**Javier Andrés Monjes Solórzano**
**202100081**

# Manual Técnico

**IDE Utilizado:** IntelliJ IDEA

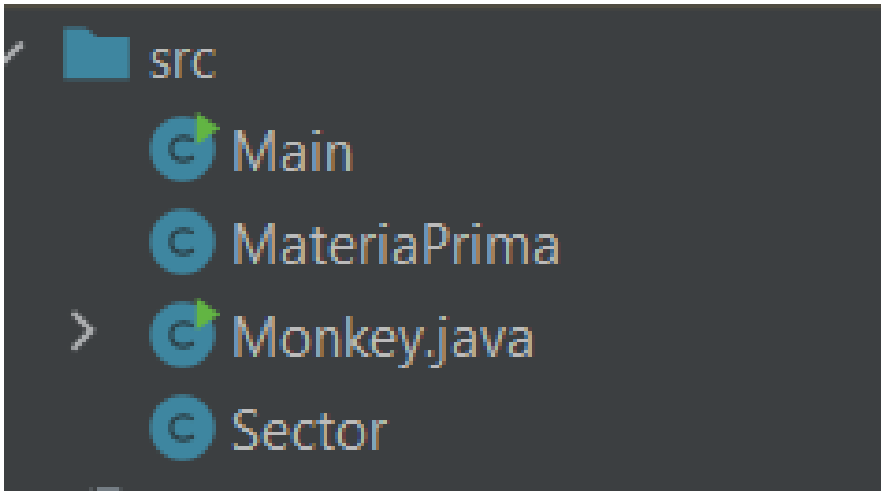
**Versión Java Utilizada:** 19

**Sistema Operativo:** Windows 10

**Requerimiento de la aplicación:** 20 MB espacio libre y mínimo 4 GB de ram

A continuación, se mostrarán partes importantes de código y estructura del proyecto para poder comprender como funciona todo.

**Estructura del proyecto:** El proyecto consta de los siguientes archivos, los cuales se desglosarán más adelante.




Cabe resaltar, que en el Archivo Monkey se encuentran distintas clases las cuales ayudan a poder estructurar de manera adecuada el proyecto.

**Javier Andrés Monjes Solórzano**
**202100081**

**Clase Sector:** Clase que le da vida a los objetos de tipo sector, a continuación, se muestran los atributos de la clase, así como su constructor.

```java
public class Sector {
    private int id;
    private String nombre;
    private int tiempo;
    private int costo;
    private int cantidadMateria;
    private Rectangle2D rectangulo;
    public Sector(int id, String nombre, int tiempo, int costo, int cantidadMateria,Rectangle2D rectangu
        this.id=id;
        this.nombre=nombre;
        this.tiempo=tiempo;
        this.costo=costo;
        this.cantidadMateria=cantidadMateria;
        this.rectangulo=rectangulo;
    }
```

**Clase MateriaPrima:** Clase que le da vida a los objetos de tipo MateriaPrima, a continuación, se muestran los atributos de la clase, así como su constructor.

```java
65 usages
public class MateriaPrima {
    3 usages
    private int id;
    3 usages
    private int sector;
    3 usages
    private boolean procesoTerminado;
    3 usages
    private Ellipse2D circulo;
    30 usages
    public MateriaPrima (int id,int sector, boolean procesoTerminado, Ellipse2D circulo){
        this.id=id;
        this.sector=sector;
        this.procesoTerminado=procesoTerminado;
        this.circulo=circulo;
    }
```

**Javier Andrés Monjes Solórzano**
**202100081**

**Clase Lamina:** Clase que hereda de un Jpanel, se usa para agregar un panel al Jframe principal, este panel va a constar de lo siguiente:

- Labels y Lineas para dibujar:

```
JLabel tituloInventario = new JLabel( text: "Inventario: " + Monkey.inventario.getCantidadMateria());
4 usages
JLabel tituloProduccion = new JLabel( text: "Produccion: " + Monkey.produccion.getCantidadMateria());
4 usages
JLabel tituloEmpaquetado = new JLabel( text: "Empaquetado: " + Monkey.empaquetado.getCantidadMateria()
4 usages
JLabel tituloSalida = new JLabel( text: "Salida: " + Monkey.salida.getCantidadMateria());
4 usages
JLabel tituloEspera = new JLabel( text: "En Espera: " + Monkey.enEspera.getCantidadMateria());
4 usages
JLabel labelTiempo = new JLabel( text: "Segundos: " + Monkey.cronometro);
no usages
Line2D baseLine = new Line2D.Double( x1: 50,  y1: 100,  x2: 250,  y2: 100);
```

- Sectores representados por rectángulos

- Un método que pinta todo el panel

```
public void paintComponent(Graphics g){

    super.paintComponent(g);

    Graphics2D g2=(Graphics2D)g;

    for(Sector sector: sectores){
        if (Objects.equals(sector.getNombre(),  b: "inventario")){
            g2.setColor(Color.CYAN);
            tituloInventario.setBounds( x: 350,  y: 375,  width: 200,  height: 30);
            tituloInventario.setText("Inventario: "+Monkey.inventario.getCantidadMateria());
            add(tituloInventario);
            g2.fill(sector.getRectangulo());
```

**Javier Andrés Monjes Solórzano**
**202100081**

**Clase MarcoConfiguracion:** Clase que hereda de un Jframe, este Jframe se encarga de mostrarle al usuario la vista de configuración donde se dejan establecidos los parámetros de la simulación.

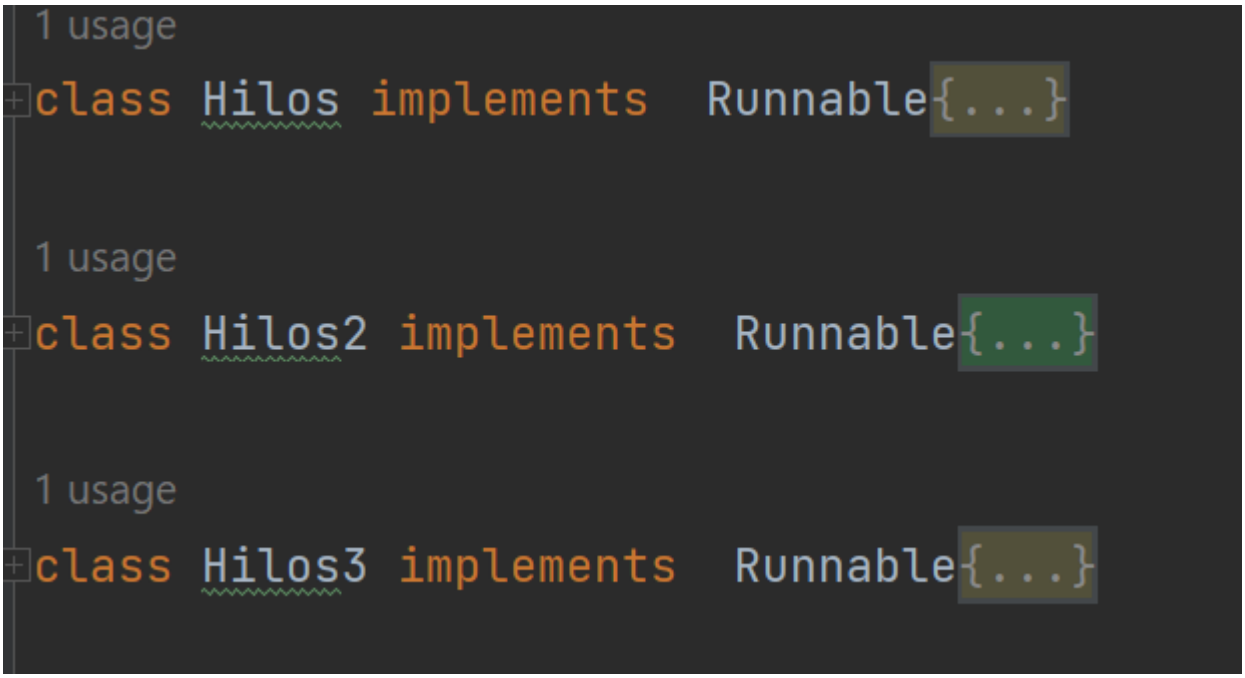
```
class marcoConfiguracion extends JFrame{

    1 usage
    public  marcoConfiguracion(){
        setBounds( x: 0, y: 0, width: 600, height: 600);
        setTitle ("Monkey");
        setLayout(null);


        JLabel tituloEmpresa = new JLabel( text: "MONKEY");
        tituloEmpresa.setLayout(null);
        tituloEmpresa.setVisible(true);
        tituloEmpresa.setForeground(Color.BLACK);
```

**Clase MarcoSimulación:** Clase que hereda de un Jframe, este Jframe se encarga de mostrarle al usuario de manera gráfica la simulación del proceso, en este Frame se agrega la lámina donde contiene todas las figuras, esa lámina es una instancia de la clase Lámina antes mencionada.

```
class marcoSimulacion extends JFrame{
    4 usages
    static Lamina laminaSectores=new Lamina();

    1 usage
    public marcoSimulacion(){
        setBounds( x: 0, y: 0, width: 1000, height: 600);
        setTitle ("Monkey");
        laminaSectores.crarLabels();
        laminaSectores.crearSectores();
        add(laminaSectores, BorderLayout.CENTER);

        JPanel laminaBotones=new JPanel();

        ponerBoton(laminaBotones, titulo: "Iniciar Simulación!", new ActionListener(){
```

**Javier Andrés Monjes Solórzano**
**202100081**

**Implementación de hilos:** Los hilos hacen que las aplicaciones puedan tener múltiples tareas y procesos a la vez, por lo que para la solución de esta práctica, fue muy importante la utilización de hilos, para esto se crearon 3 clases que implementan la interfaz Ruunable, estas son:




**Hilos1:** Este hilo lo que hace es manejar el proceso de las materias, en resumen por cada materia le va definiendo los tiempos de cada parada y los de movimiento, a continuación se adjuntan capturas del código comentado.

**Javier Andrés Monjes Solórzano**
**202100081**

```
public void run(){
    if(Monkey.conteo<30){
        if(Monkey.conteo>=0){
            System.out.println(Monkey.conteo);
            // Restamos 1 al contador de materias en espera:
            Monkey.enEspera.setCantidadMateria(Monkey.enEspera.getCantidadMateria()-1);
            //elComponente.paint(elComponente.getGraphics());
            elComponente.repaint();
            // ============================================Zona Invetario=======================
            // Agregamos 1 al contador del sector inventario
            Monkey.inventario.setCantidadMateria(Monkey.inventario.getCantidadMateria()+1);
            // Cambiamos las coordenadas de la materia para que entre a inventario
            materia.setCirculo(new Ellipse2D.Double(Monkey.matrizCoordenadas[materia.getId()-1][0],M
            // Repintamos
            //elComponente.paint(elComponente.getGraphics());
```

```
    try {
        Thread.sleep( millis: Monkey.inventario.getTiempo()* 1000L);
    } catch (InterruptedException e) {
        throw new RuntimeException(e);
    }
    // Repintamos
    //elComponente.paint(elComponente.getGraphics());
    elComponente.repaint();
    // restamos 1 al contador del sector inventario
    Monkey.inventario.setCantidadMateria(Monkey.inventario.getCantidadMateria()-1);
    // ============================================Zona Producción=======================
    // Agregamos 1 al contador del sector producción
    Monkey.produccion.setCantidadMateria(Monkey.produccion.getCantidadMateria()+1);
    // Cambiamos las coordenadas de la materia para que entre a producción
    materia.setCirculo(new Ellipse2D.Double(Monkey.matrizCoordenadas[materia.getId()-1][2],M
    // Repintamos
```

```
    elComponente.repaint();
    // Espera la materia al tiempo indicado
    try {
        Thread.sleep( millis: Monkey.produccion.getTiempo()* 1000L);
    } catch (InterruptedException e) {
        throw new RuntimeException(e);
    }
    // Repintamos
    //elComponente.paint(elComponente.getGraphics());
    elComponente.repaint();
    // restamos 1 al contador del sector produccion
    Monkey.produccion.setCantidadMateria(Monkey.produccion.getCantidadMateria()-1);

    // ============================================Zona Empaquetado=====================
    // Agregamos 1 al contador del sector empaquetado
    Monkey.empaquetado.setCantidadMateria(Monkey.empaquetado.getCantidadMateria()+1);
    // Cambiamos las coordenadas de la materia para que entre a empaquetado
```

**Javier Andrés Monjes Solórzano**
**202100081**

```
Monkey.empaquetado.setCantidadMateria(Monkey.empaquetado.getCantidadMateri
// Cambiamos las coordenadas de la materia para que entre a empaquetado
materia.setCirculo(new Ellipse2D.Double(Monkey.matrizCoordenadas[materia.getId()-1][4],M
// Repintamos
//elComponente.paint(elComponente.getGraphics());
elComponente.repaint();
// Espera la materia al tiempo indicado
try {
    Thread.sleep( millis: Monkey.empaquetado.getTiempo()* 1000L);
} catch (InterruptedException e) {
    throw new RuntimeException(e);
}
// Repintamos
//elComponente.paint(elComponente.getGraphics());
elComponente.repaint();
// restamos 1 al contador del sector empaquetado
Monkey.empaquetado.setCantidadMateria(Monkey.empaquetado.getCantidadMateria()-1);
```

```
// ================================================Zona Salida==============
// Agregamos 1 al contador del sector salida
Monkey.salida.setCantidadMateria(Monkey.salida.getCantidadMateria()+1);
// Cambiamos las coordenadas de la materia para que entre a salida
materia.setCirculo(new Ellipse2D.Double(Monkey.matrizCoordenadas[materia.getId()-1][6],
// Repintamos
//elComponente.paint(elComponente.getGraphics());
elComponente.repaint();
// Espera la materia al tiempo indicado
try {
    Thread.sleep( millis: Monkey.salida.getTiempo()* 1000L);
} catch (InterruptedException e) {
    throw new RuntimeException(e);
}
// Repintamos
//elComponente.paint(elComponente.getGraphics());
elComponente.repaint();
```

```
//elComponente.paint(elComponente.getGraphics());
elComponente.repaint();

// restamos 1 al contador del sector salida
Monkey.salida.setCantidadMateria(Monkey.salida.getCantidadMateria()-1);

// Repintamos
//elComponente.paint(elComponente.getGraphics());
elComponente.repaint();
// Se elimina la materia que terminó el proceso
materia.setSector(-1);
// Repintamos
//elComponente.paint(elComponente.getGraphics());
elComponente.repaint();
Monkey.cantidadTerminadas++;
```

**Javier Andrés Monjes Solórzano**
**202100081**

**Hilos3:** Este hilo lo que hace es manejar el proceso del cronómetro, por cada segundo aumenta el contador de segundos, y se repinta el panel para que se actualice la vista, su método run que ejecuta el Hilo es el siguiente:

```
class Hilos3 implements  Runnable{
    2 usages
    private Component elComponente;
    1 usage
    public Hilos3(Component elComponente){
        this.elComponente=elComponente;

    }

    public void run(){
        while(!Monkey.terminoProceso){
            try {
                Thread.sleep( millis: 1000);
            } catch (InterruptedException e) {
                throw new RuntimeException(e);
            }
```

```
            } catch (InterruptedException e) {
                throw new RuntimeException(e);
            }
            Monkey.cronometro++;
            elComponente.repaint();
        }

    }

}
```

**Javier Andrés Monjes Solórzano**
**202100081**

**Hilos2:** Este hilo lo dejé de último ya que es la mente maestra, es el encargado de implementar tanto el hilo1 como el hilo3, su método run es el siguiente:

```
class Hilos2 implements  Runnable{
    3 usages
    private Component elComponente;
    1 usage
    public Hilos2(Component elComponente){
        this.elComponente=elComponente;

    }

    public void run(){
        try {
            Thread.sleep( millis: 5000);
        } catch (InterruptedException e) {
            throw new RuntimeException(e);
        }
        Runnable runCronometro=new Hilos3(elComponente);
```

```
    }
    Runnable runCronometro=new Hilos3(elComponente);
    Thread hiloCronometro=new Thread(runCronometro);
    hiloCronometro.start();
    for (int i=0;i<30;i++){
        if(Monkey.conteo<30){
            try {
                Thread.sleep( millis: 2000);
            } catch (InterruptedException e) {
                throw new RuntimeException(e);
            }
            Monkey.conteo++;
            Runnable r=new Hilos(elComponente,Monkey.materias.get(Monkey.conteo));
            Thread t=new Thread(r);
            t.start();
        }
```

**Javier Andrés Monjes Solórzano**
**202100081**

```
        }

    }
    boolean yaTermino=false;
    while(!yaTermino){
        System.out.println(Monkey.cantidadTerminadas);
        if (Monkey.cantidadTerminadas==30) {
            yaTermino=true;
        }
    }
    Monkey.terminoProceso=true;
    JOptionPane.showMessageDialog( parentComponent: null, message: "<html><p style=\"color:blue; font:


}

}
```

## Librerías utilizadas:

```
import javax.swing.*;
import java.awt.*;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.awt.event.MouseAdapter;
import java.awt.event.MouseEvent;
import java.awt.geom.Ellipse2D;
import java.awt.geom.Line2D;
import java.awt.geom.Path2D;
import java.awt.geom.Rectangle2D;
import java.util.ArrayList;
import java.util.Objects;
import java.util.Scanner;
```