بسم الله الرحمن الرحيم

تعالي ياصديقي نشوف مشكله - هنشوف هي مشكله ولا لا - وهي ان الناس اللي كتبت ال php عملتها single inheritance language ودا عكس معظم اللغات اللي ال class فيها يقدر ي inherit من اكتر من class ف نفس الوقت .. طيب هم ليه يعملوا كدا اصلا ؟ هل لان ال multiple inheritance مكنش ظهر وقتها ؟ ف الحقيقه لا .. هم عملوها كدا لاسباب منها :

- ان ال single inheritance بيخلي الكود صغير وسهل انه يتفهم بسهوله ,, ع عكس ال multiple inheritance اللي ممكن يخلي الكود معقد جدا .
- لو انا بعمل inherent من اكتر من class وفيهم properties متشابهه ف الاسم او methods متشابهه ف دا ممكن يطلعلي bugs كتير انا ف غني عنها .

طبعا الكلام دا ف بدايه اللغه كان خفيف ولطيف لكن لمه اللغه بقت كبيره وليها مستخدمين كتير وال systems اللي ال users عايزين يبنوها بال php دا مكنتش حاجه لطيفه لانها كانت بتعمل لل user زي limitation بتخليه ممكن يفكر يغير اللغه اصلا عشان يقدر يعمل ال system بتاعه ,, عشان كدا ال php حلت المشكله دي بال trait .

<u>ايه هو بقي ال trait ؟</u>

ال trait هو حاجه شبه ال class – شبه ال class لاننا مبنقدرش ناخد منه object – بنسخدمه عشان نقدر نحل مشكله ال multiple inheritance الموجوده عندنا ف ال php .

نقدر ف ال trait نعمل :

- Properties بال access modifier اللي احنا عايزينه (public - protected - private) .
- كمان ممكن نعمل ال methods بتاعتنا ك اي class عادي .
- وال method ممكن نعملها abstract method واي class حابب يستخدم ال trait دا لازم يعمل implement لل method دي .
    - ولاحظ هنا اني قولت (يستخدم) مش يورث ,, ودا لان ال trait مش class نقدر نورثه لكن هو حاجه نقدر نستخدمها (use) .
- كمان نقدر نعدل ال access modifiers لل methods واحنا بنستخدم ال trait ف ال class .

```php
trait Greeting
{
    protected $name;

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

    public function sayHello($name)
    {
        return "Hello $name, I am $this->name";
    }

    abstract public function sayGoodBye();
}

trait Work
{
    private $job;

    public function job($job)
    {
        $this->job = $job;
    }

    protected function startWork()
    {
        return "Working as : $this->job";
    }
}
```

```php
class Person
{
    use Greeting, Work {
        Work::startWork as public;
    }

    public function sayGoodBye()
    {
        return 'good bye';
    }
}

$user = new Person('abdo');
echo $user->sayHello('ali'); // Hello ali, I am abdo
echo "\n";
$user->job('php backend developer');
echo $user->startWork(); // Working as : php backend developer
echo "\n";
echo $user->sayGoodBye(); // good bye
```

ف المثال دا انا عندي 2 trait وهنلاقي فيهم :

- Properties with different access modifiers .
- Methods and abstract methods .

ولمه استخدمنا ال traits دول ف ال class بتاعنا :

- استخدمنا ال use عشان نقدر نستدعي ال trait بتاعنا جوه ال class .
- كمان قدرنا نغير ال access modifier بتاع ال startWork method .
- كمان كان لازم نعمل implement لل abstract method اللي اسمها sayGoodBye .