ADSI

Clases y Objetos Java - POO

Instructor: Gustavo Adolfo Rodríguez Q.

garodriguez335@misena.edu.co

ADSI

CLASES Y OBJETOS CON JAVA

Dentro de esta guía vamos a desarrollar los siguientes temas:

- Organización de las clases del API de Java.
- Tipos de Acceso.
- Tipos primitivos de datos.
- Tipos complejos de datos.
- Declaración de una clase.
- Declaración de atributos de la clase.
- Declaración de métodos de la clase.
- Declaración del constructor de la clase.
- Declaración de un objeto.
- Instanciación de una clase.

1. ORGANIZACIÓN DE LAS CLASES DEL API DE JAVA

El API, es un conjunto de librerías de clases que están disponibles para los programadores y cada una de ellas realiza una función específica.

Existe una jerarquía de clases dentro de Java y la raíz de toda esta jerarquía de clases es la clase **Object.**

Cuando creamos una nueva clase y no especificamos el padre de esta, Java interpreta que esta nueva clase hereda directamente de la clase **Object.**

Puesto que existe un gran número de clases y que algunas de ellas pueden llegar a tener nombres repetidos, es necesario organizar estas clases de alguna forma. En Java, las clases son organizadas en grupos, donde cada uno de estos grupos recibe el nombre de **paquete.**

Los nombres de los paquetes son un conjunto de palabras separadas por puntos que insinúan el propósito de las clases que alberga. Por ejemplo, el paquete donde se encuentran las principales clases del lenguaje es **java.lang.**

Otro ejemplo, es el paquete donde se encuentran las clases encargadas de manejar la entrada y salida de datos: **java.io**

Java nos permite también crear nuestros propios paquetes pero debemos seguir las convenciones básicas de nombrado de paquetes, entre las que se encuentran:

- Los nombres de los paquetes siempre se escriben en minúsculas.
- Los nombres de los paquetes son palabras separadas por puntos.

Algunos ejemplos de nombres válidos para los paquetes son:

- control
- vista
- org
- org.sena
- org.sena.comunicacion
- org.sena.control
- org.unicauca.modelo.datos

2. TIPOS DE ACCESO

Los tipos de acceso en java son especificados utilizando palabras reservadas del lenguaje, estas palabras son:

- **private :** Identifica el tipo de acceso privado.
- **protected:** Identifica al tipo de acceso protegido.
- **public :** Identifica al tipo de acceso público.

Estos tipos de acceso se aplican tanto a los atributos de la clase como a sus métodos.

3. TIPOS PRIMITIVOS DE DATOS

Los tipos primitivos de datos son aquellos que el lenguaje Java heredó del lenguaje C. Estos tipos primitivos no pertenecen a ninguna clase, por tanto, no son objetos sino variables.

Entre los tipos primitivos encontramos:

- **int**
- **long**
- **double**
- **float**
- **boolean**
- **byte**
- **char**

4. TIPOS COMPLEJOS DE DATOS

Los tipos complejos de datos son aquellos que hacen parte del lenguaje Java. Estos tipos de datos son clases que generalmente representan como objetos a los tipos primitivos de datos.

Entre los tipos complejos de datos encontramos:

- **Integer**
- **Long**
- **Float**
- **Boolean**
- **Byte**
- **String**
- **Cualquier otra clase del API o propia**

5. DECLARACION DE UNA CLASE

Para la declaración de una clase básica en Java que no hereda de ninguna otra clase se utiliza el siguiente modelo:

```
public class NombreClase {
...
}
```

Las palabras en color azul identifican las palabras reservadas del lenguaje. Las llaves marcan el inicio y el final de la clase.

Los tres puntos seguidos son simplemente para representar en este documento que existe contenido dentro de la definición de la clase. Sin embargo, cuando estemos escribiendo el código de la clase estos **no son necesarios**.

Para la declaración de una clase que hereda de otra clase se utiliza el siguiente modelo:

```
public class ClaseHija extends ClasePadre {
...
}
```

Las palabras en color azul identifican las palabras reservadas del lenguaje. Las llaves marcan el inicio y el final de la clase.

Los tres puntos seguidos son simplemente para representar en este documento que existe contenido dentro de la definición de la clase. Sin embargo, cuando estemos escribiendo el código de la clase estos **no son necesarios**.

Existen algunas convenciones con respecto a los nombres de las clases entre las que se encuentran:

- Los nombres de las clases siempre empiezan con letra Mayúscula.
- Si el nombre de la clase tiene más de una palabra se escribe la primera letra de cada palabra con mayúsculas.
- Los nombres de las clases no pueden empezar por número o caracteres especiales.

Algunos ejemplos de nombres **válidos** de clases son:

- Persona
- Auto
- ArbolMaderable
- MesaDeNoche
- AutoCompetencia
- Avion
- AvionComercial

Algunos ejemplos de nombre **no válidos** de clases son:

- persona
- 1auto
- &arbolmaderable
- mesadenoche
- Autocompetencia
- Avion..
- AvionComercial#@

6. DECLARACION DE ATRIBUTOS DE LA CLASE

Cuando se desea declarar un tributo de clase se debe seguir el siguiente esquema:

tipo_de_acceso **TipoDeDato nombreAtributo** ***;***

Donde el tipo de acceso puede ser public, protected o private.

Tipo de dato puede ser un tipo primitivo o cualquier tipo complejo.

Existen algunas convenciones para el nombrado de atributos en Java entre las que se encuentran:

- Los nombres de los atributos siempre empiezan con letras minúsculas.
- Si el nombre del atributo tiene más de una palabra, la primera palabra se escribe en minúsculas y la primera letra de la segunda palabra es mayúscula.
- Los nombres de los atributos no pueden empezar por número o caracteres especiales

Algunos ejemplos de declaración válida de atributos de clase son:

- public int edad;
- protected boolean mayorDeEdad;
- private String nombre;
- private Integer estatura;
- private long milímetros;

Algunos ejemplos de declaración no válida de atributos de clase son:

- public edad int;
- publico boolean mayorDeEdad;
- private String NOMBRE;
- private Integer Estatuara.

7. DECLARACION DE METODOS DE LA CLASE

Para realizar la declaración de un método en una clase se debe seguir este esquema:

tipo_de_acceso **TipoDeRetorno nombreDelMetodo (TipoDato param1) {**

…

}

Donde el tipo de acceso puede ser public, protected o private.

TipoDeRetorno puede ser cualquier tipo de dato bien sea primitivo o complejo, o la palabra reservada ***void*** cuando el método no retorna nada.

Entre paréntesis y separados por comas se deben describir cada uno de los argumentos que necesita el método para su funcionamiento.

Existen algunas convenciones para el nombrado de método de clase en Java entre las que se encuentran:

- Los nombres de los métodos siempre empiezan con letras minúsculas.
- Si el nombre del método tiene más de una palabra, la primera palabra se escribe en minúsculas y la primera letra de la segunda palabra es mayúscula.
- Los nombres de los métodoss no pueden empezar por número o caracteres especiales

Algunos ejemplos de declaración válida de métodos de clase son:

- public int calcularEdad () { … }
- protected void reir (int tiempoRisa) { … }
- private String obtenerApellido() { … }
- private void comer () { … }

Algunos ejemplos de declaración no válida de atributos de clase son:

- public int CAlcularEdad () { … }
- protected REIR (int tiempoRisa) { … }
- private String obtenerapellido() { … }
- private void 1#comER () { … }

8. DECLARACION DEL CONSTRUCTOR DE LA CLASE

El método constructor de una clase, es método especial que permite la creación de objetos de una clase particular. Este método no sigue las mismas normas que rigen la declaración de los otros métodos de la clase.

Para crear un método constructor de una clase se sigue un esquema como el mostrado a continuación:

public **NombreDeClase ( ) { … }**

El nombre del método constructor debe ser el mismo nombre de la clase a la cual pertenece.

9. DECLARACION DE UN OBJETO

Para la declaración de un objeto que pertenece a una clase particular se siguen los mismos lineamientos que para la declaración de atributos. La única diferencia es que ya no es necesario declara algún tipo de acceso.

Para declara un objeto de una clase particular se sigue el esquema mostrado a continuación:

TipoDeDato nombreObjeto ***;***

TipoDeDato puede ser cualquier tipo bien sea primitivo o complejo, que pertenezca a las clases del API o que pertenezca al nuestras propias clases.

Los siguientes son ejemplos válidos de declaración de objetos:

- String mensaje;
- String saludo;
- Persona juanito;
- Avion fuerzaAereaUno;

10. INSTANCIACIÓN DE UNA CLASE

Para la instanciación de una clase es necesario primero haber declarado un objeto del tipo de clase que se desea instanciar.

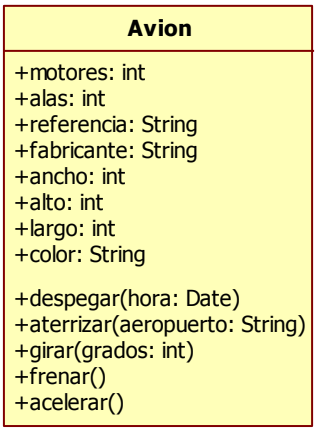
La instanciación de clases se realiza de la siguiente forma

nombreObjeto = *new* ConstructorDeClase ( );

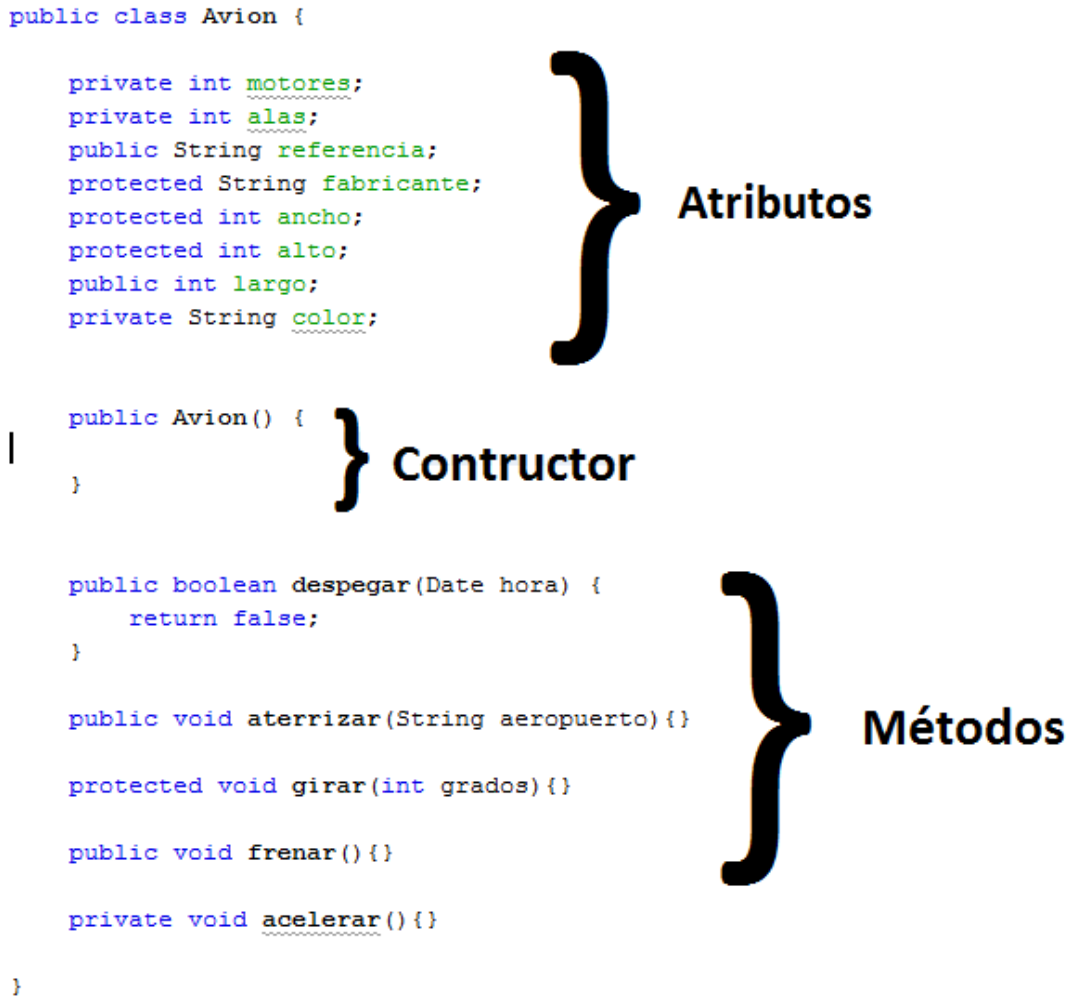
Los siguientes son ejemplos de instanciación de clases de los objetos declarados en la sección anterior:

- mensaje = new String();
- saludo = new String();
- juanito = new Persona();
- fuerzaAereaUno = new Avion();

Para unificar los conceptos expuestos en el documento, veamos cómo se realiza el paso desde los diagramas UML de la clase **Avion** hacia el código Java.



A continuación se muestra el código Java de la clase **Avion** del diagrama UML anterior.

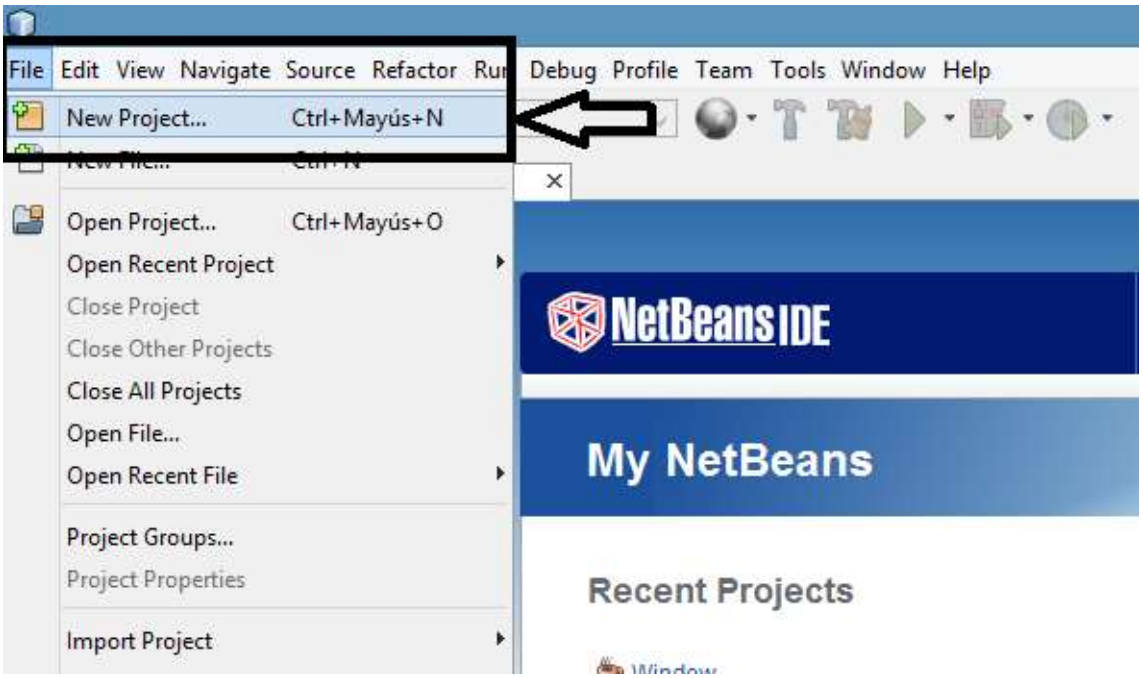


11. CREACION PROYECTO

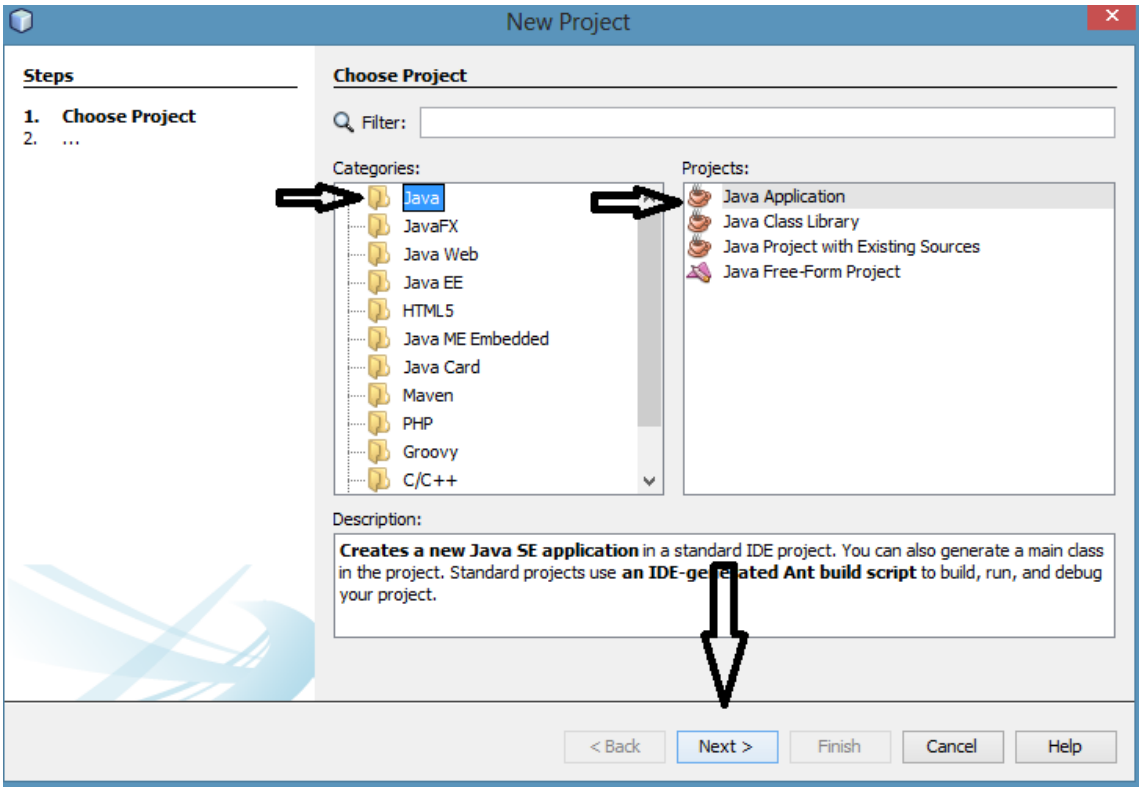
Para la creación de un proyecto Java en *NetBeans* vea el video tutorial que se encuentra en los recursos crearProyectoJava_demo.swf.

Al crear el primer proyecto puedes ver la estructura inicial en la parte izquierda del IDE, encontraras un paquete con el nombre del proyecto y una clase con el nombre del proyecto o main. En mi caso cree un proyecto que llame test, creó el paquete inicial con la clase que contiene el método ***main(), test/Main.java.***

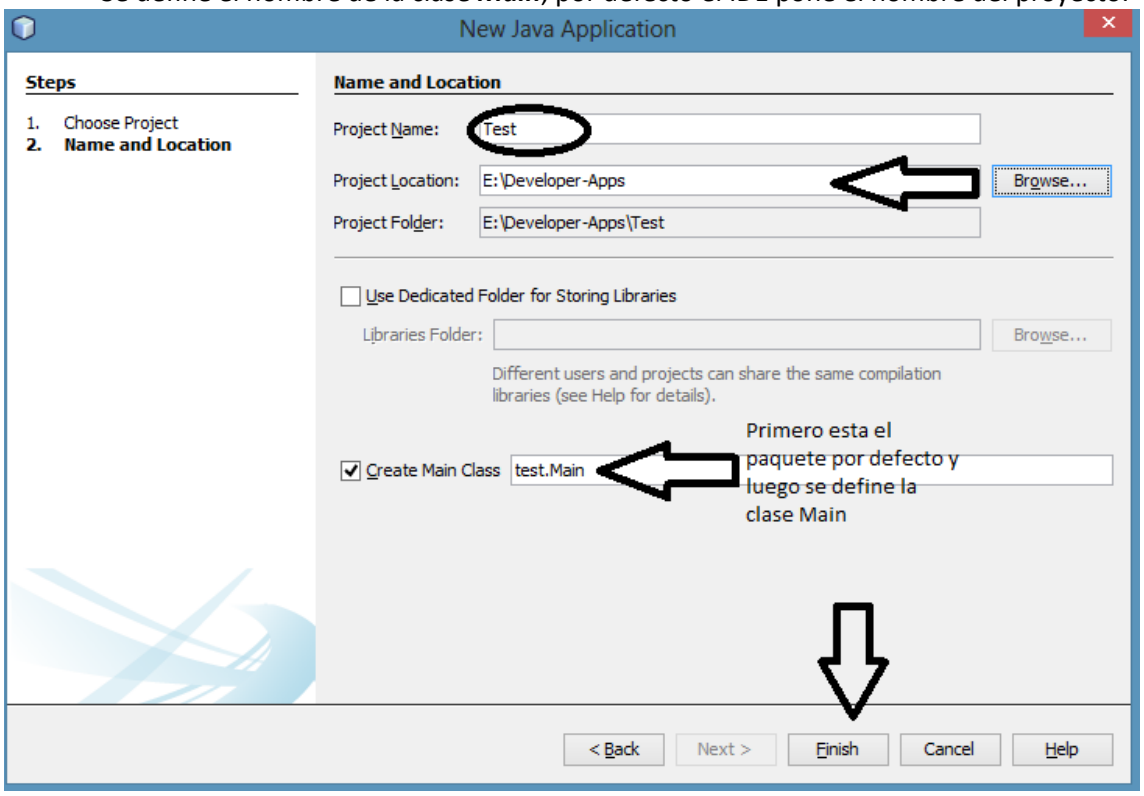
- Ir a ***File*** o ***Archivo***
- Seleccionar ***New Project*** o ***Nuevo Proyecto***



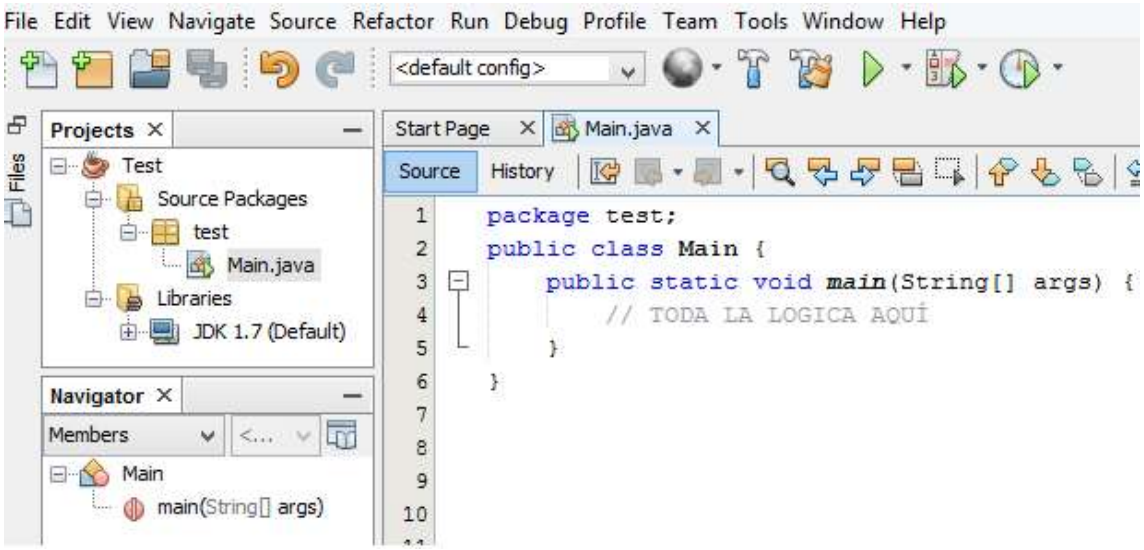
- Se abre una ventada donde se debe seleccionar la categoría ***Java***
- Luego se selecciona el tipo de proyecto ***Java Application***



- Se define el nombre del proyecto
- Se selecciona la ruta donde se guardara el proyecto
- Se define el nombre de la clase ***Main***, por defecto el IDE pone el nombre del proyecto.



- En la parte izquierda estarán los paneles relacionado con los archivos, proyectos y navegación.
- Al crear el proyecto veremos el nombre del proyecto
- ***Sourse Packages:*** ruta donde se deben poner las clases y paquetes fuentes.
    - ***test:*** paquete inicial del proyecto donde está la clase Main o inicializadora.
        - ***Main.java:*** Clase inicializadora con método ***main()***;
- ***Libraries:*** JDK de Java, y Librerías externas.
- Parte central y principal, visor de código.



12. Clase principal y método *main()*

Todo proyecto en Java debe tener una clase principal que generalmente se llama como el proyecto, esta clase provee el método ***main().*** Podemos decir que esta es la clase es la iniciadora de nuestro proyecto o el hilo principal de ejecución, aunque esto no quiere decir que sea el protagonista de todo el programa.

13. Diagrama de Clases