DESARROLLO WEB CON PHP

Centro ADM

Barragán Jimenez Jonathan

30 de junio de 2018

Índice

1. Orientación a objetos
2. Orientación a objetos en PHP
   - Declaración de clases
   - Modificadores de visibilidad
   - Eventos y mensajes entre objetos
   - Métodos constructores
   - Propiedades Estáticas
   - Métodos Estáticos
   - Valores constantes
   - Operador de resuloción de ámbito(::)
   - Superclases y Subclases

Índice

1. Orientación a objetos
2. Orientación a objetos en PHP
   - Declaración de clases
   - Modificadores de visibilidad
   - Eventos y mensajes entre objetos
   - Métodos constructores
   - Propiedades Estáticas
   - Métodos Estáticos
   - Valores constantes
   - Operador de resuloción de ámbito(::)
   - Superclases y Subclases

Orientación a objetos

Técnica de programación que define clase y usa objetos y sus interacciones para diseñar aplicaciones.

Índice

1 Orientación a objetos

2 Orientación a objetos en PHP
- Declaración de clases
- Modificadores de visibilidad
- Eventos y mensajes entre objetos
- Métodos constructores
- Propiedades Estáticas
- Métodos Estáticos
- Valores constantes
- Operador de resuloción de ámbito(::)
- Superclases y Subclases

Declaración de clases
Modificadores de visibilidad
Eventos y mensajes entre objetos
Métodos constructores
Propiedades Estáticas
Métodos Estáticos
Valores constantes
Operador de resuloción de ámbito(::)
Superclases y Subclases

Orientación a objetos en PHP

Empezó en php4. Tienen muchas maneras extravagantes de inicar un objeto. Veremos las más apegadas a los otros lenguajes con paradigma orientados a objetos.

Índice

1. Orientación a objetos
2. Orientación a objetos en PHP
   - Declaración de clases
   - Modificadores de visibilidad
   - Eventos y mensajes entre objetos
   - Métodos constructores
   - Propiedades Estáticas
   - Métodos Estáticos
   - Valores constantes
   - Operador de resuloción de ámbito(::)
   - Superclases y Subclases

Declaración de clases
Modificadores de visibilidad
Eventos y mensajes entre objetos
Métodos constructores
Propiedades Estáticas
Métodos Estáticos
Valores constantes
Operador de resuloción de ámbito(::)
Superclases y Subclases

Declaración de clases

En PHP se usa la palabra reservada class seguido del nombre de la clase. El nombre puede estar formado con letras y números, pero no puede empezar con un número.

Ejemplo

```php
<?php
class Clase {
  //cuerpo de la clase
}
```

Atributos o propiedades

Sintáxis:

$ < nombreAtributo > [= < valor por omisión >];

Ejemplo

```php
<?php
class Clase {
  public $atributo1 = "";
  public $atrbuto2;
  .
  .
  .

}
```

Métodos

Se le llama **firma o encabezado** de un método al nombre del método, al número y tipo de los parametros.

Ejemplo

```php
<?php
class Clase {
  public $atributo1 = "";
  public $atrbuto2;

  public function metodo($par1,..,$parN){
    //Codigo del metodo.
  }

}
```

En PHP sólo se puede tener un método con un mismo nombre, no importa si la firma es diferente.

Índice

1. Orientación a objetos
2. Orientación a objetos en PHP
   - Declaración de clases
   - Modificadores de visibilidad
   - Eventos y mensajes entre objetos
   - Métodos constructores
   - Propiedades Estáticas
   - Métodos Estáticos
   - Valores constantes
   - Operador de resuloción de ámbito(::)
   - Superclases y Subclases

Declaración de clases
Modificadores de visibilidad
Eventos y mensajes entre objetos
Métodos constructores
Propiedades Estáticas
Métodos Estáticos
Valores constantes
Operador de resuloción de ámbito(::)
Superclases y Subclases

Modificadores de visibilidad

Son usados para determinar la visibilidad tanto de atributos como métodos. es decir qué objetos, según la clase a la que pertenecen, pueden hacer uso de ellos (atributos y métodos). Son tres

- **public**
- **protected**
- **private**

Modificadores de visibilidad

- **public:**
  acceso desde cualquier contexto (clase). Cualquier objeto puede hacer uso de este recurso.
- **protected**
  acceso desde la clase propietaria y las subclases.
- **private**
  acceso sólo desde la clase propietaria. Sólo los objetos de la misma clase pueden hacer uso del recurso.

Índice

1. Orientación a objetos
2. Orientación a objetos en PHP
   - Declaración de clases
   - Modificadores de visibilidad
   - Eventos y mensajes entre objetos
   - Métodos constructores
   - Propiedades Estáticas
   - Métodos Estáticos
   - Valores constantes
   - Operador de resuloción de ámbito(::)
   - Superclases y Subclases

Declaración de clases
Modificadores de visibilidad
Eventos y mensajes entre objetos
Métodos constructores
Propiedades Estáticas
Métodos Estáticos
Valores constantes
Operador de resuloción de ámbito(::)
Superclases y Subclases

Eventos y mensajes entre objetos

Un **evento** se genera cuando un objeto usa los métodos de otro objeto.
Mensaje es la respuesta del objeto cuyo método fue invocado.

El acceso y modificación a los datos de un objeto debe hacerse a través de sus métodos.

Índice

1. Orientación a objetos
2. Orientación a objetos en PHP
   - Declaración de clases
   - Modificadores de visibilidad
   - Eventos y mensajes entre objetos
   - Métodos constructores
   - Propiedades Estáticas
   - Métodos Estáticos
   - Valores constantes
   - Operador de resuloción de ámbito(::)
   - Superclases y Subclases

Métodos constructores

- Su nombre debe ser **__construct()**
- Es llamado automáticamente cada vez que un objeto de la clase es creado.
- **NO puede tener** un enunciado return

Declaración de clases
Modificadores de visibilidad
Eventos y mensajes entre objetos
Métodos constructores
Propiedades Estáticas
Métodos Estáticos
Valores constantes
Operador de resuloción de ámbito(::)
Superclases y Subclases

Métodos constructores

- PHP proporciona un método constructor por omisión cuando el programador no lo hace.
- No acepta como métodos constructores métodos que tengan el mismo nombre de la clase
- Los métodos constructores no se invocan de manera explícita

Ejemplo Instanciación

```
//Creacion de un objeto
$clase = new Clase();
```

Pseudovariable $this

$this es un mecanismo por el cual una clase se refiere a una instancia y tiene acceso a sus miembros
Para poder acceder a los miembros del objeto se necesita el operador -> seguido del nombre del atributo o método que se requiera.

Índice

1. Orientación a objetos
2. Orientación a objetos en PHP
   - Declaración de clases
   - Modificadores de visibilidad
   - Eventos y mensajes entre objetos
   - Métodos constructores
   - Propiedades Estáticas
   - Métodos Estáticos
   - Valores constantes
   - Operador de resuloción de ámbito(::)
   - Superclases y Subclases

Declaración de clases
Modificadores de visibilidad
Eventos y mensajes entre objetos
Métodos constructores
Propiedades Estáticas
Métodos Estáticos
Valores constantes
Operador de resuloción de ámbito(::)
Superclases y Subclases

Propiedades Estáticas

Propiedades propias de la clase, es decir, no necesitan la instanciación de ningún objeto para existir.

Propiedades Estáticas

Sintaxis:
<**ModificadorDeVisivilidad** > **static $** < **nombre**> **=** <**valor** >**;**
Donde :

- **ModificadorDeVisivilidad** es public, protected o private.
- **nombre** es una cadena de caracteres, dígitos y _ pero no puede empezar con dígito
- **static** es una palabra reservada del lenguaje.
- **valor** debe ser un valor constante, no puede se expresiones o resultados de funciones.

Ejemplo

```
Class Clase{
  public static $valor = 0;
}
echo Clase::$valor;
```

Índice

1. Orientación a objetos
2. Orientación a objetos en PHP
   - Declaración de clases
   - Modificadores de visibilidad
   - Eventos y mensajes entre objetos
   - Métodos constructores
   - Propiedades Estáticas
   - Métodos Estáticos
   - Valores constantes
   - Operador de resuloción de ámbito(::)
   - Superclases y Subclases

Declaración de clases
Modificadores de visibilidad
Eventos y mensajes entre objetos
Métodos constructores
Propiedades Estáticas
Métodos Estáticos
Valores constantes
Operador de resuloción de ámbito(::)
Superclases y Subclases

Métodos Estáticos

Métodos propios de la clase, es decir, no necesitan la instanciación de ningún objeto para poder ser usados.

Métodos Estáticos

Sintaxis:
<ModificadorDeVisivilidad > static function $ < nombre> ($arg1,...,$argN);
Donde :

- **ModificadorDeVisivilidad** es public, protected o private.
- **static** es una palabra reservada del lenguaje.
- **function** es una palabra reservada del lenguaje.
- **nombre** es una cadena de caracteres, dígitos y _ pero no puede empezar con dígito

Ejemplo

```
Class Clase{
  public static function getAtributo(){
    return true;
  }
}
echo Clase::getAtributo();
```

Declaración de clases
Modificadores de visibilidad
Eventos y mensajes entre objetos
Métodos constructores
Propiedades Estáticas
Métodos Estáticos
Valores constantes
Operador de resuloción de ámbito(::)
Superclases y Subclases

Métodos Estáticos

La pseudovariable $this no está disponible dentro de un método estático ya que la declaración del método en esta forma supone que se mandará a llamar fuera desde la definición de la clase y no desde un objeto de la misma.

Índice

1. Orientación a objetos
2. Orientación a objetos en PHP
   - Declaración de clases
   - Modificadores de visibilidad
   - Eventos y mensajes entre objetos
   - Métodos constructores
   - Propiedades Estáticas
   - Métodos Estáticos
   - Valores constantes
   - Operador de resuloción de ámbito(::)
   - Superclases y Subclases

Valores constantes

Son valores constantes propios de una clase que permanecen invariables.

Valores constantes

Sintaxis:

const <nombre > = < valor>;

Donde :

- **const** es una palabra reservada del lenguaje.
- **nombre** es una cadena de caracteres, dígitos y _ pero no puede empezar con dígito
- **valor** debe ser un valor constante, no puede ser expresiones o resultados de funciones

Índice

1. Orientación a objetos
2. Orientación a objetos en PHP
   - Declaración de clases
   - Modificadores de visibilidad
   - Eventos y mensajes entre objetos
   - Métodos constructores
   - Propiedades Estáticas
   - Métodos Estáticos
   - Valores constantes
   - Operador de resuloción de ámbito(::)
   - Superclases y Subclases

Declaración de clases
Modificadores de visibilidad
Eventos y mensajes entre objetos
Métodos constructores
Propiedades Estáticas
Métodos Estáticos
Valores constantes
Operador de resuloción de ámbito(::)
Superclases y Subclases

Operador de resuloción de ámbito(::)

- Nombrado como Paamayinm Nekudotayim. Dos puntos en hebreo.
- Se utiliza para acceder a elementos estáticos o constantes de una clase desde fuera o dentro de ella.
  - Uso en contexto externo:
    < NombreDeLaClase>:: <elemento >
  - Uso en contexto interno
    self:: <elemento >

Índice

1. Orientación a objetos
2. Orientación a objetos en PHP
   - Declaración de clases
   - Modificadores de visibilidad
   - Eventos y mensajes entre objetos
   - Métodos constructores
   - Propiedades Estáticas
   - Métodos Estáticos
   - Valores constantes
   - Operador de resuloción de ámbito(::)
   - Superclases y Subclases

Superclases y Subclases

La clase de la que se hereda se suele llamar clase base, clase padre o superclase , mientras que la clase que la extiende se conoce como clase derivada, clase hija o subclase.

Tipo de herencia

- Herencia simple
- Herencia multiple

Herencia simple

Una clase hereda de una única clase

- La subclase copia el comportamiento del padre, siempre y cuando no sobreescriba métodos.
- La subclase puede agregar comportamiento específico.
- La subclase puede ocupar el constructor del padre dentro de su constructor si así lo requiere. parent::__construct($a, ..., $n);

Metodos Final

Son métodos declarados en la clase padre con la palabra reservada final para que no puedan ser sobreescritos por las subclases.

Metodos abstractos

Son métodos declarados que sólo presentan la firma pero no la implementación. Usan la palabra reservada abstract para en su declaración.

Clase abstractas

- Son clases que no se pueden instanciar.
- Tienen definido al menos un método abstracto.
- 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.

Interfaces

- Las interfaces nos permiten especificar el comportamiento de una clase sin tener que implementar el comportamiento.
- En PHP las interfaces se definen con la palabra reservada interface seguido del nombre de la interfaz.
- Los métodos de la interfaz sólo definen su firma y terminan con un punto y coma.
- Todos deben tener modificador de visibilidad public.

Interfaces

Se utiliza la palabra reservada implements para indicar que una clase está implementando los métodos listados en una interfaz.