Manual Técnico Boxworld

Herramientas de Desarrollo

Las herramientas (ide's, lenguaje de programación, servidor web, librerías, etc.) usadas para el desarrollo de las Aplicaciones Server (Desktop) y cliente (Android), se resumen en el siguiente listado:

Lenguaje: Java 20

Java Versión: "20-ea" 2023-03-21

Analizador Léxico: JFlex

Versión: 1.8.2

Sitio de descarga: https://jflex.de/download.html

Analizador Sintáctico: Cup

Versión: 0.11b

Sitio de descarga: http://www2.cs.tum.edu/projects/cup/

IDE Apache NetBeans IDE 12

Sistema Operativo

La aplicación web y cliente, fueron desarrolladas bajo el siguiente sistema operativo:

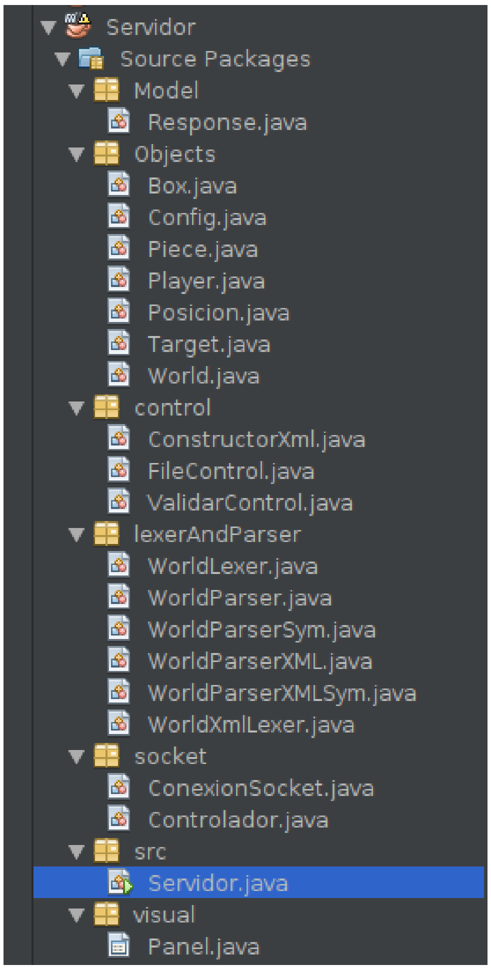
OS: Ubuntu Linux

Versión Kernel: x86_64 Linux ubuntu22.10

Organización del Código Fuente de la aplicación Server

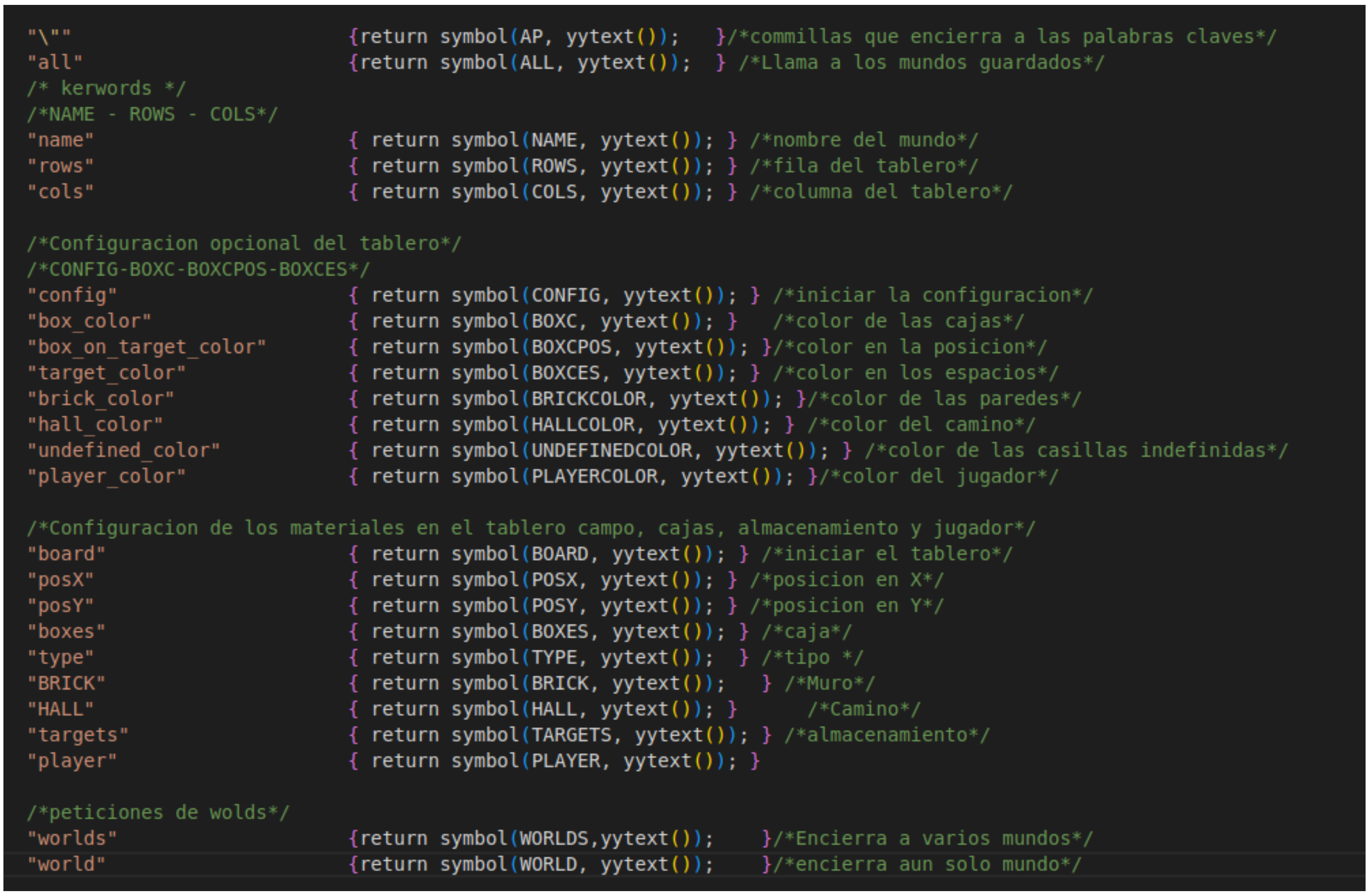
La aplicación Server, que como ya se indicó fue desarrollada en el lenguaje Java, presenta la siguiente estructura.

La parte web del proyecto, Servidor (nombre que recibe la aplicación Server) sigue la siguiente estructura:

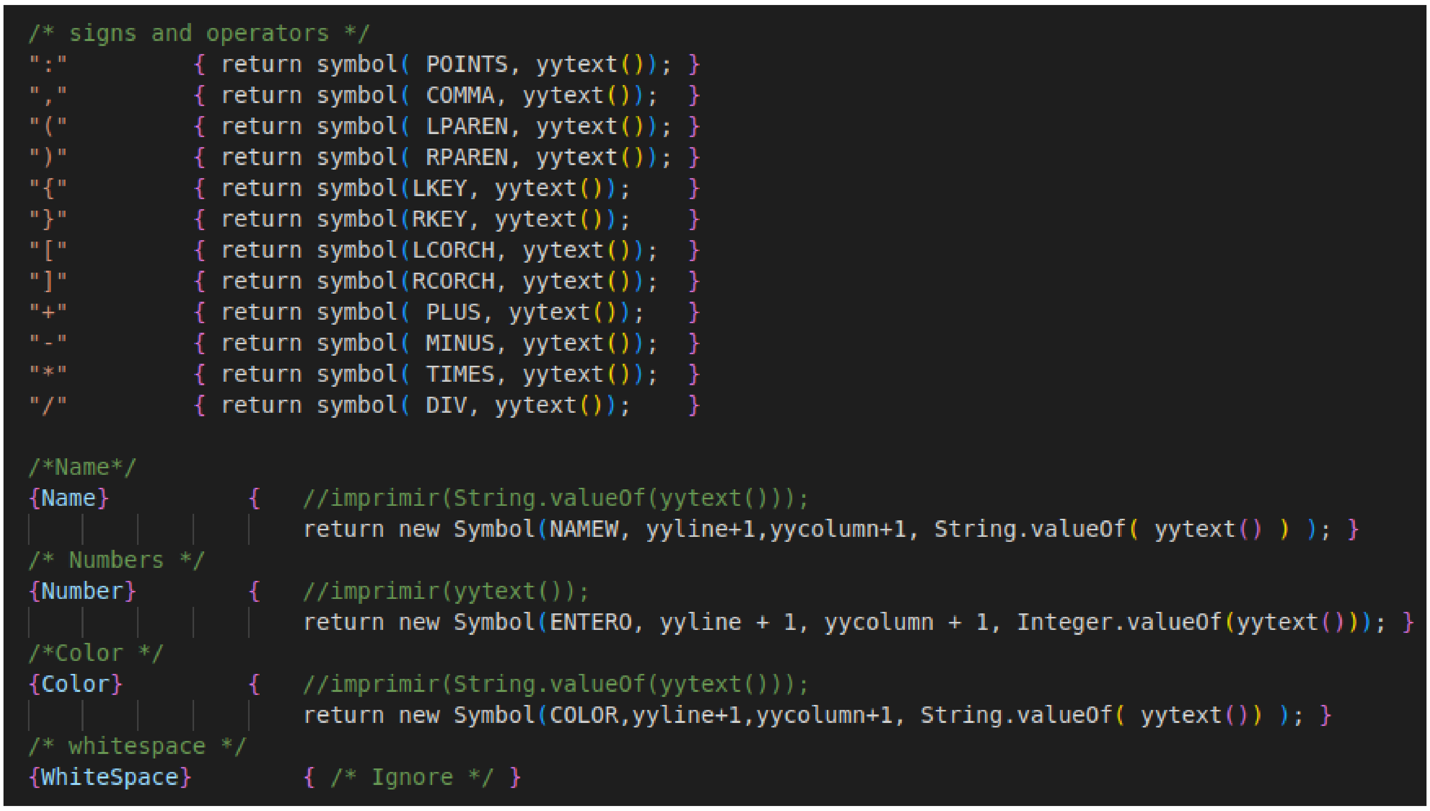


```
▼ Servidor
  ▼ Source Packages
    ▼ Model
        Response.java
    ▼ Objects
        Box.java
        Config.java
        Piece.java
        Player.java
        Posicion.java
        Target.java
        World.java
    ▼ control
        ConstructorXml.java
        FileControl.java
        ValidarControl.java
    ▼ lexerAndParser
        WorldLexer.java
        WorldParser.java
        WorldParserSym.java
        WorldParserXML.java
        WorldParserXMLSym.java
        WorldXmlLexer.java
    ▼ socket
        ConexionSocket.java
        Controlador.java
    ▼ src
        Servidor.java
    ▼ visual
        Panel.java
```

Reglas del analizador Léxico (Analizador de la parte del análisis del Json recibido)

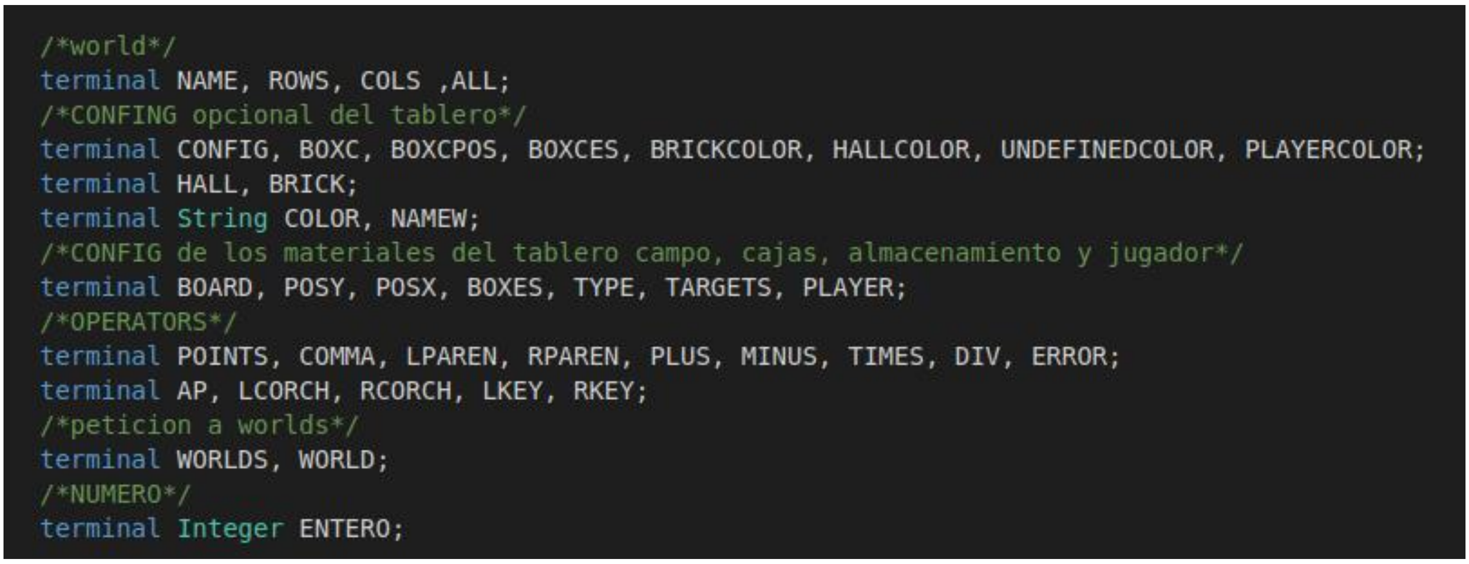


```
"\""                        {return symbol(AP, yytext());   }/*commillas que encierra a las palabras claves*/
"all"                       {return symbol(ALL, yytext());  } /*Llama a los mundos guardados*/
/* kerwords */
/*NAME - ROWS - COLS*/
"name"                      { return symbol(NAME, yytext()); } /*nombre del mundo*/
"rows"                      { return symbol(ROWS, yytext()); } /*fila del tablero*/
"cols"                      { return symbol(COLS, yytext()); } /*columna del tablero*/

/*Configuracion opcional del tablero*/
/*CONFIG-BOXC-BOXCPOS-BOXCES*/
"config"                    { return symbol(CONFIG, yytext()); } /*iniciar la configuracion*/
"box_color"                 { return symbol(BOXC, yytext()); }   /*color de las cajas*/
"box_on_target_color"       { return symbol(BOXCPOS, yytext()); }/*color en la posicion*/
"target_color"              { return symbol(BOXCES, yytext()); } /*color en los espacios*/
"brick_color"               { return symbol(BRICKCOLOR, yytext()); }/*color de las paredes*/
"hall_color"                { return symbol(HALLCOLOR, yytext()); } /*color del camino*/
"undefined_color"           { return symbol(UNDEFINEDCOLOR, yytext()); } /*color de las casillas indefinidas*/
"player_color"              { return symbol(PLAYERCOLOR, yytext()); }/*color del jugador*/

/*Configuracion de los materiales en el tablero campo, cajas, almacenamiento y jugador*/
"board"                     { return symbol(BOARD, yytext()); } /*iniciar el tablero*/
"posX"                      { return symbol(POSX, yytext()); } /*posicion en X*/
"posY"                      { return symbol(POSY, yytext()); } /*posicion en Y*/
"boxes"                     { return symbol(BOXES, yytext()); } /*caja*/
"type"                      { return symbol(TYPE, yytext());  } /*tipo */
"BRICK"                     { return symbol(BRICK, yytext());   } /*Muro*/
"HALL"                      { return symbol(HALL, yytext()); }      /*Camino*/
"targets"                   { return symbol(TARGETS, yytext()); } /*almacenamiento*/
"player"                    { return symbol(PLAYER, yytext()); }

/*peticiones de wolds*/
"worlds"                    {return symbol(WORLDS,yytext());    }/*Encierra a varios mundos*/
"world"                     {return symbol(WORLD, yytext());    }/*encierra aun solo mundo*/
```



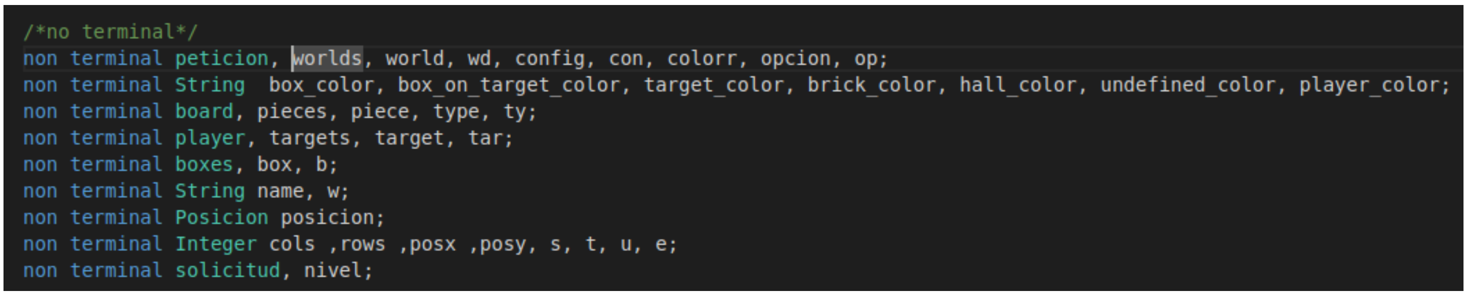
```
/* signs and operators */
":"         { return symbol( POINTS, yytext()); }
","         { return symbol( COMMA, yytext());  }
"("         { return symbol( LPAREN, yytext()); }
")"         { return symbol( RPAREN, yytext()); }
"{"         { return symbol(LKEY, yytext());    }
"}"         { return symbol(RKEY, yytext());    }
"["         { return symbol(LCORCH, yytext());  }
"]"         { return symbol(RCORCH, yytext());  }
"+"         { return symbol( PLUS, yytext());   }
"-"         { return symbol( MINUS, yytext());  }
"*"         { return symbol( TIMES, yytext());  }
"/"         { return symbol( DIV, yytext());    }

/*Name*/
{Name}          {   //imprimir(String.valueOf(yytext()));
                    return new Symbol(NAMEW, yyline+1,yycolumn+1, String.valueOf( yytext() ) ); }
/* Numbers */
{Number}        {   //imprimir(yytext());
                    return new Symbol(ENTERO, yyline + 1, yycolumn + 1, Integer.valueOf(yytext())); }
/*Color */
{Color}         {   //imprimir(String.valueOf(yytext()));
                    return new Symbol(COLOR,yyline+1,yycolumn+1, String.valueOf( yytext()) ); }
/* whitespace */
{WhiteSpace}        { /* Ignore */ }
```

Nodo Terminal



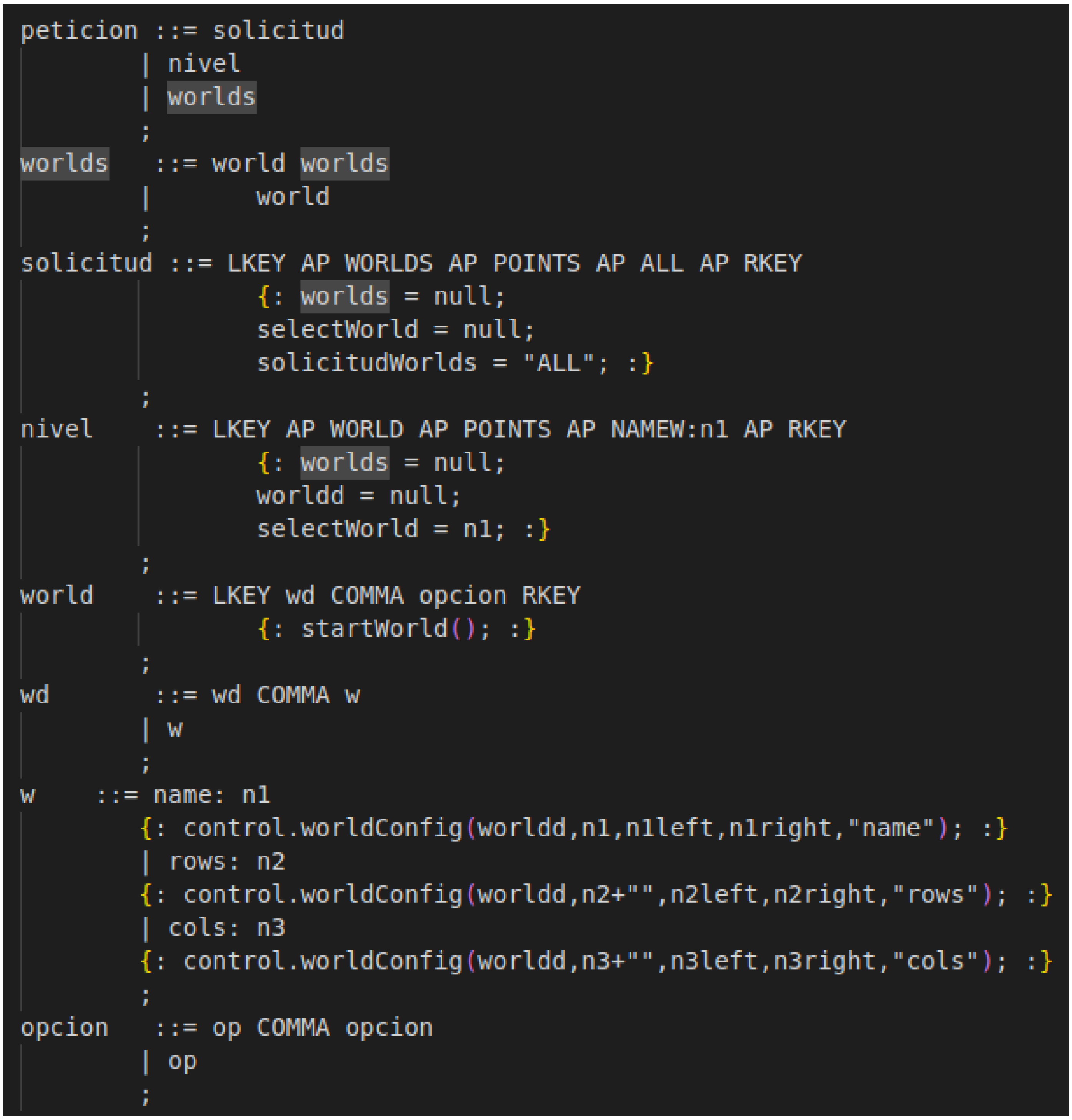
```
/*world*/
terminal NAME, ROWS, COLS ,ALL;
/*CONFING opcional del tablero*/
terminal CONFIG, BOXC, BOXCPOS, BOXCES, BRICKCOLOR, HALLCOLOR, UNDEFINEDCOLOR, PLAYERCOLOR;
terminal HALL, BRICK;
terminal String COLOR, NAMEW;
/*CONFIG de los materiales del tablero campo, cajas, almacenamiento y jugador*/
terminal BOARD, POSY, POSX, BOXES, TYPE, TARGETS, PLAYER;
/*OPERATORS*/
terminal POINTS, COMMA, LPAREN, RPAREN, PLUS, MINUS, TIMES, DIV, ERROR;
terminal AP, LCORCH, RCORCH, LKEY, RKEY;
/*peticion a worlds*/
terminal WORLDS, WORLD;
/*NUMERO*/
terminal Integer ENTERO;
```

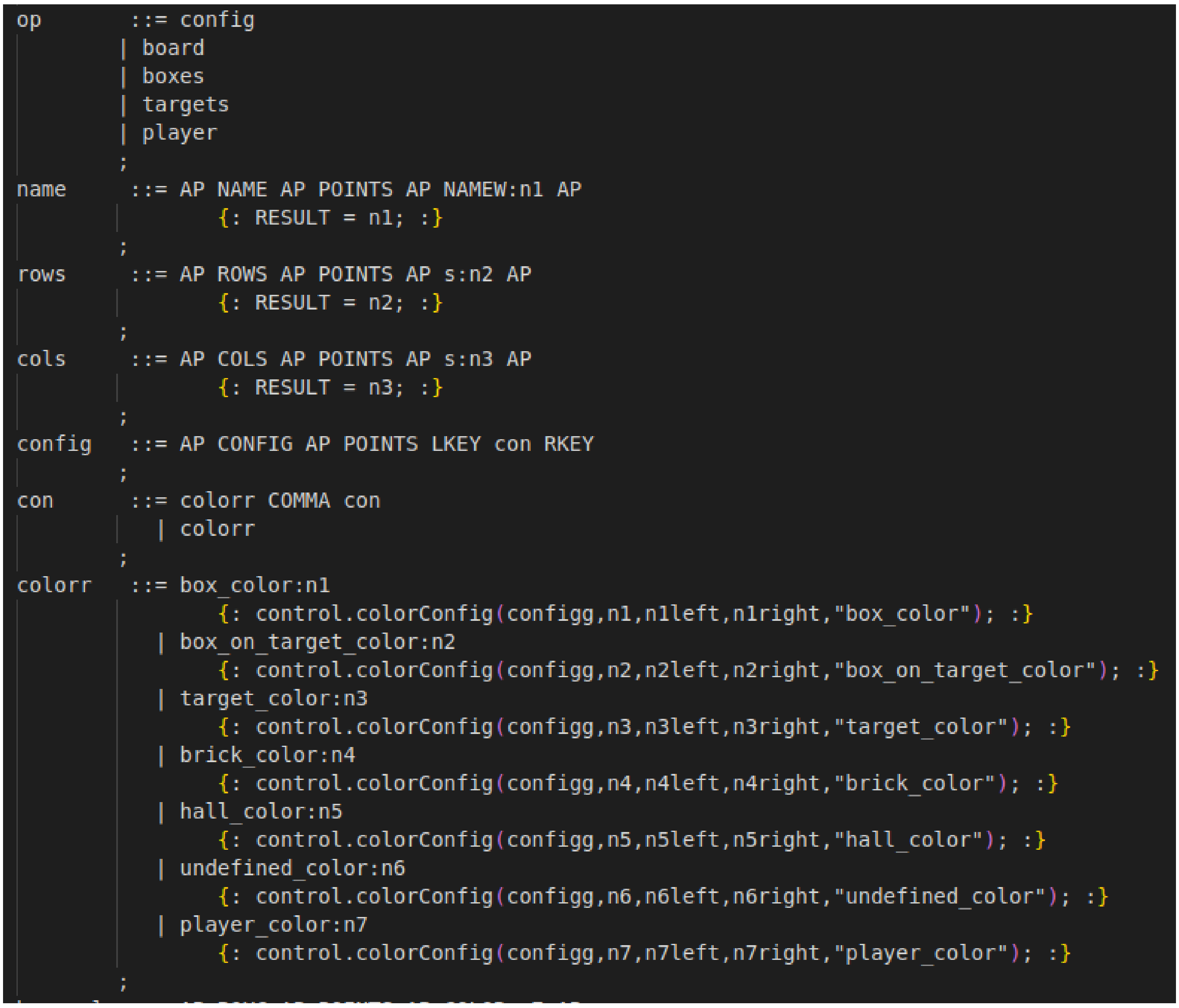
Nodo No Terminal



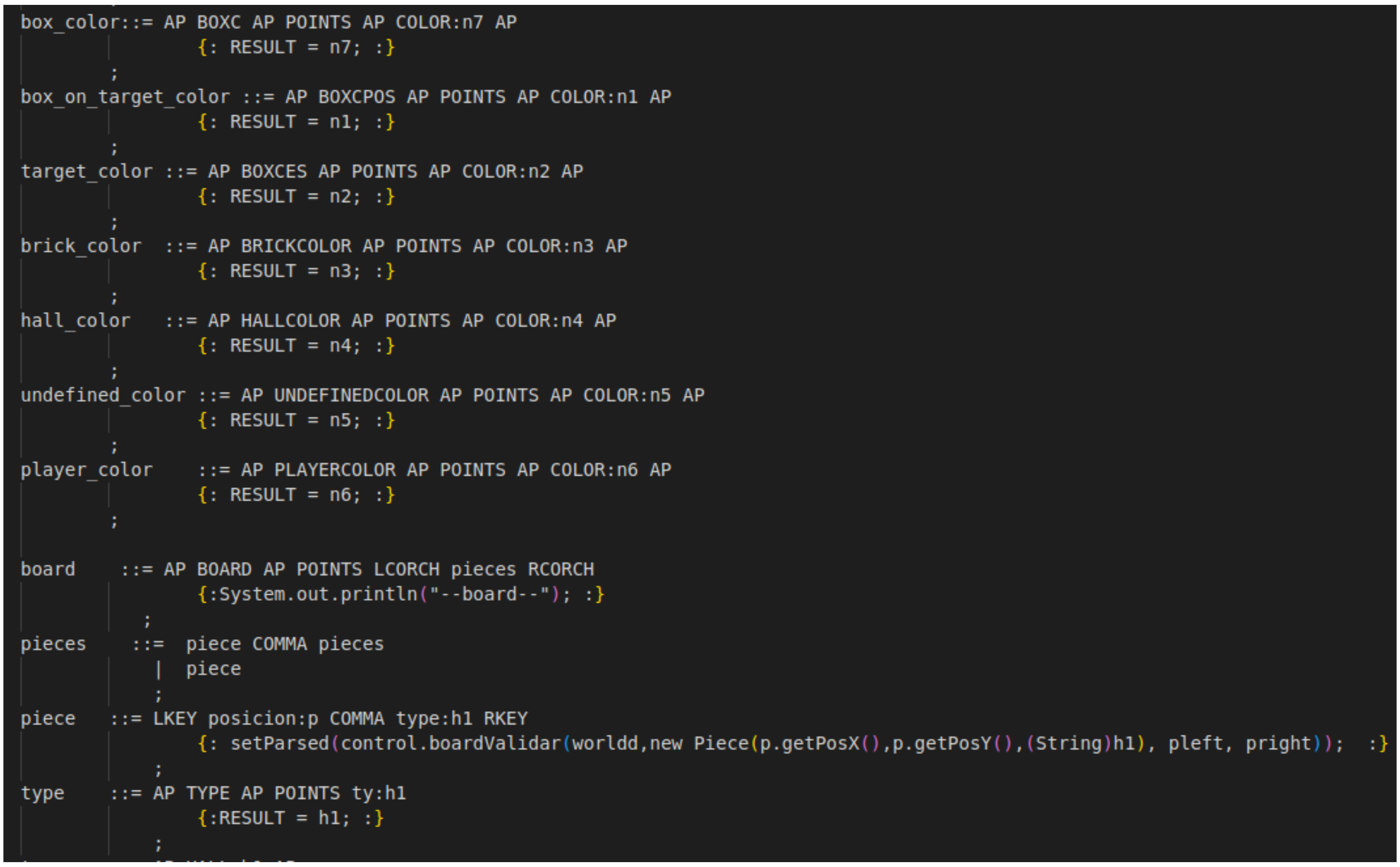
```
/*no terminal*/
non terminal peticion, worlds, world, wd, config, con, colorr, opcion, op;
non terminal String  box_color, box_on_target_color, target_color, brick_color, hall_color, undefined_color, player_color;
non terminal board, pieces, piece, type, ty;
non terminal player, targets, target, tar;
non terminal boxes, box, b;
non terminal String name, w;
non terminal Posicion posicion;
non terminal Integer cols ,rows ,posx ,posy, s, t, u, e;
non terminal solicitud, nivel;
```

Análisis Sintáctico

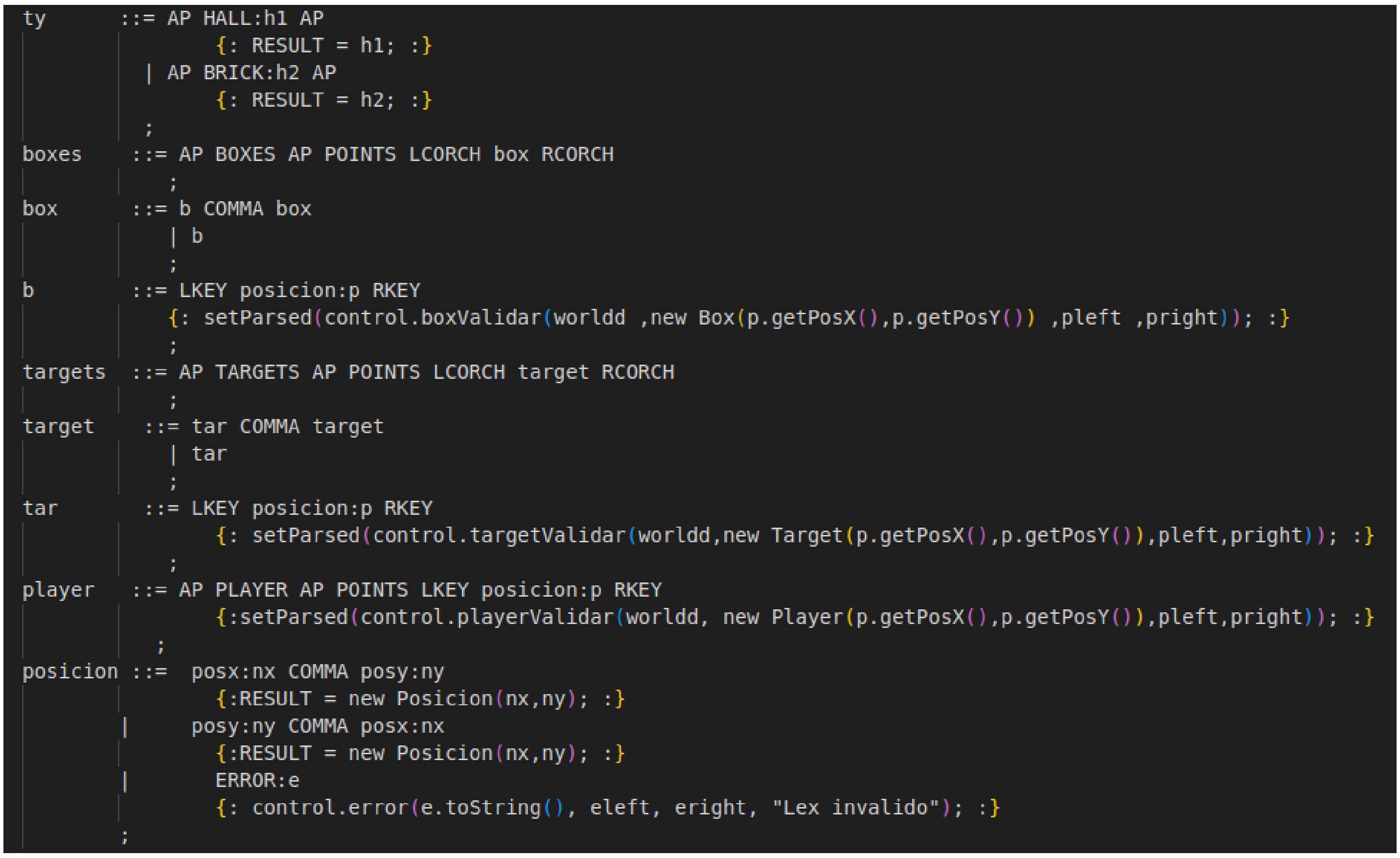

```
peticion ::= solicitud
          | nivel
          | worlds
          ;
worlds    ::= world worlds
          |       world
          ;
solicitud ::= LKEY AP WORLDS AP POINTS AP ALL AP RKEY
                {: worlds = null;
                selectWorld = null;
                solicitudWorlds = "ALL"; :}
          ;
nivel     ::= LKEY AP WORLD AP POINTS AP NAMEW:n1 AP RKEY
                {: worlds = null;
                worldd = null;
                selectWorld = n1; :}
          ;
world     ::= LKEY wd COMMA opcion RKEY
                {: startWorld(); :}
          ;
wd        ::= wd COMMA w
          | w
          ;
w    ::= name: n1
          {: control.worldConfig(worldd,n1,n1left,n1right,"name"); :}
          | rows: n2
          {: control.worldConfig(worldd,n2+"",n2left,n2right,"rows"); :}
          | cols: n3
          {: control.worldConfig(worldd,n3+"",n3left,n3right,"cols"); :}
          ;
opcion    ::= op COMMA opcion
          | op
          ;
```



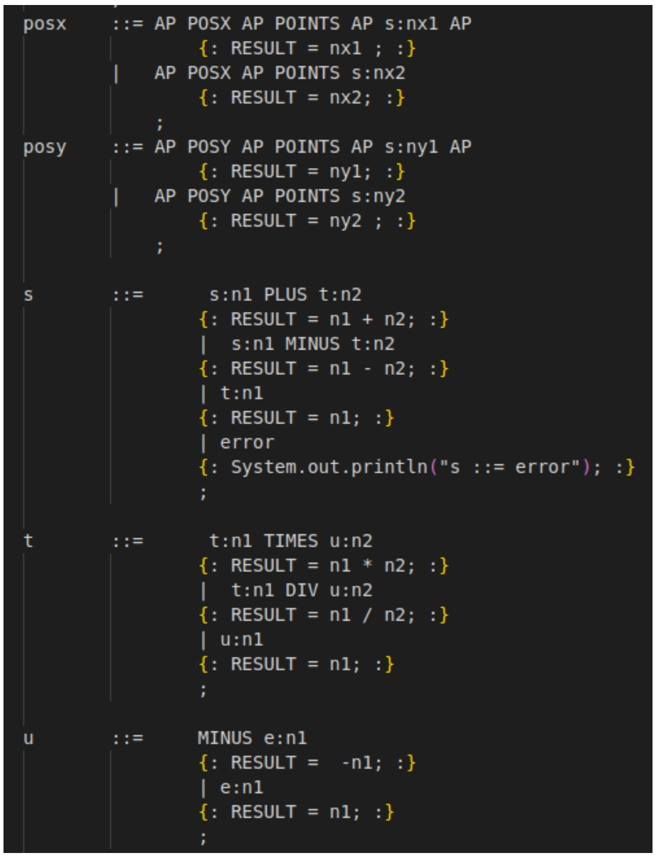
```
op       ::= config
        | board
        | boxes
        | targets
        | player
        ;
name     ::= AP NAME AP POINTS AP NAMEW:n1 AP
                {: RESULT = n1; :}
        ;
rows     ::= AP ROWS AP POINTS AP s:n2 AP
                {: RESULT = n2; :}
        ;
cols     ::= AP COLS AP POINTS AP s:n3 AP
                {: RESULT = n3; :}
        ;
config   ::= AP CONFIG AP POINTS LKEY con RKEY
        ;
con      ::= colorr COMMA con
           | colorr
        ;
colorr   ::= box_color:n1
                {: control.colorConfig(configg,n1,n1left,n1right,"box_color"); :}
           | box_on_target_color:n2
                {: control.colorConfig(configg,n2,n2left,n2right,"box_on_target_color"); :}
           | target_color:n3
                {: control.colorConfig(configg,n3,n3left,n3right,"target_color"); :}
           | brick_color:n4
                {: control.colorConfig(configg,n4,n4left,n4right,"brick_color"); :}
           | hall_color:n5
                {: control.colorConfig(configg,n5,n5left,n5right,"hall_color"); :}
           | undefined_color:n6
                {: control.colorConfig(configg,n6,n6left,n6right,"undefined_color"); :}
           | player_color:n7
                {: control.colorConfig(configg,n7,n7left,n7right,"player_color"); :}
        ;
```



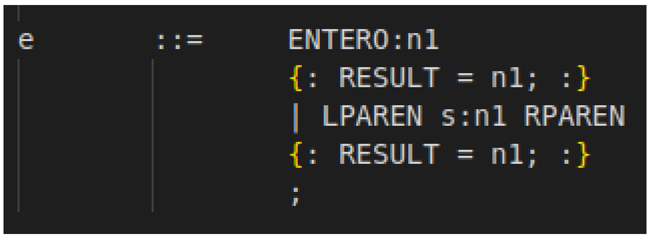
```
box_color::= AP BOXC AP POINTS AP COLOR:n7 AP
                {: RESULT = n7; :}
        ;
box_on_target_color ::= AP BOXCPOS AP POINTS AP COLOR:n1 AP
                {: RESULT = n1; :}
        ;
target_color ::= AP BOXCES AP POINTS AP COLOR:n2 AP
                {: RESULT = n2; :}
        ;
brick_color  ::= AP BRICKCOLOR AP POINTS AP COLOR:n3 AP
                {: RESULT = n3; :}
        ;
hall_color   ::= AP HALLCOLOR AP POINTS AP COLOR:n4 AP
                {: RESULT = n4; :}
        ;
undefined_color ::= AP UNDEFINEDCOLOR AP POINTS AP COLOR:n5 AP
                {: RESULT = n5; :}
        ;
player_color    ::= AP PLAYERCOLOR AP POINTS AP COLOR:n6 AP
                {: RESULT = n6; :}
        ;

board    ::= AP BOARD AP POINTS LCORCH pieces RCORCH
                {:System.out.println("--board--"); :}
           ;
pieces    ::=  piece COMMA pieces
            |  piece
            ;
piece   ::= LKEY posicion:p COMMA type:h1 RKEY
                {: setParsed(control.boardValidar(worldd,new Piece(p.getPosX(),p.getPosY(),(String)h1), pleft, pright));  :}
            ;
type    ::= AP TYPE AP POINTS ty:h1
                {:RESULT = h1; :}
            ;
```



```
ty       ::= AP HALL:h1 AP
                {: RESULT = h1; :}
           | AP BRICK:h2 AP
                {: RESULT = h2; :}
           ;
boxes     ::= AP BOXES AP POINTS LCORCH box RCORCH
             ;
box       ::= b COMMA box
             | b
             ;
b         ::= LKEY posicion:p RKEY
             {: setParsed(control.boxValidar(worldd ,new Box(p.getPosX(),p.getPosY()) ,pleft ,pright)); :}
             ;
targets   ::= AP TARGETS AP POINTS LCORCH target RCORCH
             ;
target     ::= tar COMMA target
             | tar
             ;
tar        ::= LKEY posicion:p RKEY
                {: setParsed(control.targetValidar(worldd,new Target(p.getPosX(),p.getPosY()),pleft,pright)); :}
             ;
player    ::= AP PLAYER AP POINTS LKEY posicion:p RKEY
                {:setParsed(control.playerValidar(worldd, new Player(p.getPosX(),p.getPosY()),pleft,pright)); :}
            ;
posicion ::=  posx:nx COMMA posy:ny
                {:RESULT = new Posicion(nx,ny); :}
        |     posy:ny COMMA posx:nx
                {:RESULT = new Posicion(nx,ny); :}
        |       ERROR:e
                {: control.error(e.toString(), eleft, eright, "Lex invalido"); :}
        ;
```



```
posx    ::= AP POSX AP POINTS AP s:nx1 AP
                {: RESULT = nx1 ; :}
        |   AP POSX AP POINTS s:nx2
                {: RESULT = nx2; :}
            ;
posy    ::= AP POSY AP POINTS AP s:ny1 AP
                {: RESULT = ny1; :}
        |   AP POSY AP POINTS s:ny2
                {: RESULT = ny2 ; :}
            ;

s       ::=      s:n1 PLUS t:n2
                {: RESULT = n1 + n2; :}
                |  s:n1 MINUS t:n2
                {: RESULT = n1 - n2; :}
                | t:n1
                {: RESULT = n1; :}
                | error
                {: System.out.println("s ::= error"); :}
                ;

t       ::=      t:n1 TIMES u:n2
                {: RESULT = n1 * n2; :}
                |  t:n1 DIV u:n2
                {: RESULT = n1 / n2; :}
                | u:n1
                {: RESULT = n1; :}
                ;

u       ::=     MINUS e:n1
                {: RESULT =  -n1; :}
                | e:n1
                {: RESULT = n1; :}
                ;
```



```
e       ::=     ENTERO:n1
                {: RESULT = n1; :}
                | LPAREN s:n1 RPAREN
                {: RESULT = n1; :}
                ;
```