Programación Orientada a Objetos

Introducción

CEIS

2019-01

Agenda

Iniciando
- Las tres P
- Investigación

POOB-Curso
- Descripción
- Prácticas, lenguajes y herramientas

Orientación a objetos
- Materia prima
- Clases y objetos
- Atributos
- Métodos
- Casa

Herramienta. BlueJ
- General
- Editar
- Compilar
- Ejecutar
- Documentar

Agenda

Iniciando
- Las tres P
- Investigación

POOB-Curso
- Descripción
- Prácticas, lenguajes y herramientas

Orientación a objetos
- Materia prima
- Clases y objetos
- Atributos
- Métodos
- Casa

Herramienta. BlueJ
- General
- Editar
- Compilar
- Ejecutar
- Documentar

Programa

Componentes

Calidad

Programa

Componentes

- Ejecutable
- Fuentes
- Manual de usuario
- Manual técnico

Calidad

- Corrección
- Extensibilidad
- Facilidad de Uso
- Eficiencia
- Portabilidad

Proceso

Etapas

Calidad

Proceso

Etapas

1. Requisitos
2. Análisis
3. Diseño
4. Construcción
5. Pruebas

Calidad

- Cronograma
- Alcance
- Presupuesto

Proceso

Etapas

1. Requisitos

   ¿ Qué necesita el cliente?

2. Análisis

   ¿ Qué vamos a hacer?

3. Diseño

   ¿ Cómo lo vamos a hacer?

4. Construcción

   ¡ Hacerlo!

5. Pruebas

   ¿ Lo hicimos bien?

Calidad

- Cronograma
- Alcance
- Presupuesto

Personas

Técnicos

- Requisitos
- Análisis
- Diseño
- Construcción
- Pruebas

Personas

Técnicos

- Requisitos
- Análisis

Analista

- Diseño

Diseñador

¡ ARQUITECTO !

- Construcción
- Pruebas de unidad

Programador

¡ EQUIPO DE CALIDAD !

Investigación

Lo ágil

The Rules and Practices of Extreme Programming. *Lessons Learned*

Planning

- User stories are written.
- Release planning creates the schedule.
- Make frequent small releases.
- The Project Velocity is measured.
- The project is divided into iterations.
- Iteration planning starts each iteration.
- Move people around.
- A stand-up meeting starts each day.
- Fix XP when it breaks.

Designing

- Simplicity.
- Choose a system metaphor.
- Use CRC cards for design sessions.
- Create spike solutions to reduce risk.
- No functionality is added early.
- Refactor whenever and wherever possible.

Coding

- The customer is always available.
- Code must be written to agreed standards.
- Code the unit test first.
- All production code is pair programmed.
- Only one pair integrates code at a time.
- Integrate often.
- Use collective code ownership.
- Leave optimization till last.
- No overtime.

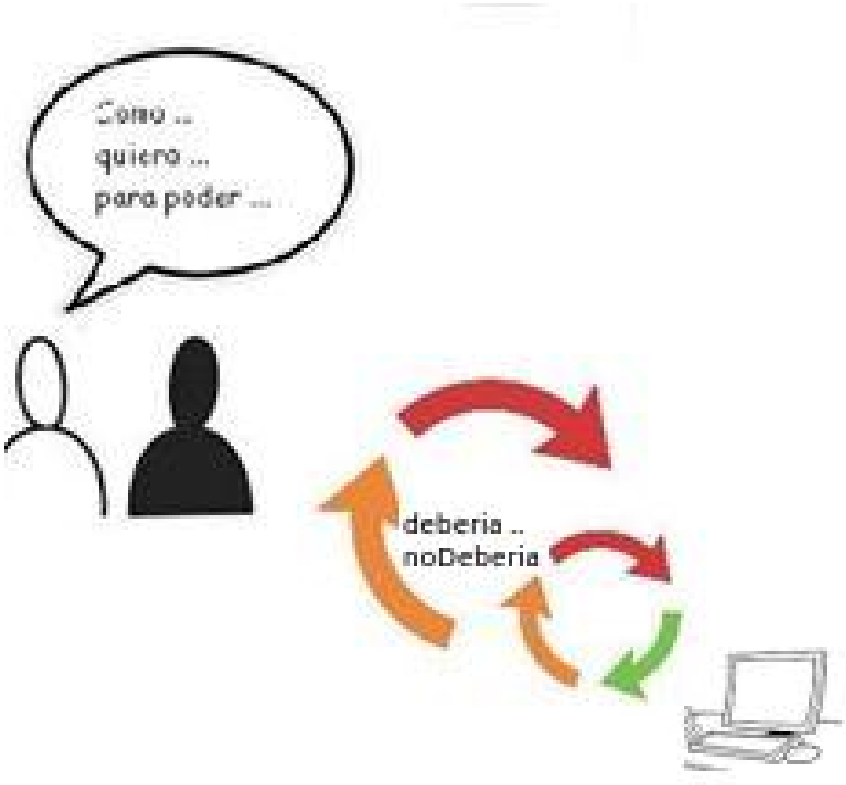
Testing

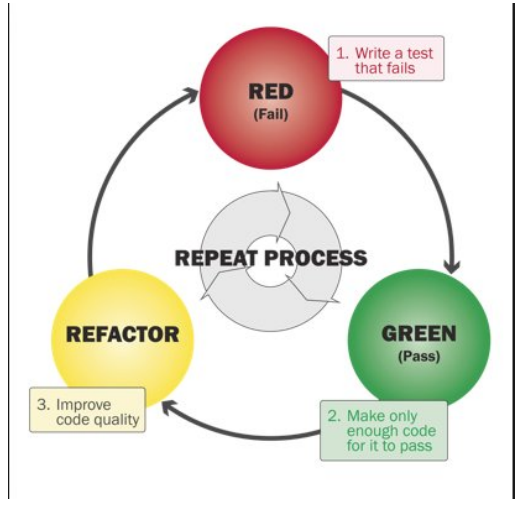
- All code must have unit tests.
- All code must pass all unit tests before it can be released.
- When a bug is found tests are created.
- Acceptance tests are run often and the score is published.

ExtremeProgramming.org home | XP Map | Email the webmaster

Investigación

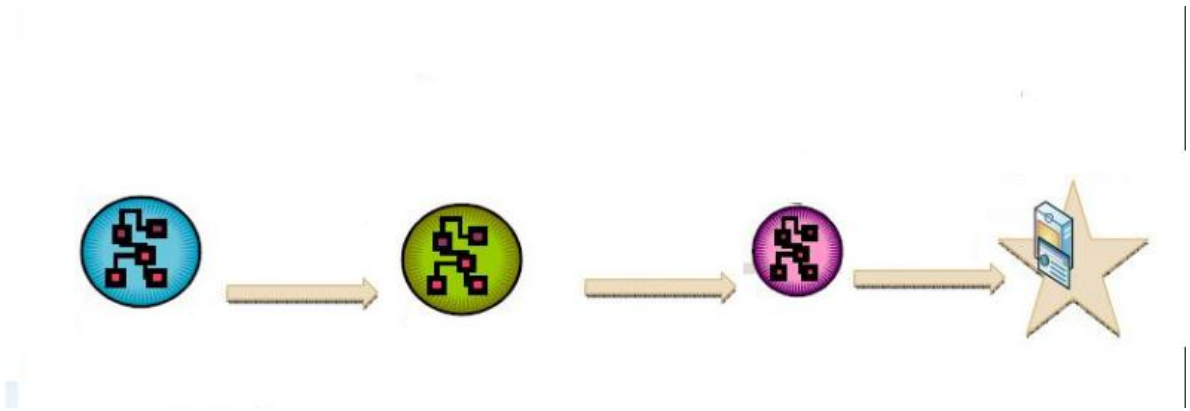
BDD





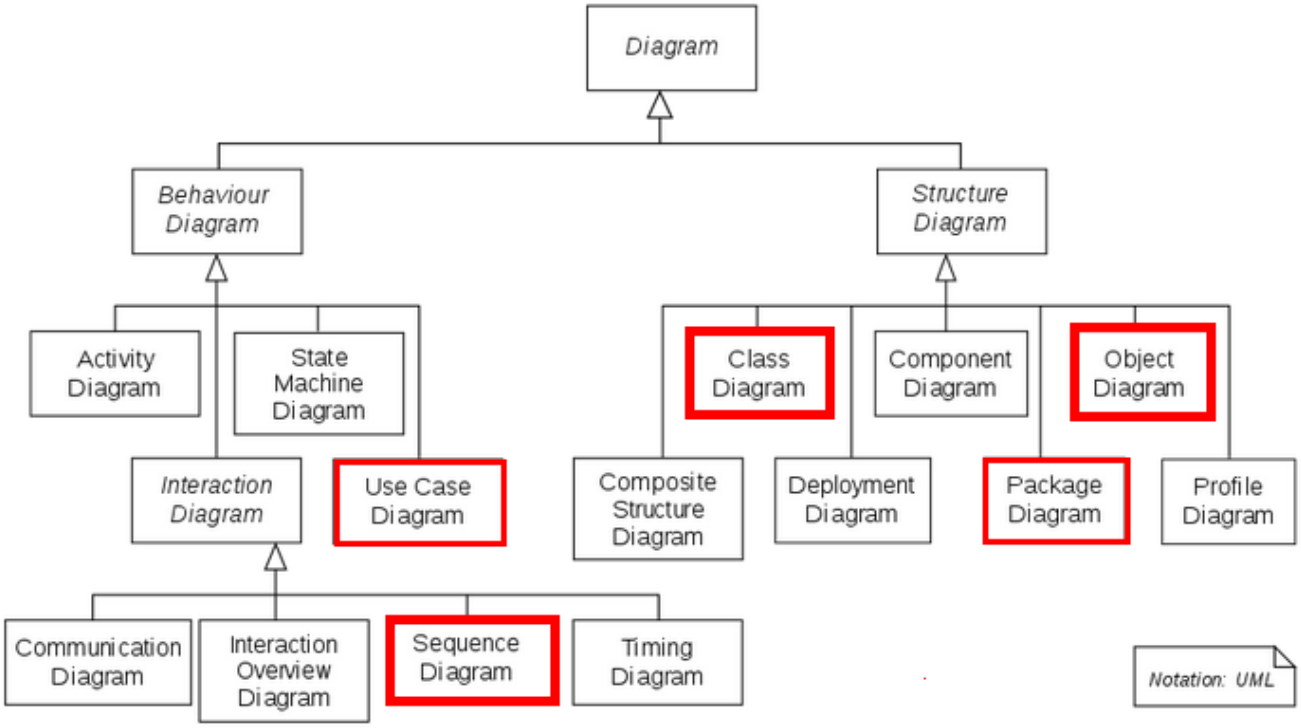
Proceso

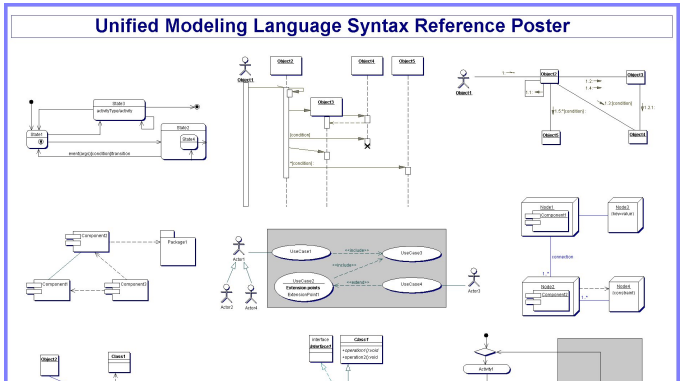
MDD



Investigación

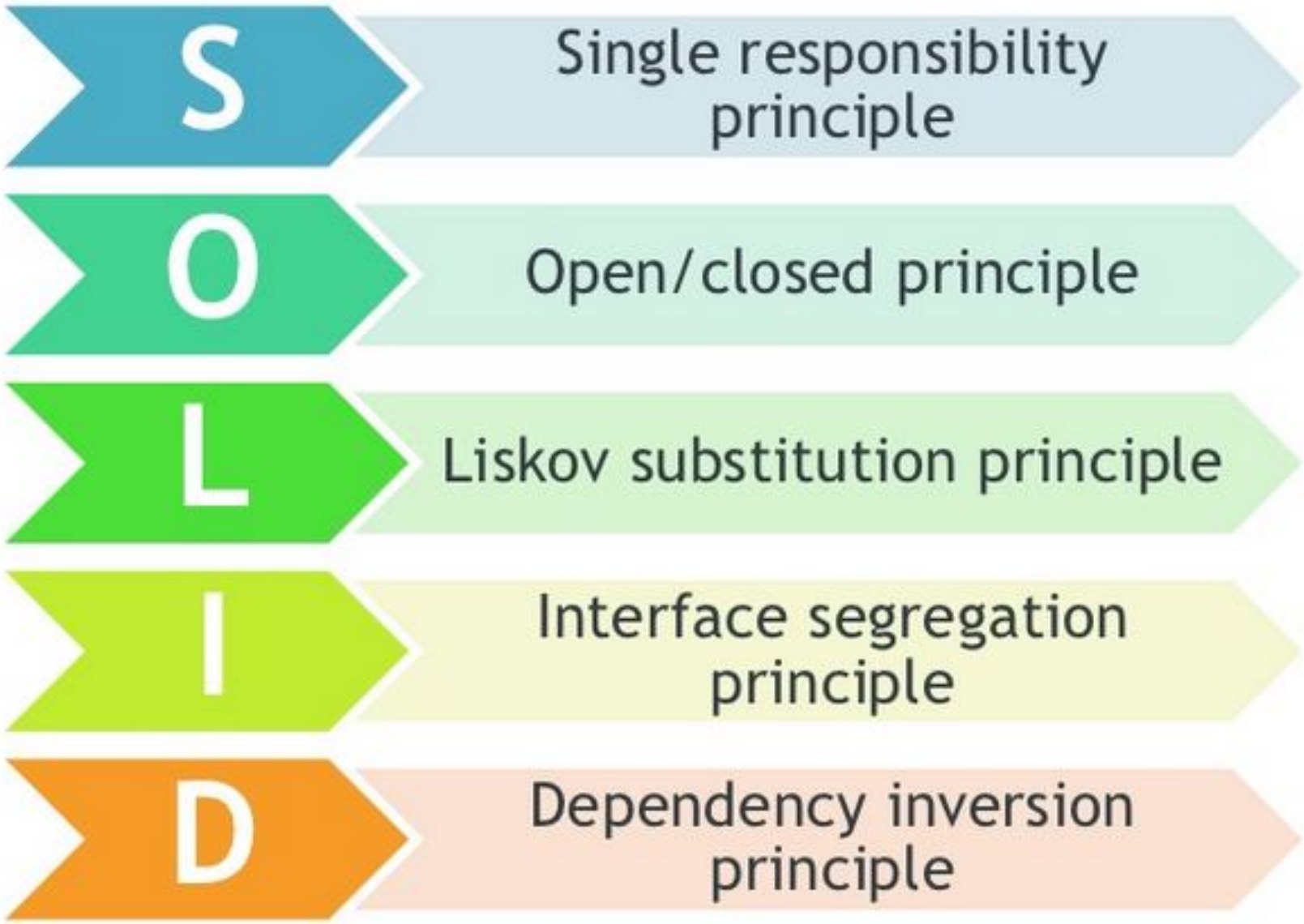
UML





Investigación

SOLID



Investigación

Java

Agenda

Iniciando
- Las tres P
- Investigación

POOB-Curso
- Descripción
- Prácticas, lenguajes y herramientas

Orientación a objetos
- Materia prima
- Clases y objetos
- Atributos
- Métodos
- Casa

Herramienta. BlueJ
- General
- Editar
- Compilar
- Ejecutar
- Documentar

Objetivo

¿Cuál es el propósito?

Construir un producto software o mejorar uno existente.

¡Lograr que el software funcione y evolucione !

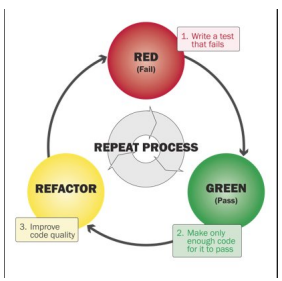
Objetivo

¿Cuál es el propósito?

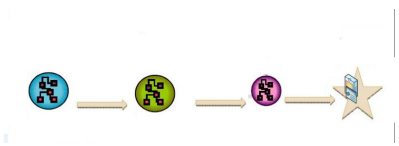
Construir un producto software o mejorar uno existente.

¡Lograr que el software funcione y evolucione !

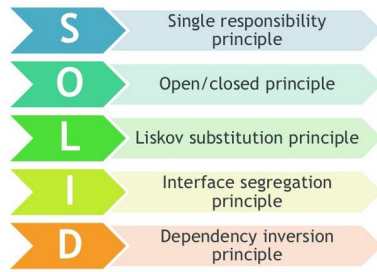
BDD



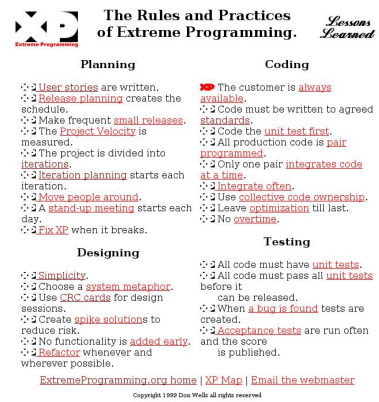
MDD



SOLID



XP



POOB

Metodología

- Clase

  Teoría

  Trabajo en clase

- Laboratorio

  Semanas pares Viernes a.m.

  Sustentación con el monitor

  [Entrega final Mc]

- Proyecto

  Inicial. [1ero y 2do tercio] Cuatro ciclos dos por tercio

  Final. [3er tercio] Estructura + dos ciclos

  [Entrega Ju]

POOB

Evaluación

- 50% Examen parcial

  T1 [Semana 6 (Vi)], T2 [Semana 11 (Vi)] T3 [Proyecto]

- 15% Quices y trabajos en clase

- 15% Laboratorio

  Maratón *HackerRank Java* BONO 3er tercio

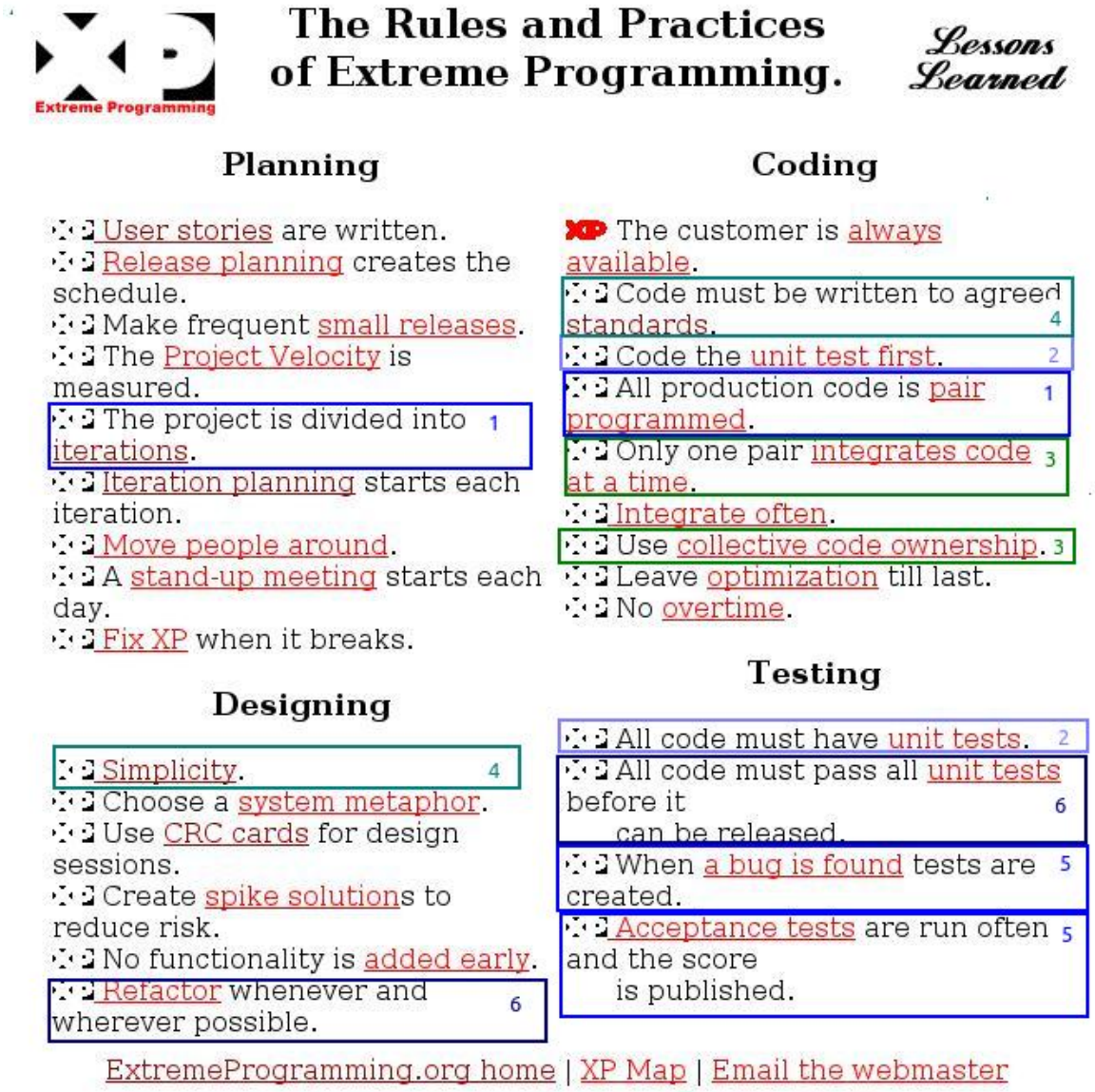
  Inicio: semana 1 Cierre: semana 16

  Evaluación de verificación

- 20% Proyecto

Prácticas

XP



The Rules and Practices of Extreme Programming.

Extreme Programming

Lessons Learned

Planning

- User stories are written.
- Release planning creates the schedule.
- Make frequent small releases.
- The Project Velocity is measured.
- The project is divided into iterations. 1
- Iteration planning starts each iteration.
- Move people around.
- A stand-up meeting starts each day.
- Fix XP when it breaks.

Designing

- Simplicity. 4
- Choose a system metaphor.
- Use CRC cards for design sessions.
- Create spike solutions to reduce risk.
- No functionality is added early.
- Refactor whenever and wherever possible. 6

Coding

- The customer is always available.
- Code must be written to agreed standards. 4
- Code the unit test first. 2
- All production code is pair programmed. 1
- Only one pair integrates code at a time. 3
- Integrate often.
- Use collective code ownership. 3
- Leave optimization till last.
- No overtime.

Testing

- All code must have unit tests. 2
- All code must pass all unit tests before it can be released. 6
- When a bug is found tests are created. 5
- Acceptance tests are run often and the score is published. 5

ExtremeProgramming.org home | XP Map | Email the webmaster

Lenguajes

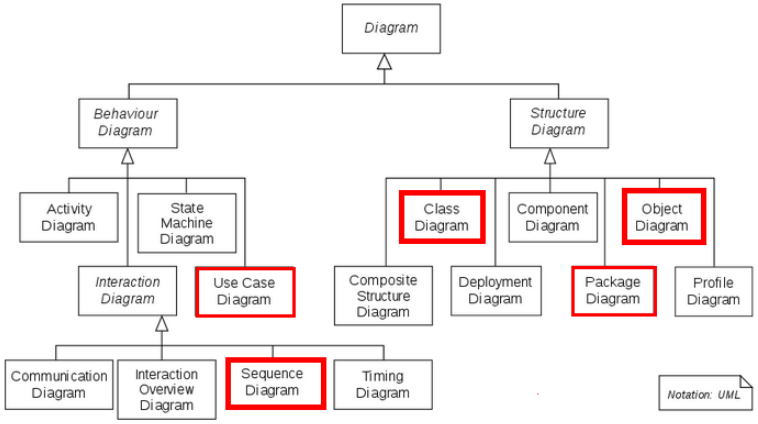


Diagram
Behaviour Diagram
Structure Diagram
Activity Diagram
State Machine Diagram
Interaction Diagram
Use Case Diagram
Class Diagram
Component Diagram
Object Diagram
Composite Structure Diagram
Deployment Diagram
Package Diagram
Profile Diagram
Communication Diagram
Interaction Overview Diagram
Sequence Diagram
Timing Diagram
Notation: UML


Java

Herramientas

Herramientas

- JDK Conjunto de herramientas de desarrollo
- JUnit Herramienta de pruebas unitarias

- **BlueJ Ambiente de desarrollo**
- ECLIPSE Ambiente de desarrollo

- ASTAH Herramienta de modelado

- Trello Administración de proyectos
- GitHub Plataforma para alojar proyectos

Herramientas

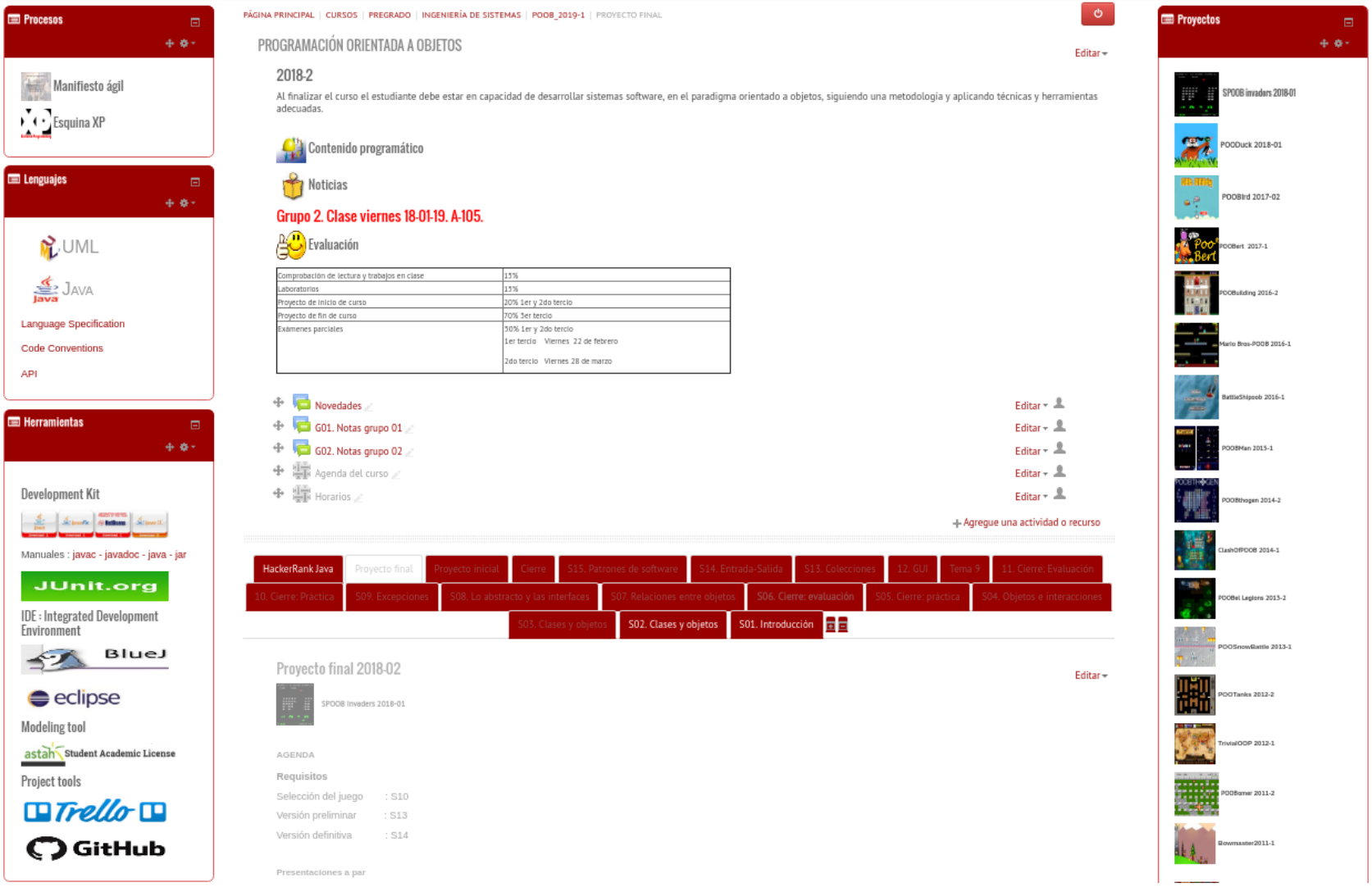
JDK

There have been similar discontinuities in the way in which the **Java Development Kit** (**JDK**) has been referred to. The JDK is the software "bundle" used by developers to build Java applications and consisting of

- The Java Virtual Machine (JVM)
- The Java compiler (`javac`)
- The Java Archive (`jar`) utility
- The Java documentation (`javadoc`) utility

Herramientas

Moodle



Agenda

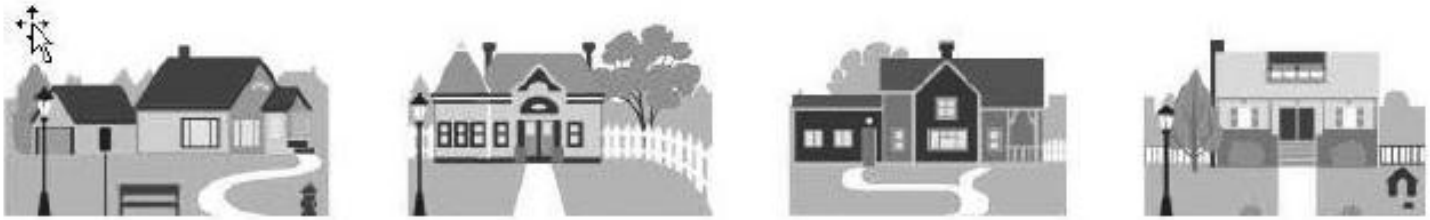
Iniciando
- Las tres P
- Investigación

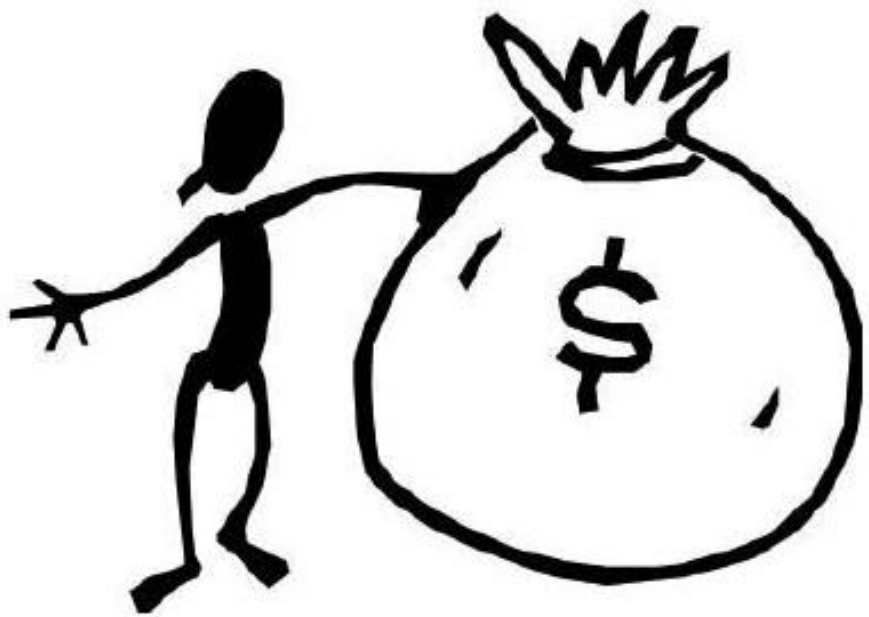
POOB-Curso
- Descripción
- Prácticas, lenguajes y herramientas

Orientación a objetos
- Materia prima
- Clases y objetos
- Atributos
- Métodos
- Casa

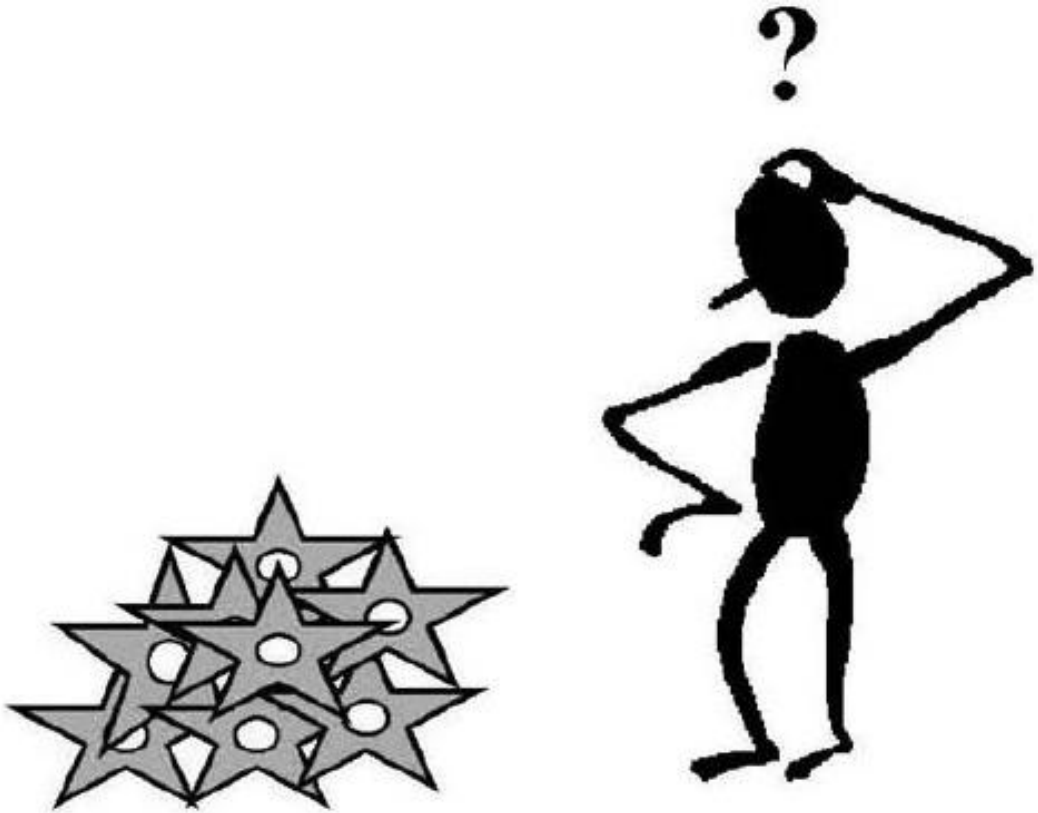
Herramienta. BlueJ
- General
- Editar
- Compilar
- Ejecutar
- Documentar

Historia





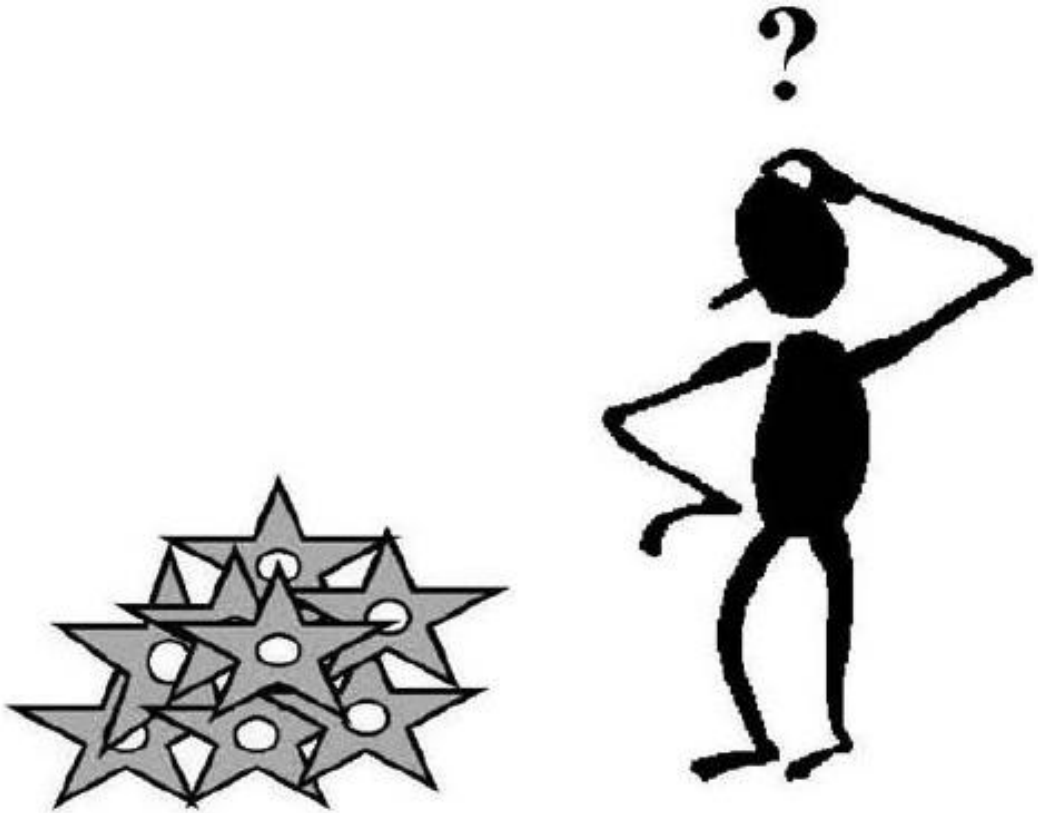
Historia



Historia

Historia

Clases y objetos



¿Qué son?

Objeto

Objeto (en el mundo real)

Objeto (en software)

Objeto

Objeto (en el mundo real)

Un **objeto** es algo mental o físico hacia el cual dirigimos nuestros sentimientos, pensamiento o acción.

Objeto (en software)

Objeto

Objeto (en el mundo real)

Un **objeto** es algo mental o físico hacia el cual dirigimos nuestros sentimientos, pensamiento o acción.

Objeto (en software)

Un **objeto** es un artefacto software que representa una abstracción de un objeto del mundo real por medio de su estado (datos) y comportamiento (funciones).

Clase

Clase (en el mundo real)

Clase (en software)

Clase

Clase (en el mundo real)

Una **clase** es una abstracción que describe todas las características comunes de todos los objetos de un grupo similar de objetos.

Clase (en software)

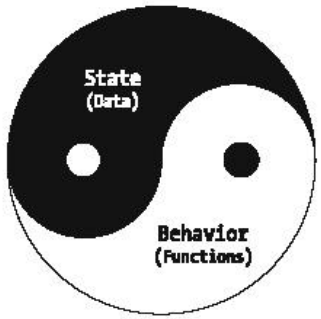
Clase

Clase (en el mundo real)

Una **clase** es una abstracción que describe todas las características comunes de todos los objetos de un grupo similar de objetos.

Clase (en software)

Una **clase** define las características - atributos y métodos - que cada objeto que pertenece a la clase posee. Puede ser visto como un molde.



- Atributos
- Métodos

Objetos

Objeto (en software)

Un **objeto** es un artefacto software que representa una abstracción de un objeto del mundo real por medio de su estado (datos) y comportamiento (funciones).

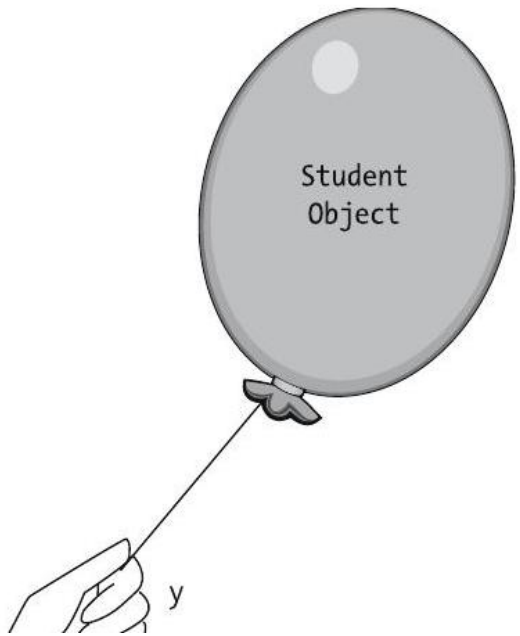
Objetos

Objeto (en software)

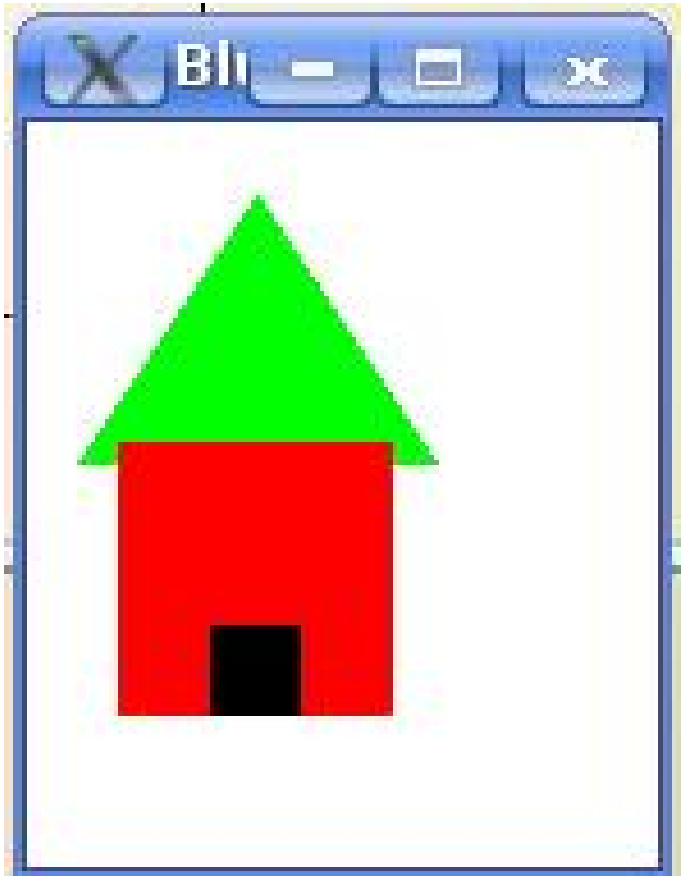
Un **objeto** es un artefacto software que representa una abstracción de un objeto del mundo real por medio de su estado (datos) y comportamiento (funciones).

Objeto

```
y = new Student();
```

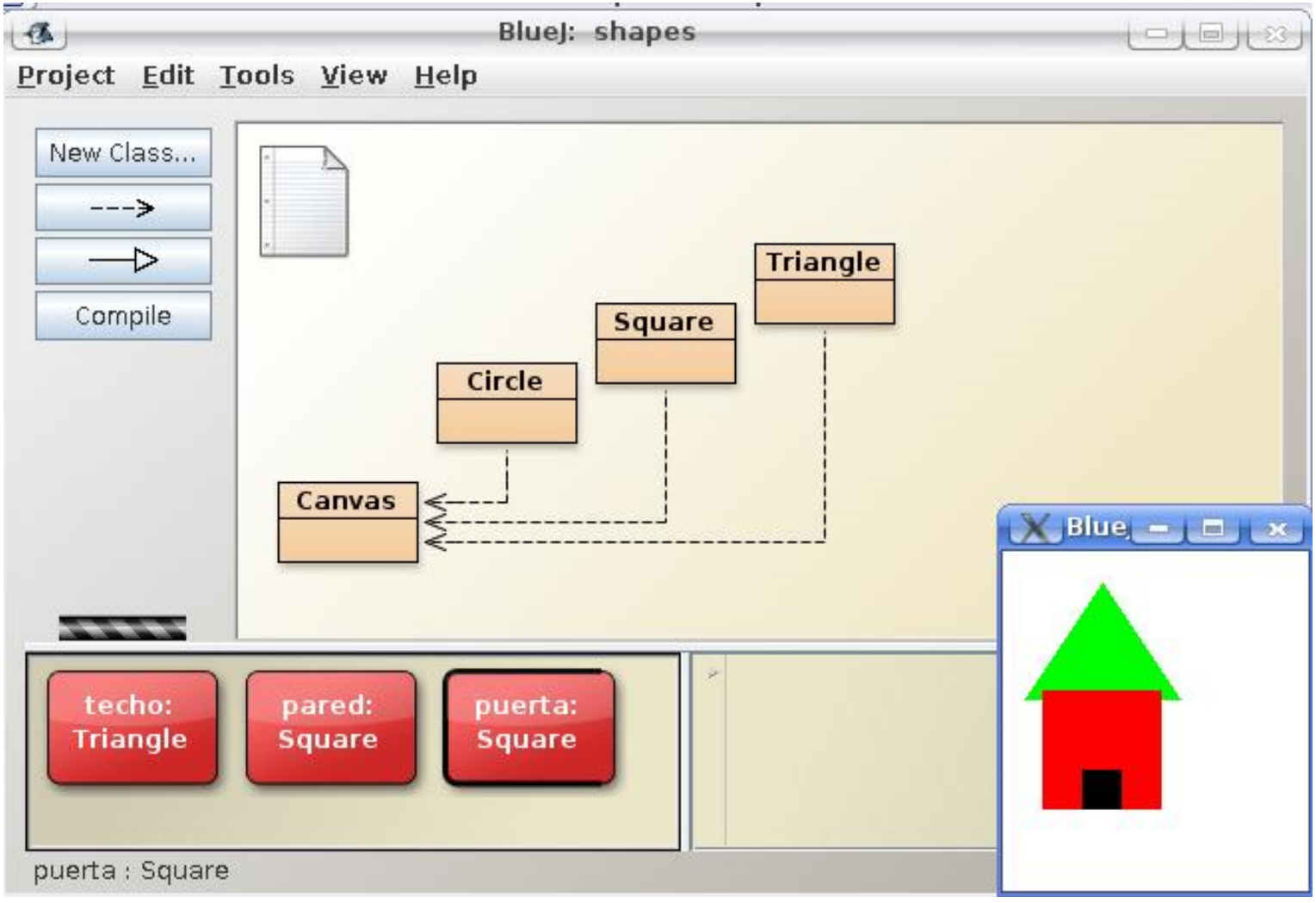


Clases y objetos



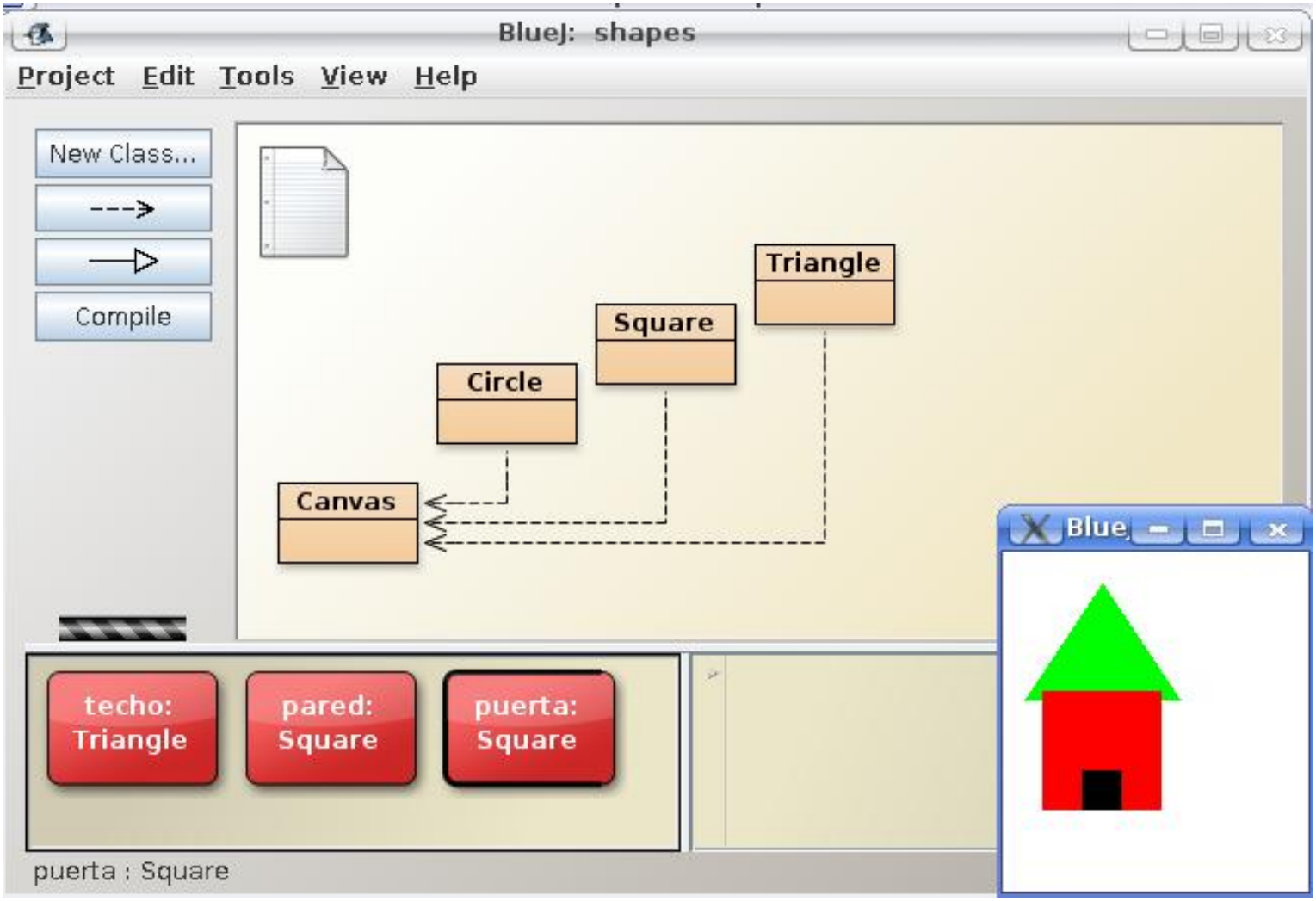
¿Clases?¿Objetos?

Clases y objetos



¿Clases?¿Objetos?

Clases y objetos



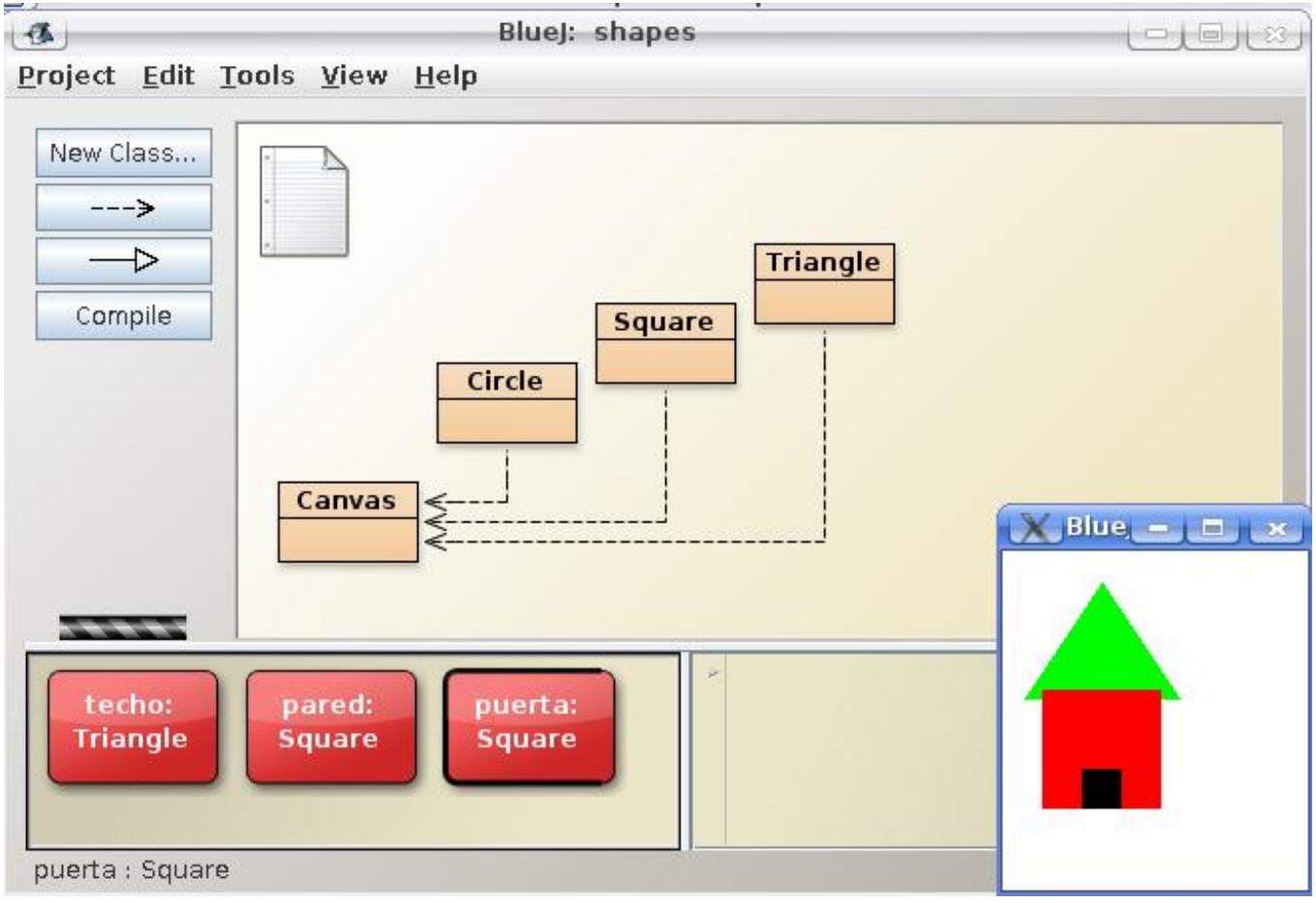
Clase cuadrado ¿Qué datos debe tener?

Clases y objetos

```
public class Square {
    private int size;
    private int xPosition;
    private int yPosition;
    private String color;
    private boolean isVisible;


    ...
}
```

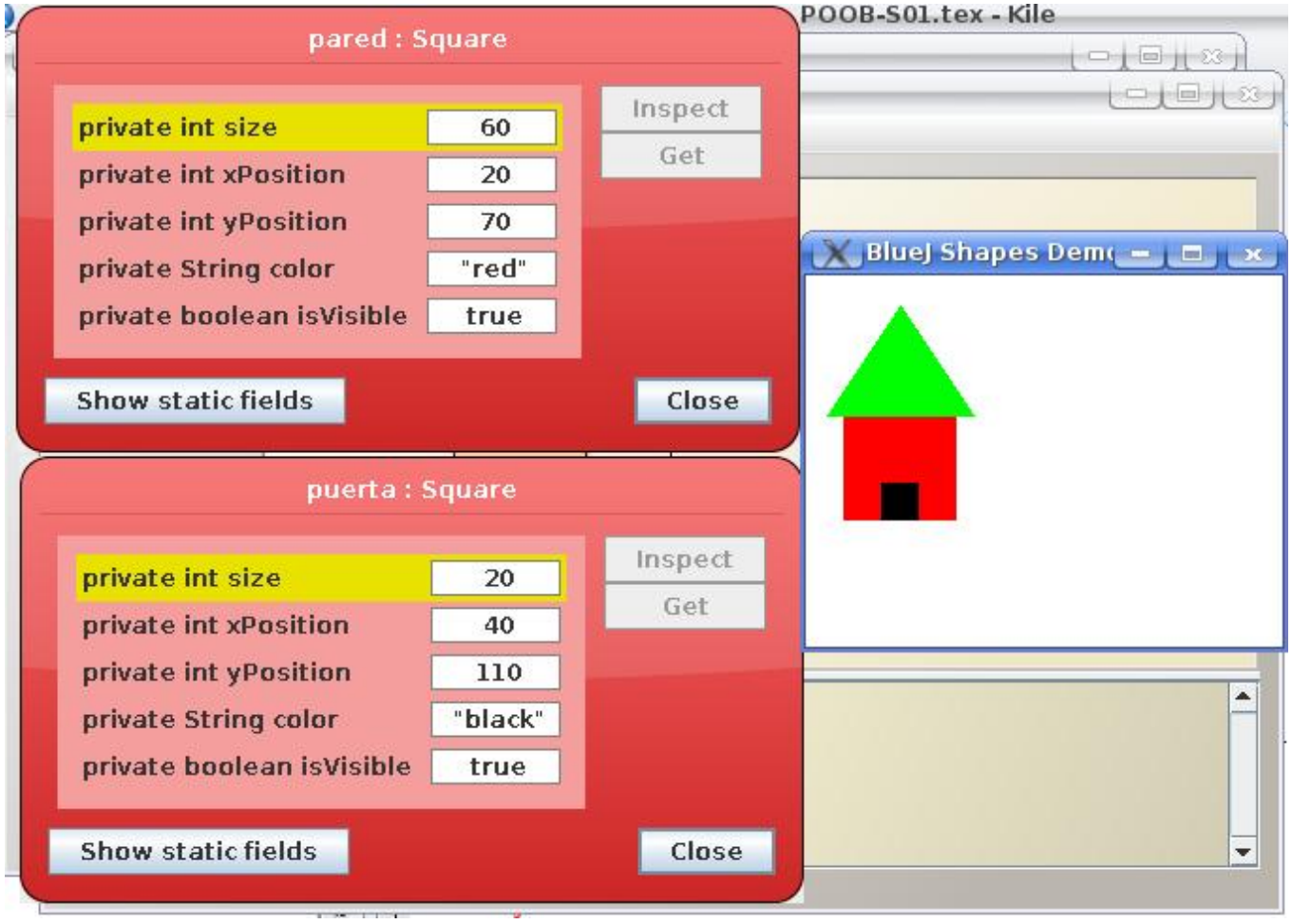
Clase cuadrado ¿Atributos?

Clases y objetos



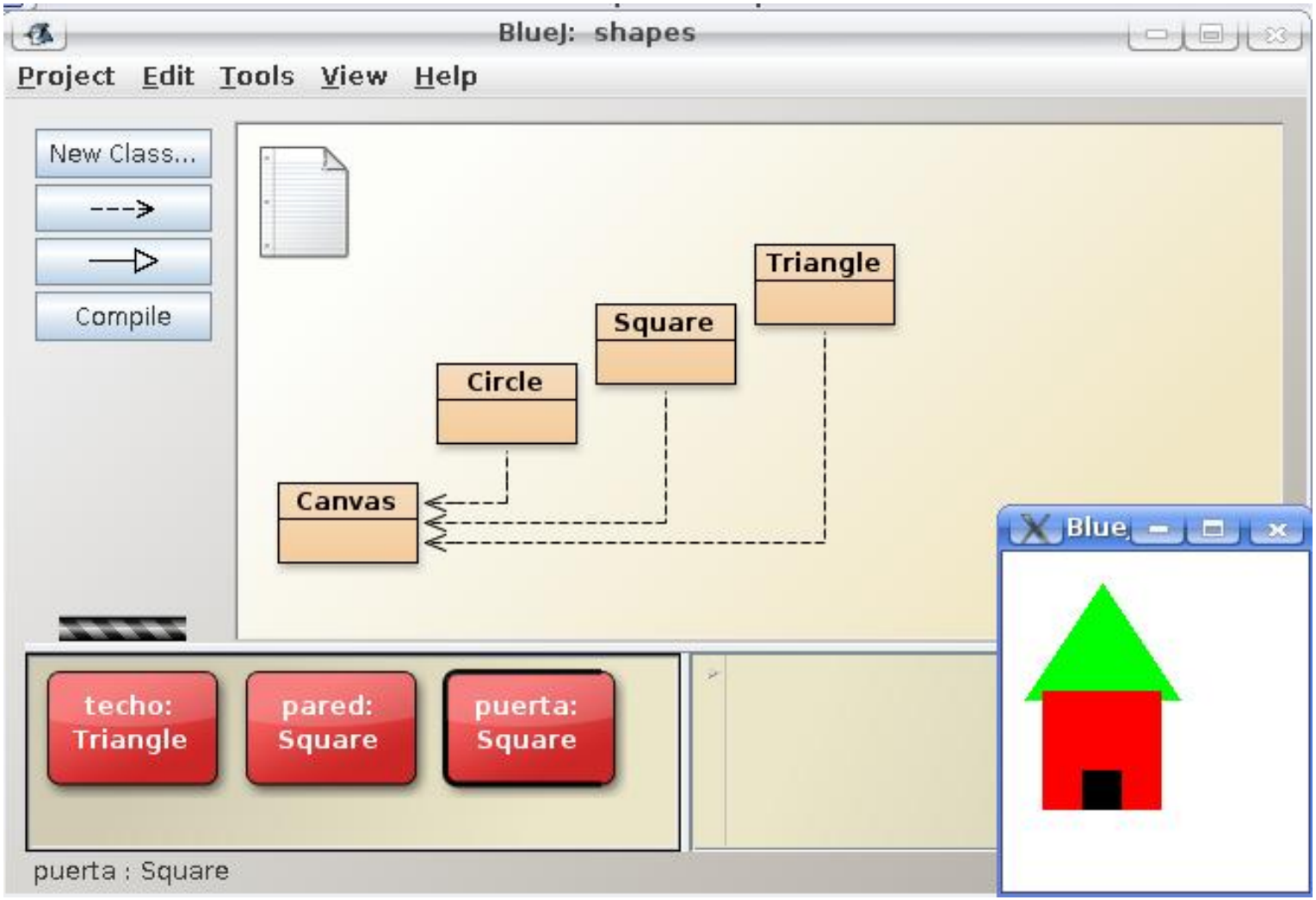
Objetos cuadrados ¿Valores de atributos?

Clases y objetos



Objetos cuadrados ¿Valores de atributos?

Clases y objetos



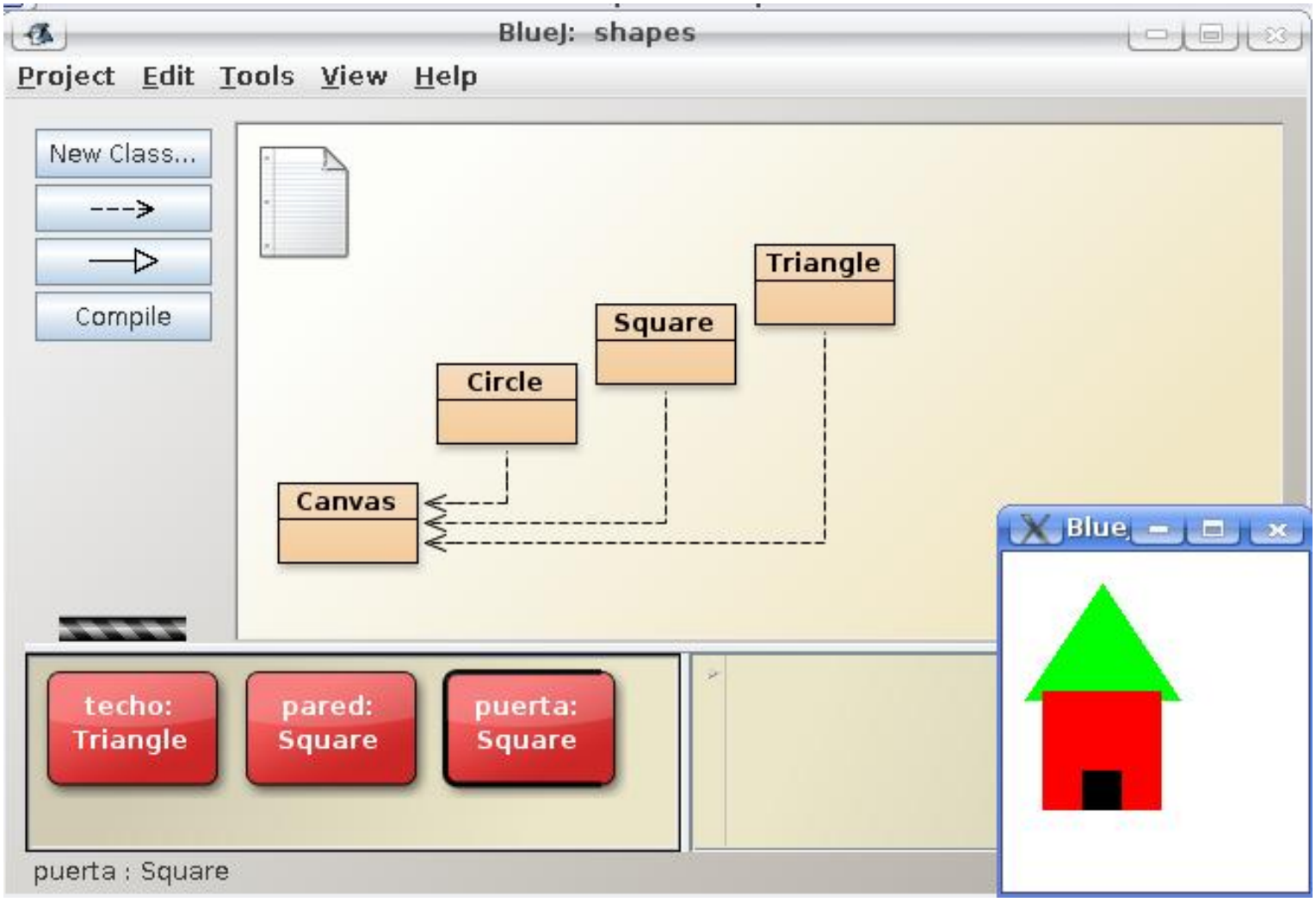
¿Método para crear un cuadrado?

Clases y objetos

```
public class Square {
    private int size;
    private int xPosition;
    private int yPosition;
    private String color;
    private boolean isVisible;

    /**
     * Create a new square at default position with default color.
     */
    public Square() {
        size = 30;
        xPosition = 60;
        yPosition = 50;
        color = "red";
        isVisible = false;
    }
    ...
}
```

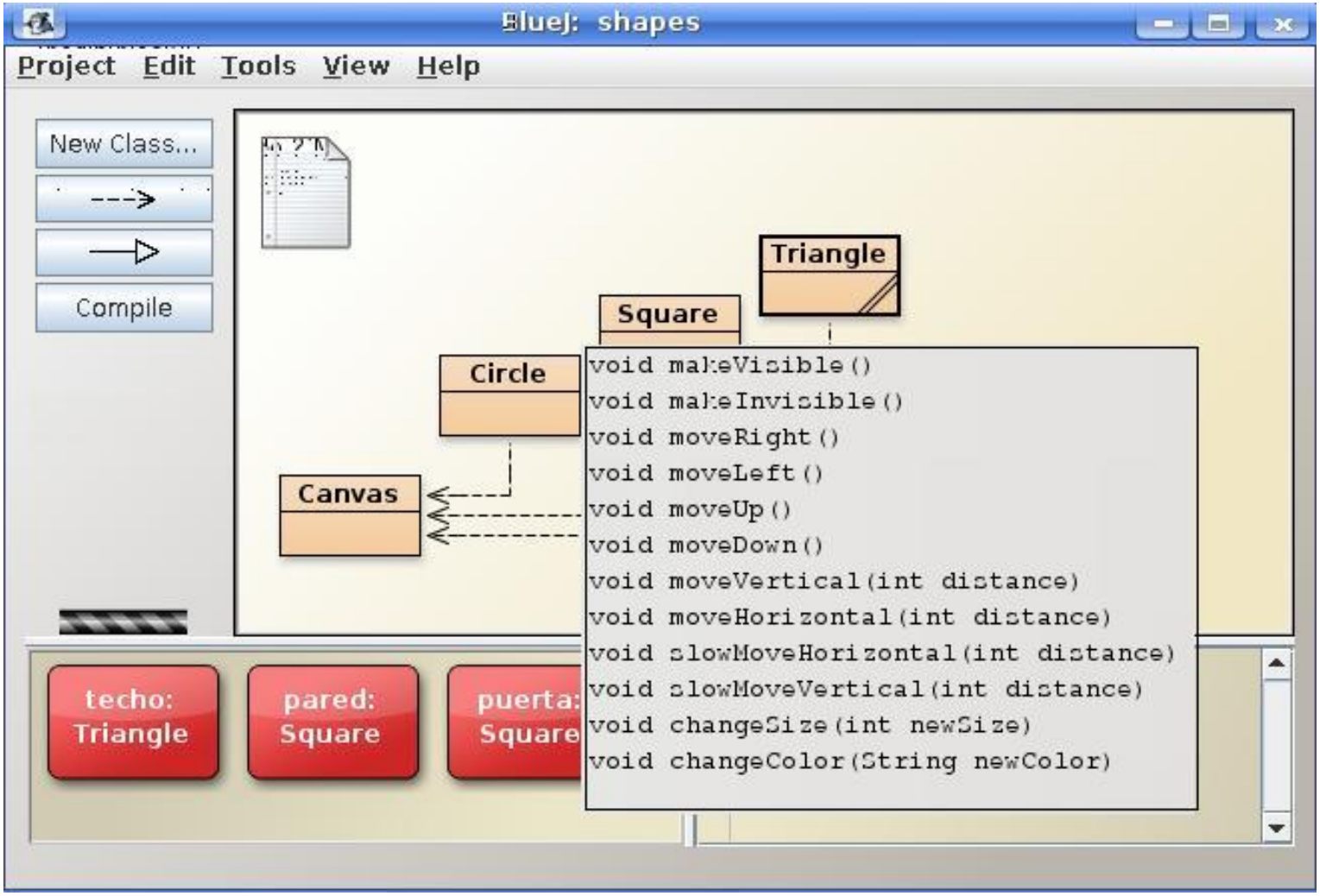
¿Método para crear un cuadrado?

Clases y objetos



Clase cuadrado ¿Qué debe permitir hacer?

Clases y objetos



Objetos cuadrados ¿Métodos?

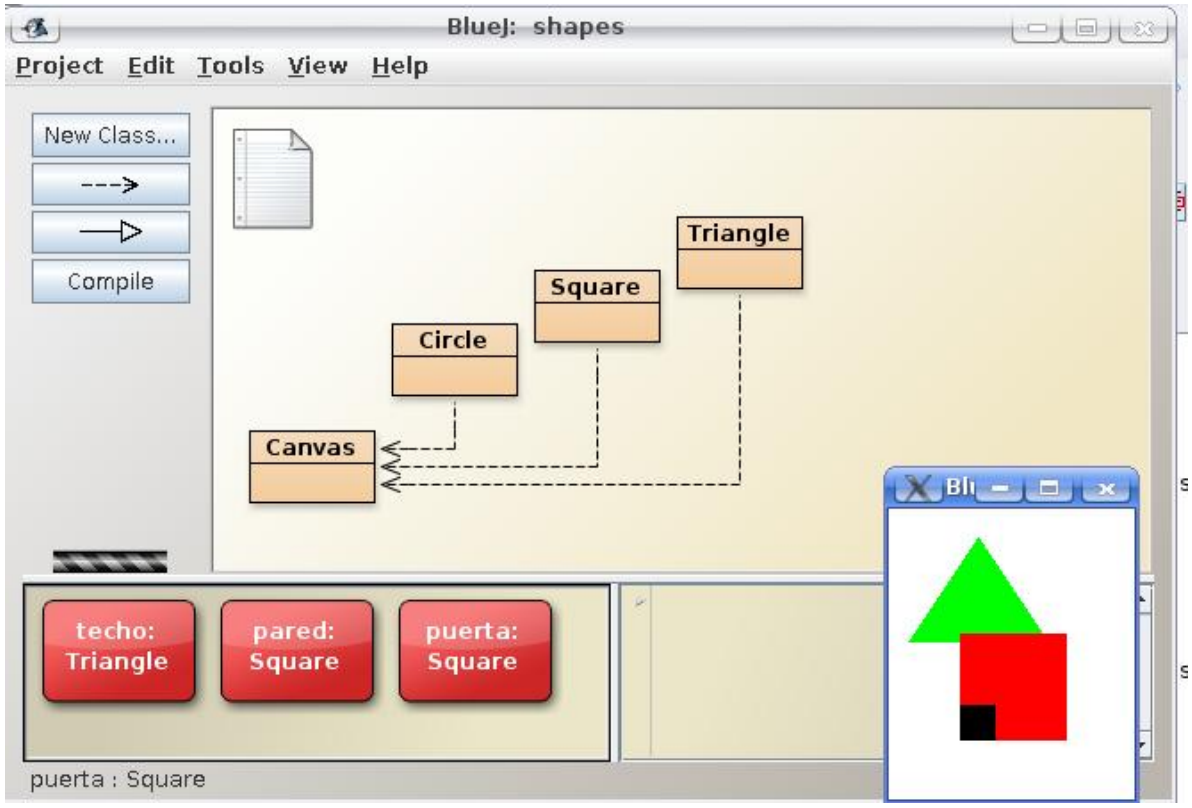
Clases y objetos

```
public class Square {
    private int size;
    private int xPosition;
    private int yPosition;
    private String color;
    private boolean isVisible;


    ...

    /**
     * Move the square horizontally by 'distance' pixels.
     */
    public void moveHorizontal(int distance) {
        erase();
        xPosition += distance;
        draw();
    }
```

Clases y objetos

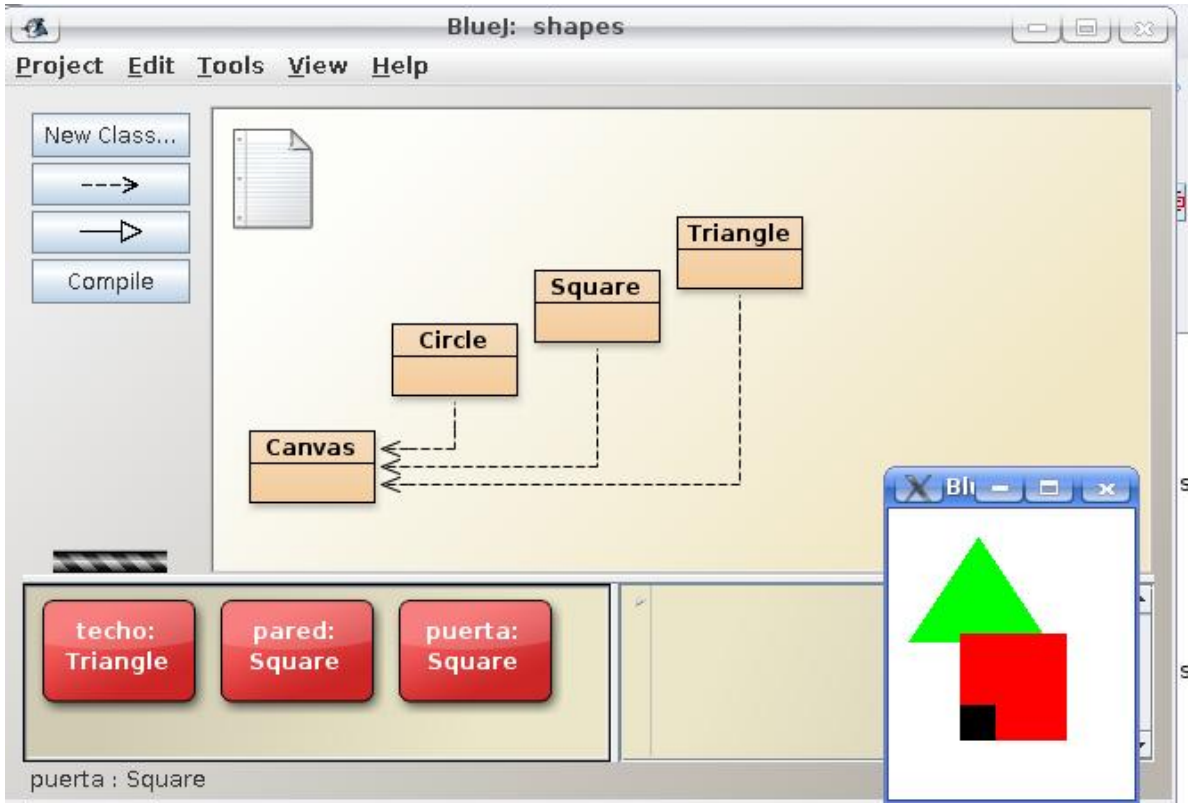
```
public class Square {
    private int size;
    private int xPosition;
    private int yPosition;
    private String color;
    private boolean isVisible;
    ...

    /**
     * Slowly move the square horizontally by 'distance' pixels.
     */
    public void slowMoveHorizontal(int distance) {
        int delta;

        if(distance < 0) {
            delta = -1;
            distance = -distance;
        } else  {
            delta = 1;
        }

        for(int i = 0; i < distance; i++) {
            xPosition += delta;
            draw();
        }
    }
    ...
}
```

Clases y objetos



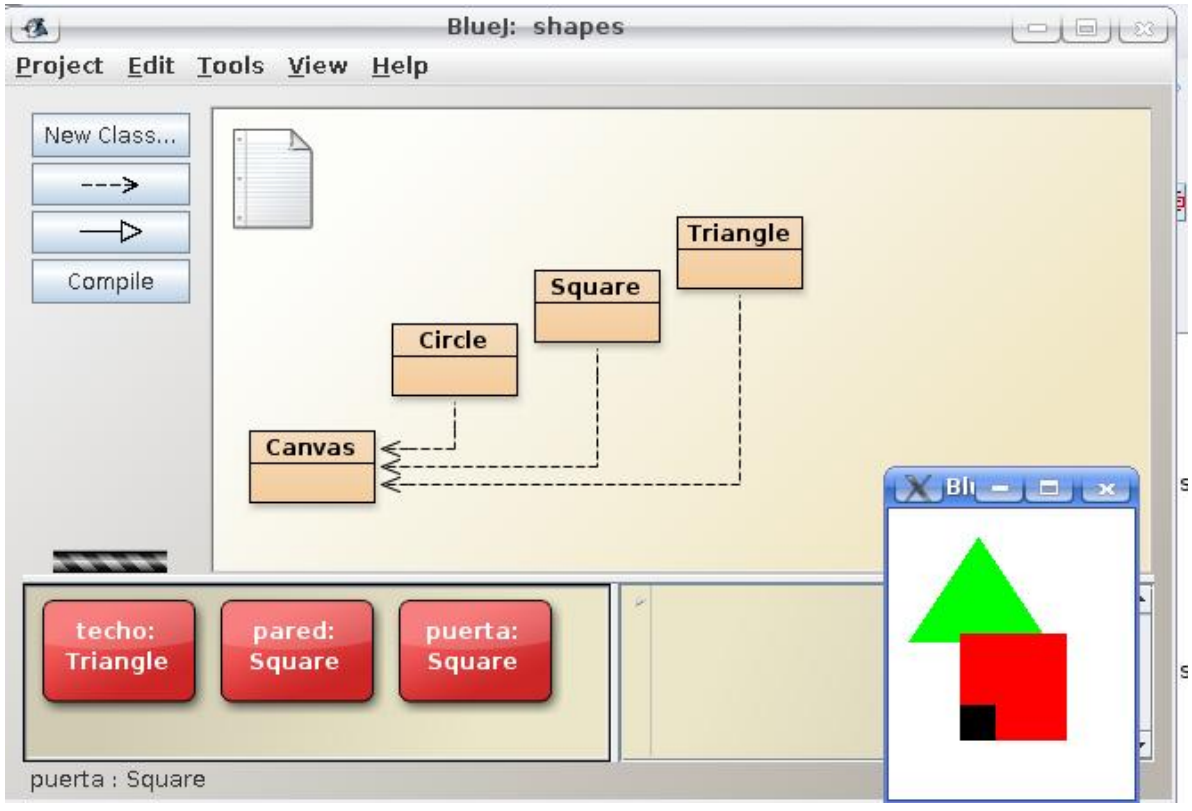
¿Qué pasó?

Clases y objetos



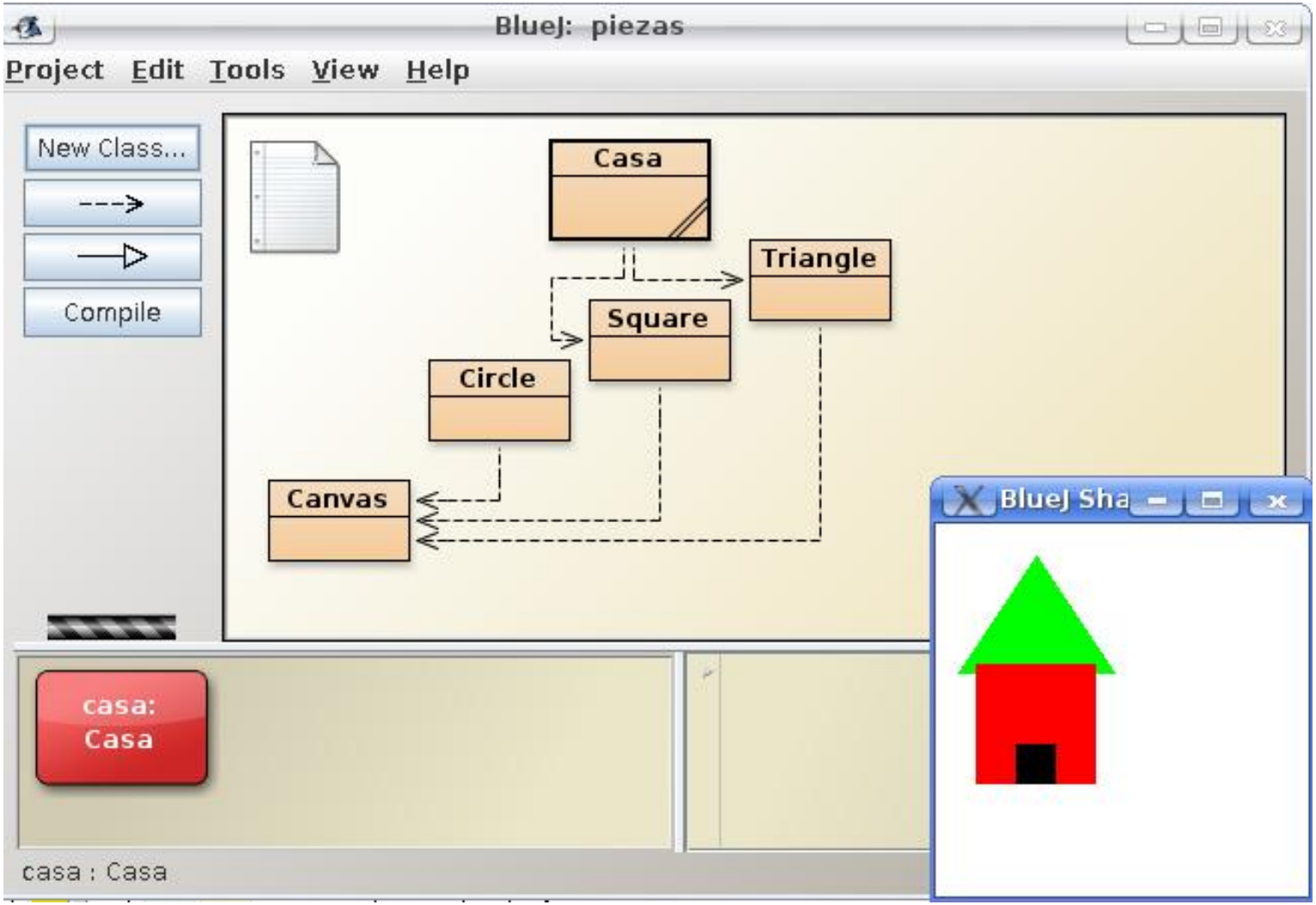
¿Quíen no existe? Aunque la vemos ...

Clases y objetos



¿Qué hacemos?

Clases y objetos



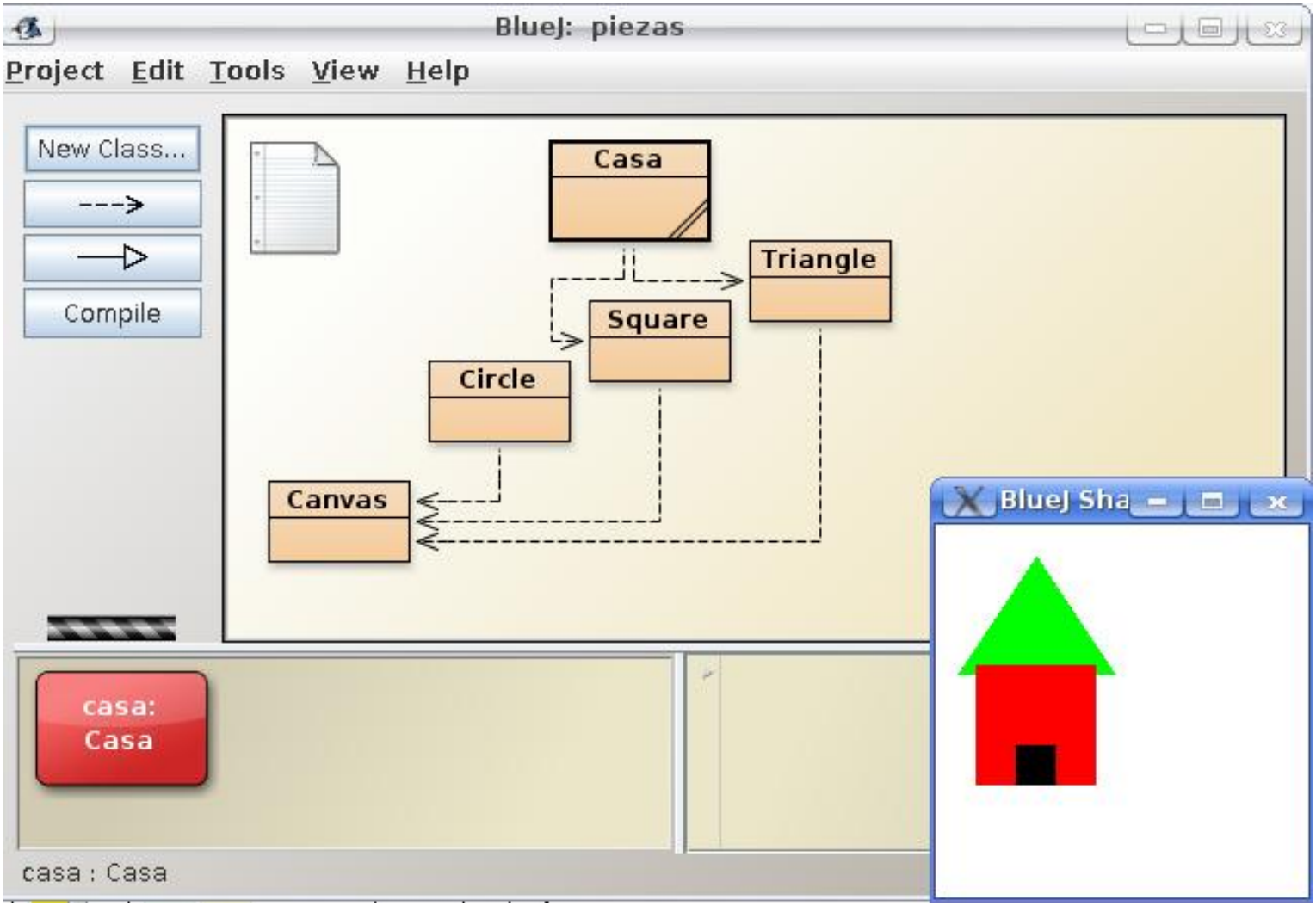
¿Atributos?

Clases y objetos

```
public class Casa {

    private Triangle techo;
    private Square pared;
    private Square puerta;
```

Ingeniería reversa: ¿Diseño?

Clases y objetos

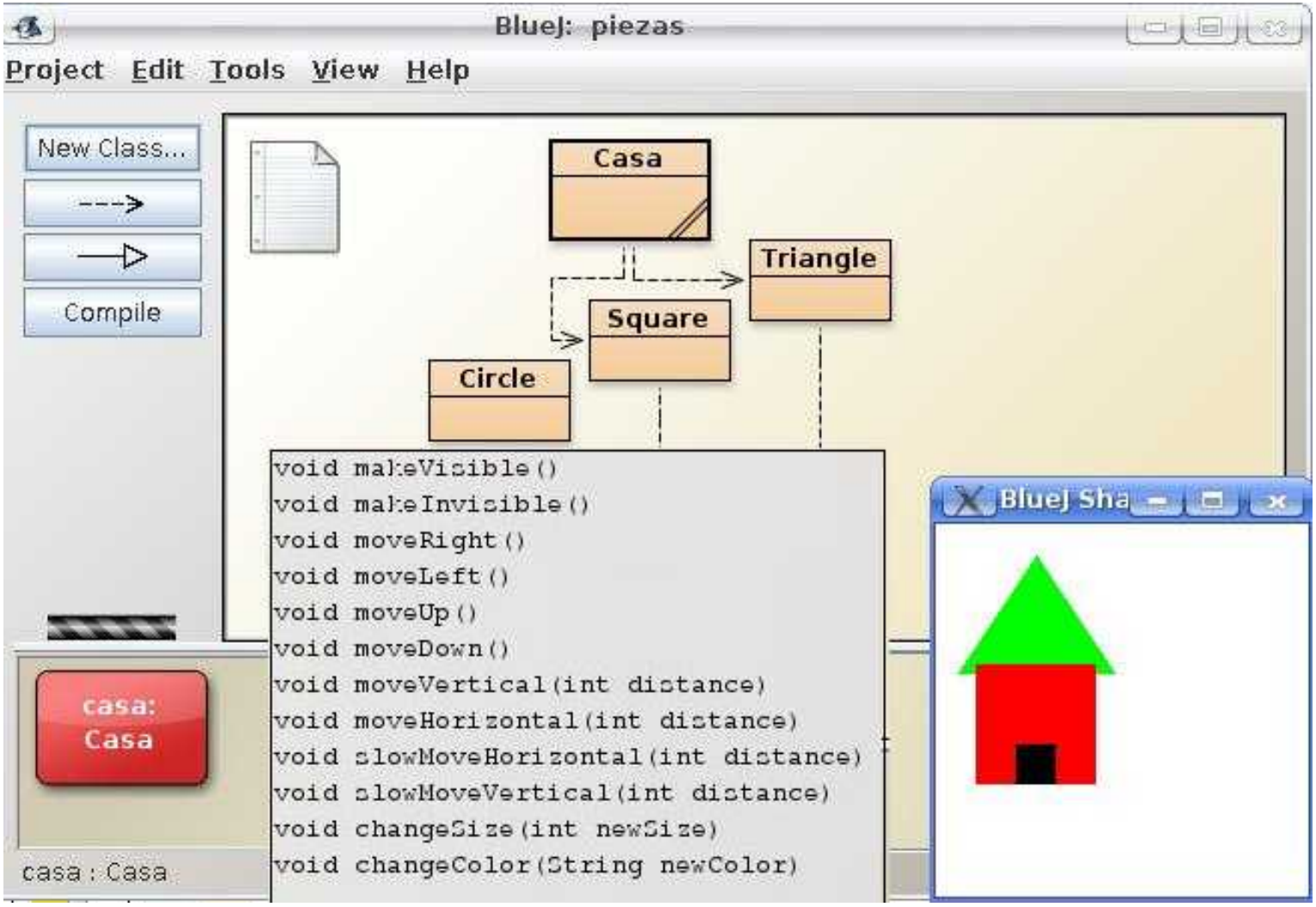


¿Creador? (No considerar tamaño ni ubicación)

Clases y objetos

```
public class Casa {

    private Triangle techo;
    private Square pared;
    private Square puerta;

    public Casa() {
        techo=new Triangle();
        techo.changeSize(60,80);
        pared=new Square();
        pared.changeSize(60);
        pared.moveDown();
        pared.moveLeft();
        pared.moveLeft();
        puerta=new Square();
        puerta.changeSize(20);
        puerta.changeColor("black");
        puerta.moveDown();
        puerta.moveDown();
        puerta.moveDown();
        puerta.moveLeft();
    }
```

Clases y objetos



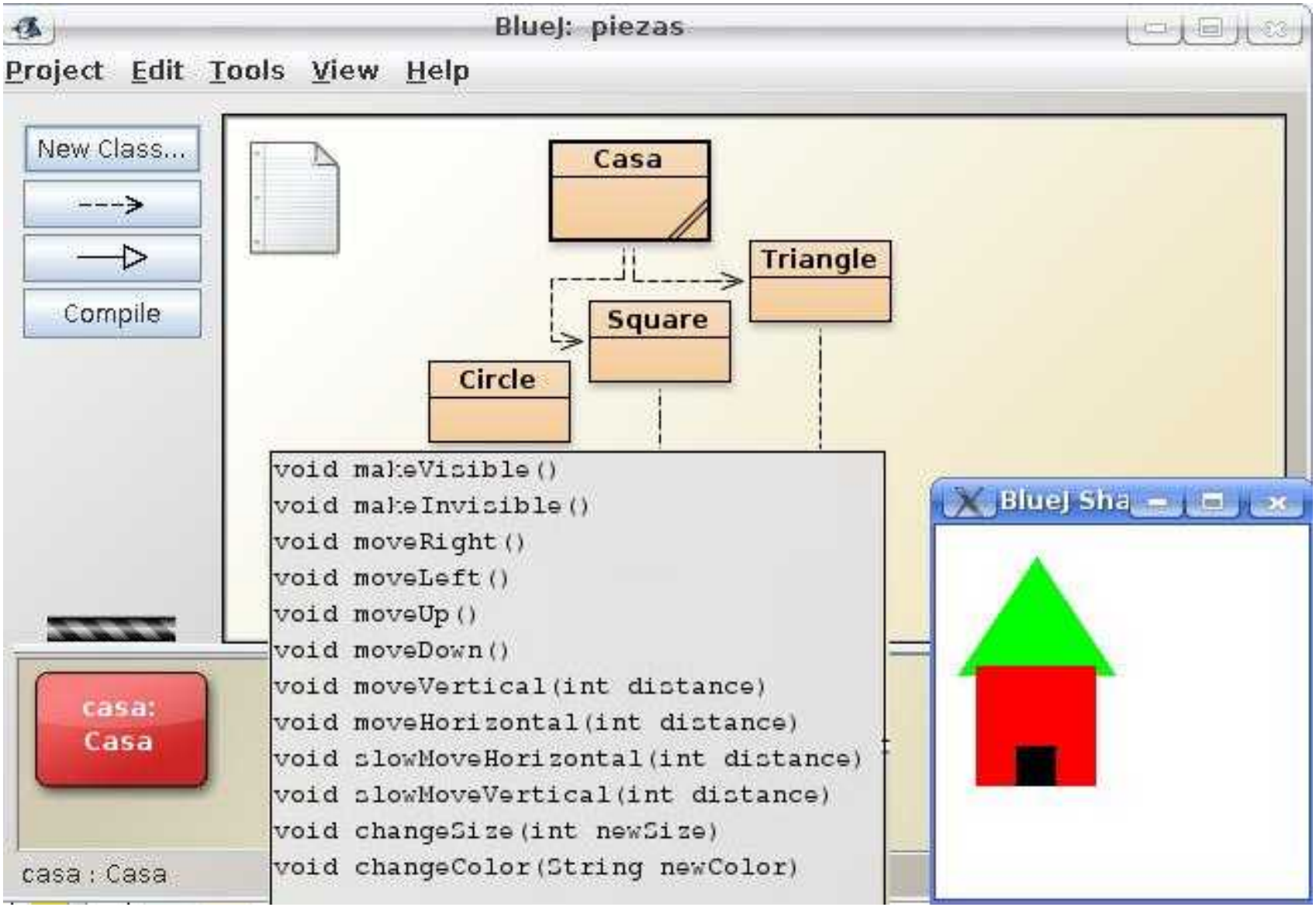
¿Hacer visible la casa?

Clases y objetos

```
    public void makeVisible(){
        techo.makeVisible();
        pared.makeVisible();
        puerta.makeVisible();
    }
}
```

Ingeniería reversa: ¿Diseño?

Clases y objetos



¿Mover despacio la casa? Diseño - Código

Agenda

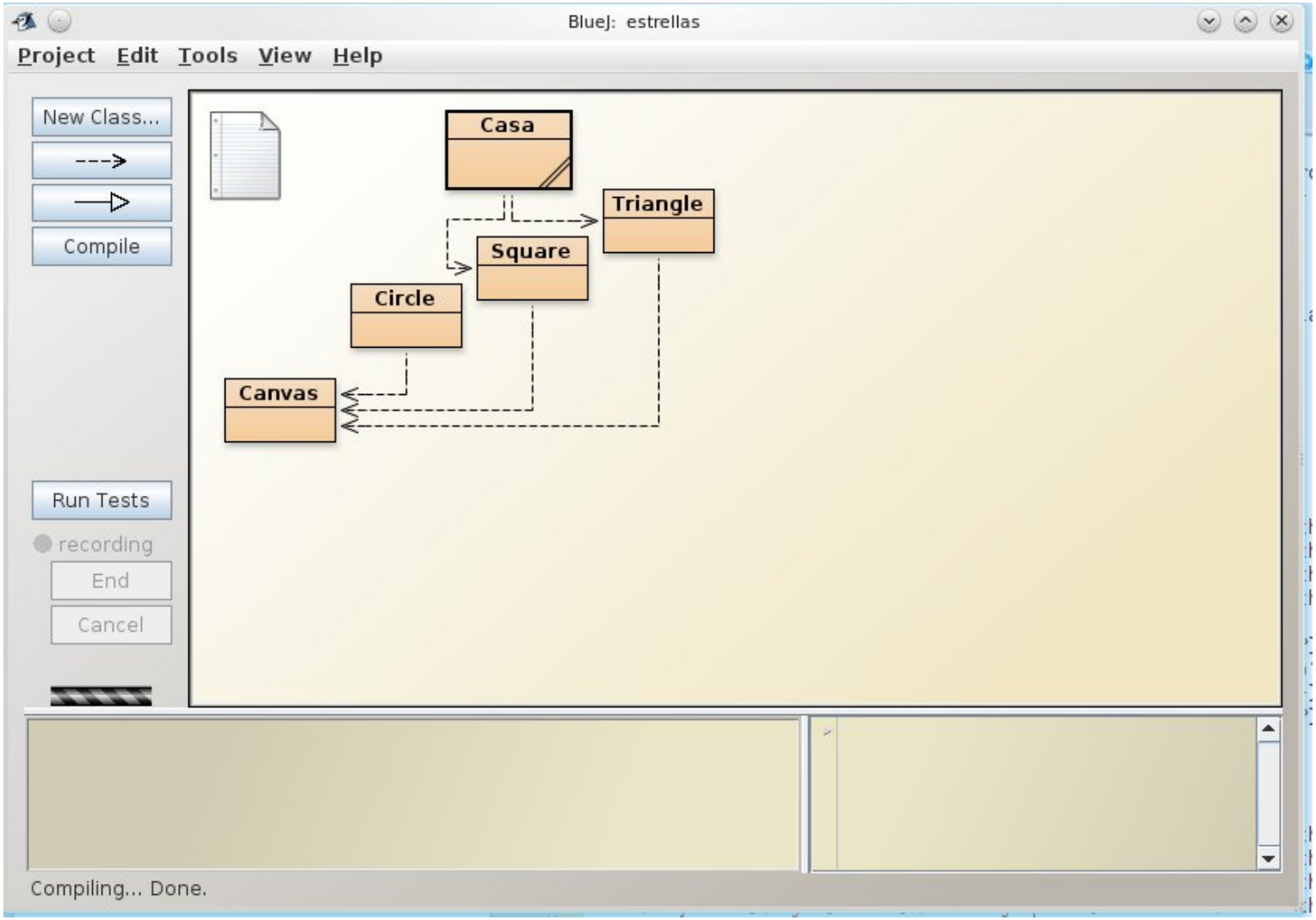
Iniciando
- Las tres P
- Investigación

POOB-Curso
- Descripción
- Prácticas, lenguajes y herramientas

Orientación a objetos
- Materia prima
- Clases y objetos
- Atributos
- Métodos
- Casa

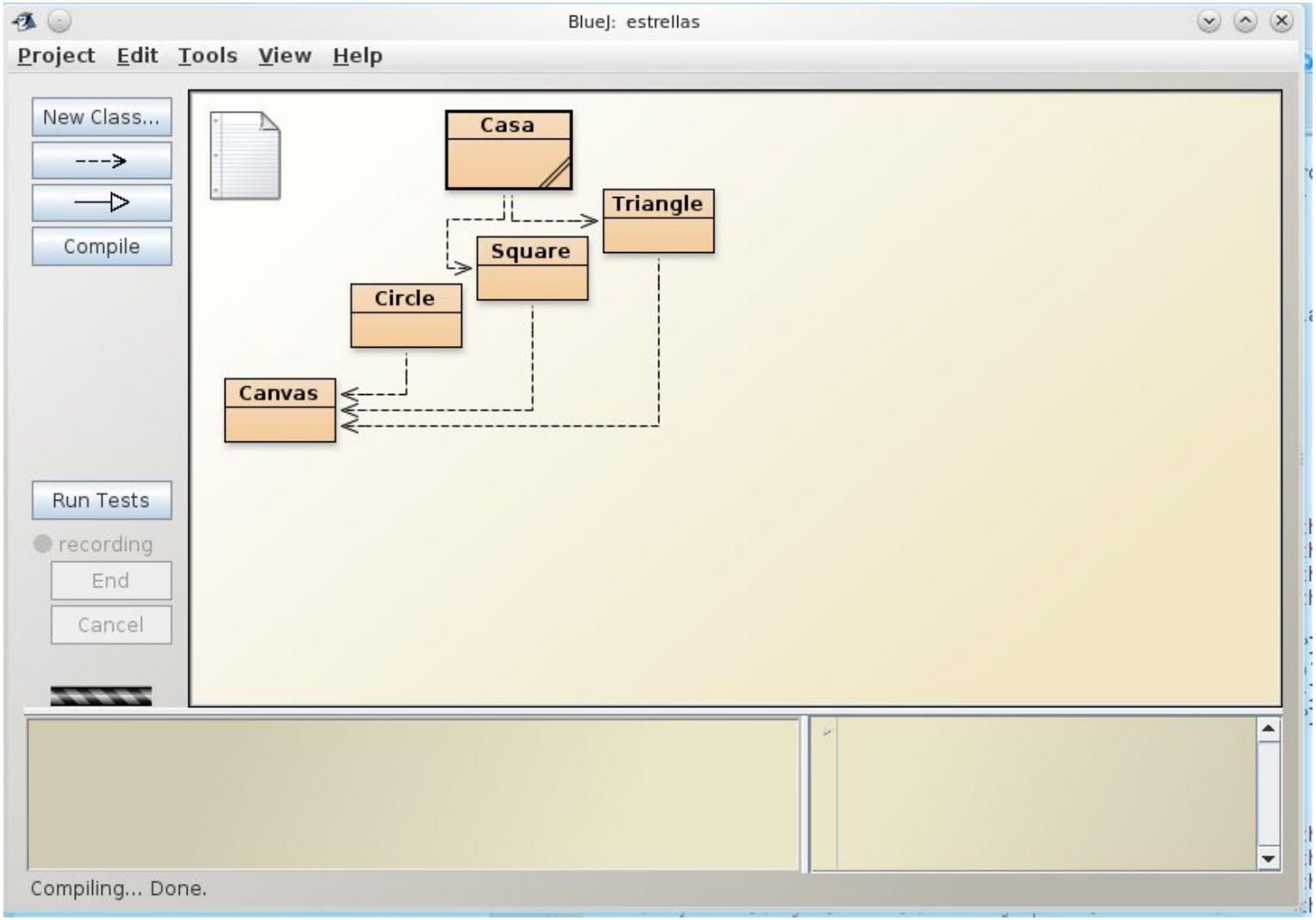
Herramienta. BlueJ
- General
- Editar
- Compilar
- Ejecutar
- Documentar

General



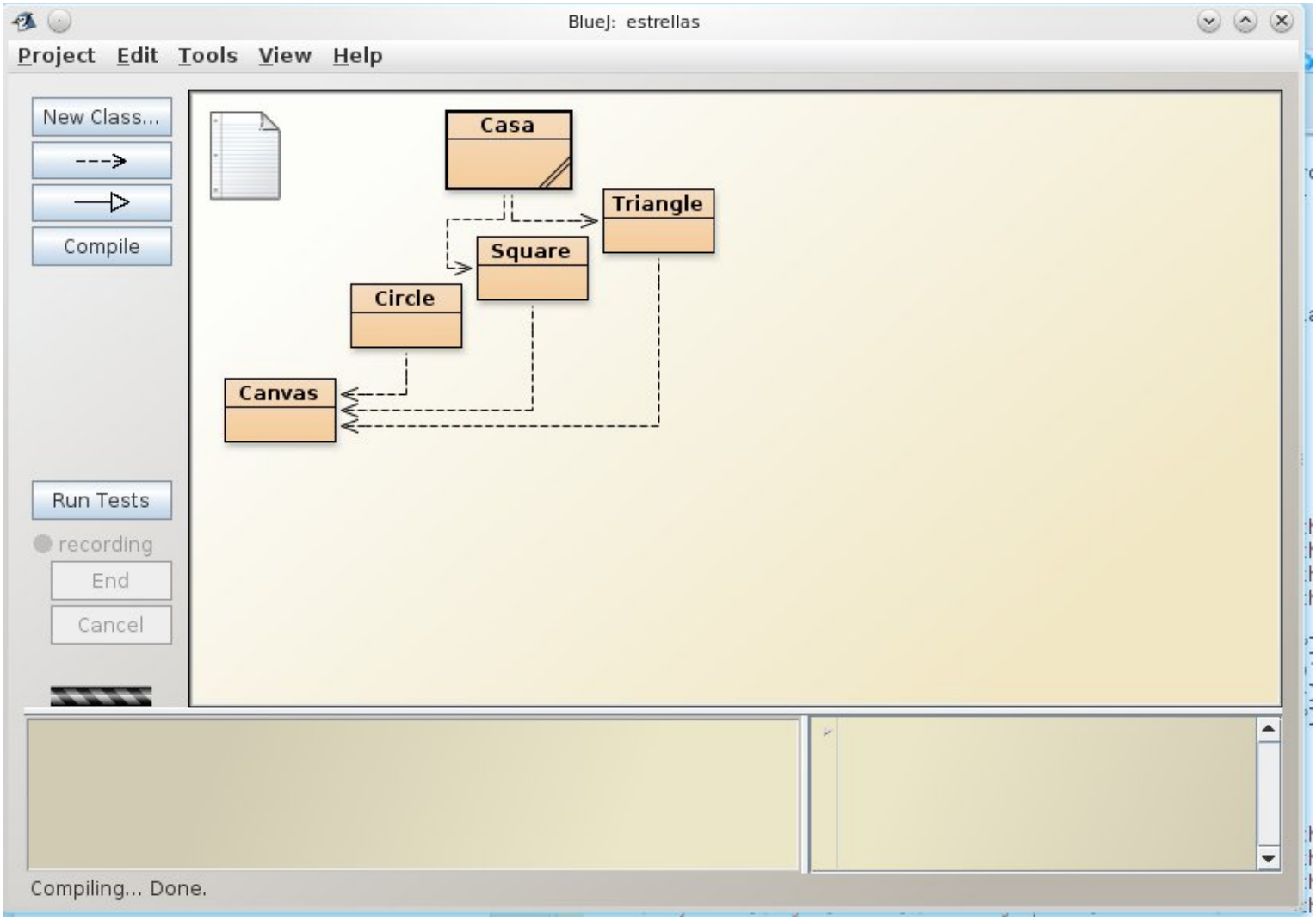
¿Qué debería permitir?

General



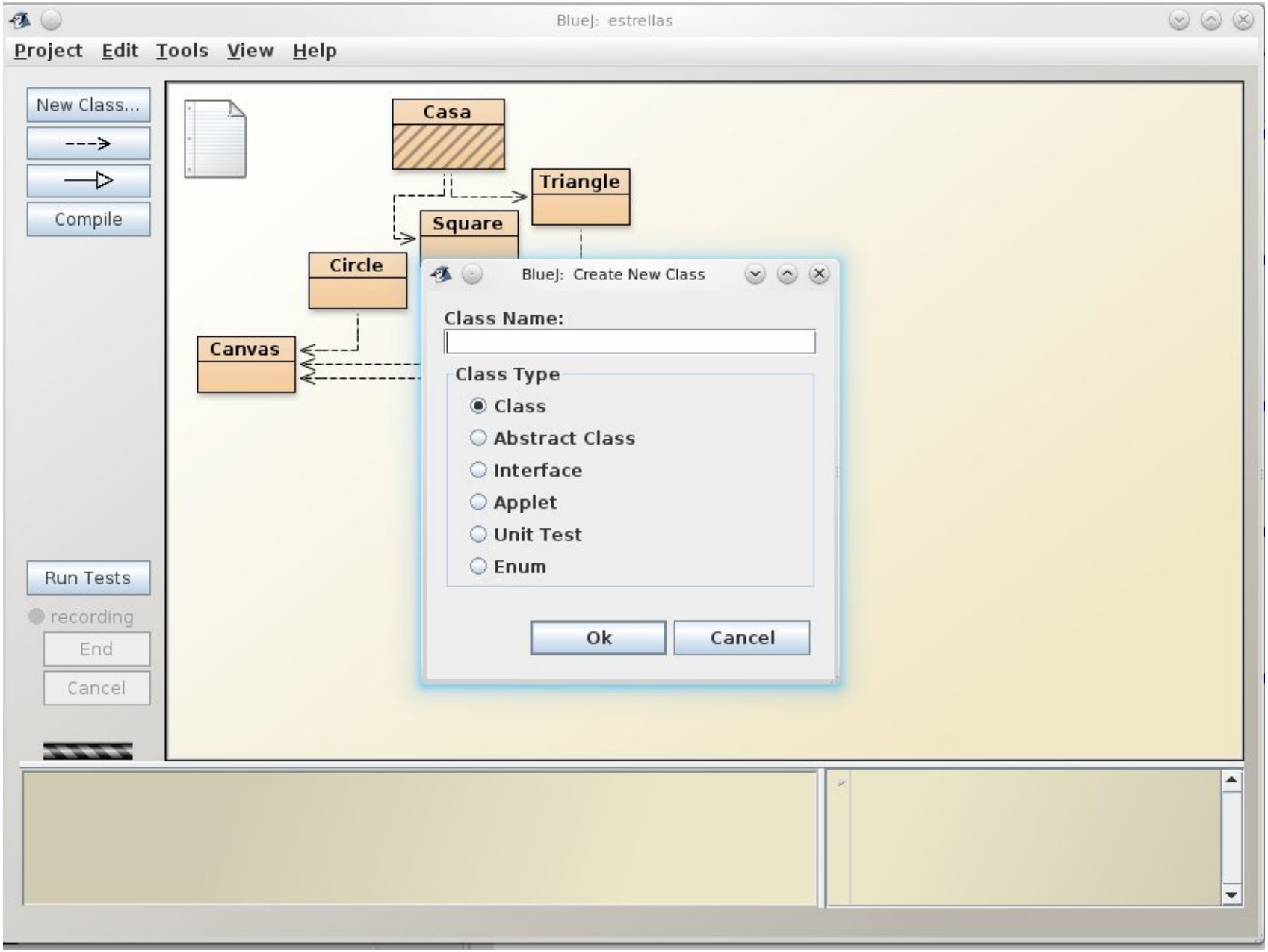
¿Qué usa?

General



¿En qué está desarrollado?

Editar


BlueJ: estrellas
Project Edit Tools View Help
New Class...
Compile
Run Tests
recording
End
Cancel
Casa
Triangle
Square
Circle
Canvas
BlueJ: Create New Class
Class Name:
Class Type
Class
Abstract Class
Interface
Applet
Unit Test
Enum
Ok
Cancel

Editar

```java
import java.awt.*;
import java.awt.geom.*;

/**
 * A circle that can be manipulated and that draws itself on a canvas.
 *
 * @author  Michael Kolling and David J. Barnes
 * @version 1.0  (15 July 2000)
 */

public class Circle {
    public static final double PI=3.1416;

    private int diameter;
    private int xPosition;
    private int yPosition;
    private String color;
    private boolean isVisible;

    /**
     * Create a new circle at default position with default color.
     */
    public Circle() {
        diameter = 30;
        xPosition = 20;
        yPosition = 60;
        color = "blue";
        isVisible = false;
    }

    /**
     * Make this circle visible. If it was already visible, do nothing.
     */
    public void makeVisible() {
        isVisible = true;
        draw();
    }
```

¿Cómo se llama la clase?

Editar

```java
import java.awt.*;
import java.awt.geom.*;

/**
 * A circle that can be manipulated and that draws itself on a canvas.
 *
 * @author  Michael Kolling and David J. Barnes
 * @version 1.0  (15 July 2000)
 */

public class Circle {
    public static final double PI=3.1416;

    private int diameter;
    private int xPosition;
    private int yPosition;
    private String color;
    private boolean isVisible;

    /**
     * Create a new circle at default position with default color.
     */
    public Circle() {
        diameter = 30;
        xPosition = 20;
        yPosition = 60;
        color = "blue";
        isVisible = false;
    }

    /**
     * Make this circle visible. If it was already visible, do nothing.
     */
    public void makeVisible() {
        isVisible = true;
        draw();
    }
```

¿Cuántos atributos tiene?

Editar

```java
import java.awt.*;
import java.awt.geom.*;

/**
 * A circle that can be manipulated and that draws itself on a canvas.
 *
 * @author  Michael Kolling and David J. Barnes
 * @version 1.0  (15 July 2000)
 */

public class Circle {
    public static final double PI=3.1416;

    private int diameter;
    private int xPosition;
    private int yPosition;
    private String color;
    private boolean isVisible;

    /**
     * Create a new circle at default position with default color.
     */
    public Circle() {
        diameter = 30;
        xPosition = 20;
        yPosition = 60;
        color = "blue";
        isVisible = false;
    }

    /**
     * Make this circle visible. If it was already visible, do nothing.
     */
    public void makeVisible() {
        isVisible = true;
        draw();
    }
```

¿Qué diferencia existe entre PI y diameter?

Editar

```
import java.awt.*;
import java.awt.geom.*;

/**
 * A circle that can be manipulated and that draws itself on a canvas.
 *
 * @author  Michael Kolling and David J. Barnes
 * @version 1.0  (15 July 2000)
 */

public class Circle {
    public static final double PI=3.1416;

    private int diameter;
    private int xPosition;
    private int yPosition;
    private String color;
    private boolean isVisible;

    /**
     * Create a new circle at default position with default color.
     */
    public Circle() {
        diameter = 30;
        xPosition = 20;
        yPosition = 60;
        color = "blue";
        isVisible = false;
    }

    /**
     * Make this circle visible. If it was already visible, do nothing.
     */
    public void makeVisible() {
        isVisible = true;
        draw();
    }
```

¿Qué otras clases se usan?

Editar

```java
import java.awt.*;
import java.awt.geom.*;

/**
 * A circle that can be manipulated and that draws itself on a canvas.
 *
 * @author  Michael Kolling and David J. Barnes
 * @version 1.0  (15 July 2000)
 */

public class Circle {
    public static final double PI=3.1416;

    private int diameter;
    private int xPosition;
    private int yPosition;
    private String color;
    private boolean isVisible;

    /**
     * Create a new circle at default position with default color.
     */
    public Circle() {
        diameter = 30;
        xPosition = 20;
        yPosition = 60;
        color = "blue";
        isVisible = false;
    }

    /**
     * Make this circle visible. If it was already visible, do nothing.
     */
    public void makeVisible() {
        isVisible = true;
        draw();
    }
```

¿Cuántos métodos vemos?

Editar

```
import java.awt.*;
import java.awt.geom.*;

/**
 * A circle that can be manipulated and that draws itself on a canvas.
 *
 * @author  Michael Kolling and David J. Barnes
 * @version 1.0  (15 July 2000)
 */

public class Circle {
    public static final double PI=3.1416;

    private int diameter;
    private int xPosition;
    private int yPosition;
    private String color;
    private boolean isVisible;

    /**
     * Create a new circle at default position with default color.
     */
    public Circle() {
        diameter = 30;
        xPosition = 20;
        yPosition = 60;
        color = "blue";
        isVisible = false;
    }

    /**
     * Make this circle visible. If it was already visible, do nothing.
     */
    public void makeVisible() {
        isVisible = true;
        draw();
    }
```

¿Qué diferencia existe entre Circle y makeVisible?

Editar

```java
import java.awt.*;
import java.awt.geom.*;

/**
 * A circle that can be manipulated and that draws itself on a canvas.
 *
 * @author  Michael Kolling and David J. Barnes
 * @version 1.0  (15 July 2000)
 */

public class Circle {
    public static final double PI=3.1416;

    private int diameter;
    private int xPosition;
    private int yPosition;
    private String color;
    private boolean isVisible;

    /**
     * Create a new circle at default position with default color.
     */
    public Circle() {
        diameter = 30;
        xPosition = 20;
        yPosition = 60;
        color = "blue";
        isVisible = false;
    }

    /**
     * Make this circle visible. If it was already visible, do nothing.
     */
    public void makeVisible() {
        isVisible = true;
        draw();
    }
```

¿Cómo se crea un círculo? (Llámelo point)

Editar

```java
import java.awt.*;
import java.awt.geom.*;

/**
 * A circle that can be manipulated and that draws itself on a canvas.
 *
 * @author  Michael Kolling and David J. Barnes
 * @version 1.0  (15 July 2000)
 */

public class Circle {
    public static final double PI=3.1416;

    private int diameter;
    private int xPosition;
    private int yPosition;
    private String color;
    private boolean isVisible;

    /**
     * Create a new circle at default position with default color.
     */
    public Circle() {
        diameter = 30;
        xPosition = 20;
        yPosition = 60;
        color = "blue";
        isVisible = false;
    }

    /**
     * Make this circle visible. If it was already visible, do nothing.
     */
    public void makeVisible() {
        isVisible = true;
        draw();
    }
```

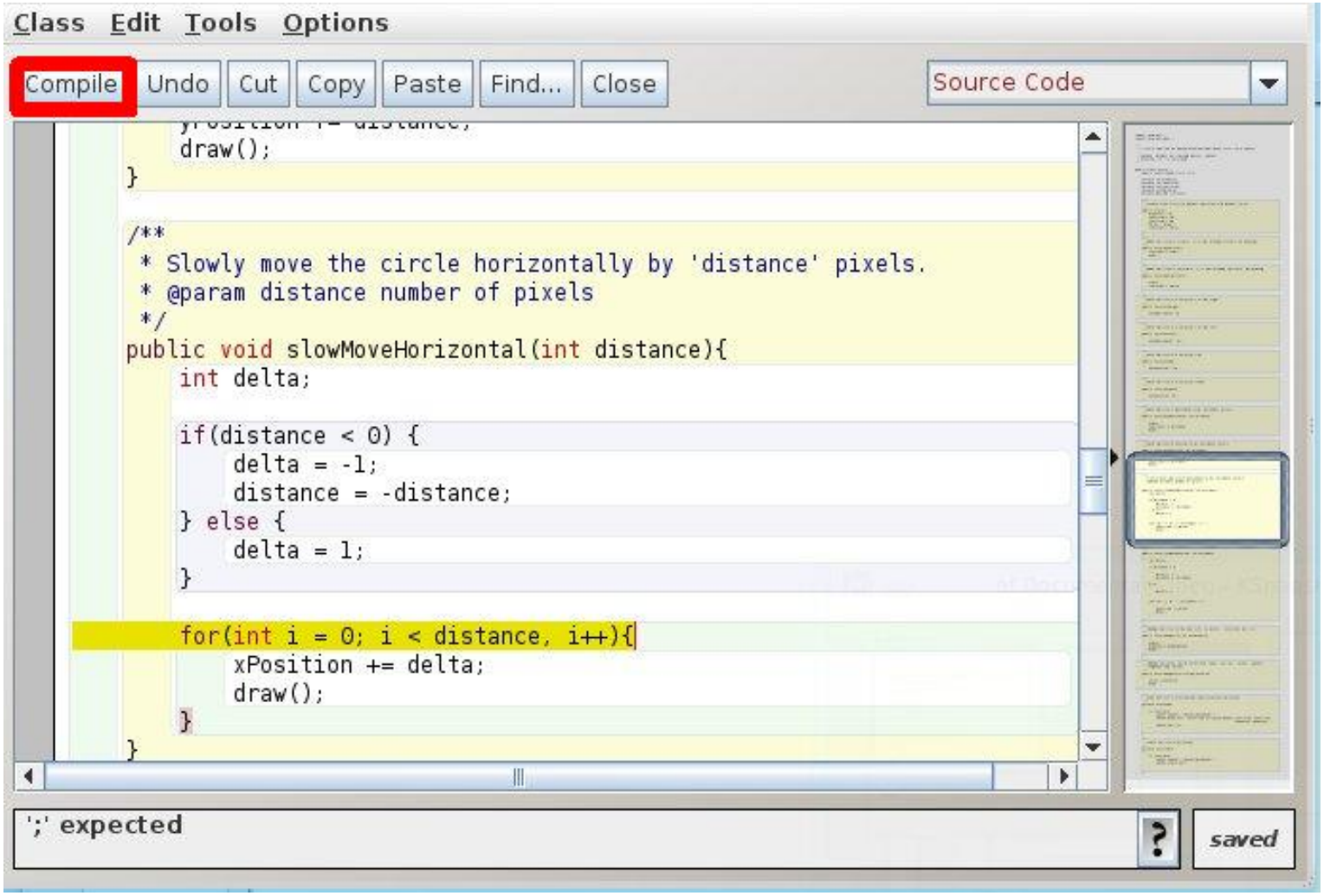
¿Cómo se hace visible el círculo anterior?

Editar

```
import java.awt.*;
import java.awt.geom.*;

/**
 * A circle that can be manipulated and that draws itself on a canvas.
 *
 * @author  Michael Kolling and David J. Barnes
 * @version 1.0  (15 July 2000)
 */

public class Circle {
    public static final double PI=3.1416;

    private int diameter;
    private int xPosition;
    private int yPosition;
    private String color;
    private boolean isVisible;

    /**
     * Create a new circle at default position with default color.
     */
    public Circle() {
        diameter = 30;
        xPosition = 20;
        yPosition = 60;
        color = "blue";
        isVisible = false;
    }

    /**
     * Make this circle visible. If it was already visible, do nothing.
     */
    public void makeVisible() {
        isVisible = true;
        draw();
    }
```

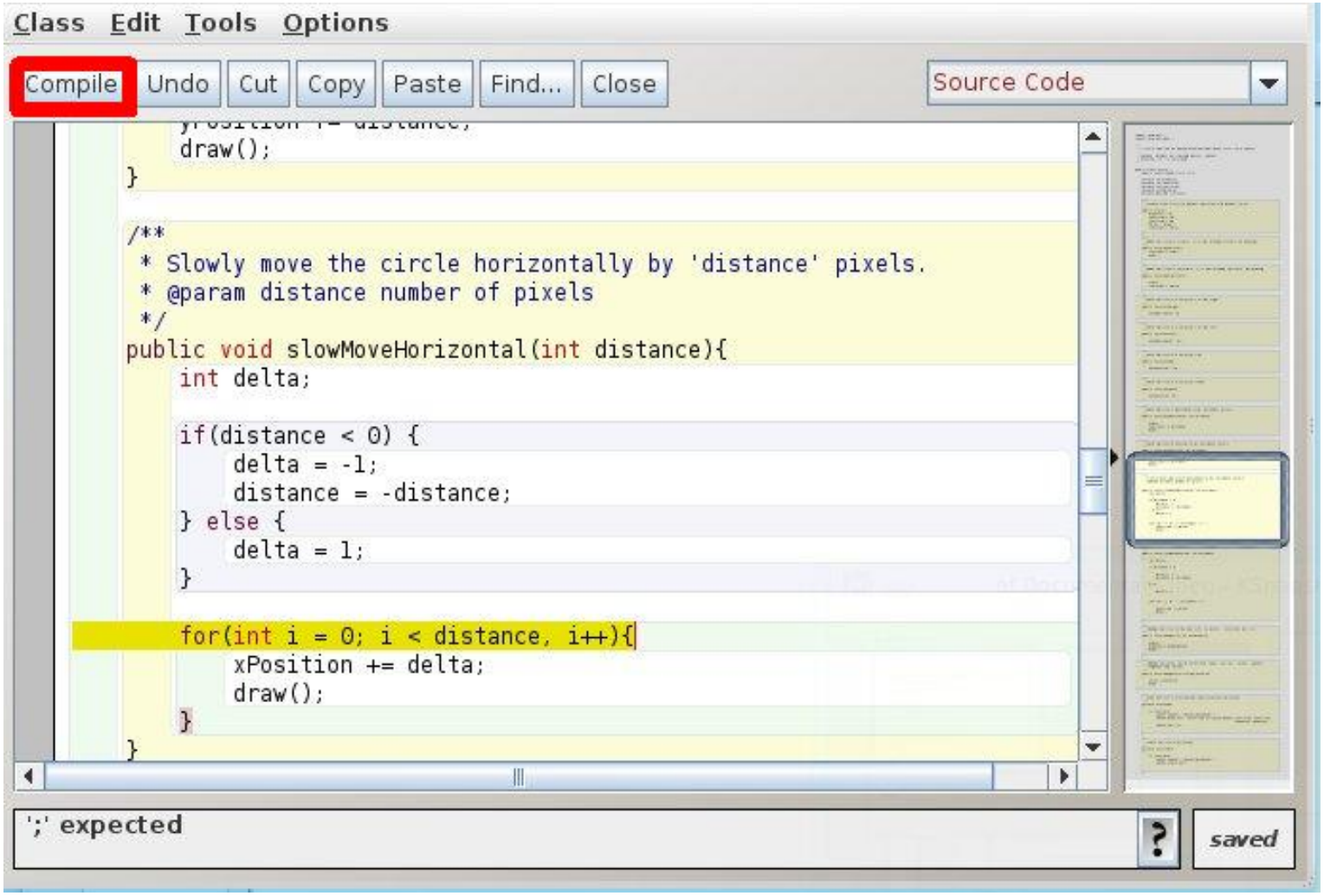
¿Cuál es el nombre del archivo?

Compilar



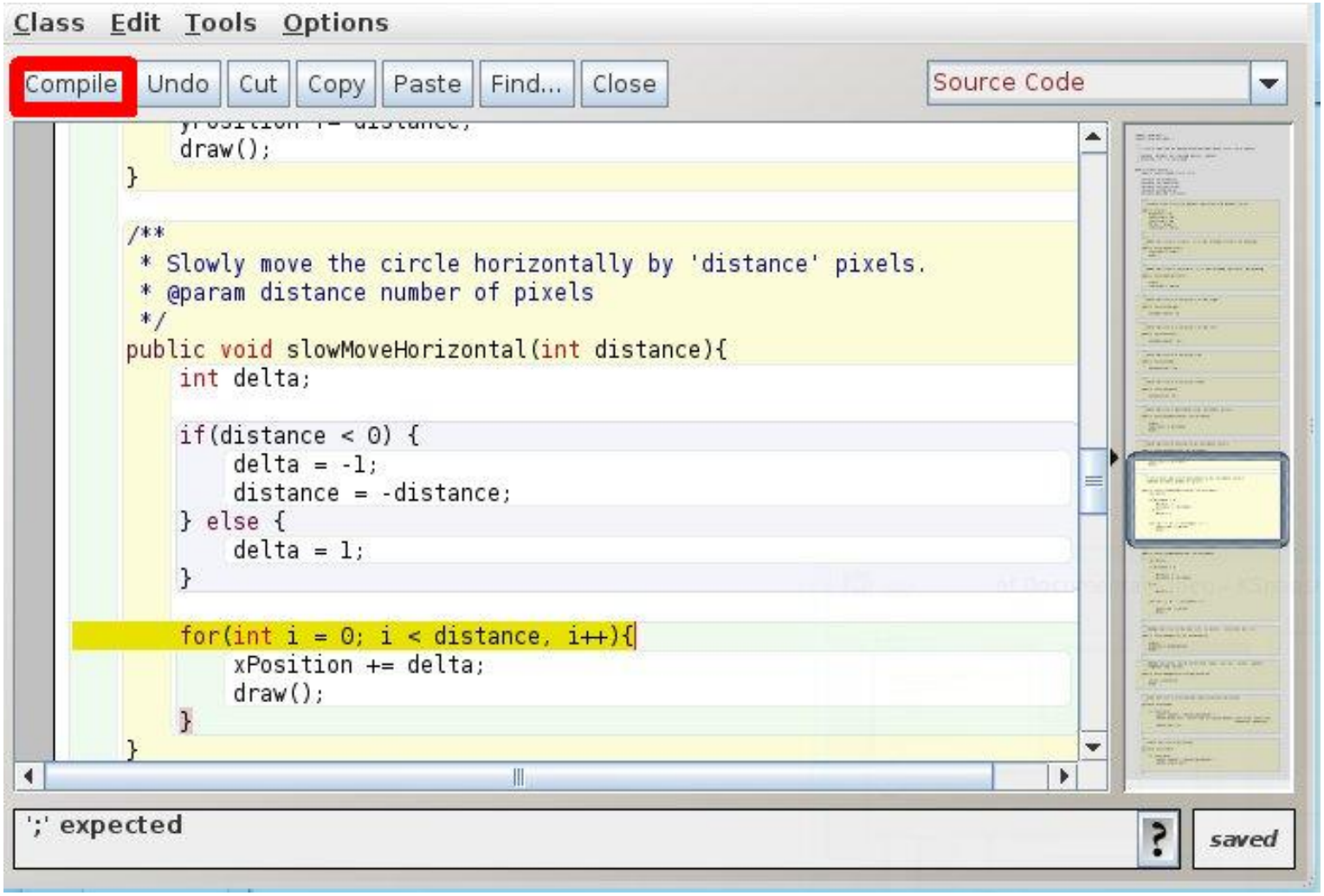
¿Cuál es la herramienta básica JDK?

Compilar



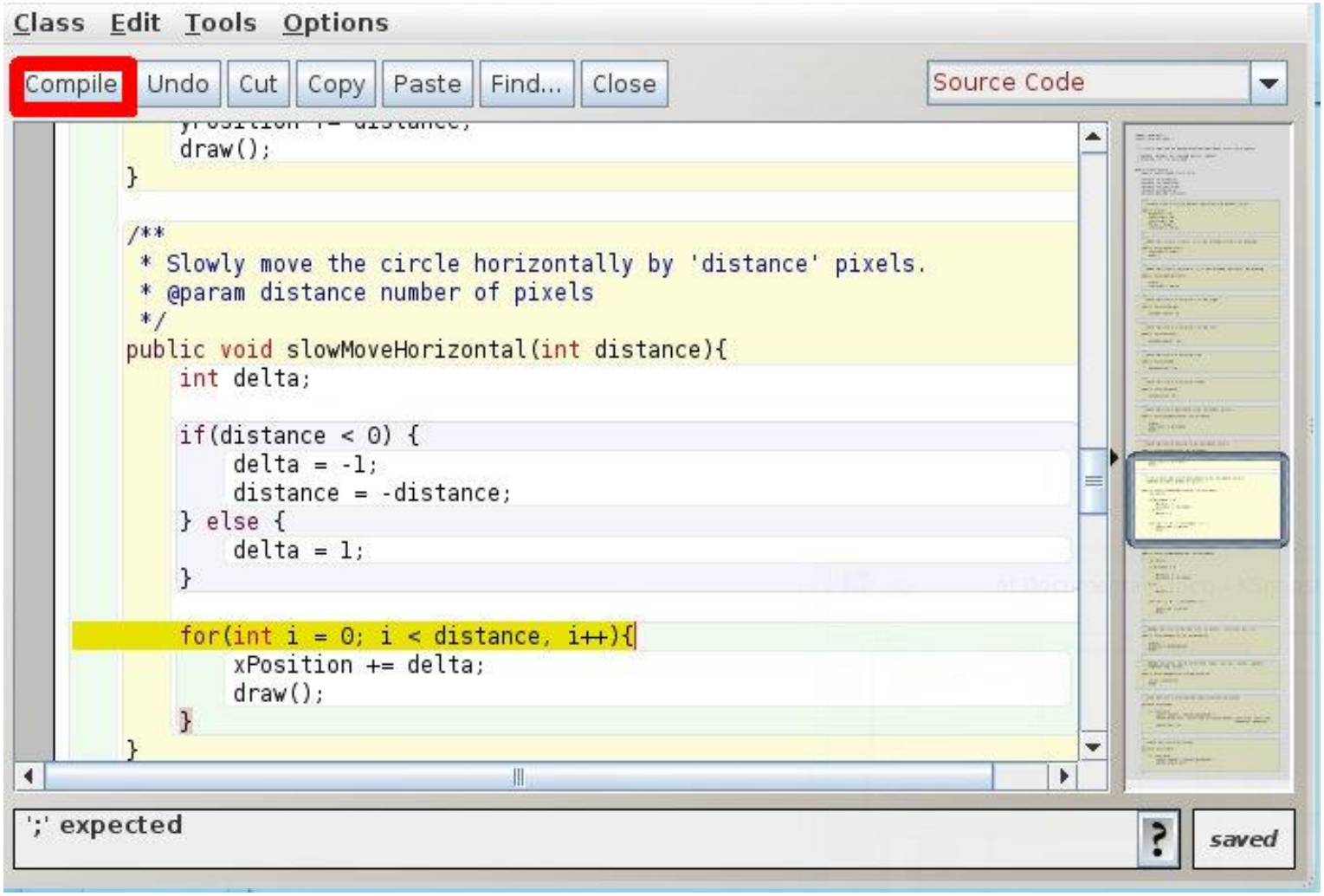
¿Qué error tiene el código?

Compilar



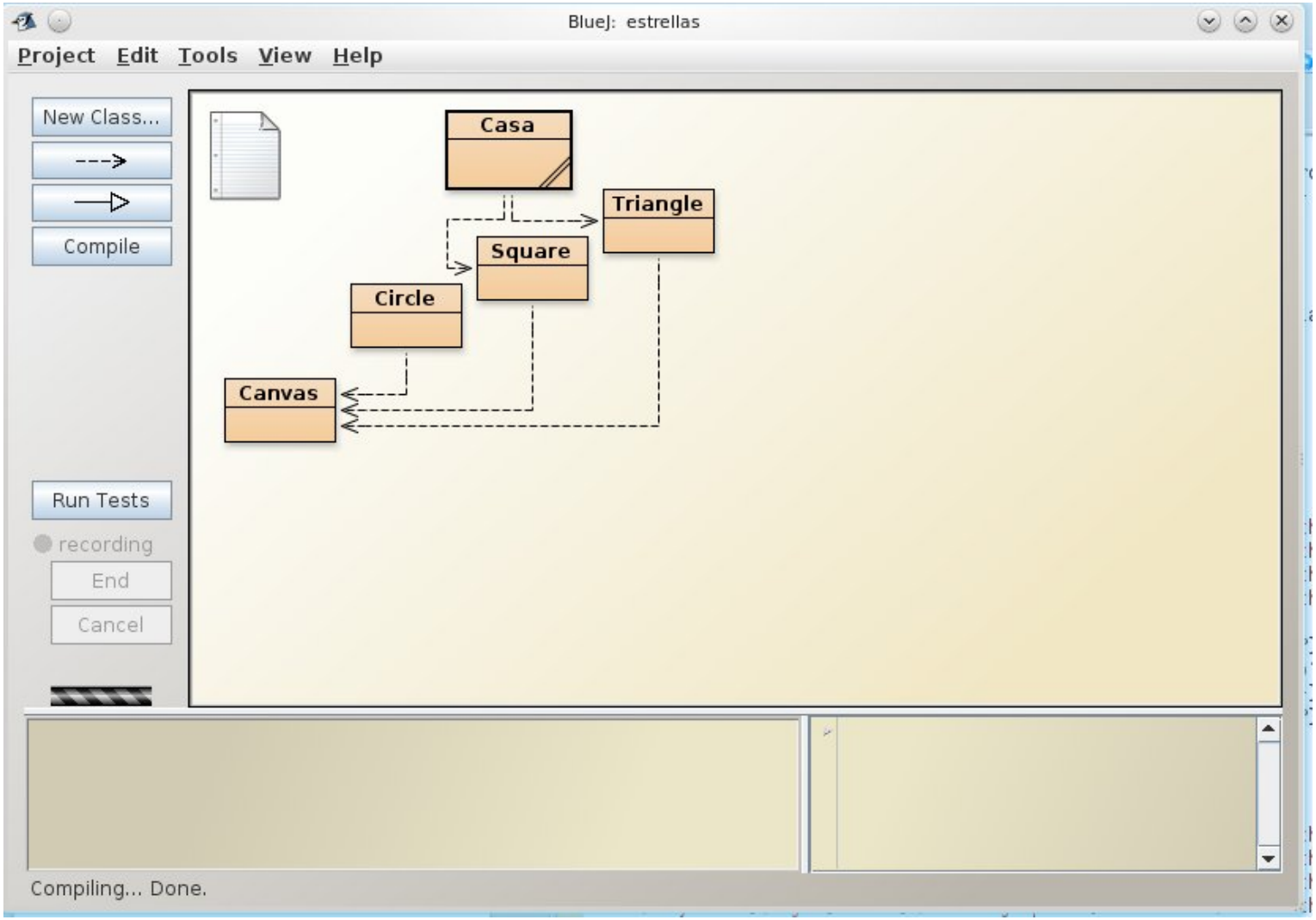
¿Qué hace el método?

Compilar



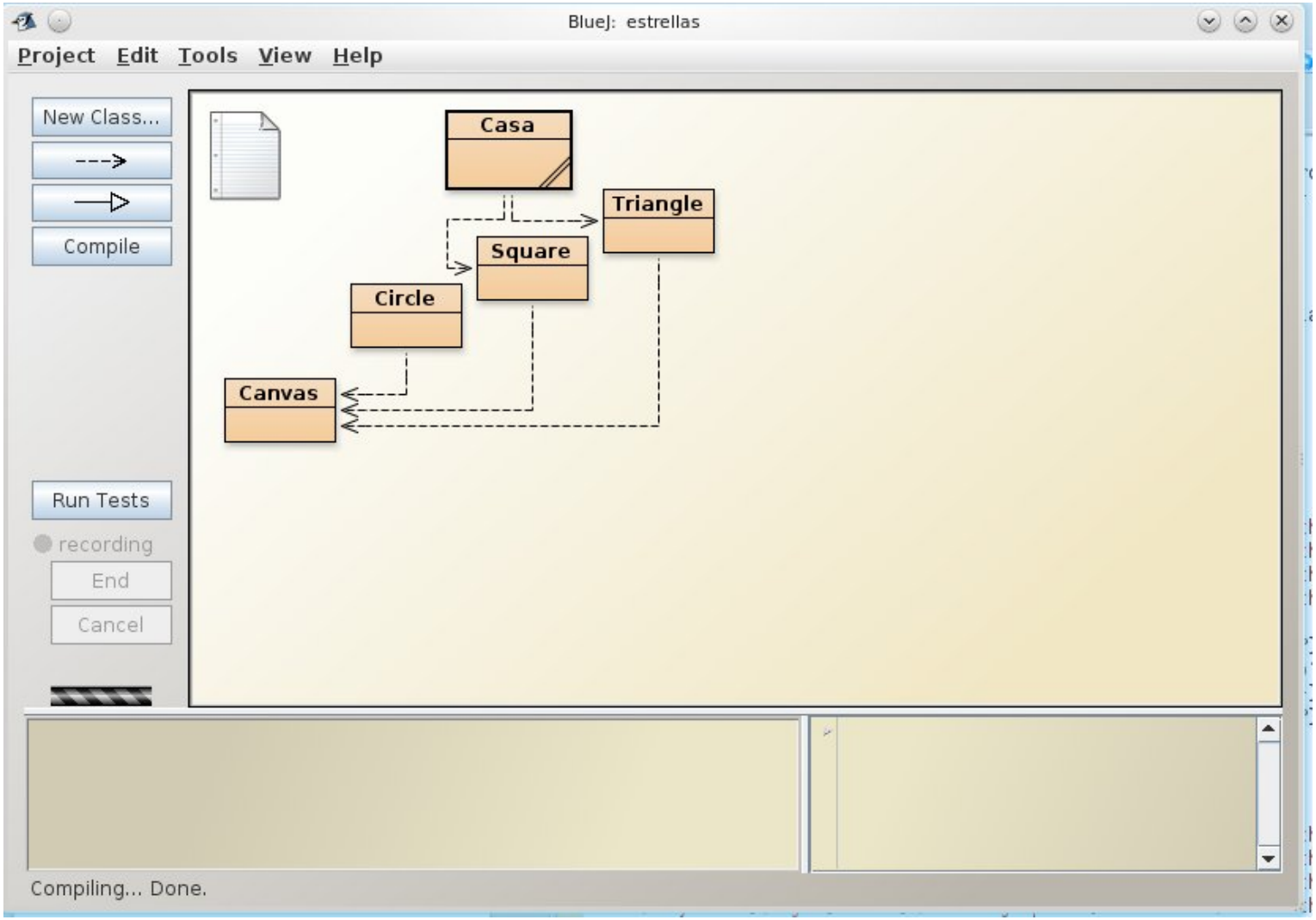
¿Cómo lo hace?

Ejecutar



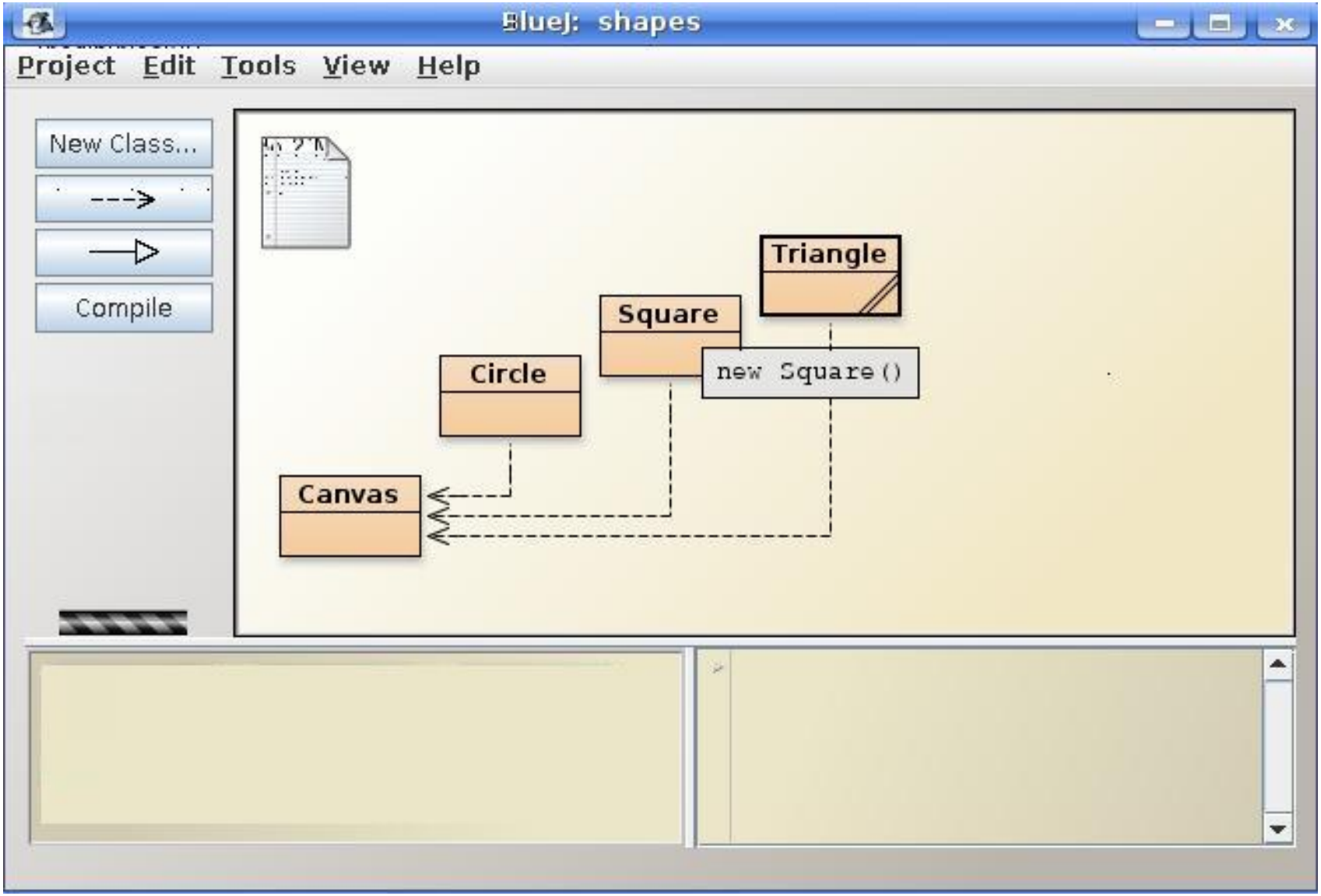
¿Cuál es la herramienta básica JDK?

Ejecutar



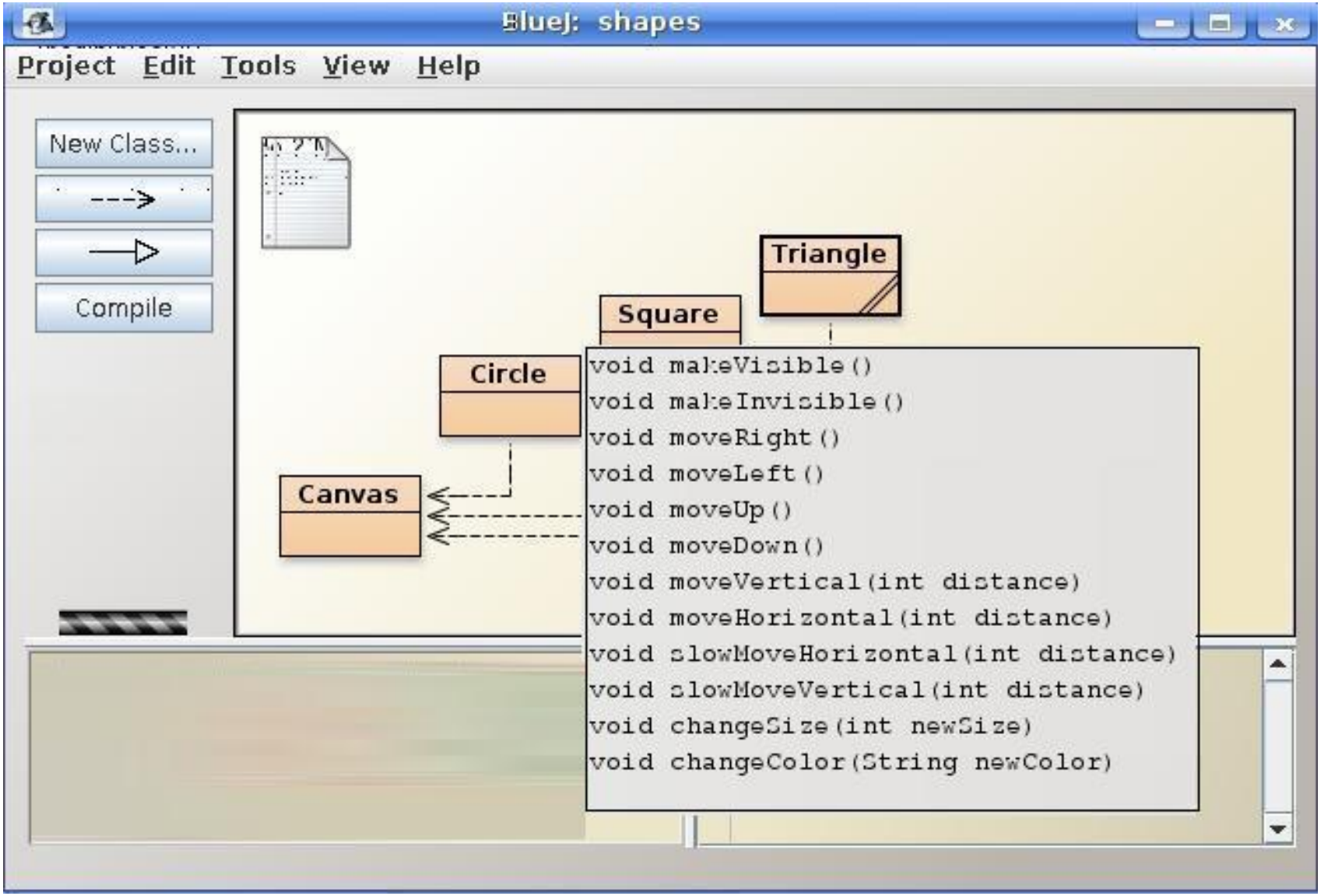
¿A quiénes podemos ejecutar?

Ejecutar



En Square, ¿Cuántos métodos de clase?

Ejecutar



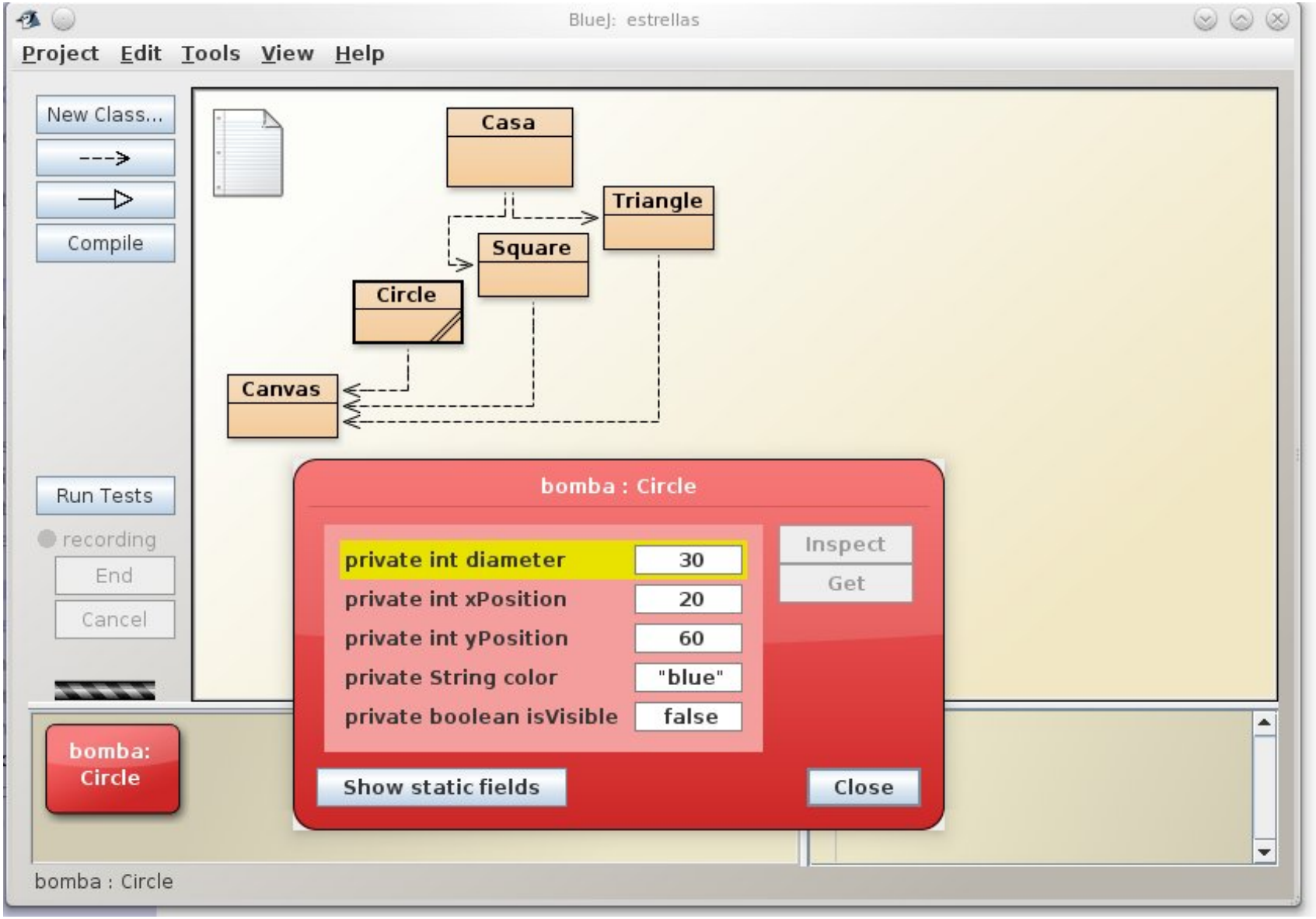
En Square, ¿Cuántos métodos de objeto?

Ejecutar

```java
import java.awt.*;
import java.awt.geom.*;

/**
 * A circle that can be manipulated and that draws itself on a canvas.
 *
 * @author  Michael Kolling and David J. Barnes
 * @version 1.0.  (15 July 2000)
 */

public class Circle{

    public static final double PI=3.1416;

    private int diameter;
    private int xPosition;
    private int yPosition;
    private String color;
    private boolean isVisible;

    /**
     * Create a new circle at default position with default color.
     */
    public Circle(){
        diameter = 30;
        xPosition = 20;
        yPosition = 15;
        color = "blue";
        isVisible = false;
    }

    /**
```

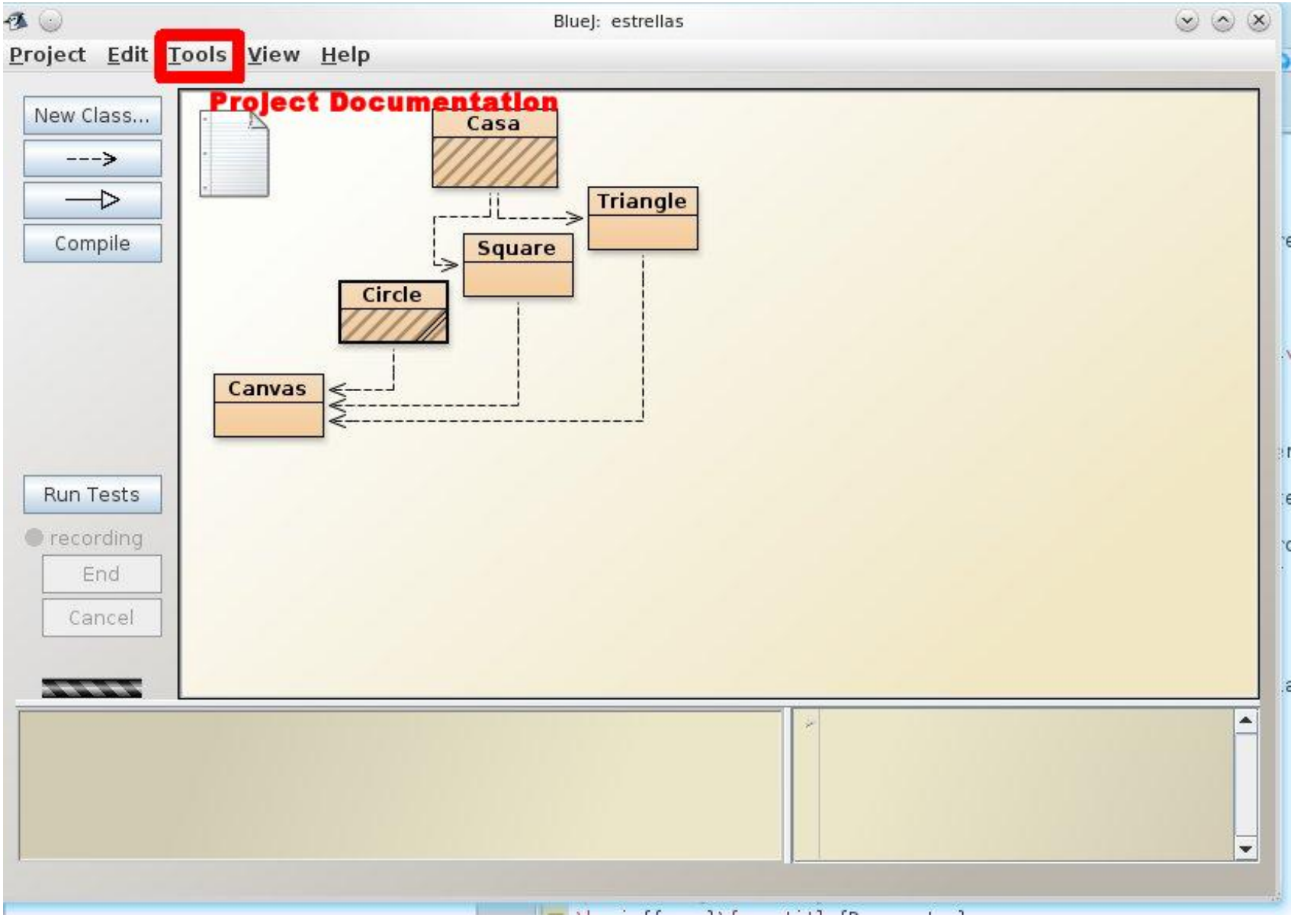
En Circle, ¿Cuáles atributos de clase?

Ejecutar



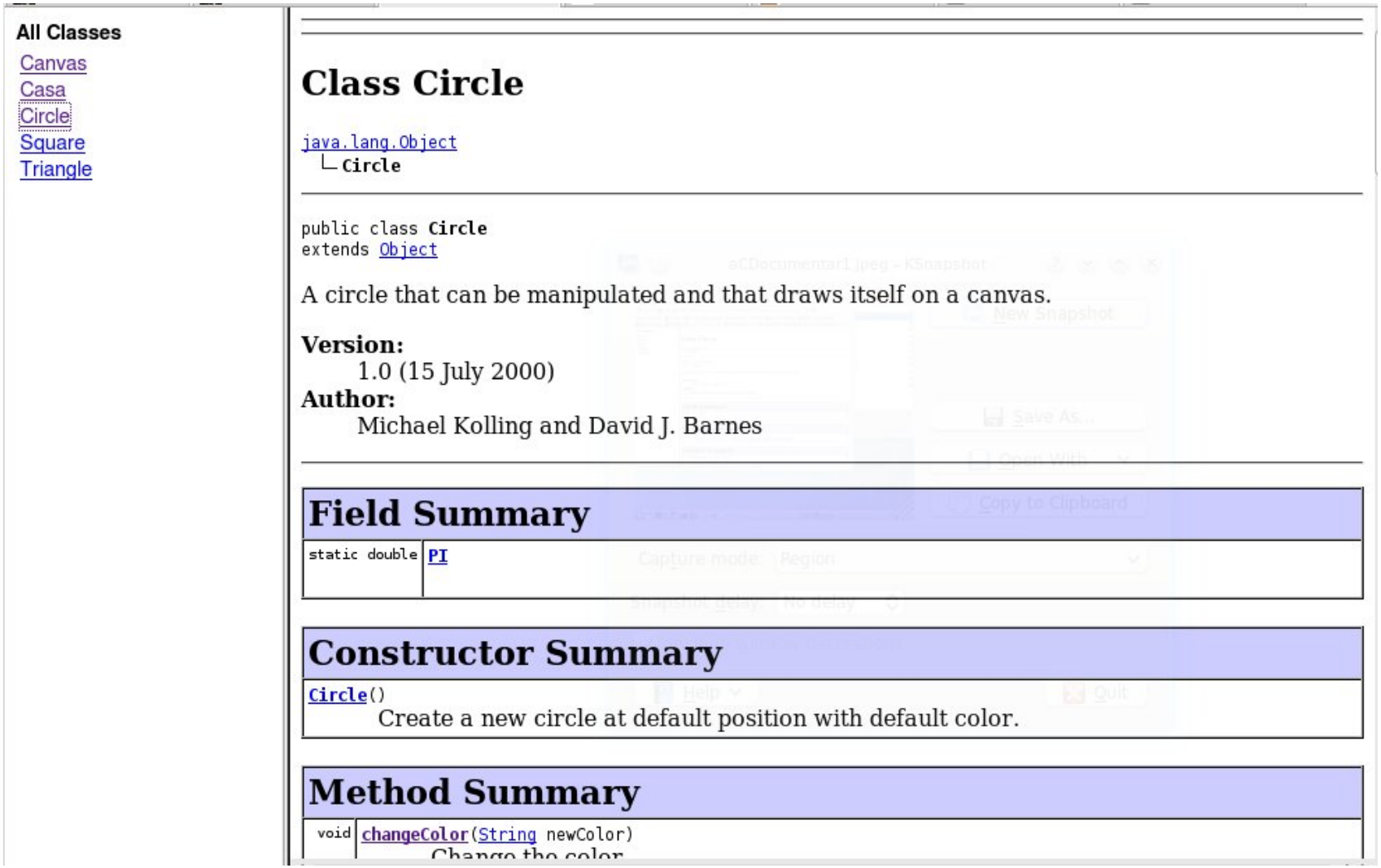
En Circle, ¿Cuáles atributos de objeto?

Documentar



¿Cuál es la herramienta básica JDK?

Documentar



All Classes

Canvas
Casa
Circle
Square
Triangle

Class Circle

java.lang.Object
  └ Circle

public class **Circle**
extends Object

A circle that can be manipulated and that draws itself on a canvas.

Version:
1.0 (15 July 2000)
Author:
Michael Kolling and David J. Barnes

Field Summary

static double	PI

Constructor Summary

Circle() Create a new circle at default position with default color.

Method Summary

void	changeColor(String newColor) Change the color

¿Qué se ve? ¿Qué no se vió?

Documentar

```
import java.awt.*;
import java.awt.geom.*;

/**
 * A circle that can be manipulated and that draws itself on a canvas.
 *
 * @author  Michael Kolling and David J. Barnes
 * @version 1.0  (15 July 2000)
 */

public class Circle {
    public static final double PI=3.1416;

    private int diameter;
    private int xPosition;
    private int yPosition;
    private String color;
    private boolean isVisible;

    /**
     * Create a new circle at default position with default color.
     */
    public Circle() {
        diameter = 30;
        xPosition = 20;
        yPosition = 60;
        color = "blue";
        isVisible = false;
    }

    /**
     * Make this circle visible. If it was already visible, do nothing.
     */
    public void makeVisible() {
        isVisible = true;
        draw();
```