

# **F r a m e S c r i p t**

## **Q U I C K R E F E R E N C E**

**ELMSOFT INC.**  
7954 Helmart Drive  
Laurel Maryland 20723  
USA

FrameScript Quick Reference version 2.0  
October 2000

Copyright © 1997-2000 ElmSoft, Inc. All rights reserved.

ElmSoft, Inc. ("ElmSoft") and its licensors retain all ownership rights to the FrameScript computer program and other computer programs offered by ElmSoft (hereinafter collectively called "ElmSoft Software") and their documentation. Use of ElmSoft Software is governed by the license agreement accompanying your original media. The ElmSoft Software source code is a confidential trade secret of ElmSoft. You may not attempt to decipher, decompile, develop, or otherwise reverse engineer ElmSoft Software, or knowingly allow others to do so. Information necessary to achieve the interoperability of the ElmSoft Software with other programs may be available from ElmSoft upon request. You may not develop passwords or codes or otherwise bypass the security features of ElmSoft Software.

This manual, as well as the software described in it, is furnished under license and may only be used or copied in accordance with the terms of such license. The information in this manual is furnished for informational use only, is subject to change without notice, and should not be construed as a commitment by ElmSoft. ElmSoft assumes no responsibility or liability for any errors or inaccuracies that may appear in this book.

Except as permitted by such license, no part of this publication may be reproduced, stored in a retrieval system, or transmitted, in any form or by any means, electronic, mechanical, recording, or otherwise, without the prior written permission of ElmSoft.

Please remember that existing artwork or images that you may desire to scan as a template for your new image may be protected under copyright law. The unauthorized incorporation of such artwork or images into your new work could be a violation of the rights of the author. Please be sure to obtain any permission required from such authors.

FrameScript and ElmSoft are trademarks of ElmSoft.

Adobe, the Adobe logo, Acrobat, Acrobat Exchange, Adobe Type Manager, ATM, Display PostScript, Distiller, Exchange, Frame, FrameMaker, FrameMaker+SGML, FrameMath, FrameReader, FrameViewer, FrameViewer Retrieval Tools, Guided Editing, InstantView, PostScript, and SuperATM are trademarks of Adobe.

IN NO EVENT WILL APPLE, ITS DIRECTORS, OFFICERS, EMPLOYEES, OR AGENTS BE LIABLE TO YOU FOR ANY CONSEQUENTIAL, INCIDENTAL, OR INDIRECT DAMAGES (INCLUDING DAMAGES FOR LOSS OF BUSINESS PROFITS, BUSINESS INTERRUPTION, LOSS OF BUSINESS INFORMATION, AND THE LIKE) ARISING OUT OF THE USE OR INABILITY TO USE THE APPLE SOFTWARE EVEN IF APPLE HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES. BECAUSE SOME STATES DO NOT ALLOW THE EXCLUSION OR LIMITATION OF LIABILITY FOR CONSEQUENTIAL OR INCIDENTAL DAMAGES, THE ABOVE LIMITATIONS MAY NOT APPLY TO YOU.

The following are copyrights of their respective companies or organizations:

PANTONE® Computer Video simulations displayed may not match PANTONE-identified solid color standards. Use current PANTONE Color Reference Manuals for accurate color.

Six-color Process System Patent Pending - Pantone, Inc.

PANTONE Open Color Environment™ (POCE™) © Pantone, Inc., 1994, 1996.

The following are trademarks or registered trademarks of their respective companies or organizations:

Apple, AppleLink, AppleScript, AppleTalk, Balloon Help, Finder, ImageWriter, LaserWriter, PowerBook, QuickDraw, QuickTime, TrueType, XTND System and Filters, Macintosh, and Power Macintosh are used under license / Apple Computer, Inc.

Microsoft, MS-DOS, Windows / Microsoft Corporation

Sun Microsystems, Sun Workstation, TOPS, NeWS, NeWSprint, OpenWindows, SunView, SunOS, NFS, Sun-3, Sun-4, Sun386i, SPARC, SPARCstation / Sun Microsystems, Inc.

All other brand or product names are trademarks or registered trademarks of their respective companies or organizations.

Written and designed at Elmsoft, Inc., 7954 Helmart Drive, Laurel, MD 20723, USA

For civilian agencies: Restricted Rights Legend. Use, reproduction, or disclosure is subject to restrictions set forth in subparagraphs (a) through (d) of the commercial Computer Software Restricted Rights clause at 52.227-19 and the limitations set forth in ElmSoft's standard commercial agreements for this software. Unpublished rights reserved under the copyright laws of the United States. The contractor/manufacturer is Elmsoft, Inc., 7954 Helmart Drive, Laurel, MD 20723, USA.

# FrameScript Quick Command Summary

## Add Command

Add a command object to a menu.

```
Add CmdObject(cmdvar) To(Menuobjectvar);
```

Add a member to a list

```
Add Member(membervalue) To(listvar)
  [After(memberNumber)] [Before(memberNumber)];
```

Add a menu object to another menu or menu bar.

```
Add MenuObject(menuvar) To(Menuobjectvar);
```

Add a menu separator to an existing object.

```
Add MenuSepObject(menusepvar) To(Menuobjectvar);
```

Add a Property to a property list.

```
Add Property To(PropertyListvar) PropertyName(PropertyValue);
```

## Apply Commands

Apply a page layout properties from one page (master page) to body page

```
Apply PageLayout DestPage(BodyPageObject)
  SourcePage(MasterPageObject);
```

Sets the text properties of a range of text.

```
Apply TextProperties Properties(propvar)
  DocObject(docobject) or TextRange(rangevar);
  and/or
  CondFmt(conditionformatname) [CondFmt('condname') ... ]
```

## CallClient

The CallClient command allows you to send a message to another Frame Api Client or another script.

```
CallClient FrameClient (clientname) Message(msg) ReturnVal(retVar);
```

or

```
CallClient ScriptName (scriptname) Message(msg) ReturnVal(retVar);
```

## Clear Command

Clear all the Changebars from the specified document.

```
Clear ChangeBars [DocObject(docvar)];
```

Clear the text

```
Clear Text [DocObject(docvar)]  
  [Interactive CutTblCells VisibleOnly DontDeleteHiddenText];
```

## Close Command

Close an open book.

```
Close Book [BookObject(bookvar)] [IgnoreMods];
```

Close an open document

```
Close Document [DocObject(docvar)] [IgnoreMods];
```

Close an open text file.

```
Close Textfile Object(filevar);
```

## Copy Command

Copy the current selection to the clipboard.

```
Copy Text [DocObject(docvar)]  
  [Interactive VisibleOnly];
```

## Cut Command

Clears the current selection from the specified document and copy it to the clipboard.

```
Cut Text [DocObject(docvar)]  
  [Interactive CutTblCells VisibleOnly DontDeleteHiddenText];
```

## Delete Command

Delete a disk file.

```
Delete File(filename);
```

Delete a FrameMaker object.

```
Delete Object(objectvar);
```

Delete columns from a FrameMaker table.

```
Delete TableColumns TableObject(tablevar)  
  StartCol(startcolumnnumber) NumCols(numberofcolumns);
```

Delete rows from a FrameMaker table.

```
Delete TableRows TableObject(tablevar)  
  RowObject(rowvar) NumRows(numberofrows);
```

Delete Text from a document.

```
Delete Text Textrange(rangevar);
```

Deletes the contents of a text inset.

```
Delete TiContents(textinsetvar);
```

Delete a FrameScript variable definition

```
Delete Var(varname);
```

## Demote Command

Demotes the specified structural element or elements. (FM+SGML only)

```
Demote [DocObject (docvar)] [BookObject(bookvar)]  
[Element(eltvar)] [ElementRange(eltrangeVar)];
```

## DialogBox Commands

Display a dialog box allowing the user to select a file from the file system.

```
DialogBox Type(ChooseFile) [Directory(dirname)] [Title(dialogTitle)]  
[Init(InitialString)]  
[Mode(SelectFile OpenFile SaveFile OpenDirectory)]  
NewVar(varname) [Button(varname)];
```

Display a dialog box allowing the user to type in data.

```
DialogBox Type({Int Metric String}) [Title(dialogTitle)]  
[Init(initial value)] [Units(Inch MM CM PICA DIDOT)]  
NewVar(varname) [Button(varname)];
```

Display a dialog box allowing the user to select from a list of string items.

```
DialogBox Type(ScrollBox) List(stringlist) [Title(dialogTitle)]  
[Caption(TitlebarText)]  
[Init(numberOfInitialSelection)]  
[ReturnIndex(indexvarname)]  
NewVar(varname) [Button(varname)];
```

display a dialog box allowing the user to enter data in up to 3 edit boxes and 4 checkboxes.

```
DialogBox Type(MEdit) [Title(DialogTitleString)] [Title2(TitleString)]  
[Label1(Label1String)] [Label2(Label2String)] [Label3(Label3String)]  
[Edit1(Edit1Var)] [Edit2(Edit2Var)] [Edit3(Edit3Var)]  
[CheckBox1Label(Cbx1LabelString) CheckBox1(Cbx1Var)]  
[CheckBox2Label(Cbx2LabelString) CheckBox2(Cbx2Var)]  
[CheckBox3Label(Cbx3LabelString) CheckBox3(Cbx3Var)]  
[CheckBox4Label(Cbx4LabelString) CheckBox4(Cbx4Var)]  
[Button3Label(string)] [Button4Label(string)]  
[Button(buttonvarname)];
```

## Display Command

Displays an expression in a dialog box.

```
Display expression;
```

## DocCompare

The DocCompare command lets you perform the document compare function similar to the FrameMaker UI's

File->Utilities->Compare Documents function.

```
DocCompare OldDocObject(oldDocVar) NewDocObject(newDocVar)
  [InsertCondName(insCondFmtName)] [DeleteCondName(delCondFmtName)]
  [DeleteText(delTextString)]
  [Threshold(percent)]
  [SummaryOnly] [ChangeBars] [Hyperlinks] [OpenSummary] [OpenComposite]
  [ReturnComposite(rcompDocVar) [ReturnSummary(rsumDocVar)];
```

## Event Command

Start an event subroutine.

```
Event EventName
  commandlist;
EndEvent
```

## Execute Command

Execute a FrameCommand (also known as Fcodes).

```
Execute FrameCommand commandname;
```

Execute a Hypertext command.

```
Execute HyperText [DocObject(docvar)] Command(hypertextcommand);
```

## Export Object

Exports a Frame Graphic Object file to a file on your hard disk.

```
Export Object (objVar) File(filename)
  [ExportOptions]
```

## Find Command

Find a string in a FrameMaker text object

```
Find String(expression)
  InObject(textobjectvar) or InRange(textrange)
  Start(textlocation)
  {Nocase Wholeword Prefix Suffix}
  [ScrollTo] [Center]
  ReturnStatus(TrueFalsevar)
  ReturnString(stringvar)
  ReturnRange(rangevar);
```

Find a string in another string.

```
Find String(expression) InString(expression)
  Start(startchar)
  {Nocase Back Wholeword Prefix Suffix}
  ReturnStatus(TrueFalsevar)
  ReturnString(stringvar)
  ReturnPos(charposition);
```

Find a position in a document and scroll to it.

```
Find TextRange(textrangevar) ScrollTo;
```

Find a member in a list and return the position in the list.

```
Find Member(searchvalue) InList(listvar)
  ReturnStatus(TrueFalsevar)
  ReturnPos(listpositionvar);
```

Frame UI Find command.

```
Find FromTextLoc[(TextLocVar)] searchobject
  [Case WholeWord Backward Wildcard Wrap]
  ReturnRange(textRangeVar)
  ReturnStatus(TrueFalsevarname)
  ReturnElementRange(eltRangeVar);
```

## Generate Command

Generate list files from a book

```
Generate BookFiles BookObject(bookvar)
  [Interactive Visible];
```

## Get Command

Get a member from a list dataname.

```
Get Member Number(membernumber)
  From(listvar) NewVar(varname);
```

Get a FrameMaker object by name.

```
Get Object DocObject (docvar) Type(objecttype)
  Name(objectname) NewVar(varname);
```

Get a substring from a string.

```
Get String FromString(expression)
  [StartPos(startchar)]
  [EndPos(endchar)]
  [RemoveLeading('LeadingCharactersToRemove')]
  [RemoveTrailing('TrailingCharactersToRemove')]
  [RemoveChars('CharactersToRemove')]
  [ReplaceFirst('xxx') With('yyy') ...]
  [ReplaceLast('xxx') With('yyy') ...]
  [ReplaceAll('xxx') With('yyy') ...]
  [Uppercase] [Lowercase]
  [Reverse]
NewVar(varname);
```

Gets a list of text items from a FrameMaker text object.

```
Get TextList InObject(objectvar) or InRange(rangevar)
  textListTypes
NewVar(varname);
```

Get the text properties from a text location in a document

```
Get TextProperties DocObject(docvar) or TextLoc(locvar)
  NewVar(varname);
```

## If Command

Control statement for choosing one execution path or another.

```
If expression
  commandlist;
[Else
  commandlist;]
Endif
```

## Import Command

Import a file into a FrameMaker document.

```
Import File File(filename) [DocObject(docvar)]
  [TextLoc(locvar)] [ByCopy] [ByRef]
  [FilterFormatId(formatid)] (Frame 5.5 and above)
  [NewVar(textinsetobjectvar)] (if ByRef is chosen);
```

Import element definitions from another open document or book (FM+SGML only).

```
Import ElementDefs [DocObject(docvar) or BookObject(bookvar)]
  FromDocObject(docvar)
  [RemoveOverrides RemoveBookInfo DoNotImportEDD NoNotify];
```

Import formats from another open document.

```
Import Formats DocObject(docvar) or BookObject(bookvar)
  FromDocObject(docvar)
  [Pgf Font Page Table Cond RefPage Var XRef
  Color Math RemovePageBreaks RemoveExceptions
  DocumentProps CombinedFonts (for Frame 5.5 and above)];
```

## Install Command

Installs a FrameScript into the current session.

```
Install Script File(scriptfilename) [Name(scriptname)]
  [Label(menulabel)] [Shortcut(string)] When(whenenabled);
```

## Leave Commands

Causes control to go to the next command after the current loop.

```
LeaveLoop;
```

Causes the current subroutine to stop. The script continues at that point.

```
LeaveSub;
```

## Local

Declares the named variable to be local to the current subroutine.

```
Local var1;
```

## Loop Commands

Control command that lets you execute a sequence of commands repeatedly through a defined set of FrameMaker objects (such as all the open documents or all the documents in a book).

```
Loop ForEach(objecttype) In(object) LoopVar(varname)
  commandlist;
EndLoop
```

Control command that lets you execute a sequence of commands repeatedly until the specified conditions are satisfied.

```
Loop While(expression) Until(expression)
  [LoopVar(varname) InitVal(initialValue) Incr(incr)]
  commandlist;
EndLoop
```

## Merge

Merges the selected structural element or elements from the specified document into the first or last element in the selection. FrameMaker+SGML only.

```
Merge [DocObject (docvar)] [BookObject(bookvar)] [IntoFirst] [IntoLast]
      [ElementRange(eltrangeVar)];
```

## MsgBox Command

Displays a message box and allows the user to press various buttons.

```
MsgBox stringexpression
      [Mode(OkCancel CancelOK YesNo NoYes Warn Note)]
      [Button(buttonpushedvar)];
```

## New Commands

Creates a new Anchored Frame object.

```
New AFrame [NewVar(varname)] [DocObject(docvar) or TextLoc(locvar)];
```

Creates a new Arc object.

```
New Arc [NewVar(varname)] ParentObject(objvar);
```

Create a new attribute:

```
New Attribute NewVar(varname) AttrName(name) [AllowSpecial(spFlag)]
      [Value(objectvar2)];
```

Create a new attribute definition:

```
New AttributeDef NewVar(varname)
      AttrDefName(name) [AttrType(type)]
      [Choice(choiceValue)] [Choice(choiceValue)] [...]
      [Default(defaultValue)] [Default(defaultValue)] [...]
      [Required(reqValue)] [Flag(flagValue)]
      [RangeMin(rangeMinValue)] [RangeMax(rangeMaxValue)]
```

Create a new attribute definition list:

```
New AttributeDefList NewVar(attrDefListVar)
      AttrDefName(name) [AttrType(type)]
      [Choice(choiceValue)] [Choice(choiceValue)] [...]
      [Default(defaultValue)] [Default(defaultValue)] [...]
      [Required(reqValue)] [Flag(flagValue)]
      [RangeMin(rangeMinValue)] [RangeMax(rangeMaxValue)]
      . . .
      . . .
      . . .
```

Create a new attribute List:

```
New AttributeList NewVar(varname)
      AttrName(name1) [AllowSpecial(spFlag)] [Value(objectvar2)]
      AttrName(name2) [AllowSpecial(spFlag)] [Value(objectvar2)]
      . . . ;
```

Creates a new `BodyPage` object.

```
New BodyPage [NewVar(varname)] [DocObject(docvar)] or [PrevObject(objvar)];
```

Creates a new `FrameMaker Book`.

```
New Book [NewVar(varname)] [DocObject(docvar)] Name(objname);
```

Creates a new `FrameMaker Book Component`.

```
New BookComponent [NewVar(varname)]
  BookObject(bookvar) or [PrevObject(objvar)]
  [Name(componentName) ElementLoc(eltLoc)]; (FM+SGML only)
```

Creates a new `FrameMaker Character Format`.

```
New CharacterFormat [NewVar(varname)] [DocObject(docvar)] Name(CharFmtName);
```

Creates a new `FrameMaker Color`.

```
New Color [NewVar(varname)] [DocObject(docvar)] Name(ColorName);
```

Creates a new `FrameMaker Menu Command`.

```
New Command [Name(cmdname)] Label(cmdLabel) EventProc(eventname)
  [ShortCut(shortcutstring)] [AddTo(Menuvar)] [NewVar(varname)];
```

Creates a new `FrameMaker ConditionFormat`.

```
New ConditionFormat [NewVar(varname)] [DocObject(docvar)] Name(CondName);
```

Creates a new `FrameMaker Document`.

```
New Document
  Template(filename)
  or
  Portrait or Landscape
  or
  Width(metricvalue) Height(metricvalue) NumCols(intvalue)
  ColumnGap(metricvalue) TopMargin(metricvalue)
  Bottommargin(metricvalue)
  LeftInsideMargin(metricvalue) RightOutsideMargin(metricvalue)
  {SingleSided FirstPageRight FirstPageLeft} [Invisible]
  [NewVar(docvar)];
```

Create a new structural element (FM+SGML only)

```
New Element [DocObject(docvar)] ElementDef(eltDefObject)
  [ElementDefName(elementDefinitionName)]
  ElementLoc(eltLoc) or TextLoc(textLoc);
```

Create a new structural element Definition (FM+SGML only)

```
New ElementDef [NewVar(varname)] [DocObject(docvar)] Name(EltDefName);
```

Create an element location variable.

```
New ElementLoc NewVar(varname)
  Parent(elementvar) Child(elementvar) [Offset(integervalue)];
```

Create an element range variable.

```
New ElementRange NewVar(varname)
    Parent(eltvar) Child(eltvar2) [Offset(offset1)]
    [Parent(eltvar3)] [Child(eltvar4)] [Offset(offset2)];
```

Creates a new FrameMaker Ellipse.

```
New Ellipse [NewVar(varname)] ParentObject(objvar);
```

Creates a new FrameMaker Flow.

```
New Flow [NewVar(varname)] [DocObject(docvar)] or [ParentObject(objvar)];
```

Creates a format change list

```
New FormatChangeList Name(objectname) NewVar(varname) [DocObject(docvar)];
```

```
New FormatChangeList ParentObject(parentObj) NewVar(varname);
```

Create a new structural format rule(FM+SGML only)

```
New FormatRule [RuleType(TypeOfRule)] ParentObject(parentobj) NewVar(varname);
```

Create a new structural format rule clause(FM+SGML only)

```
New FormatRuleClause ParentObject(parentobj) NewVar(varname);
```

Creates a new FrameMaker FootNote.

```
New FootNote [NewVar(varname)] [DocObject(docvar)] or [TextLoc(locvar)];
```

Creates a new FrameMaker Group.

```
New Group [NewVar(varname)] ParentObject(objvar);
```

Creates a new FrameMaker Inset.

```
New Inset [NewVar(varname)] ParentObject(objvar);
```

Creates a new FrameScript Integer variable.

```
New Integer NewVar(varname) value(intval);
```

Creates a new FrameScript Integer list variable.

```
New IntList NewVar(varname) value(val1) value(val2) ...val(n);
```

Creates a new FrameMaker Line (Graphic).

```
New Line [NewVar(varname)] ParentObject(objvar);
```

Create a new Library variable

```
New LibVar Path(DirectoryName) NewVar(varname);
```

Creates a new FrameMaker Marker.

```
New Marker [NewVar(varname)] [DocObject(docvar)] or [TextLoc(locvar)]
    [MarkerTypeName(name)];
```

Creates a new FrameMaker MarkerType (Frame 5.5 and above).

```
New MarkerType [NewVar(varname)] [DocObject(docvar)] Name(objname);
```

Creates a new FrameMaker MasterPage.

```
New MasterPage [NewVar(varname)] [DocObject(docvar)] Name(objname);
```

Creates a new FrameMaker Math.

```
New Math [NewVar(varname)] ParentObject(objvar);
```

Creates a new FrameMaker Menu.

```
New Menu [Name(menuname)] Label(Label) [AddTo(Menuvar)] [NewVar(varname)];
```

Creates a new FrameMaker Menu Item Separator.

```
New MenuItemSeparator [Name(cmdname)] [AddTo(Menuvar)] [NewVar(varname)];
```

Creates a new FrameScript metric variable.  
`New Metric NewVar(varname) value(metval);`

Creates a new FrameScript Metric List.  
`New MetricList NewVar(varname) value(metval1) value(metval2)...value(metvaln);`

Create a new Object variable  
`New Object NewVar(varname) Value(expression)  
 IntValue(integer expression) DocObject(docvar);`

Creates a new FrameMaker Paragraph.  
`New Paragraph [NewVar(varname)] [DocObject(docvar)] or [PrevId(objvar)]  
 [PgFmtName(paragraphFormatName)] [Text(text expression)];`

Creates a new FrameMaker ParagraphFormat.  
`New ParagraphFormat [NewVar(varname)] [DocObject(docvar)] Name(objname);`

Create a new Point list variable.  
`New PointList NewVar(varname)  
 X(expression) Y(expression) [X(expr) Y(expr)] ...;`

Creates a new FrameMaker Polygon.  
`New Polygon [NewVar(varname)] ParentObject(objvar);`

Creates a new FrameMaker PolyLine.  
`New PolyLine [NewVar(varname)] ParentObject(objvar);`

Creates a new Property List variable.  
`New PropertyList NewVar(varname)  
 PropertyName(PropertyValue) PropertyName(PropertyValue) ... ;`

Creates a new FrameScript Real variable.  
`New Real NewVar(varname) value(realval);`

Creates a new FrameMaker Rectangle.  
`New Rectangle [NewVar(varname)] ParentObject(objvar);`

Creates a new FrameMaker ReferencePage.  
`New ReferencePage [NewVar(varname)] [DocObject(docvar)] Name(objname);`

Creates a new FrameMaker RoundRect.  
`New RoundRect [NewVar(varname)] ParentObject(objvar);`

Creates a new FrameMaker RulingFormat.  
`New RulingFormat [NewVar(varname)] [DocObject(docvar)] or Name(objname);`

Create a new script variable:  
`New ScriptVar File(FileName) NewVar(varname);`

Create a new subroutine variable:  
`New SubVar [File(FileName)] Subname(subname) NewVar(varname);`

Creates a new FrameScript String variable.  
`New String NewVar(varname) {value(stringval) or IntValue(AsciiCharValue)};`

Creates a new FrameScript StringList variable.  
`New StringList NewVar(varname) value(val1) value(val2) ...value(valn);`

Creates and inserts a table at the specified location.

```
New Table [NewVar(varname)] [DocObject(docvar)] or [TextLoc(locvar)];  
    Format(formatname)  
    [NumCols(numberofcolumns)]  
    [BodyRows(num)]  
    [HeaderRows(num)] [FooterRows(num)];
```

Add columns to an existing table.

```
New TableCols TableObject(tablevar)  
    [StartCol(startcolnumber)]  
    [NumCols(numberofcolumns)]  
    [Direction(Left or Right)];
```

Creates a new FrameMaker TableFormat.

```
New TableFormat [NewVar(varname)] [DocObject(docvar)] Name(objname);
```

Add rows to an existing table.

```
New TableRows  
    TableObject(tblvar)  
    [StartRow(rowobject)]  
    [BodyRows(num)] Direction(Above or Below)  
    [HeaderRows(num)] [FooterRows(num)];
```

Create a new tab list variable.

```
New TabList NewVar(varname)  
    Tab(expression) [TabType(tabtypeval)] [TabLeader(leader)] [TabDecimal(decval)]  
    [Tab(expression) ...];
```

Adds text to a location in a document.

```
New Text [DocObject(docvar)] or [TextLoc(locvar)] or [Object(pgfvar)] String  
    [NewVar(textLocVar)];
```

Creates a new Text File.

```
New TextFile File(filename) NewVar(filevar) [IOType(READONLY WRITEONLY APPEND)];
```

Creates a new FrameMaker TextFrame.

```
New TextFrame [NewVar(varname)] ParentObject(objvar);
```

Creates a new FrameMaker TextLine.

```
New TextLine [NewVar(varname)] ParentObject(objvar);
```

Creates a new FrameScript TextLoc variable.

```
New TextLoc Object(objectvar) Offset(integer) NewVar(varname);
```

Creates a new FrameScript TextRange variable.

```
New TextRange Object(objvar1) [Offset(offsetvar1)]  
    Object(objvar2) [Offset(offsetvar2)] NewVar(varname);
```

Creates a new FrameMaker TiApiClient.

```
New TiApiClient [NewVar(varname)] [DocObject(docvar)] or [TextLoc(locvar)];
```

Creates a new FrameScript UByteList variable.

```
New UByteList NewVar(varname) value(val1) value(val2) ...value(valn);
```

Creates a new FrameScript UIntList variable.

```
New UIntList NewVar(varname) value(val1) value(val2) ...val(n);
```

Creates a new FrameMaker UnanchoredFrame.

```
New UnanchoredFrame [NewVar(varname)] ParentObject(objvar);
```

Creates a new FrameMaker Variable.

```
New Variable [NewVar(varname)]  
  [DocObject(docvar)] or [TextLoc(locvar)] Format(formatname);
```

Creates a new FrameMaker VariableFormat.

```
New VariableFormat [NewVar(varname)] [DocObject(docvar)] Name(objname);
```

Creates a new FrameMaker Cross reference.

```
New XRef [NewVar(varname)]  
  [DocObject(docvar)] or [TextLoc(locvar)] Format(formatname);
```

Creates a new FrameMaker XRefFormat.

```
New XRefFormat [NewVar(varname)] [DocObject(docvar)] Name(objname);
```

## Open Commands

Opens a FrameMaker Book.

```
Open Book [File(filename)] [NewVar(bookvar)] [openoptions];
```

Opens a FrameMaker Document.

```
Open Document [File(filename)] [NewVar(bookvar)] [openoptions];
```

Opens a Textfile.

```
Open Textfile File(filename) NewVar(filevar) [IOType(ReadOnly Append WriteOnly)];
```

## Paste Command

Pastes from the clipboard to the current insertion point of the specified document.

```
Paste Text [DocObject(docvar)]  
  [{Interactive VisibleOnly DontDeleteHiddenText DontApplyAllRows  
  ReplaceCells InsertBelowRight}];
```

## PopClipboard

Move the entry on top of the clipboard stack to the clipboard itself.

```
PopClipboard;
```

## Print Command

Prints a FrameMaker book.

```
Print Book [BookObject(bookvar)];
```

Prints a FrameMaker Document.

```
Print Document [DocObject(docvar)];
```

## PushClipboard

Move the contents of the clipboard to the clipboard stack.

```
PushClipboard;
```

## Quit Command

Closes the FrameMaker application.

```
Quit Session;
```

## Read Command

Read a line in a text file.

```
Read File(filevar) [Rewind] NewVar(svar);
```

## Remove Commands

Remove an attribute from the specified structural elements.

```
Remove Attribute(AttrName) [DocObject(docvar)] [BookObject(bookvar)]  
From(elementvar);
```

Removes the command object from the specified menu object.

```
Remove CommandObject(commandvar) From(Menuobjectvar);
```

Removes a member from a list.

```
Remove Member[(membervalue)] [Number(membernumber)] From(listvar);
```

Removes the Menu object from the specified menu object or menu bar.

```
Remove MenuObject(menuvar) From(Menuobjectvar);
```

Removes the Menu Separator object from the specified menu object.

```
Remove MenuSepObject(menusepvar) From(Menuobjectvar);
```

Removes a property from a property list.

```
Remove Property(PropertyName) From(proplistvar);
```

Remove an attribute that has no value from the specified structural elements.

```
Remove.UndefAttr(AttrName) [DocObject(docvar)] [BookObject(bookvar)]  
From(objvar);
```

## Replace Command

Removes the command object from the specified menu object.

```
Replace Member[(membervalue)] [Number(membernumber)] In(listvar) With(newmembervalue);
```

## Return Command

Send a message back from a notification routine.

```
Return {Cancel Value SkipStep};  
(SkipStep is available under Frame 5.5 and above)
```

## Run Command

Runs a subroutine.

```
Run [scriptname.]SubroutineName {parm1(value1) ... parmN(valuen);  
Run [Object].method {parm1(value1) ... parmN(valuen);
```

## Save Command

Save a Book Document to disk.

```
Save Book [BookObject(bookvar)] [File(newfilename)] [FilterId(formatid)]  
saveoptions;
```

The FilterId option is available only on Frame 5.5 and above.

Save a Document to disk.

```
Save Document [DocObject(docvar)] [File(newfilename)] [FilterId(formatid)]  
saveoptions;
```

The FilterId option is available only on Frame 5.5 and above.

## Select Command

Display a quick keys box for user selection.

```
Select QuickKeys DocObject(docvar) List(stringlist)  
Prompt(promptstring) NewVar(varname);
```

Select cells in a table.

```
Select TableCells TableObject(tablevar)  
[TopRow(integer)] [BottomRow(integer)]  
[LeftCol(integer)] [RightCol(integer)]  
[SelectTable];
```

## Set Command

Assign a new value to a property.

```
Set [varname.]propertyname = expression;
```

Assign a new value to a FrameScript variable.

```
Set varname = expression;
```

## Sort command

Sort the items in a StringList or IntList variable.

```
Sort List(listVar) [NewVar(sortedList)]  
[Ascending | Descending] [Nocase] [Indirect]
```

## Split command

Split a structured element into two elements

```
Split Element [DocObject(docvar) or BookObject(bookvar)];
```

## Straddle Command

Straddle or Unstraddle cells in a table.

```
Straddle TableCells CellObject(cellvar)
  [NumRows(rowcount)] [NumCols(colcount)] [{On Off}];
```

## Sub Command

Start a new FrameScript subroutine.

```
Sub subroutineName
  commandlist;
EndSub
```

## Uninstall Command

Uninstall a FrameScript Script.

```
Uninstall Script Name(scriptname);
```

## Unwrap command

Unwraps (removes structures) from the specified text range.(FM+SGML only)

```
Unwrap [DocObject (docvar)] [BookObject(bookvar)]
  [Element(eltvar)] [ElementRange(eltrangeVar)];
```

## Update Command

Update the a Book.

```
Update Book BookObject(bookVAr) NewVar(docobj) [updateOptions];
```

Update the display of a document.

```
Update ReDisplay [DocObject(docvar)];
```

Update the formatting of a document.

```
Update Formatting [DocObject(docvar)];
```

Update the Hyphenation of a document.

```
Update Hyphenating [DocObject(docvar)];
```

Update the Equation settings of a document back to the default values.

```
Update ResetEquationSettings [DocObject(docvar)];
```

Update the Reference Frames of a document.

```
Update ResetRefFrames [DocObject(docvar)];
```

Update the Paragraph Numbering of a document by restarting the paragraph numbering.

```
Update RestartPgfNums [DocObject(docvar)];
```

Update a text Inset of a document.

```
Update TextInset(objectvar) [DocObject(docvar)];
```

Update the variables of a document.

```
Update Variables [DocObject(docvar)];
```

Update the cross references of a document.

```
Update XRefs [DocObject(docvar)]  
  [{ForceUpdate Internal OpenDocs ClosedDocs Everything}];
```

## Wrap command

Wraps (inserts a structured element) around the specified text range (FM+SGML only)

```
Wrap [DocObject (docvar)] [BookObject(bookvar)]  
  {ElementDef(eltdefvar) or ElementDefName(stringvar)}  
  [Element(eltvar)] [ElementRange(elrangeVar)];
```

## Write Command

Write a string to the selected output device.

```
Write {outputdest} string expression;  
  where outputdest is one of the following:  
  Console  
  Display  
  Object(filevariable)
```