• Asignan un comportamiento a las clases que la implementan.

• Permite aislar clases haciendo el código mas reutilizable.

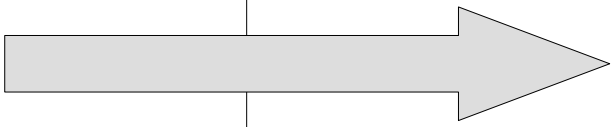
• Se declaran los métodos sin definirlos.

• Obliga a la clase que implementa la interfaz a definir dichos métodos.

• Puede incluir la declaración de constantes.

• Diferencia: Una clase abstracta puede incluir métodos implementados y no implementados o abstractos, atributos constantes y otros no constantes.

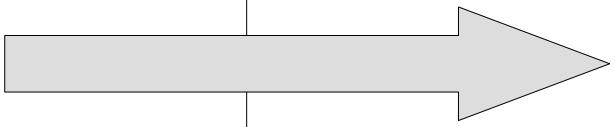
• Son útiles cuando una clase debe utilizar objetos diferentes que tienen un comportamiento similar.




```
public class Chico
{
    public String nombre;

}
```

```
public interface Mayor18 {
  public void comprarBebidaConAlcohol();
}
```

```
public class Boliche {

    public Mayor18 persona;

    public void ingresar(Mayor18 p)
    {
        this.persona=p;
    }

   public void comenzarPromoBebidas()
   {
     persona.comprarBebidaConAlcohol();
   }

}
```




```
Chico c = new Chico();

Boliche boliche = new Boliche();

boliche.ingresar(c);
```

La clase "Chico" no implementa la interfaz.

El objeto "Chico" no puede ser tratado como del tipo de dato "Mayor18".

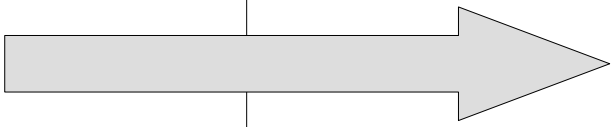
```
public void ingresar(Mayor18 p)
```

```
public interface Mayor18 {
  public void comprarBebidaConAlcohol();
}
```


```
public class Persona implements Mayor18
{
    public String nombre;

    @Override
    public void comprarBebidaConAlcohol()
    {
        System.out.println("Dame una
                            cerveza");
    }
}
```

```
public interface Mayor18 {
  public void comprarBebidaConAlcohol();
}
```

```
public class Boliche {

    public Mayor18 persona;

    public void ingresar(Mayor18 p)
    {
        this.persona=p;
    }

  public void comenzarPromoBebidas()
  {
    persona.comprarBebidaConAlcohol();
  }

}
```




```
Persona per = new Persona();

Boliche boliche = new Boliche();

boliche.ingresar(per);
```

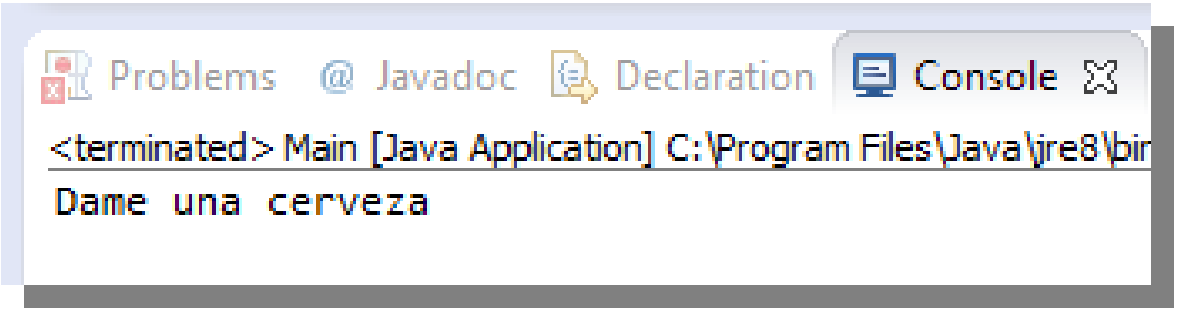
```
public void ingresar(Mayor18 p)
```


```
Persona per = new Persona();

Boliche boliche = new Boliche();

boliche.ingresar(per);
```

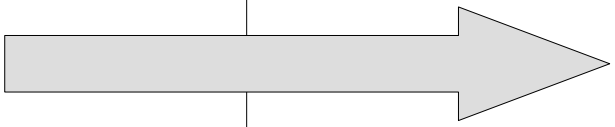
```
public void ingresar(Mayor18 p)
```

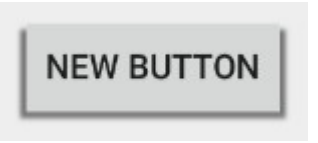
```
boliche.comenzarPromoBebidas();
```

```
@Override
public void comprarBebidaConAlcohol()
{
  System.out.println("Dame una
                      cerveza");
}
```



Intefaces

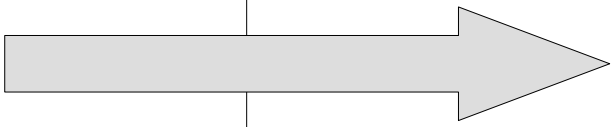


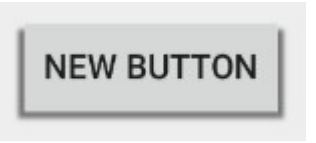



```
public class Oreja implements Escuchador
{
    public String nombre;

    @Override
    public void eventoClick()
    {
        System.out.println("Se hizo
                            Click!");
    }
}
```

```
public interface Escuchador {
  public void eventoClick();
}
```

```
public class Boton {

    public Escuchador escuchador;

    public void setEsc(Escuchador e)
    {
        this.escuchador=e;
    }

  public void eventoTouchEnPantalla()
  {
     escuchador.eventoClick();
  }

}
```



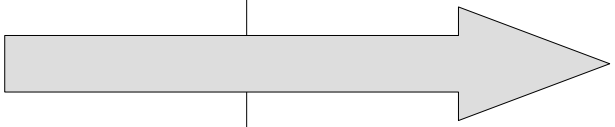


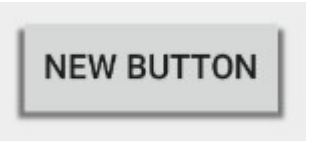

```
Oreja o = new Oreja();

Boton boton = new Boton();

boton.setEsc(o);
```

```
public void setEsc(Escuchador e)
```

```
public interface Escuchador {
  public void eventoClick();
}
```

Intefaces

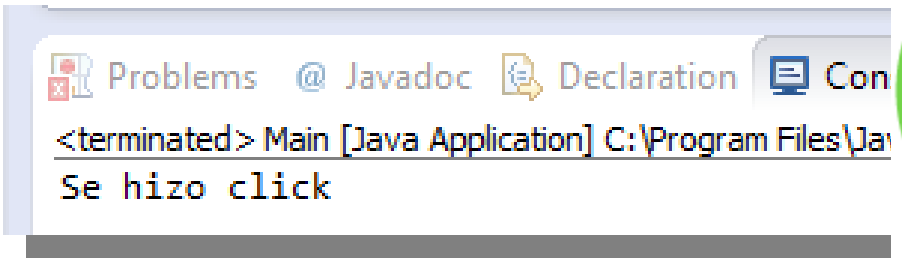





```
Oreja o = new Oreja();

Boton boton = new Boton();

boton.setEsc(o);
```

```
public void setEsc(Escuchador e)
```

```
boton.eventoTouchEnPantalla();
```



```
@Override
public void eventoClick()
{
    System.out.println("Se hizo
                        Click!");
}
```