**MANUAL TECNICO**

A continuación se le presenta el manual técnico del programa, con los requerimientos mínimos y el entorno de desarrollo así como las herramientas utilizadas para la creación de dicho programa. Puntos importantes a considerar en el código, para su buen funcionamiento y darle un soporte adecuado. En resumen, el programa es un traductor de texto, entrada en formato uweb y salida en formato html, de un lenguaje llamado Hscript con Struct.

**PLATAFORMA DE DESARROLLO**

El programa fue desarrollado bajo el sistema operativo Windows 10 de 64 bits, en el IDE Netbeans en su versión 8.0.2, con ayuda del editor de texto Atom, bajo el lenguaje de programación Java en su versión mas actualizada a enero de 2019, Java 8 201.

Para ejecutar el análisis del código, tanto léxico como sintáctico se utilizaron dos herramientas aceptadas por Java, Cup y JFlex. A continuación, se detalla un breve resumen de ambas.

**JFLEX**

Es una herramienta JAVA que permite actuar sobre aquellas cadenas de un fichero de texto que encajan en una expresión regular.

Flex es un meta compilador que permite generar rápidamente analizadores léxicos que se integran con Java.

Jlex es una herramienta desarrollada en Java que toma como entrada un archivo “entrada”, con este crea un archivo fuente java entrada.lex.java correspondiente al analizador léxico. En otras palabras, Jlex es un generador de analizadores léxicos. Los analizadores léxicos toman como entrada una cadena de caracteres y la convierten en una secuencia de tokens.

**CUP**

JAVA CUP es un parser-generador. Es un analizador sintáctico que construye un parser para gramáticas tipo LALR(1), con código de producción y asociación de fragmentos de código JAVA. Cuando una producción en particular es reconocida, se genera un archivo fuente Java, parser.java que contiene una clase parser, con un método Symbol parser().

Para ejecutar ambos programas, basta con crear dos archivos, un Lexer y otro parser, agregar las librerias correspondientes y escribir en un archivo con extensión “bat” lo siguiente:

**java -jar JLex/jflex-full-1.7.0.jar Lexer --encoding utf-8**

**java -jar Cup/java-cup-11b.jar -parser parser Parser**

y automáticamente crea dos nuevas clases de java, su lexer y su parser respectivamente, que serán los encargados de realizar el análisis completo del código.

**GRAMATICA UTILIZADA:**

**MAYUSCULAR = TERMINALES**

**MINUSCULULAS = NO TERMINALES**

S::= TEXTO R\_COMPI CIERRA\_ETIQUETA inner TEXTO SLASH R\_COMPI CIERRA\_ETIQUETA

inner::= inner t\_inner

| t\_inner

t\_inner::= TEXTO R\_CABECERA CIERRA\_ETIQUETA inner\_cabecera TEXTO SLASH R\_CABECERA CIERRA\_ETIQUETA {:

| TEXTO R\_CUERPO op\_cuerpo CIERRA\_ETIQUETA inner\_cuerpo TEXTO SLASH R\_CUERPO CIERRA\_ETIQUETA

op\_cuerpo::= R\_FONDO IGUAL CADENA

| {:/\*epsilon\*/:}

inner\_cabecera::= TEXTO R\_TITULO CIERRA\_ETIQUETA TEXTO SLASH R\_TITULO CIERRA\_ETIQUETA

inner\_cuerpo::= inner\_cuerpo:a tipo\_cuerpo

| tipo\_cuerpo

tipo\_cuerpo::=TEXTO:a R\_PARRAFO op\_parrafo:b CIERRA\_ETIQUETA TEXTO:c SLASH R\_PARRAFO CIERRA\_ETIQUETA

|TEXTO:a SLASH R\_SALTO CIERRA\_ETIQUETA

|TEXTO:a R\_TABLA op\_tabla:b CIERRA\_ETIQUETA lista\_tabla:c TEXTO:d SLASH R\_TABLA CIERRA\_ETIQUETA

|TEXTO:a R\_IMAGEN list\_img:b CIERRA\_ETIQUETA TEXTO:c SLASH R\_IMAGEN CIERRA\_ETIQUETA

|TEXTO:a R\_TEXTOA CIERRA\_ETIQUETA TEXTO:b SLASH R\_TEXTOA CIERRA\_ETIQUETA

|TEXTO:a R\_TEXTOB CIERRA\_ETIQUETA TEXTO:b SLASH R\_TEXTOB CIERRA\_ETIQUETA

|TEXTO R\_BOTON op\_boton CIERRA\_ETIQUETA TEXTO SLASH R\_BOTON CIERRA\_ETIQUETA

|TEXTO R\_ESPACIO CIERRA\_ETIQUETA inner\_cuerpo TEXTO SLASH R\_ESPACIO CIERRA\_ETIQUETA

|ABRE\_HS list\_hs R\_CIERRAHS

op\_parrafo::= ID:a IGUAL CADENA:b

|{: /\*epsilon\*/ :}

op\_boton::= ID:a IGUAL CADENA:b ID:c IGUAL CADENA:d

list\_img::= list\_img:a op\_imagen

| op\_imagen

op\_imagen::= ID:a IGUAL CADENA

| ID:a IGUAL NUM

op\_tabla::= ID:a IGUAL ID

| ID:a IGUAL CADENA

| {:/\*epsilon\*/:}

lista\_tabla::= lista\_tabla:a TEXTO:b R\_FILA CIERRA\_ETIQUETA lista\_fila:c TEXTO:d SLASH R\_FILA CIERRA\_ETIQUETA

| TEXTO:a R\_FILA CIERRA\_ETIQUETA lista\_fila:b TEXTO:c SLASH R\_FILA CIERRA\_ETIQUETA

lista\_fila::= lista\_fila:a tipo\_columna

| tipo\_columna

tipo\_columna::= TEXTO:a R\_COLUMNAC CIERRA\_ETIQUETA lista\_columna:b TEXTO:c SLASH R\_COLUMNAC CIERRA\_ETIQUETA

|TEXTO:a R\_COLUMNA CIERRA\_ETIQUETA lista\_columna:b TEXTO:c SLASH R\_COLUMNA CIERRA\_ETIQUETA

|TEXTO:a R\_COLUMNA CIERRA\_ETIQUETA TEXTO:c SLASH R\_COLUMNA CIERRA\_ETIQUETA

|TEXTO:a R\_COLUMNAC CIERRA\_ETIQUETA TEXTO:c SLASH R\_COLUMNAC CIERRA\_ETIQUETA

lista\_columna::= lista\_columna:a op\_columna

|op\_columna

op\_columna::= TEXTO:a R\_PARRAFO op\_parrafo:b CIERRA\_ETIQUETA TEXTO:c SLASH R\_PARRAFO CIERRA\_ETIQUETA

|TEXTO:a SLASH R\_SALTO CIERRA\_ETIQUETA

|TEXTO:a R\_IMAGEN list\_img:b CIERRA\_ETIQUETA TEXTO:c SLASH R\_IMAGEN CIERRA\_ETIQUETA

|TEXTO:a R\_BOTON op\_boton:b CIERRA\_ETIQUETA TEXTO:c SLASH R\_BOTON CIERRA\_ETIQUETA

|ABRE\_HS list\_hs:a R\_CIERRAHS

list\_hs::= list\_hs:a op\_hs

| op\_hs

op\_hs::= VAR1 VARIABLE:a IGUAL block\_echo:b PUNTOYCOMA

| block\_if

|R\_ECHO block\_echo:a PUNTOYCOMA

|block\_repetir

| VAR2 VARIABLE:a IGUAL var\_obj:b PUNTOYCOMA

| st\_set

| st\_insert

var\_obj::= R\_CREARPARRAFO ABRE lst\_parrafo:a CIERRA

| R\_CREARTEXTOA ABRE cad\_var:a CIERRA

| R\_CREARTEXTOB ABRE cad\_var:a CIERRA

|R\_CREARIMAGEN ABRE lst\_img:a CIERRA

|R\_CREARTABLA ABRE lst\_table:a CIERRA

|R\_CREARBOTON ABRE lst\_boton:a CIERRA

st\_set::= VAR2 VARIABLE:a CONCATENACION R\_SETALTO ABRE expr\_logica:b CIERRA PUNTOYCOMA

|VAR2 VARIABLE:a CONCATENACION R\_SETPATH ABRE block\_echo:b CIERRA PUNTOYCOMA

|VAR2 VARIABLE:a CONCATENACION R\_SETANCHO ABRE expr\_logica:b CIERRA PUNTOYCOMA

|VAR2 VARIABLE:a CONCATENACION R\_SETBORDE ABRE expr\_logica:b CIERRA PUNTOYCOMA

|VAR2 VARIABLE:a CONCATENACION R\_SETTEXTO ABRE block\_echo:b CIERRA PUNTOYCOMA

|VAR2 VARIABLE:a CONCATENACION R\_SETCONTENIDO ABRE block\_echo:b CIERRA PUNTOYCOMA

|VAR2 VARIABLE:a CONCATENACION R\_SETALINEACION ABRE block\_echo:b CIERRA PUNTOYCOMA

st\_insert::= VAR2 VARIABLE:a CONCATENACION R\_INSERTAR ABRE CIERRA PUNTOYCOMA

| VAR2 VARIABLE:a CONCATENACION R\_CLICKBOTON ABRE block\_echo:b CIERRA PUNTOYCOMA

st\_get::= VAR2 VARIABLE:a CONCATENACION R\_GETALTO ABRE CIERRA

|VAR2 VARIABLE:a CONCATENACION R\_GETPATH ABRE CIERRA

|VAR2 VARIABLE:a CONCATENACION R\_GETANCHO ABRE CIERRA

|VAR2 VARIABLE:a CONCATENACION R\_GETTEXTO ABRE CIERRA

|VAR2 VARIABLE:a CONCATENACION R\_GETCONTENIDO ABRE CIERRA

|VAR2 VARIABLE:a CONCATENACION R\_GETALINEACION ABRE CIERRA

lst\_parrafo::= cad\_var

| cad\_var:a COMA cad\_var:b

lst\_img::= cad\_var

| cad\_var:a COMA expr\_logica

| cad\_var:a COMA expr\_logica:b COMA expr\_logica

lst\_boton::= cad\_var

| cad\_var:a COMA cad\_var

lst\_table::= ls\_row

ls\_row::= ls\_row:a COMA CORCHETEIZQUIERDA ls\_col:b CORCHETEDERECHA

| CORCHETEIZQUIERDA ls\_col:a CORCHETEDERECHA

ls\_col::= ls\_col:a COMA block\_echo

| block\_echo

block\_echo::= block\_echo:a CONCATENACION e\_echo

| e\_echo

e\_echo::= expr\_logica

block\_if::= R\_IF ABRE expr\_logica:a CIERRA LLAVEIZQUIERDA list\_hsp:b LLAVEDERECHA

| R\_IF ABRE expr\_logica:a CIERRA LLAVEIZQUIERDA list\_hsp:b LLAVEDERECHA R\_ELSE LLAVEIZQUIERDA list\_hsp:c LLAVEDERECHA

block\_repetir::= R\_REPETIR ABRE expr\_rep:a CIERRA LLAVEIZQUIERDA list\_hsp:b LLAVEDERECHA

expr\_rep::=expr\_rep:a MAS expr\_rep

| expr\_rep:a MENOS expr\_rep

| expr\_rep:a POR expr\_rep

| expr\_rep:a ENTRE expr\_rep

| MENOS1 expr\_rep

%prec MENOS1

| ENTERO

|VAR1 VARIABLE

|st\_get:a

expr\_logica::= expr\_logica:a AND expr\_logica

| expr\_logica:a OR expr\_logica

|NOT expr\_logica

|expr

expr ::=expr:a IGUAL\_LOGICO expr

|expr:a DESIGUAL expr

|expr:a MAYOR expr

|expr:a MAYOR\_IGUAL expr

|expr:a MENOR expr

|expr:a MENOR\_IGUAL expr

|expr:a MAS expr

| expr:a MENOS expr

| expr:a POR expr

| expr:a ENTRE expr

| MENOS expr

%prec MENOS1

| ENTERO

|VAR1 VARIABLE

|R\_TRUE

|R\_FALSE

|CADENA

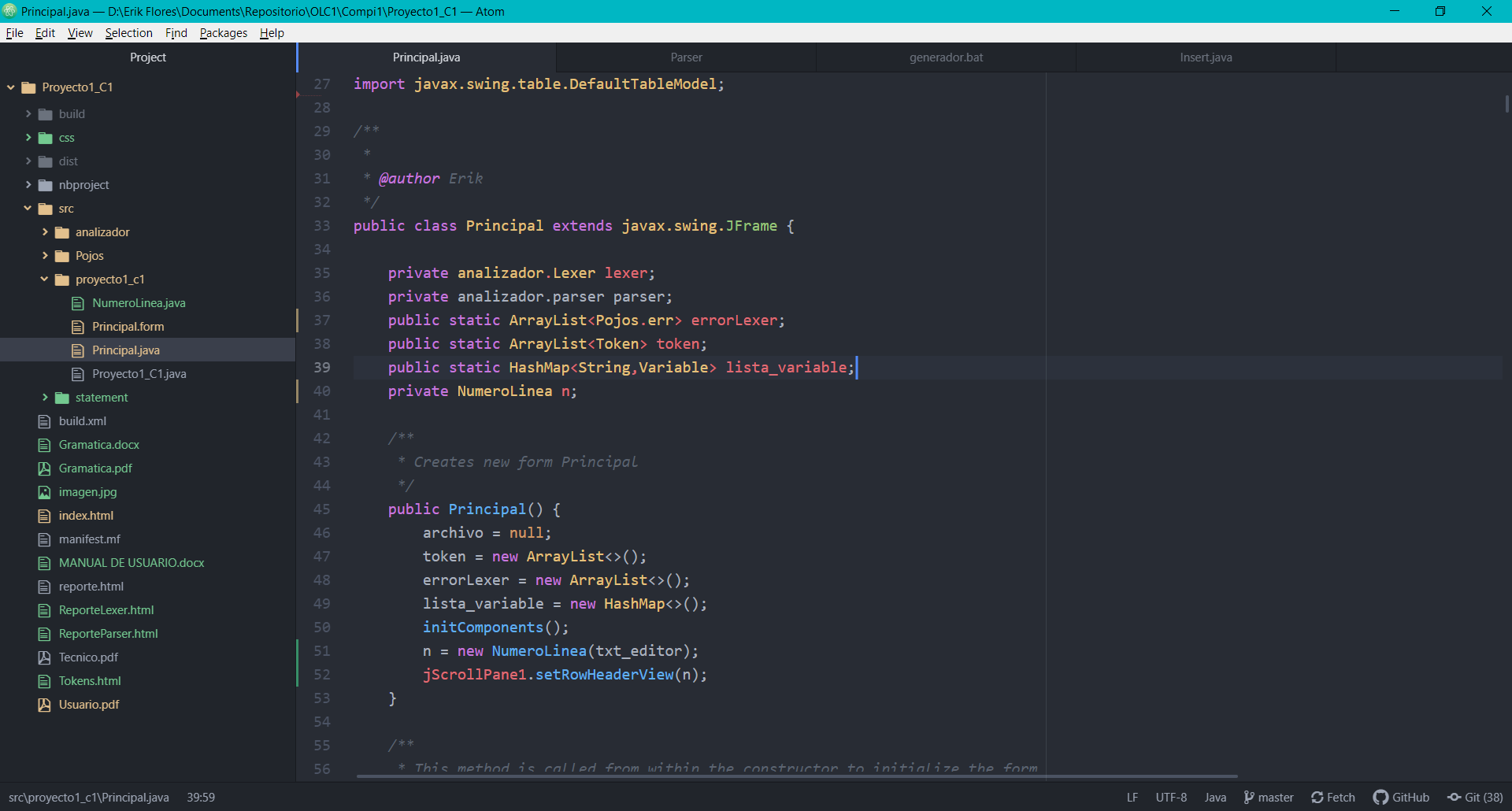
|st\_get

|ABRE expr\_logica:a CIERRA

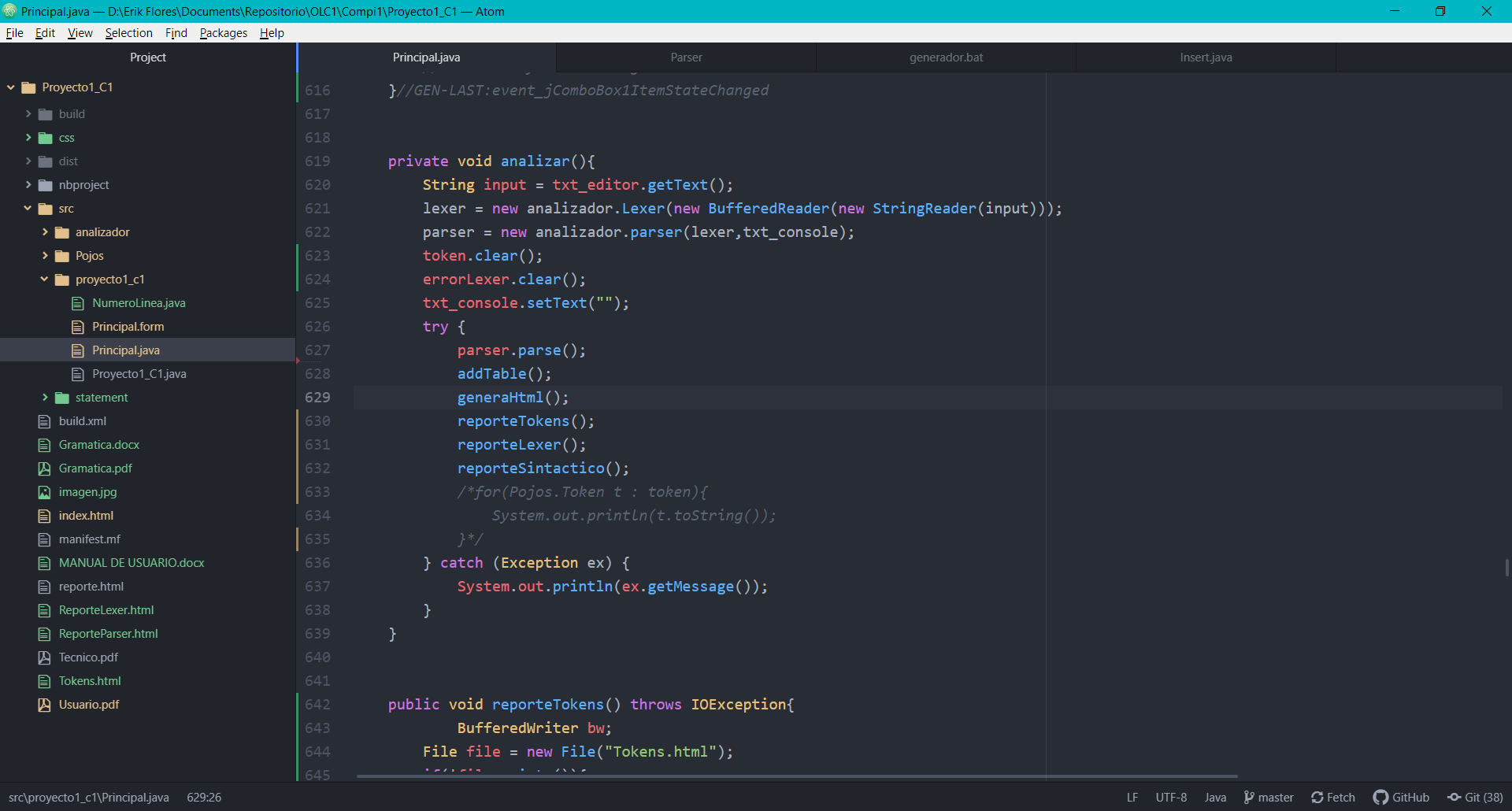
**PUNTOS IMPORTANTES DEL CODIGO**

Se muestran algunos de los puntos importantes del código.

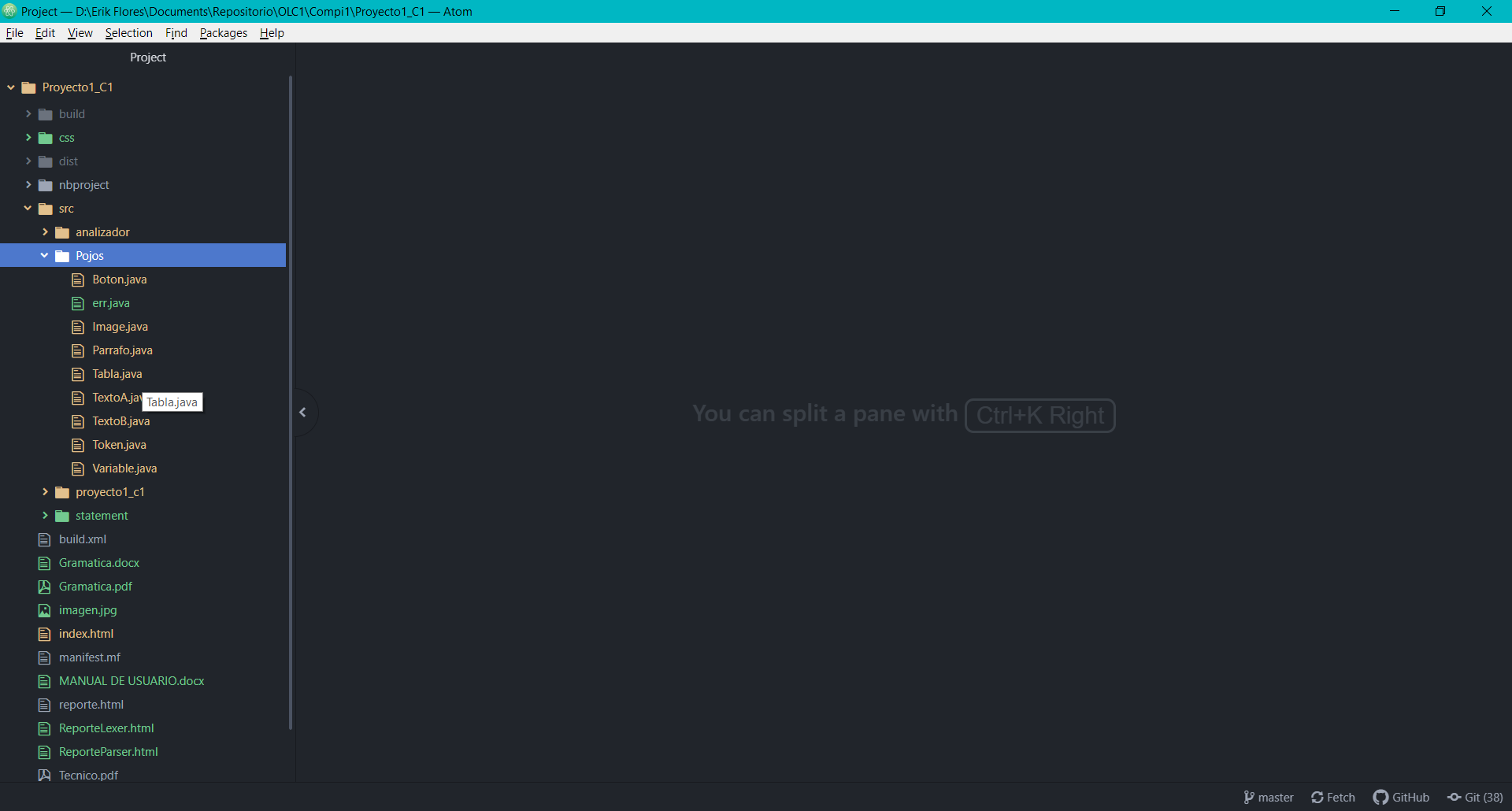
1. Listas globales estáticas, que guardan la lista de tokens, variables y la lista de errores, estas se encuentran en la clase Principal.



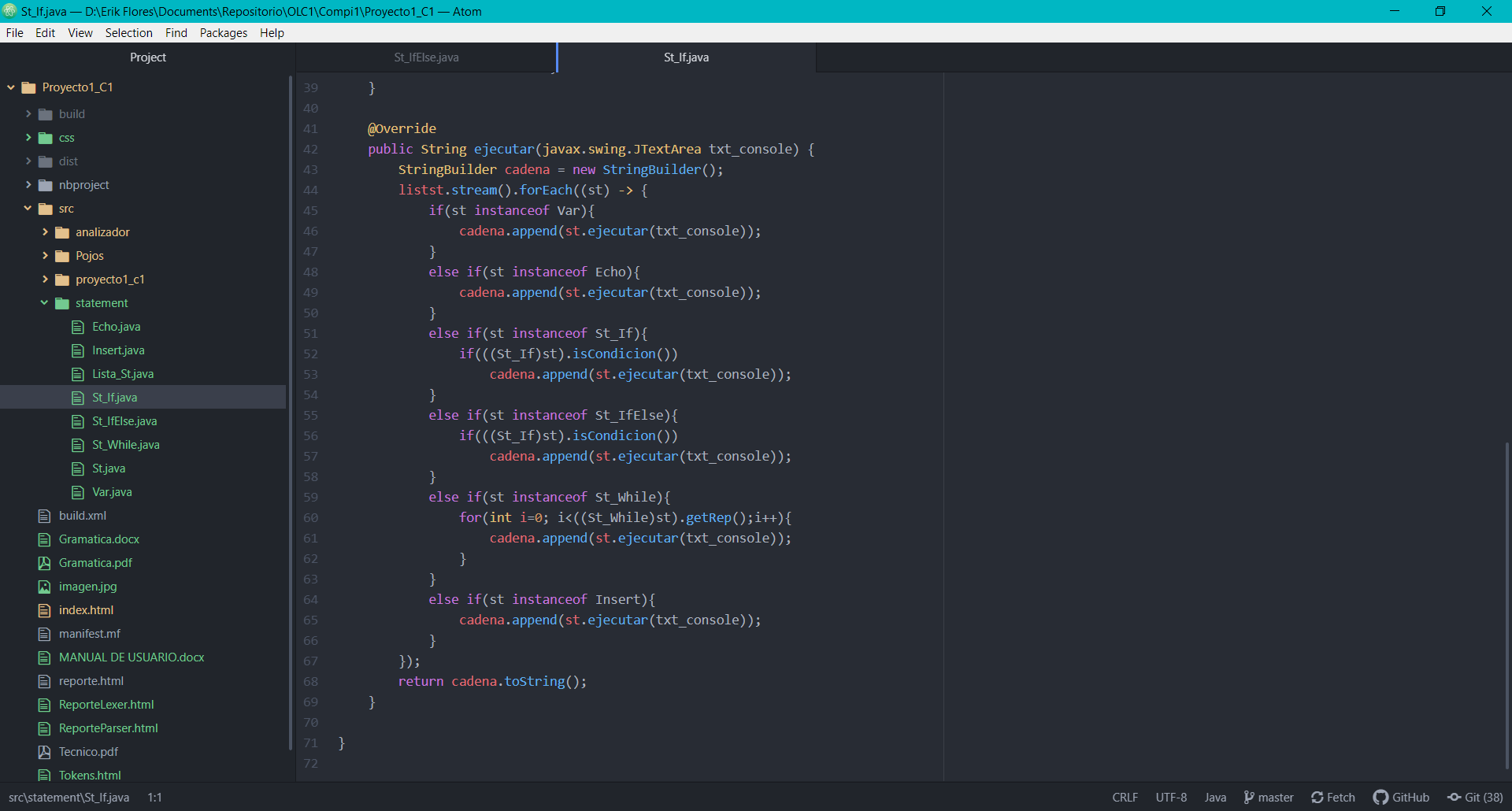
1. Método analizar, este método es el encargado de analizar todo el texto de entrada para su traducción a html, se encuentra en la clase Principal.



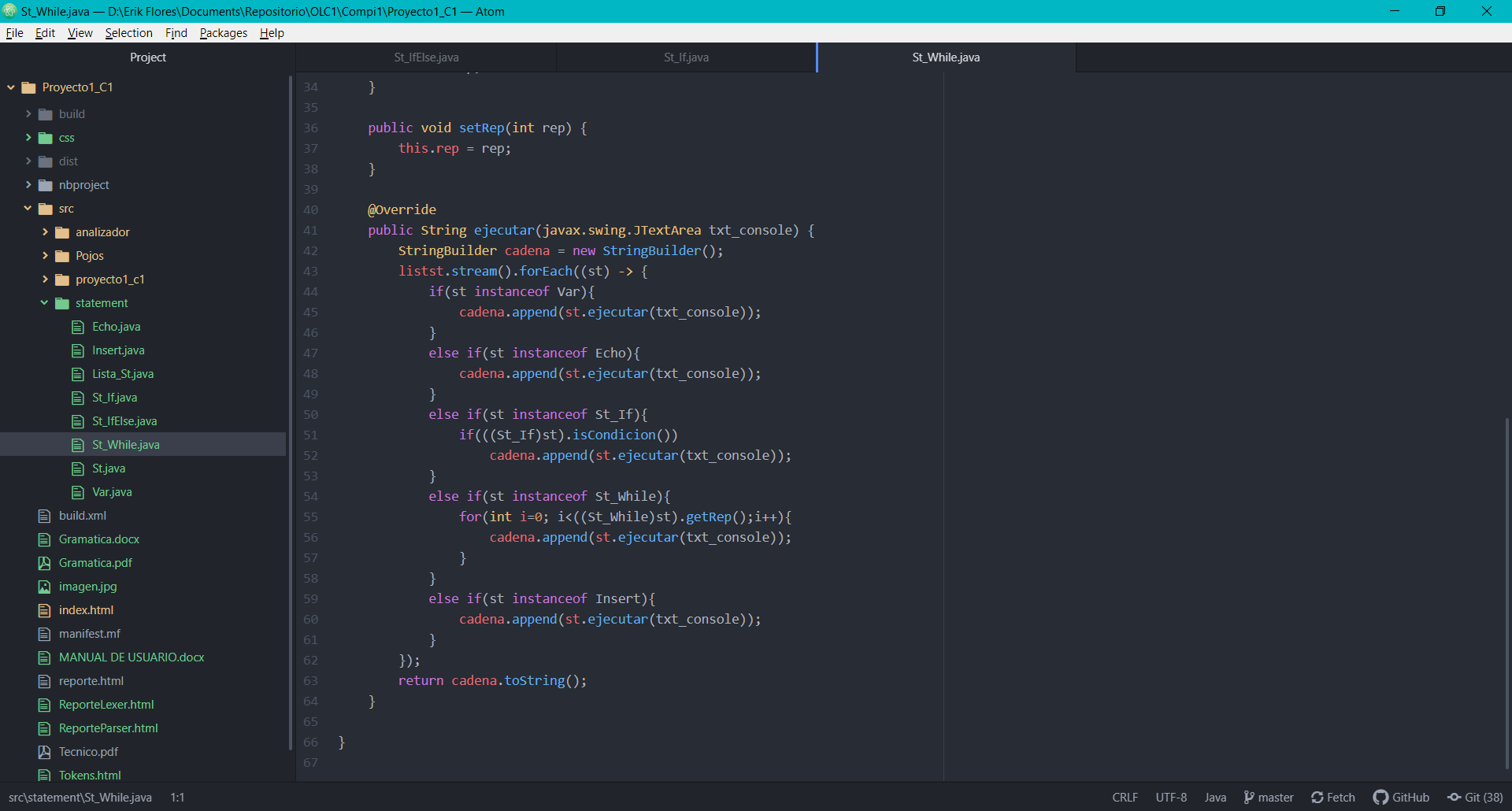
1. Se crearon clases con el nombre de cada etiqueta que se pueda insertar, como por ejemplo Tabla, TextoA, etc. Para su manejo en el lenguaje Hscript.



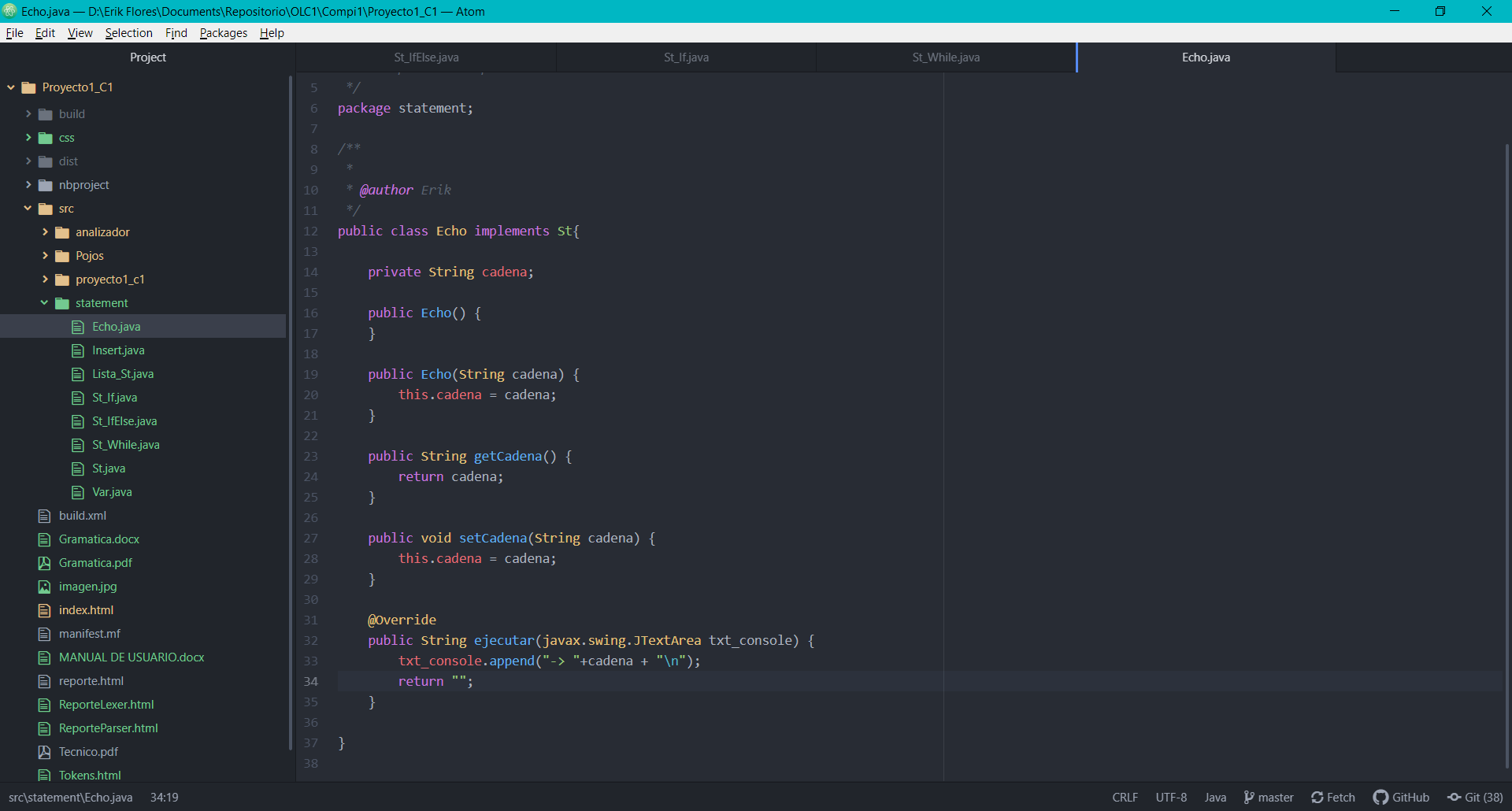
1. Un punto muy importante en la aplicación, se centra en el package statement, que tiene una interface denominada st, esta obtiene todos los bloques recibidos por hscript como el bloque if, echo, repetir entre otros, cada uno de estos bloques tiene a su vez una lista de bloques st, esto con el fin de ir agregando los pedazos de bloques que va recibiendo el archivo, y al finalizar realizar las validaciones respectivas y si se cumplen ejecutarlos, la interface tiene un método denominado ejecutar, que es al que se manda a llamar en caso de que se cumpla la condición. Un ejemplo sobre esto se muestra a continuación.



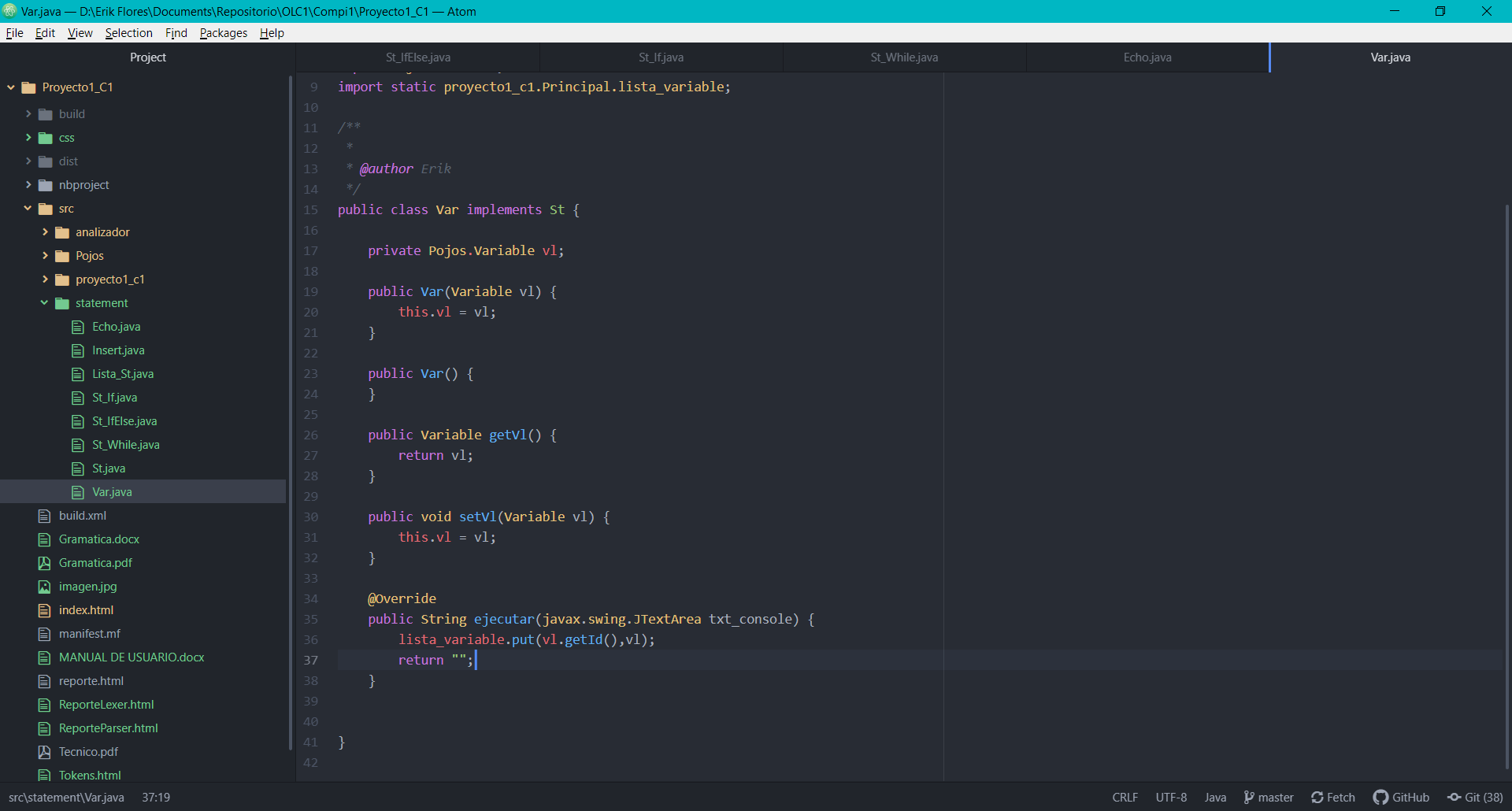
Bloque if.



Bloque repetir.



Bloque echo.



Bloque variable.

1. Ejecución de las cadenas a través de Cup, posteriormente como lo mencionado arriba, los bloques se van llamando dentro de si, siempre y cuando se vayan cumpliendo las condiciones.

