

UML

- [Diseño App](#)

Diseño App

UML

Unified Modeling Language UML. Es decir que estamos hablando de la unificación del lenguaje de modelos orientados a objetos. Suena un poco raro dicho del tirón pero no es más que un lenguaje que debemos considerar a la hora de crear nuestros diagramas en programación.

Atributos

Los atributos de una clase pueden ser de tres tipos:

1. público - public (+). El atributo será visible tanto fuera como dentro de la clase.
2. privado - private (-). El atributo sólo será accesible desde dentro de la clase.
3. protegido - protect (#). El atributo no será accesible desde fuera de la clase pero podrá ser usado por otros métodos de la clase o subclases que se extiendan.

Métodos

Son las funciones del programa, la forma en como se interactúa con el entorno, también son de tres tipos:

1. public (+). Método accesible de todos los lados.
2. private (-). Método no accesible desde fuera de la clase.
3. protected (#). Método no accesible desde fuera de la clase, pero si accesible por otros métodos de la clase y las subclases.

Relaciones

Herencia

Una subclase hereda los métodos y atributos de la clase padre. Además esta subclase podrá tener sus propios atributos y métodos.

[Herenciauml.png](#) Image not found. type unknown

Composición

Es un tipo de relación dependiente en la que un objeto más complejo es conformado por objetos más pequeños. Se podría usar la pregunta "Tiene un" para asegurarnos que es de este tipo.

[Composicion.png](#) Image not found. type unknown

Asociación

La asociación se da cuando dos objetos quieren trabajar juntos. Un punto a tomar muy en cuenta es que ambos objetos son independientes entre sí. Se podría usar la pregunta "Usa un".

[Asociacion.png](#) Image not found. type unknown

Agregación

Muy similar a la relación por asociación ya que solo varía en la multiplicidad, en vez de ser uno a uno es uno a muchos.

[Agregacion.png](#) Image not found. type unknown

Dependencia

Representa un tipo de relación en la que una clase es instanciada. El uso más particular de este tipo de relación es para denotar la dependencia que tiene una clase de otra

[Dependencia.png](#) Image not found. type unknown