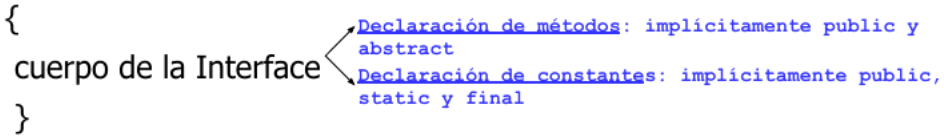
Clase 2-abstract e interface

miércoles, 27 de agosto de 2025 14:41

Abstract: concepto abstracto, no instanciable. Da la interface de un objeto y no una implementación particular.
Las clases abstractas pueden tener algo de implementación.
Manipular un conjuinto de clases a través de una interface común.
Las subclases tienen que tener implementación para todos los métodos abstractos de la superclasse

Interface: colección de definiciones de métodos sin implementación y de declaraciones.
Solo define firmas.
Sí o sí los métodos son públicos.
No se pueden instanciar
Una clase puede implemnentar varias interfaces.
En interfaces hay tipo herencia múltiple:
podes implementar varias interfaces:
Porque si se llega a repetir el nombre de un método no pasa nada porque no tiene nada xd.



Pública, constante y de clase.
Pueden definirse métodos de clase?
Se contradice ser abstract y static a la vez: abstract es que se reimplementa, static es que esa es la implementación para todos.

```
public interface Centrable
{  void setCentro(double x, double y);
   double getCentroX();
   double getCentroY();
}
```
Declaración de Métodos

```
public interface Posicionable extends Centrable
{  void setEsquinaSupDer(double x, double y);
   double getEsquinaDerX();
   double getEsquinaDerY();
}
```
Declaración de Métodos

Si implementa posicionable debe implementar 6 métodos

- Las **interfaces** no tienen implementación, por ende no tienen almacenamiento asociado y en consecuencia **no causa ningún problema combinarlas**.
- En JAVA una clase puede implementar tantas interfaces como desee. Cada una de estas interfaces provee de un tipo de dato nuevo y solamente la clase tiene implementación. Por lo tanto se logra un mecanismo de combinación de interfaces sin complicaciones. Las interfaces son una alternativa a la **herencia múltiple**.

Interfaz marker:
Solo tiene un nombre.
Cuando una clase implementa eso, es una marca.
Serializable es para marcar si se puede escribir o leer un dato, es como que si implementa serializable, entonces puede escribirse o leerse. No trae nada más.,

Le agregan implementaciones a las interfaces porque necesito poder actualizar interfaces. Sino, si cmab io la interfaz tengo que actualizar todas las clases que la implementan.

- En algunas situaciones la elección es de diseño.
- Es posible combinar clases abstractas e interfaces: definir un tipo como una interface y luego una clase abstracta que la implementa parcialmente, proveyendo implementaciones de defecto que las subclases aprovecharán (*skeletal implementation*). Patrón **Template Method.** En colecciones se usa este patrón: **AbstractSet, AbstractList, AbstractMap, etc.**

```
abstract class Mamifero {
public abstract void comer();
}
```

```
abstract class Mascota{
public void respirar(){…}
public abstract void entretener();}
```

Java NO soporta herencia múltiple de clases, por lo tanto si se desea que una clase sea además del tipo de su superclase de otro tipo diferente es necesario usar interfaces.

Desde java 8 se meten métodos default o defender para que sean una implementación predeterminada. Son métodos definidos dentro de la interfaz.

Herencia y polimorfismo

```
public class Vertebrado{
private int cantpatas;
public Vertebrado(){
    System.out.println("Constructor de Vertebrado");
}
public void desplazar(){
    System.out.println("Vertebrado.desplazar()");
}
public void comer(){
System.out.println("Vertebrado.comer()");
}
}
```

```
public class Mamifero extends Vertebrado{
public Mamifero(){
      System.out.println("Constructor de Mamifero");
}
public void comer(){
      System.out.println("Mamifero.comer()");
}
}
```

```
public class Perro extends Mamifero{
private String nom;
public Perro(){
      System.out.println("Constructor de Perro");
}
public void setNombre(String n){this.nom=n;}
public String getNombre(){return nom;}
public void comer(){
      System.out.println("Perro.comer()");
}
public void jugar(){
      System.out.println("Perro.jugar()");
}
public static void main(String[] args){
      Perro p=new Perro();
}
}
```

Laboratorio de Software - Claudia Queiruga

Cuando llamo al constructor de perro, llama al constructor de mamifero y de vertebrado.
Cuando paso parámetros,