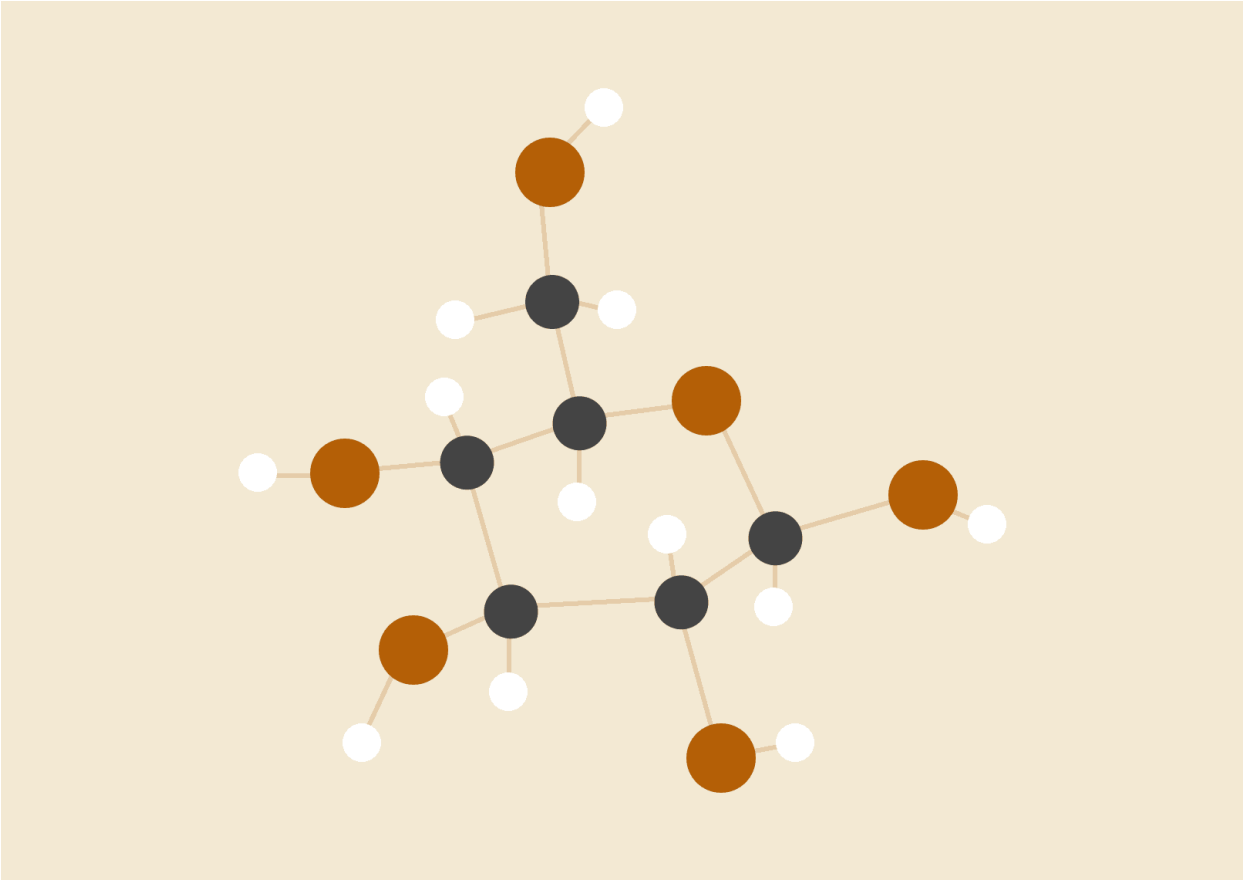
MANUAL TECNICO: PROYECTO 1

Manual Tecnico de Proyecto 1



Fernando Jose Rodriguez Ramirez 202030542

INTRODUCCIÓN

El presente manual indica varias especificaciones del software de copia.

ANALIZADOR LÉXICO

El analizador léxico de Java utiliza las siguientes expresiones regulares y estados para su funcionamiento.

<YYINITIAL> {

```
//Se declaran las palabras reservadas

import {return symbol(sym.IMPORT, yytext());}


int {return symbol(sym.PR_INT, yytext());}

boolean {return symbol(sym.PR_BOOLEAN, yytext());}

String {return symbol(sym.PR_STRING, yytext());}

char {return symbol(sym.PR_CHAR, yytext());}

double {return symbol(sym.PR_DOUBLE, yytext());}

Object {return symbol(sym.PR_OBJECT, yytext());}


if {return symbol(sym.IF, yytext());}

else {return symbol(sym.ELSE, yytext());}

for {return symbol(sym.FOR, yytext());}

while {return symbol(sym.WHILE, yytext());}

do {return symbol(sym.DO, yytext());}

switch {return symbol(sym.SWITCH, yytext());}
```

default {return symbol(sym.DEFAULT, yytext());}

public {return symbol(sym.PUBLIC, yytext());}

private {return symbol(sym.PRIVATE, yytext());}

protected {return symbol(sym.PROTECTED, yytext());}

final {return symbol(sym.FINAL, yytext());}

class

this

super

new

case

break

return

"."

","

":"

";"

"{"

"}"

"("

")"

"["

"]"

"+"

```
"*"

"-"

"/"

"%"

"="

"+="

"-="


">"

"<"

"=="

"!="

">="

"<="

"!"

"&&"

"||"

"++"

"--"

"\"" {yybegin(STRING);}


"-"?{D}+

"-"?{D}+("f"|"F"|"d"|"D")

"-"?{D}+"."{D}+("f"|"F"|"d"|"D")?
```

```
        "//" {yybegin(COMENTARIO_LINEA);}
        "/*" {yybegin(COMENTARIO_MULTILINEA);}

        ({L}|"$"|"_")({L}|{D}|"$"|"_")*
        //Se ignoraran espacios sueltos
        {espacio}|{salto}
        [^] { agregarError }
}

<STRING> {
        ({L}|{D}|{espacio})
        "\"" {yybegin(YYINITIAL)}
        {salto}|[^] { agregarError; yybegin(YYINITIAL);}
}

<COMENTARIO_LINEA> {
        {salto} {yybegin(YYINITIAL);}
        [^]

<COMENTARIO_MULTILINEA> {
        "*/" {yybegin(YYINITIAL);}
        [^]
}
```

El analizador lexico del lenguaje Copy tiene las siguientes expresiones y estados:

```
<YYINITIAL> {
        nombre-proyecto {return symbol(sym.NOMBRE_PROYECTO, yytext());}
        directorio-json {return symbol(sym.DIRECTORIO_JSON, yytext());}
        directorio-def {return symbol(sym.DIRECTORIO_DEF, yytext());}
        ":" {return symbol(sym.DOS_PUNTOS, yytext());}
        "," {return symbol(sym.COMA, yytext());}
        "\"" {yybegin(STRING);}
        //Se ignoraran espacios sueltos
        {espacio}|{salto} {/*Ignorar*/}
        [^] {System.out.println("Caracter no definido en:"+(yyline+1)+",
column:"+(yycolumn+1)+" "+buffer.toString());resetBuffer();}
}

<STRING> {
        "\"" {String lexema = buffer.toString();yybegin(YYINITIAL);resetBuffer();return
symbol(sym.STRING,lexema);}
        {salto}+ {System.out.println("Error: Cadena
incompleta");yybegin(YYINITIAL);resetBuffer();}
        [^] {buffer.append(yytext());}
}
```

El analizador lexico del lenguaje JSON utiliza las siguientes expresiones y estados:

```
<YYINITIAL> {
        "," { return symbol(sym.COMA, yytext()); }
        "[" { return symbol(sym.L_CORCHETE, yytext()); }
        "]" { return symbol(sym.R_CORCHETE, yytext()); }
        "{" { return symbol(sym.L_LLAVE, yytext()); }
        "}" { return symbol(sym.R_LLAVE, yytext()); }
        ":" { return symbol(sym.DOS_PUNTOS, yytext()); }
        "true" { return symbol(sym.TRUE, yytext()); }
        "false" { return symbol(sym.FALSE, yytext()); }
        "null" { return symbol(sym.NULL, yytext()); }
        {numeros} { return symbol(sym.NUMERO, yytext()); }
        "\"" {yybegin(STRING);}
        {espacio}|{salto} {/*Ignorar*/}
        [^] { System.out.println("Caracter no definido en:"+(yyline+1)+",
column:"+(yycolumn+1)+" "+yytext()); agregarError(symbol(sym.EOF, yytext()), "El
caracter ingresado ha esta definido"); resetBuffer(); }
}
<STRING> {
        "\"" {String lexema = buffer.toString();yybegin(YYINITIAL);resetBuffer();return
symbol(sym.STRING,lexema);}
        {salto}+ {System.out.println("Error: Cadena
incompleta");yybegin(YYINITIAL);resetBuffer();}
        [^] {buffer.append(yytext());}
}
```

El analizador léxico del lenguaje Def utiliza las siguientes expresiones y estados:

```
<YYINITIAL> {
        Integer {return symbol(sym.PR_INTEGER, yytext());}
        String {return symbol(sym.PR_STRING, yytext());}
        RESULT {return symbol(sym.RESULT, yytext());}
        html {return symbol(sym.TAG_HTML, yytext());}
        h1 {return symbol(sym.TAG_HEADER,yytext());}
        h2 {return symbol(sym.TAG_HEADER,yytext());}
        table {return symbol(sym.TAG_TABLE,yytext());}
        for {return symbol(sym.TAG_FOR,yytext());}
        iterador {return symbol(sym.ITERADOR,yytext());}
        hasta {return symbol(sym.HASTA,yytext());}
        tr {return symbol(sym.TAG_TR,yytext());}
        th {return symbol(sym.TAG_TH, yytext());}
        td {return symbol(sym.TAG_TD, yytext());}
        br {return symbol(sym.TAG_BR, yytext());}
        "." {return symbol(sym.PUNTO, yytext());}
        ":" {return symbol(sym.DOS_PUNTOS, yytext());}
        "," {return symbol(sym.COMA, yytext());}
        ";" {return symbol(sym.PUNTO_COMA, yytext());}
        "(" {return symbol(sym.PAR_IZQ, yytext());}
        ")" {return symbol(sym.PAR_DER, yytext());}
        "[" {return symbol(sym.CORCH_IZQ, yytext());}
        "]" {return symbol(sym.CORCH_DER, yytext());}
```

```
        "+" {return symbol(sym.CRUZ, yytext());}
        "*" {return symbol(sym.ASTERISCO, yytext());}
        "-" {return symbol(sym.GUION, yytext());}
        "/" {return symbol(sym.BARRA, yytext());}
        "=" {return symbol(sym.IGUAL, yytext());}
        ">" {return symbol(sym.MAYOR_QUE, yytext());}
        "<" {return symbol(sym.MENOR_QUE, yytext());}
        "</" {return symbol(sym.FIN_TAG, yytext());}
        "$" {return symbol(sym.DOLAR,yytext());}
        "$$" {return symbol(sym.DOBLE_DOLAR,yytext());}
        "</".*."/>" {/*Ignorar comentarios*/}
        "\"" {yybegin(STRING);}

        "-"?{D}+ {return symbol(sym.ENTERO, yytext());}
        ({L}|"_")({L}|{D}|"_")* {return symbol(sym.VARIABLE_IDENTIFICADOR, yytext());}
        {espacio}|{salto} {/*Ignorar*/}
        [^] { yybegin(TEXTO); buffer.append(yytext());}
}

<STRING> {
        "\"" {String lexema = buffer.toString();yybegin(YYINITIAL);resetBuffer();return
symbol(sym.STRING,lexema);}
        {salto}+ {System.out.println("Error: Cadena
incompleta");yybegin(YYINITIAL);resetBuffer();}
```

[^] {buffer.append(yytext());}

}

<TEXTO> {

{espacio}|{salto} {yybegin(YYINITIAL); String lexema = buffer.toString(); resetBuffer(); return symbol(sym.TEXTO, lexema);}

[^] {buffer.append(yytext());}

}

ANALIZADOR SINTACTICO

El lenguaje Java utiliza el las siguientes producciones en su analisis sintactico:

Se puede encontrar la definicion en el apartado de Analisis de la aplicacion del servidor.

El lenguaje Copy utiliza el las siguientes producciones en su analisis sintactico:

Terminales:

NOMBRE_PROYECTO, DIRECTORIO_JSON, DIRECTORIO_DEF, STRING, DOS_PUNTOS, COMA

No Terminales:

inicio, declaraciones_campos, campo, informacion_campo ;

Se puede encontrar la definicion en el apartado de Analisis de la aplicacion de la aplicacion cliente.

El lenguaje JSON utiliza el las siguientes producciones en su analisis sintactico:

Terminales:

COMA, DOS_PUNTOS, L_CORCHETE, R_CORCHETE, L_LLAVE, R_LLAVE, TRUE, FALSE, NULL, NUMERO, STRING ;

No Terminales:

json, arreglo, objeto, lista_pares, lista_paresd, par, valor, lista_valores, lista_valoresd ;

El lenguaje Def utiliza el las siguientes producciones en su analisis sintactico:

Terminales:

PR_INTEGER, PR_STRING, TAG_HTML, TAG_HEADER, TAG_TABLE, TAG_FOR, ITERADOR, HASTA, TAG_TR, TAG_TH, TAG_TD, TAG_BR, RESULT, FIN_TAG, PUNTO,COMA,DOS_PUNTOS,PUNTO_COMA,LLAVE_IZQ,LLAVE_DER,PAR_IZQ,PAR_DER,CORCH_IZQ,CORCH_DER, MENOS_UNITARIO, CRUZ,ASTERISCO,GUION,BARRA,IGUAL,MAYOR_QUE,MENOR_QUE,DOLAR,DOBLE_DOLAR,ENTERO,VARIABLE_IDENTIFICADOR, STRING, TEXTO ;

No Terminales:

inicio, declaracion_variables, declaracion_variable, declaracion_html, declaraciones_etiquetas, etiqueta, etiquetad, tag_for, acceso_variable, variable, texto_plano, texto, declaraciond, declaraciondd, asignacion, result, resultd, resultdd, expresion, expresion_matematica, valor_numerico

Se puede encontrar la definicion en el apartado de Analisis de la aplicacion de la aplicacion cliente.