INSTITUTO TECNOLÓGICO DE COSTA RICA

ESCUELA DE INGENIERÍA EN COMPUTACIÓN

COMPILADORES E INTERPRETES

Documentación Analizador sintáctico y léxico

Integrantes:

Andrés Gutiérrez Salas - 201223823

Profesor:

Ignacio Trejos

Cartago, Costa Rica **Junio, 08, 2020**

Analizador sintáctico y léxico

1. Se modificó el IDECompiler para desplegar el mensaje léxico y para las llamadas del generador de HTML y XML.

2. Para el lexico se modifico Token.java donde se agregaron las palabras reservadas **and**, **elsif**, **exit**, **next**, **nil**, **private**, **rec**, **repeat**, **return**, **to**, **until**, como nuevas alternativas. Se eliminó la palabra reservada **begin.**
   Se modifico el Scanner.java para la generación y la salida de HTML.

3. Se modifico Token.java donde se agregaron las palabras reservadas **and**, **elsif**, **exit**, **next**, **nil**, **private**, **rec**, **repeat**, **return**, **to**, **until**, como nuevas alternativas. Se eliminó la palabra reservada **begin.**

4. Para la generación del HTML se creó el HTWLViewer.java para manejar la tipografía correcta y la escritura del HTML, se modifico el Scanner.java para que el HTML maneje de mejor manera los tokens, espacios en blancos, tabuladores, salto de línea, fin de línea, fin de texto y los retornos de carro. Luego se modificó el IDECompiler para realizar una doble llamada léxica para la generación del HTML.

5. Del Parser.java se elimino el single-Command de los siguientes comandos:
   | "let" Declaration "in" single-Command
   | "if" Expression "then" single-Command "else" single-Command
   | "while" Expression "do" single-Command
   Se eliminó completamente el comando begin.

   Se agregaron los comandos:
   |"nil"
   | "let" Declaration "in" Command "end"
   | "if" Expression "then" Command ("elsif" Expression "then" Command)* "else" Command "end"
   | "repeat" "while" Expression "do" Command "end"
   | "repeat" "until" Expression "do" Command "end"
   | "repeat" "do" Command "while" Expression "end"
   | "repeat" "do" Command "until" Expression "end"
   | "repeat" "var" Identifier "in" Expression "to" Expression "do" Command "end"
   y se agregó el **end** para la terminación de los comandos compuestos, y todos los comandos nuevos se modificaron para que la gramática los aceptara.

   Con esto se modificaron los AbstractSyntraxTrees, los Visitors, el

Drawing Tree y Layout visitor para desplegar correctamente el AbstractSyntaxTree en el IDE del programa que se está compilando.

Se modificó el Declaration para pasar a ser compound-Declaration para luego agregar las dos nuevas declaraciones:
| "rec" Proc-Funcs "end"
| "private" Declaration "in" Declaration "end"

Para la declaración “rec” Proc-Funcs “end” se crearon dos nuevos métodos llamados **parseProcFuncs y parseProcFunc** que cuando se encuentra el Token “rec”, este llama al método **parseProcFuncs**, que además de encargarse del proc y del func como tal, se encarga de llamar a **parseProcFunc** que es el encargado de la parte **Proc-Func ("and" Proc-Func)+** que al tener el **+**, permite que proc y func se realicen una o más veces.

Para la declaración **Private** sólo se agregó una nueva declaración como un nuevo case que es el que se encarga de la declaración **"private" Declaration "in" Declaration "end" .**

En single-Declaration se modificó el caso del token “proc” para que se lea de la forma **"proc" Identifier "(" Formal-Parameter-Sequence ")" "~" Command "end".**
También en single-Declaration se modificó la declaración de la variable inicializada “var” para que sirviera tanto con el oken **COLON** como con el token **BECOMES**

6. Las rutinas se cambiaron según fue necesario en la modificaciones y agregaciones de comandos y declaraciones.

7. No fueron detectados errores sintácticos

8. Tanto para este punto como para el 9, como se explica en el punto 5, para el manejo de los árboles sintácticos se modificaron los AbstractSyntraxTrees, los Visitors, el Drawing Tree y Layout visitor para desplegar correctamente el AbstractSyntaxTree en el IDE del programa que se está compilando.

10. Para la creación del XML, se crearon dos clases, el XMLWriter y el XMLVisitor. El XMLWriter se encarga de la escritura del XML y el XMLVisitor es una clase que se implementa de Visitor.java que ya cuenta con todos los comandos, expresiones,declaraciones,...,etc. Pero con la diferencia de que el XMLVisitor se

encarga de mandar la información que encuentra al XMLWriter para que este se pueda generar de forma correcta.

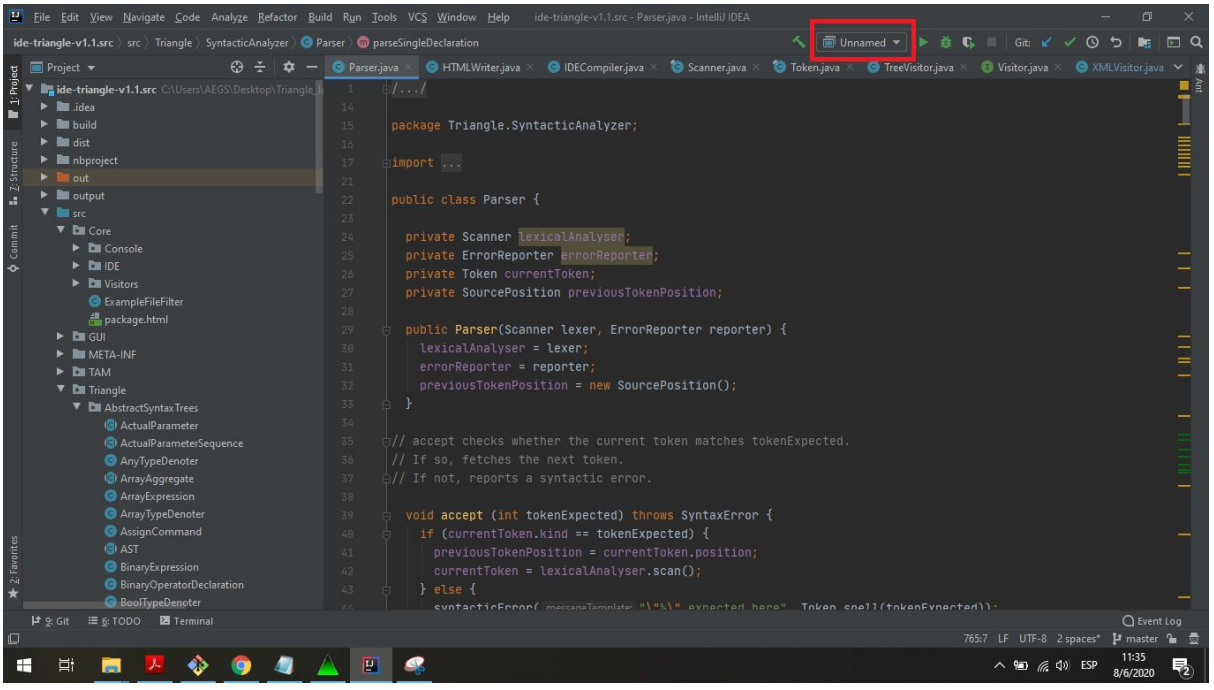
11. Se crearon tanto pruebas negativas como positivas para todo lo modificado y agregado y el objetivo de estar es probar que funcionan correctamente y el resultado es el esperado que sería que el programa compila de forma exitosa, se cree el AbstractSyntaxTrees y se pueda ver desde el IDE y el HTML y el XML sean generados de forma correcta.

12. No hubo discusión como tal al ser yo el único integrante pero los resultados obtenidos fueron como se esperaban.

13. Fue interesante y frustrante a la vez, tratar de encontrar las clases necesarias y donde tenían que modificarse o agregarse las cosas, pero una vez se hacía y se probaba y se veía que funcionaba, da una sensación de felicidad al ver que si fue uno capaz de hacerlo pese a no ser un proyecto realizado desde el inicio por uno, sino que por muchas personas.

14. Soy el único miembro así que yo me encargue de todo.

15. Para compilarlo en Intellij, se tiene que abrir el proyecto, se tiene que ir a Add Configuration que en mi caso ya sale Unnamed porque ya se configuró.



Luego se va a abrir una ventana, ahi se le da al + y se selecciona la opción Application, y ahí luego donde dice Main Class, solo se selecciona la clase Main.java

16. Para la ejecución del programa de forma directa en Windows, solo es necesario correr el **ide-triangle-v1.2.jar** que se encuentra dentro la carpeta **Triangle_Java_IDE_Gutierrez_Andres** ya con eso se abre el IDE de triángulo.

En la misma carpeta **Triangle_Java_IDE_Gutierrez_Andres** esta la carpeta output que es donde se crean los HTML y los XML luego de compilar el programa creado en triángulo.