

FM is a small language designed for expressing the content of flip movies. It uses objects of externally defined types, allows calls to methods on those objects, and understands simple expressions and the “if” control statement.

1.  $program \rightarrow \mathbf{movie\ name\ \{ movieBody \}\ EOF}$
2.  $movieBody \rightarrow prologBlock\ pageBlocks\ | pageBlocks$
3.  $prologBlock \rightarrow \mathbf{prolog\ \{ prologStatements \}}$
4.  $prologStatements \rightarrow prologStatement\ | prologStatements\ prologStatement$
5.  $prologStatement \rightarrow variableDeclaration$
11.  $variableDeclaration \rightarrow id : type();\ | id : type(exprList);$
12.  $pageBlocks \rightarrow pageBlock\ | pageBlocks\ pageBlock$
13.  $pageBlock \rightarrow \mathbf{show\ ( integer )\ \{ pageStatements \}}$
14.  $pageStatements \rightarrow pageStatement\ | pageStatements\ pageStatement$
15.  $pageStatement \rightarrow$   
 $\quad \{ pageStatements \}$   
 $\quad | methodCall;$   
 $\quad | id = expr;$   
 $\quad | \mathbf{if\ (boolExpr)\ pageStatement}$   
 $\quad | \mathbf{if\ (boolExpr)\ pageStatement\ else\ pageStatement}$
16.  $expr \rightarrow term\ | expr + term\ | expr - term$
17.  $term \rightarrow factor\ | term * factor\ | term / factor$
18.  $factor \rightarrow integer\ | real\ | ( expr )\ | id\ | methodCall$
19.  $methodCall \rightarrow id()\ | id(exprList)\ | id.id()\ | id.id(exprList)$
20.  $exprList \rightarrow expr\ | exprList , expr$
21.  $boolExpr \rightarrow relExpr\ | !( relExpr )$
22.  $relExpr \rightarrow expr == expr\ | expr > expr\ | expr < expr$
23.  $type \rightarrow id$

## Language Notes

Comments, blanks, and other whitespace are ignored except as needed to separate adjacent syntactic tokens. A comment begins with the token `//` and continues to the end of the line.

There are three undefined nonterminals in the grammar: *id*, *integer*, and *real*. An *integer* consists of 1 or more digits (0-9) and denotes a decimal integer. A *real* consists of one or more digits (0-9) followed by a decimal point “.”, followed by one or more digits (0-9). An identifier *id* must begin with a letter, and consists of 1 or more letters, digits, and underscores. Upper- and lower-case letters are distinct, thus `aa`, `AA`, `Aa`, and `aA` are four different identifiers.

The keywords in the grammar (**movie**, **if**, etc.) are reserved and may not be used as identifiers.

All integer values are 32-bit, two's complement numbers.

The fm language includes binary arithmetic operators `+`, `-`, `*` and `/`. There are no unary `+` or `-` operators. The value `-n` can be computed by evaluating `0-n`.

A *bool-exp* is a logical expression, which may only be used as a condition in an if statement. Logical expressions do not have integer values and cannot be stored in variables.

In conditional statements, each `else` is paired with the nearest previous unpaired `if`.

- 4.  $prologStatements \rightarrow prologStatement \mid prologStatements prologStatement$
- 4.1  $prologStatements \rightarrow prologStatement prologTail$
- 4.2  $prologTail \rightarrow prologStatement prologTail \mid \epsilon$
  
- 12.  $pageBlocks \rightarrow pageBlock \mid pageBlocks pageBlock$
- 12.1  $pageBlocks \rightarrow pageBlock pageBlocksTail$
- 12.2  $pageBlocksTail \rightarrow pageBlock pageBlocksTail \mid \epsilon$
  
- 14.  $pageStatements \rightarrow pageStatement \mid pageStatements pageStatement$
- 14.1  $pageStatements \rightarrow pageStatement pageTail$
- 14.2  $pageTail \rightarrow pageStatement pageTail \mid \epsilon$
  
- 16.  $expr \rightarrow term \mid expr + term \mid expr - term$
- 16.1  $expr \rightarrow term exprTail$
- 16.2  $exprTail \rightarrow + term exprTail \mid - term exprTail \mid \epsilon$
  
- 17.  $term \rightarrow factor \mid term * factor \mid term / factor$
- 17.1  $term \rightarrow factor termTail$
- 17.2  $termTail \rightarrow * factor termTail \mid / factor termTail \mid \epsilon$
  
- 19.  $methodCall \rightarrow id() \mid id(exprList) \mid id.id () \mid id.id(exprList)$
- 19.1  $methodCall \rightarrow id callEnd$
- 19.2  $callEnd \rightarrow () \mid (exprList) \mid .id() \mid .id(exprList)$
  
- 20.  $exprList \rightarrow expr \mid exprList , expr$
- 20.1  $exprList \rightarrow expr exprListTail$
- 20.2  $exprListTail \rightarrow , expr exprListTail \mid \epsilon$