USAC
TRICENTENARIA
Universidad de San Carlos de Guatemala

MANUAL DE TÉCNICO:

Proyecto: No.1

Organizacion Lenguajes y Compiladores II “A”

Universidad De San Carlos De Guatemala
Centro Universitario De Occidente
División De Ciencias De La Ingeniería

Manual Tecnico

Creador Reproductor de Musica

OBJETIVOS

1.1. Objetivos Específicos

- Familiarizar al estudiante con la herramienta JFlex
- Familiarizar al estudiante con la herramienta CUP
- Aplicar conocimientos de analisis lexico y sintactico..

1.2. Objetivos Generales

- Aplicar el concepto de compiladores como una alternativa para la resolución de problemas.
- Que el estudiante entienda el funcionamiento de los analizadores de sintaxis ascendentes.
- Combinar la funcionalidad de JFlex y Cup en aplicaciones reales.
- Implementar las fases de análisis léxico, sintáctico y semántico de un compilador.
- Manejar errores léxicos, sintácticos y semánticos.

Presentación

El siguiente manual guiara a los usuarios que harán soporte al sistema, el cual les dará a conocer los requerimientos y la estructura para la construcción de el Creador y reproductor de música en java utilizando tecnologias como cup y jflex para el analisis de los archivos de entrada para el servidor y un cliente que sera una app de Movil desarrollada en Kotlin.

Requerimientos Mínimos de Software y Hardware

- **Windows:** Windows 10 (8u51 y superiores) Windows 8.x (escritorio) Windows 7 SP1 Windows Vista SP2 Windows Server 2008 R2 SP1 (64 bits) Windows Server 2012 y 2012 R2 (64 bits).
- **Memoria RAM:** 128 MB.
- **Espacio en disco:** 124 MB para JRE; 2 MB para Java Update.
- **Procesador Mínimo:** Pentium 2 a 266 MHz
- **Linux**

  Oracle Linux 5.5+1
  Oracle Linux 6.x (32 bits), 6.x (64 bits)2
  Oracle Linux 7.x (64 bits)2 (8u20 y superiores)
  Ubuntu Linux 12.04 LTS, 13.x
  Ubuntu Linux 14.x (8u25 y superiores)

2. HERRAMIENTAS UTILIZADAS PARA EL DESARROLLO

- Java JDK. 8
- jflex1.6.1
- cup 11
- Kotlin

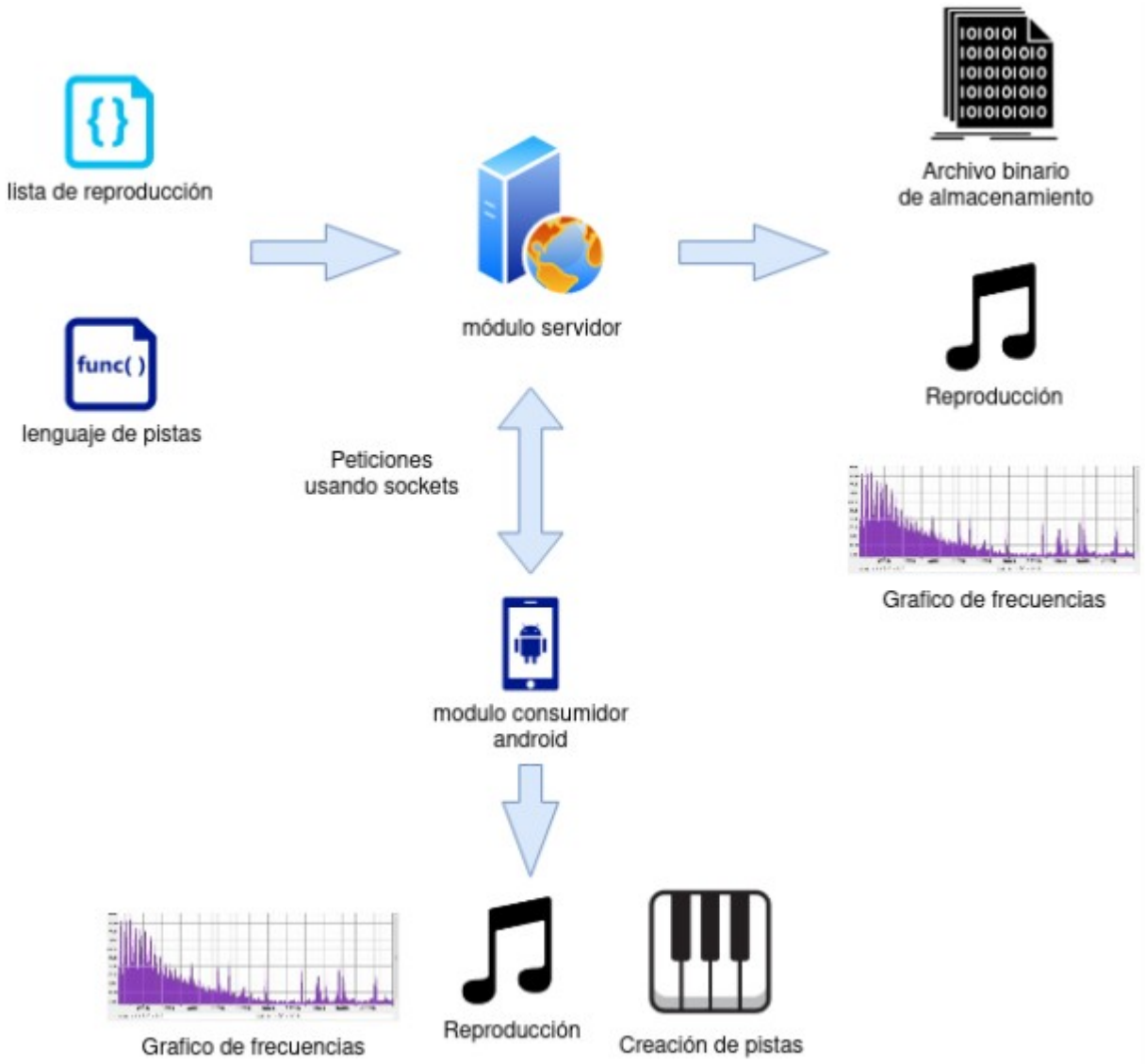
IDE:

Netbeans 8.2

SISTEMA OPERATIVO:

Linux Ubuntu 18.04

ARQUITECTURA DEL SISTEMA:



ESTRICTURA DE LOS ARCHIVOS DE ENTRADA:
ver en:

3. CONFIGURACIÓN

Ver el manual de instalación

4. Gramatica:

```
S ::=
       pista:s
       | ERRORSENTENCE LN S:s
          ;

pista::=   PISTA IDENTIFICADOR:id EXTIENDE extiende:ext LN
cuerpoPista:cPista
          | PISTA IDENTIFICADOR:id LN cuerpoPista:cPista
          | PISTA ERRORSENTENCE  EXTIENDE extiende:ext LN  cuerpoPista:cPista
          | PISTA ERRORSENTENCE LN cuerpoPista:cPista
       ;

extiende::= extiende:ext COMA IDENTIFICADOR:id
          |IDENTIFICADOR:id            {
;


cuerpoPista::=   cuerpoPista:cuerpo LN cuerpPistaP:cuerpoP LN
              |cuerpPistaP:cuerpoP

;


cuerpPistaP::=        MetodoFuncioDec:fun
                 | TAB declaracionVar:decVar
;



MetodoFuncioDec::=      KEEP MetodoFuncion:met
                           |MetodoFuncion:met
;



tipos::=    SENTERO
          |SDOBLE
          |SBOOLEAN
          |SCARACTER
          |SCADENA
;

tabList::=   TAB tabList
          |TAB
;
```

```
decrementosuma ::=    expresion:exp SUMASUMA LN
                    | expresion:exp MENOSMENOS LN
                    | expresion:exp   LN
;



sentencias ::=   tabList declaracionVar:sent  sentencias:sentLs
               | tabList llamadaAsignacionSentencias:sent LN sentencias:sentLs
               | tabList decrementosuma:sent LN sentencias:sentLs
               | tabList sentenciaIfElse:sent sentencias:sentLs
               | tabList switchsentencia:sent sentencias:sentLs
               | tabList forSentencia:sent sentencias:sentLs
               | mientrasSentecia:sent sentencias:sentLs
               | hacerMientrasSentencia:sent sentencias:sentLs
               | CONTINUESENTENCE:sent sentencias:sentLs
               | BREAKSENTENCE:sent sentencias:sentLs
               | RETURNSENTENCE:sent sentencias:sentLs
               | MENSAJE PARABRE expr:exp PARCIERRA sentencias:sentLs
               |
               | ERRORSENTENCE:sent LN sentencias:sentLs
               |MetodoFuncionNative:funNatva  sentencias:sentLs
               ;

listaSentencias::=   sentencias listaSentencias
                    |sentencias
;


// VARIABLES


declaraVar::=   VAR  tipos:tip listaID:idList
              |KEEP VAR tipos:tip listaID:idList
;

declaracionVar ::=   declaraVar:dec  IGUAL  expresion:exp
                    |declaraVar:dec
;


listaID::=  listaID:lsIds COMA IDENTIFICADOR:id
           |IDENTIFICADOR:id
;

llamadaAsignacionSentencias::=      IDENTIFICADOR:id IGUAL expresion:exp
```

```
;
//----------------------

/*  INSTRUCCIONES CONDICIONALES */

sentenciaIfElse ::=   IF PARABRE expresion:exp PARCIERRA LN sentencias:sent
LN
                  | IF PARABRE expresion:exp PARCIERRA LN sentencias:sent LN
ELSE LN sentencias:sentElse LN
                  | IF PARABRE expresion:exp PARCIERRA LN sentencias:sent LN
ELSE LN  sentenciaIfElse:sentElseIf LN
;
//------------
CONTINUESENTENCE ::= CONTINUAR LN
          ;
BREAKSENTENCE ::= SALIR LN
          ;
//RETURNSENTENCE ::=  return expresion:exp LN
 //              | return LN                  ;


//SWITCH


switchsentencia ::=  SWITCH PARABRE expresion:exp PARCIERRA LN
                  |SWITCH PARABRE expresion:exp PARCIERRA LN caselist:casels
defaultOp:def
                  | SWITCH PARABRE expresion:exp PARCIERRA LN caselist:casels
                  | SWITCH PARABRE expresion:exp PARCIERRA  defaultOp:def
;

caselist::= caselist:casesF caseFinal:casef
          | caseFinal:casef
       ;

caseFinal::=    CASE expresion:exp LN  sentencias:sent LN
              ;

defaultOp::=    DEFAULT LN sentencias:sent LN
;


//------------------------------
//CICLOS
//FOR
forSentencia ::= PARA PARABRE asignacionesFor:asign PUNTOYCOMA
expresion:expr PUNTOYCOMA decrementosuma:dec PARCIERRA
sentencias:sent LN
```

```
        ;

asignacionesFor::=   tipos:tipo IDENTIFICADOR:id IGUAL expresion:exp
                | IDENTIFICADOR:id IGUAL expresion:exp
        ;

//MIENTRAS WHILE
mientrasSentecia::= MIENTRAS PARABRE expresion:expr PARCIERRA LN
sentencias:list LN
        ;

//DOWHILE

hacerMientrasSentencia::= HACER sentencias:list LN MIENTRAS PARABRE
expresion:expr PARCIERRA
;



//declaracion METODO FUNCION


MetodoFuncion::=      tipos:tipo IDENTIFICADOR:id PARABRE
declaracionParametro:param PARCIERRA  LN sentencias:ls
                | tipos:tipo IDENTIFICADOR:id PARABRE PARCIERRA   LN
sentencias:ls
                | VOID  IDENTIFICADOR:id PARABRE declaracionParametro:params
PARCIERRA  LN sentencias:ls
                | VOID  IDENTIFICADOR:id PARABRE PARCIERRA    LN  sentencias:ls
                | VOID  PRINCIPAL:id PARABRE PARCIERRA   LN sentencias:ls
;

MetodoFuncionNative::=  REPRODUCIR PARABRE notas:nota COMA
expresion:exp1 COMA expresion:exp2 COMA expresion:exp3 PARCIERRA
                    |ESPERAR PARABRE num:ms COMA num:canal PARCIERRA
                    |ORDENAR PARABRE parSumarizar:exp COMA formaOrdenar:f
PARCIERRA
                    |SUMARIZAR PARABRE parSumarizar:sum PARCIERRA
                    |LONGITUD PARABRE parLongitud:sum PARCIERRA
;

formaOrdenar::=     ASCENDENTE
                |DESCENDENTE
                |PARES
                |IMPARES
                |PRIMOS
;
```

```
parSumarizar::=     IDENTIFICADOR:p
                  |declaracionesArr:arr
;

parLongitud::=       IDENTIFICADOR:p
                  |CADENA:cadena
                  |declaracionesArr:arr
;

num::=  ENTERO:num
       |DECIMAL:num
;

notas::=    IDENTIFICADOR:id
;



declaracionParametro ::=  parametro:param  COMA
declaracionParametro:params
                    | parametro:parametro
                    | sentenciaError COMA declaracionParametro:params
                    | sentenciaError
;

parametro::= tipos:tipo IDENTIFICADOR:id
           ;

//----------------------------------------
expresion::=     expr:exp
               |primitivas:exp
;

expr ::=    MENOS expresion:der                %prec UMENOS
          | NOT expresion:der
          | ESNULO expresion:der

          | expresion:der MAS expresion:iz
          | expresion:der MENOS expresion:iz
          | expresion:der ASTERISCO expresion:iz
          | expresion:der DIV expresion:iz
          | expresion:der POT expresion:iz
          | expresion:der MODULO expresion:iz

          | expresion:der MAYOR expresion:iz
          | expresion:der MENOR expresion:iz
          | expresion:der MAYORIGUAL expresion:iz
          | expresion:der MENORIGUAL expresion:iz
```

```
          | expresion:der IGUAL IGUAL expresion:iz
          | expresion:der NOIGUAL expresion:iz

          | expresion:der AND expresion:iz
          | expresion:der NAND expresion :iz
          | expresion:der OR expresion :iz
          | expresion:der NOR expresion:iz
          | expresion:der XOR expresion:iz

          | PARABRE expresion:expr PARCIERRA
          | IDENTIFICADOR:id PARABRE listaExpresion:e PARCIERRA
          |IDENTIFICADOR:id PARABRE PARCIERRA
;

primitivas::=   IDENTIFICADOR:p
              | DECIMAL:p
              | ENTERO:p
              | CADENA:p
              | CHAR:p
              | BOOLF:p
              | BOOLT:p

          ;

listaExpresion::=         listaExpresion:lsexp COMA expresion:exp

                          |expresion:exp
;


ERRORSENTENCE ::=   ERRORSENTENCE FINALERROR
              | FINALERROR
          ;
FINALERROR ::=  error
          ;
```