Creando Clases y Objetos

Roy Alvear – ralvear@indap.cl

Creando Clases y Objetos

- Conceptos Básicos de Programación Orientado a Objetos
- Clases
- Clases Abstractas
- Interfaces
- Espacios de Nombre

Conceptos Básicos de POO

- ¿Qué es la Programación Orientada a Objetos?
- Leyes de Deméter (LoD)
- Definiciones Conceptuales

Conceptos Básicos de POO

- ¿Qué es la Programación Orientada a Objetos?

La programación orientada a objetos (POO) es un paradigma de programación que usa los objetos en sus interacciones. Está basado en varias técnicas, incluyendo herencia, abstracción, polimorfismo y encapsulamiento entre otras.

Conceptos Básicos de POO

- Leyes de Deméter (LoD)

LoD o **Principio de Menos Conocimiento** es una directriz utilizada particularmente en la POO.

- Cada unidad debe tener un limitado conocimiento sobre otras unidades y solo conocer aquellas unidades estrechamente relacionadas a la unidad actual.
- Cada unidad debe hablar solo a sus amigos y no hablar con extraños.
- Solo hablar con sus amigos inmediatos.

Conceptos Básicos de POO

- Definiciones Conceptuales
  - Abstracción
  - Encapsulamiento
  - Herencia
  - Polimorfismo
  - Sobrecarga
  - Cohesión
  - Acoplamiento

Clases

- Una clase es un TDA o Tipo de Dato Abstracto, es decir, una plantilla para la creación de objetos de datos según un modelo predefinido.
- Representan entidades o conceptos, como los sustantivos en el lenguaje. Cada clase es un modelo que define un conjunto de **atributos** y **métodos**.

Clases

- Ejemplo:

Clase Punto

Atributo	Método
CoordenadaX	obtenerCoordenadas
CoordenadaY	FijarCoordenadas

Clases

```php
<?php
class Punto
{
    private $coordenadaX;
    private $coordenadaY;
    public function obtenerCoordenadas()
    {
        return array($this->coordenadaX,$this->coordenadaY);
    }
    public function fijarCoordenadas($x,$y)
    {
        $this->coordenadaX = $x;
        $this->coordenadaY = $y;
    }
}
?>
```

Clases

- Visibilidad
  - **private** : Solamente la clase actual
  - **protected** : La clase actual y sus hijas
  - **public** : Todos
- Siguiendo LoD, los atributos **siempre** deben ser privados

Clases Abstractas

- Las clases definidas como abstractas no se pueden instanciar y cualquier clase que contiene al menos un método abstracto debe ser definida como tal.
- Los métodos definidos como abstractos simplemente declaran la firma del método, pero no pueden definir la implementación.
- Cuando se hereda de una clase abstracta, todos los métodos definidos como abstractos en la declaración de la clase madre deben ser definidos en la clase hija.

Clases Abstractas

```php
<?php
abstract class Vehiculo
{
    protected $velocidadMaxima;
    protected $velocidadPromedio;
    function __construct($velocidadMaxima,$velocidadPromedio)
    {
        $this->velocidadMaxima = $velocidadMaxima;
        $this->velocidadPromedio = $velocidadPromedio;
    }
    final public function obtenerVelocidadMaxima()
    {
        return $this->velocidadMaxima;
    }
    final public function obtenerVelocidadPromedio()
    {
        return $this->velocidadMaxima;
    }
    abstract public function desgaste();
}
?>
```

Interfaces

- Las interfaces de objetos permiten crear código con el cual especificamos qué métodos deben ser implementados por una clase, sin tener que definir cómo estos métodos son manipulados.
- Todos los métodos declarados en una interfaz deben ser públicos, ya que ésta es la naturaleza de una interfaz.

```php
<?php
interface PaginaWeb
{
    public function getHtmlCode();
    public function addHeadCSSLink($url,$media);
}
?>
```

Espacios de Nombre

- Los espacios de nombres proporcionan una manera para agrupar clases, interfaces, funciones y constantes relacionadas.
- Soluciona el problema del conflicto de nombres entre el código que se crea y las de terceros.
- El espacio de nombre debe ser la primer instrucción en el encabezado del archivo PHP.

```
 9
10  namespace Standard;
11
12  use Standard;
13
```

Preguntas