

Programación II

C

Clase 9

Maximiliano Neiner

Temas a Tratar

- ❖ Herencia
- ❖ Derivación de Clases
- ❖ Uso de Clases Selladas

Temas a Tratar

Herencia

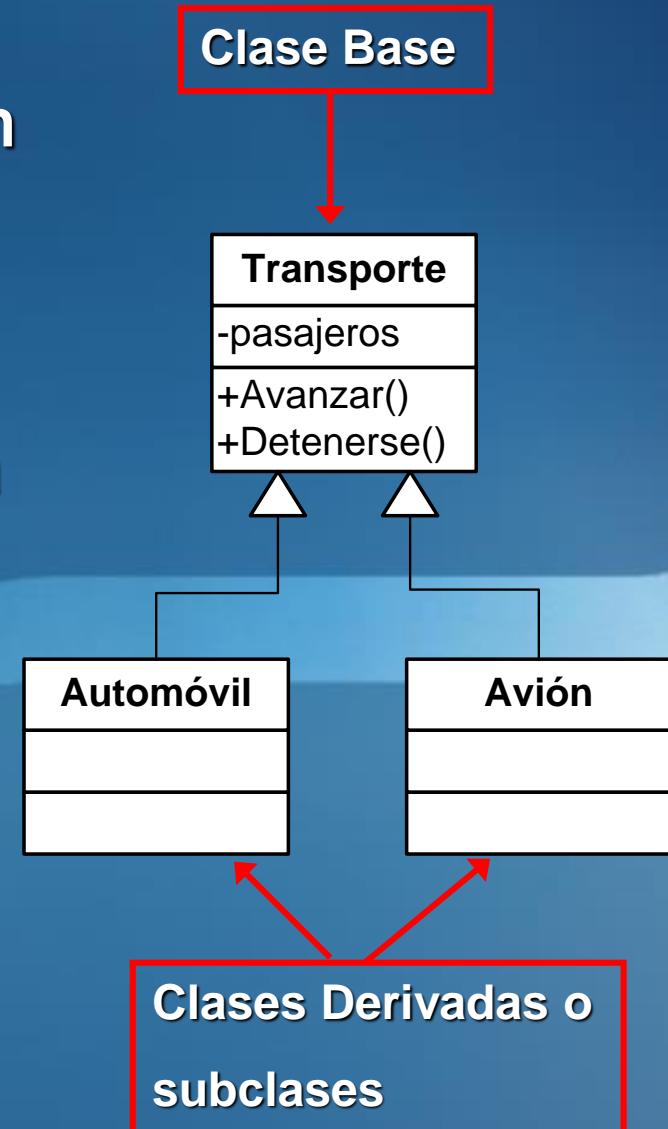
- Concepto.

Derivación de Clases

Uso de Clases Selladas

Concepto de Herencia (1/3)

- Es una relación entre clases en la cual una clase comparte la estructura y comportamiento definido en otra clase.
- Cada clase que hereda de otra posee:
 - Los atributos de la clase base además de los propios.
 - Soporta todos o algunos de los métodos de la clase base.
- Una subclase hereda de una clase base.



Concepto de Herencia (2/3)

- ❖ El propósito principal de la herencia es el de organizar mejor las clases que componen una determinada realidad, y poder agruparlas en función de atributos y comportamientos comunes. A la vez que cada una se especializa según sus particularidades.
- ❖ La herencia permite crear nuevas clases a partir de otras ya existentes (en lugar de crearlas partiendo de cero).
- ❖ La clase en la que está basada la nueva clase se llama *clase base* o *padre*, mientras que la clase hija se conoce como *clase derivada*.

Concepto de Herencia (3/3)

- ☒ Existen dos tipos de herencias:
 - ☒ Herencia Simple:
 - Una clase derivada puede heredar sólo de una clase base (los lenguajes .NET soportan este tipo de herencia)
 - ☒ Herencia Múltiple:
 - Una clase derivada puede heredar de una o más clases base (C++ es un ejemplo de lenguaje que soporta este tipo de herencia).

Temas a Tratar

▣ Herencia

▣ Derivación de Clases

- Extensión de Clases Base
- Acceso a Miembros de la Clase Base
- Llamada a Constructores de la Clase Base

▣ Uso de Clases Selladas

Sintaxis

[modificadores] class NombreClase : NombreClaseBase

Dónde:

- ✖ **Modificadores:** son modificadores de visibilidad y/o de clase.
- ✖ **Class:** Le indica al compilador que el bloque de código es una declaración de clase.
- ✖ **Operador (:) Le** indica al compilador que es una subclase de la clase que precede al operador.
- ✖ **NombreClaseBase:** Es el nombre de la clase padre.

Herencia de la clase derivada

- ▣ Una clase derivada hereda todo de su clase base, excepto los constructores.
- ▣ Los miembros públicos de la clase base se convierten implícitamente en miembros públicos de la clase derivada.
- ▣ Sólo los miembros de la clase base tienen acceso a los miembros privados de esta clase, aunque la clase derivada también los hereda.
- ▣ Una clase derivada no puede ser más accesible que su clase base. Por ejemplo, no es posible衍生 una clase pública de una clase privada.

Temas a Tratar

▣ Herencia

▣ Derivación de Clases

- Extensión de Clases Base
- Acceso a Miembros de la Clase Base
- Llamada a Constructores de la Clase Base

▣ Uso de Clases Selladas

Modificador Protected

- ❖ El significado del modificador de acceso **protected** depende de la relación entre la clase que tiene el modificador y la clase que intenta acceder a los miembros que usan el modificador.
- ❖ Para una clase derivada, la palabra reservada **protected** es equivalente a la palabra **public**.
- ❖ Entre dos clases que no tengan una relación base-derivada, por el contrario, los miembros protegidos de una clase se comportan como miembros **privados** para la otra clase.

Miembros Heredados

- ❖ Cuando una clase derivada hereda un miembro **protected**, ese miembro también es implícitamente un miembro protegido de la clase derivada.
- ❖ Esto significa que todas clases que deriven directa o indirectamente de la clase base pueden acceder a los miembros protegidos.
- ❖ Los métodos de una clase derivada sólo tienen acceso a sus propios miembros heredados con protección. No pueden acceder a los miembros protegidos de la clase base a través de referencias a ésta.

Ejemplo

```
class ClaseBase
{
    protected int _edad;
}

class ClaseDerivada : ClaseBase
{
}

class Clase : ClaseDerivada
{
    void Compila()
    {
        Console.WriteLine(_edad);
    }
}

class ClaseError : ClaseBase
{
    void Falla(ClaseBase t)
    {
        Console.WriteLine(t._edad);
    }
}
```

Temas a Tratar

▣ Herencia

▣ Derivación de Clases

- Extensión de Clases Base
- Acceso a Miembros de la Clase Base
- Llamada a Constructores de la Clase Base

▣ Uso de Clases Selladas

La Palabra Clave Base

- Para hacer una llamada a un constructor de la clase base desde un constructor de la clase derivada se usa la palabra reservada **base**, que tiene la siguiente sintaxis:

```
[modificadores] Constructor( [ListaArgs]) : base([ListaArgs]) {...}
```

Ejemplo

```
class ClaseBase
{
    protected int _edad;
    public ClaseBase(int edad)
    {
        this._edad = edad;
    }
}

class ClaseDerivada : ClaseBase
{
    public ClaseDerivada(int edad) : base (edad)
    { }
}
```

Constructores (1/2)

- ☞ Si la clase derivada no hace una llamada explícita a un constructor de la clase base, el compilador de C# usará implícitamente un constructor de la forma `:base()`.
- ☞ Esto implica que una declaración de constructor de la forma:

```
Constructor() {...}
```

- ☞ Es equivalente a:

```
Constructor() : base() {...}
```

Constructores (2/2)

- El comportamiento implícito anterior es válido en muchos casos porque:
 - Una clase sin clases base explícitas extiende implícitamente la clase **System.Object**, que contiene un constructor público sin parámetros (por defecto).
 - Si una clase no contiene ningún constructor, el compilador utilizará inmediatamente el constructor por defecto.
 - El compilador no creará un constructor por defecto si una clase tiene su propio constructor explícito.
 - No obstante, el compilador generará un mensaje de error si el constructor indicado no coincide con ningún constructor de la clase base.

Ejemplo

```
class ClaseBase
{
    protected int _edad;
    public ClaseBase(int edad)
    {
        this._edad = edad;
    }
}

class ClaseDerivada : ClaseBase
{
    public ClaseDerivada(int edad) // Error aquí
    { }
}
```

Demo

Temas a Tratar

- ❖ Herencia
- ❖ Derivación de Clases
- ❖ Uso de Clases Selladas
 - Concepto

Clases Selladas

- ❖ La mayor parte de las clases son autónomas y no están diseñadas para que otras clases deriven de ellas.
- ❖ Para que el programador pueda comunicar mejor sus intenciones al compilador y a otros programadores, C# permite declarar una clase como **sealed** (sellada).
- ❖ La derivación de una clase sellada no está permitida (no se puede heredar de ella).

Sintaxis

[modificadores] **sealed** class NombreClase

- ❖ Microsoft® .NET Framework contiene muchos ejemplos de clases selladas.
- ❖ Por ejemplo la clase **System.String**
- ❖ Esta clase está sellada y, por tanto, ninguna otra clase puede derivar de ella.

Demo