web разработка

ООП Полиморфизъм
Абстрактни класове
Интерфейси

Съдържание

- isA, hasA - връзки между класовете - *преговор*
- Спесификаторите за достъп при наследяване - *преговор*
- Final
- Абстрактни класове и методи
- Интерфейси

final

final

Когато декларацията на метод започва с ключовата дума **final**, този метод не може да бъде предефиниран в класовете наследници.

```
class Test {

....

 final public function moreTesting() {

      echo "I am a final method and cannot be overriden!";

  }

...

}
```

final

Когато клас е означен като **final** - не може да бъде наследяван.

final

```
final class BaseClass {
   public function test() {
      //
   }

   // Here it doesn't matter if you specify the function as
final or not
   final public function moreTesting() {
      //
   }
}
```

```
class ChildClass extends BaseClass {

}

// Results in Fatal error: Class ChildClass may not inherit
from final class (BaseClass)
```

Абстрактни класове и методи

Абстрактни класове и методи

Логиката, която се съдържа в група класове на приложението, може да бъде отделена в абстрактен клас, който да бъде наследяван от тези класове.

Условието е - *от абстрактния клас не можем да създаваме обекти*.

Абстрактния клас може да бъде чисто описателен - да посочим какви методи съдържа. Конкретната имплементация на методите реализираме в класа наследник, когато е различна за различните наследници.

...С абстрактния клас правим скица на обект, чийто точен чертеж се реализира в класовете наследници...

Абстрактни класове и методи - 2

```
abstract class AUser {
        public function get_user(){
    //описваме функционалността на метода, ако ще бъде една и съща за всички класове наследници
    }
    public function get_role(){
    //описваме функционалността на метода, ако ще бъде една и съща за всички класове наследници
    }
}
```

Абстрактни класове и методи - 3

```
class User extends AUser {

        public function get(){

        //ако get_user() е описана в абстрактния AUser,

        //предефинираме метода, при необходимост

        return $this->get_user();

    }

}
```

Абстрактни класове и методи - 4

Абстрактния клас може да съдържа абстрактни методи.

```
abstract class AUser {

….

abstract function can();      //няма описание на метода, защото е абстрактен и

                                  //ще бъде описан в класа/класовете наследник/наследници

…….

}
```

Абстрактни класове и методи - 5

Абстрактни методи - предназначение -

При проектиране на приложението, все още не е ясна конкретната реализация на даден метод, но е ясно че той трябва да присъства. Ще бъде описан в наследниците на абстрактния метод.

Абстрактният клас може да съдържа или не абстрактни методи.

Но **ако съдържа абстрактен метод**, **класът задължително е абстрактен.**

Интерфейси

Интерфейс

Абстрактен клас, който съдържа само абстрактни методи.

В него няма свойства.

Интерфейс - 2

```
interface iUser {
        public function get_user();
        public function get_role();
}
```

**В името, като добра практика поставяме i*

Интерфейс - 3

Интерфейсът задължително се *наследява**, но с ключовата дума ***implements.***

```
class User implements iUser {



}
```

**в смисъла на по-скоро присвояване на описаната в интерфейса функционалност, от даден клас*

Интерфейс - 3

- Класът, който използва интерфейса трябва задължително да опише метода, деклариран в интерфейса.
- В описанието на метода трябва задължително да фигурират параметрите, които са посочени в интерфейса.
- В интерфейса не трябва да присъстват private и protected методи
- Интерфейс може да наследява друг интерфейс с ключовата дума extends

Интерфейс - 4

Един клас може да използва *няколко интерфейса*.

```
class User implements iUser, iAdmin … {

}
```

*докато клас може да наследява само един клас