

no references

Diagram illustrating the structure of a variable declaration statement:

TYPE — id — = — EXPRESSION

The diagram shows a sequence of four components connected by horizontal lines: a yellow rectangular box labeled "TYPE", a yellow rounded rectangle labeled "id", a yellow rounded rectangle labeled "=", and a yellow rectangular box labeled "EXPRESSION". The sequence is flanked by double arrowheads pointing outwards.

referenced by:

- 
- The diagram illustrates the components of a function declaration in a grammar. It consists of a sequence of elements connected by lines: a yellow box labeled 'TYPE', a yellow oval labeled 'id', a yellow oval labeled '(', a yellow box labeled 'PARAMS', a yellow oval labeled ')', a yellow oval labeled '{', a yellow box labeled 'BODY', and a yellow oval labeled '}'. The sequence is flanked by double arrows pointing outwards, indicating it is a template for a grammar rule.

referenced by:

- 

referenced by:

- 
- A diagram showing the components of a function call expression. It consists of a yellow rounded rectangle labeled 'id', followed by a yellow rounded rectangle containing an opening parenthesis '(', then a yellow rectangle labeled 'PARAM\_VALUES', followed by a yellow rounded rectangle containing a closing parenthesis ')'. Arrows indicate the flow from left to right between these components.

referenced by:

- 

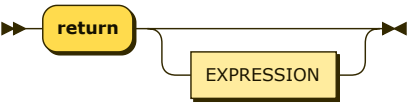
1/7

`::= ( EXPRESSION ( ',' EXPRESSION ) * ) ?`

referenced by:

- CALL\_METHOD

**RETURN:**

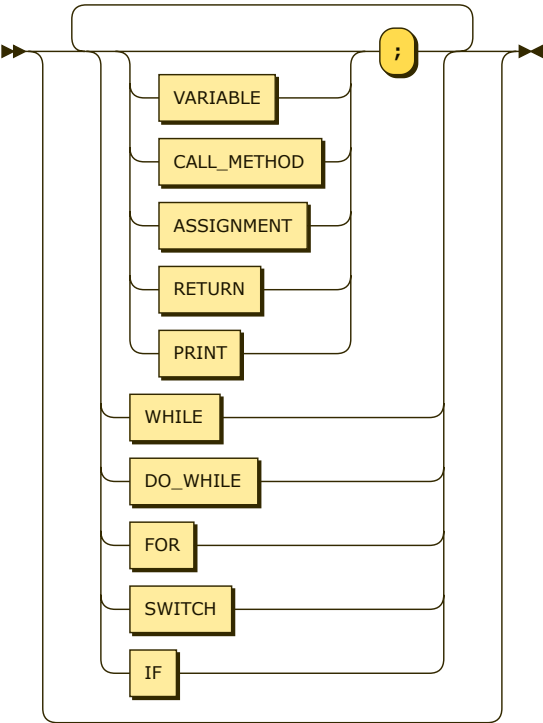


`RETURN ::= 'return' EXPRESSION?`

referenced by:

- BODY

**BODY:**



`BODY ::= ( ( VARIABLE | CALL_METHOD | ASSIGNMENT | RETURN | PRINT ) ? ';' | WHILE | DO_WHILE | FOR | SWITCH | IF ) *`

referenced by:

- DO\_WHILE
- FOR
- IF
- METHODS
- SWITCH
- WHILE

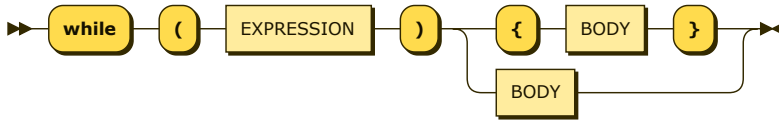
**ASSIGNMENT:**



`ASSIGNMENT ::= 'id' '=' EXPRESSION`

referenced by:

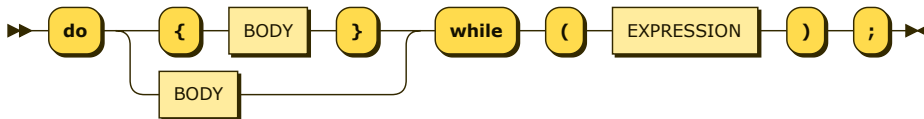
- BODY
- FOR

**WHILE:**

WHILE ::= 'while' '(' EXPRESSION ')' ( '{' BODY '}' | BODY )

referenced by:

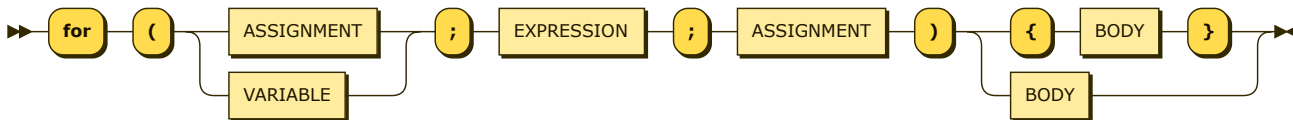
- BODY

**DO\_WHILE:**

DO\_WHILE ::= 'do' ( '{' BODY '}' | BODY ) 'while' '(' EXPRESSION ')' ';' ;

referenced by:

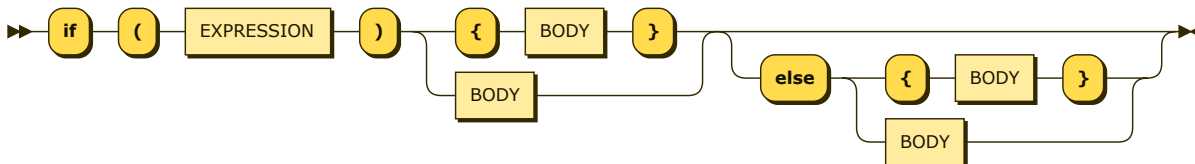
- BODY

**FOR:**

FOR ::= 'for' '(' ( ASSIGNMENT | VARIABLE ) ';' EXPRESSION ';' ASSIGNMENT ')' ( '{' BODY '}' | BODY )

referenced by:

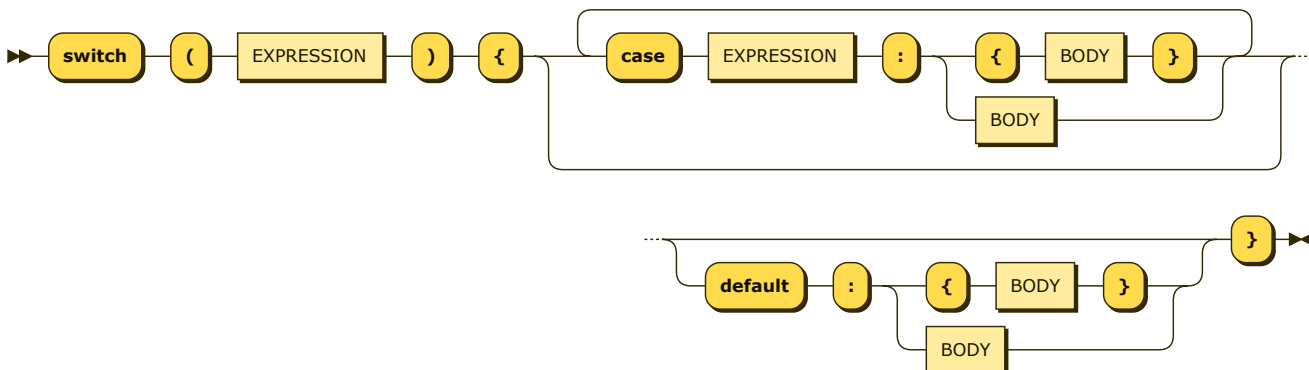
- BODY

**IF:**

IF ::= 'if' '(' EXPRESSION ')' ( '{' BODY '}' | BODY ) ( 'else' ( '{' BODY '}' | BODY ) ) ?

referenced by:

- BODY

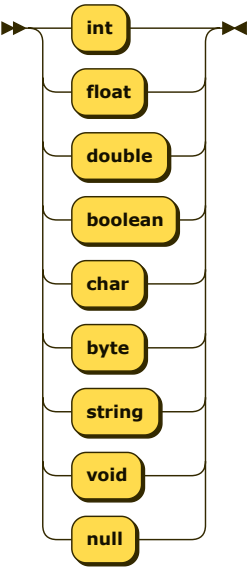
**SWITCH:**

SWITCH ::= 'switch' '(' ( 'EXPRESSION ' ) ' {' ( 'case' EXPRESSION ':' ( ' {' BODY ' }' | BODY ) ) \* ( 'default' ':' ( ' {' BODY ' }' | BODY ) ) ? '}'

referenced by:

- [BODY](#)

TYPE:



TYPE ::= 'int'  
          | 'float'  
          | 'double'  
          | 'boolean'  
          | 'char'  
          | 'byte'  
          | 'string'  
          | 'void'  
          | 'null'

referenced by:

- [METHODS](#)
- [PARAMS](#)
- [VARIABLE](#)

PRINT:

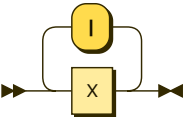


PRINT ::= 'print' '(' 'EXPRESSION ' )

referenced by:

- [BODY](#)

EXPRESSION:



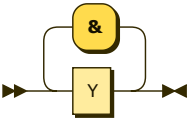
EXPRESSION ::= X ( '|' X ) \*

referenced by:

- [ASSIGNMENT](#)
- [C](#)
- [DO WHILE](#)
- [FOR](#)
- [IF](#)

- PARAM\_VALUES
- PRINT
- RETURN
- SWITCH
- VARIABLE
- WHILE

**X:**

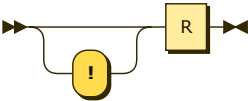


X ::= Y ( '&' Y )\*

referenced by:

- EXPRESSION

**Y:**

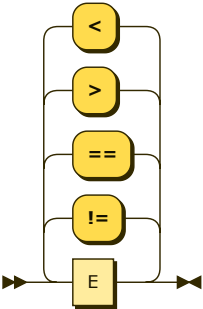


Y ::= '!' ? R

referenced by:

- X

**R:**

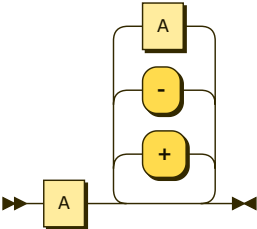


R ::= E ( ( '!= ' | '== ' | '>' | '<' ) E )\*

referenced by:

- Y

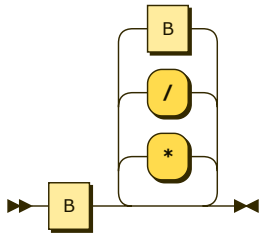
**E:**



E ::= A ( '+' | '-' | A )\*

referenced by:

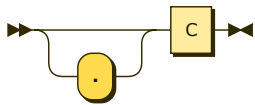
- R

**A:**

$$A ::= B ( '*' \mid '/' \mid B )^*$$

referenced by:

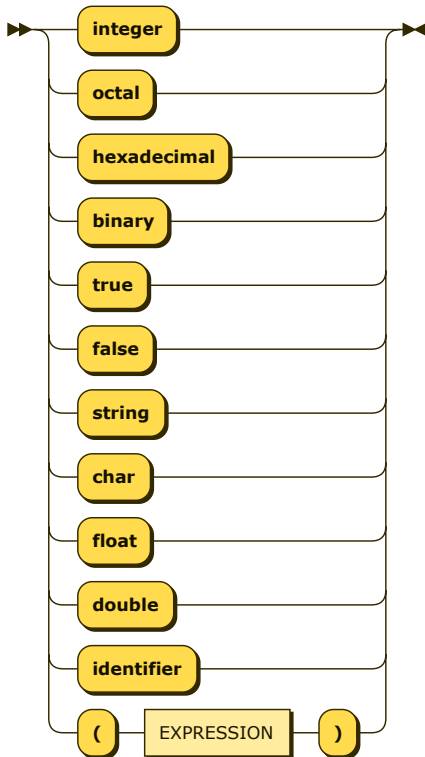
- E

**B:**

$$B ::= '.'? C$$

referenced by:

- A

**C:**

$$C ::= 'integer' \mid 'octal' \mid 'hexadecimal' \mid 'binary' \mid 'true' \mid 'false' \mid 'string' \mid 'char' \mid 'float' \mid 'double' \mid 'identifier' \mid '(' \text{EXPRESSION} ')'$$

referenced by:

- [B](#)

---

... generated by [RR - Railroad Diagram Generator](#) 