Interfaces y constantes

UF5

Repaso! - ¿Qué es la herencia?

Herencia en POO = Cuando una clase *nace y depende* de otra clase.

La clase hija heredará todas las propiedades y métodos públicos y protegidos de la clase principal. Y además, puede tener sus propias propiedades y métodos.

Una clase heredada se define utilizando la palabra clave **extends**.

Anulación de métodos heredados

Los métodos heredados se pueden anular redefiniendo los métodos, usando el mismo nombre, en la clase secundaria (hija).

Interfaces

Las **interfaces** de objetos son contratos que tienen que cumplir las clases que las implementan.

Contienen métodos vacíos que obligan a las clases a implementarlos, promoviendo así un estándar de desarrollo.

Si una clase implementa una interfaz, está obligada a usar todos los métodos de la misma (y los mismos tipos de argumentos de los métodos), de lo contrario dará un error fatal.

Se puede implementar más de una interfaz en cada clase, y pueden extenderse entre ellas mediante extends.

Una interface puede extender una o más interfaces.

Todos los métodos declarados en una **interfaz** deben ser públicos.

Ejemplo Interfaz

Para definir una **interfaz** se utiliza la palabra **interface**, y para extenderla se utiliza **implements**.

```
interface Vehiculo{
    public function getTipo();
    public function getRuedas();
}
class Coche implements Vehiculo{
    public function getTipo(){
        echo "Coche";
    }
    public function getRuedas(){
        echo "4";
    }
}
class Moto implements Vehiculo{
    public function getTipo(){
        echo "Moto";
    }
    public function getRuedas(){
        echo "2";
    }
}
```

Ejemplo Interfaz

Las clases Coche y Moto están obligadas a definir las funciones getTipo() y getRuedas().

Así se crea un **modelo** para todas las clases de vehículo que deben definir unos métodos mínimos.

Si por ejemplo getTipo() tuviera un argumento de tipo Array: getTipo(Array $tipos), las clases que implementen la interfaz deberán importar un argumento del mismo tipo.

```
interface Vehiculo{
    public function getTipo();
    public function getRuedas();
}
class Coche implements Vehiculo{
    public function getTipo(){
        echo "Coche";
    }
    public function getRuedas(){
        echo "4";
    }
}
class Moto implements Vehiculo{
    public function getTipo(){
        echo "Moto";
    }
    public function getRuedas(){
        echo "2";
    }
}
```

Constantes de clase

Las **constantes de clase** pueden resultar útiles si queremos definir algún dato constante dentro de una clase.

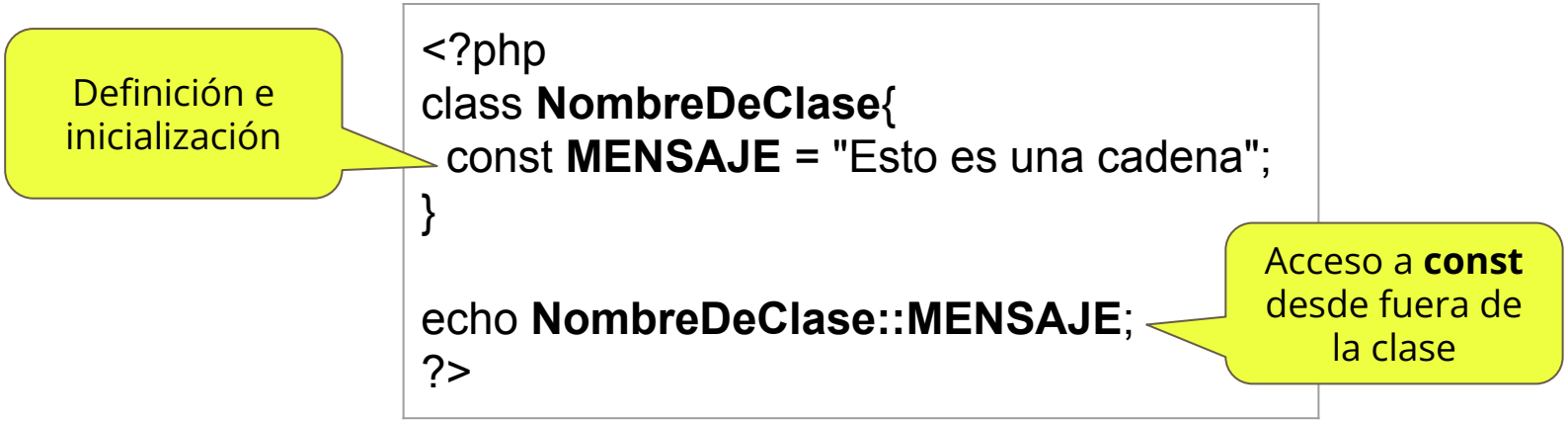
Una constante de clase se declara dentro de una clase con la palabra clave **const**.

Una constante no se puede cambiar una vez declarada.

Se recomienda nombrar las constantes con letras mayúsculas, aunque distinguen entre mayúsculas y minúsculas.

Constantes de clase - Ejemplo de acceso

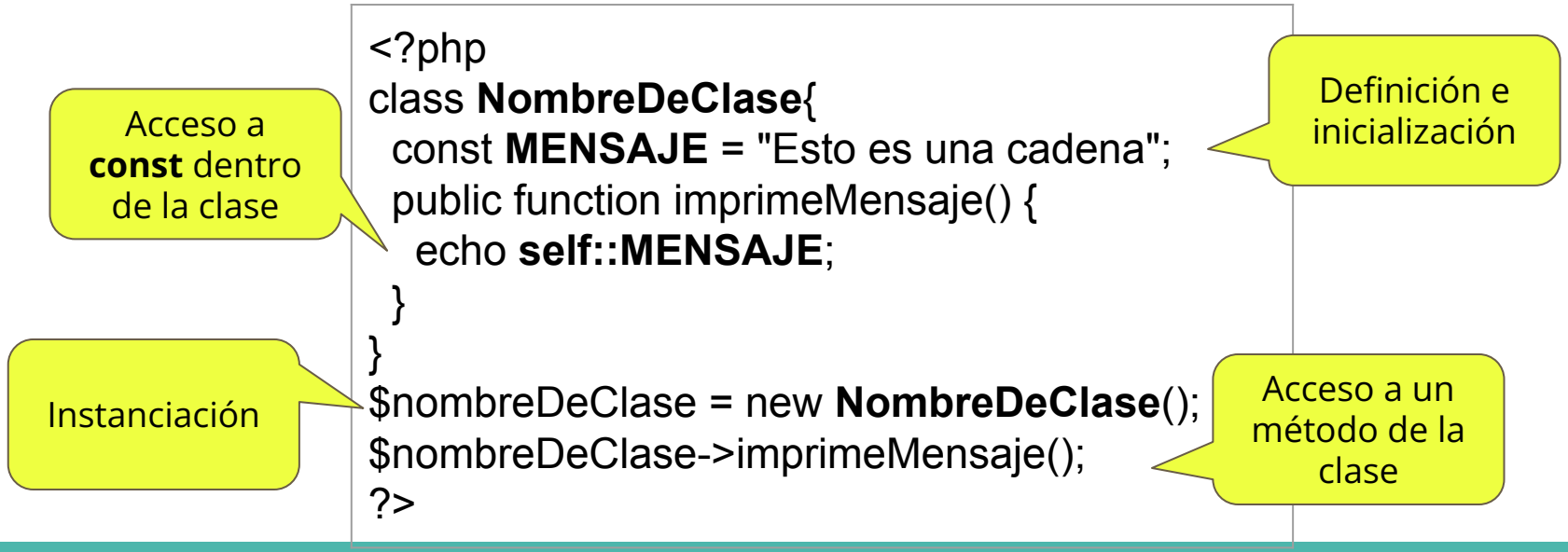
Podemos **acceder a una constante** desde fuera de la clase usando el **nombre de la clase** seguido del **operador** de resolución de alcance (**::**) **y** seguido del **nombre de la constante**:



Operador de resolución de alcance (::)

Constantes de clase - Ejemplo de acceso

O podemos **acceder a una constante** desde dentro de la clase usando la palabra clave **self** seguida del operador de resolución de alcance (**::**) seguido del **nombre de la constante**:



Ejercicio