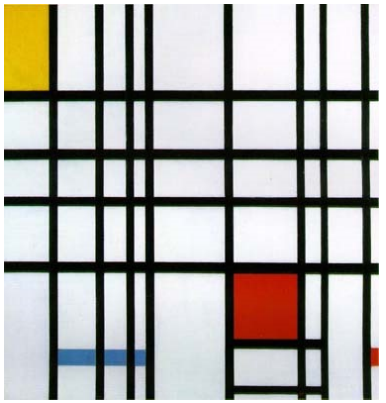
Clases abstractas, interfaces y herencia múltiple

Clases abstractas e interfaces

- Se relacionan con la herencia
  - Definen qué puede hacer un conjunto de clases relacionadas
- Definiciones
  - Clase abstracta: Contiene al menos un método abstracto
  - Interfaz: Todos los métodos son abstractos
  - Método abstracto: Declara una funcionalidad que debe ser implementada en todas las clases derivadas



Arte Abstracto, P. Mondrian, 1921
Composición en rojo, amarillo y azul

Propiedades clases abstractas

- Se declara con la palabra reservada `abstract` como prefijo de la cabecera de clase

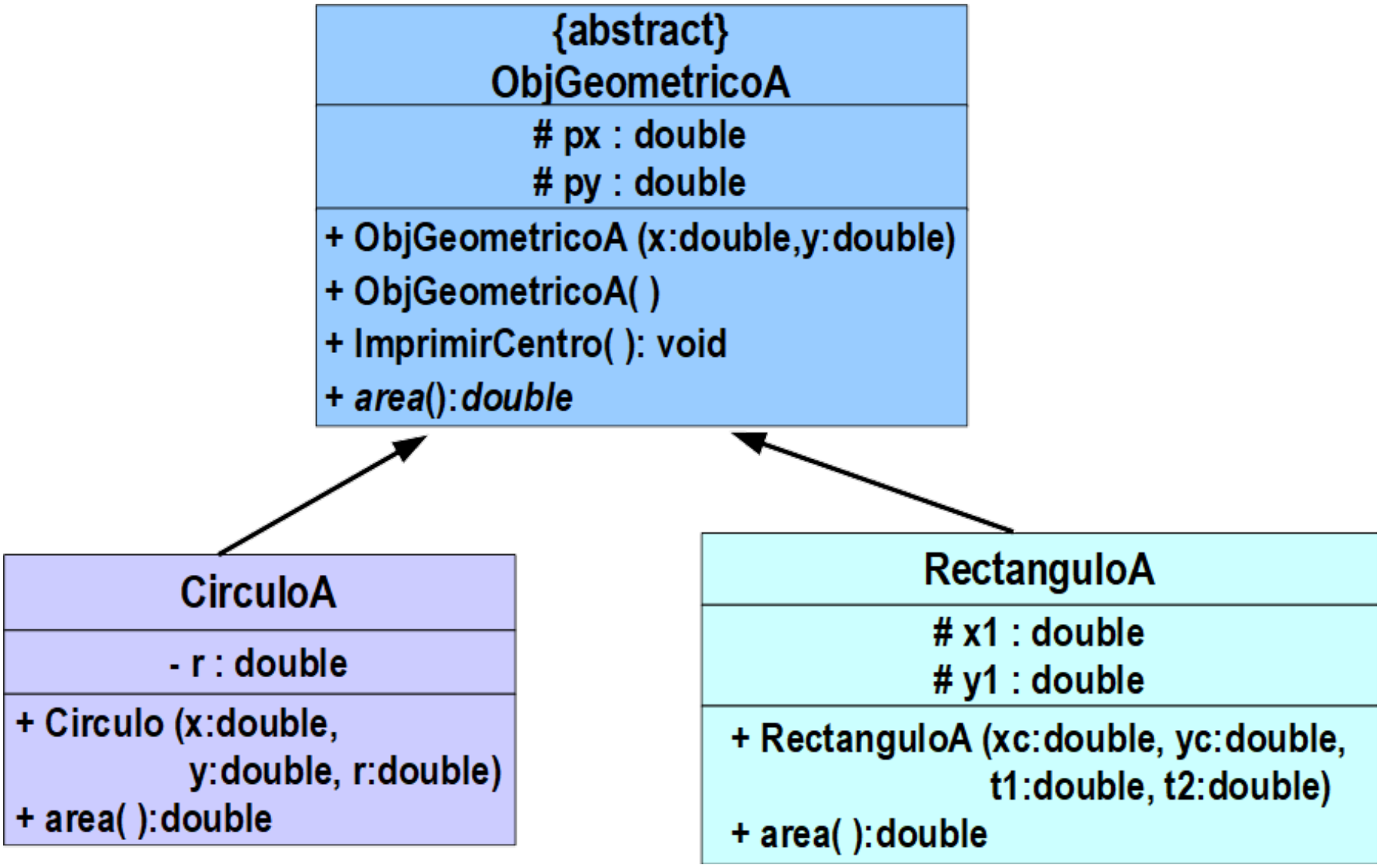
```
abstract public class ObjGeometricoA
```

- Pueden tener variables de instancia y métodos no abstractos
- Si una clase derivada no implementa un método abstracto es también clase abstracta
- No se pueden crear objetos de clases abstractas.

```
abstract public class AutoCompacto
public class Volks extends AutoCompacto

AutoCompacto vocho = new AutoCompacto();
Volks vocho = new Volks();
AutoCompacto vocho = new Volks()
```

Ejemplo clase abstracta



Implementación

```
1   package interfaces;
2
3   abstract public class ObjGeometricoA {
4   protected double px, py;
5       public ObjGeometricoA(double x, double y){px=x; py=y;}
6       public ObjGeometricoA(){ px=py=0;   }
7       public void imprimirCentro() {
8           System.out.println("("+px+","+py+")");
9       }
10      public abstract double area();
11  }
```

```
1   package interfaces;
2
3   public class CirculoA extends ObjGeometricoA{
4       private double r ;
5       public CirculoA(double  x, double y, double r) {
6           super(x,y);
7           this.r=r;
8       }
9       public double area() { return Math.PI * r * r;}
10  }
```

Uso clase abstracta

```
1   package interfaces;
2
3   public class RectanguloA extends ObjGeometricoA{
4   private double base, altura;
5      public RectanguloA(double xc, double yc, double t1, double t2)
6      {
7         super(xc,yc);
8         base=t1;
9         altura=t2;
10     }
       public double area() {  return base*altura;  }
12  }
```

```
ObjGeometricoA [] figuras = new ObjGeometricoA[10];
for( int i=0; i<10; i++)
        if( i%2 == 0)figuras[i]= new CirculoA(0,i,i+2);
   else figuras[i]= new RectanguloA(i,0,3,3);
```

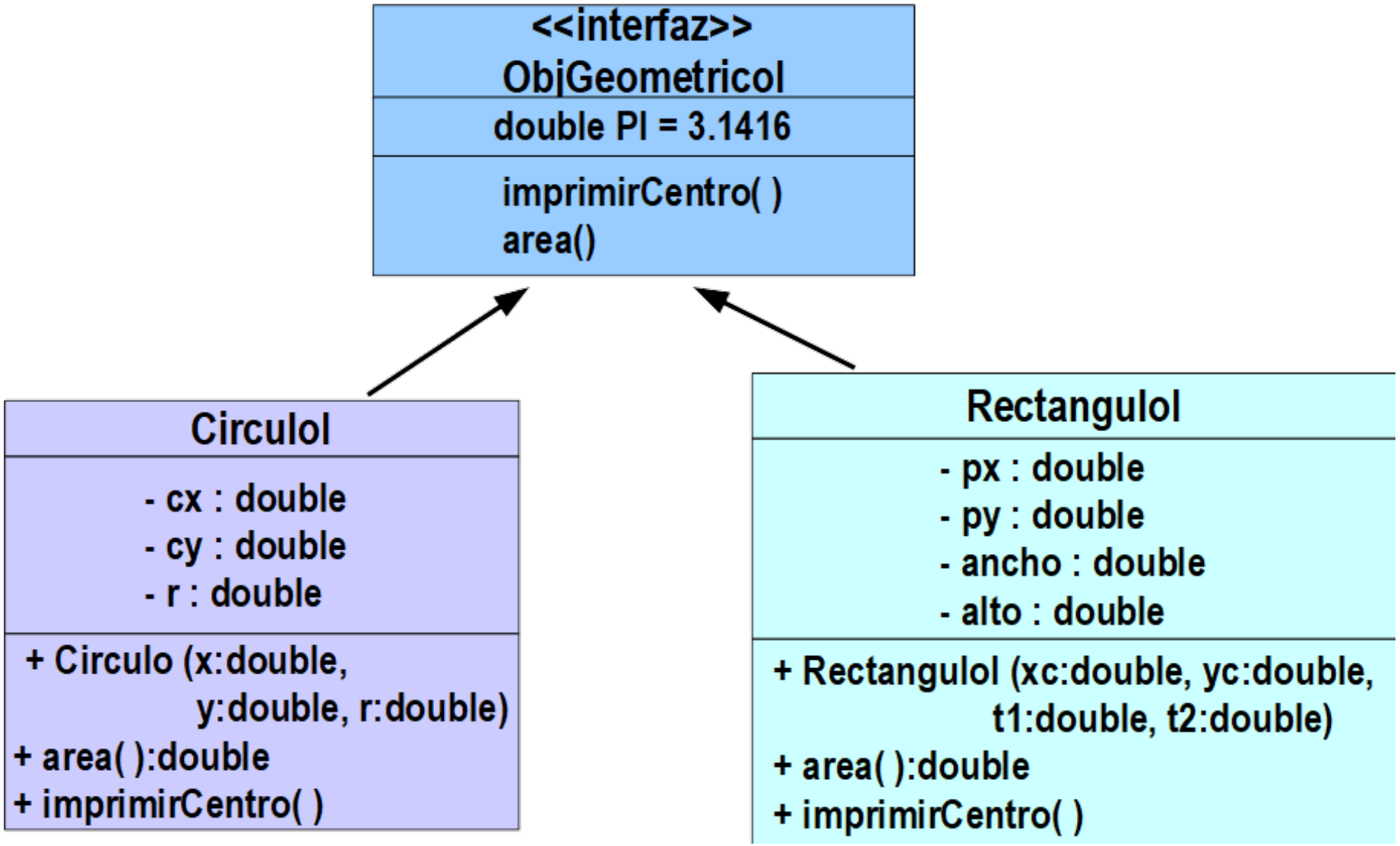
Interfaz

- Es un elemento java que dice lo que se ofrece, pero no dice cómo se hace
- Se usa la palabra reservada `interface` en la clase base e `implements` en las clases derivadas

```
interface ObjGeometricoI
class CirculoI implements ObjGeometricoI
```

- Características
  - No incluye constructores
  - No se pueden instanciar objetos
  - Valores constantes “public static final” y métodos “public”
    - No hace falta escribir el modificador

Ejemplo Interfaz



Implementación interfaces

```
package interfaces;
interface ObjGeometricoI {
    double PI = 3.1416;
    void imprimirCentro();
    double area();
}
```

```
package interfaces;
class CirculoI implements ObjGeometricoI{
    private double r ;
    private double px,py;
    public CirculoI(double  x, double y, double r) {
                px=x;
                py=y;
                this.r=r;
    }
    public double area() { return PI * r * r;}
    public void imprimirCentro() {
       System.out.println("("+px+","+py+")");
   }
}
```

Ejemplo: uso de interfaces

```
import interfaces.*;
public class PruebaFigurasI {
  public static void main(String[] args)    {
    CirculoI cr = new CirculoI(2.0,2.5,2.0);
    RectanguloI cd = new RectanguloI(3.0,3.5,4.25,3.25);
    System.out.print("Centro del circulo:");
    cr.imprimirCentro();
    System.out.print("Centro del rectangulo:");
    cd.imprimirCentro();
    System.out.println("Area del circulo:"+cr.area());
    System.out.println("Area del rectangulo:"+cd.area());
    }
}
```

```
run:
Centro del circulo:(2.0,2.5)
Centro del rectangulo:(3.0,3.5)
Area del circulo:12.5664
Area del rectangulo:13.8125
BUILD SUCCESSFUL (total time: 0 seconds)
```

¿Cuándo usar interfaces?

- Cuando sabemos qué queremos; pero
  - no sabemos (aún) cómo hacerlo
  - lo haremos de varias maneras
  - lo hará otro

- Ejemplos
  - Elementos de interacción como botones, clics de ratón, etc.
- Cuando una clase hereda de varias clases base
  - En java no hay herencia múltiple
    - La interfaz nos permite simularla
    - Permite que una clase implemente varias interfaces

Ejemplo herencia múltiple

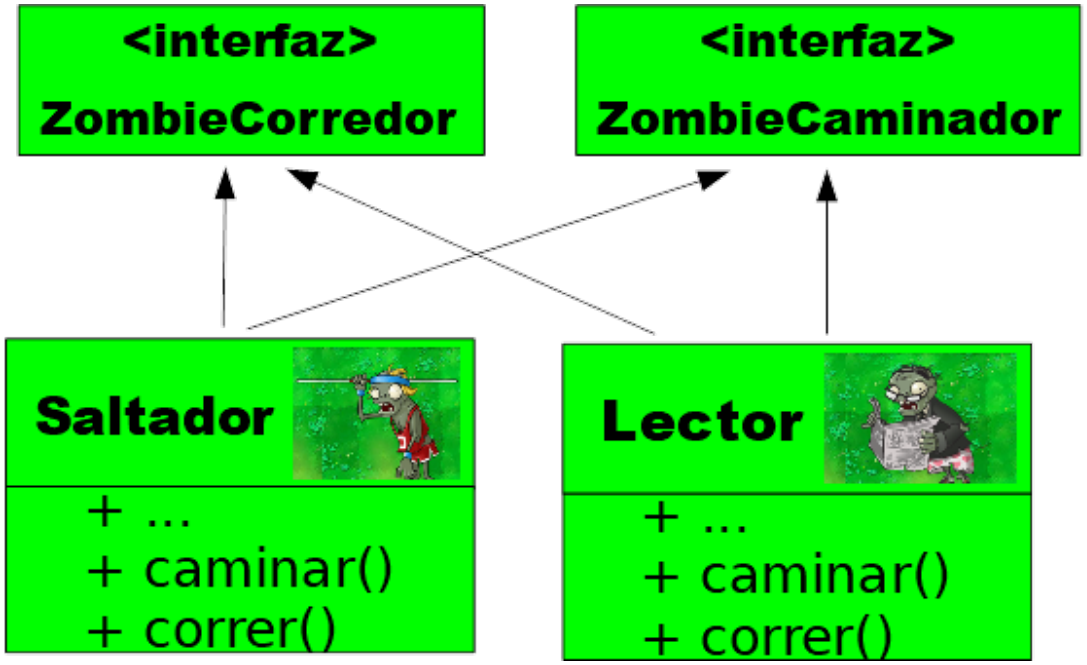
```
Interface Trazo{
    double perimetro( );
}
```

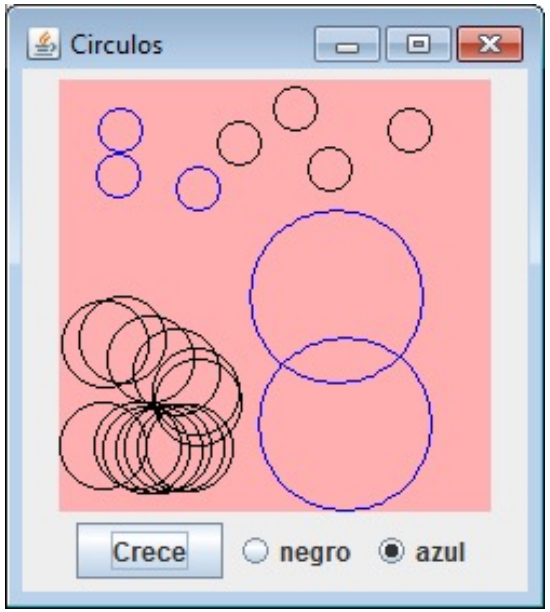
```
Interface Superficie{
    double area( );
}
```

```
class Rectangulo implements Trazo, Superficie {
    private double base, altura;

    Rectangulo(double base, double altura) {
        this.base = base
        this.altura = altura;
    }
    public double perimetro() {
            return 2 * (base + altura);
    }
    public double area() {
        return base * altura;
    }
}
```

Otros ejemplos de herencia múltiple






```
public class Circulos extends JFrame
                      implements ItemListener,ActionListener,MouseListener{
```