LABORATORIO 4: CONCURRENCIA

Juan Antonio de la Puente - 21/10/15

Objetivos

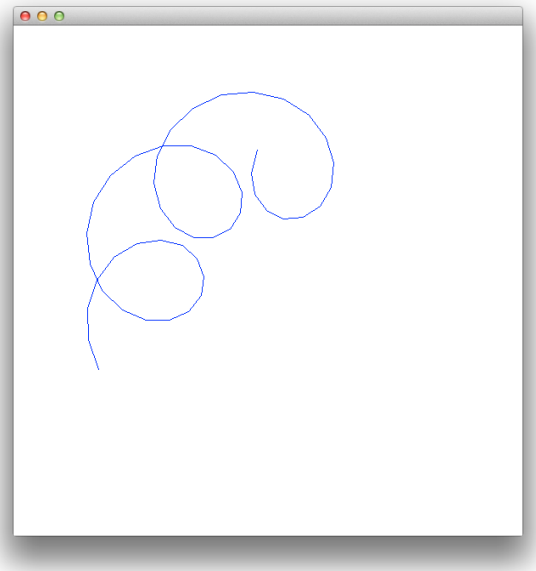
- Entender el comportamiento de los programas concurrentes
- Implementar y modificar un programa concurrente
- Entender los problemas causados por el uso de recursos compartidos
- Programar un monitor para controlar el acceso a un recurso compartido

Advertencia

En este trabajo no se utilizarán las herramientas de corrección automática de código, pero aún así se deberán respetar escrupulosamente las instrucciones que se proporcionan en relación con los nombres de clases y métodos.

Introducción

Para realizar esta práctica partiremos de una clase denominada *Gusano*, que dibuja un «gusano» de una cierta longitud que se mueve a lo largo de un camino definido por una función paramétrica.



La función se define para diferentes valores de un parámetro t (que representa, por ejemplo, el tiempo) de la siguiente forma:

$x = f_x(t)$

$y = f_y(t)$

Por ejemplo,

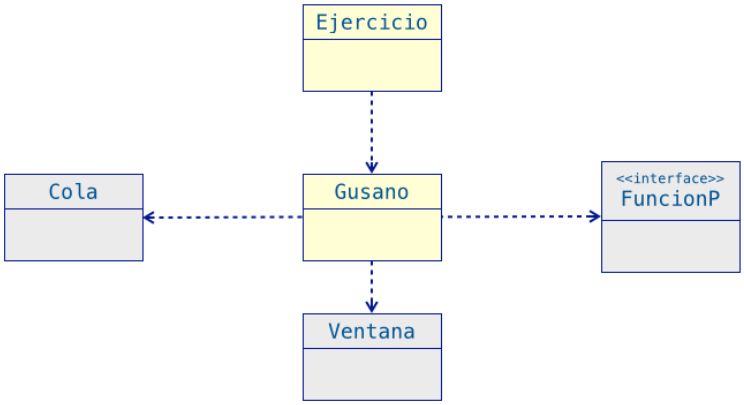
$x = 2 \cos t + \cos 8t$

$y = 2 \text{ sen } t + \text{sen } 8t$

La clase Gusano empieza a trazar la curva cuando se invoca el método

```
Gusano.arranca(desde, hasta, dt)
```

La figura siguiente muestra el diagrama de clases del código de la práctica.



Las clases coloreadas en gris se proporcionan como base para la práctica y no deben modificarse. *FuncionP* es un interfaz que se usa para describir la estructura de una función paramétrica, de la que se proporcionan dos implementaciones, denominadas *Funcion00* y *Funcion01*, respectivamente.

La práctica emplea la clase *GUI*, que realiza la interfaz gráfica. Esta clase permite trazar diversas líneas en una ventana en blanco.

La clase *Ejercicio* contiene el método *main()*, que hay que completar. Inicialmente se proporciona una versión secuencial, que se debe modificar para ejecutar varias hebras que dibujen diferentes curvas a la vez.

Actividades

1. Descarga del proyecto desde GitHub (trabajo previo)

La estructura del proyecto a desarrollar, así como algunas de las clases necesarias se encuentran disponibles en el repositorio de la asignatura en GitHub: https://github.com/ALED-UPM/Tema3.git

Antes del inicio de la sesión de laboratorio, deberá:

1. Clonar el repositorio Tema3.
2. Importar el proyecto *ALED-lab4* al entorno de Eclipse.
3. Examinar las clases *Gusano* y *Ejercicio*.
4. Ejecutar el programa a partir de la clase *Ejercicio*, y comprobar que se obtiene una figura similar a la mostrada en la página anterior.

Conteste a las preguntas 1-4 del cuestionario que se encuentra al final de este documento.

2. Modificar el método `main()` para arrancar dos hebras concurrentes (trabajo en laboratorio)

Se trata de cambiar el método principal de la clase *Ejercicio* para arrancar dos hebras concurrentes que ejecuten sendos ejemplares de la clase *Gusano* con diferentes parámetros.

La primera hebra ejecutará el código siguiente:

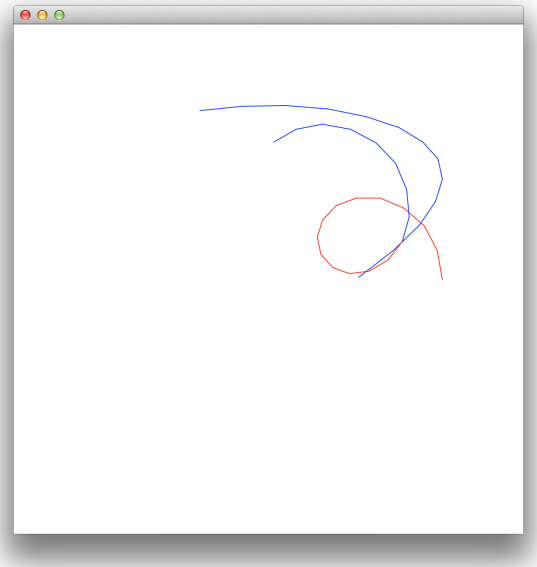
```
FuncionP funcion = new Funcion00();
Gusano gusano = new Gusano(Color.BLUE, funcion, 30);
gusano.arranca(0, 100, 0.05);
```

y la segunda hebra ejecutará:

```
FuncionP funcion = new Funcion01();
Gusano gusano = new Gusano(Color.RED, funcion, 15);
gusano.arranca(10, 30, 0.05);
```

Implemente el código y ejecútelo.

Al comienzo de la ejecución se debe obtener un resultado similar al siguiente (obsérvese la mezcla de colores en un mismo gusano):



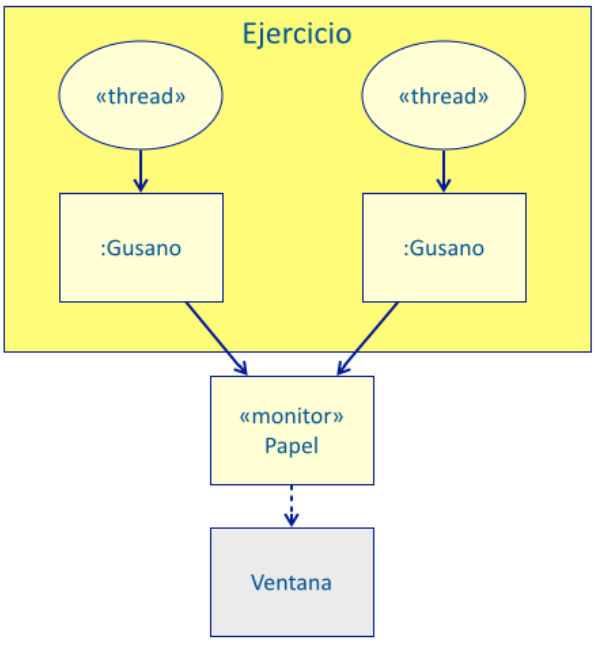
Razone cuál es la causa de este comportamiento erróneo.

Conteste las preguntas 5-8 del cuestionario que se encuentra al final de este documento.

3. Añadir un monitor para evitar las condiciones de carrera en el acceso a la pantalla (trabajo en laboratorio)

La *Ventana* es un objeto compartido por las hebras que dibujan los gusanos, por lo que no es seguro utilizarla tal cual en un entorno concurrente (no es *thread-safe*).

Para asegurar la exclusión mutua en las operaciones gráficas se debe desarrollar un monitor de acuerdo con el diagrama siguiente:



Implemente la clase *Papel* de acuerdo con el siguiente esquema:

```
public final class Papel {
     // Vuelve a dibujar la pantalla
     public ... void pinta() { Ventana.refresca(); }

     // Borra toda la pantalla
     public ... void borra() { Ventana.borra(); }

     // Borra un objeto
     public ... void borra(Object objecto) { Ventama.borra(objecto);    }

     // Fija el color del dibujo
     public ... void ponColor(Color color) { Ventana.ponColor(color);}

     // Dibuja una línea
     public ... Object pinta(double x0, double y0, double x1, double y1)
        {return Ventana.pinta(x0, y0, x1, y1);}
}
```

Modifique la clase *Gusano* de manera que utilice los métodos de la clase *Papel* en vez de los de *Ventana*.

Ejecute el código con estas modificaciones y compruebe que se dibujan las curvas correctamente.

Entregas

a) Durante la sesión de laboratorio

Las **preguntas** planteadas deben **entregarse en papel al final de la sesión**. Las preguntas 1-4 estarán resueltas antes del laboratorio, y el resto podrán resolverse en el laboratorio.

El **código fuente** de la práctica, incluyendo las clases proporcionadas originalmente y la clase *Ejercicio* modificada debe **entregarse en un archivo comprimido** en el servidor *moodle* de la asignatura antes del final de la sesión de laboratorio.

El nombre del archivo comprimido debe ser `ALED-Lab4.zip`. El código fuente debe estar en el directorio relativo `ALED-lab4/src/es/upm/dit/aled/lab4/`.

b) Después de la sesión de laboratorio

El **código fuente completo** de la práctica, incluyendo las clases proporcionadas originalmente, las clases *Ejercicio* y *Gusano* modificadas, y la clase *Papel* desarrollada, debe **entregarse en un archivo comprimido** en el servidor *moodle* de la asignatura antes de las 23:55 del 13 de noviembre de 2015, siguiendo las mismas instrucciones que para la entrega de laboratorio en lo que respecta al nombre y ruta de los ficheros.

La práctica será evaluada por el profesor, y se le proporcionará información sobre los posibles fallos detectados. La nota es orientativa. Se calificará con un máximo de 5 puntos las respuestas al cuestionario y el código entregado en la sesión de laboratorio, y con otros 5 puntos el código de la entrega final. Se considerará la corrección del código, su estructura y legibilidad, y la documentación de los métodos desarrollados.

Para la evaluación de la entrega sólo se considerará el último envío realizado. No se aceptarán entregas fuera de plazo bajo ningún concepto.

AVISO MUY IMPORTANTE

Se recuerda a los alumnos que el trabajo es individual, y que la copia de entregas supondrá el **suspenso en la asignatura de forma automática**, **tanto para quien copia como para quien se deja copiar**.

No está permitido:

- Realizar este trabajo en grupo.
- Copiar el trabajo de otro alumno, ni permitir la copia del propio trabajo, ni siquiera parcialmente.
- Usar código publicado sin citar el origen.

CUESTIONARIO

Responda a las siguientes preguntas según se le indica en el enunciado. Entregue al profesor esta hoja con sus respuestas antes de terminar la sesión.

Nombre:_____ **DNI/NIE**:_____

1. Enumere los métodos que tiene la clase *Gusano*. ¿Qué método dibuja la curva?

2. ¿Qué parámetro de la clase *Gusano* determina la forma de la curva que se dibuja?

3. ¿En qué parte del programa se define la forma concreta de la curva que dibuja *Gusano*?

4. ¿En qué parte del programa hay que crear las hebras para dibujar dos curvas concurrentemente?

5. ¿Por qué se ejecuta incorrectamente el programa con las dos hebras?

6. ¿Cómo se llama la situación que se produce cuando el resultado de la ejecución depende de las velocidades de ejecución de las hebras? ¿Ocurre esto en este programa?

7. ¿Qué modificador hay que añadir a los métodos de la clase *Papel* para que se comporte como un monitor?

8. ¿Qué métodos de la clase *Gusano* hay que modificar para usar *Papel* en vez de *Ventana*?