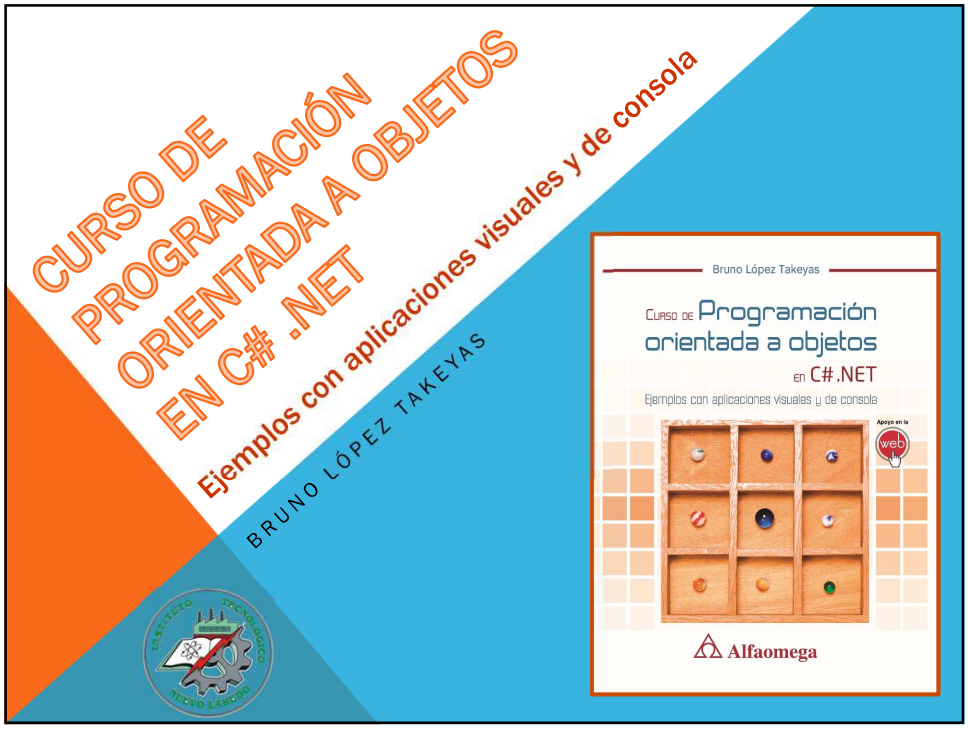

CURSO DE PROGRAMACIÓN ORIENTADA A OBJETOS EN C# .NET
Ejemplos con aplicaciones visuales y de consola
BRUNO LÓPEZ TAKEYAS
Bruno López Takeyas
Curso de Programación orientada a objetos en C#.NET
Ejemplos con aplicaciones visuales y de consola
Alfaomega


CAPÍTULO 5
RESTRICCIÓN DE TIPOS
PROHIBIR
2

Restricciones de tipos de parámetros

- Se usan al diseñar clases genéricas
- Se utilizan para "obligar" a enviar ciertos tipos como parámetros a las clases genéricas
- Se implementan usando la sentencia `where`

3

Restricciones de tipos de parámetros (*cont.*)

Restricción	Descripción
where T: struct	El argumento de tipo debe ser un tipo de valor. Se puede especificar cualquier tipo de valor excepto Nullable. Para obtener más información, consulte Utilizar tipos que aceptan valores NULL.
where T : class	El argumento de tipo debe ser un tipo de referencia; esto se aplica también a cualquier tipo de clase, interfaz, delegado o matriz.
where T : new()	El argumento de tipo debe tener un constructor público sin parámetros. Cuando se utiliza la restricción `new()` con otras restricciones, debe especificarse en último lugar.
where T : <nombre de clase base>	El argumento de tipo debe ser la clase base especificada, o bien debe derivarse de la misma.
where T: <nombre de interfaz>	El argumento de tipo debe ser o implementar la interfaz especificada. Se pueden especificar varias restricciones de interfaz. La interfaz con restricciones también puede ser genérica.
where T : U	El argumento de tipo proporcionado para T debe ser o derivar del argumento proporcionado para U.

4

La restricción *where*

- Obliga a que una clase genérica utilice alguna restricción particular. P. ejem:
  - Una estructura
  - Una clase
  - Un constructor default
  - La implementación de una interfase
  - Una clase base o derivada
- Si no la contiene… ERROR !!!

5

Ejemplo de uso de *where*

- Un restaurante de comida rápida ofrece los siguientes productos:
  - Hamburguesas
  - Pizzas
  - Tacos
  - Tortas
  - Papas fritas
- Una motocicleta reparte cada alimento en su respectiva caja

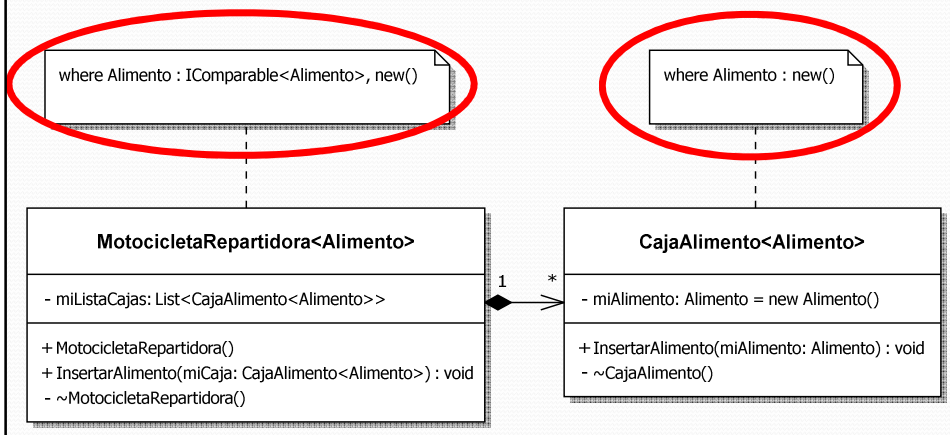
6

Ejemplo de uso de *where (cont.)*

- Cada alimento tiene los siguientes datos:
  - Nombre del alimento
  - Precio
  - Domicilio de entrega
  - Distancia en metros desde el restaurante
- La motocicleta transporta varias cajas con alimentos a la vez y las reparte de acuerdo a la distancia en metros (iniciando por el más cercano).

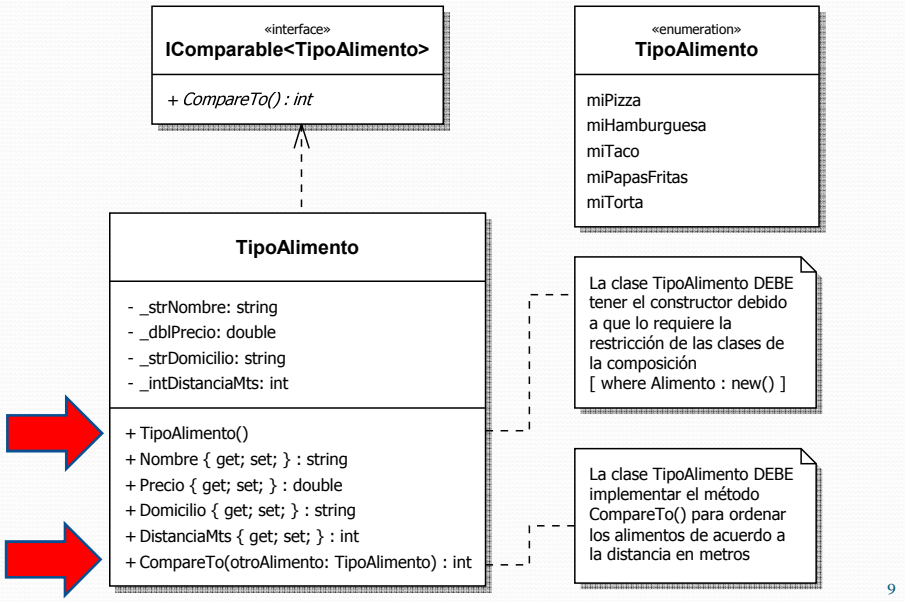
7

Diseño de la composición usando la sentencia *where*



8

Diseño de la clase TipoAlimento



La clase CajaAlimento

```
public class CajaAlimento<Alimento> where Alimento:new()
{
   private Alimento miAlimento;

   public CajaAlimento() {
      miAlimento = new Alimento();
   }

   public void InsertarAlimento(Alimento miAlimento) {
      this.miAlimento = miAlimento;
   }

   ~CajaAlimento() {
      // Elimina el objeto miAlimento
      miAlimento = default(Alimento);
   }
}
```

10

La clase MotocicletaRepartidora

```
class MotocicletaRepartidora<Alimento> where Alimento :
IComparable<Alimento>, new()
{
   private List<CajaAlimento<Alimento>> miListaCajas;

   public MotocicletaRepartidora() {
      miListaCajas = new List<CajaAlimento<Alimento>>();
   }

   public void InsertarAlimento(CajaAlimento<Alimento> miCaja) {
      miListaCajas.Add(miCaja);
   }

   ~MotocicletaRepartidora() {
      miListaCajas.Clear();
   }
}
```

11

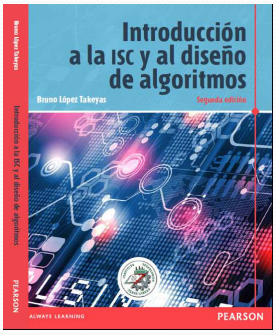
Implementación de la sentencia *where*

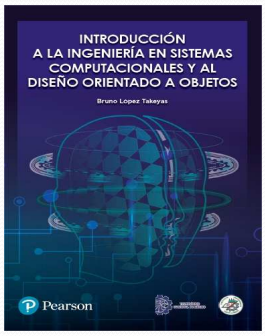
- Cada alimento **DEBE** implementar el método `CompareTo()` de la interfase `IComparable`
- El método `CompareTo()` compara los alimentos y los ordena de acuerdo a la distancia de la entrega
- La sentencia `where` de la clase `MotocicletaRepartidora` **OBLIGA** a que todos los objetos de su composición implementen el método `CompareTo()`

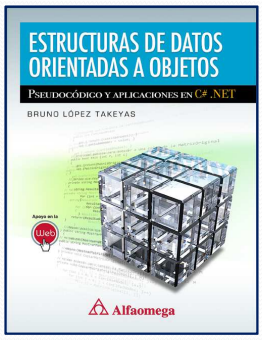
12

Otros títulos del autor

http://www.itnuevolaredo.edu.mx/Takeyas/Libro







bruno.lt@nlaredo.tecnm.mx

Bruno López Takeyas