MEC ACADEMY

Object oriented programming

MEC Academy

Agenda

- Final keyword.
- Abstraction.
- abstract class & method.
- Interface.
- abstract class VS interface.
- Traits.
- Namespace.

MEC ACADEMY

Final Keyword

Final Keyword

- **The final keyword** prevents child classes from overriding a method or constant by prefixing the definition with final.
- If the class itself is being defined **final** then ***it cannot be extended***.

MEC ACADEMY

Final Keyword

Examples:

```php
<?php
class BaseClass {
   public function test() {
       echo "BaseClass::test() called\n";
   }

   final public function moreTesting() {
       echo "BaseClass::moreTesting() called\n";
   }
}

class ChildClass extends BaseClass {
   public function moreTesting() {
       echo "ChildClass::moreTesting() called\n";
   }
}
// Results in Fatal error: Cannot override final method BaseClass::moreTesting()
?>
```

MEC ACADEMY

Final Keyword

Examples:

```php
<?php
final class BaseClass {
   public function test() {
       echo "BaseClass::test() called\n";
   }

   // As the class is already final, the final keyword is redundant
   final public function moreTesting() {
       echo "BaseClass::moreTesting() called\n";
   }
}

class ChildClass extends BaseClass {
}
// Results in Fatal error: Class ChildClass may not inherit from final class (BaseClass)
?>
```

MEC ACADEMY

Abstraction

ABSTRACTION

Abstraction

Abstraction is a process of hiding the implementation details and showing only functionality to the user.

Abstraction can be achieved by **abstract class and method** or **interface**

MEC ACADEMY

abstract class

is a restricted class that cannot be used to create objects (to access it, it must be inherited from another class).

abstraction level (0 → 100).

abstract method

- **Abstract method**: can only be used in an abstract class, and it does not have a body.

- The body is provided by the subclass (inherited from).

MEC ACADEMY

Rules for PHP Abstract class

1. An abstract class must be declared with an abstract keyword.
2. It can have abstract and non-abstract methods.
3. It cannot be Instantiated.
4. It can have final methods
5. It can have constructors and static methods also.

MEC ACADEMY

interface

- An **interface in PHP** is declared by using the **interface keyword**.
- It provides total abstraction, because all the methods in an interface are declared that don’t have body .
- abstraction level (100%).
- Can’t create instance(object) from interface.

MEC ACADEMY

Interface properties

- The PHP compiler adds public and abstract keywords before the interface method.
- The PHP compiler adds public, static and final keywords before data members.
- Interface can’t have data members(variables) it has methods only.
- But can have constant variables.

MEC ACADEMY

Interface implementation

- In this example, the Animal interface has only one method. Its implementation is provided by Cat and Dog classes .

- Interface syntax

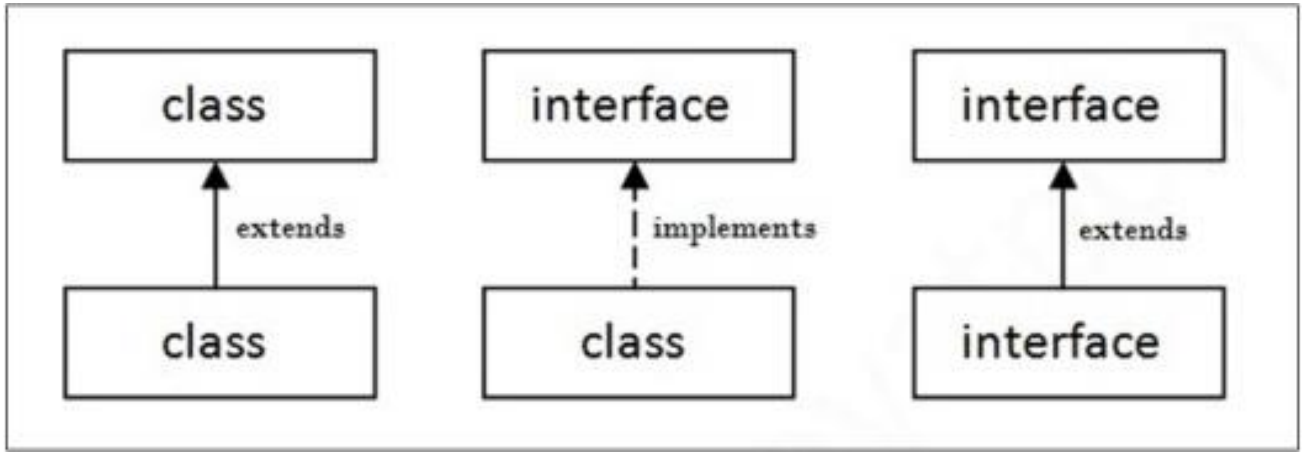
```php
<?php
interface InterfaceName {
  public function someMethod1();
  public function someMethod2($name, $color);
  public function someMethod3() : string;
}
?>
```

```php
<?php
// Interface definition
interface Animal {
  public function makeSound();
}

// Class definitions
class Cat implements Animal {
  public function makeSound() {
    echo " Meow ";
  }
}

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

MEC ACADEMY

Relationship between classes and interfaces

- A class extends another class, an interface extends another interface, but a **class implements an interface**.



Interface uses

- It is used to achieve abstraction
- We can support multiple inheritance

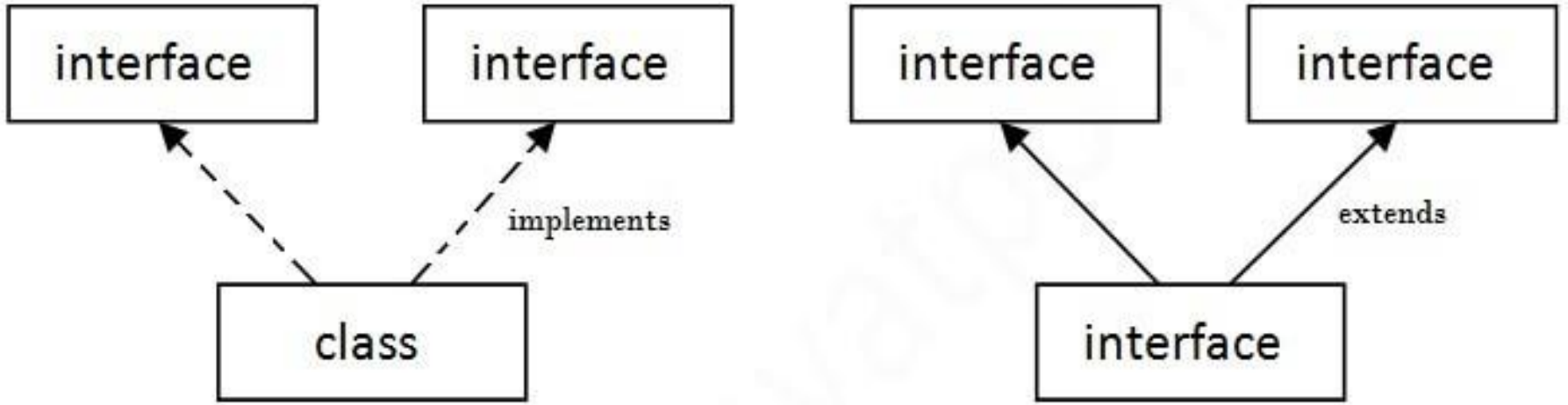
MEC ACADEMY

Multiple inheritance

- In PHP you can’t extend a class from multiple classes due to the program confusion and more likely to cause error PHP doesn’t support multiple inheritance although other languages support it like C++,python...
- If a class implements multiple interfaces, or an interface extends multiple interfaces, it is known as multiple inheritance.

MEC ACADEMY

Multiple inheritance using interfaces



Multiple Inheritance in

Abstract class VS interface

Interface	Abstract class
Interface support multiple inheritance	Abstract class does not support multiple inheritance
Interface does'n Contains Data Member	Abstract class contains Data Member
Interface does'n contains Cunstructors	Abstract class contains Cunstructors
An interface Contains only incomplete member (signature of member)	An abstract class Contains both incomplete (abstract) and complete member
An interface cannot have access modifiers by default everything is assumed as public	An abstract class can contain access modifiers for the subs, functions, properties
Member of interface can not be Static	Only Complete Member of abstract class can be Static

To read more about difference click here…

MEC ACADEMY

Traits

MEC ACADEMY

Traits

- **Traits** are a mechanism for code reuse in single inheritance languages such as PHP.
- Problem with extending classes is that you can only extend one. This is a little limiting.
- With traits its possible for PHP classes to inherit methods and prosperities from multi sources.
- You can’t extend or implement trait.
- You can’t instantiate trait (can’t create object of it).
- Its support classes not replacing it.
- Can have methods but can’t have __construct.
- Has priority over class.
- **Traits** Like Classes in syntax but use key word **trait** instead of class

namespace

- In PHP, namespaces are a way to organize and encapsulate code elements such as classes, interfaces, functions, and constants. They provide a way to avoid naming conflicts between different parts of your code or between your code and third-party libraries.

Next Session

- ❑ Database Fundamental
- ❑ Database Design.

MEC ACADEMY

THANKS
Any questions?

MEC ACADEMY