
Programación II

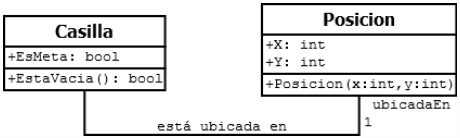
Relaciones entre clases

Las relaciones más comunes entre clases son:

- Asociación
- Agregación
- Composición
- Herencia

Asociación

- Es la relación más general, la que menos semántica lleva asociada.
- Denota la dependencia entre dos clases
- La representación es UML es una linea

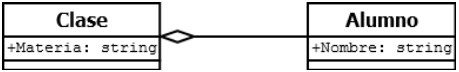


Cuando programamos dos clases relacionadas mediante una asociación deberiamos colocar un campo en cada clase con el endpoint de la relación para tener una referencia a la otra clase. Hay que tener en cuenta la cardinalidad para saber si solo necesitamos una variable o una lista.

```
class Casilla
{
    public Posicion ubicadaEn { get; set; }
}

class Posicion
{
    public Casilla Casilla { get; set; }
}
```

Agregación

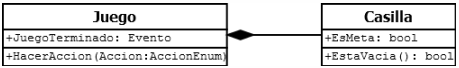
- Provee la semantica de **Todo/Parte de**
- No se realiza ninguna vinculación acerca del ciclo de vida de las clases vinculadas
- La cardinalidad de la relación generalmente es de 1 a muchos



```
class Clase
{
    public List<Alumno> Alumnos { get; set; }
}

class Alumno
{
    public Clase Clase { get; set; }
}
```

Composición

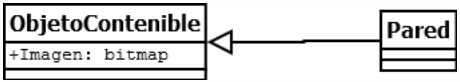
- Es una variación de la agregación.
- Expresa una relación de dependencia y asocia los ciclos de vida de las partes con el del todo
- El todo es reponsable del ciclo de vida de las partes, lo cual quiere decir, administrar la creación y destrucción



```
class Juego
{
    private List<Casilla> Casillas = new List<Casilla>();

    public void CrearCasilla() { }

    ~Juego()
    {
        foreach (Casilla c in Casillas)
        {  /*codigo necesario para eliminar la casilla   */  }
    }
}

class Casilla
{
    public Juego Juego { get; set; }
}
```

Herencia

- Permite expresar relaciones de generalización/especialización
- Una clase B que hereda de una clase A tiene todas las características de A más la características propias
- **La herencia puede ser:**
  - Simple: toda clase herede de una sola clase
  - Múltiple: una clase puede tener más de una clase padre



```
class ObjetoContenible
{
    public System.Drawing.Bitmap Imagen;
}

class Pared : ObjetoContenible
{

}
```

Ejercicio

- Implementar todas las clases definidas en el diagrama de clases del Sokoban, teniendo en cuenta las relaciones entre clases, sin importar por ahora la lógica del juego.