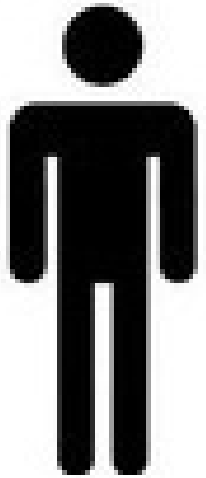
POO: Clases Abstractas

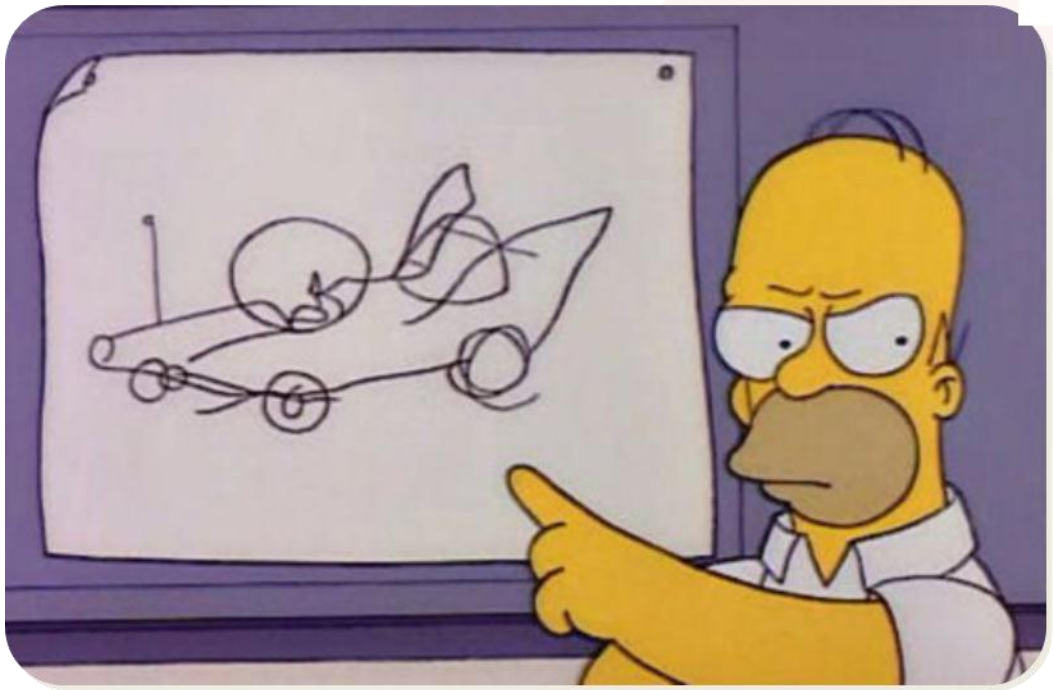
Clases Abstractas

La abstracción permite resaltar la parte mas representativa de algo, ignorando detalles para centrarse en lo principal.



Podemos decir que lo que se ve es un hombre, aquí estamos aplicando el concepto de abstracción, ya que nos fijamos en lo mas representativo de algo, en este caso vemos que se tiene una cabeza, tronco, brazos y pies, con esto es suficiente para saber que es una persona sin fijarnos en los detalles mencionados anteriormente.

Abstracción



Homero Simpson construyendo el auto de sus sueños

Énfasis en el ¿qué hace? mas que en el ¿cómo lo hace?

Clases Abstractas

La Abstracción en java solo tiene lógica mediante la Herencia, ya que **una clase abstracta posee al menos un método abstracto el cual no tiene implementación**, el comportamiento de estos métodos lo definen las clases concretas que lo hereden.

Las clases abstractas permiten crear métodos generales con un comportamiento común para otras clases concretas sin importar sus características ni el comportamiento que usen para dichos métodos.

Clases Abstractas

Las clases Abstractas pueden ser usadas cuando existan varias clases con características o acciones comunes pero con diferentes comportamientos.

Mediante el uso de la herencia y componentes abstractos hacemos mas óptima y organizada nuestra aplicación.

Se debe tener en cuenta que a diferencia de las clases concretas, las clases abstractas no se pueden instanciar. (crear objetos de estas)

Clases Abstractas

En Java representamos la abstracción mediante la palabra reservada **abstract**

```java
public abstract class Principal{

    /**Método Abstracto sin implementación*/
    public abstract void metodoAbstracto();

}
```

Método sin implementación

```java
class subClase extends Principal{

 @Override
 public void metodoAbstracto() {
  /**Implementación definida por la clase concreta*/
 }

}
```

Gracias a la herencia se puede usar el método para darle el comportamiento deseado

Clases Abstractas

Reglas:

- Una clase Abstracta No puede ser instanciada (no se pueden crear objetos directamente - new), solo puede ser heredada.
- Si al menos un método de la clase es **abstract**, esto obliga a que la clase completa sea definida **abstract**, sin embargo la clase puede tener el resto de métodos no abstractos.
- Los métodos **abstract** no llevan cuerpo (no llevan los caracteres {}).
- La primer subclase concreta que herede de una clase **abstract** debe implementar todos los métodos de la **superclase**.

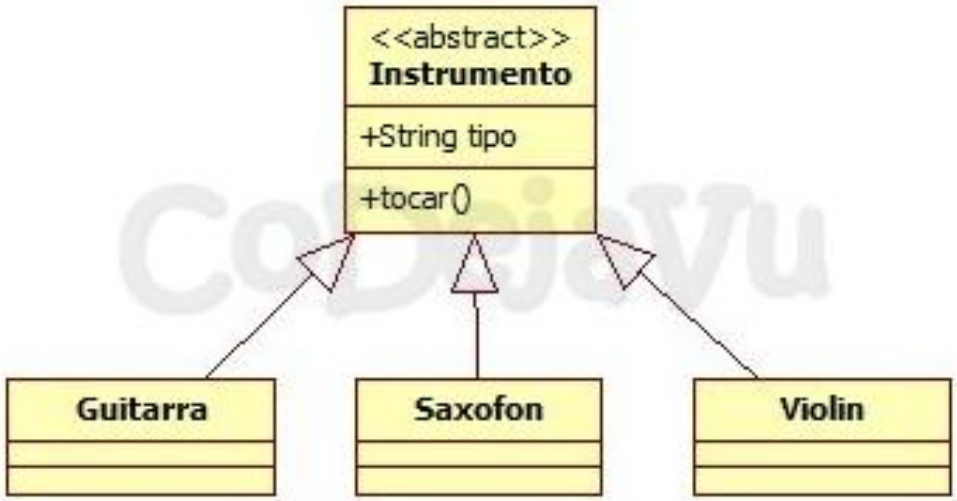
Clases Abstractas

Al trabajar clases y métodos abstractos, no solo se mantiene la aplicación mas organizada y fácil de entender sino que también al no poder instanciar una clase abstracta se asegura que las propiedades especificas de esta, solo estén disponibles para sus clases hijas.

Con las Clases Abstractas lo que se hace es definir un proceso general que luego será implementado por las clases concretas que hereden dichas funcionalidades

Clases Abstractas

Si se tiene una clase que hereda de otra Abstracta, java obliga a poner en el código todos los métodos abstractos de la clase padre para que sean implementados, así se convierten en concretos y su funcionalidad o cuerpo será definido dependiendo de para que la necesite, de esa manera si se tiene otra clase que también hereda del mismo padre, se implementará el mismo método pero con un comportamiento distinto



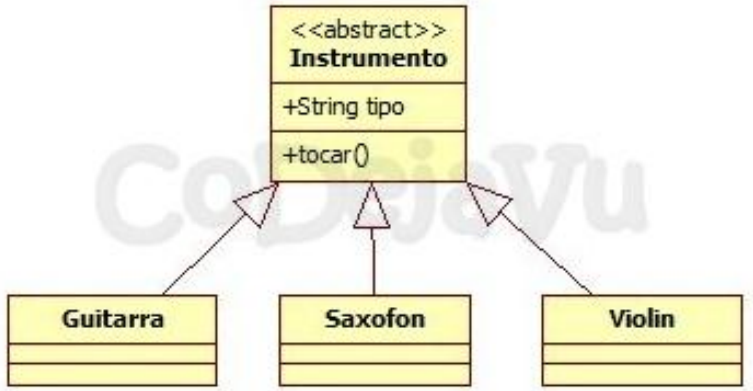
Todos los instrumentos musicales comparten el método tocar() por medio de la herencia con la clase Instrumento, sin embargo cada instrumento se toca de manera distinta, por lo tanto la implementación del método depende de cada clase.

Clases Abstractas

```java
/**
 * Clase Abstracta Instrumento
 */
abstract class Instrumento{

 public String tipo;

 public abstract void tocar();
}
```




```java
/**
 * Clase Concreta Guitarra, hija de Instrumento
 */
class Guitarra extends Instrumento {

 public Guitarra(){
  tipo="Guitarra";
 }

 public void tocar() {
  System.out.println("Tocar La Guitarra");
 }
}
```

```java
/**
 * Clase Concreta Violin, hija de Instrumento
 */
class Violin extends Instrumento {

 public Violin(){
  tipo="Violin";
 }

 public void tocar() {
  System.out.println("Tocar El violin");
 }
}
```

FUENTES

- Fundamentos de programación. Problemas resueltos de programación en lenguaje Java. José María Pérez Menor, Jesús Carretero Pérez, Félix García Carballeira, and José Manuel Pérez Lobato. Madrid: Paraninfo, 2003. p[1]-22

- http://codejavu.blogspot.com/2013/05/conceptos-de-programacion-orientada.html

- http://codejavu.blogspot.com/2013/05/clases-abstractas.html

- JavaWorld – Revista Digital