Universidad San Carlos De Guatemala 21/02/2019

Facultad de Ingeniería

Escuela de Ciencias y Sistemas

Organización de lenguajes y compiladores 1

Primer Semestre 2019

Ing. Mario José Bautista Fuentes

Aux. Nery Galvez

Bruno Marco José Coronado Morales 20170932

Manual Técnico

Introducción

Para el proyecto solicitado se requirió un sistema capaz de analizar y procesar un lenguaje de etiquetas con código embebido incrustado entre etiquetas y que se genere una representación visual de todos los elementos que se definen en el programa de entrada mediante la forma de una pagina HTML.

Requerimientos Mínimos

Se recomienda ejecutar la solucion en un sistema Linux o Windows 7+ que cuente con el JRE 8.x.

Plataforma de Desarrollo

La solución se desarrollo sobre un sistema operativo de la familia de Linux con la utilización del IDE IntelliJ IDEA Ultimate Edition.

Implementación JFLEX

Para la implementación del analizador léxico del proyecto se utilizo JFLEX para la creación de un scanner, el cual fuera capaz de soportar el lenguaje solicitado.

Implementación CUP

Para la implementación del analizador sintáctico del proyecto se utilizo Java CUP para la creacion de un parser, el cual fuera capaz de soportar la sintaxis del lenguaje solicitado.

Gramática

A::= menorQue compi mayorQue B menorQue barra compi mayorQue aceptacion

B::= B C
|C

C::= menorQue cabecera mayorQue D menorQue barra cabecera mayorQue
|menorQue cuerpo F I1 menorQue barra cuerpo mayorQue
|menorQue cabecera mayorQue menorQue barra cabecera mayorQue
|menorQue cuerpo F menorQue barra cuerpo mayorQue

D::= D E
|E

E::= menorQue titulo textoEntreEtiquetas barra titulo mayorQue
|menorQue titulo mayorQue menorQue barra titulo mayorQue

F::= G mayorQue
|mayorQue

G::= G H
|H

H::= fondo igual textoEntreComillas

I::= menorQue I2 I3 menorQue barra espacio
|menorQue I2 I3 barra espacio
|menorQue parrafo J barra parrafo
|menorQue barra salto
|menorQue textoA M barra textoA
|menorQue textoB N barra textoB
|menorQue O0 O mayorQue menorQue barra imagen
|menorQue P0 P mayorQue menorQue barra boton
|menorQue Q0 Q R0 R menorQue barra tabla
|menorQue inicioHS U interrogacion

I1::= I1 I
|I

I2::= espacio
|espacio menorQue

I3::= I3 I4
|I4

I4::= menorQue I2 I3 menorQue barra espacio
|menorQue I2 I3 barra espacio
|menorQue parrafo J barra parrafo
|menorQue barra salto
|menorQue textoA M barra textoA
|menorQue textoB N barra textoB
|menorQue O0 O mayorQue menorQue barra imagen
|menorQue P0 P mayorQue menorQue barra boton
|menorQue menorQue inicioHS U interrogacion
|menorQue inicioHS U interrogacion
|inicioHS U interrogacion
|textoEntreEtiquetas:a
|I2 I3 menorQue barra espacio
|I2 I3 barra espacio
|parrafo J barra parrafo
|barra salto
|textoA M barra textoA
|textoB N barra textoB
|O0 O mayorQue menorQue barra imagen
|P0 P mayorQue menorQue barra boton
|menorQue Q0 Q R0 R menorQue barra tabla
|Q0 Q R0 R menorQue barra tabla
|menorQue menorQue I2 I3 menorQue barra espacio
|menorQue menorQue I2 I3 barra espacio
|menorQue menorQue parrafo J barra parrafo
|menorQue menorQue barra salto
|menorQue menorQue textoA M barra textoA
|menorQue menorQue textoB N barra textoB
|menorQue menorQue O0 O mayorQue menorQue barra imagen
|menorQue menorQue P0 P mayorQue menorQue barra boton
|menorQue textoEntreEtiquetas:a

J::= K textoEntreEtiquetas:a
|textoEntreEtiquetas:a
|mayorQue menorQue
|barra salto
|J barra salto

K::= K L:a
|L:a

L::= alineacion igual tipoAlineacion:a

M::= textoEntreEtiquetas:a
|mayorQue menorQue

N::= textoEntreEtiquetas:a
|mayorQue menorQue

O0::= imagen

O::= O O1
|O1
;

O1::= path igual textoEntreComillas:a
|alto igual numero:a
|ancho igual numero:a

P0::= boton

P::= P P1
|P1
;

P1::= id igual textoEntreComillas:a
|texto igual textoEntreComillas:a

Q0::= tabla

Q::= Q Q1
|Q1
;

Q1::= borde igual booleano:a

R0::= mayorQue

R::= R R1
|R1
;

R1::= menorQue R2 mayorQue S menorQue barra fila mayorQue

R2::= fila

S::= S S1
|S1
;

S1::= menorQue T3 T barra columnaC
|menorQue T0 T barra columna
|menorQue T3 T menorQue barra columnaC

|menorQue T0 T menorQue barra columna

T0::= columna

T3::= columnaC

T::= T T1
|T1
;

T1::= textoEntreEtiquetas:a
|menorQue barra salto
|barra salto
|menorQue parrafo J barra parrafo
|parrafo J barra parrafo
|menorQue O0 O mayorQue menorQue barra imagen
|O0 O mayorQue menorQue barra imagen
|menorQue P0 P mayorQue menorQue barra boton
|P0 P mayorQue menorQue barra boton
|menorQue inicioHS U interrogacion
|inicioHS U interrogacion
;

U::= U U1
|U1
;

U1::= dolar identificador:a igual V:b puntoYComa
|echo W:a puntoYComa
|numeral identificador:a igual X:b puntoYComa
|dolar identificador:a igual numeral U2 punto AH:b
|numeral U2 punto AI
|numeral U2 punto insertar parentesisAbre parentesisCierra puntoYComa}
|AQ00 parentesisAbre AQ0 parentesisCierra llaveAbre AO1 llaveCierra
|AQ00 parentesisAbre AQ0 parentesisCierra llaveAbre AO1 llaveCierra controlElse
llaveAbre AO10 llaveCierra

|U5 parentesisAbre AN parentesisCierra llaveAbre AO llaveCierra
;

U5::= repetir

U2::= identificador:a
;

```
V::= textoEntreComillas:a
|V1:a
|booleano:a
;

V1::= V2:a

V2::= V2:a mas V2:b
|V2:a menos V2:b
|V2:a multiplicacion V2:b
|V2:a division V2:b
|parentesisAbre V2:a parentesisCierra
|entero:a
|decimal:a
|menos entero:a
|menos decimal:a

W::= W1:a

W1::= W1:a punto W1:b
{:
RESULT = a + b;
:}
|dolar identificador:a
{:
RESULT = Principal.retornarValorCadenaVariable(a);
:}
|textoEntreComillas:a
{:
RESULT = a;
:}
;

X::= X1:a parentesisCierra
{:
RESULT = a;
:}
;

X1::= X2:a parentesisAbre Y
{:
RESULT = a;
:}
|X3:a parentesisAbre Z
{:
RESULT = a;
:}
```

```
|X4:a parentesisAbre AA
{:
RESULT = a;
:}
|X5:a parentesisAbre AB
{:
RESULT = a;
:}
|X6:a parentesisAbre AC
{:
RESULT = a;
:}
|X7:a parentesisAbre AD
{:
RESULT = a;
:}
;

X2::= crearParrafo
{:
structParrafo = new StructParrafo();
RESULT = 1;
:}
;

X3::= crearTextoA
{:
structTextoA = new StructTextoA();
RESULT = 2;
:}
;

X4::= crearTextoB
{:
structTextoB = new StructTextoB();
RESULT = 3;
:}
;

X5::= crearImagen
{:
structImagen = new StructImagen();
RESULT = 4;
:}
;

X6::=crearTabla
```

```
{:
structTabla = new StructTabla();
RESULT = 5;
:}
;

X7::= crearBoton
{:
structBoton = new StructBoton();
RESULT = 6;
:}
;

Y::= textoEntreComillas:a coma tipoAlineacion:b
{:
structParrafo = new StructParrafo(a,b);
:}
|textoEntreComillas:a coma dolar identificador:b
{:
if(verificarAlineacion(b))
structParrafo = new StructParrafo(a,Principal.retornarValorCadenaVariable(b));
:}
|dolar identificador:a coma tipoAlineacion:b
{:
structParrafo = new StructParrafo(Principal.retornarValorCadenaVariable(a),b);
:}
|dolar identificador:a coma dolar identificador:b
{:
if(verificarAlineacion(b))
structParrafo = new StructParrafo(Principal.retornarValorCadenaVariable(a),
Principal.retornarValorCadenaVariable(b));
:}
|textoEntreComillas:a
{:
structParrafo.setContenido(a);
:}
|dolar identificador:a
{:
structParrafo.setContenido(Principal.retornarValorCadenaVariable(a));
:}
;

Z::= textoEntreComillas:a
{:
structTextoA = new StructTextoA(a);
:}
|dolar identificador:a
```

```
{:
structTextoA.setContenido(Principal.retornarValorCadenaVariable(a));
:}
;

AA::= textoEntreComillas:a
{:
structTextoB = new StructTextoB(a);
:}
|dolar identificador:a
{:
structTextoB.setContenido(Principal.retornarValorCadenaVariable(a));
:}
;

AB::= textoEntreComillas:a
{:
structImagen = new StructImagen(a);
:}
|dolar identificador:a
{:
structImagen = new StructImagen(Principal.retornarValorCadenaVariable(a));
:}
|textoEntreComillas:a coma dolar identificador:b
{:
structImagen = new StructImagen(a, Principal.retornarValorCadenaVariable(b));
:}
|textoEntreComillas:a coma V1:b
{:
structImagen = new StructImagen(a, b.toString());
:}
|dolar identificador:a coma dolar identificador:b
{:
structImagen = new StructImagen(Principal.retornarValorCadenaVariable(a),
Principal.retornarValorCadenaVariable(b));
:}
|dolar identificador:a coma V1:b
{:
structImagen = new StructImagen(Principal.retornarValorCadenaVariable(a), b.toString());
:}
|textoEntreComillas:a coma dolar identificador:b coma dolar identificador:c
{:
structImagen = new StructImagen(a, Principal.retornarValorCadenaVariable(b),
Principal.retornarValorCadenaVariable(c));
:}
|textoEntreComillas:a coma dolar identificador:b coma V1:c
{:
```

```
structImagen = new StructImagen(a, Principal.retornarValorCadenaVariable(b),
c.toString());
:}
|textoEntreComillas:a coma V1:b coma dolar identificador:c
{:
structImagen = new StructImagen(a, b.toString(),
Principal.retornarValorCadenaVariable(c));
:}
|textoEntreComillas:a coma V1:b coma V1:c
{:
structImagen = new StructImagen(a, b.toString(), c.toString());
:}
|dolar identificador:a coma dolar identificador:b coma dolar identificador:c
{:
structImagen = new StructImagen(Principal.retornarValorCadenaVariable(a),
Principal.retornarValorCadenaVariable(b), Principal.retornarValorCadenaVariable(c));
:}
|dolar identificador:a coma dolar identificador:b coma V1:c
{:
structImagen = new StructImagen(Principal.retornarValorCadenaVariable(a),
Principal.retornarValorCadenaVariable(b), c.toString());
:}
|dolar identificador:a coma V1:b coma dolar identificador:c
{:
structImagen = new StructImagen(Principal.retornarValorCadenaVariable(a), b.toString(),
Principal.retornarValorCadenaVariable(c));
:}
|dolar identificador:a coma V1:b coma V1:c
{:
structImagen = new StructImagen(Principal.retornarValorCadenaVariable(a), b.toString(),
c.toString());
:}
;

AC::= AC coma AC
|AF AE corcheteCierra
{:
structTabla.agregarFila();
:}
;

AD::= textoEntreComillas:a coma textoEntreComillas:b
{:
structBoton = new StructBoton(a,b);
:}
|textoEntreComillas:a coma dolar identificador:b
{:
```

```
structBoton = new StructBoton(a, Principal.retornarValorCadenaVariable(b));
:}
|dolar identificador:a coma textoEntreComillas:b
{:
structBoton = new StructBoton(Principal.retornarValorCadenaVariable(a), b);
:}
|dolar identificador:a coma dolar identificador:b
{:
structBoton = new StructBoton(Principal.retornarValorCadenaVariable(a),
Principal.retornarValorCadenaVariable(b));
:}
;

AF::= corcheteAbre
{:
structTabla.crearFila();
:}
;

AE::= AE coma AE
|textoEntreComillas:a
{:
structTabla.agregarElementoAFila(a);
:}
|dolar identificador:a
{:
structTabla.agregarElementoAFila(Principal.retornarValorCadenaVariable(a));
:}
|booleano:a
{:
structTabla.agregarElementoAFila(a);
:}
|V1:a
{:
structTabla.agregarElementoAFila(a.toString());
:}
;

AH::= getContenido AM
{:
Struct struct = Principal.retornarStruct(identificadorStruct);
switch(struct.getTipo()){
case "parrafo": RESULT = ((StructParrafo)struct.getValor()).getContenido();
break;
case "textoA": RESULT = ((StructTextoA)struct.getValor()).getContenido();
break;
case "textoB": RESULT = ((StructTextoB)struct.getValor()).getContenido();
```

```
break;
default: Principal.errores.add(new Token("NO EXISTE FUNCION PARA ESTA ESTRUCTURA",
"ERROR SINTACTICO - FUNCION NO ENCONTRADA", 0, 0));
break;
}
tipoDato = "cadena";
:}
|getAlineacion AM
{:
Struct struct = Principal.retornarStruct(identificadorStruct);
switch(struct.getTipo()){
case "parrafo": RESULT = ((StructParrafo)struct.getValor()).getAlineacion();
break;
default: Principal.errores.add(new Token("NO EXISTE FUNCION PARA ESTA ESTRUCTURA",
"ERROR SINTACTICO - FUNCION NO ENCONTRADA", 0, 0));
break;
}
tipoDato = "cadena";
:}
|getPath AM
{:
Struct struct = Principal.retornarStruct(identificadorStruct);
switch(struct.getTipo()){
case "imagen": RESULT = ((StructImagen)struct.getValor()).getPath();
break;
default: Principal.errores.add(new Token("NO EXISTE FUNCION PARA ESTA ESTRUCTURA",
"ERROR SINTACTICO - FUNCION NO ENCONTRADA", 0, 0));
break;
}
tipoDato = "cadena";
:}
|getAncho AM
{:
Struct struct = Principal.retornarStruct(identificadorStruct);
switch(struct.getTipo()){
case "imagen": RESULT = ((StructImagen)struct.getValor()).getAncho();
break;
default: Principal.errores.add(new Token("NO EXISTE FUNCION PARA ESTA ESTRUCTURA",
"ERROR SINTACTICO - FUNCION NO ENCONTRADA", 0, 0));
break;
}
tipoDato = "entero";
:}
|getAlto AM
{:
Struct struct = Principal.retornarStruct(identificadorStruct);
switch(struct.getTipo()){
```

```
case "imagen": RESULT = ((StructImagen)struct.getValor()).getAlto();
break;
default: Principal.errores.add(new Token("NO EXISTE FUNCION PARA ESTA ESTRUCTURA",
"ERROR SINTACTICO - FUNCION NO ENCONTRADA", 0, 0));
break;
}
tipoDato = "entero";
:}
|getTexto AM
{:
Struct struct = Principal.retornarStruct(identificadorStruct);
switch(struct.getTipo()){
case "boton": RESULT = ((StructBoton)struct.getValor()).getTexto();
break;
default: Principal.errores.add(new Token("NO EXISTE FUNCION PARA ESTA ESTRUCTURA",
"ERROR SINTACTICO - FUNCION NO ENCONTRADA", 0, 0));
break;
}
tipoDato = "cadena";
:}
;

AI::= setContenido parentesisAbre AJ:a parentesisCierra puntoYComa
{:
switch(Principal.retornarStruct(identificadorStruct).getTipo()){
case "parrafo":
StructParrafo strParrafo = (StructParrafo)
(Principal.structs.get(Principal.indiceStruct(identificadorStruct)).getValor());
strParrafo.setContenido(a);
Principal.structs.set(Principal.indiceStruct(identificadorStruct), new Struct("parrafo",
identificadorStruct, strParrafo));
identificadorStruct = "";
break;
case "textoA":
StructTextoA strTextoA = (StructTextoA)
(Principal.structs.get(Principal.indiceStruct(identificadorStruct)).getValor());
strTextoA.setContenido(a);
Principal.structs.set(Principal.indiceStruct(identificadorStruct), new Struct("textoA",
identificadorStruct, strTextoA));
identificadorStruct = "";
break;
case "textoB":
StructTextoB strTextoB = (StructTextoB)
(Principal.structs.get(Principal.indiceStruct(identificadorStruct)).getValor());
strTextoB.setContenido(a);
Principal.structs.set(Principal.indiceStruct(identificadorStruct), new Struct("textoB",
identificadorStruct, strTextoB));
```

```
identificadorStruct = "";
break;
default: Principal.errores.add(new Token("NO EXISTE FUNCION PARA ESTA ESTRUCTURA",
"ERROR SINTACTICO - FUNCION NO ENCONTRADA", 0, 0));
break;
}
:}
|setAlineacion parentesisAbre AJ:a parentesisCierra puntoYComa
{:
switch(Principal.retornarStruct(identificadorStruct).getTipo()){
case "parrafo":
StructParrafo strParrafo = (StructParrafo)
(Principal.structs.get(Principal.indiceStruct(identificadorStruct)).getValor());
strParrafo.setAlineacion(a);
Principal.structs.set(Principal.indiceStruct(identificadorStruct), new Struct("parrafo",
identificadorStruct, strParrafo));
identificadorStruct = "";
break;
default: Principal.errores.add(new Token("NO EXISTE FUNCION PARA ESTA ESTRUCTURA",
"ERROR SINTACTICO - FUNCION NO ENCONTRADA", 0, 0));
break;
}
:}
|setPath parentesisAbre AJ:a parentesisCierra puntoYComa
{:
switch(Principal.retornarStruct(identificadorStruct).getTipo()){
case "imagen":
StructImagen strImagen = (StructImagen)
(Principal.structs.get(Principal.indiceStruct(identificadorStruct)).getValor());
strImagen.setPath(a);
Principal.structs.set(Principal.indiceStruct(identificadorStruct), new Struct("imagen",
identificadorStruct, strImagen));
identificadorStruct = "";
break;
default: Principal.errores.add(new Token("NO EXISTE FUNCION PARA ESTA ESTRUCTURA",
"ERROR SINTACTICO - FUNCION NO ENCONTRADA", 0, 0));
break;
}
:}
|setAlto parentesisAbre AK:a parentesisCierra puntoYComa
{:
switch(Principal.retornarStruct(identificadorStruct).getTipo()){
case "imagen":
StructImagen strImagen = (StructImagen)
(Principal.structs.get(Principal.indiceStruct(identificadorStruct)).getValor());
strImagen.setAlto(a);
```

```
Principal.structs.set(Principal.indiceStruct(identificadorStruct), new Struct("imagen",
identificadorStruct, strImagen));
identificadorStruct = "";
break;
default: Principal.errores.add(new Token("NO EXISTE FUNCION PARA ESTA ESTRUCTURA",
"ERROR SINTACTICO - FUNCION NO ENCONTRADA", 0, 0));
break;
}
:}
|setAncho parentesisAbre AK:a parentesisCierra puntoYComa
{:
switch(Principal.retornarStruct(identificadorStruct).getTipo()){
case "imagen":
StructImagen strImagen = (StructImagen)
(Principal.structs.get(Principal.indiceStruct(identificadorStruct)).getValor());
strImagen.setAncho(a);
Principal.structs.set(Principal.indiceStruct(identificadorStruct), new Struct("imagen",
identificadorStruct, strImagen));
identificadorStruct = "";
break;
default: Principal.errores.add(new Token("NO EXISTE FUNCION PARA ESTA ESTRUCTURA",
"ERROR SINTACTICO - FUNCION NO ENCONTRADA", 0, 0));
break;
}
:}
|setBorde parentesisAbre AL:a parentesisCierra puntoYComa
{:
switch(Principal.retornarStruct(identificadorStruct).getTipo()){
case "tabla":
StructTabla strTabla = (StructTabla)
(Principal.structs.get(Principal.indiceStruct(identificadorStruct)).getValor());
strTabla.setBorde(a);
Principal.structs.set(Principal.indiceStruct(identificadorStruct), new Struct("tabla",
identificadorStruct, strTabla));
identificadorStruct = "";
break;
default: Principal.errores.add(new Token("NO EXISTE FUNCION PARA ESTA ESTRUCTURA",
"ERROR SINTACTICO - FUNCION NO ENCONTRADA", 0, 0));
break;
}
:}
|setTexto parentesisAbre AJ:a parentesisCierra puntoYComa
{:
switch(Principal.retornarStruct(identificadorStruct).getTipo()){
case "boton":
StructBoton strBoton = (StructBoton)
(Principal.structs.get(Principal.indiceStruct(identificadorStruct)).getValor());
```

```
strBoton.setTexto(a);
Principal.structs.set(Principal.indiceStruct(identificadorStruct), new Struct("boton",
identificadorStruct, strBoton));
identificadorStruct = "";
break;
default: Principal.errores.add(new Token("NO EXISTE FUNCION PARA ESTA ESTRUCTURA",
"ERROR SINTACTICO - FUNCION NO ENCONTRADA", 0, 0));
break;
}
:}
|clickBoton parentesisAbre AJ:a parentesisCierra puntoYComa
{:
switch(Principal.retornarStruct(identificadorStruct).getTipo()){
case "boton":
StructBoton strBoton = (StructBoton)
(Principal.structs.get(Principal.indiceStruct(identificadorStruct)).getValor());
strBoton.click(a);
Principal.structs.set(Principal.indiceStruct(identificadorStruct), new Struct("boton",
identificadorStruct, strBoton));
identificadorStruct = "";
break;
default: Principal.errores.add(new Token("NO EXISTE FUNCION PARA ESTA ESTRUCTURA",
"ERROR SINTACTICO - FUNCION NO ENCONTRADA", 0, 0));
break;
}
:}
;

AJ::= textoEntreComillas:a
{:
RESULT = a;
:}
|dolar identificador:a
{:
RESULT = Principal.retornarValorCadenaVariable(a);
:}
|tipoAlineacion:a
{:
RESULT = a;
:}
|numeral identificador punto AH1
;

AK::= dolar identificador:a
{:
RESULT = Principal.retornarValorCadenaVariable(a);
:}
```

```
|V1:a
{:
RESULT = a.toString();
:}
;

AL::= dolar identificador:a
{:
RESULT = Principal.retornarValorCadenaVariable(a);
:}
|booleano:a
{:
RESULT = a;
:}
;


AM::= parentesisAbre parentesisCierra puntoYComa
;

AN::= dolar identificador:a
{:
contadorRepetidor++;
repetidores.add(Integer.parseInt(Principal.retornarValorCadenaVariable(a)));
:}
|entero:a
{:
contadorRepetidor++;
repetidores.add(Integer.parseInt(a));
:}
;

AO::= AO AP
|AP
;


AP::= echo W:a puntoYComa
{:
if(contadorRepetidor > 0){
for(int i = 0; i < repetidores.get(contadorRepetidor); i++)
Principal.txtConsola.append(a+"\n");
}else{
for(int i = 0; i < repetidores.get(0); i++)
Principal.txtConsola.append(a+"\n");
}
:}
```

|numeral U2 punto insertar parentesisAbre parentesisCierra puntoYComa
{:
if(contadorRepetidor > 0){
for(int i = 0; i < repetidores.get(contadorRepetidor); i++){
Struct struct = Principal.retornarStruct(identificadorStruct);
switch(struct.getTipo()){
case "parrafo": ((StructParrafo)struct.getValor()).insertar();
break;
case "textoA": ((StructTextoA)struct.getValor()).insertar();
break;
case "textoB": ((StructTextoB)struct.getValor()).insertar();
break;
case "imagen": ((StructImagen)struct.getValor()).insertar();
break;
case "tabla": ((StructTabla)struct.getValor()).insertar();
break;
case "boton": ((StructBoton)struct.getValor()).insertar();
break;
default: Principal.errores.add(new Token("ESTRUCTURA INVALIDA", "ERROR SINTACTICO - FUNCION NO ENCONTRADA", 0, 0));
break;
}
}
}else{
for(int i = 0; i < repetidores.get(0); i++){
Struct struct = Principal.retornarStruct(identificadorStruct);
switch(struct.getTipo()){
case "parrafo": ((StructParrafo)struct.getValor()).insertar();
break;
case "textoA": ((StructTextoA)struct.getValor()).insertar();
break;
case "textoB": ((StructTextoB)struct.getValor()).insertar();
break;
case "imagen": ((StructImagen)struct.getValor()).insertar();
break;
case "tabla": ((StructTabla)struct.getValor()).insertar();
break;
case "boton": ((StructBoton)struct.getValor()).insertar();
break;
default: Principal.errores.add(new Token("ESTRUCTURA INVALIDA", "ERROR SINTACTICO - FUNCION NO ENCONTRADA", 0, 0));
break;
}
}
}
:}
|AQ00 parentesisAbre AQ0 parentesisCierra llaveAbre AO1 llaveCierra

```
{:
contadorIf--;
:}
|AQ00 parentesisAbre AQ0 parentesisCierra llaveAbre AO1 llaveCierra controlElse
llaveAbre AO10 llaveCierra
{:
contadorIf--;
:}
|repetir parentesisAbre AN parentesisCierra llaveAbre AO llaveCierra
{:
contadorRepetidor--;
:}
;

AO1::= AO1 AP1
|AP1
;

AP1::= echo W:a puntoYComa
{:
if(contadorIf > 0){
if(condicionales.get(contadorIf)){
Principal.txtConsola.append(a+"\n");
}
}else{
if(condicionales.get(0)){
Principal.txtConsola.append(a+"\n");
}
}
:}
|numeral U2 punto insertar parentesisAbre parentesisCierra puntoYComa
{:
if(contadorIf > 0){
if(condicionales.get(contadorIf)){
Struct struct = Principal.retornarStruct(identificadorStruct);
switch(struct.getTipo()){
case "parrafo": ((StructParrafo)struct.getValor()).insertar();
break;
case "textoA": ((StructTextoA)struct.getValor()).insertar();
break;
case "textoB": ((StructTextoB)struct.getValor()).insertar();
break;
case "imagen": ((StructImagen)struct.getValor()).insertar();
break;
case "tabla": ((StructTabla)struct.getValor()).insertar();
break;
case "boton": ((StructBoton)struct.getValor()).insertar();
```

```
break;
default: Principal.errores.add(new Token("ESTRUCTURA INVALIDA", "ERROR SINTACTICO -
FUNCION NO ENCONTRADA", 0, 0));
break;
}
}
}else{
if(condicionales.get(0)){
Struct struct = Principal.retornarStruct(identificadorStruct);
switch(struct.getTipo()){
case "parrafo": ((StructParrafo)struct.getValor()).insertar();
break;
case "textoA": ((StructTextoA)struct.getValor()).insertar();
break;
case "textoB": ((StructTextoB)struct.getValor()).insertar();
break;
case "imagen": ((StructImagen)struct.getValor()).insertar();
break;
case "tabla": ((StructTabla)struct.getValor()).insertar();
break;
case "boton": ((StructBoton)struct.getValor()).insertar();
break;
default: Principal.errores.add(new Token("ESTRUCTURA INVALIDA", "ERROR SINTACTICO -
FUNCION NO ENCONTRADA", 0, 0));
break;
}
}
}
:}
|controlIf parentesisAbre AQ0 parentesisCierra llaveAbre AO1 llaveCierra
{:
contadorIf--;
:}
|controlIf parentesisAbre AQ0 parentesisCierra llaveAbre AO1 llaveCierra controlElse
llaveAbre AO10 llaveCierra
{:
contadorIf--;
:}
|U5 parentesisAbre AN parentesisCierra llaveAbre AO llaveCierra
{:
contadorRepetidor--;
:}
;

AO10::= AO10 AP10
|AP10
;
```

```
AP10::= echo W:a puntoYComa
{:
if(contadorIf > 0){
if(!condicionales.get(contadorIf)){
Principal.txtConsola.append(a+"\n");
}
}else{
if(condicionales.get(0)){
Principal.txtConsola.append(a+"\n");
}
}
:}
|numeral U2 punto insertar parentesisAbre parentesisCierra puntoYComa
{:
if(contadorIf > 0){
if(!condicionales.get(contadorIf)){
Struct struct = Principal.retornarStruct(identificadorStruct);
switch(struct.getTipo()){
case "parrafo": ((StructParrafo)struct.getValor()).insertar();
break;
case "textoA": ((StructTextoA)struct.getValor()).insertar();
break;
case "textoB": ((StructTextoB)struct.getValor()).insertar();
break;
case "imagen": ((StructImagen)struct.getValor()).insertar();
break;
case "tabla": ((StructTabla)struct.getValor()).insertar();
break;
case "boton": ((StructBoton)struct.getValor()).insertar();
break;
default: Principal.errores.add(new Token("ESTRUCTURA INVALIDA", "ERROR SINTACTICO -
FUNCION NO ENCONTRADA", 0, 0));
break;
}
}
}else{
if(condicionales.get(0)){
Struct struct = Principal.retornarStruct(identificadorStruct);
switch(struct.getTipo()){
case "parrafo": ((StructParrafo)struct.getValor()).insertar();
break;
case "textoA": ((StructTextoA)struct.getValor()).insertar();
break;
case "textoB": ((StructTextoB)struct.getValor()).insertar();
break;
case "imagen": ((StructImagen)struct.getValor()).insertar();
```

```
break;
case "tabla": ((StructTabla)struct.getValor()).insertar();
break;
case "boton": ((StructBoton)struct.getValor()).insertar();
break;
default: Principal.errores.add(new Token("ESTRUCTURA INVALIDA", "ERROR SINTACTICO -
FUNCION NO ENCONTRADA", 0, 0));
break;
}
}
}
:}
|controlIf parentesisAbre AQ0 parentesisCierra llaveAbre AO1 llaveCierra
{:
contadorIf--;
:}
|controlIf parentesisAbre AQ0 parentesisCierra llaveAbre AO1 llaveCierra controlElse
llaveAbre AO10 llaveCierra
{:
contadorIf--;
:}
|U5 parentesisAbre AN parentesisCierra llaveAbre AO llaveCierra
{:
contadorRepetidor--;
:}
;

AQ00::= controlIf
{:
condicionales = new ArrayList();
:}
;

AQ0::= AQ:a
{:
contadorIf++;
condicionales.add(a);
:}
;

AQ::= AQ:a and AQ:b
{:
if(a && b)
RESULT = true;
else
RESULT = false;
:}
```

```
|AQ:a or AQ:b
{:
if(a | b)
RESULT = true;
else
RESULT = false;
:}
|not AQ:a
{:
if(!a)
RESULT = true;
else
RESULT = false;
:}
|parentesisAbre AQ:a parentesisCierra
{:
RESULT = a;
:}
|AR:a
{:
RESULT = a;
:}
;

AR::= AS:a menorQue AS:b
{:
if(a < b)
RESULT = true;
else
RESULT = false;
:}
|AS:a mayorQue AS:b
{:
if(a > b)
RESULT = true;
else
RESULT = false;
:}
|AS:a menorIgual AS:b
{:
if(a <= b)
RESULT = true;
else
RESULT = false;
:}
|AS:a mayorIgual AS:b
{:
```

```
if(a >= b)
RESULT = true;
else
RESULT = false;
:}
|AS:a noIgual AS:b
{:
if(a != b)
RESULT = true;
else
RESULT = false;
:}
|AS:a igualIgual AS:b
{:
if(a == b)
RESULT = true;
else
RESULT = false;
:}
|AS:a
{:
RESULT = (a > 0)? true : false;
:}
;

AS::= dolar identificador:a
{:
Variable var = Principal.retornarVariable(a);
switch(var.getTipo()){
case "entero": RESULT = Double.parseDouble(var.getValor().toString());
break;
case "decimal": RESULT = Double.parseDouble(var.getValor().toString());
break;
case "booleano": RESULT = (Boolean.valueOf(var.getValor().toString())) ? 1.0 : 0;
break;
default: Principal.errores.add(new Token("TIPO DATO INVALIDO", "ERROR SINTACTICO -
TIPO DATO NO ESPERADO", 0, 0));
break;
}
:}
|booleano:a
{:
RESULT = (Boolean.valueOf(a)) ? 1.0 : 0;
:}
|V1:a
{:
RESULT = Double.parseDouble(a.toString());
```

:}
;

AH1::= getContenido AH2
|getAlineacion AH2
|getPath AH2
|getAncho AH2
|getAlto AH2
|getTexto AH2
;

AH2::= parentesisAbre parentesisCierra
;