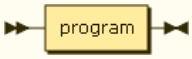
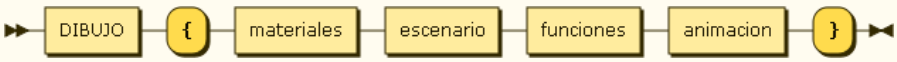
start_rule:




```
start_rule
        ::= program
```

no references

program:




```
program  ::= DIBUJO '{' materiales escenario funciones animacion '}'
```

referenced by:

- start_rule

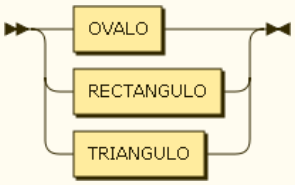
materiales:


```
materiales
        ::= MATERIALES '{' ( INTEGER_VALUE tipoFigura color LLAMADO NOMBRE_PROPIO DE expresion POR expresion ';' )* '}'
```

referenced by:

- program

tipoFigura:

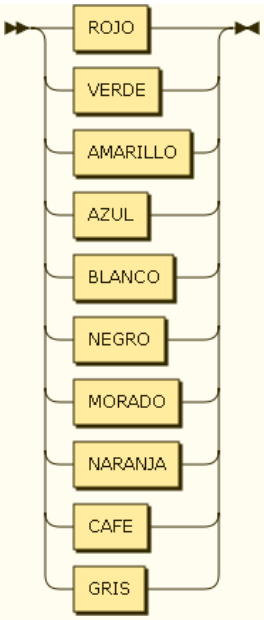



```
tipoFigura
        ::= OVALO
          | RECTANGULO
          | TRIANGULO
```

referenced by:

- materiales

color:

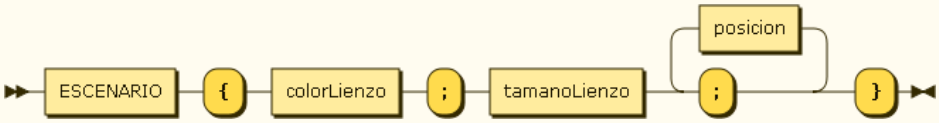



```
color    ::= ROJO
           | VERDE
           | AMARILLO
           | AZUL
           | BLANCO
           | NEGRO
           | MORADO
           | NARANJA
           | CAFE
           | GRIS
```

referenced by:

- cambioColor
- colorLienzo
- materiales

escenario:




```
escenario
         ::= ESCENARIO '{' colorLienzo ';' tamanoLienzo ';' ( posicion ';' )* '}'
```

referenced by:

- program

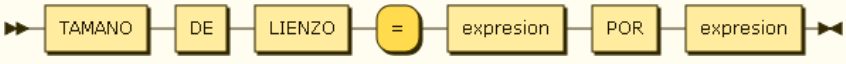
colorLienzo:


```
colorLienzo
         ::= COLOR DE LIENZO '=' color
```

referenced by:

- escenario

tamanoLienzo:

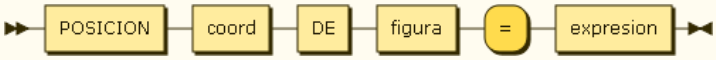



```
tamanoLienzo
         ::= TAMANO DE LIENZO '=' expresion POR expresion
```

referenced by:

- escenario

posicion:

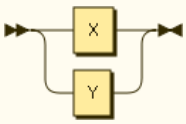



```
posicion ::= POSICION coord DE figura '=' expresion
```

referenced by:

- escenario
- instruccion

coord:

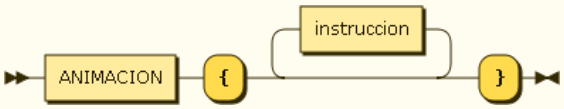



```
coord    ::= X
           | Y
```

referenced by:

- posicion

animacion:

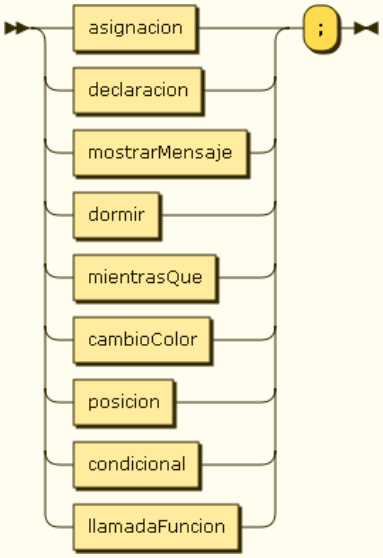



```
animacion
         ::= ANIMACION '{' instruccion* '}'
```

referenced by:

- program

instruccion:

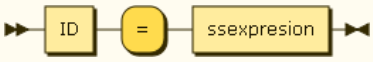



```
instruccion
         ::= ( asignacion | declaracion | mostrarMensaje | dormir | mientrasQue | cambioColor | posicion | condicional | llamadaFuncion ) ';'
```

referenced by:

- animacion
- condicional
- func
- mientrasQue

asignacion:

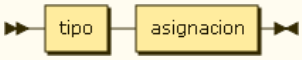



```
asignacion
        ::= ID '=' ssexpresion
```

referenced by:

- declaracion
- instruccion

declaracion:

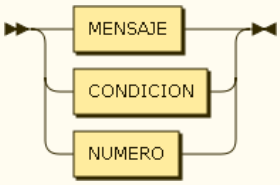



```
declaracion
        ::= tipo asignacion
```

referenced by:

- instruccion

tipo:

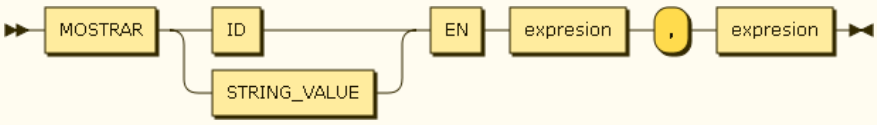



```
tipo    ::= MENSAJE
          | CONDICION
          | NUMERO
```

referenced by:

- declaracion
- func

mostrarMensaje:

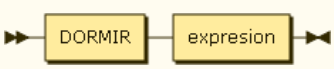



```
mostrarMensaje
        ::= MOSTRAR ( ID | STRING_VALUE ) EN expresion ',' expresion
```

referenced by:

- instruccion

dormir:

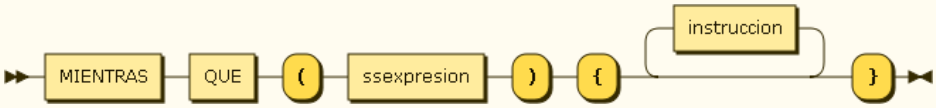



```
dormir   ::= DORMIR expresion
```

referenced by:

- instruccion

mientrasQue:

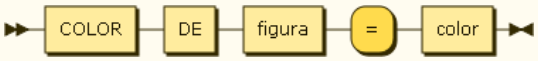



```
mientrasQue
        ::= MIENTRAS QUE '(' ssexpresion ')' '{' instruccion* '}'
```

referenced by:

- instruccion

cambioColor:

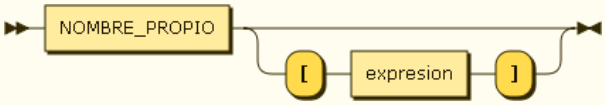



```
cambioColor
        ::= COLOR DE figura '=' color
```

referenced by:

- instruccion

figura:

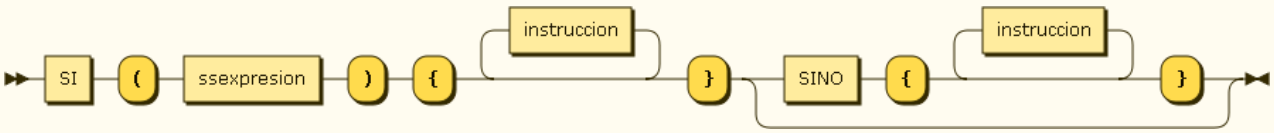



```
figura   ::= NOMBRE_PROPIO ( '[' expresion ']' )?
```

referenced by:

- cambioColor
- posicion

condicional:

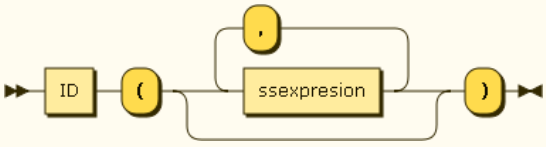



```
condicional
        ::= SI '(' ssexpresion ')' '{' instruccion* '}' ( SINO '{' instruccion* '}' )?
```

referenced by:

- instruccion

llamadaFuncion:

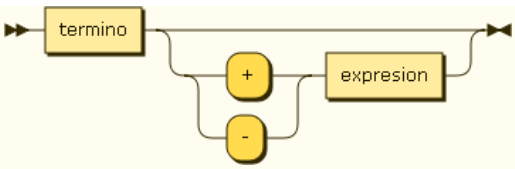



```
llamadaFuncion
        ::= ID '(' ( ssexpresion ( ',' ssexpresion )* )? ')'
```

referenced by:

- instruccion

expresion:

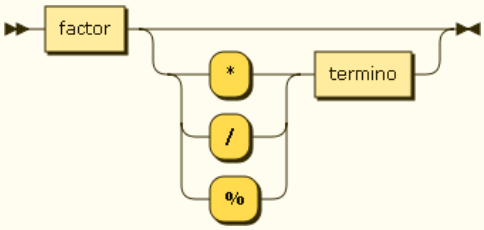



```
expresion
        ::= termino ( ( '+' | '-' ) expresion )?
```

referenced by:

- dormir
- expresion
- factor
- figura
- materiales
- mostrarMensaje
- posicion
- sexpresion
- tamanoLienzo

termino:

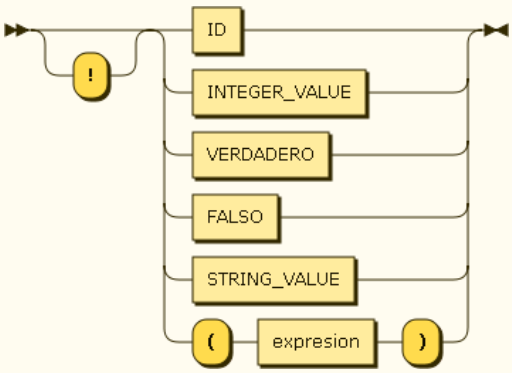



```
termino  ::= factor ( ( '*' | '/' | '%' ) termino )?
```

referenced by:

- expresion
- termino

factor:

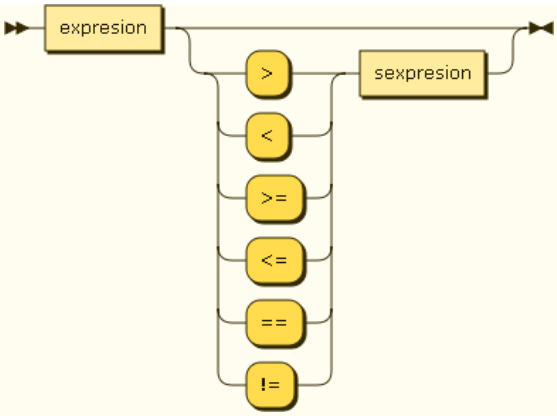



```
factor   ::= '!'? ( ID | INTEGER_VALUE | VERDADERO | FALSO | STRING_VALUE | '(' expresion ')' )
```

referenced by:

- termino

sexpresion:

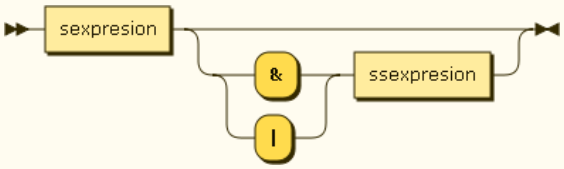



```
sexpresion
        ::= expresion ( ( '>' | '<' | '>=' | '<=' | '==' | '!=' ) sexpresion )?
```

referenced by:

- sexpresion
- ssexpresion

ssexpresion:

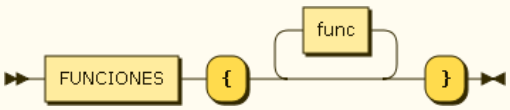



```
ssexpresion
        ::= sexpresion ( ( '&' | '|' ) ssexpresion )?
```

referenced by:

- asignacion
- condicional
- llamadaFuncion
- mientrasQue
- ssexpresion

funciones:

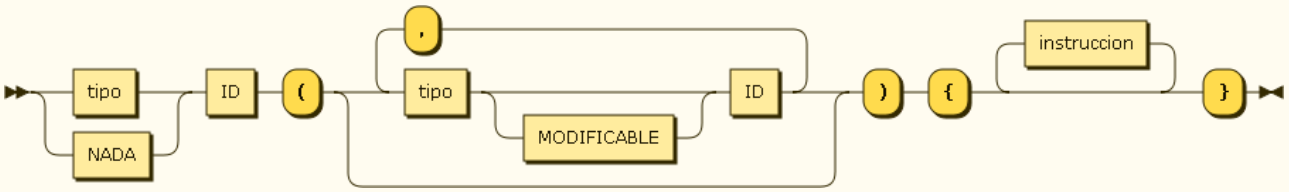



```
funciones
        ::= FUNCIONES '{' func* '}'
```

referenced by:

- program

func:




```
func    ::= ( tipo | NADA ) ID '(' ( tipo MODIFICABLE? ID ( ',' tipo MODIFICABLE? ID )* )? ')' '{' instruccion* '}'
```

referenced by:

- funciones

... generated by Railroad Diagram Generator