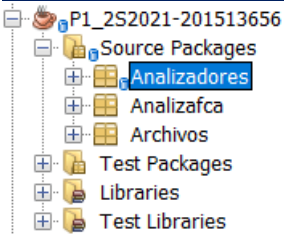
Universidad de San Carlos de Guatemala
Facultad de Ingeniería
Escuela de Ciencias y Sistemas
Organización de Lenguajes y Compiladores 1
Kimberly Julissa Estupe Chen
201513656

Manual Técnico

Guatemala 05 de septiembre del 2021

Manual Técnico

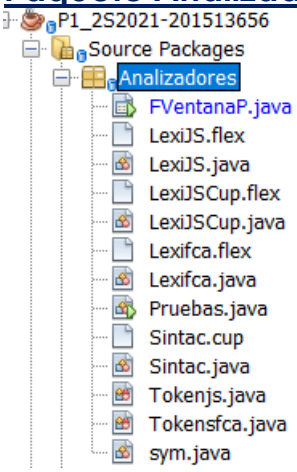
Estructura de l proyecto



Se compone por tres paquetes principales:

- Analizadores: Contiene el analizador js
- Analizafca: contiene el analizador sintáctico del lenguaje fca
- Archivos: contiene las clases de errores

Paquete Analizadores:



✓ Métodos FVentana:

```
void addToken(String lexema, String tipo, String archivo, int linea,
int columna){
    RTokens token = new RTokens(lexema, tipo, archivo, linea, columna)
    ;
    rTokens.add(token);
}


ublic static void AddES(String lexema, String tipo,String archivo,
int linea, int columna) {
    TErrores er = new TErrores(lexema, tipo, archivo, linea, columna);
    ErrorM.add(er);
}
```

Agregan tokens y errores a sus respectivas listas

Genera los repostes de tokens y de errores

```
public void Reporte(String Titulo, String relleno, String NameDoc){
    Archivo arc = new Archivo();
    DateTimeFormatter fechar_hora = DateTimeFormatter.ofPattern("dd/
    MM/yyyy  HH:mm");
    String html = "<!DOCTYPE HTML5>\n"
            + "<html>\n"
            + " <head><center>\n"
            + "   <meta charset=\"UTF-8\"/>\n"
            + "   <h1 style=\"color:#EC7063\">"+Titulo+"</h1>\n"
            + " </center></head>\n"
            + " <body bgcolor=\"#061626\"><center>\n"
            + "   <h2 style=\"color:#BF67EC\">Kimberly Julissa Estupe
```

```
   AnLexico ();
   AnSintac();
}//GEN-LAST:event_jmiEjecutarActionPerformed

public void AnLexico (){
    try {
        // TODO add your handling code here:
        File archivo = new File ("archivo.txt");// Crea el archivo donde se guadara ka
        entrada
        PrintWriter escribe; // Escribe en el archivo
        escribe = new PrintWriter(archivo);
        escribe.print(jtEntrada.getText());
        escribe.close();
    } catch (FileNotFoundException ex) {
        Logger.getLogger(FVentanaP.class.getName()).log(Level.SEVERE, null, ex);
    }
```

Mandan a realizar los análisis léxico y sintáctico de los diferentes lenguajes

LexiJs.flex: Analizar lexico de Js

```
(class) {Lexejs=yytext(); Lineajs =yyline; Colujs=yycolumn; return Clase;}
(do) {Lexejs=yytext(); Lineajs =yyline; Colujs=yycolumn; return Do;}
(while) {Lexejs=yytext(); Lineajs =yyline; Colujs=yycolumn; return While;}
(if) {Lexejs=yytext(); Lineajs =yyline; Colujs=yycolumn; return If;}
(else) {Lexejs=yytext(); Lineajs =yyline; Colujs=yycolumn; return Else;}
(var | let | const) {Lexejs=yytext();Lineajs =yyline; Colujs=yycolumn;  return Variable;}
(llamada){D} {Lexejs=yytext(); Lineajs =yyline; Colujs=yycolumn; return Llamada;}
(for) {Lexejs=yytext();Lineajs =yyline; Colujs=yycolumn; return For;}
(switch) {Lexejs=yytext();Lineajs =yyline; Colujs=yycolumn; return Switch;}
(break) {Lexejs=yytext();Lineajs =yyline; Colujs=yycolumn; return Break;}
(require) {Lexejs=yytext();Lineajs =yyline; Colujs=yycolumn; return Require;}
(true)      {Lexejs = yytext();Lineajs =yyline; Colujs=yycolumn; return True;}
(false)      {Lexejs = yytext();Lineajs =yyline; Colujs=yycolumn; return False;}
(console) {Lexejs=yytext();Lineajs =yyline; Colujs=yycolumn; return Console;}
```

LexicCup.flex Y Sintac.cup: Realizan el analisis sintactico de Js

```
(class)             {return new Symbol(sym.Clase, yycolumn, yyline, yytext());}
(do)                {return new Symbol(sym.Do, yycolumn, yyline, yytext());}
(while)             {return new Symbol(sym.While, yycolumn, yyline, yytext());}
(if)                {return new Symbol(sym.If, yycolumn, yyline, yytext());}
(else)              {return new Symbol(sym.Else, yycolumn, yyline, yytext());}
(var | let | const) {return new Symbol(sym.Variable, yycolumn, yyline, yytext());
(llamada){D}        {return new Symbol(sym.Llamada, yycolumn, yyline, yytext());}
(for)               {return new Symbol(sym.For, yycolumn, yyline, yytext());}
(switch)            {return new Symbol(sym.Switch, yycolumn, yyline, yytext());}
(break)             {return new Symbol(sym.Break, yycolumn, yyline, yytext());}
(require)           {return new Symbol(sym.Require, yycolumn, yyline, yytext());}
(true | false)      {return new Symbol(sym.Bool, yycolumn, yyline, yytext());}
(console)           {return new Symbol(sym.Console, yycolumn, yyline, yytext());}
(log)               {return new Symbol(sym.Log, yycolumn, yyline, yytext());}
(default)           {return new Symbol(sym.Default, yycolumn, yyline, yytext());}
(case)              {return new Symbol(sym.Case, yycolumn, yyline, yytext());}
```

```
terminal Clase, Do, While, If, Else, Llamada, For, Console, Log,
        Default, Case, Break, Variable, Switch, Require,
        Logico, Not, Matematico, Suma, Resta, Igual, Relacionales, Incremento, Bool,
        ParenA, ParenC, LlaveA, LlaveC, Punto, Coma, PComa, DPuntos,
        Cadena, Numero, Identificador, Char;

//---------------------------------- NO TERMINALES
-------------------------------------------------
/*Son creadas para cup*/
non terminal INIS, INI, SENTENCIA, SENTENCIAS, INSTRUCCIONES, INSTRUCCION, PARENTESIS,
METODO, METODOS;
non terminal ELSE, CASE, CASES, FINEXPRE, TIPOD, FOR, DECLARAFOR;
non terminal DECLARA, EXPRESIONES, EXPRESION, PARAMETRO,PARAMETROS, SUMAS;
non terminal PARENMETODOS, CASBREAK, ULTIDE, TIPOMETODOS, CICLOSLLAVES, TERMINALESS,LLAMADA,
P_LLAMADA;
```

Lexifca.flex: Analizar lexico de fca

```
DefinirGlobales {Lexefca=yytext();Lineafca =yyline;Colufca=yycolumn;  return Definir_Globales;
}
GraficaBarras   {Lexefca=yytext();Lineafca =yyline;Colufca=yycolumn;  return Grafica_Barras;}
GraficaLineas   {Lexefca=yytext();Lineafca =yyline;Colufca=yycolumn;  return Grafica_Lineas;}
GraficaPie      {Lexefca=yytext();Lineafca =yyline;Colufca=yycolumn;  return Grafica_Pie;}
Compare         {Lexefca=yytext();Lineafca =yyline;Colufca=yycolumn;  return Compare;}
string          {Lexefca=yytext();Lineafca =yyline;Colufca=yycolumn;  return Dato_String;}
double          {Lexefca=yytext();Lineafca =yyline;Colufca=yycolumn;  return Dato_Double;}
EjeX            {Lexefca=yytext();Lineafca =yyline;Colufca=yycolumn;  return Eje_X;}
Titulo          {Lexefca=yytext();Lineafca =yyline;Colufca=yycolumn;  return Titulo;}
Valores         {Lexefca=yytext();Lineafca =yyline;Colufca=yycolumn;  return Valores;}
TituloX         {Lexefca=yytext();Lineafca =yyline;Colufca=yycolumn;  return Titulo_X;}
TituloY         {Lexefca=yytext();Lineafca =yyline;Colufca=yycolumn;  return Titulo_Y;}
Archivo         {Lexefca=yytext();Lineafca =yyline;Colufca=yycolumn;  return Archivo;}
```

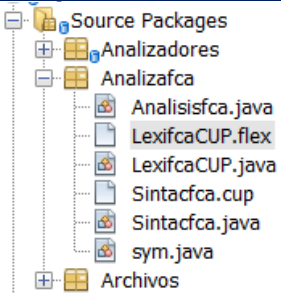
Tokenjs: Es una clase enum que contiene los tokens del analizador Js

```
public enum Tokenjs
    Clase,
    Do,
    While,
    If,
    Else,
    Variable,
    For,
    Switch,
    Break,
    Require,
    Llamada,
    True,
    False,
    Console,
    Log,
    Default,
    Cadena,
    Numero,
    Identificador,
    Comentario_Multilinea,
    Comentario,
    Char,
    ERROR
}
```

Pruebas: ejecuta los analizadores.

```
public class Pruebas {
    public static void main(String[] args) throws Exception {
        String R1 = "src/Analizadores/";
        String R2="src/Analizafca/LexifcaCUP.flex";
        String[] RS = {"-parser", "Sintac","src/Analizadores/Sintac.cup"};
        String[] RS2 = {"-parser", "Sintacfca","src/Analizafca/Sintacfca.cup"};

        AnalizarJS(R1+"LexiJS.flex", R1+"LexiJSCup.flex", RS);
        Analizarfca(R1+"Lexifca.flex",R2,RS2 );
    }

    //********************************** ANALIZADORES LEXICO Y SINTACTICO FCA ********************************************************************
    public static void Analizarfca(String R1,String R2,String[] RS) throws IOException, Exception{...28 lines }


    //********************************** ANALIZADORES LEXICO Y SINTACTICO JS *********************************************************************
    public static void AnalizarJS(String R1,String R2,String[] RS) throws IOException, Exception{...28 lines }
```

Paquete Analizafca

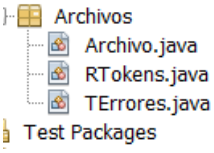


Lexifca.flex y Sintacfca.cup: realizan el análisis sintáctico del lenguaje fca

```
DefinirGlobales {return new Symbol(sym.Globales, yycolumn, yyline, yytext());}
GraficaBarras   {return new Symbol(sym.GBarras, yycolumn, yyline, yytext());}
GraficaLineas   {return new Symbol(sym.GLineas, yycolumn, yyline, yytext());}
GraficaPie      {return new Symbol(sym.GPie, yycolumn, yyline, yytext());}
Compare         {return new Symbol(sym.Compare, yycolumn, yyline, yytext());}
string          {return new Symbol(sym.String, yycolumn, yyline, yytext());}
double          {return new Symbol(sym.Double , yycolumn, yyline, yytext());}
EjeX            {return new Symbol(sym.EjeX , yycolumn, yyline, yytext());}
Titulo          {return new Symbol(sym.Titulo , yycolumn, yyline, yytext());}
Valores         {return new Symbol(sym.Valores , yycolumn, yyline, yytext());}
TituloX         {return new Symbol(sym.TituloX , yycolumn, yyline, yytext());}
TituloY         {return new Symbol(sym.TituloY , yycolumn, yyline, yytext());}
```

```
//---------------------------------- TERMINALES
------------------------------------------------
/*viene del Lex*/
terminal Globales, GBarras, GLineas, GPie, Compare, String, Double, EjeX, Titulo, Valores,
    TituloX, TituloY, Archivo,
    ParenA, ParenC, LlaveA, LlaveC, CorcheA, CorcheC,
    Cadena, Numero, Identificador, Ruta;
```

Paquete Archivos:



Archivo:

```java
public class Archivo {
    ///////////  Generar Reportes ///////////////
    public void Reportes(String text, String fileName){
        File file = new File(fileName);
        BufferedWriter buffer = null;
        try {
            buffer = new BufferedWriter(new FileWriter(file));
            buffer.write(text);
        } catch (IOException e) {
            System.out.println("Error E/S: "+e);
        }
        finally{
            try {
                buffer.close();
                //************** ABRIR EL ARCHIVO DIRECTAMENTE ***********//////////
                try {
                    File path = new File (fileName);
                    Desktop.getDesktop().open(path);
                }catch (IOException ex) {
                    ex.printStackTrace();
                }
                ////////////////////////////////////////////////////////////////////

            } catch (IOException ex) {
                Logger.getLogger(Archivo.class.getName()).log(Level.SEVERE, null, ex);
            }
        }
    }
```

Genera un archivo con la información del archivo, y el nombre con la extensión requerida

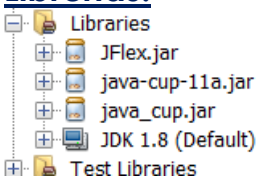
RTokens: clase de tipo RTokens que contendrá los Tokens de los análisis léxicos

```java
public class RTokens {
    String lexema,token,archivo;
    int linea, columa;

    public RTokens(String lexema, String token, String archivo, int linea, int columa) {
        this.lexema = lexema;
        this.token = token;
        this.archivo = archivo;
        this.linea = linea;
        this.columa = columa;
    }
```

Terrores: clase de tipo TErrores que contendrá los errores de los análisis léxico y sintáctico de los diferentes archivos analizados

```java
public class TErrores {
    String lexema,tipo,archivo;
    int linea,columna;

    public TErrores(String lexema, String tipo, String archivo, int linea, int columna) {
        this.lexema = lexema;
        this.tipo = tipo;
        this.archivo = archivo;
        this.linea = linea;
        this.columna = columna;
    }
```

Librerías:



Las librerías utilizadas son JFlex y Cup
El JDK es en la versión 1.8

IDE utilizado:

NetBeans IDE 8.2

Para este proyecto se utilizo NetBeans IDE en su versión 8.2