Universidad de San Carlos de Guatemala

Facultad de Ingeniería

Escuela de Ciencias y Sistemas

Organización de Lenguajes y Compiladores 1

Ing. Mario Bautista

Aux. Emely García

Aux. José Morán

MANUAL TECNICO FIUSAC COPY ANALYZER

Marvin Eduardo Catalán Véliz

201905554

Guatemala, 4 de sep. de 21

Índice

REQUERIMIENTO TÉCNICO PARA EJECUCION 3
HERRAMIENTAS UTILIZADAS PARA EL DESARROLLO 3
JAVA 3
¿Qué es? 3
Versión 3
IDE 3
¿Qué es? 3
IDE utilizado: NetBeans 3
LAPTOP 4
Librerías 4
Cup y jflex: 4
Jfreechart 4
Funcionalidad y explicación 5
Interfaz: 5
OLC1.CopyAnalizer: 5
Analyzers y AnalyzersJavascript: 5
- GeneratorJs/Generador 6
- ParserJs.cup/Parser.cup 6
- Lexjs.lex/Lex.lex 6

Lexjs.lex/Lex.lex 7
Parserjs.cup/Parser.cup 7
Generatorjs/Generator 7
LógicaFCA: 8
Repitenciametodo.java y Repitenciaclase.java 8
Variablesglobales.java 8
Logicajs/Logica: 9

REQUERIMIENTO TÉCNICO PARA EJECUCION

- Máquina virtual de Java

HERRAMIENTAS UTILIZADAS PARA EL DESARROLLO

JAVA

¿Qué es?

Java es un lenguaje de programación y una plataforma informática comercializada por primera vez en 1995 por Sun Microsystems. Hay muchas aplicaciones y sitios web que no funcionarán a menos que tenga Java instalado y cada día se crean más. Java es rápido, seguro y fiable. Desde portátiles hasta centros de datos, desde consolas para juegos hasta súper computadoras, desde teléfonos móviles hasta Internet, Java está en todas partes.

Versión

```
java version "16.0.2" 2021-07-20
Java(TM) SE Runtime Environment (build 16.0.2+7-67)
Java HotSpot(TM) 64-Bit Server VM (build 16.0.2+7-67, mixed mode, sharing)
```

IDE

¿Qué es?

en inglés Integrated Development Environment (IDE), es una aplicación informática que proporciona servicios integrales para facilitarle
al desarrollador o programador el desarrollo de software.

Normalmente, un IDE consiste de un editor de código fuente, herramientas de construcción automáticas y un depurador.

IDE utilizado: NetBeans

Product Version: Apache NetBeans IDE 12.4
Java: 16.0.2; Java HotSpot(TM) 64-Bit Server VM 16.0.2+7-67
Runtime: Java(TM) SE Runtime Environment 16.0.2+7-67
System: Windows 10 version 10.0 running on amd64; Cp1252; es_GT (nb)
User directory: C:\Users\marvi\AppData\Roaming\NetBeans\12.4
Cache directory: C:\Users\marvi\AppData\Local\NetBeans\Cache\12.4

LAPTOP

Especificaciones del dispositivo

Nombre del dispositivo	DESKTOP-KMQA5BN
Procesador	Intel(R) Core(TM) i5-7200U CPU @ 2.50GHz 2.71 GHz
RAM instalada	12.0 GB (11.9 GB utilizable)
Id. del dispositivo	369A16F7-DB15-4299-B4CD-780C88B080B8
Id. del producto	00325-96001-12028-AAOEM
Tipo de sistema	Sistema operativo de 64 bits, procesador x64
Lápiz y entrada táctil	Compatibilidad con entrada táctil con 10 puntos táctiles

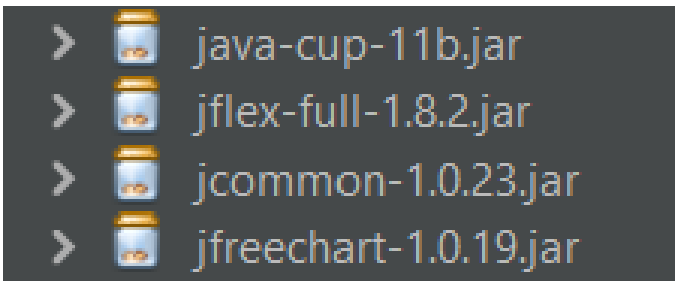
Librerías

Cup y jflex:

Se tiene por un lado un generador de analizadores léxicos (JFlex) y por otro lado un generador de analizadores sintácticos (CUP), se pueden integrar cada uno de los procesos de dichos metacompiladores para de esta forma obtener un compilador mucho mas completo y personalizado, esto ahorra de manera abismal la generación de código al momento de programar un compilador.

Jfreechart

JfreeChart es una biblioteca de código abierto desarrollada en Java. Se puede utilizar en aplicaciones basadas en Java para crear una amplia variedad de gráficos. Con JFreeChart podemos crear todos los tipos principales de graficos 2D y 3D,como gráficos circulares,histogramas,graficos de lineasre,grafico XY y grafico 3D.



Funcionalidad y explicación

El programa está basado en su mayoría utilizando programación orientada a objetos, o al menos es lo que se intentó plasmar para poder tener un código, optimo y organizado. El programa está dividido en distintos paquetes los cuales se describen a continuación:

Interfaz:

Este paquete se utiliza para crear el Jframe donde se editarán los códigos que se analizarán en el programa de Copy analizar, se generó automáticamente la mayoría del código de este paquete gracias a la facilidad del ide netbeans para crear un antifaz gráfico solamente arrastrando componentes.

OLC1.CopyAnalizer:

Este paquete esta creado única y exclusivamente para el manejo de la lógica que hay detrás del editor de textos de la interfaz gráfica como tal, se utilizan distintas clases de tipo públicas para que funcionen las opciones básicas de un editor que son guardar, abrir y guardar como. De la misma manera se utiliza otra clase llamada archivo que maneja la lógica de el almacenamiento como tal de archivos obtenidos del texto de la interfaz. Estas clases se encuentran perfectamente comentadas en el código fuente.

Analyzers y AnalyzersJavascript:

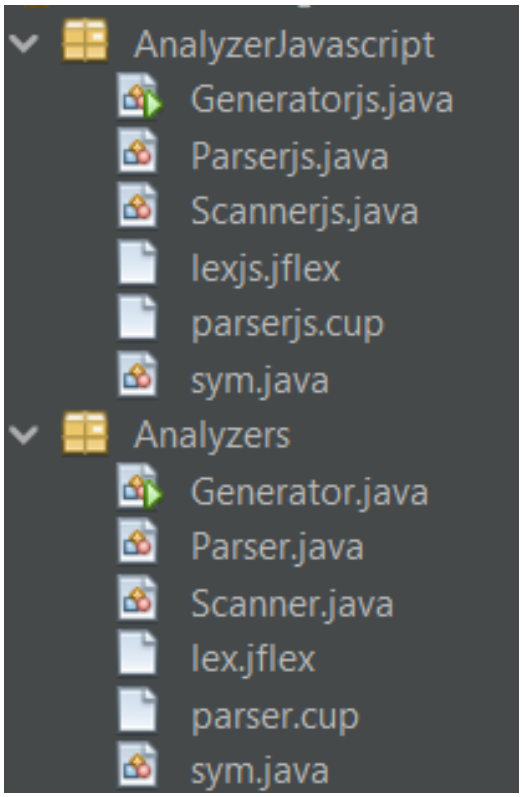
Como se puede observar se tomó la decisión de hablar sobre estos dos paquetes en un mismo renglón, esto se debe a que tanto el paquete Analyzers como el paquete AnalyzersJavascript tienen la misma estructura, a diferencia que uno analiza lenguaje FCA y otro JavaScript.

Se pueden estar preguntando ¿FCA? El lenguaje FCA es un lenguaje sencillo el cual su sintaxis se explica en el manual de usuario, y el lenguaje Javascript por ser un lenguaje utilizado a nivel internacional se asumirá que se tiene el conocimiento del mismo.

Luego de una breve explicación de lo que contienen estos dos paquetes en el proyecto de FIUSAC COPY ANALYZER, se puede dar un contexto o una explicación más profunda y técnica.

Se puede observar que contamos con 6 ficheros dentro de nuestras clases de estos ficheros solamente se codificaron 3 los cuales son:

- GeneratorJs/Generador
- ParserJs.cup/Parser.cup
- Lexjs.lex/Lex.lex



Como podemos observar hay 3 ficheros los cuales no se codificaron ya que estos ficheros se generan automáticamente gracias a las librerías de Flex que nos permiten ejecutar una serie de instrucciones para crea un analizador léxico como sintáctico.

Explicaremos más a fondo cada uno de estos

Lexjs.lex/Lex.lex

Este archivo no sigue la sintaxis de java como tal, aunque aún así puede recibir como parámetros código de Java, el objetivo principal de este documento es declarar todos los tokens o terminales de nuestro lenguaje a compilar, se declara todo lo que podamos recibir en nuestra sintaxis desde espacios blancos hasta palabras reservadas.

Parserjs.cup/Parser.cup

Este archivo a diferencia de el .lex obedeciendo todos tokens y terminales declarados en el .lex se utiliza para crear producciones y gramáticas permitidas en la sintaxis del lenguaje que estemos aceptando, en este caso los archivos que tienen terminación de js son los que están destinados al análisis exclusivo de JavaScript, y los que no tienen más que su nombre están destinados para el análisis léxico y sintáctico de el lenguaje FCA.

Generatorjs/Generator

Este es el único archivo o clase de java de la cual se escribe el código manualmente, como podemos observar no es algo más que la declaración de objetos para que puedan ser compilados y ejecutados nuestros analizadores sintácticos y léxicos descritos con Flex.

El archivo Sym.java se genera también a partir de el .lex y el .cup este archivo contiene todos los símbolos constantes y los declara como enteros finales, también declara un arreglo de string en donde están agregados todos nuestros terminales agregados en el .lex.

LógicaFCA:

En este paquete se contiene casi en su totalidad la lógica de la obtención de datos envío de datos y todo lo relacionado con lógica del analizador de copias.

Las clases finales o que solamente se utilizan como objetos para ser almacenados o instanciados son:

Repitenciametodo.java y Repitenciaclase.java

Cabe resaltar que Repitenciaclase contiene un arreglo de objetos de tipo repitencia método, ya que se utiliza para obtener los métodos o funciones que vienen dentro de una clase en específico.

Variablesglobales.java

Esta clase se utiliza única y exclusivamente para el análisis de FCA, ya que en este se guardan los identificadores, valor y tipo. En el lenguaje js solamente se analiza la sintaxis no se almacenan datos de variables o cosas así.

Logicajs/Logica:

Estas son las dos clases más importantes ya que unifican a los 4 analizadores (2 sintácticos y 2 léxicos), dentro de estas clases podremos encontrar todos, o casi todas las obtenciones de parámetros. El objetivo si se pueden analizar cada uno de los métodos y funciones es de almacenar los datos para poder manipularlos y ser comparados, al menos de parte de javascript. Ahora en el FCA si involucramos un poco de lógica para sobrescribir variables y añadirle la lógica a la sobreescritura.