OOP 03
Nueva edición - 2020

PROGRAMANDO

holamundo.co

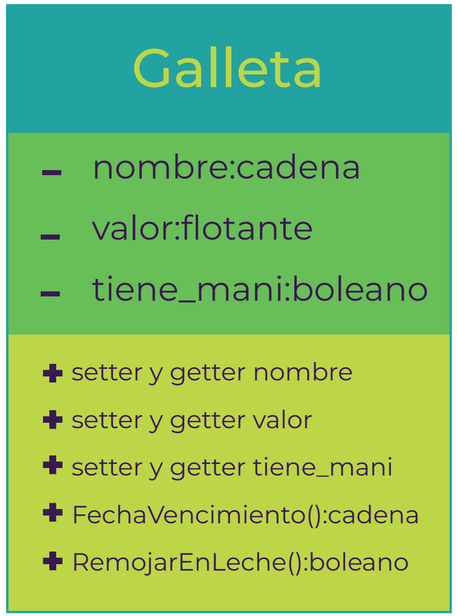
@xacarana

Agenda

Valores por defecto
¿Qué es un constructor?
Constructor por defecto
Firma de un constructor
Palabra reservada this
Relaciones entre clases
Ejercicios

Valores por defecto

Cuando se crea un objeto a partir de una clase. A los miembros dato, se les asigna un valor por defecto, que depende del tipo de dato.



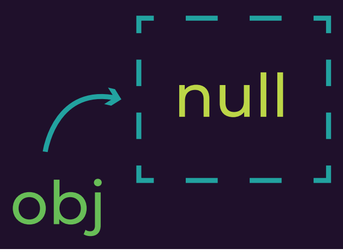
Si se crea una instancia de la clase Galleta, al usar los getters se obtienen un valor, es decir las variables se inicializan cuando se crea el objeto.

!

No se recomienda asignar un valor directamente al miembro dato en la clase.

Valores por defecto

```
Galleta obj;
```



Al declarar la variable obj, solo se reserva un espacio null en memoria, NO hay información, ya que no existe el objeto.

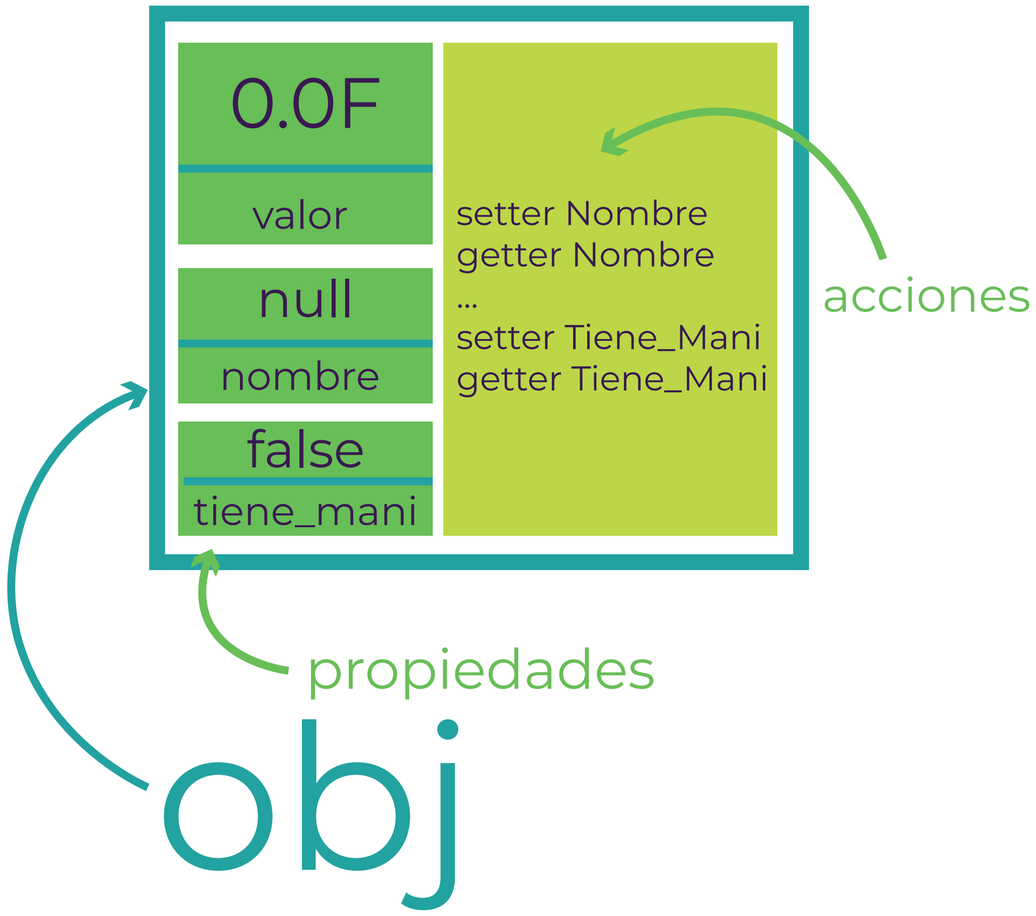
Al usar el operador new se crea el objeto y se asignan los valores iniciales a todos los miembros dato.

```
obj = new Galleta();
```

```
Console.Write(obj.Tiene_mani);

Cmd > false
```

Si se imprime usando el getter de la clase de algún miembro dato, obtenemos un valor.

Valores por defecto



La imagen muestra la representación de una instancia acabada de crear

Cuando se crea un objeto a partir de una clase, sus miembros datos, se inicializan con un valor por defecto, dependiendo del tipo de dato.

!

Para ver una lista de los valores por defecto de los datos primitivos visitar el siguiente enlace

@xacarana

Valores por defecto

Cuando se crea un objeto a partir de la definición de su clase, se crean copias genéricas.

Para personalizar la copia, es decir el objeto, se modifican sus características, cambiando sus propiedades usando los setter y getters.

```
Galleta galleta_01 = new Galleta();
```

Se crea una instancia genérica de la clase.

```
Galleta galleta_02 = new Galleta();
galleta_02.Nombre = "James";
galleta_02.Apellido = "Bond";
```

Se personaliza, el objeto, cambiando sus propiedades.

! **Recuerde** encapsular los miembros dato

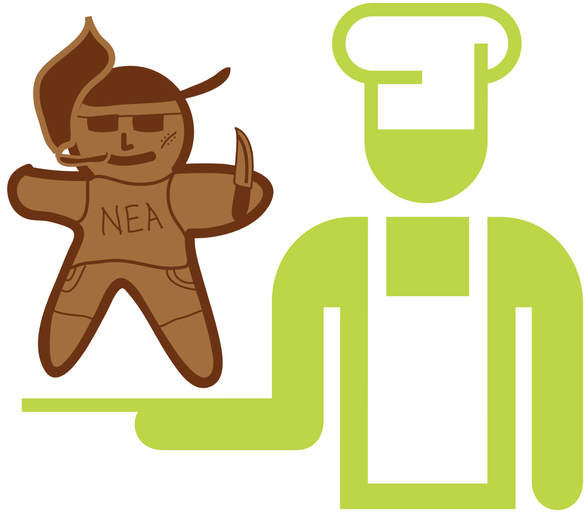
Valores por defecto

Todos los objetos, tienen valores por defecto, si los consultamos utilizando los getters de la clase, obtenemos resultados parecidos a los que muestra la imagen.

Para cambiar los valores usamos los setters. De esta forma se personaliza el objeto, como se mostro anteriormente con la galleta James Bond.

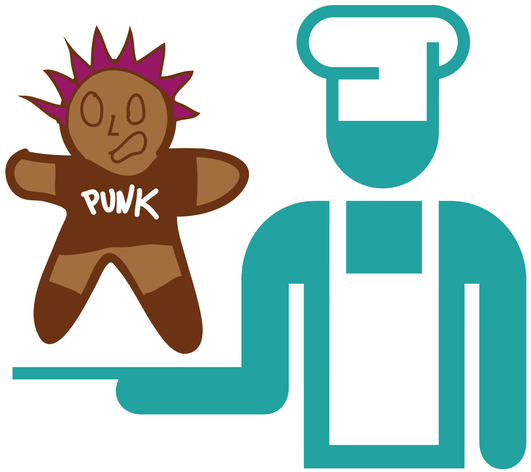
El valor por defecto, depende del tipo de dato.

¿Qué es un constructor?

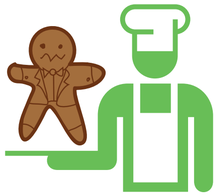


Frecuentemente se requiere inicializar un objeto con valores personalizados inmediatamente cuando se crea el objeto, en este caso se usan los contructores.

Los constructores también se pueden usar para asignar valores por defecto, diferentes a los tradicionales al objecto.



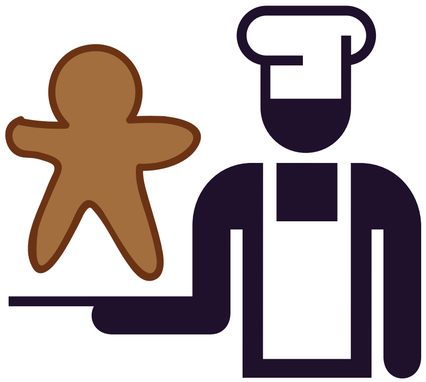
El constructor, sería el panadero, el cual personaliza los productos.



Siempre hay un constructor que crea objetos con valores por defecto.

Si se usa un constructor diferente al constructor por defecto, se debe incluir el constructor vacío, si se quiere seguir usando esta característica para crear instancias, sin pasar argumentos.

Constructor por defecto



```
Galleta g = new Galleta();
```

Constructor por defecto

```
Galleta(){
  //Sentencias
}
```

Firma de un constructor

```
[Modificador] NombreClase(listaparametros){
  //sentencias de inicialización de objeto
}
```

Un constructor es similar a una función, la diferencia es que los constructores no tienen retorno, ya que su objetivo es inicializar o ejecutar funciones al momento del instanciamiento de la clase.

Constructores personalizados

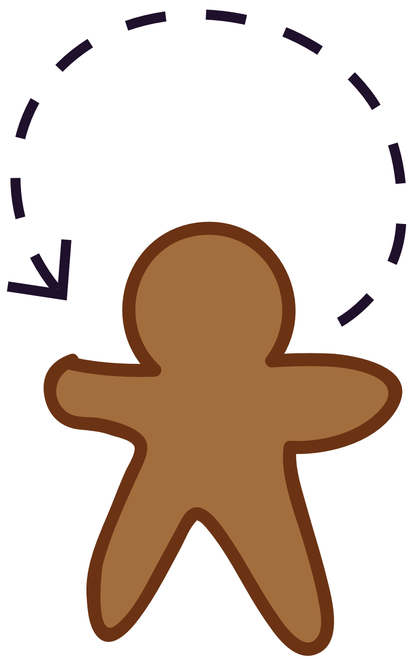
Una clase puede ten-
er, varios constructores,
los cuales deben tener
el mismo nombre pero
con diferente lista de
parámetros o tipo de
dato en el caso en el que
los constructores
reciban el mismo núme-
ro de valores, como
muestra la imagen.

```
Galleta(){
  Precio = 1000;
}

Galleta(float precio){
   Precio = precio;
}

Galleta(string tipo, float peso){
   Tipo = tipo;
   Peso = peso;
   if(tipo==”fortuna”)
   {
     CrearFrase();
   }
}
```

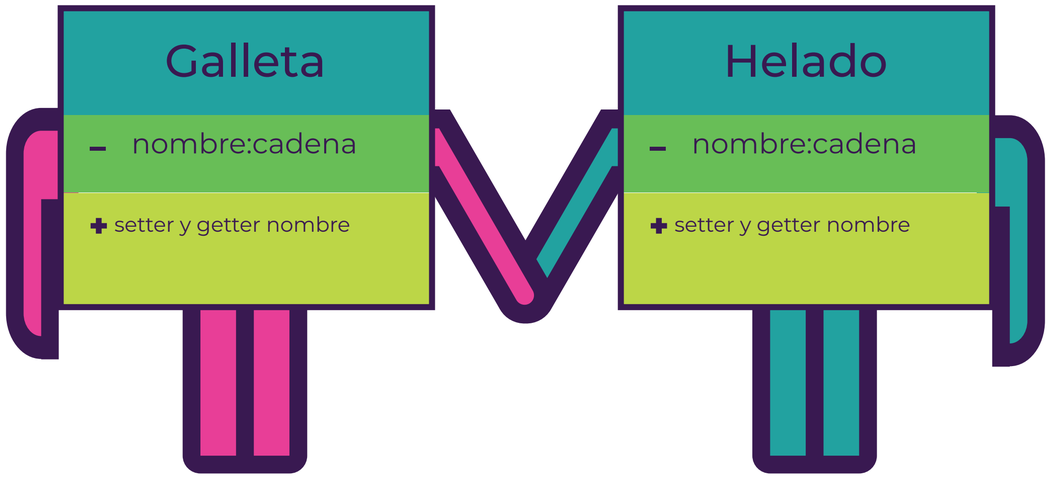
Palabra reservada this

Es una referencia al mismo objeto, se usa para autoasignación de propiedades en los constructores o en cualquier contexto dentro de la clase. Evita además conflictos con nombres de valores que llegan a la clase.

```
this.Nombre = “John Doe”;
```



Relaciones entre clases



Luego de revisar el esqueleto de la clase y sus principales características, es importante empezar a crear relaciones, ya que un programa esta compuesto por un grupo de conceptos representados en clases interconectadas.

Multiplicidad

Indica cuántas instancias de una clase pueden asociarse con las otras clases de la relación, se escribe usando la notación M..N, donde M es el límite inferior y N, el superior.

- Si el límite inferior es 0, la asociación es opcional. Si el límite inferior es 1 o superior implica que la asociación es obligatoria y como mínimo han de ocurrir ese número de instancias.
- El límite superior puede estar acotado por un número exacto o ser indefinido. En este último caso se usa el símbolo * (asterisco).
- Si ambos límites coinciden, se colapsan en un único número. Por ejemplo 1..1 pasa a ser 1.

Asociación

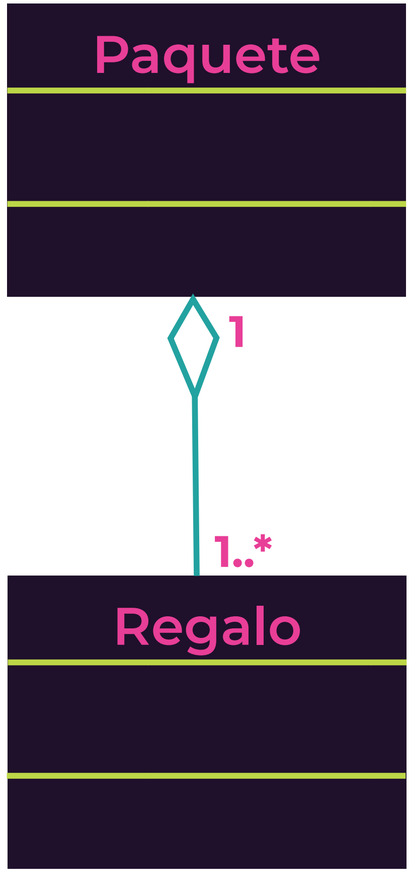
Relación binaria, se representa con una línea continua. Puede relacionar cualquier número de clases. Se le puede asignar un nombre, y en sus extremos se puede hacer indicaciones, como el rol que desempeña la asociación, los nombres de las clases relacionadas, su multiplicidad, su visibilidad, entre otras.

Composición

Implica que los componentes de un objeto sólo pueden pertenecer a un solo objeto agregado, de forma que cuando el objeto agregado es destruido, todas sus partes son destruidas también. La relación se representa con un rombo relleno.

- Para representar el todo y sus partes
- Si se elimina el contenedor, el contenido también es eliminado

Agregación



La agregación es una asociación con unas connotaciones semánticas más definidas, es más especifica: la agregación es la relación parte-de, que presenta a una entidad como un agregado de partes (en orientación a objeto, un objeto como agregado de otros objetos). Debe ser una asociación binaria.

- Si se elimina una clase no es necesario eliminar otra que tenga relación
- Si se elimina el contenedor, el contenido usualmente no se destruye
- Clasifica o cataloga contenido para distinguilo del todo (contenedor)

Ejercicios

- ¿Cuál es el valor por defecto de los principales tipos de variable?
- ¿Por qué no se recomienda asignar un valor directamente a un miembro dato?
- ¿Cuántos constructores se pueden crear en una clase?
- Realice 3 constructores de la clase Postre, cree 5 instancias, usando las diferentes versiones de los constructores, adicional agregue el constructor por defecto.
- ¿Por qué los constructores no tienen un valor de retorno?
- ¿Cual es el límite totla en la cantidad de variables de la lista de parámentros de un constructor? ¿Se puede recibir cualquier tipo de dato?
- Cree 5 clases adicionales a la clase galleta, y cree una relación asociativa con las nuevas clases, tenga en cuenta el contexto de la panaderia.
- Complete las propiedades de la clase Galleta, hasta alcanzar un mínimo de 10 items
- ¿Como se puede asignar un arreglo a un objeto en tiempo de ejecución?
- ¿Cuál es el valor por defecto de un array de tipo Galleta?
- Cree tres ejemplos donde se evidencia la relación de composición
- Cree tres ejemplos donde se evidencia la relación de agregación
- Realice un diagrama de clases donde se evidencie las relaciones entre la clase, arbol, hoja, flor y tronco.