Curso POO PHP Interfaces

Interfaces

Un interface é como unha clase baleira que soamente contén declaracións de métodos. Defínense utilizando a palabra **interface**.

Por exemplo, antes viches que podías crear novas clases herdadas de Produto, como TV ou Ordenador. Tamén viches que nas subclases podías redefinir o comportamento dos métodos para que xerase unha saída en HTML diferente para cada tipo de produto. Se queres asegurarte de que todos os tipos de produtos teñan un método de nome **mostra_codigo**, podes crear un interface como o seguinte.

```
interface iMostra {
    public function mostra_codigo();
}
```

E cando crees as subclases deberás indicar coa palabra **implements** que teñen que implementar os métodos declarados neste interface.

```
class TV extends Produto implements iMostra {
    ?
    public function mostra_codigo() {
        print "<p>" . $this->codigo . "</p>";
    }
    ?
}
```

Todos os métodos que se declaren nun interface deben ser públicos. E ademais de métodos, os interfaces poderán conter constantes pero non atributos.

Un interface é como un contrato que a clase debe cumprir. Ao implementar todos os métodos declarados no interface asegúrase a interoperabilidade entre clases. Se sabes que unha clase implementar un interface determinado, sabes que nome teñen os seus métodos, que parámetros lles debes pasar e, probablemente, poderás descubrir doadamente con que obxectivo foron escritos.

Por exemplo, na libraría de PHP está definido o interface **Countable**.

```
Countable {
    abstract public int count ( void );
}
```

Se creas unha clase para a cesta da compra na tenda web, poderías implementar este interface para contar os produtos que figuran nesta.

Múltiples interfaces

Antes aprendiches que en PHP5 unha clase só pode herdar doutra. En PHP5 non existe a herdanza múltiple. Non obstante, si é posible crear clases que implementar varios interfaces, simplemente separando a lista de interfaces por comas despois da palabra implements.

```
class TV extends Produto implements iMostra, Countable {
    ?
}
```

A única restrición é que os nomes dos métodos que se deban implementar nos distintos interfaces non coincidan. É dicir, no noso exemplo, o interface iMostra non podería conter un método *count*, pois este xa está declarado en Countable.

En PHP5 tamén se poden crear novos interfaces herdando doutros xa existentes. Faise da mesma forma que coas clases, utilizando a palabra **extends**.

Funcións relacionadas cos interfaces

Función	Significado
get_declared_interfaces	Devolve un array cos nomes dos interfaces declarados
interface_exists	Devolve true se existe o interface que se indica, ou false no caso contrario

--Víctor Lourido 13:59 30 jun 2013 (CEST)