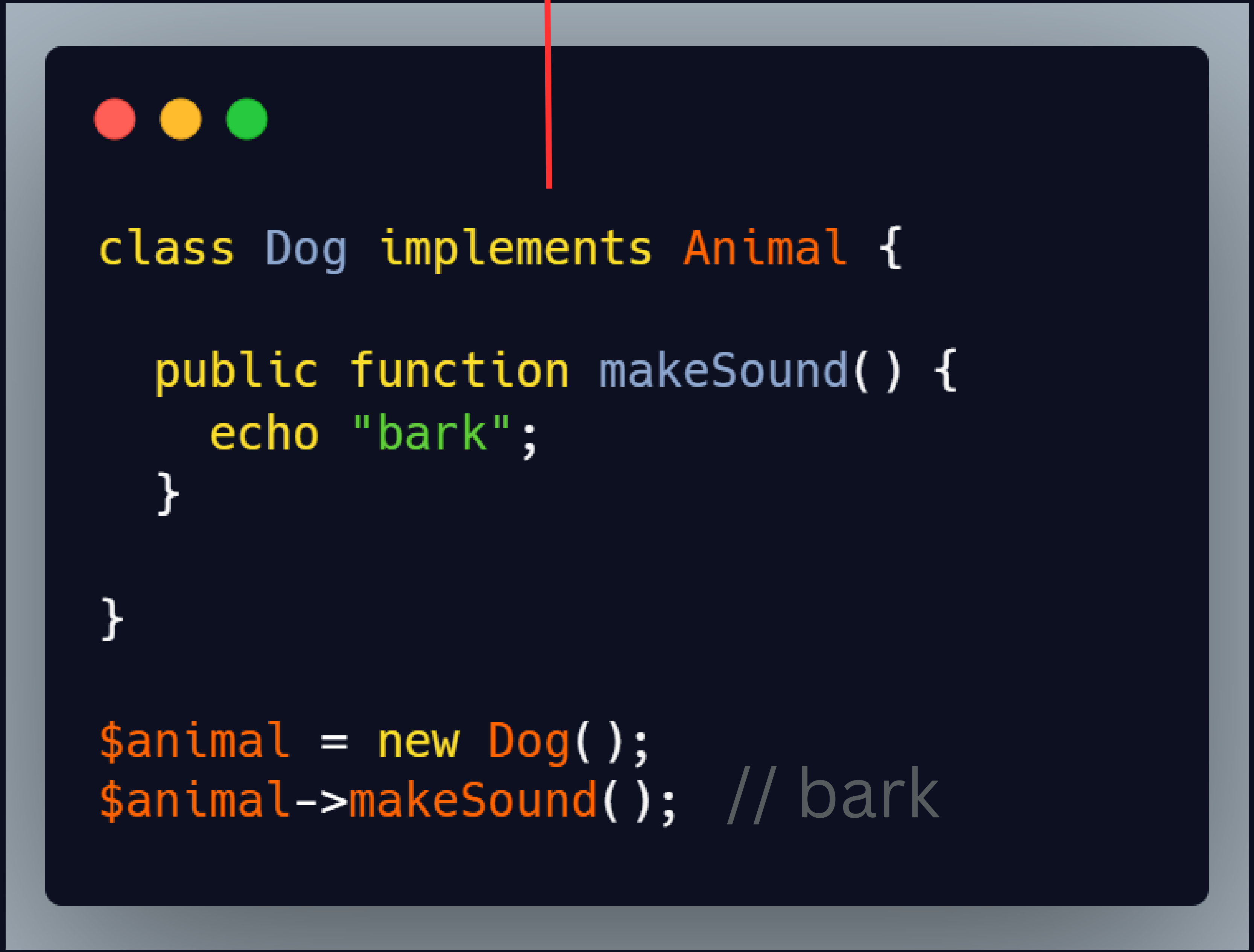
```
interface InterfaceName {
  public function Method1();
  public function Method2($name, $color);
  public function Method3() : string;
}
```

Example ...



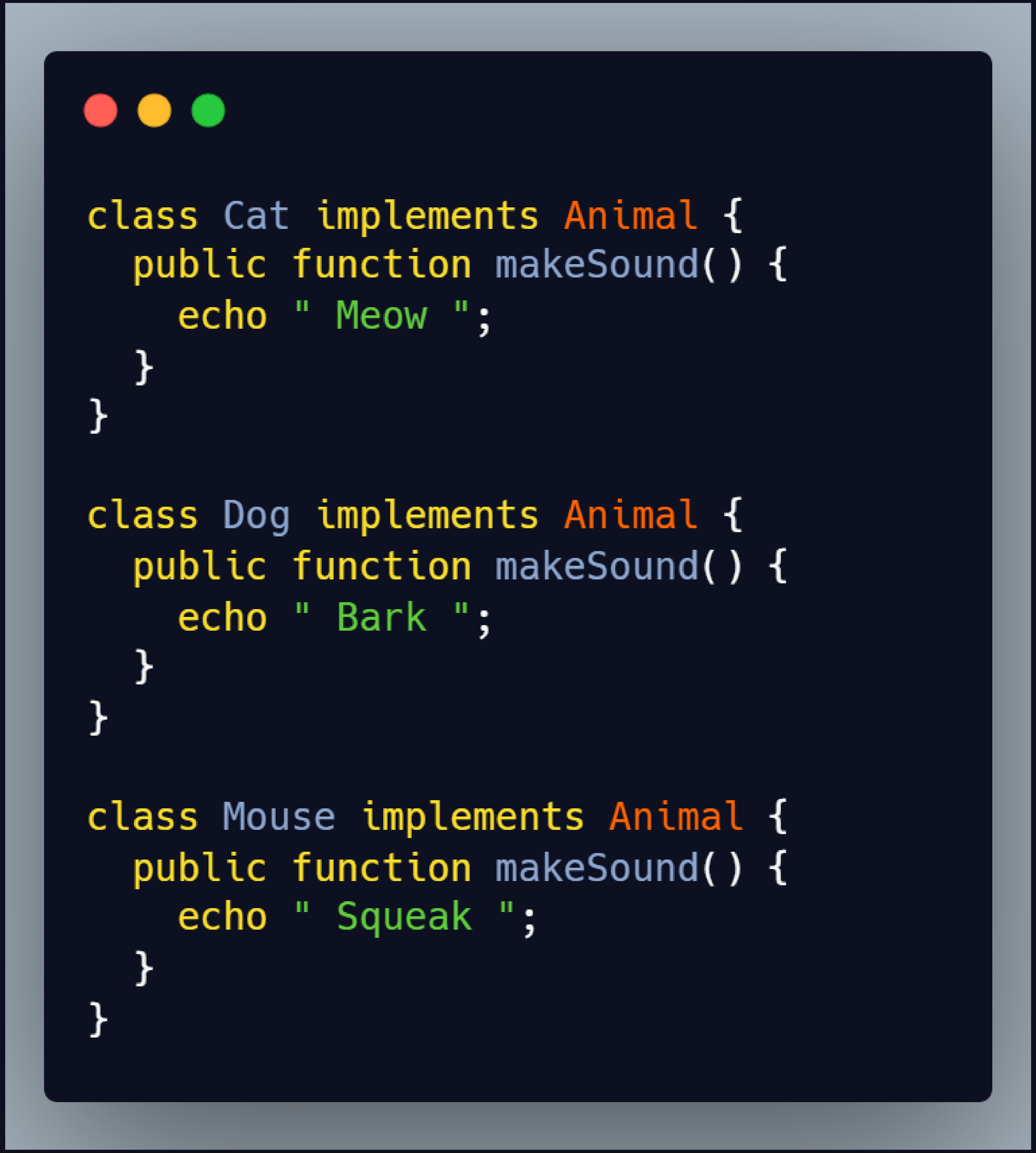
```php
interface Animal {

  public function makeSound();

}
```

Keyword To use The interface



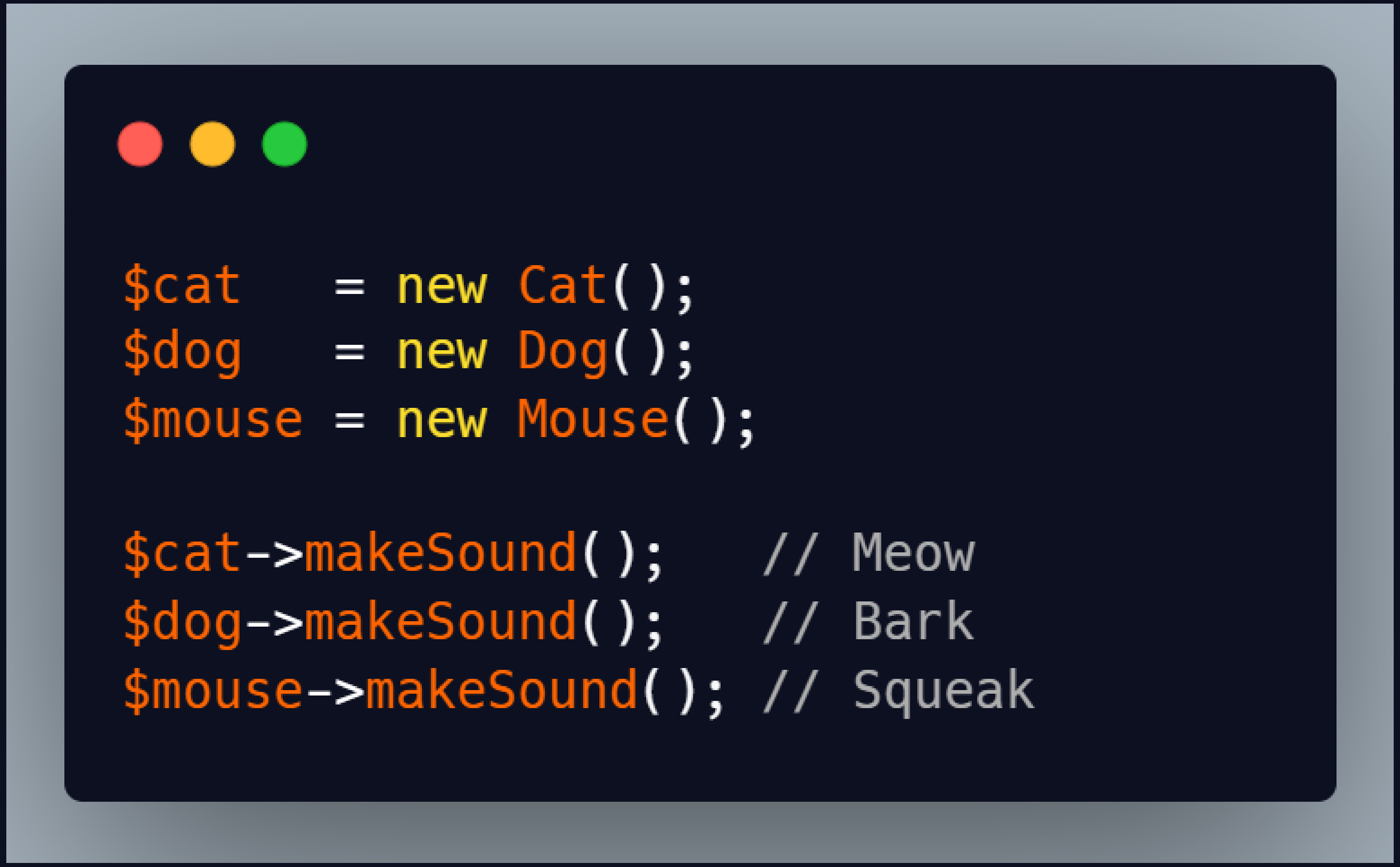
```php
class Dog implements Animal {

  public function makeSound() {
    echo "bark";
  }

}

$animal = new Dog();
$animal->makeSound();  // bark
```

Example ...



```php
class Cat implements Animal {
  public function makeSound() {
    echo " Meow ";
  }
}

class Dog implements Animal {
  public function makeSound() {
    echo " Bark ";
  }
}

class Mouse implements Animal {
  public function makeSound() {
    echo " Squeak ";
  }
}
```

In This Case i Achieved Polymorphism
But Will Take about it later

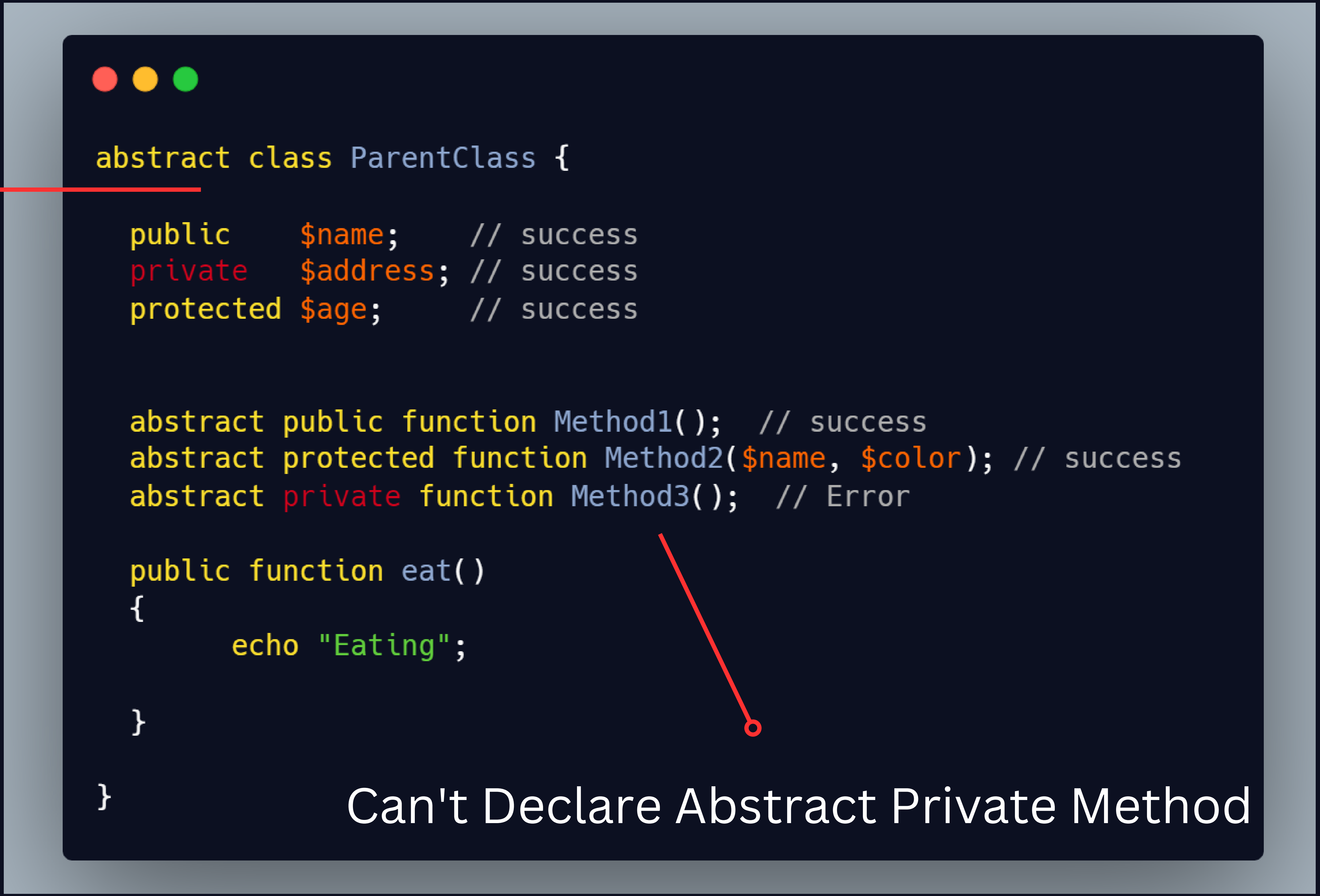


```php
$cat   = new Cat();
$dog   = new Dog();
$mouse = new Mouse();

$cat->makeSound();   // Meow
$dog->makeSound();   // Bark
$mouse->makeSound(); // Squeak
```

Abstract Class

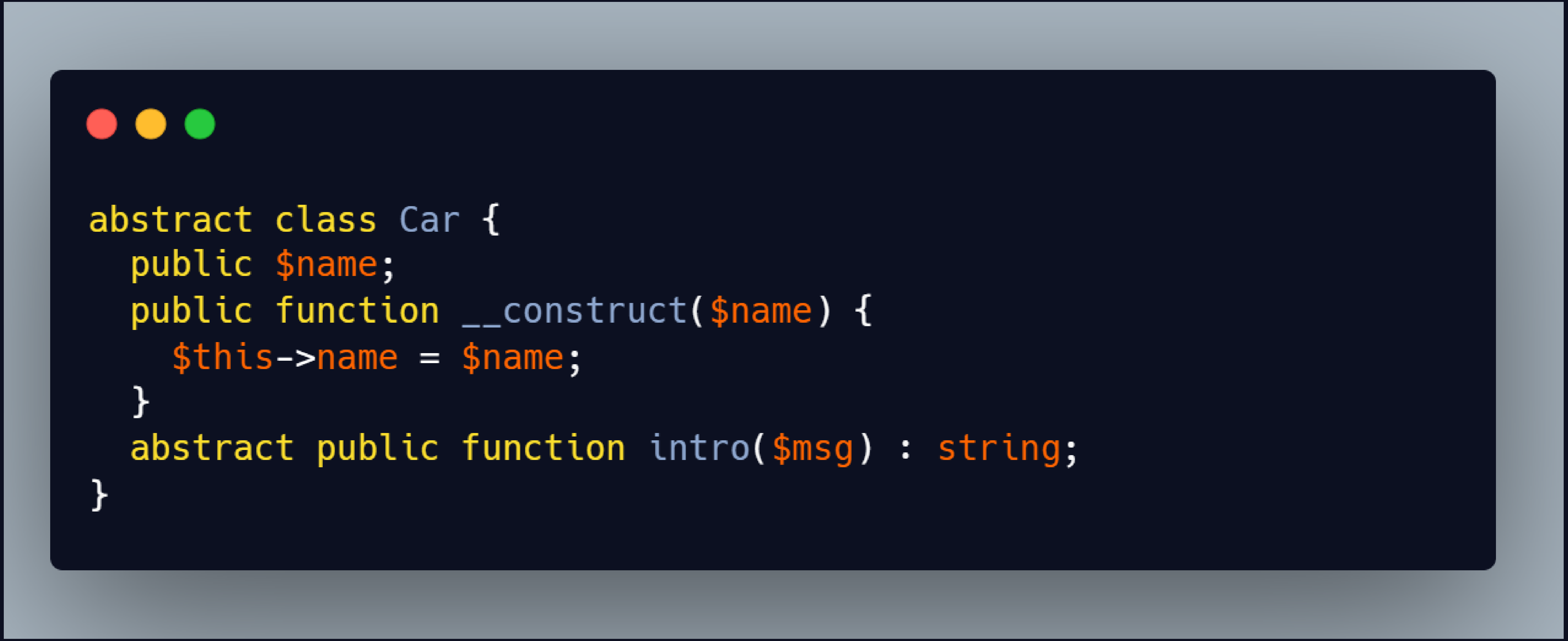
- It's Also Another Way To Achieve Abstraction
- Can Define Properties In it
- Methods Can Be Public Or Protected Not Private
- We Declare Abstract Class By Using Abstract Keyword
- It's a Normal Class But Has At Least One Abstract Method and Can Has Properties We Use extends Keyword to use Abstract Class
- The child class method must be defined with the same name, and the same or a less restricted access modifier Not Private
- the type and number of required arguments must be the same or add more

Code ...



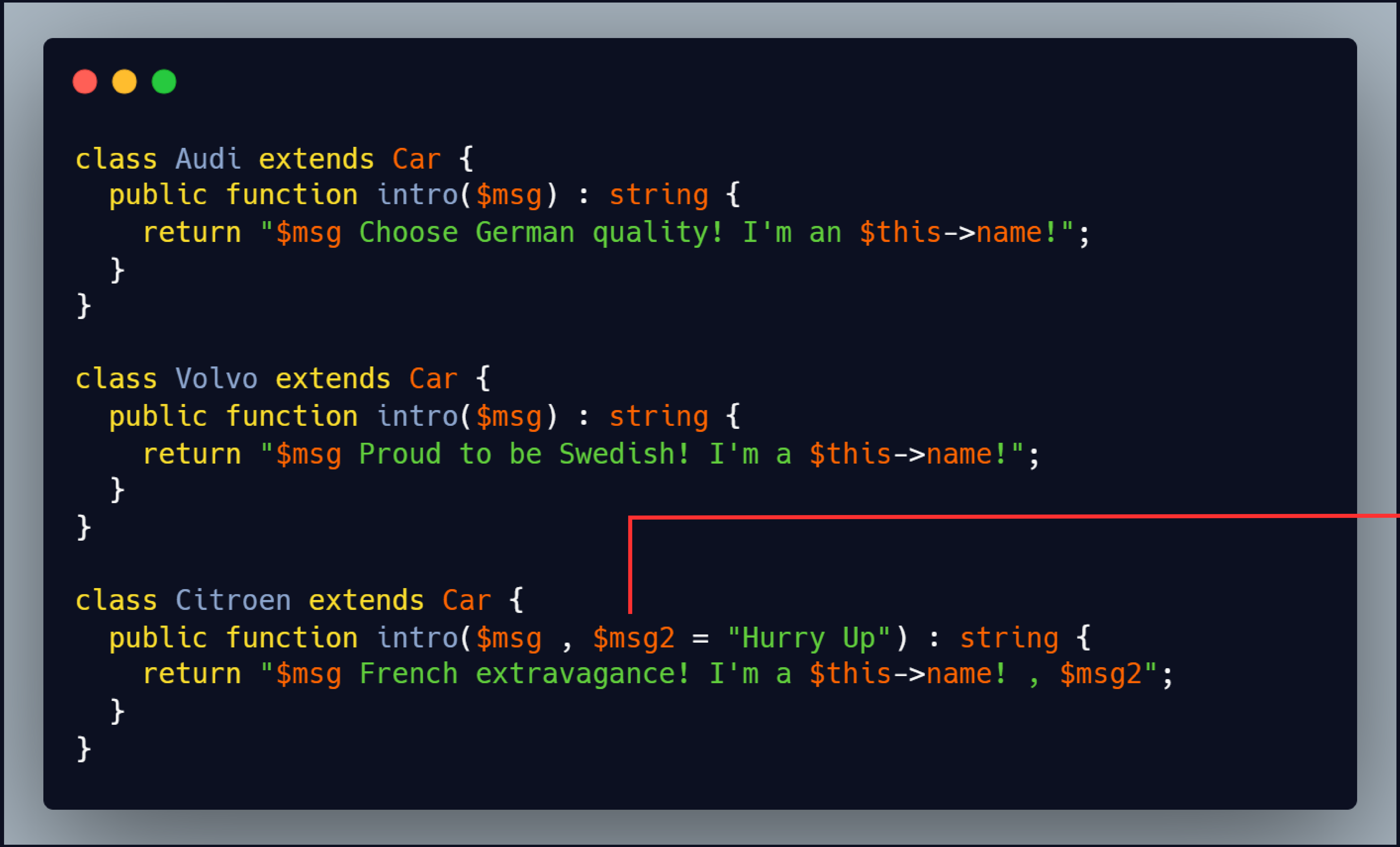
```php
abstract class ParentClass {

  public    $name;    // success
  private   $address; // success
  protected $age;     // success


  abstract public function Method1();  // success
  abstract protected function Method2($name, $color); // success
  abstract private function Method3();  // Error

  public function eat()
  {
        echo "Eating";

  }

}
```

Examble...



```
abstract class Car {
  public $name;
  public function __construct($name) {
    $this->name = $name;
  }
  abstract public function intro($msg) : string;
}
```

Examble Con...

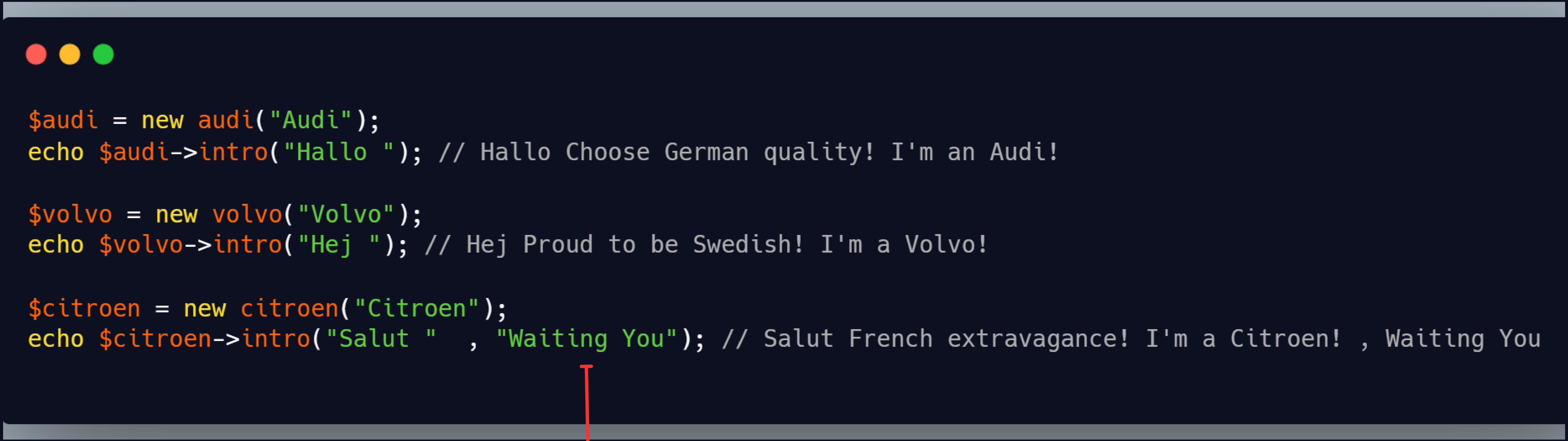


```php
class Audi extends Car {
  public function intro($msg) : string {
    return "$msg Choose German quality! I'm an $this->name!";
  }
}

class Volvo extends Car {
  public function intro($msg) : string {
    return "$msg Proud to be Swedish! I'm a $this->name!";
  }
}

class Citroen extends Car {
  public function intro($msg , $msg2 = "Hurry Up") : string {
    return "$msg French extravagance! I'm a $this->name! , $msg2";
  }
}
```

Use Extends Keyword To Use Abstract Class

Optional Arguments

Result



```
$audi = new audi("Audi");
echo $audi->intro("Hallo "); // Hallo Choose German quality! I'm an Audi!

$volvo = new volvo("Volvo");
echo $volvo->intro("Hej "); // Hej Proud to be Swedish! I'm a Volvo!

$citroen = new citroen("Citroen");
echo $citroen->intro("Salut "  , "Waiting You"); // Salut French extravagance! I'm a Citroen! , Waiting You
```

I Can Send This Parameter Or Not cauz it's Optional

Interface VS Abstract Class

Interface	Abstract Class
Interface class supports multiple inheritance feature	Abstract class does not support multiple inheritances.
Doesn't contain a data member.	Contain a data member.
Contains incomplete members which refer to the signature of the member.	Contains both incomplete(i.e. abstract) and complete members.
Abstract methods Must be Pablic	Abstract Methods can be Public or Protrctrd But Not Private

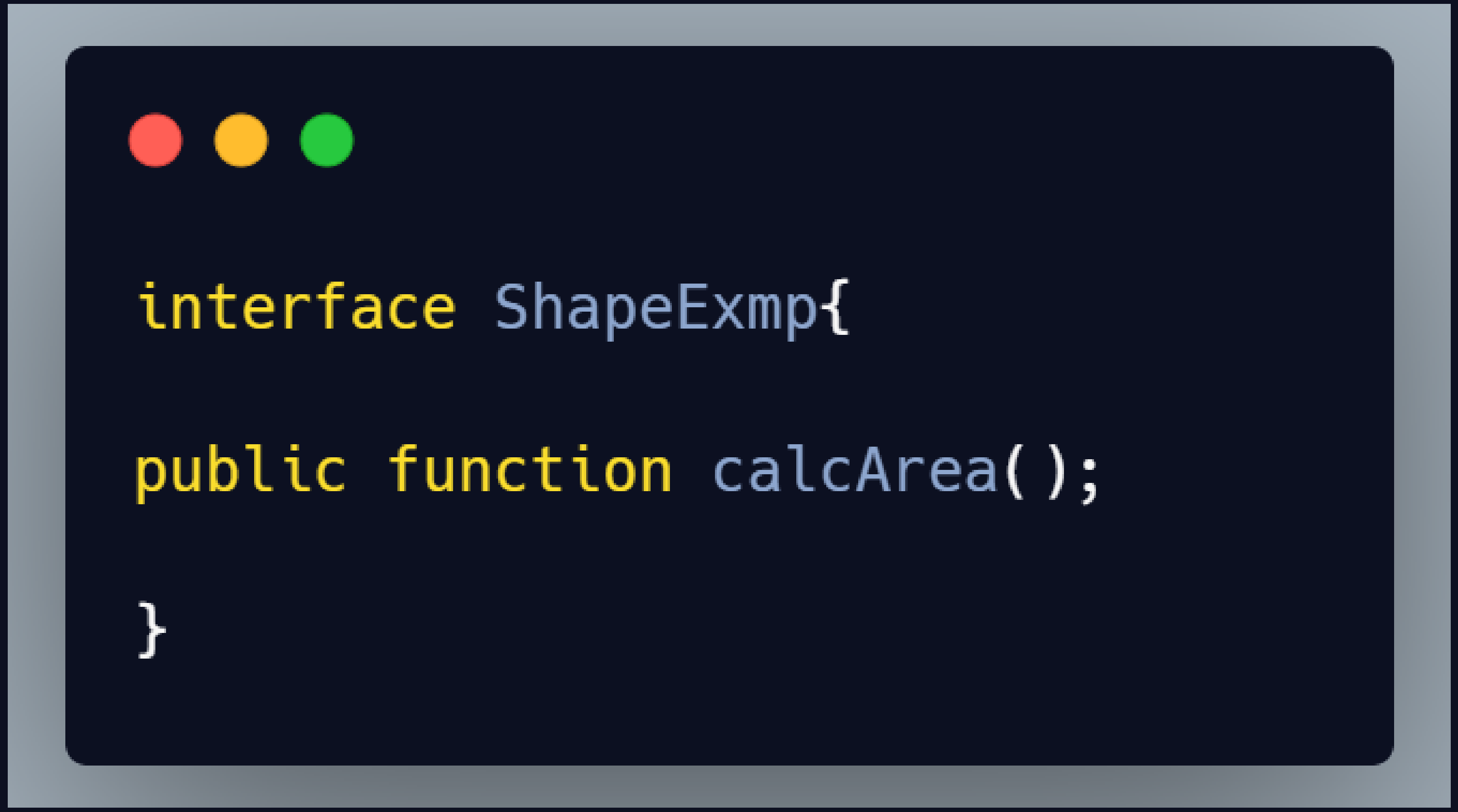
POLYMORPHISM

What is a Polymorphism?

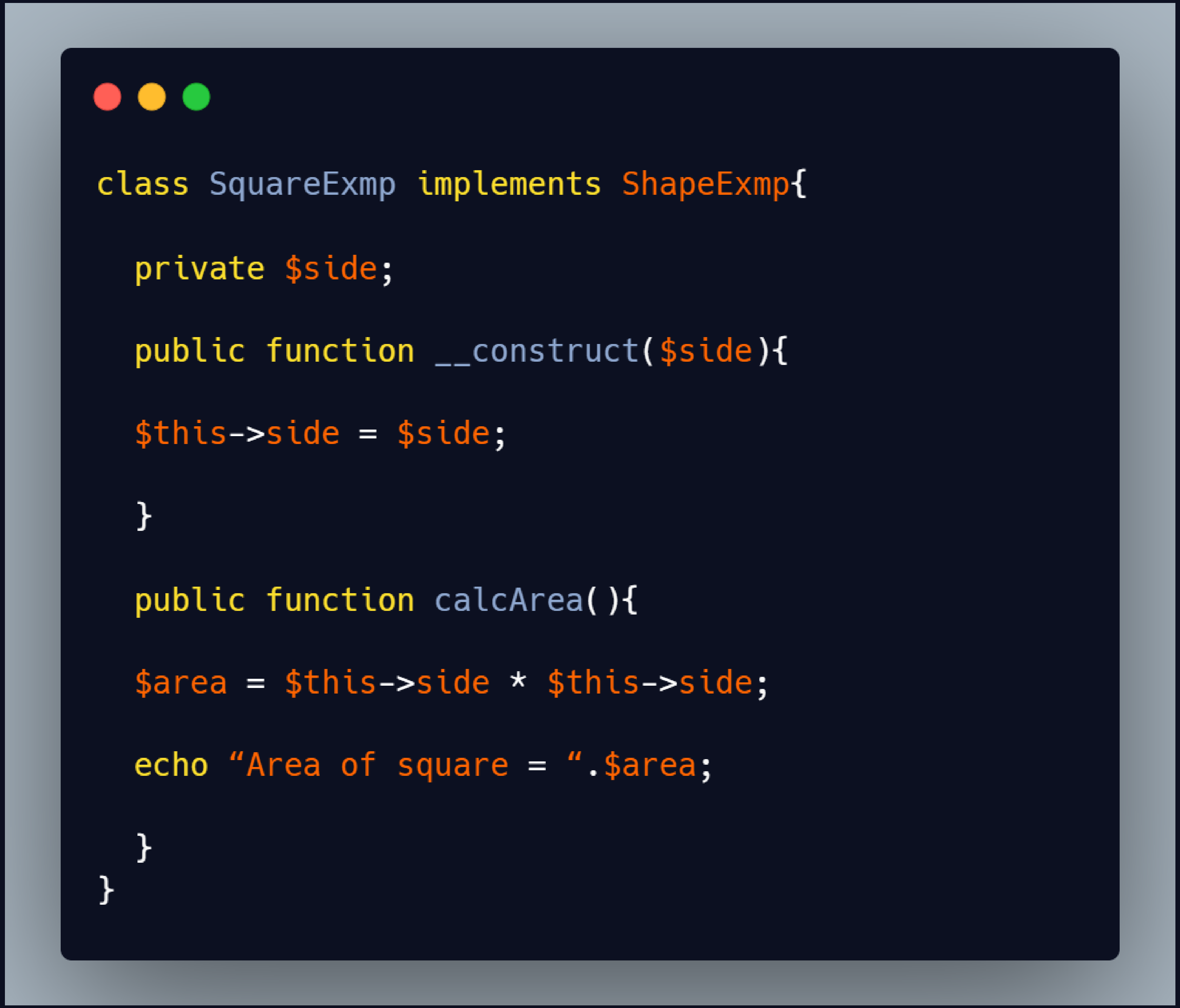
- Polymorphism derived from Greek words poly meaning many and morphism meaning forms
- Allows you to create classes with different functionalities in a single interface.
- There are two types of Polymorphism
  a. Compile time polymorphism also Know as function overloading
     - PHP Doesn't Support it
  b. Run time polymorphism also Know as function overriding
- You Can Achieve Polymorphism in PHP in 2 way
  c. Interface
  d. Abstract Class

Example

Code ...

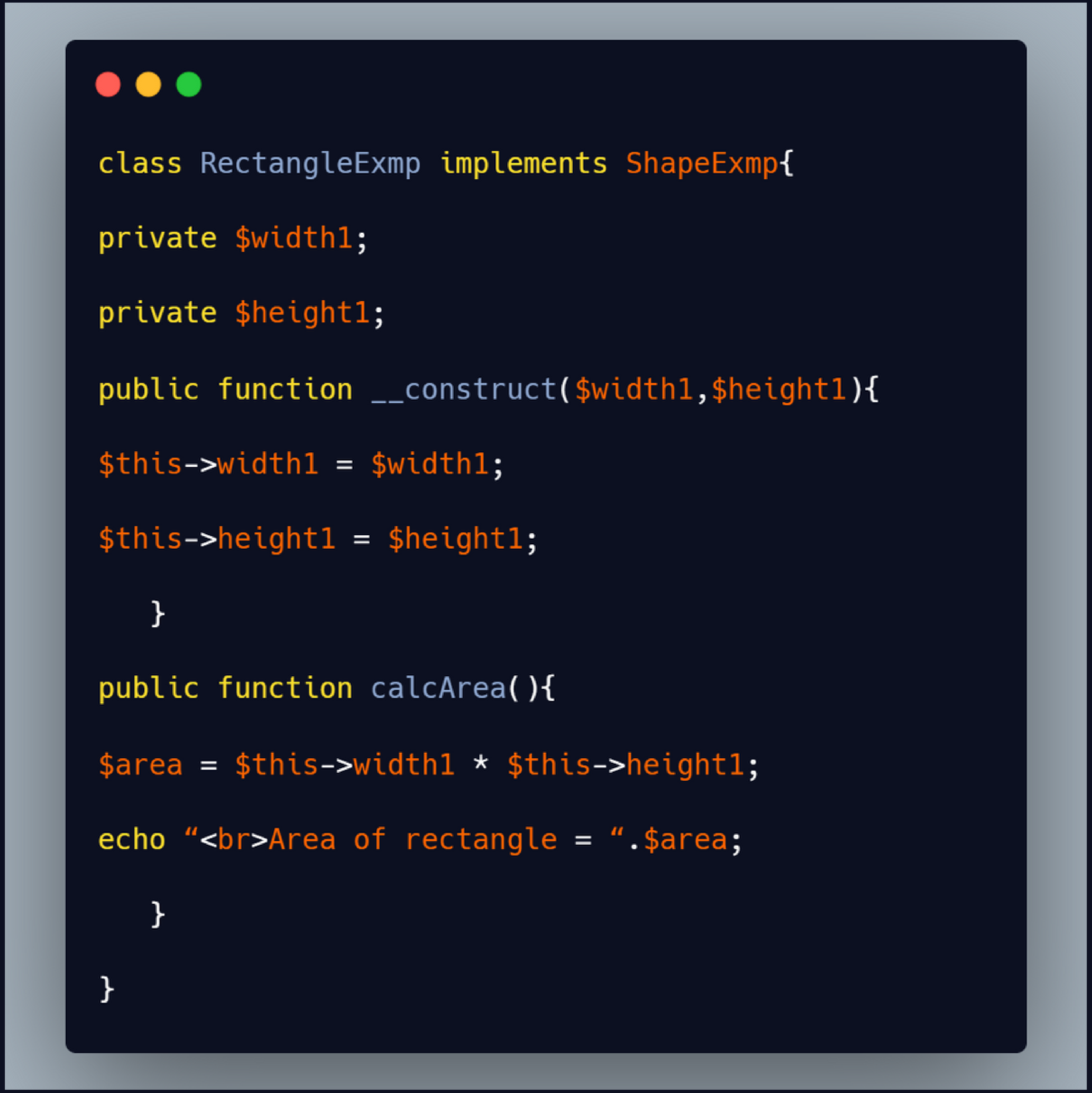


```
interface ShapeExmp{

public function calcArea();

}
```

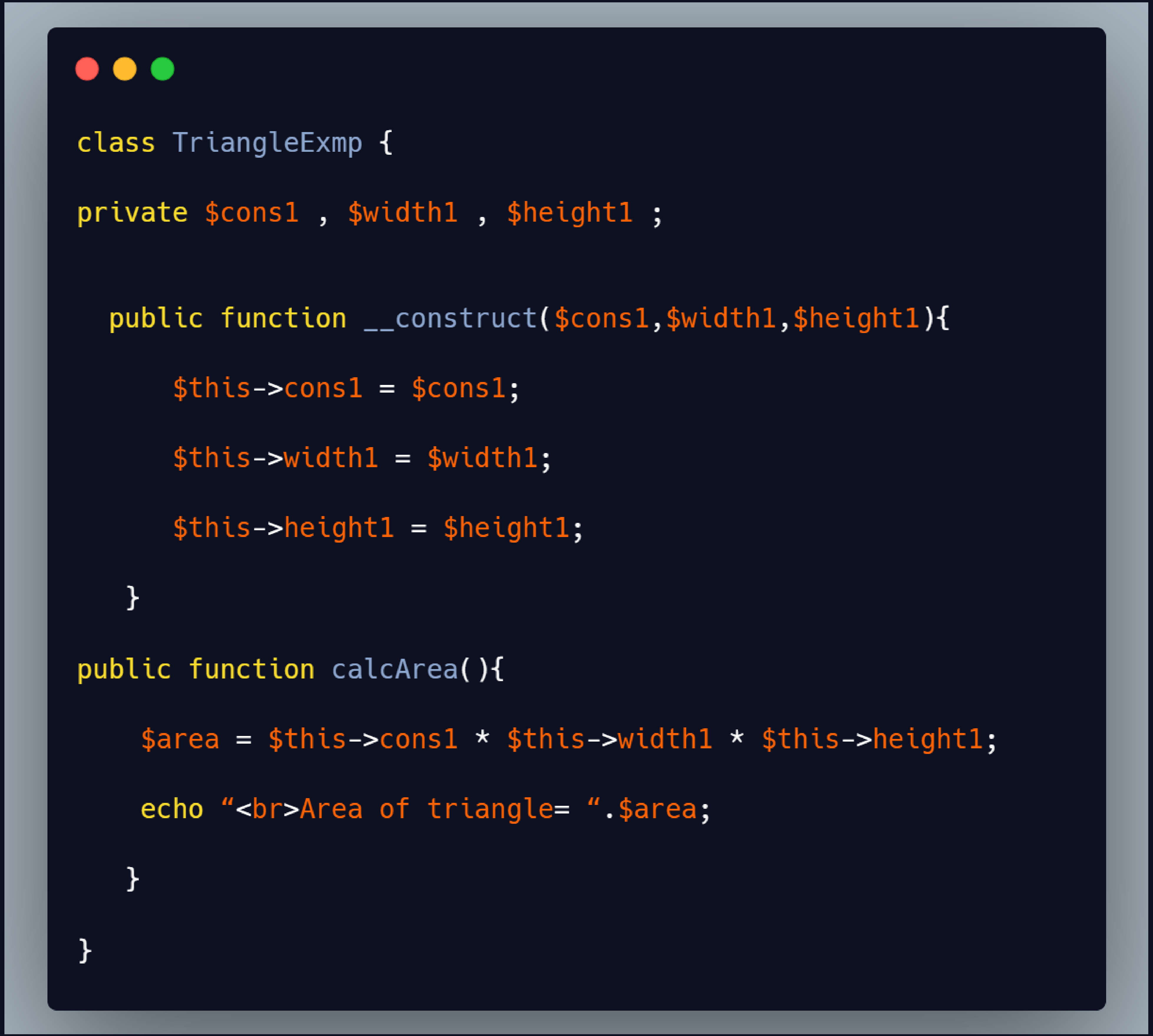


```
class SquareExmp implements ShapeExmp{

  private $side;

  public function __construct($side){

  $this->side = $side;

  }

  public function calcArea(){

  $area = $this->side * $this->side;

  echo “Area of square = “.$area;

  }
}
```

Code ...

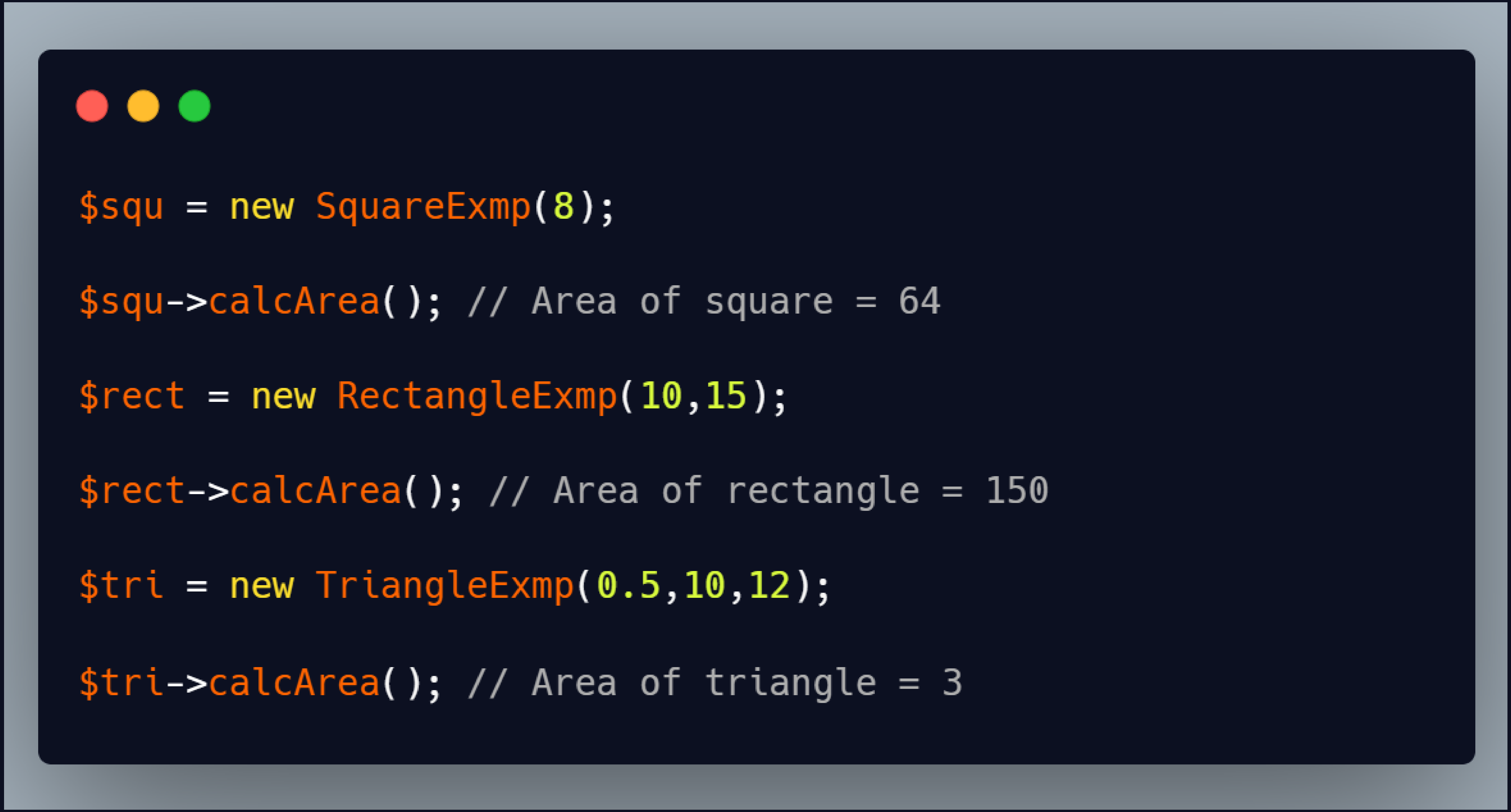


```
class RectangleExmp implements ShapeExmp{

private $width1;

private $height1;

public function __construct($width1,$height1){

$this->width1 = $width1;

$this->height1 = $height1;

   }

public function calcArea(){

$area = $this->width1 * $this->height1;

echo “<br>Area of rectangle = “.$area;

   }

}
```



```
class TriangleExmp {

private $cons1 , $width1 , $height1 ;


  public function __construct($cons1,$width1,$height1){

      $this->cons1 = $cons1;

      $this->width1 = $width1;

      $this->height1 = $height1;

   }

public function calcArea(){

    $area = $this->cons1 * $this->width1 * $this->height1;

    echo “<br>Area of triangle= “.$area;

   }

}
```

Result ...



```
$squ = new SquareExmp(8);

$squ->calcArea(); // Area of square = 64

$rect = new RectangleExmp(10,15);

$rect->calcArea(); // Area of rectangle = 150

$tri = new TriangleExmp(0.5,10,12);

$tri->calcArea(); // Area of triangle = 3
```

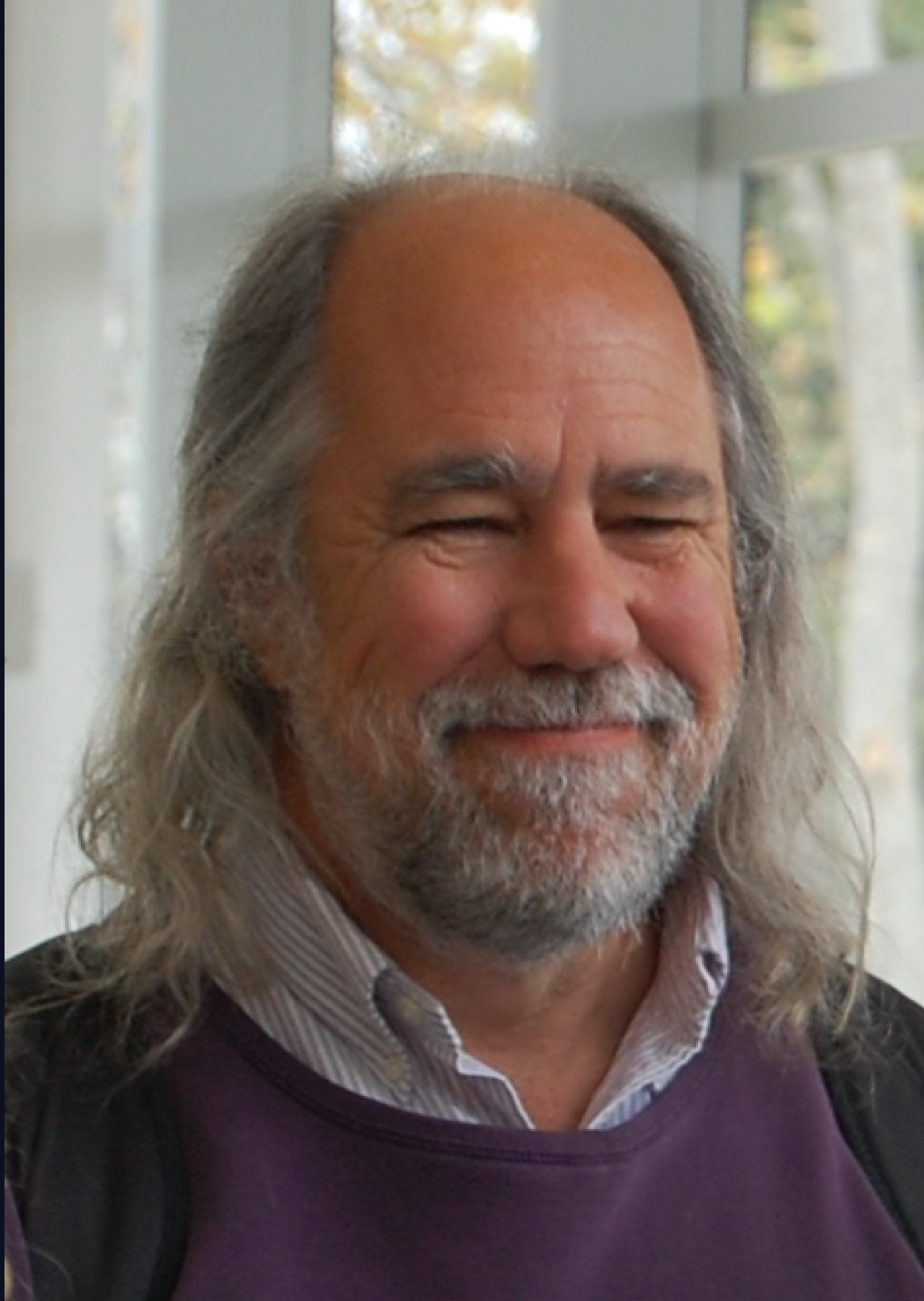
Resources

- upGrad
- C# Corner
- W3 School
- Java Point
- PHP Manual
- PHP Tutorial

Quote

The function of a good software is to make the complex appear to be simple

Grady Booch



Thank You

A b d e l r a h m a n A b d u l l a h