UNIVERSIDAD SAN CARLOS DE GUATEMALA
CENTRO UNIVERSITARIO DE OCCIDENTE

ESTUDIANTE:
JORGE ANIBAL BRAVO RODRÍGUEZ

CARNÉ:
202131782

AUXILIAR:
ADOLFO SON

CURSO:
COMPILADORES 1 - LABORATORIO

PRACTICA 1

Herramientas de desarrollo

Ultramarine Linux
Ultramarine Linux es un sistema operativo de código abierto basado en Linux que ofrece una plataforma estable y segura para el desarrollo y la ejecución de aplicaciones. Se caracteriza por su equilibrio entre funcionalidades avanzadas y eficiencia, lo que lo convierte en una opción ideal para entornos de tamaño mediano que requieren un sistema operativo confiable y flexible.

Visual Studio Code
Visual Studio Code es un editor de código fuente ligero y altamente personalizable desarrollado por Microsoft. Se destaca por su amplia gama de extensiones y su integración con herramientas de desarrollo populares. En entornos de tamaño mediano, Visual Studio Code proporciona un entorno de desarrollo versátil y eficaz para programadores que buscan una solución ágil y potente.

OpenJDK
Es una implementación de código abierto del lenguaje de programación Java. Es el proyecto de referencia para el desarrollo de Java y la base para muchas otras distribuciones de Java, como las de Oracle.

NetBeans
Es un entorno de desarrollo integrado (IDE) gratuito y de código abierto para desarrollar aplicaciones de software. Es muy popular para el desarrollo en Java, pero también soporta otros lenguajes. Ofrece una amplia gama de herramientas para facilitar la programación, como autocompletado, depuración y refactorización.

Maven
Es una herramienta de gestión de proyectos de software. Se utiliza para automatizar la construcción, la documentación y las pruebas de proyectos Java. Define un formato estándar para describir los proyectos y sus dependencias, lo que facilita la colaboración y la reutilización de código.

JFlex
Es un generador de analizadores léxicos. Se utiliza para crear programas que reconocen patrones en texto, como palabras clave, números y símbolos. Es una herramienta esencial para la construcción de compiladores e intérpretes.

Java CUP
Es un generador de analizadores sintácticos. Se utiliza para crear programas que reconocen la estructura gramatical de un lenguaje de programación. Es complementario a JFlex y juntos se utilizan para construir compiladores e intérpretes. Para esta práctica se utilizó el jar proporcionado por los desarrolladores de cup como dependencia instalada localmente en el sistema a través de maven.

Clase Grafico.java y derivadas
La clase Grafico es una clase abstracta que sirve como tipo de dato y modelo para las clases derivadas Circulo, Linea, Poligono, Rectangulo y Cuadrado. Cuenta con un método abstracto llamado establecerGrafico que recibe la vista de trabajo (un JPanel extendido) para poder interactuar con ella graficando su contenido. el método establecerGrafico se especializa en cada clase que hereda de Grafico para generar la figura que representa.

Las animaciones se basan en el cálculo de coordenadas intermedias de 10 imagenes para podes ser ejecutadas posteriormente por la vista de trabajo. Los graficos se crean a partir de las clases y métodos proporcionados por AWT.

Atributos Clave

- **color:** Define el color del gráfico.
- **animado:** Indica si el gráfico está en animación.
- **lineal:** Determina si la animación sigue una trayectoria lineal o curva.
- **orden:** Se utiliza para establecer un orden de dibujo o renderizado.
- **frameActual, contadorFrames, posxAnimacion, posyAnimacion:** Atributos relacionados con la animación, controlando los frames y las posiciones en cada frame.
- **posxDestino, posyDestino:** Las coordenadas de destino hacia donde se mueve el gráfico durante la animación.
- **posx, posy:** Las coordenadas actuales del gráfico.

Métodos Importantes

- **establecerGrafico(VistaDeTrabajo vista):** Este método abstracto, que debe ser implementado en las subclases, se encarga de inicializar el gráfico en una vista específica.
- **ejecutarAnimacion():** Este método es el motor de la animación. Calcula las nuevas posiciones del gráfico en función de si la animación es lineal o curva y actualiza las coordenadas en cada frame.
- **setAnimado, isAnimado, setLineal, isLineal:** Métodos getters y setters para los atributos animado y lineal.
- **establecerDestino:** Establece las coordenadas de destino para la animación.

Comportamiento

1. **Creación de un gráfico:** Se crea una instancia de una subclase de Grafico y se inicializa con un color y otras propiedades según se necesite.
2. **Establecer la animación:** Se llama a setAnimado(true) para iniciar la animación y a setLineal para especificar el tipo de trayectoria.
3. **Establecer el destino:** Se usa establecerDestino para definir el punto final de la animación.
4. **Ejecutar la animación:** En cada frame, se llama a ejecutarAnimacion para actualizar las coordenadas del gráfico y redibujarlo.

Vista de trabajo

Esta clase de Java, VistaDeTrabajo, se encarga de la parte visual del programa utilizando tanto las clases Lexer, Parser como las clases Graficos.

Atributos:

- graficos: Lista que almacena los objetos gráficos creados.
- ultimo: Referencia al último gráfico agregado (usado para su animacion).
- parser y lexer: Objetos usados para analizar el código fuente que define los gráficos.
- rutaArchivo y nombreArchivo: Almacenan la ruta y nombre del archivo que contiene el código fuente.

Métodos:

- **VistaDeTrabajo(VentanaPrincipal principal):** Constructor por defecto que inicializa la interfaz gráfica y algunas variables.
- **VistaDeTrabajo(VentanaPrincipal principal, String ruta):** Constructor que además de lo anterior, carga el código fuente de un archivo existente.
- **agregarFigura(Grafico grafico):** Agrega un nuevo objeto gráfico a la lista graficos.
- **agregarAnimacionUltimaFigura(boolean lineal, int x, int y, int orden):** Configura la animación del último gráfico agregado.
- **instanciarGraficos():** Analiza el código fuente usando parser y lexer para crear los objetos gráficos definidos.
- **dibujarGraficos():** Recorre la lista graficos y llama a cada objeto para que se dibuje en el panel.
- **ejecutarAnimaciones():** Recorre la lista graficos y ejecuta la animación de cada uno mientras esté activa.
- **exportarPanelAPNG(String rutaArchivo):** Exporta el contenido del panel como una imagen PNG.
- **exportarAPDF(String rutaArchivo):** Exporta el contenido del panel como un archivo PDF usando la librería Apache PDFBox.
- **guardarContenido():** Guarda el código fuente actual en el archivo previamente abierto.
- **obtenerGrafico():** Devuelve el objeto Graphics del panel para que los objetos gráficos lo usen para dibujarse.

Ventana principal

La clase VentanaPrincipal actúa como el **contenedor principal** del programa graficador. Sirve como el punto de entrada visual para la aplicación, proporcionando una interfaz gráfica de usuario (GUI) a través de la cual el usuario interactúa con el programa.

Estructura y Componentes Clave

- **Herencia:** Extiende la clase JFrame, lo que le otorga las características básicas de una ventana, como título, tamaño, posición y capacidad de cerrar.
- **Componentes principales:**
    - JTabbedPane: Un componente que permite organizar el contenido de la ventana en múltiples pestañas. Cada pestaña representa una vista o documento diferente.
    - JPanel: Un contenedor genérico que se utiliza para agrupar otros componentes. Cada pestaña en el JTabbedPane está respaldada por un JPanel.
    - JMenuItem: Elementos de menú que permiten al usuario realizar acciones, como abrir un nuevo documento o abrir un archivo existente.
- **Variables de instancia:**
    - vistas: Una lista enlazada (LinkedList) que almacena referencias a los paneles asociados a cada pestaña. Esto permite realizar un seguimiento de las pestañas abiertas y realizar operaciones sobre ellas.
    - contadorSinTitulo: Un contador entero que se utiliza para generar títulos únicos para las pestañas que se crean sin un nombre específico.

Funcionalidades Principales

- **Creación de la ventana:** El constructor de la clase inicializa la ventana, crea los componentes principales (menú, pestañas) y establece las propiedades iniciales de la ventana.
- **Gestión de pestañas:**
    - **Agregar pestañas:** El método agregarPestana permite añadir una nueva pestaña al JTabbedPane, asociando un panel específico a la pestaña y asignándole un título.
    - **Cerrar pestañas:** El método cerrarPestana elimina una pestaña del JTabbedPane y actualiza la lista de vistas en consecuencia.
- **Manejo de eventos:**
    - La clase implementa los manejadores de eventos para los elementos del menú (por ejemplo, "Nuevo" y "Abrir"). Estos manejadores se encargan de realizar las acciones correspondientes cuando el usuario selecciona un elemento del menú.

Clases de reportes

Son clases que no están relacionadas entre si en términos de herencia. Simplemente sirven como contenedores para contadores de elementos específicos, como colores o errores, y permiten a la clase Parser y Lexer poder establecer un recuento de los datos que se solicita reportar, así como agregar información específica como linea y columna en donde se detectaron, además de retornar un modelo de tabla DefaultTableModel con la información recopilada para ser desplegada después por la clase Reportes.

JFlex

Identificadores

Expresión regular	**Sym retornado**
[a-zA-Z_][a-zA-Z0-9_]*	IDENTIFICADOR

Números: todo se parsea a float

Expresión regular	**Sym retornado**
[0-9]+	NUMERO
[0-9]+ “.” [0-9]+	NUMERO

Símbolos

Expresión regular	**Sym retornado**
+	SUMA
-	RESTA
*	MULTIPLICACION
/	DIVISION
(	PARA
)	PARC
,	COMA

Palabras reservadas

Colores	
Expresión regular	**Sym retornado**
azul	AZUL
rojo	ROJO
amarillo	AMARILLO
verde	VERDE
morado	MORADO
naranja	NARANJA
turquesa	TURQUESA
negro	NEGRO
cafe	CAFE

Animación	
Expresión regular	**Sym retornado**
linea	LINEA
curva	CURVA
animar	ANIMAR
objeto	OBJETO
anterior	ANTERIOR

Gráficos	
Expresión regular	**Sym retornado**
graficar	GRAFICAR
circulo	CIRCULO
cuadrado	CUADRADO
rectangulo	RECTANGULO
linea	LINEA
poligono	POLIGONO

CUP
Definición gramatical
// Números
num ::= ENTERO
num ::= DECIMAL
num ::= num * num
num ::= num / num
num ::= (num)
num ::= num + num
num ::= num - num

//Color
color ::= AZUL | ROJO | AMARILLO | VERDE | MORADO | NARANJA | TURQUESA | NEGRO | CAFE

// Gráficos
graficar ::= GRAFICAR figura

figura ::= CIRCULO (IDENTIFICADOR , num, num, num, color)
figura ::= CUADRADO (IDENTIFICADOR, num, num, num, color)
figura ::= RECTANGULO (IDENTIFICADOR, num, num, num, num, color)
figura ::= LINEA (IDENTIFICADOR, num, num, num, num, color)
figura ::= POLIGONO (IDENTIFICADOR, num, num, num, num, num, color)

Instrucciones de gráficos	
Figura	**Composición**
Círculo	**nombre, posx, posy, radio, color**
Cuadrado	**nombre, posx, posy, tamaño lado, color**
Rectángulo	**nombre, posx, posy, ancho, alto, color**
Línea	**nombre, posx1, posy1, posx2, posy2, color**
Polígono	**nombre, posx, posy, cantidad lados, ancho, alto, color**

// Animar objeto
animar ::= ANIMAR OBJETO ANTERIOR (tipo_animacion, num, num, num)
tipo_animacion::= LINEA
tipo_animacion ::= CURVA

Instrucciones de animación
tipo, destinox, destinoy, orden

Diagrama de clases