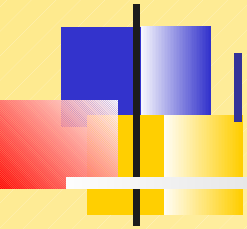
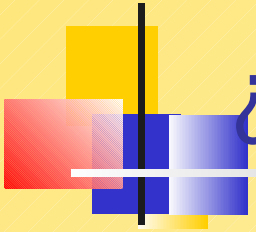
UNIVERSIDAD TECNICA
FEDERICO SANTA MARIA



Interfaces y Clases Internas

ELO-329: Diseño y programación orientados a objetos

Agustín J. González



¿Por qué existen?

- NOTA: El término interfaz aquí NO se refiere a las interfaces gráficas.
- Aquí las interfaces son una manera de describir qué debería hacer una clase sin especificar el cómo.
- Las clases internas son clases anidadas dentro de otra clase.
- Interfaces y clases internas son recursos esenciales en el manejo de *interfaces gráficas* en Java.

Interfaces

- Una interfaz es la descripción de algún servicio que posteriormente alguna clase puede implementar.
- Por ejemplo, si un alumno sabe alemán, tenemos idea de lo que él es capaz. Además de ser persona (herencia) él cumple la interfaz “interprete de alemán”. También podríamos decir que él es un “interprete de alemán” (nuevamente la relación de herencia).
- En Java cada clase puede tener sólo una clase base (herencia no múltiple), por ello Interfaces son usadas para abordar casos como el ejemplo previo.
- Instancias de la clase que implementa una Interfaz pueden ser usadas donde se espera una instancia de la interfaz. Es similar a usar una instancias de una subclase cuando se espera un objeto de la clase base.

Interfaces (cont.)

- No se permite crear objetos instancias de un Interfaz. Por la misma razón que no se puede crear instancias de clases abstractas. New InterfazX();

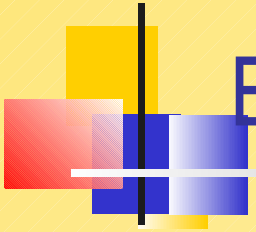
- Todos los métodos de una Interfaz son públicos. No es necesario indicarlo.
- Pueden incluir constantes. En este caso son, por omisión, public static y final.
- Aspectos Sintácticos:

Definición de una interfaz:

```
public interface Comparable
{
    int compareTo (Object other);
}
```

Implementación de una interfaz:

```
class Employee implements Comparable
{
....
public int compareTo(Object other)
{
 ....// implementación
}
}
```



Ejemplos

- Consideremos la extensión de la clase Employee para que podamos ordenar arreglos de empleados según su salario.
- Ver EmployeeSortTest.java
- Ver documentación de clase Arrays e interfaz Comparable

Clases Internas

- Son clases definidas dentro de otras.
  - Se usan para dar tener acceso a miembros de la clase anfitriona – incluso si son privados.
  - Se usan como mecanismo de encapsulamiento.
  - Son muy útiles para crear programas conducidos por eventos. (event-driven programming)
- Las clases internas existen sólo para el compilador, ya que éste las transforma en clases regulares separando la clase externa e interna con signo $.
- La máquina virtual no distingue la clases internas.
- Ver InnerClassTest.java
- También se pueden definir al interior de un método.

Clases internas

```
public void start(double rate)
 {
    private class InterestAdder implements ActionListener
    {
            public InterestAdder(double aRate)
            {           rate = aRate;    }
            public void actionPerformed(ActionEvent event)
            {
            double interest = balance * rate / 100; // update interest
                        balance += interest;
                        NumberFormat formatter = NumberFormat.getCurrencyInstance();
                        System.out.println("balance=" + formatter.format(balance));
            }
             private double rate;
    }
   ActionListener adder = new InterestAdder(rate);
   Timer t = new Timer(1000, adder);
   t.start();
}
```

Clases internas anónimas

Cuando necesitamos sólo una instancia de la clase, no necesitamos darle un nombre. Decimos que tal clase es interna y anónima. Ver AnonymousInnerClassTest

```
public void start(final double rate)
  {
    ActionListener adder = new   //sigue en otra línea..
      ActionListener()
      {
        public void actionPerformed(ActionEvent event)
        {
                    double interest = balance * rate / 100;
          balance += interest;
                    NumberFormat formatter = NumberFormat.getCurrencyInstance();
          System.out.println("balance=" + formatter.format(balance));
        }
      };

    Timer t = new Timer(1000, adder);
    t.start();
  }
```