



UNIVERSIDAD
Popular del Cesar

Ingeniería de Sistemas

Patrones de Diseño de Software



Profesor: Deivis Martínez Acosta

Clases y Objetos

En el mundo del software, un objeto es un componente de software con estructura similar a los objetos del mundo real. Cada objeto está compuesto por un juego de datos (propiedades y/o atributos) que son las variables que describen las características esenciales del objeto, también lo compone un juego de métodos (comportamientos) que describen cómo el objeto se comporta.

Clases

La clase es la estructura fundamental en la programación orientada a objetos. Esta se puede tomar como una plantilla, un prototipo o un anteproyecto de los objetos. Consiste en dos tipos de miembros que son llamados campos (propiedades o atributos) y métodos. Los campos especifican el tipo de datos definidos por la clase, mientras que los métodos especifican las operaciones.

De otra forma podemos decir que es una construcción que le permite crear sus propios tipos personalizados agrupando las variables de otros tipos, métodos y eventos.

Una clase es como un plano. Define los datos y el comportamiento de un tipo.


Abstracción que define un tipo de objeto especificando qué propiedades (atributos) y operaciones disponibles va a tener.



Objetos

Un objeto es una instancia de una clase.

Las clases proporcionan la propiedad de la reusabilidad, los programas de software pueden usar una clase una y otra vez para crear muchos objetos.



Instancia de una clase

Para crear un objeto o una instancia de una clase, utilizamos el operador new, ejemplo, se escribe el siguiente código si se desea crear una instancia de la clase String.

```
String saludo = new String("Hola Especialización");
```

Declaración de una clase en lenguaje de programación.


La declaración de una clase no es diferente a escribir en sintaxis del lenguaje una clase de la programación orientada a objetos. Recordemos que una clase es el modelo de determinado tipos de objetos, es la definición del objeto en si.



Clases y Objetos

En detalle la sintaxis se refiere a:
class es la palabra reservada del lenguaje que se usa para especificar que se está declarando una clase.

Variables, campos o estados son exactamente eso, variables de clase, estados de los objetos de esta clase.






Clases y Objetos

Funciones, métodos o mensajes, son las declaraciones a las funciones o mensajes para lograr la comunicación con los objetos de este tipo o clase.

Constructor, es un método que se ejecuta cuando se “instancia” una clase o se crea un nuevo objeto de esta clase.



Modificador de acceso de una clase

Es la forma de determinar como otras clases pueden acceder a la clase, los modificadores de acceso más usados de una clase son private: Se permite acceder a la clase solo desde clases que se encuentren ubicadas dentro del mismo paquete. public: Con este se permite que se acceda a la clase desde cualquier otra clase, no importando su ubicación.

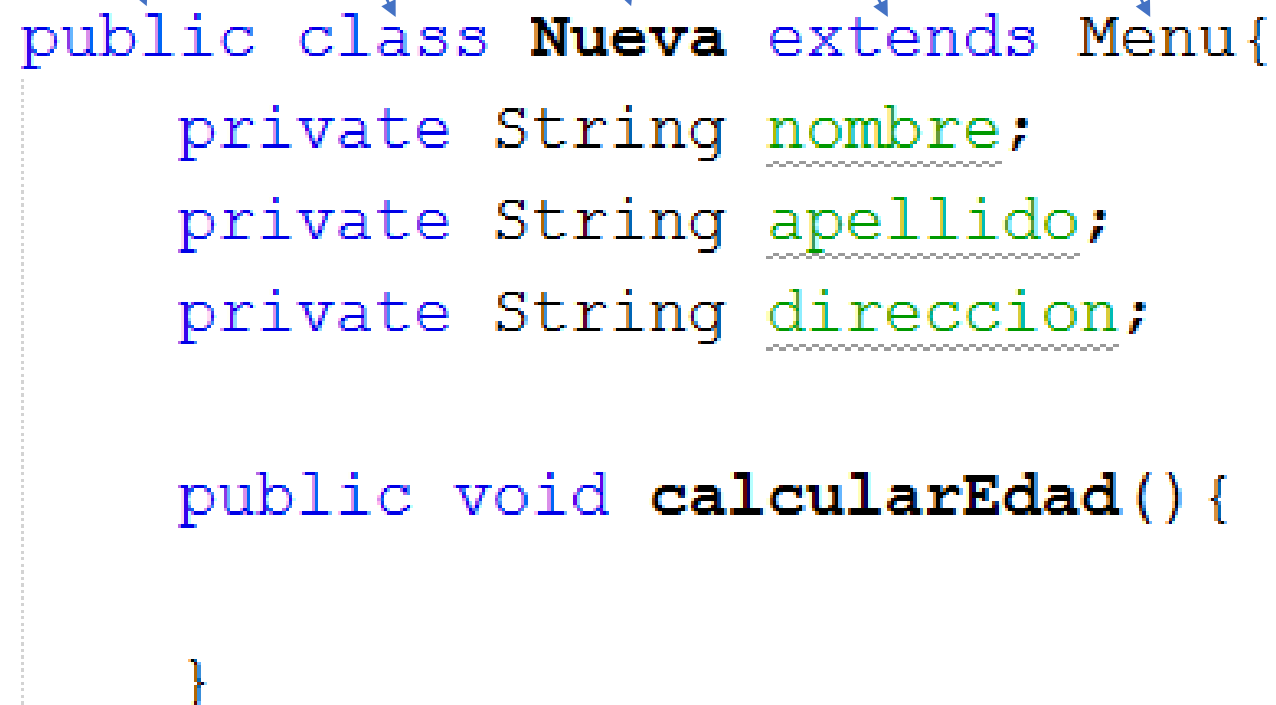
Hacer Herencia

Declaración de una clase en que hereda de otra clase.

```
ModificadorAcceso class NombreClaseHija extends ClasePadre {  
// campos o estados;  
// Constructor; y  
// Funciones, métodos o mensajes  
}
```

La clase hija contiene todos los métodos y variables de la ClasePadre

ModificadorAcceso class NombreClaseHija extends ClasePadre {



```
public class Nueva extends Menu{  
    private String nombre;  
    private String apellido;  
    private String direccion;  
  
    public void calcularEdad() {  
  
    }  
}
```



Métodos de Instancia

Para realizar la llamada de un método de un objeto instancia de una clase seguimos la siguiente notación:

`miObjetoInstacia.miNombreMetodo(parámetros)`



Métodos estáticos

Cuando realizamos la llamada a un método estático se está realizando una llamada al método en la clase y no es necesario tener una instancia de la clase. Se diferencian de los demás métodos en la clase al momento de escribirlos por que tienen la palabra reservada “static”.

```
static MiClase.metodoEstatico( parámetros )
```


Convenciones en el nombre de un método

Existen algunas reglas básicas para el nombre de un método, no significa que los lenguajes restrinjan si se utiliza un nombre o el otro, solo que por orden y estándar se siguen los siguientes enunciados:

El nombre debe ser un verbo, ejemplo: correr, obtener, traer, llevar, partir, etc.

Clases y Objetos

Se debe escribir en minúscula y si es compuesto de varias palabras se debe iniciar la siguiente palabra con mayúscula.

Si el nombre es compuesto por varias palabras, la primera palabra es un verbo y la segunda un adjetivo.


Las **variables internas** son llamadas en minúscula y las compuestas como los métodos.



constructor

Un constructor es un método especial de una clase, éste método es llamado automáticamente al crear un objeto de esa clase. La función del constructor es iniciar el objeto.

Un constructor se identifica porque tiene el mismo nombre que la clase a la que pertenece y no retorna ningún valor.



Clases y Objetos

Toda clase de objetos contiene al menos un constructor, aun cuando no se haya definido ninguno se crea uno por defecto al momento de crear un objeto.

Un constructor por defecto se puede declarar o puede ser omitido al crear una clase.

Se puede crear uno, dos o más constructores según se necesite.

Clases y Objetos

Existen 3 diferentes tipos de constructores más usados.

- 1.- Constructor por defecto.
- 2.- Constructor de copia.
- 3.- Constructor común (Con argumentos).

Ejemplo de una clase con constructor por defecto sin definir:

```
public class Persona {  
    private String nombres;  
    private String apellidos;  
    private String celular;  
}
```

Clases y Objetos

Persona persona=new Persona();

La clase anterior omitió el constructor y al crear el objeto “persona” vemos que se llama a el constructor de la clase, aun cuando no se creó ningún constructor se crea uno por defecto.

El operador “new” nos indica que va a crear un objeto con ese constructor.

Clases y Objetos

También se puede definir el constructor por defecto de la siguiente manera:

```
public class Persona {  
    private String nombre;  
    private String apellidoPaterno;  
    private String apellidoMaterno;  
    public Persona(){  
    }  
}  
  
Persona persona=new Persona();
```

Constructor de copia

El constructor de copia se utiliza para inicializar a un objeto con otro objeto de la misma clase.

Ejemplo

```
public class Persona {  
    private String nombres;  
    private String apellidos;  
    private String celular;
```

Clases y Objetos

```
//Constructor por defecto  
public Persona(){  
}  
  
//Constructor copia  
public Persona(Persona persona){  
    this.nombres=persona.nombres;  
    this.apellidos=persona.apellidos;  
    this.celular=persona.celular;  
}  
}
```

La forma de implementación sería la siguiente.

```
Persona persona=new Persona();  
Persona personaCopia=new Persona(persona);
```

Constructor común.

El constructor común es aquel que recibe parámetros para asignarles los valores iniciales a un nuevo objeto, se crea de la siguiente manera.

```
public class Persona {  
    private String nombres;  
    private String apellidos;  
    private String celular;
```

Clases y Objetos

```
//Constructor común  
public Persona(String nombres, String  
apellidos, String celular){  
this.nombres = nombres;  
this.apellidos = apellidos;  
this.celular = celular;  
}  
}  
Persona personaCopia=new Persona("Andra  
Carolina", "Serna Redondo","3214567890");
```

Destructores

Un destructor es un método opuesto a un constructor, éste método en lugar de crear un objeto lo destruye liberando la memoria de nuestra computadora para que pueda ser utilizada por alguna otra variable u objeto.



UNIVERSIDAD
Popular del Cesar

Vamos a la práctica