
	VICERRECTORADO DOCENTE	Código: GUIA-PRL-001
	CONSEJO ACADÉMICO	Aprobación: 2016/04/06
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		

Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación

Universidad Politécnica Salesiana

Vicerrectorado Docente

Código del Formato:	GUIA-PRL-001
Versión:	VF1.0
Elaborado por:	Directores de Área del Conocimiento Integrantes Consejo Académico
Fecha de elaboración:	2016/04/01
Revisado por:	Consejo Académico
Fecha de revisión:	2016/04/06
Aprobado por:	Lauro Fernando Pesántez Avilés Vicerrector Docente
Fecha de aprobación:	2016/14/06
Nivel de confidencialidad:	Interno

	VICERRECTORADO DOCENTE	Código: GUIA-PRL-001
	CONSEJO ACADÉMICO	Aprobación: 2016/04/06
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		

Descripción General

Propósito


El propósito del presente documento es definir un estándar para elaborar documentación de guías de práctica de laboratorio, talleres o centros de simulación de las Carreras de la Universidad Politécnica Salesiana, con la finalidad de lograr una homogenización en la presentación de la información por parte del personal académico y técnico docente.


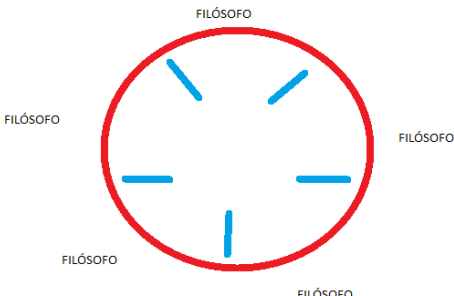
Alcance


El presente estándar será aplicado a toda la documentación referente a informes de prácticas de laboratorio, talleres o centros de simulación de las Carreras de la Universidad Politécnica Salesiana.

Formatos

- Formato de Guía de Práctica de Laboratorio / Talleres / Centros de Simulación – para Docentes
- Formato de Informe de Práctica de Laboratorio / Talleres / Centros de Simulación – para Estudiantes

	VICERRECTORADO DOCENTE	Código: GUIA-PRL-001
	CONSEJO ACADÉMICO	Aprobación: 2016/04/06
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		


		FORMATO DE GUÍA DE PRÁCTICA DE LABORATORIO / TALLERES / CENTROS DE SIMULACIÓN – PARA DOCENTES	
CARRERA: COMPUTACIÓN		ASIGNATURA: Programación Aplicada	
NRO. PRÁCTICA:	1	TÍTULO PRÁCTICA: Hilos en Java	
OBJETIVO: Identificar los cambios importantes de Java Diseñar e Implementar las nuevas técnicas de programación concurrente Entender cada una de las características de Thread en Java.			
INSTRUCCIONES (Detallar las instrucciones que se dará al estudiante):	1. Revisar los conceptos fundamentales de Thread en Java		
	2. Establecer como implementar Thread en Java		
	3. Implementar y diseñar los nuevos componentes de concurrencia		
	4. Realizar el informe respectivo según los datos solicitados.		
ACTIVIDADES POR DESARROLLAR (Anotar las actividades que deberá seguir el estudiante para el cumplimiento de la práctica)			
1. Revisar la teoría y conceptos de Thread en Java			
2. Diseñar e implementar las características de Java para generar una simulación 2D del siguiente enunciado:			
Problema del Filósofo: En una mesa hay procesos que simulan el comportamiento de unos filósofos que intentan comer de un plato. Cada filósofo tiene un cubierto a su izquierda y uno a su derecha y para poder comer tiene que conseguir los dos. Si lo consigue, mostrará un mensaje en pantalla que indique «Filósofo 2 (numero) comiendo». Después de comer, soltará los cubiertos y esperará al azar un tiempo entre 1000 y 5000 milisegundos, indicando por pantalla «El filósofo 2 está pensando».			
En general todos los objetos de la clase Filósofo está en un bucle infinito dedicándose a comer y a pensar.			
Simular este problema en un programa Java que muestre el progreso de todos sin caer en problemas de sincronización a través de un método grafico.			
			


	VICERRECTORADO DOCENTE	Código: GUIA-PRL-001
	CONSEJO ACADÉMICO	Aprobación: 2016/04/06
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		

3. Probar y modificar el método para que nos permita cambiar el numero de filósofos.
4. Realizar práctica codificando con las nuevas características de Java, patrones de diseño, Thread, etc.
5. Fecha de Entrega: 11 Enero del 2021 23:55
RESULTADO(S) OBTENIDO(S): Realizar procesos de Hilos en Java. Entender las aplicaciones de codificación de las nuevas características de concurrencia. Entender las funcionalidades de sincronización y manejo de grupo de Thread dentro de Java.
CONCLUSIONES: Aprenden a trabajar en grupo dentro de plazos de tiempo establecidos, manejando el lenguaje de programación de Java.
RECOMENDACIONES: Realizar el trabajo dentro del tiempo establecido.

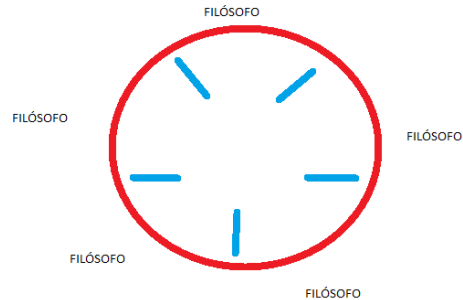
Docente / Técnico Docente: _____

Firma: _____

	FORMATO DE INFORME DE PRÁCTICA DE LABORATORIO / TALLERES / CENTROS DE SIMULACIÓN – PARA ESTUDIANTES	
CARRERA:		ASIGNATURA:
NRO. PRÁCTICA:		TÍTULO PRÁCTICA:
OBJETIVO ALCANZADO:		
ACTIVIDADES DESARROLLADAS		
1. Revisar la teoría y conceptos de Thread en Java		
2. Diseñar e implementar las características de Java para generar una simulación 2D del siguiente enunciado: Problema del Filósofo: En una mesa hay procesos que simulan el comportamiento de unos filósofos que intentan comer de un plato. Cada filósofo tiene un cubierto a su izquierda y uno a su derecha y para poder comer tiene que conseguir los dos. Si lo consigue, mostrará un mensaje en pantalla que indique «Filósofo 2 (numero) comiendo». Después de comer, soltará los cubiertos y esperará al azar un tiempo entre 1000 y 5000 milisegundos, indicando por pantalla «El filósofo 2 está pensando». En general todos los objetos de la clase Filósofo están en un bucle infinito dedicándose a comer y a pensar.		

	VICERRECTORADO DOCENTE	Código: GUIA-PRL-001
	CONSEJO ACADÉMICO	Aprobación: 2016/04/06
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		

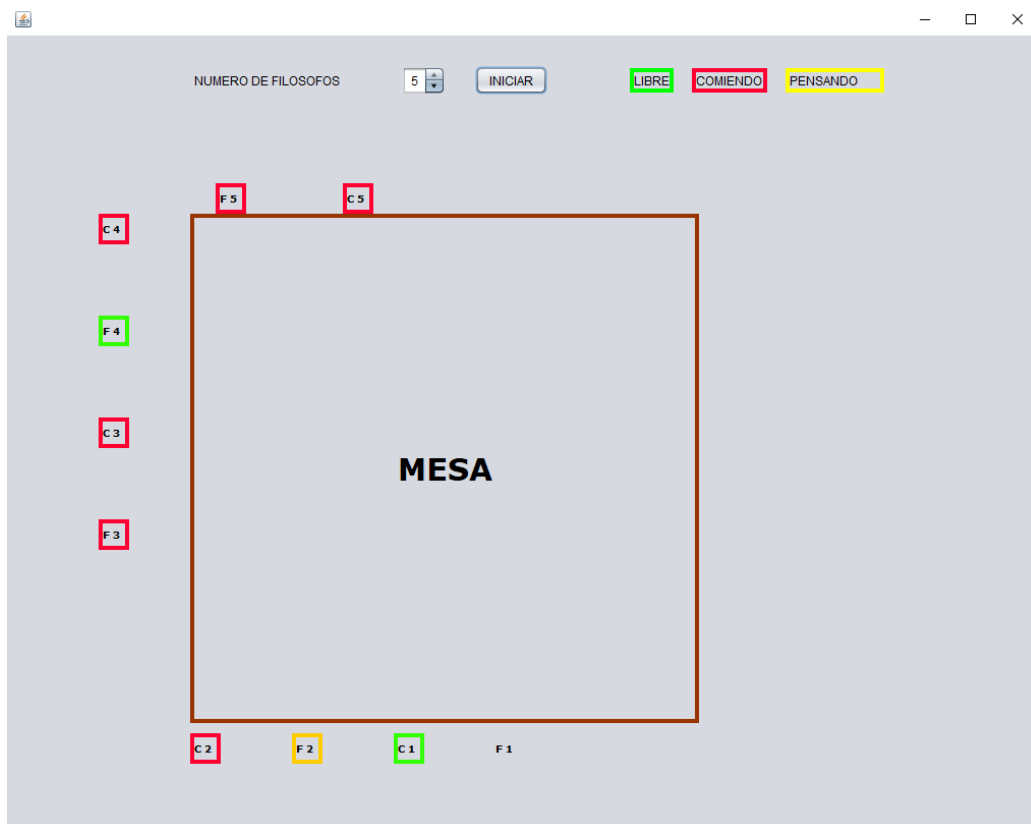
Simular este problema en un programa Java que muestre el progreso de todos sin caer en problemas de sincronización a través de un método gráfico.



3. Probar y modificar el método para que nos permita cambiar el número de filósofos.

RESULTADO(S) OBTENIDO(S):

Vista:



En esta vista podemos encontrar un JSpinner donde podemos seleccionar el numero de filósofos, con un mínimo de dos, además tenemos una nomenclatura de como nuestros filósofos y cubiertos se van a encontrar según su color.

Código Hilo Filósofo.

```
package practicahilos;

import java.awt.Color;
import java.util.List;
import java.util.Random;
import javax.swing.JLabel;


/**
 *
 * @author Paul Idrovo
 */
public class FilosofoHilo extends Thread {

    private final int numero;
    private int segundoCubierto;
    private boolean estado;
    private boolean pensando;
    private List<Cubierto> cubiertos;
    private List<JLabel> imagenesCubiertos;
    private List<JLabel> imagenesFilosofos;

    public FilosofoHilo(int numero, List<Cubierto> cubiertos, List<JLabel> imagenesCubiertos, List<JLabel>
imagenesFilosofos, int segundoCubierto) {
        this.numero = numero;
        this.segundoCubierto = segundoCubierto;
        this.estado = true;
        this.pensando = true;
        this.cubiertos = cubiertos;
        this.imagenesCubiertos = imagenesCubiertos;
        this.imagenesFilosofos = imagenesFilosofos;
    }

    @Override
    public void run() {
        esperarXsegundos();
        while (true) {
            //System.out.println(numero + " - " + (cubiertos.size() - 1));
            if (pensando) {
                if (estado) {
                    if (numero == 0) {
                        estadosCubiertosFilosofos(numero, cubiertos.size() - 1);
                        if (!estado && !cubiertos.get(numero).isEstado() && !cubiertos.get(cubiertos.size() -
1).isEstado()) {
                            esperarXsegundos();
                        }
                    } else {
                        estadosCubiertosFilosofos(numero, numero - 1);
                        if (!estado && !cubiertos.get(numero).isEstado() && !cubiertos.get(numero - 1).isEstado()) {
                            esperarXsegundos();
                        }
                    }
                } else {
                    if (numero == 0) {
                        pensar(numero, cubiertos.size() - 1);
                    } else {
                        pensar(numero, numero - 1);
                    }
                }
            }
        }
    }

    public synchronized void estadosCubiertosFilosofos(int i, int posicion) {
        if (estado && cubiertos.get(i).isEstado() && cubiertos.get(posicion).isEstado()) {
            estado = false;
            cubiertos.get(i).setEstado(false);
            cubiertos.get(posicion).setEstado(false);
            imagenesFilosofos.get(i).setBorder(javax.swing.BorderFactory.createMatteBorder(4, 4, 4, 4, new
java.awt.Color(255, 0, 51)));
            imagenesCubiertos.get(i).setBorder(javax.swing.BorderFactory.createMatteBorder(4, 4, 4, 4, new
java.awt.Color(255, 0, 51)));
            imagenesCubiertos.get(posicion).setBorder(javax.swing.BorderFactory.createMatteBorder(4, 4, 4, 4, new
java.awt.Color(255, 0, 51)));
            //System.out.println(imagenesCubiertos.get(i).getText() + " --- " +
imagenesCubiertos.get(posicion).getText() + " --- " + imagenesFilosofos.get(i).getText());
            System.out.println("FILOSOSOFO " + (i + 1) + " Estado = Comiendo");
        }
    }
}
```

	VICERRECTORADO DOCENTE	Código: GUIA-PRL-001
	CONSEJO ACADÉMICO	Aprobación: 2016/04/06
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		

```

        //esperarXsegundos();
    }
}

public void pensar(int i, int posicion) {
    pensando = false;
    imagenesFilosofos.get(numero).setBorder(javax.swing.BorderFactory.createMatteBorder(4, 4, 4, 4, new
java.awt.Color(255, 204, 0)));
    imagenesCubiertos.get(numero).setBorder(javax.swing.BorderFactory.createMatteBorder(4, 4, 4, 4, new
java.awt.Color(51, 255, 0)));
    imagenesCubiertos.get(posicion).setBorder(javax.swing.BorderFactory.createMatteBorder(4, 4, 4, 4, new
java.awt.Color(51, 255, 0)));
    cubiertos.get(i).setEstado(true);
    cubiertos.get(posicion).setEstado(true);
    System.out.println("FILOSOSO " + (i + 1) + " Estado = PENSANDO ");
    esperarXsegundos();
    pensando = true;
    estado = true;
    imagenesFilosofos.get(numero).setBorder(javax.swing.BorderFactory.createMatteBorder(4, 4, 4, 4, new
java.awt.Color(51, 255, 0)));
}

private void esperarXsegundos() {
    try {
        int tiempo = (int) (Math.random() * 4000) + 1000;
        Thread.sleep(tiempo);
    } catch (InterruptedException ex) {
        Thread.currentThread().interrupt();
    }
}
}

```

Para lograr que no exista problema con los cubiertos se sincronizo el método estadosCubiertosFilosofos, este método se encarga de revisar el estado del filosofo y los cubiertos a su alrededor, dependiendo del estado de estos tres procederá a cambiar su estado y color para que otros filósofos no accedan a los cubiertos. Para esta clase en su constructor se envía tres listas, una de los labels que contienen los filósofos, otra de los labels que contiene los cubiertos y una lista de objeto tipo cubierto.

CONCLUSIONES:

Con esta practica podemos entender la importancia de la sincronización de los hilos, teniendo en cuenta que no siempre podemos utilizar hilos, esto se da porque pueden intentar acceder a la misma información o incluso en otros casos pueden requerir información que aun no ha procesado otro hilo.

Nombre de estudiante: _____

Firma de estudiante: _____