Universidad de San Carlos de Guatemala

Laboratiorio OLC1

Erick Ivan Mayorga Rodriguez

201901758

Manual Técnico - ExpAnalyzer

7 de Marzo de 2022

Entorno de Desarrollo

Lenguajes:

- Java

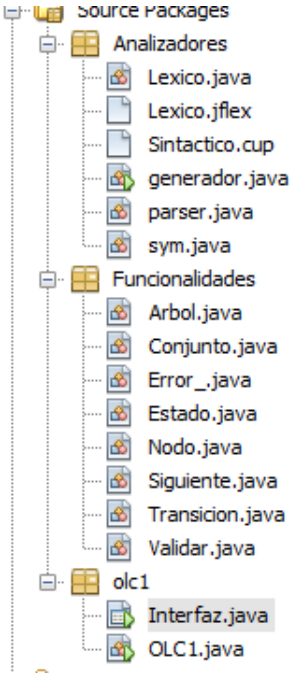
IDE:

- Apache NetBeans 12.6

Librerías:

- Java JCup 11b
- Java Jflex 1.7.0

Estructura



Las clases del paquete analizador se encargan exclusivamente del análisis léxico y sintáctico de la cadena de texto introducida desde el archivo de entrada, todo esto con la ayuda de JFlex y JCup respectivamente.

La clase generador del paquete analizadores compila los archivos .flex y .cup para la creación de sus clases en código de notación java y asi puedan ser manejadas las estructuras de análisis léxico y sintáctico desde la clase generada parser.

Para el manejo de objetos de utiliza la clase Arbol, siendo este un árbol binario que cuenta con nodos de tipo Nodo, los cuales a su vez tienen hojas izquierdas y derechas con atributos como valor, anulabilidad, un identificador y un identificador especial para las hojas.

```
public class Arbol {
    public Nodo raiz;
    public String nombre;
    public int contadorS;

    public int contadorEstado;
    public boolean crearEstadoNuevo;
    public ArrayList<Integer> sigEstadoNuevo;

    public int contadorTransicionEstado;
    //Listas Para Crear Tablas
    public ArrayList<Siguiente> listaSiguientes;
    public ArrayList<Transicion> listaTransiciones;
    public ArrayList<Estado> listaEstados;
    boolean transicionFinal = false;

    int contadorTerminalesGraficar;
    public ArrayList<String> listaTerminales;
```

Dentro de la clase Arbol se encuentran arreglos de las demás clases, dado que este gestiona los mismos para la generación de archivos para su posterior traficación, además del orden lógico de los mismos.

A continuación se mostraran las variables que contienen los arreglos declarados en la clase árbol:

```
*/
public class Siguiente {
    int id;
    public String valorRama;
    public ArrayList<Integer> siguientes;

    public Siguiente(int id) {
        this.id = id;
        this.siguientes = new ArrayList<>();
    }

    //Metodo para agregar no repetidos
    public void agregarSiguiente(int valorAgregar) {
        boolean repetido = false;
        for (int valorLista : this.siguientes) {
            if (valorLista == valorAgregar) {
                repetido = true;
                break;
            }
        }
        if (!repetido) {
            siguientes.add(valorAgregar);
        }
    }

}
```

Clase Siguiente con un método para evitar números repetidos.

```
public class Transicion {
    public int idTransicion;
    public String valorTransicion;
    public int estadoObjetivo;

    public Transicion(){
    }

}
```

Clase transición que almacena los valores de la transición.

```java
/**
 *
 * @author emayo
 */
public class Estado {
    public ArrayList<Integer> valoresSiguientes;
    public int idEstado;
    public ArrayList<Transicion> transicionesEstado;

    public Estado(){
        this.valoresSiguientes = new ArrayList<>();
        this.transicionesEstado = new ArrayList<>();
    }



}
```

Clase Estado que representa al estado con cada una de sus transiciones.