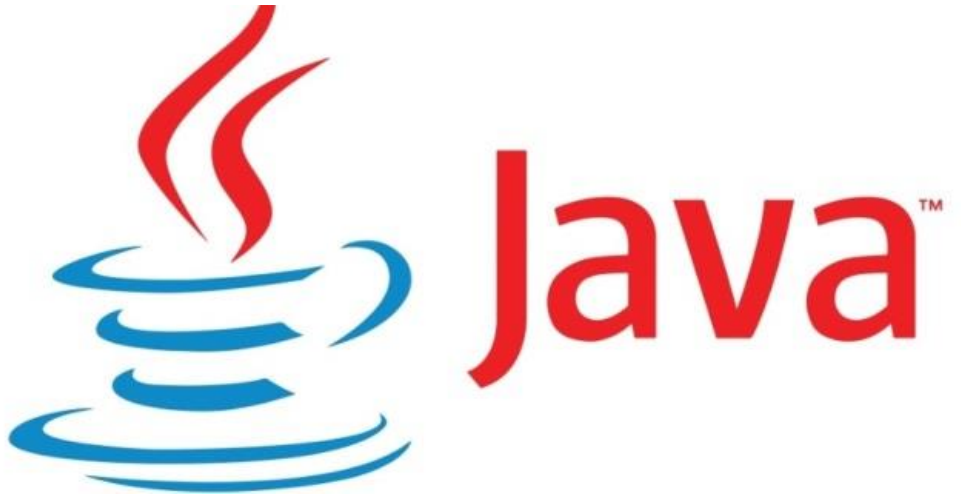
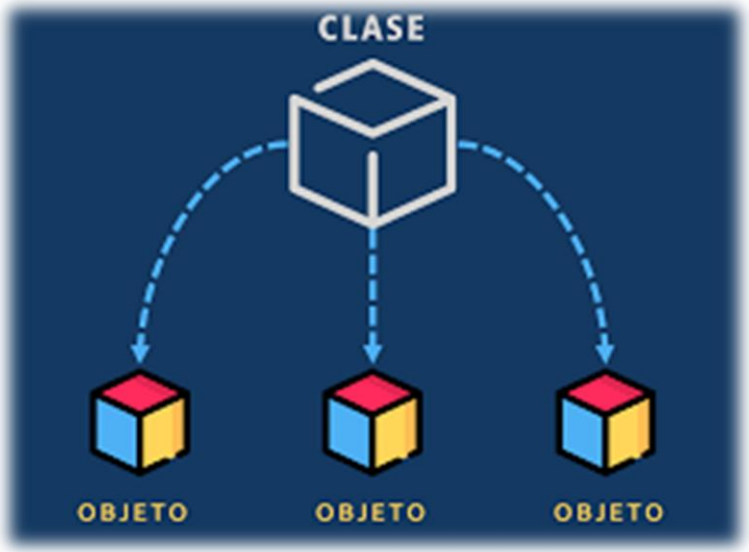


Java 8 SE Fundamentals

Describing Objects and Classes

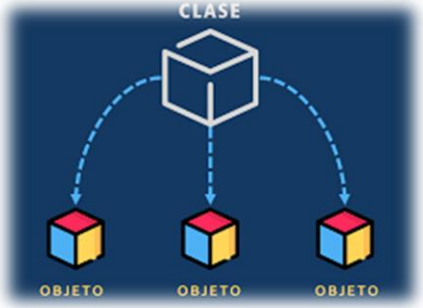
- ✓ Working with objects and classes
- ✓ Defining fields and methods
- ✓ Declaring, Instantiating, and Initializing Objects
- ✓ Working with Object References
- ✓ Doing more with ArraysList
- ✓ Introducing the Soccer League Use Case

Trabajando con objetos y clases.



Programación orientada a objetos.

Establece que todos los problemas deberían ser resueltos mediante entidades que absorban un estado interno y un las tareas posibles sobre estado (el objeto), con un diseño que permita entender la arquitectura de cada entidad (la clase).



Objetos.

En el mundo real un objeto es una entidad, un nombre o cualquier cosa que tenga una identidad propia.

Objetos.

Objetos.


RESERVED

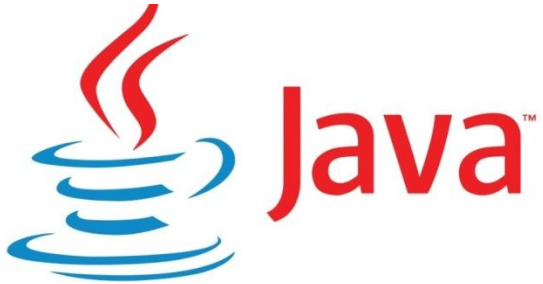

amazon
Online Shopping
Online Shopping


EL PROYECTO ADAM
NETFLIX

Objetos.

Objetos.

Objetos.



Los objetos se componen de *atributos* y *métodos* que pueden ser accesibles desde dentro del objeto o desde fuera, según los niveles de acceso diseñados.

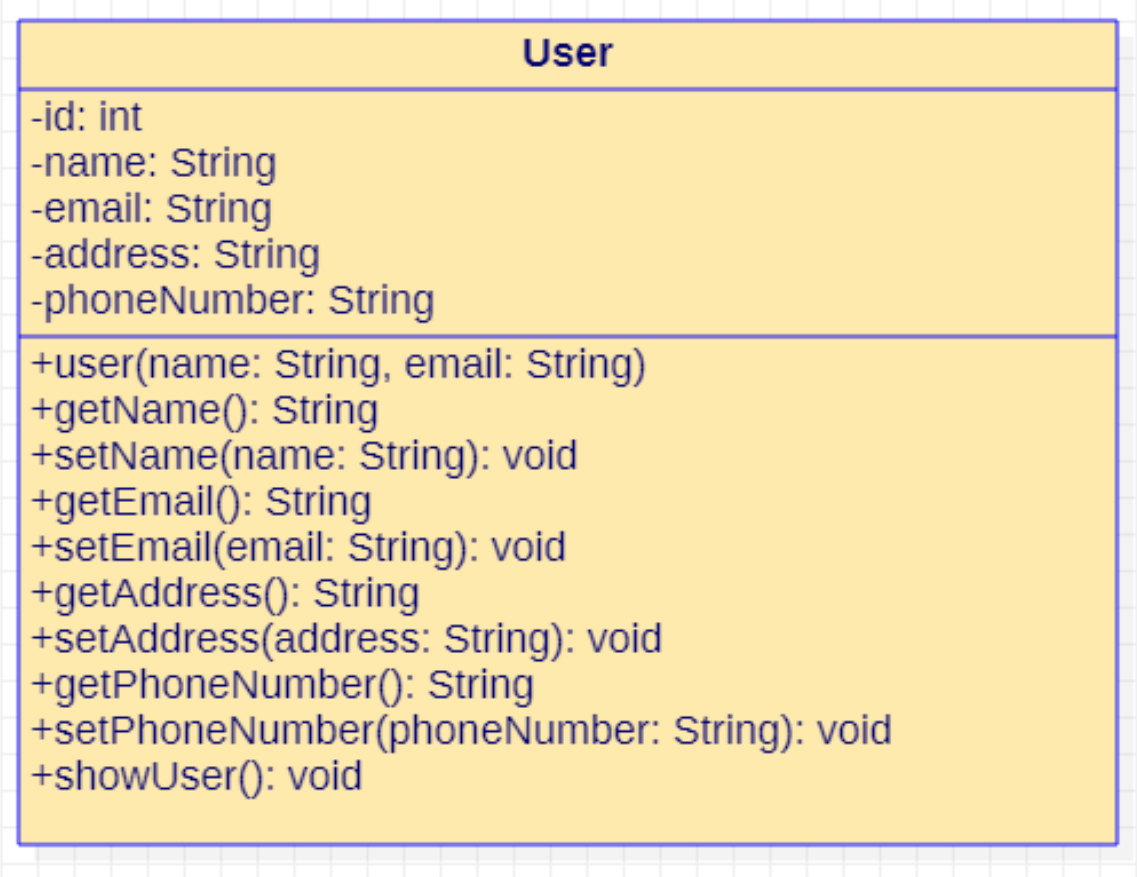
La entidad en cuestión será la Persona. Cada persona podrá retener datos que le describan, por ejemplo, el nombre, la edad, su correo, etc. A estos datos les llamaremos *los atributos de la entidad*.

Clase.

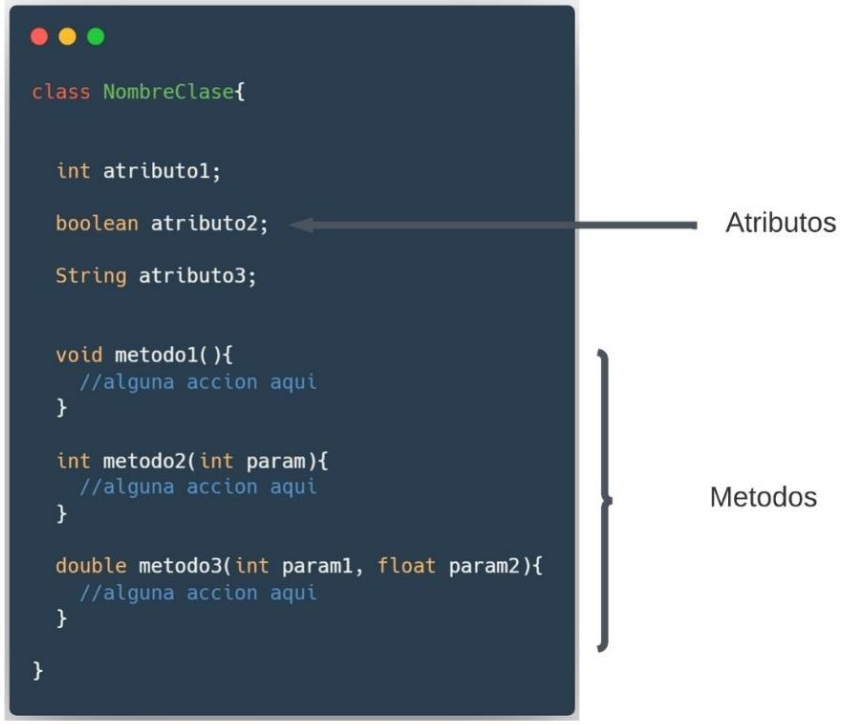
Es un prototipo (plantilla) que contiene una colección de datos (atributos) y un conjunto de métodos (comportamiento) que manipulan los datos o que interactúan con objetos relacionados.

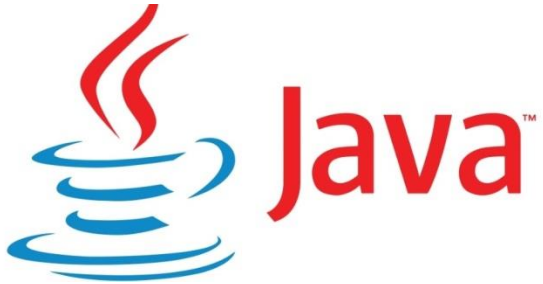
A los datos y métodos se les conoce como miembros de la clase.

Diagrama UML de una clase en Java.



Definiendo atributos y métodos de una clase.

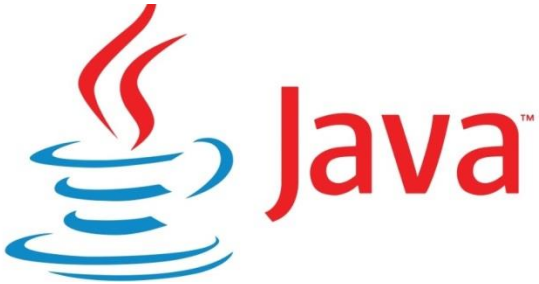


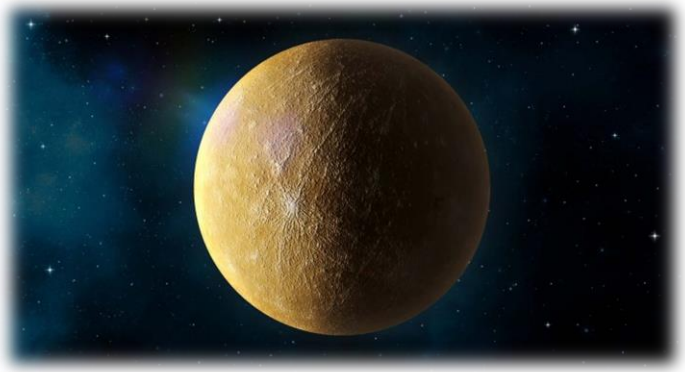


Ejercicio.

1. Crear la clase User antes descrita en Java.

Objetos de la clase Planeta.





Mercurio

Venus

Tierra

Marte

Ejercicio.

1. Diseñar en un bloc de notas la clase Planeta.

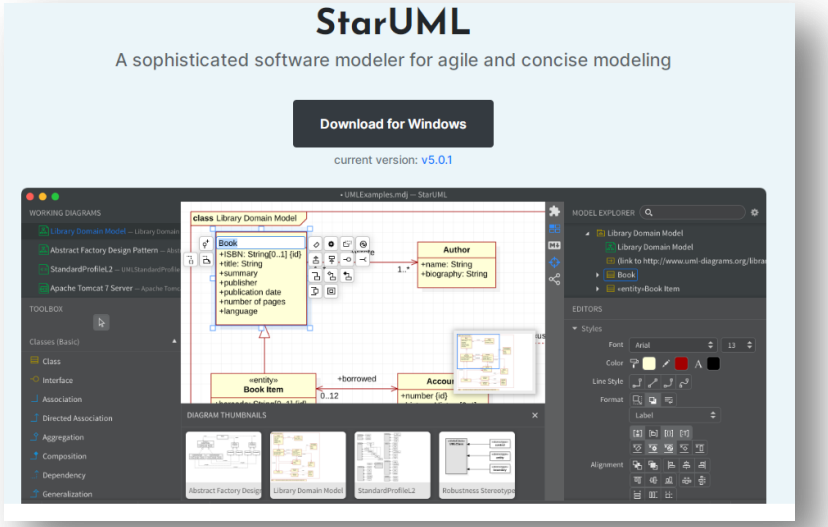
2. Crear el diagrama UML en StarUML de la clase Planeta y luego crearla en Java (IntelliJ).

Software para crear diagramas UML.

1. Sitio oficial de StarUML

   https://staruml.io/

2. Download



Objetos de la clase Automóvil

BMW i8

Ford Bronco

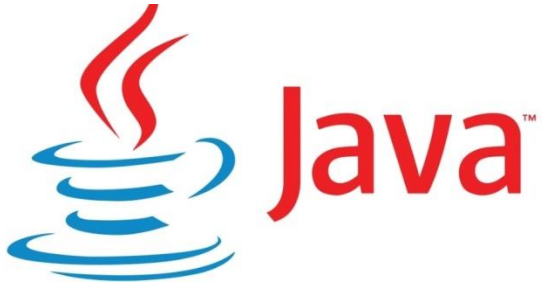
Subaru BRZ

Chevrolete Corvette

Ejercicios.

1. Diseñar en un bloc de notas la clase Automóvil.

2. Crear el diagrama UML en StarUML de la clase Automóvil y luego crearla en Java (IntelliJ).



Declarando, instanciando e inicializando objetos.

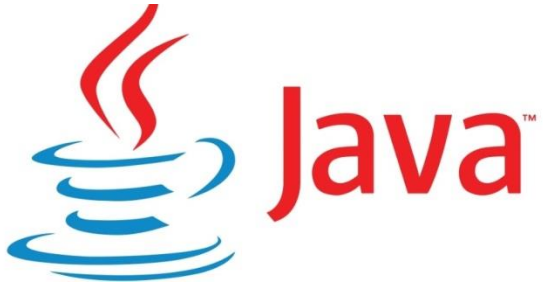
Las clases son definiciones de objetos y los objetos se crean a partir de las clases.

```java
public class Pruebas {

 /** Metodo principal */
 public static void main(String[] args) {

 // Creamos un circulo con radio = 1;

 Circulo circulo1 = new Circulo();
 System.out.println("El area del circulo 1 de radio "
 circle1.radio + " es " + circulo1.getArea());

  // Creamos un circulo con radio = 1, y despues es cambiado;

 Circulo circulo2 = new Circulo();
 circulo2.setRadio(25);
 System.out.println("El area del circulo 2 de radio "
 circle2.radio + " es " + circulo2.getArea());

 // Modificamos el radio de circulo1
 //calculamos de nuevo el area
 circulo1.radio = 8;
 System.out.println("El area del circulo 1 de radio "
 circle1.radio + " es " + circulo1.getArea());

}
```

Trabajando con referencias a objetos. Ejercicios.

1. Crear objetos de la clase Planeta y aplicarles el método mostrarPlaneta().

2. Crear objetos de la clase Automóvil y aplicarles el método mostrarAuto().



Pilares de la programación orientada a objetos.

Existen un conjunto de ideas fundamentales que forman los cimientos del desarrollo de software. A estos 4 conceptos que vamos a ver les llamamos los **4 pilares** de la programación orientada a objetos.

Estos pilares son: **encapsulamiento, abstracción, herencia y polimorfismo.**

Encapsulamiento.

El término **encapsulamiento** en Java, consiste en ocultar atributos de un objeto de manera que solo se pueda cambiar mediante operaciones definidas en ese objeto.

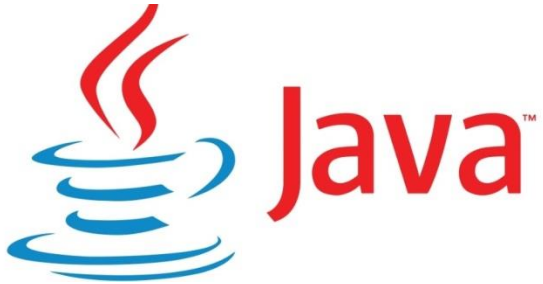
Encapsulamiento.

public: nos indica que es accesible desde cualquier clase [interface].

private: solo es accesible desde a clase actual.

protected: accesible desde la clase actual, sus descendientes o el paquete del que forma parte.

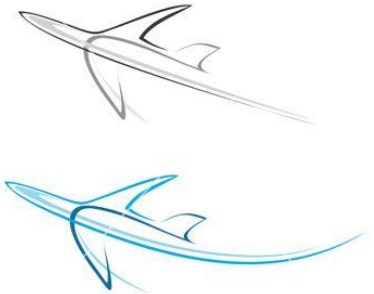
sin ninguna palabra: accesible desde cualquier clase del paquete.

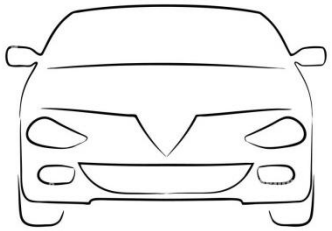


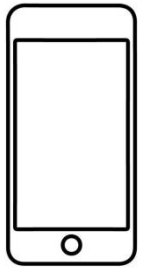
Ejercicios.

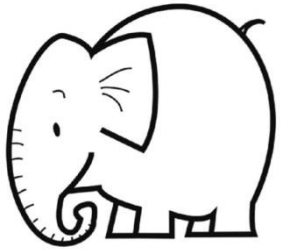
1. Encapsulemos la clase Automóvil.

Abstracción.





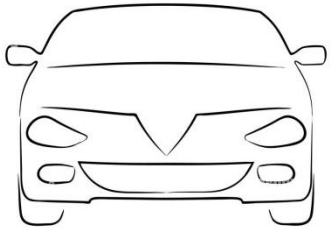


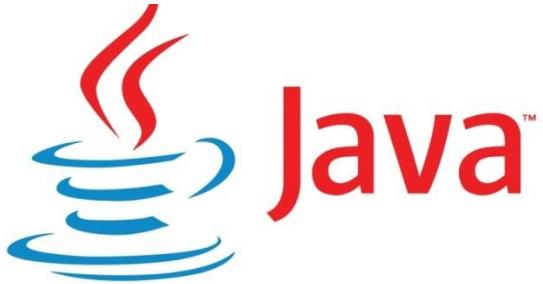


Abstracción.

Según la RAE, abstracción es: *“Hacer caso omiso de algo, o dejarlo a un lado.”*

Y ofrece como ejemplo: *“Centremos la atención en lo esencial abstrayendo DE consideraciones marginales.”*





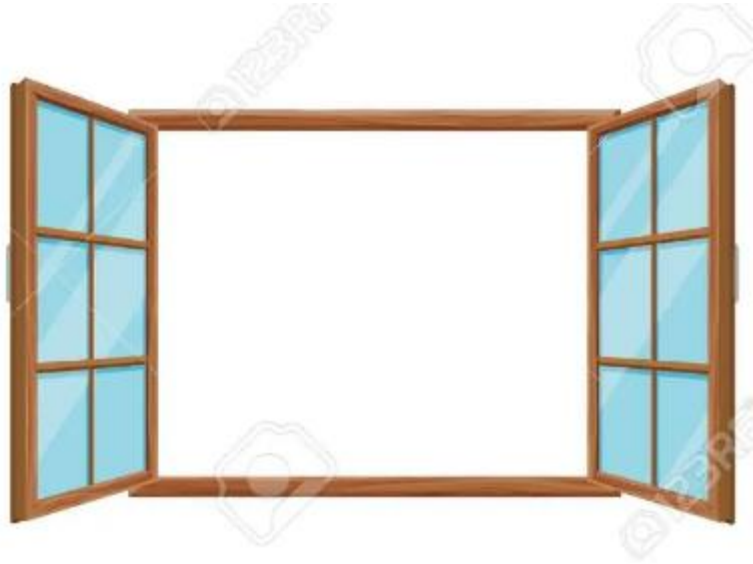
Herencia.

Compartir código es una importante y crucial característica de cualquier proyecto de software.

Compartir código permite ahorrar trabajo cuando queremos hacer un cambio en nuestro sistema.

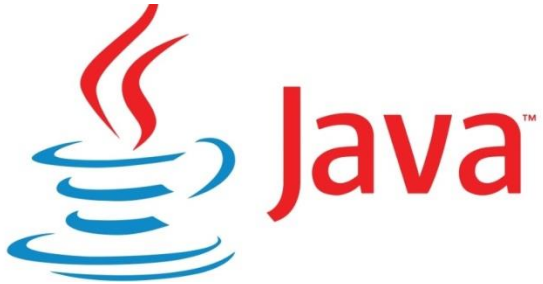
Polimorfismo.

Polimorfismo.



Polimorfismo.

Polimorfismo significa de muchas formas. En nuestro caso llamamos polimorfismo cuando un método recibe un parámetro que abarca varios tipos.



Ejercicio.

1. Hagamos una práctica orientada a una clínica deportiva que ilustre: encapsulamiento, abstracción, herencia y polimorfismo.

Haciendo más con ArrayList.

Afortunadamente una vez que sabemos usar ArrayList, ésta misma nos sirve para almacenar objetos de cualquier clase como muchos objetos de la clase Doctor o de la clase Deportista por ejemplo.

Ejercicio.

1. Crear en Java una lista de Personas que contenga Doctores y Deportistas.

Introducing the Soccer League Use Case

https://creately.com/diagram/example/if4sdljl3/soccer-league