Clases en Java

* La clase es la unidad fundamental de programación en Java.
* Un programa Java Orientado a Objetos está formado por un conjunto de clases. A partir de esas clases se crearán objetos que interactuarán entre ellos enviándose mensajes para resolver el problema.
* Una claserepresenta al conjunto de objetos que comparten una estructura y un comportamiento comunes.
* Puede considerarse como una *plantilla*o *prototipo* de objetos:define los **atributos** que componen ese tipo de objetos y los **métodos** que pueden emplearse para trabajar con esos objetos.
* Las clases incluyen por tanto atributos (variables de clase) y métodos (funciones o procedimientos). Los atributos definen el estado de cada objeto de esa clase y los métodos su comportamiento.
* Los atributos debemos considerarlos como la zona más interna, oculta a los usuarios del objeto. El acceso a esta zona se realizará a través de los métodos.

## El paquete (**package**)

* Los paquetes son una forma de organizar grupos de clases. Un paquete contiene un conjunto de clases relacionadas bien por finalidad, por ámbito o por herencia.
* Los paquetes resuelven el problema del conflicto entre los nombres de las clases. Al crecer el número de clases crece la probabilidad de designar con el mismo nombre a dos clases diferentes.
* Las clases tienen ciertos privilegios de acceso a los miembros dato y a las funciones miembro de otras clases dentro de un mismo paquete.
* El proyecto queda estructurado en disco de manera jerárquica correspondiendo cada paquete con una subcarpeta.
* ***package***, ***import***.

Constructor en Java

Un **constructor** es un método especial de una clase que se llama automáticamente siempre que se declara un objeto de esa clase.

Los objetos se crean con la palabra reservada ***new***.

Su función es inicializar el objeto y sirve para asegurarnos que los objetos siempre contengan valores válidos.

Cuando se crea un objeto en Java se realizan las siguientes operaciones de forma automática:

1.  Se asigna memoria para el objeto.

2. Se inicializan los atributos de ese objeto con los valores predeterminados por el sistema. (*Ejercicio para casa: descubrir cuáles son esos valores por defecto dependiente del tipo del atributo (int, String, boolean, etc.*)

3. Se llama al constructor de la clase que puede ser uno entre varios.

El constructor de una clase tiene las siguientes características:

* Tiene el mismo nombre que la clase a la que pertenece.
* En una clase puede haber varios constructores con el mismo nombre y distinto número de argumentos (se puede sobrecargar)
* No se hereda.
* No puede devolver ningún valor (incluyendo void).

Ámbitos de visibilidad

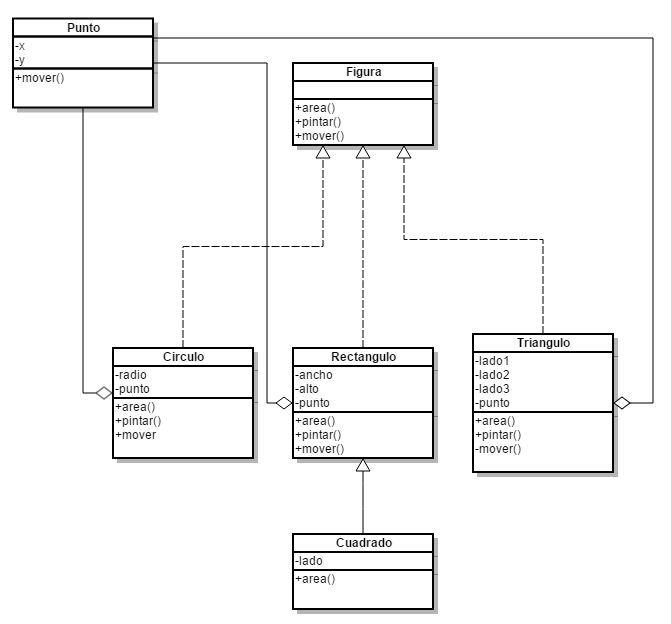
El **modificador de acceso** se utiliza para definir el nivel de ocultación o visibilidad de los miembros de la clase (atributos y métodos) y de la propia clase.

Los modificadores de acceso **ordenados de menor a mayor visibilidad** son:

|  |  |  |
| --- | --- | --- |
| **MODIFICADOR DE ACCESO** | **EFECTO** | **APLICABLE A** |
| **private** | Restringe la visibilidad al **interior de la clase**. Un atributo o método definido como private solo puede ser usado en el interior de su propia clase. | Atributos  Métodos |
| **<Sin modificador>** | Cuando no se especifica un modificador, el elemento adquiere el *acceso por defecto o friend*. También se le conoce como acceso de package (paquete). Solo puede ser usado por las **clases dentro de su mismo paquete**. | Clases  Atributos  Métodos |
| **protected** | Se emplea en la herencia. El elemento puede ser utilizado por **cualquier clase dentro de su paquete** **y por cualquier subclase** independientemente del paquete donde se encuentre. | Atributos  Métodos |
| **public** | Es el nivel máximo de visibilidad. El elemento es visible desde cualquier clase. | Clases  Atributos  Métodos |

Composición, Herencia y polimorfismo

Práctica. Imaginemos que necesitamos para un juego pintar en la pantalla diferentes figuras geométricas que podrán moverse de sitio. También necesitaremos calcular su área ya que, dependiendo de ella, se las figuras se moverán a diferente velocidad.



Composición: Círculo, Rectángulo, Triángulo y Cuadrado **tienen un** atributo Punto.

Herencia: Un Cuadrado **es un** Rectángulo y “*hereda*” todas las características de éste.

Polimorfismo: Nos permite tratar a todas las ***Figuras*** de la misma forma independientemente del tipo de que sean (reutilización del código) a la vez que nos da la flexibilidad para atender a particularidades (sobreescritura).

Clases Abstractas vs Interfaces

* Una clase abstracta es una clase que puede tener métodos sin implementar y por tanto no podremos crear instancias directamente de ella. ***abstract***
* Las subclases heredan o extienden: ***extends***
* Un interfaz es un conjunto de definiciones de métodos (sin implementar) y de constantes. ***interface***
* Las subclases implementan: ***implements***
* Una misma clase puede implementar 0, 1 o muchos interfaces.
* En java una misma clase sólo puede heredar de otra clase (aunque sea abstracta). Otros lenguajes orientados a objetos (c++, Python…) sí permiten la herencia múltiple.

Ejercicio Final:

Cambiar el proyecto de las firguras geométricas para que el interfaz Figura pase de ser una clase abstracta que implementa los métodos mover() y pintar() y deje sin implementar el método área() para que lo hagan cada una de sus subclases. *La clase que prueba el código no puede ser modificada!*