
a filesafe screenplay formatter

gScript

version 2.9a

By Guy Gallo

This manual was produced using Doc-To-Help®, by WexTech Systems, Inc.

Contents

Introduction	1
What does gScript do?.....	1
Formatting Options.....	2
DRAFT.....	3
MASTER.....	3
SHOOTING.....	4
What gScript does that other formatters don't.....	4
Installation.....	5
Requirements.....	5
To Install gScript.....	5
Running the Demo.....	6
Getting Started.....	6
Creating a New Project.....	6
gScript Explained	11
gScript Styles and Layout.....	11
Definitions and Usage.....	11
Styles and "follow" specifications.....	13
Styles and Line Spacing.....	14
Applying Styles.....	14
Headers and Footers.....	15
The Title Page.....	17
Interface Tools.....	18
Keyboard Customizations:.....	18
Menu Item Additions.....	19
The Toolbar in gScript.....	20
gScript's Use of Outline Levels.....	21
Glossaries/Abbreviations.....	22
Miscellaneous Usage Notes.....	23
Printing Act Numbers.....	23
Colors.....	23
Revision Marks.....	23
Accessing Macros.....	24
A Note on Double Carriage Returns.....	24
A Note on Removing Extra Carriage Returns.....	24
Scene Numbering.....	25
FormatScript Examined	27

FormatScript - Introduction.....	27
Mode.....	28
Format.....	28
Options.....	30
Vertical Margins.....	31
PushButtons.....	31
Limitations.....	31
 gScript Macros Reference	 33
Overview.....	33
Alphabetical Listing of Macros.....	33
ApplyParen.....	33
AutoClose.....	34
AutoNew.....	35
AutoOpen.....	35
BackupThisDoc.....	35
ChangeLayout.....	36
CheckForStranded.....	37
ConvertStyles.....	37
ConvertThisPara.....	39
CreateTitlePage.....	39
EchoOnOff.....	41
FormatScript.....	41
FudgeAllParas.....	41
FudgeParaNarrower.....	41
FudgeParaWider.....	41
gsAdvancedOptions.....	41
gsAutoNew.....	44
gsControlCenter.....	44
gsLib.....	45
InsertABScene.....	45
InsertContinuing.....	46
InsertOmittedScene.....	46
ListGlossaries.....	47
NumberAllSlugs.....	48
NumberThisSlug.....	49
NumTopContinueds.....	49
ReactivateScreen.....	49
RemoveBlankParas.....	50
ResetAllDefaults.....	50
RevisionAccept.....	50
RevisionSearch.....	50
RevisionUndo.....	50
RunHelp.....	51
SceneReport.....	51
SetBackupOptions.....	51

SideBySideDialogue.....	52
SmartEnter.....	52
SmartTab.....	53
SpaceBeforeDecrease.....	54
SpaceBeforeIncrease.....	54
StripFormatting.....	54
StripNumbers.....	54
ToggleHidden.....	54
ToggleOutline.....	55
ToggleSlugIndent.....	55
Advanced Topics.....	57
Updating Older gScript Documents.....	57
What's different in 2.9?.....	58
Changing Default Information.....	58
Title Page.....	59
Margins and Paragraph Indents.....	59
Header.....	59
A Table of gScript's Macros.....	59
Advanced Margin Tweaking.....	64
Stranded Continueds.....	65
Finding stranded continueds.....	65
Making paragraphs wider.....	65
Decreasing Space Before a paragraph.....	65
GSCRIPT.INI Settings.....	66
Price, Support, and Updates.....	69
Price and Ordering.....	69
Support.....	69
History.....	69
Possible additions to future versions.....	70
Reaching the Author.....	70
Electronic Mail Addresses:.....	70
Mailing Address.....	70
Bugs and Suggestions.....	70
Notices.....	71
About the Author.....	72
Glossary of Terms.....	73
Index.....	75

Introduction

What does gScript do?

gScript (A. K. A The Gadfly Script Formatter) is a suite of styles and macros for Word for Windows to aid in the editing and formatting of a screenplay.

In the olden days, writers hired script services to prepare their manuscripts for submission. With the introduction of the Personal Computer, screenwriters either settled for the best their word processor could do, or purchased a second application to create a properly formatted print version of the script.

gScript greatly reduces the effort required to enter, edit, and format a screenplay.

gScript formats a screenplay with proper indents, font selection and size, and strings inserted at the top and bottom of the page (when the script is formatted for printing). It is used in two stages:

- During the creation of the screenplay, gScript simplifies editing with specially tailored styles, key assignments, toolbar assignments, and macros.
- After a draft of a screenplay is complete you can automatically format it with industry standard "tags" marking continued scenes and dialogue by running a single macro: FormatScript . The resulting layout conforms to the standard established ages ago (before the dawn of the Personal Computer), by the Writer's Guild of America and the Studio System. This format is based on the formula you've heard a zillion times:

One page equals one minute of screen time.

That may seem like an arbitrary, producer-ish concern. In fact it possesses a general truth that can be extremely helpful during the writing of the script.

The font used in this template is Courier 12 pt. If you are accustomed to measuring indentations in terms of characters (rather than in terms of inches), Courier 12 pt is a 10 "pitch" font: there are ten characters per inch.

Feature	Specification:
Font:	Courier 12 point, 10 pitch (that is, 12 points high, 10 characters per inch). This is the same font you might remember as standard on the antique IBM Selectric typewriters.
Margins:	Left - 1.5"; Right - 1.0"; Top - 1"; Bottom 1"
Character Name:	2.5" (25 Characters) in from left margin
Dialogue:	1.5" (15 Characters) in from left margin
Parenthetical	2.0" (20 Characters) in from left margin (the first line has a hanging indent of one character).
Transition	4.5" (45 Characters) in from left margin

The fact is, you can cheat and make your screenplays *appear* shorter by increasing the margins for action and dialogue; you can use proportionally spaced fonts (like the Times Roman 12pt used in this document). The reasons against doing so are two fold:

- No one is fooled. A script that comes in at 120 pages of 10pt Helvetica is, we all know, even producers know, actually 132 pages of script.
- It helps the actual composition of the screenplay if it is formatted according to these styles and indents. The flow of the script, the tempo and timing is simply easier to see, to feel, if the script is formatted in Courier 10 pitch with proper margins. You can sense how much description is on the page, how much dialogue.

In fact, I have come to the conclusion, after an odd experience with a major studio, that it is better to begin your script with more generous margins. The short form of

the story is as follows: I presented a script that had been fudged only very slightly: a point here and there, a fraction of an inch on the left and right margins. When printed the script looked quite normal. An executive decided that the script *read* long and sent it to the production department to be re-typed using the studio's standard script format. What had been delivered as a 121 page screenplay was returned to me as a 158 page behemoth. Not only were their margins larger, and the paragraph widths smaller, they had also inserted lots of (continuing) parentheticals. The studio used the extra forty-seven pages (by their count) to insist upon another, free, revision. Since I had fudged I had no moral ground to stand on.

As a result of the above I have increased the left margin in this template from 1.25 to 1.5 and shortened the width of the dialogue style.

Release Version 2.9 of gScript is a major upgrade. If you are wondering what happened to versions 2.1-2.8: there weren't any. The version released for Word 3.0 will be gScript 3.0.

Formatting Options

gScript provides three ways to format your screenplays: The first, Draft, is simply how gScript comes out of the box. The other two options, Master and Shooting, should be considered "post" options. That is, you might choose to refine your screenplay's layout and page breaks using either of these options before submitting the finished draft to a producer.

DRAFT

DRAFT: This mode is the default. This major difference between formatting paragraphs with this template and other Word templates you may be familiar with is that instead of using the Ribbon or Ctrl+Shift key combinations to apply styles to a paragraph (as you type) you can select styles from either dedicated toolbar icons (see "The Toolbar in gScript" on page 20) or with Alt+Key combinations. This ability to apply styles with Alt keys is simple (and more mnemonic). I suppose I got used to these particular combinations in Word for DOS.

Key combination	Paragraph format
Alt+S	slugline
Alt+A	action
Alt+C	character name
Alt+D	dialogue

Alt+P	parenthetical
Alt+B	transition break

As you enter these paragraph styles their definitions take care of indents, spacing, and keeping paragraphs together to avoid breaking dialogue between pages or separating a character name from its dialogue, or a slugline from its action.

For more information about the default styles, see "gScript Styles" on page 11.

MASTER

Master refers to what is known as a "master shot script" or "master scene script". The scenes in this type of screenplay are not, usually, broken down by precise camera setups or camera movement, or shot length. This is intended for first drafts of a screenplay. It is the most readable form, good for sending to people who are not familiar with the technical aspects of film-making.

This is the most economical way to format a screenplay. You do so by running FormatScript either from the gScript Control Center, or the Tools menu, and selecting "Master - dialogue only" in the Format option box.

Characteristics:

The effect of this is that when FormatScript encounters a page break that would fall in the middle of a paragraph of dialogue, it will insert a hard page break and place the "more" string ("-more-" by default) at the bottom of the page, and then begin the next page with the Character Name and the "dialogue continued" string ("cont'd" by default).

It does not insert any "continued" strings either at the top or bottom of a page.

For this reason it is the most economical way to print a screenplay. The approximate page savings is 15%. For instance, a script that paginates at 136 pages in Draft mode, will paginate at around 128 or 129 (depending, of course, on lots of small variables).

In this mode it is usually not necessary to have scene numbers for each slugline, but that's a matter of personal preference. I like to use scene numbers because it makes reference to a scene easier (even if your working copy has added pages between the time you submitted the screenplay and Sam calls you in for the meeting...)

See "Master - dialogue only" on page 28.

SHOOTING

This format is the most thorough and most accurately mimics the format of a screenplay as it would be produced by a script service (in the olden days).

Usually this format is used when submitting a second draft or a director's draft, prior to going into production.

To format a script in Shooting format you must run the main formatting macro FormatScript, accessed either from the Control Center, or the Tools menu, and

selecting "Shooting - dialogue and Action" in the Format option box.

Characteristics:

Scenes are numbered, significant camera angles and setups are included.

There are two additional options (presented as check boxes beneath "Break scenes and dialogue") which will affect how the screenplay is formatted.

See "Shooting - scene & dialogue" on page 29.

What gScript does that other formatters don't

Michael Gross, a Beta Tester, described gScript as "Filesafe". What he meant was: gScript makes no changes to your document that cannot be *easily* reversed. It does not tamper with the disk file. The formatted script can optionally be saved to another file name. The changes that are made to your draft screenplay can be removed by executing a single command: StripFormatting.

Many other script formatters require some kind of conversion from your word processor to a format they can read and understand. Even if they can convert in the other direction, back to your word processor, something is inevitably lost in the translation. Not so with gScript.

More specifically, since gScript is integrated with Word for Windows it possesses several advantages:

Note: even in "draft" mode the styles in gScript are such that you will never get "stranded" character names at the bottom of a page, and dialogue will never be broken.

- You don't have to create an ASCII file for formatting. gScript is a unified system capable of creating and printing both "draft" scripts, and fully formatted "presentation" scripts.
- Other formatters either ignore, strip, or choke on some of the most useful tools found in powerful word processors. With gScript, since you have complete control over what styles are hidden, you can fill your screenplay with notes, outline levels, footnotes, annotations and the like, and still create a formatted print file without changing a thing.
- There is no need for two copies of your screenplay (i.e. one draft, one presentation), since gScript is capable of stripping out all of the added strings and all of the altered styles.

Installation

Requirements

gScript requires:

- Microsoft Windows 3.0 or later (Version 3.1 is preferable)
- At least 2MB of memory (4MB or greater is preferable)

- Microsoft Word for Windows, version 2.0c (If you have version 2.0, 2.0a, or 2.0b of Word for Windows you can order a free upgrade to 2.0c by calling 1-800-426-9400)

To Install gScript

1. Place the distribution disk in drive A:.
2. From the Program Manager or File Manager select **Run** from the **File** menu.
3. Type **a:setup**, and select **OK**

The setup program automatically determines your template directory and your Windows directory. It will ask you to confirm the directory in which to place the demo and documentation files.

4. After entering a satisfactory **gScript** directory, press **OK**.

The setup program will copy the following files to the listed locations:

File	Location	Purpose
README.TXT	User specified directory.	Release notes and history.
GSFDEMO.DOC	User specified directory.	A Word for Windows document containing a self-running demo of gScript, frequently asked questions.
GSCRIPT.DOT	DOT-Path	The main template.
GSCRIPT.HLP	Windows directory	A Windows Help file.
GSCRIPT.INI	Windows directory	An ASCII text file that contains various important settings.

Note: If you are updating from a previous version of gScript the installation program will detect the presence of the older files. It will ask for confirmation before renaming an existing GSCRIPT.DOT to GSF_20.DOT. It will also ask for confirmation before deleting the version 2.0 help file (which was stored in the Word directory in the last release).

Running the Demo

To see **gScript** in action, simply load the file **GSDEMO.DOC** into Word for Windows and follow the instructions. The demo takes about five minutes to run and will give you a glimpse at the power of **gScript**.

Getting Started

If you are new to Word for Windows or are unfamiliar with the use of styles and templates, please review Chapter 8 "Formatting with Styles" (page 189) and Chapter 37 "Document Templates" (page 699) in the *Word for Windows User's Guide*.

Creating a New Project

To get up and running with gScript create a new document based on the GSCRIPT.DOT template:

- Select New from the File menu.
- Highlight GSCRIPT in the list of templates
- Press Enter

You will be presented with the following dialog box:

Title

The title entered in the first edit box will be automatically stored in the FileSummaryInfo title field for this document. That information will be used on the title page and in the header. You can change this title at any time by selecting Summary Info from the File menu.

File name

The file name entered in the second edit box will be used, once combined with the default Word extension of DOC, to save the new document to your hard disk. It's an unfortunate limitation of DOS (and therefore of Windows) that a file name cannot be more than characters. That doesn't give you much room.

It is important that you enter *only* the file name and not the extension. In this example I named my new file FATAL_2, and titled it Fatal Attraction II (how's that for wishful thinking?)

If you enter the name of a file that already exists in the chosen directory, gScript will warn you and return to the dialog box so you can enter a different file name.

Directory

By default, this edit box contains the path name of the currently "logged" directory. That is, this is the directory where, if you simply did a FileSave, Word would put your new document.

You can type in a different path. The only limitation is that the path must already exist.

I have often wanted to be able to create a directory at the same time that I create and save a file. That's why I put in the Pick button:

After you have chosen a title, filename, and the directory in which to store your new project you will be asked a final question: whether or not to create a new copy of GSCRIPT.DOT.

Pick

This button brings up the following dialog box:

Current Dir

The line at the top of the dialog box will always reflect the current directory.

List box

The list box on the left should look familiar. This is how directories and drives were displayed in older Windows applications.

Changing Directories

You can "navigate" your directory/drive structure by double clicking on the highlighted directory or drive in the list. Remember that [..] always means "the parent of the current directory." So in the above example clicking on [..] will move from E:\DATA\FILM (the current directory) up one level to E:\DATA (the parent).

You can also navigate by selecting the directory or drive with either the mouse or the Up/Down arrow keys and then pressing the "Change To Selected Directory" button with either a double click or the speedkey combination Alt+C.

Pick the Current Directory.

This button logs to the current directory, cancels the dialog box and returns to the New Project dialog box. The Directory edit box will then contain the chosen directory information.

Make New Dir Beneath Current

This is the real power of this little utility. You can create a new directory without having to leave Word.

If you have named your file before accessing the Pick Directory facility, the file name will be proposed as the name of the new directory as well.

After you have chosen a title, filename, and the directory in which to store your new project you will be asked a final question: whether or not to create a new copy of GSCRIPT.DOT.

Create New Template

Once all the information is entered into the New Project dialog box, press enter and you will then see the following dialog box:

This dialog box may seem a bit confusing to the completely new to Word. But it provides an important ability and so deserves some explanation here at the top of our discussion.

Word uses templates to store several types of information:

- Layout and formatting -- Things like margins, styles, page size, where to print the document. These attributes are copied into the new document at the time of its creation.
- Interface properties -- Macros, menu, key, toolbar shortcuts, and glossaries. These "utilities" are stored in the template, not in the document.

One of the features used by gScript that makes writing a screenplay much simpler is the glossary. You can store a character name under a one letter abbreviation (or glossary name). However, since glossaries are stored in the template and not in the document, if you have more than one screenplay based on GSCRIPT.DOT, your list of glossary/abbreviations may get unwieldy (you will have abbreviations from all your screenplays in the list). And worse, you might want to use the same abbreviation for different names in different projects.

Right at the outset, when creating a new project, you have the option to generate a *new* copy of GSCRIPT.DOT that will be specific to the project at hand.

In the example used above, if I were to select the second option button, a new template named FATAL_2.DOT would be created (and placed in the same directory as GSCRIPT.DOT and all the other templates).

It isn't necessary to create multiple copies of GSCRIPT.DOT (one for each screenplay). But as you become more familiar with the use of glossaries, it may be an option you will look for in the future.

After selecting OK in the above dialog box the title and author information will be updated on the title page (and in the headers) and the cursor will be positioned at the first slugline. A beep signals the beginning of the fun part.

gScript Explained

gScript Styles and Layout

Styles and their proper use give Word incredible formatting power and ease of use. Without styles this template and its utilities would not have been possible at all.

You should not delete or rename any of the custom styles defined in GSCRIPT.DOT. Doing so would disable the main formatting macro FormatScript.

You can redefine the custom styles -- changing details of their attributes, such as font name or pitch -- though I would caution against, as I said above, using fancy proportional spaced fonts.

It is possible that you will import a screenplay written in Word prior to getting gScript or a document written in another word processor. In either case it is likely that the styles in the imported document will not have the same names as those used in gScript (it is even possible that you won't have styles at all). Although it's a little work, and will likely require some manual clean-up, there are two macros to facilitate converting documents. See "ConvertStyles" on page 37 and "ConvertThisPara" on page 39.

If you are attaching this version of gScript to a document created and formatted with an earlier version of gScript (prior to 2.9) please see "Updating Older gScript Documents" on page 57.

The following [topics section](#) describes the pre-defined and built-in styles used by gScript.

Definitions and Usage

The most important styles in gScript (the ones used most frequently) are:

Style Name	Function	Key
Char	Character Name	Alt+C
Dialogue	Dialogue	Alt+D

Paren	Parenthetical: personal direction to the speaking character	Alt+P Ctrl+Shift+P
Transition	Transition Break; standard phrases that punctuate scene endings	Alt+B
heading 3	Slugline (Scene Heading)	Alt+S
Normal	Action: scene description and staging directions	Alt+A

Note that gScript has assigned these basic styles to Alt+Key combinations to ease formatting.

These key assignments, with the exception of Alt+A, do not conflict with any default key combination used by Word for Windows. It is also unlikely that you have personally customized these key combinations (since they are unavailable for customization in the ToolsOptionsKeyboard dialog box.) Alt+A, however, is used to access the Table menu. This was not an option in Word for Windows 1.x; and since I'd already determined to use this key combination for the Action style, and since I can't see how you will be inserting tables into a screenplay very often, I have decided, until convinced otherwise, to retain Alt+A for gScript's use. This will not, of course, effect Alt+A when you are in documents ruled by other templates.

Note: the Paren style has *two* key combinations. Ctrl+Shift+P applies Paren to the current paragraph. Alt+P is a smarter function: it displays a dialog box for the parenthetical and automatically inserts the parentheses. Every keystroke saved is a keystroke saved. See "ApplyParen" on page 33.

In addition gScript contains specifically defined styles that allow functional use of the outliner, footnotes, and annotations.

Style name	Function
Annotation text	Notes to yourself
Annotation reference	The Annotation note reference. It is taken from your initials as defined in ToolsOptionsUserInfo. Hint: you can change those initials to anything. My initials are "Note"

Footnote reference	Superscripted markers
Footnote text	Another way to keep notes
Heading 1	Act divisions
Heading 2	Sequence divisions
HiddenNote	Another way to annotate your script while writing.

These additional paragraph styles are all defined as hidden text so that they will not print unless you specifically tell Word to print them.

See ["gScript's Use of Outline Levels" on page 21](#) for more information about using the outliner when writing your screenplay.

gScript is designed to make great use of outline levels when composing and structuring you screenplay.

Note: If you wish to change the character attributes of the annotation or footnote styles you should do so at the outset. These styles possess a peculiarity: changes made to them to not apply retroactively. It would be best to save any changes to these styles to the template. See "Changing Default Information" on page 58.

In addition, there are styles used by gScript for various ancillary functions, and for marking the text inserted during the formatting:

Styles associated with less frequently used functions

Style	Function
ActDivision	Centered, underlined; used to mark the beginning of a new Act. The style is defined to include a page break.
ByLine	For the by line on the title page

DialogueItalic	Italic Dialogue (sometimes used for voice over dialogue)
Heading	Centered, bold, spacing expanded by 3pts. Used for the title of the screenplay on the first page
Headline	All capitals, flush left; used for emphasis and punctuation. For example: FADE IN:
SbSChar	Side by Side Character name
SbSDialogue	Side by Side Dialogue
Title	For the title on the title page
TitleAuthor	For the author on the title page

Styles used internally by FormatScript:

Style	Function
bContinued	For the bottom Continued placed between broken scenes
ByLine	For the by line on the title page
CharCont	For the character name on the page following broken dialogue
More	For the More string marking dialogue continued on the next page

tContinued

For the top Continued placed between broken scenes. Includes an automatic page break.

You should not delete any of these styles from gScript. They must exist for both FormatScript and StripFormatting to function properly. There should be no reason for you to ever apply these styles manually.

Styles and "follow" specifications

One of the neatest features of Word for Windows is the ability to define what the next paragraph style will be. As a result, for instance, if you press Alt+C for a Character Name, and then press enter, you don't have to press Alt+D to format the paragraph for Dialogue. It already is. Similarly, pressing enter in a line of dialogue automatically styles the next paragraph as Character Name. This makes for incredibly simple text entry.

The "follow" specifications for the major gScript styles are.

Current Style	Next Style
heading 1	heading 2
heading 2	HiddenNote
heading 3 (Slugline)	Normal (Action)
Normal (Action)	Normal (Action)
Char (Character Name)	Dialogue
Dialogue	Char
Paren	Dialogue

Transition	heading 3 (Slugline)

Styles and Line Spacing

If you are coming to Word for Windows from one of the less powerful word processors, then you are probably accustomed to entering double spacing by entering two carriage returns between those paragraphs that require extra spacing, as, for instance, between Action and Character Name, or between Dialogue and the next Character Name.

One of the most powerful features of Word for Windows is the fact that you can specify, when defining a paragraph format, how much space to skip before the paragraph, and how much to skip after the paragraph. As a result, there is no need for extra carriage returns.

Here are the Space Before specifications used in gScript:

Style	Space Before
heading 3 (Slugline)	2 lines
Char	1 line
Dialogue	(no space before)
Action	1 line
Transition	1 line

Not only is there no need for using extra carriage returns to space lines, they should be avoided at all costs. In order for the formatting macros to work properly your document should *not* have any extraneous blank lines or extra carriage returns. See "A Note on Double Carriage Returns" on page 24

Applying Styles

Normally you will apply styles as you type using the Alt+Key speedkeys. Some styles, such as Dialogue, will be entered automatically for you because of the

design of the "Next Style" attribute.

You can also choose styles from either the toolbar or the Format menu.

In addition there is an optional feature, SmartTab, which cycles through the major element styles using the Tab and Shift+Tab keys. [For more information see "SmartTab" on page 53.](#)

Headers and Footers

gScript's default header uses field codes to display various bits of document information. If you are completely unfamiliar with fields, please review the *Word for Windows User's Reference* Chapter 41. Or even better, Chapter 11 in my book *Take Word for Windows to the Edge* from Ziff-Davis Press..

In brief, a field is a special code (inserted either from the InsertField dialog box or manually using Ctrl+F9) that displays dynamic information. The most commonly used is {page}. The page field, when placed in a header or footer, displays the current page number.

gScript uses fields in the header, on the title page, and as a method for numbering scene headings.

Header Information

Other useful fields display information about the document as stored in the FileSummaryInfo dialog box.

The default header contains the following elements:

Element	Result
{Author} field	displays the author as contained in FileSummaryInfo
{Title} field	displays the title as contained in FileSummaryInfo
Page -	boilerplate text
{page}	the page number

Because of the definition of the header style, the above fields result in a header that looks something like:

Guy Gallo - Fatal Attraction II (Page - 2)

Your first question might be how did I manage to get the text flush left and flush right at the same time. Simple. The header style is defined with a single right aligned tab at 6.5". The information on the left is followed by a single tab.

If you edit the header footer (Header/Footer from the View menu) and then turn on ViewFieldCodes(Field Codes from the View menu), you will see the following:

{author} - {title} (Page - {page})

The fact that the header uses "fields" means that any time you change the information in FileSummaryInfo you will also be changing the header information.

Not everyone wants such an information packed header. Others want more. For instance, I have had requests to include the act information for movie of the week scripts.

It is a simple matter to accomplish either.

Deleting information is the easiest, of course. Simply edit the header and remove the fields you don't want. For instance, if all you want is the page number followed by a period, you would edit the header and delete everything but the {page} field. Follow it by a period. Precede it with a tab.

To add the act number is a little trickier. For one thing, it assumes that you have used the style ActDivision in the body of your screenplay to format the paragraphs containing the act information. If you have done that, then a very powerful field called STYLEREF can be used to display the information in the header. Styleref copies the text of the most recent instance of a style. So, if you have the text "Act I" formatted as ActDivision, the field {styleref "ActDivision"} placed in a header will display "Act I" at the top of every page.

To display act numbers in your header, edit the header, place the insertion point where you want the act information to display (in place of the author/title fields or left of the page number, for instance) and then:

1. press Ctrl+F9
2. type styleref "ActDivision"
3. press F9 to update the field

With ViewField codes on, you should see something like:

{author} - {title} {styleref "ActDivision"}(Page - {page})

If a styleref field refers to a non-existent style it will result in an error message: Error! No text of specified style in document.

For this reason, I have provided anonomous looking, but smarter field, stored in the glossary named zActNumberField:

```
{if{styleref "ActDivision"}<>"Error! No text of specified style in document."
{styleref "ActDivision"}}
```

This isn't nearly as complicated as it looks. It tests to see if the field {styleref "ActDivision"} would result in the error message and if not, display the result, if so, do nothing.

You could insert this smarter field by selecting the place in the header where you would like act numbers to print, type "zActNumberField", and press F3. (Note: I name this zActNumberField simply to keep it at the bottom of the list of glossaries.)

If you want to change the information contained in the header for all documents based on gScript and not just the current document, see "Changing Default Information" on page 58.

Header Format

The pre-defined font for Headers/Footers is Courier 12pt. This is a change from previous versions of gScript where the default header font was 10pt.

To change the font information for the header, select Styles from the format menu, find header in the drop down list, and select Define. Then select the element you wish to change.

If you want the changes you make to the active document to be saved into GSCRIPT.DOT, select Add to template. Changes saved to the template will affect all subsequent documents based on GSCRIPT.DOT.

No Footer

gScript makes no use of the footer. The reason is that when calculating page breaks in Master and Shooting modes, the FormatScript macro takes over the bottom margin setting and resets it to accommodate the new strings. It is possible that a footer would interfere with this process.

If you require a footer, be careful not to add a multi-line footer or to format the footer with a large font

I recommend not including footer information. If you have a footer in a screenplay formatted with a previous version of gScript, I advise removing it.

The Title Page

GSCRIPT.DOT comes with a pre-defined title page. When you create a new document the title page will automatically be inserted.

It has the following form:

- A paragraph formatted as "Title" containing the field {title}
- A paragraph formatted as "ByLine" containing the phrase "A screenplay by"
- A paragraph formatted as "TitleAuthor" containing the field {author}.
- And a two column table that is absolutely positioned at the bottom of the page. This table can be used to hold miscellaneous information such as return address, your agent's address, print date or version number. By default the right hand side of this table holds the field {useraddress}.

Since the title page uses fields to display the information realize that if you change the title and author information as it is contained in FileSummaryInfo you will also, automatically, be changing the title page (and the header).

The most important consequence of using fields is the effect of moving a document

from your personal machine to another machine. It is possible, if you run CreateTitlePage on someone else's machine that the address information proposed for the bottom right will be incorrect. The useraddress is specific to each machine. The fields Author and Title will travel with the document.

Note: You can replace the fields with their results:

Press F5 twice

Select TitlePage from the list of bookmarks

Press Ctrl+Shift+F9 to execute the UnLinkFields command

If you UnLinkFields (or directly type in title and author information) you will break the dynamic connection between FileSummaryInfo and the title page.

Notes:

The title page is a separate section from the rest of the document. It is marked by a bookmark named TitlePage. This bookmark is used to determine if a title page exists by several of the utility macros. **Do not delete this bookmark.**

Since the title page is section 1 and the remainder of your document is section 2 navigating using the EditGoto command requires some adjustment. EditGoto 3, for instance, moves to the third page -- which would be page two of the screenplay. The rule is simple: enter the page number you desire plus 1.

There are two other bookmarks on the title page: TitleLeft and TitleRight. These mark the left and right hand cells of the table. You can use EditGoto to move to these table cells for editing.

To change the default title page information you must edit GSCRIPT.DOT directly. See "Changing Default Information" on page 58.

You can remove the title page completely if you want. To insert a new title page execute the macro CreateTitlePage from either the Insert Menu or the Control Center.

Interface Tools

There are properties stored in a template that modify how the user interacts with Word. These changes to the menu geography, toolbar, keyboard, and glossary should be thought of as interface customizations.

gScript includes many customizations designed specifically to ease the writing and formatting of your screenplay.

Keyboard Customizations:

Here is a table of the default keyboard assignments in GSCRIPT.DOT:

Macro	Key Description
ApplyChar	Alt+C

ApplyDialogue	Alt+D
Applyheading3	Alt+S
ApplyNormal	Alt+A
ApplyParen	Alt+P
ApplyTransition	Alt+B
CheckForStranded	Alt+Ctrl+C
EditGlossary (built-in command)	Ctrl+G
FudgeParaNarrower	Alt+-
FudgeParaWider	Alt+=
heading 1	Ctrl+Shift+1
heading 2	Ctrl+Shift+2
gsControlCenter	Ctrl+Shift+C
ListGlossaries	Alt+G

NumberThisSlug	Alt+N
ReActivateScreen	Alt+Ctrl+F10
RevisionAccept	Alt+Ctrl+A
RevisionSearch	Alt+Ctrl+S
RevisionUndo	Alt+Ctrl+U
RunHelp	Alt+Ctrl+H
SmartEnter	{Enter}
SmartTab	{Tab}
SmartTabShift	Shift+Tab
SpaceBeforeDecrease	Alt+Shift+-
SpaceBeforeIncrease	Alt+Shif+t=
ToggleHidden	Ctrl+Shift+H

ToggleOutline	Ctrl+Shift+O
ToggleSlugIndent	Alt+Ctrl+I

Note on Modifying gScript's Key Assignments

Word's built-in key assignment utility is located on the Options dialog box, accessed through the Tools Menu. This is often referred to as ToolsOptionsKeyboard.

ToolsOptionsKeyboard only allows you to assign a limited number of key combinations: Ctrl+Shift+ a-z, 0-1, f1-f12.

The macro language, WordBasic, supports many more combinations. That's how I managed to assign functions to combinations such as Alt+Ctrl+I or Alt+G.

Because ToolsOptionsKeyboard doesn't support assigning such key combinations, it doesn't support deleting them either.

You can add additional combinations. But you cannot remove (from ToolsOptionsKeyboard) the more esoteric assignments.

You can do it with WordBasic. For those of you who are interested in getting this deep into the internals of key assignments I suggest two things:

- Get a copy of gToolBox (probably available wherever you got gScript). It contains a couple of macros to extend the available key combinations.
- Get a copy of my book on Word: *Take Word for Windows to the Edge*.

Menu Item Additions

Below is a list of the items added to the menus.

Added to Edit

Template Glossary - Runs ListGlossaries. See ["ListGlossaries" on page 47](#).

Added to Insert

Title Page - Runs CreateTitlePage, See ["CreateTitlePage" on page 39](#).

AB Scenes - Runs InsertABScene, See ["InsertABScene" on page 45](#).

Omitted Scenes - Runs InsertOmittedScene, [See "InsertOmittedScene" on page 46.](#)

Side By SideDialogue - Inserts a two column table and the necessary styles to begin entering side-by-side dialogue. [See "SideBySideDialogue" on page 52.](#)

Added to Format

The commands added to the Format menu are the most frequently used; they are the commands to format paragraphs for the various parts of a screenplay.

For a description of the style definitions of the various styles, please refer to "Definitions and Usage" on page 10.

1: Slugline applies the style "heading 3"

2: Action applies the style "Normal"

3: Character applies the style "Char"

4: Dialogue applies the style "Dialogue"

5: Parenthetical applies the style "Paren"

6: Break applies the style "Transition"

Note that the Alt speed key combinations are listed to the right of the command description on the pull down menu. Note as well that the number prefacing each menu item is underlined. This allows you to access these commands by pressing Alt+O (to pull down the Format menu), followed by 1-6.

Added to Tools

Control Center- Displays the gScript Control Center. [See "gsControlCenter" on page 44.](#)

gScript Options - Displays the Options menu. [See "gsAdvancedOptions" on page 42.](#)

Format Script... - Runs FormatScript on the finished draft. [See "FormatScript Examined" on page 27.](#)

Scene Report - Generates a scene report document containing a table of scenes, their page and their length. [See "SceneReport" on page 51.](#)

Remove Formatting - Runs StripFormatting to remove the styles and strings inserted by FormatScript. [See](#)

"StripFormatting" on page 54.

Added to Help

gScript Help - Loads the gScript Help file.

Note: I have not removed any of the default menu items. If, however, you find your menus too cluttered you *can* actually delete some of the built-in command menu items that you suspect you will never need when writing a screenplay, such as ToolsCalculate or ToolsCreateEnvelope. To remove an item simply go to the ToolsOptionsMenus dialog box, select the menu item to remove, making sure you are operating on GSCRIPT.DOT, and press the Delete button.

The Toolbar in gScript

Below is a legend for the toolbar visible in any document based on GSCRIPT.DOT.

This can be modified if you so desire.

gScript's Use of Outline Levels

The outline definitions in gScript are solely based on my own preferences. In writing screenplays I have found it useful to work from a very skeletal outline that consists of general act divisions, beneath which are sequence divisions, beneath which are the actual scene sluglines. For example:

Note: the act divisions entered as heading level 1 should be considered "working" divisions. They will not be printed in the formatted script. They should not be confused with visible act divisions formatted with the ActDivision style.

Note: even if you don't use heading levels 1 and 2, sluglines must be formatted as heading 3. Otherwise FormatScript will not function properly.

The above graphic shows a document with only outline levels showing: Heading 1 for Act Divisions; and Heading 2 for "Sequence" descriptions such as "Titles", "Girl meets Boy", and the like. Sluglines are always (and must be) formatted as Heading 3. This organization allows for simple navigation through a long script.

To make using the outline simpler, gScript includes a small macro named ToggleOutline which is present on the toolbar as . It can also be accessed with the speed key Ctrl+Shift+O.

Outline levels can be entered in one of two ways:

- Alt+Shift in combination with the left and right direction keys.
- The speedkeys defined as part of the style definition:

Ctrl+Shift in combination with 1, 2, or 3

When not in Outline View the first method formats a paragraph in relation to the most recent heading level. For instance, Alt+Shift+RightArrow will format a paragraph as heading 2 if the most recent prior heading is heading 1.

Alt+Shift+LeftArrow formats the current paragraph at the same level as the preceding level.

When in Outline View these keys promote/demote the current outline level.

You can also absolutely format a paragraph as a given heading level with the following speedkeys:

level	speedkey
heading 1	Ctrl+Shift+1
heading 2	Ctrl+Shift+2
heading 3	Ctrl+Shift+3 Alt+S

Note: Since heading 1 and heading 2 styles are defined as hidden, you must toggle the display of hidden text on in order to work with outline levels.

And a warning: I have seen Word do strange things when using Alt+Shift+LeftArrow when **not** in Outline View. The symptom is to format all subsequent body text as the previous heading level up to the next heading. If you press either of the Alt+Shift+Direction key combinations when not in Outline View and the following text looks weird **immediately press Alt+Backspace or Ctrl+Z to Undo**. For this reason (unexplained and unduplicated at Microsoft), I recommend using the Ctrl+Shift speedkey combinations when not in Outline View.

Glossaries/Abbreviations

If you are completely unfamiliar with Word's Glossary, see Chapter 13 of the *Word for Windows User's Guide*.

The glossary function allows you to store boilerplate text, such as character names, scraps of descriptions (to be moved or held in storage), with a short, mnemonic abbreviation or glossary name.

The standard EditGlossary command is accessed from the Edit menu, or with the key combination Alt+E+O. In addition gScript has a custom key combination for EditGlossary: Ctrl+G.

You define a glossary, using EditGlossary, by selecting the text to store, choosing Glossary from the Edit menu, and assigning the abbreviation.

You expand a glossary by typing the abbreviation and then typing F3.

However, there is a problem with the built-in glossary function when used to manage and display glossaries: it displays all the glossaries, both global and template.

As a result I have included a custom macro, ListGlossaries, accessed by pressing Alt+G or the G button on the toolbar. It displays a list of only those glossaries stored in the attached template. This makes it much easier to use the glossary to store multiple abbreviations and not have to wade through all the glossaries also contained in your NORMAL.DOT. See "[ListGlossaries](#)" on page 47 for more information.

In addition, SmartEnter provides a convenient alternative way to expand glossaries within paragraphs formatted as Char. See "[SmartEnter](#)" on page 52 for more information.

Note: When you use the built-in EditGlossary command, where a newly defined glossary is stored depends upon an option setting found on the FileTemplate dialog box. The three options are

Global (Available to All Documents)

With Document Template

Prompt for Each New

Be sure that you *do not* have this option set to Global. If you do, any glossary you create with EditGlossary will not show up in the ListGlossaries list box. ListGlossaries automatically saves new glossaries to the template (even if you have the above set to Prompt for Each New).

Miscellaneous Usage Notes

Printing Act Numbers

There may be times when you want to divide your screenplay into acts, not only using the heading 1 outline level (which is hidden and intended for organizational purposes only), but so that the act number displays on the first page of each act.

In order to insert an act break that will be recognized and handled appropriately by FormatScript, use the pre-defined style ActDivision.

*You should **not** insert a hard page break or section break between acts.*

The ActDivision style is defined with "Page break before" set to yes, so wherever you insert an ActDivision paragraph it will be the first paragraph of a new page.

ActDivision does not have a speedkey combination or any special macros associated with it (mainly because you should only have to apply it 3 or five times).

Simply insert a blank paragraph where you want the new act to begin and then press Ctrl+S followed by ActDivision and Enter (or choose ActDivision from the Ribbon if it is active).

To have the act number print at the top of each page you must alter the header. See "Header Information" on page 15.

Colors

gScript has pre-defined color attributes. For instance, the hidden heading levels 1 and 2 are red; heading 3 (Slugline) is blue; Annotations are red, footnotes are blue, and headers and footers are green.

You can change any of these particular paragraph styles to other colors if you wish (since they are hidden to FormatScript).

However, do not change any of the "visible" paragraph styles (Char, Dialogue, DialogueItalic, etc.) to red or cyan. FormatScript uses these colors to mark paragraphs of action and dialogue that have been broken by formatting and StripFormatting uses them as search criteria when restoring those paragraphs that were broken by FormatScript.

Revision Marks

Screenplay formatters sometimes allow you to mark "X-Changes", that is, lines that have changed from draft to draft.

Since gScript is integrated with Word for Windows you have only to turn on Revision Marks for all changes to be displayed between drafts.

There are three macros in gScript that will aid in using Revision Marks.

RevisionSearch -- looks for the next instance of revised text

RevisionAccept -- accepts the currently selected revised text

RevisionUndo -- undoes the currently selected revised text

These three macros have been installed on the toolbar (See "The toolbar" on page 31), along with a button to easily access the main ToolsRevisionMarks macro (that is, the macro that would be accessed if you selected RevisionMarks from the Tools menu).

ToolsRevision Marks can be accessed by clicking on the toolbar

Accessing Macros

Just a reminder to those of you who are new to Word's macro facility: You can access any macro in a template without using the various "access methods" such as speedkeys or menus or the toolbar. This means you can easily access macros that *don't have* access methods.

To execute a macro that does not have an access method, simply select Macros from the Tools Menu. You will be presented with a list of macros. Normally, when called from within a template based document (such as your screenplay) this list will automatically reflect the macros in the template. You can press Alt+T to insure that the correct list of macros is displayed.

From the list of macros, find the one you are interested in running. Then press Enter or Alt+R.

A macro selected in this way can also be deleted or renamed (don't do so to gScript's macros, please). If the Edit button is available (not grayed) then the macro can be edited as well. If the Edit button is grayed then the macro is encrypted.

A Note on Double Carriage Returns

One of the most common errors in entering a screenplay with gScript, and one that often results in blank pages and incorrect formatting, is the use of extra carriage returns to insert more blank space between scenes.

It is very important that you do not use extra carriage returns to create "white space" between scenes or paragraphs.

In order to change the line spacing between scenes or between sluglines you must change the definition of the style.

gScript includes a macro, ChangeLayout, to facilitate changing default line spacing. It can be accessed from the gScript Main Menu.

A Note on Removing Extra Carriage Returns

If you are importing a document from another word processor, or from a text file, it is possible that you will have double spacing doubled by the existence of multiple carriage returns. (The best indication that this is a problem is if your action paragraphs are separated by much more than one line).

If this is the case then you must strip out the extra carriage returns. You could do so manually (how tedious), or you can use Word for Windows to Search and Replace carriage return pairs with a single carriage return.

1. Select Replace from the Edit menu

2. In the Search Field enter

^p^p

3. In the Replace Field enter

^p

Whether you Replace one at a time or choose Replace All depends upon your patience and your trust. (This should work on *most* files, but cannot be guaranteed for all.)

Also, if you have a document that contains carriage return "triplets" or more, you may have to run the above EditReplace more than once.

You can also run RemoveBlankParas from the ToolsMacro dialog box.

However, if you fail to check for carriage return pairs the main formatting macro, FormatScript, will do it for you.

Scene Numbering

You can number sluglines (scene headings) in one of two ways: as you enter a new slugline pressing Alt+N inserts a field, {seq slug}, in the position specified by the Side parameter in GSCRIPT.INI.

Executing NumberAllSlugs (accessible from the Control Center) goes through the entire document inserting scene numbers either at the left, the right, or both left and right of the slugline.

You can also update all numbers or insert new numbers before running FormatScript.

Note: if you want a line that *looks* like a slugline (when printed) but isn't to be numbered, insert a paragraph of style `Headline`.

Why {Seq Slug} instead of {Autonom}?

The reason gScript uses "sequence" fields to number sluglines rather than the more automatic (and faster) "outline" numbering is simple: it is an easy matter to "refer" to the most recent {seq} field, while there is no way, that I have discovered, to refer to the most recent {autonom}. This is how gScript can duplicate the current scene number right of the slugline text, and how FormatScript can place the scene number left of the top continueds.

If you select the entire document and press Ctrl+Shift+F9, you will "unlink fields", replacing the field codes with their result.

You should only unlink {seq slug} fields when the screenplay is complete because once unlinked, the slugline numbers cannot be automatically changed (added to or subtracted from) nor can they be removed automatically.

An unintended, but useful consequence of using fields to number sluglines is that pressing F11 moves from the current point to the next field. So, if you have only Left numbers inserted (which is the default when you press Alt+N), you will move to the slugline for the next scene.

FormatScript Examined

FormatScript - Introduction

While the paragraph styles defined in gScript are useful in and of themselves, and would produce a printable copy of your screenplay, the meat of the matter lies in a macro called FormatScript.

FormatScript examines your screenplay and determines where each page break falls. It then performs one of several actions, dependent upon where the break occurs.

- If the page break falls in the middle of a scene, either between paragraphs of action or within a paragraph of action, then FormatScript inserts the bottom and top Continued Strings, if you are generating a Shooting format or adds a page break to the paragraph definition if you are generating a Master format.
- If the page break falls at a slugline FormatScript adds a page break to the paragraph definition (that is, toggles "Page break before" to on in the FormatParagraph command).
- If the page break falls within a paragraph of dialogue, FormatScript inserts the Dialogue More String, inserts the bottom and top Continued Strings (for Shooting format), and inserts the character's name at the top of the continued speech along with the Character More String.

When you run FormatScript you will see the following dialog box:

Mode

There are two Break options. They are grouped as Option Buttons. You will notice that you can only have one button active at a time.

Automatic - No Intervention

If you select this mode, gScript operates without user intervention, making intelligent decisions about page breaks when they fall within paragraphs of dialogue or action.

By default the break will be placed one line up, unless doing so would strand single lines at either the bottom of this or the top of the next page.

Interactive - Confirm Breaks

In this mode gScript will stop and prompt you for confirmation when it encounters a page break that falls within a paragraph of dialogue or action. It will present you with the following menu:

By default gScript recommends one line up from the current cursor position. However, you can break the paragraph at the insertion point, or two lines above the insertion point, or at the top of the current paragraph.

Note: selecting Para up when you are within a paragraph of dialogue actually places the break before the character name of the current paragraph of dialogue (so technically, in the case of dialogue, it is "two paragraphs up").

Note: New to version 2.9 of gScript is the ability to force a page break one line down or at the next paragraph. I put this facility in at the request of several users, but I must warn you that it is extremely precarious. It is most likely that if you force a page break at the next paragraph you will end up with "stranded continueds." If you force a break after the next line it is quite possible all will be well. Use with caution. And only if you are willing to do some manual clean-up.

Format

Master - dialogue only

This option is useful if you are printing a draft of a screenplay and do not want to break scenes.

In this mode FormatScript will check to see if a page break falls in the middle of a paragraph of dialogue, and if so, place the "more" string at the point where the dialogue is interrupted, and place the character name and the "dialogue continued" string before the continuation of the dialogue.

This mode does not insert top and bottom "continued" strings.

Shooting - scene & dialogue

If you select this option button, FormatScript will paginate your document and place breaks on each full page, determining if the page break falls in the middle of

a scene, and if so, will place a "continued" string at the bottom of the current page, and at the top of the next page.

It will also check to see if the page break falls in the middle of a paragraph of dialogue, and if so, place the "more" string at the point where the dialogue is interrupted, and place the character name and the "dialog continued" string before the continuation of the dialogue. (This is in addition to the "continued" strings.)

The following check boxes are only relevant if you are breaking both scene and dialogue paragraphs.

Number Top Continueds

Checking this option before beginning formatting will cause FormatScript to insert the current scene number before the Top Continued inserted on the continuation pages.

You should uncheck this option in either of two cases: 1) your scenes are not numbered at all, or 2) they are numbered, but you don't want them inserted on the continuation pages.

This option is only relevant when you select Shooting as the Format type. With Master it has no effect.

Count pages to each scene

If this option is checked, after FormatScript has gone through the document determining page breaks and inserting Continued and More strings, it will go back through the document and count the length of each scene and place the number to the right of the top continued string.

For example, if scene 111 is four pages long and begins on page 75, the following would be printed at the top of the following pages:

Page 76 (Nothing. This is the first "over-run" and is not noted):

111 CONTINUED:

Page 77 (This is the second full page of over-run):

111 CONTINUED: (2)

Page 78:

111 CONTINUED: (3)

Page 79:

111 CONTINUED: (4)

The process of determining each scene's length is rather complicated (and terribly clever), and so it adds approximately ten minutes to the time needed to format a full length script.

Options

The following check boxes tell FormatScript how to behave while formatting.

Renumber Scenes First

If this option is checked, gScript will renumber all sluglines before proceeding with the format. (Like StripFormatting, you can run this macro independently of FormatScript. It is called NumberAllSlugs). See "NumberAllSlugs" on page 23

Update Current Numbers

This option is useful if you have added scene numbers manually. It is unnecessary if you have selected Renumber Scenes First. Basically it insures that the sequence field codes have been numbered correctly by selecting the entire document and executing UpdateFields (an internal Word command, by the way...)

Realize that this option will also update any other fields that you may have placed in the body of your document. Unlikely, but possible.

Save To New File Name

If this option is checked then before anything else, gScript will prompt you for a new file name and make a copy of your script. This is useful if you don't trust gScript (which you shouldn't at first, I suppose). It isn't, as I've said before, necessary, since Remove All will strip all the formatting from the document.

Turn off screen display

This is a new option in version 2.9. It requires that you are running Windows 3.1. No damage will happen if you are still running Windows 3.0, but it will have no effect. (I can't imagine why anyone would still be running version 3.0. If you are, upgrade.)

What it does is turn off the screen display during the

formatting process. This speeds up formatting by about 20% (that's an estimate).

However, some people like to watch the script go flashing by page by page. So, to each his own taste.

Iconize when Automatic

If this option is checked, and the Mode selected is Automatic, then gScript reduces the Word application to an Icon. Progress of the formatting will be displayed on the Icon title.

However, due to limitations in Word and Windows, iconizing the formatting run will not allow you to work in other applications easily. That is to say, you *can* move to another application, but all processes will be slowed to a crawl. This is due to the fact that some actions in Word, such as repagination and many macro functions, do not release any time at all to the rest of Windows. This option will clear your screen, and perhaps marginally speed things up, but it will not, sadly, allow you to go on about other business at a normal rate.

Vertical Margins

There are two choices relating to top and bottom margins. Which you use will depend on the length of your script. There isn't a great deal of difference between the two (.5 inch), but if you are formatting a script with top and bottom continueds, one inch margins there seems to be a lot of "white space" on the page.

Normal (1 in)

This setting provides the most generous margins. When using this setting it is unlikely that you will create stranded continueds when you format the script in Shooting mode.

Tight (.75 in)

This setting allows one or two lines extra on every page. It should result in a script with precisely the same number of pages as Draft mode (despite the fact that it has inserted strings marking breaks in dialogue and scenes).

PushButtons

Here follows a list of the various buttons found on the FormatScript dialog box.

Begin

Begin formatting using the selected settings.

Cancel

Cancel formatting.

Help

Display the GSCRIPT.HLP file. This file should be located in your Windows directory.

Save Settings

If this button is selected gScript will write the current settings to GSCRIPT.INI. The next time you run gScript it will begin with the saved settings as the new defaults.

Layout

Runs the ChangeLayout macro. This allows you to fine tune the page layout before formatting. ChangeLayout should not be run after FormatScript.

Options

Runs gsAdvancedOptions. Allows you to change settings, such as strings to insert, before running FormatScript. See "[gsAdvancedOptions](#)" on page 42.

Limitations

At this time, FormatScript is an all or nothing proposition. That is, you cannot run it on a half formatted screenplay. If you have created a formatted screenplay, added some scenes, deleted others, and then want to format the script again, you must first StripFormatting and start over.

The next version of gScript will contain facilities for reformatting a formatted script (and automatically numbering new pages as A/B pages).

Whether or not you have to renumber all scene headings will depend on whether or not you numbered the new slugs as you entered them.

If you attempt to format a script that has already been formatted, you will be warned and asked to start over.

gScript Macros Reference

Overview

Most of the macros found in GSCRIPT.DOT are encrypted. This may sound, at first, like a limitation, but consider:

The main macro, FormatScript, is incredibly complex. Changing anything in it, deliberately or inadvertently, could result in its not functioning at all.

You can still add custom macros to the template, or modify the speed key and menu and toolbar assignments. You can, of course, add glossaries as well.

Alphabetical Listing of Macros

This section -- which describes each macro, its purpose and access methods -- is not available in the unlicensed version of gScript.

The information in this section *is* however, available in the on-line help file.

An abbreviated list of gScript's macros can be found on page 59.

To receive a printed version of the gScript manual, including the 24 pages of the macro reference, please contact Advanced Support Group at 1-800-788-0787.

Advanced Topics

Updating Older gScript Documents

If you have used versions of gScript prior to 2.9 there are a few things to consider. It might be wise, as stated above, to make a copy of your older GSCRIPT.DOT to another name.

Since this version of gScript provides a template of the same name (GSCRIPT.DOT) as the default template, it will automatically be attached to any documents created with older versions of gScript.

When the newer version of GSCRIPT.DOT is attached it provides the new and improved interface features and macros. However, attaching the newer GSCRIPT.DOT does NOT automatically update the styles contained in documents created with earlier versions of gScript.

If your older document is already formatted, and you have no plans to re-format it, then you really need to do nothing. It will print with the page breaks and inserted strings applied by gScript 2.0. However, if your script is already formatted and you want to reformat it with version 2.9, you must first strip formatting.

You can strip formatting using the older, gScript 2.0 template. Or you can run the StripFormatting macro contained in gScript 2.9. This macro will recognize a file created with version 2.0 and remind you to backup before continuing.

When you StripFormatting, the styles contained in GSCRIPT.DOT are automatically merged into the older document. This is important because there are several new styles in version 2.9: CharCont, Heading, ActDivision, and Headline.

You can manually merge styles into your older documents by following these steps:

1. Load the older document
2. Make sure that the current version of gScript is attached
3. Select Style from the Format menu
4. Select the Define button

5. Select the Merge button
6. From the list box of templates select GSCRIPT.DOT
7. Select the Merge button
8. Reply Yes when asked if you want to overwrite styles of the same name

What's different in 2.9?

The improvements and additions are too numerous to list exhaustively. If you've used 2.0 the magnitude of the difference should be evident.

However, there are some specific differences, relating to styles and FormatScript, that I would like to point out.

Options are now saved in a dedicated INI file named GSCRIPT.INI, stored in your Windows directory. You can delete the [gScript] section of your WIN.INI.

2.9 uses page view while formatting the script. The document, when in Auto mode, is zoomed down to fit into the current document work space. This is the fastest way to accomplish the more accurate breaks provided by this version. On lower resolution screens (becoming more and more rare these days), the document may look strange. If this bothers you enough to tell me I'd be interested.

The continued character name in broken dialogue used to be formatted with the More style. There is now a separate style (which you should never have to apply yourself) called CharCont.

In earlier versions of gScript hard page breaks were inserted between the bContinued and tContinued styles. This is no longer the case. tContinued is defined, during formatting, with "Page break before" set to true.

gScript still inserts a hard page break when formatting in Master mode and a page break falls within a paragraph of Action (Normal style).

During formatting, bContinued is virtually invisible because it is set, at the beginning of formatting, to a line spacing of 1pt. This is in the service of more accurate breaks.

Changing Default Information

Note: changing a template directly will have no effect on documents created before the modification. Changes are not retroactive.

gScript contains various layout and boilerplate settings which are automatically passed on to every new document based on GSCRIPT.DOT.

You can easily modify this information within the document. But it may be that you want to change this information for all subsequent documents based on GSCRIPT.DOT.

The easiest way to do this is to modify the template directly.

A template is just like a Word document. You can load and edit it as you would any other document. So, the first step in modifying the default definitions contained in GSCRIPT.DOT is to learn to load it:

1. FileOpen
2. Alt+T to pull down the list of File Types
3. Select Document Templates
4. Navigate through your directories until you are in your DOT-Path
5. Select GSCRIPT.DOT and press Enter

Title Page

You may want to change the information in the lower right to reflect your agent's address. Press F5 twice and select TitleRight. This will move you to the right hand cell of the table.

If you want to add a multi-line description (formatted as ByLine), do *not* insert a carriage return. That wouldn't work because this style contains a Space Before definition. Instead, insert a line feed: Shift+Carriage return.

Margins and Paragraph Indents

By default the margins in gScript are 1.5" left, and 1" right, top and bottom.

To change the default margins simply select PageSetup from the Format menu and fiddle the settings.

If you want to change both margins and the default paragraph indents for the screenplay styles, execute ChangeLayout by selecting Layout from the Tools menu or from the Control Center.

Header

The default header in gScript contains the title, the author, and the page. If you wish to change what will print on the top of every page (starting at page 2), you must edit the header in the template.

Load the template, select Header/Footer from the View menu. Select header and press Enter.

Since the default header uses field codes and tabs, it is advisable that you turn on ViewFieldCodes (Alt+V+C) and that you turn on ShowAll (Ctrl+Shift+8).

You can move from field to field by pressing F11.

Alter and fiddle the formatting of the header to suit your preferences.

A Table of gScript's Macros

These are the macros included in GSCRIPT.DOT. Those listed in *blue* italics are new to version 2.9.

Macro	Function	Access
ApplyChar	Character	Alt+C
ApplyDialogue	Dialogue	Alt+D
Applyheading3	Slugline (Heading 3)	Alt+S
ApplyNormal	Action (Normal)	Alt+A Ctrl+Shift+Numpad5
ApplyParen	Pops up a dialog box for the parenthetical text	Alt+P
ApplyTransition	Transition	Alt+B
AutoClose	Automatically runs BackupThisDoc if Backup is set to true in SetBackupOptions Unencrypted	
AutoNew	Runs whenever you create a new document based on GSCRIPT.DOT. It sets up various screen options (which you can change) and then runs the macro named	

	gsAutoNew Unencrypted	
AutoOpen	Runs whenever you open a document based on GSCRIPT.DOT. It maximizes the Word application and the document, then returns to the last edit location Unencrypted	
<i>BackUpThisDoc</i>	Backup the current document	Control Center
<i>ChangeLayout</i>	This macro allows you to change the number of lines skipped between scenes and between paragraphs of action or dialogue and the indents and widths of the screenplay elements	Control Center Tools Menu
<i>CheckForStranded</i>	Moves through the formatted document looking for stranded lines.	Alt+Ctrl+C
ConvertStyles	Converts style names to those expected by the FormatScript. This is particularly useful when you are converting a script already written in	Control Center

	Word for Windows or Word for DOS.	
ConvertThisPara	Converts a paragraph style to the style expected by the FormatScript. Does the same as Convert Styles, but on a paragraph by paragraph basis. This is useful if you have directly formatted paragraphs rather than styles.	Control Center
CreateTitlePage	Checks to see if a title page already exists and if not, inserts a new one, prompting you for information	Control Center
<i>EchoOnOff</i>	Turns the screen display on and off. This only works if you are using Windows 3.1 This macro does nothing when executed itself. It is called from various other macros.	
FormatScript	Formats the screenplay. This is the big cheese.	Control Center Tools Menu
<i>FudgeAllParas</i>	Reformats paragraphs that end with a line that consists of a single line	

<i>FudgeParaNarrower</i>	Decreases the width of the current paragraph by .1" with each execution.	Alt+-
<i>FudgeParaWider</i>	Increases the width of the current paragraph by .1" with each execution.	Alt++
gsAdvancedOptions	Allows you to change various settings	
gsAutoNew	Set up a new project. Called by AutoNew.	
gsControlCenter	A dialog box that displays buttons for several gScript functions. Unencrypted	Ctrl+Shift+C Toolbar Tools Menu
gsLib	Library of sub routines used by gScript macros (do not delete)	
InsertABScene	Allows you to number a scene with an alphabetic modifier: 1a, 1b, 11a.	Insert Menu
<i>InsertContinuing</i>	Inserts (continuing) parenthetical when a character's speech is broken by action description	Conctol Center

InsertOmittedScene	Allows you to specify that a scene or range of scenes has been omitted	Insert Menu
<i>ListGlossaries</i>	Shows just the gScript glossaries; allows you to add and delete glossaries	Alt+G Toolbar
NumberAllSlugs	Inserts a {seq slug} field to number all slug lines. Takes some time.	Control Center
NumberThisSlug	Numbers the current slug line	Alt+N
NumTopContinueds	Used by FormatScript to place scene length numbers at the top Continued; adds several minutes to the time needed to format a script	
<i>ReActivateScreen</i>	Necessary if Formatting is aborted when Echo is off	Alt+Ctrl+F10
RemoveBlankParas	Checks to see if the current document has blank paragraphs and removes them	
ResetAllDefaults	Gives you the option to reset the key, menu, toolbar, and GSCRIPT.INI defaults Unencrypted	

RevisionAccept	Accepts the currently selected revised text	Alt+Ctrl+A
RevisionSearch	Searches for the next instance of revised text	Alt+Ctrl+S
RevisionUndo	Undoes the currently selected revised text	Alt+Ctrl+U
RunHelp	Launches WINHELP.EXE with GSCRIPT.HLP	Alt+Ctrl+H Help Menu
SceneReport	Generate a new document with the extension RPT (same file name) containing a table of scenes and their length	Tools Menu
SetBackupOptions	Set options for BackupThisDoc	Control Center
SideBySideDialogue	Inserts a table for side by side dialogue	Control Center Insert menu
SmartEnter	Expands a glossary, if there is one and if the current style is Char. Then inserts a paragraph Unencrypted	
SmartTab	Uses tab to move from style to style intelligently	

	Unencrypted	
<i>SmartTabShift</i>	Uses the Shift+Tab to move from one style to another Unencrypted	
<i>SpaceBeforeDecrease</i>	Decreases the space before a paragraph by 1 point with each execution Unencrypted	Alt+Shift+-
<i>SpaceBeforeIncrease</i>	Increases the space before by 1 point with each execution Unencrypted	Alt+Shift++
StripFormatting	Resets your document to pre-formatting condition	
StripNumbers	Removes all {Seq slug} fields	
ToggleHidden	Turns the display of hidden text on and off	Ctrl+Shift+H
ToggleOutline	Toggles in and out of outline view	Ctrl+Shift+O
ToggleSlugIndent	Toggles the left indent of slug lines so that if there are numbers they will print in the gutter. If you have sluglines numbered, then the indent should be -.5; if you have no	Alt+Shift+I

	numbers, it should be 0". This macro toggles between the two. Speed Key: Alt+Ctrl+I
--	---

Advanced Margin Tweaking

This is a dangerous section to read. It tells you more than you may want to know, and should be used with care. You only need to know about this if you have formatted a script and found single line pages and want to try fiddling some numbers to fix the situation rather than fiddling your words.

When breaking in Auto mode, to create a Shooting script (a script with top and bottomed continueds), there are now four possible options which effect how much text is placed on each page. These four options are a combination of two settings, each with two possible switches.

First, you can alter the default margins: either Normal (1") or Tight (.75"). Second, you can specify whether or not Widow and Orphan Control is set to ON for this document.

Note: the gScript switch has the opposite syntax to the Word option: gScript is saying "Should I allow widows" while Word is saying "should I turn on Widow and Orphan Control (and therefore not allow widows?)". So, AllowWidows = 1 is equivalent to the "Widow and Orphan Control" box in ToolsOptionsPrint dialog box being unchecked.

Below is a table that describes the general relation of these two options to the amount of text on the page:

Setting:	Result
Normal (1") and AllowWidows = 0	Most whitespace, least likely to strand continueds.
Normal (1") and AllowWidows = 1	Medium (conservative all around)
Tight(.75") and AllowWidows = 0	Most efficient

Tight(.75") and AllowWidows =
1

Least white space, most likely to
have stranded continued.

When Widow and Orphan Control (usually set in ToolsOptionsPrint) is OFF (unchecked), Word will allow a single line at the top and bottom of a page. For FormatScript this means that, for instance, a three line paragraph of action can be broken after line one (leaving a single line at the bottom of page x and the other two lines on page x+1). This also means that a new scene, in which only the first line of action will fit on the bottom of a page, will not be pushed to the next page). When trying to fit as much as possible on the page, this is a nice option. HOWEVER, setting AllowWidows=1 (or true) increases the likelihood of stranded continued.

One of the major requests in version 2.0 of gScript was to have more control over how many lines will fit on a page. The method of doing this was imperfect. If I set it up to add a line on most pages, it would create more stranded lines. If I was conservative, some pages would have too much white space.

Version 2.9 does several things to rectify this. For one thing, it now uses PageView during formatting. It allows you to choose the more conservative "Normal" settings rather than Tight.

And, for the braver, it allows you to change settings in GSCRIPT.INI to try and best suit an individual screenplay.

In GSCRIPT.INI you will find two settings:

NormalBottomPoints=68

TightBottomPoints=44

These keywords determine how many points to set the bottom margin during formatting. If you increase this number, you are less likely to have stranded lines (and likely to fit fewer lines on a page). If you decrease this number, you increase the chances of stranded lines, and increase the number of lines per page.

If you feel like fiddling, change this number in increments of 2 in both directions and see what difference your settings make.

The default settings are the ones that have worked best on the five or so sample scripts I have used for testing. I am hopeful that with enough testing, by enough people, on enough scripts, we will settle upon a setting that works best.

Stranded Continueds

Though I've tried to make both Tight and Normal as accurate as possible -- striving to fit as much on the pages as appropriate -- there will still be times when, because of the nature of the last paragraph on a page, gScript will put one line too many on a page and force the bottom continued break to the next page. That's the bad news. The good news is I've come up with several small tools to help you rectify this manually (and I'm working on an automatic solution).

The manual solution consists of three parts: finding the stranded continued, making appropriate paragraphs wider, and decreasing the space before selected paragraphs.

Finding stranded continueds

There is a macro, available from Control Center and by pressing Alt+Ctrl+C, named CheckForStranded.

Once you locate a stranded continued, move to the previous page, and locate any paragraph that ends in a single word. Then fudge that paragraph's right indent (making it wider).

Making paragraphs wider

Two new macros:

FudgeParaWider - increases the current paragraph's width by .15 each time it is pressed, , accessed with Alt+=

FudgeParaNarrower - decreases the current paragraph's width by .15 each time it is pressed.accessed with Alt+-

For instance, if a paragraph ends with a single word of three characters, pressing Alt+ twice should wrap that paragraph, saving a line on the current page.

Decreasing Space Before a paragraph

If the Continued doesn't move to the current page, you can also select paragraphs and change the default space before. That is, the default space before is 1 li (or 12 pt). By pressing Alt+Shift+, you decrease the Space Before a paragraph by one point.

SpaceBeforeIncrease - increases the current paragraph's Space Before setting by one point, accessed with Alt+Shift+=

SpaceBeforeDecrease - decreases the current paragraph's Space Before setting by one point, accessed with Alt+Shift+-

GSCRIPT.INI Settings

Most Windows applications use an INI file to hold various settings and preferences. Some keep such values in a special section of WIN.INI. Others write dedicated INI files. In this version of gScript there is a dedicated INI file named GSCRIPT.INI stored in your Windows directory.

An INI file is a "pure text," or ASCII file that can be loaded into an editor (such as NOTEPAD.EXE) to be viewed or altered.

When you installed gScript, a default GSCRIPT.INI was included that contained the following settings.

Keyword=Default Setting	Meaning
TCont=CONTINUED:	Top Continued String
BCont=(CONTINUED)	Bottom Continued String
More=-more-	String inserted to indicate broken dialogue
DCont=(cont'd)	String inserted (along with Character Name) before continuation of dialogue
Mode=0	Automatic or Interactive
Scene=1	Master or Shooting
NumTop=0	Include numbers left of top continued
ScenePages=0	Count pages to a scene
NewFile=0	Save to new file before formatting
Renumber=0	Renumber sluglines before formatting
Iconize=0	Iconize Word while formatting. The Icon label will display progress reports. This allows you to continue working in other applications, but realize that things will definitely slow down

Update=0	Update fields before formatting
EchoOff=0	Turn off display during formatting
EchoOffStrip=1	Turn off display during strip formatting
EchoOffNumber=1	Turn off display during stripping numbers
Side=1	Slugline number placement: 1=Left, 2=Right, 3=Both
Backup=0	Automatically backup doc at FileClose
OverWrite=0	Warn if overwriting with backup
Backup-Path=a:\	Backup destination
AllowWidows=0	1 = Allow single lines of action at the bottom of the page, 0 = Widow and orphan control. Recommend leaving this set to 0,
Margins=1	Vertical Margins 0 = Normal .1 = Tight
NormalBottomPoints=68	Points for normal vertical during formatting
TightBottomPoints=44	Points for tight vertical during formatting

NoLeading=0	Whether or not to force line spacing to the font size
PostOption=3	What to do when gScript completes formatting
FudgeFirst=0	FudgeAllParas before formatting
SmallWords=4	FudgeAllParas will wrap words of this length that end a paragraph. Legal: 2, 3, and 4
Messages=1	When 1, non-critical messages are displayed on the status bar when 0, a message box is displayed
SmartTabs=1	Toggle SmartTab. This must be set from Options, not manually
SmartEnter=1	Toggle SmartEnter. This must be set from Options, not manually
UnprintableRegionIsHalfInch=0	The following setting, when 1, tells gScript that your printer cannot have margins smaller than .5". This setting is necessary for DeskJet
PadSpaces=2	Spaces between top CONTINUED: and scene page count.
UsePageView=1	Force use of pageview when formatting. Do not change.

If, for some reason, your GSCRIPT.INI file gets lost or corrupted you can do one of two things: recopy the default GSCRIPT.INI from your installation disk (or archive); execute ResetAllDefaults.

The first method is preferable.

Price, Support, and Updates

Price and Ordering

gScript 2.9a costs \$99.95. Upgrades from prior versions are \$20.00.

This price includes a printed and bound manual and the Windows installation utility.

You will be notified of major upgrades via mail.

Orders for gScript are handled by:

Advanced Support Group
1900 Grant Place
St. Louis, MO 63131

1-800-788-0787 (orders only)

Support

There are two ways to get support for gScript.

- Advanced Support Group: 1-314-965-5630
- Contact the author directly either at the electronic or the mail addresses listed below

History

Version 1.1 -- first release

Version 1.1a -- bug fix update

Version 2.0 -- upgrade to take advantage of new features in Word for Windows 2.x

Version 2.9 -- This release. Added many new features and improved the handling of page breaks in both Master and Draft mode. If you are wondering what happened to versions 2.1-2.8 -- there weren't any. I use 2.9 to indicate that this is a

major revision. But I don't want to call it 3.0. The version released for Word 3.0 will be numbered gScript 3.0.

Version 2.9a -- Improved the SceneReport macro; updated the ListGlossaries macro to work around a bug in Word for Window's handling of glossaries (sometimes abbreviations are expanded in the wrong font); fixed FormatScript's handling of ActDivisino numbering.

Possible additions to future versions

Increased automation of Alpha and Omitted scenes, and a-b page numbering.

The ability to re-format a formatted script to determine if pages have been added

Better handling of two column dialogue

Reaching the Author

Electronic Mail Addresses:

CompuServe ID# - 71171,3555

MCI Mail: gjgallo

BIX: ggallo

Internet gg2@columbia.cu

guygallo@factory.com

SmartNet Network, Windows Conference

InterLink Network, Windows Conference

WGA BBS #938

Mailing Address

Guy Gallo

219 East 69th Street

New York, NY 10021

Bugs and Suggestions

This release (2.9) is a major revision. Please hammer on the program and report any problems or bugs or suggestions to the above address.

Also, I am interested in hearing user suggestions for features in future versions of gScript.

Notices

gScript is copyright 1991, 1993 by Guy J. Gallo. No portion of the macros contained in this template, or this documentation, may be modified, copied, distributed or otherwise altered without the express permission of the author. This includes, but is not limited to, basing a similar application upon the macros contained in gScript, distributing the package for a fee via any method, disk or electronic.

gScript is provided as is. I make no warranties and take no responsibility for possible data loss or corruption. With a system this complex there is always the possibility of error. **Make backups first...**

About the Author

Lest you think this template is the creation of a Word for Windows programmer who knows nothing of screenwriting, I have written screenplays, including *Under The Volcano* (1984), an adaptation of *Adventures of Huckleberry Finn* (PBS 1986), a single episode of Tales from the Darkside: *The Enormous Radio* (1987), and, of course, the large handful of scripts withering in development hell.

I teach screenwriting in the Film Division of Columbia University.

Putting this together is one of the myriad ways I avoid finishing the novel...

Glossary of Terms

Auto Macros

An Auto Macro is a macro that is executed every time a condition is met. AutoNew is executed every time a new document is based on the template containing the AutoNew Macro. AutoOpen is executed every time a document based on the template containing AutoOpen is opened. Other Auto Macros are AutoClose, AutoExec, and AutoExit. See the user manual or HLP file for more detail.

Document Templates

A template is a specialized Word for Windows file which contains style information, macros, glossaries, menu, key, and toolbar assignments to be associated with a class of documents (like screenplays). When you create a Word for Windows document you base it on a template. GSCRIPT.DOT is one such template.

DOT-Path

The directory containing all of your DOT files. By default this is your Word for Windows directory. It can be any directory, but if you have moved your templates you must have a DOT-Path variable set in WIN.INI.

Field Codes

A "field" is a special code inserted into a Word document that displays information or performs an action. gScript uses fields for three functions: title page, header, and numbering scene headings. For more information on field codes see the *Word for Windows User's Guide*, chapter 41, or *Take Word for Windows to the Edge*, chapter 11.

Headline

A style used for single lines that are mid-way between Action and Slugline. That is, it requires more emphasis than Action but should not be numbered like a Slugline. For instance:

FADE UP:

or

INTERCUT:

might be formatted as **Headline**.

Hidden Text

Word for Windows allows you to define a style as "hidden". Text formatted with this style will display on the screen only when `ToolsOptionsView` is set to either `display hidden` or `ShowAll`. Text formatted as hidden will not print unless you explicitly tell Word for Windows to do so. In gScript the styles `heading 1`, `heading 2`, and `HiddenNote` are defined as hidden.

Paragraph Styles

A style is a collection of indentation, font, and spacing information saved under a specific name. For instance, "Char" contains the indentation and font information for the character name paragraphs.

Scene Slugline

The line marking the beginning of a new scene or camera setup within a scene. Sometimes called a Scene Heading. A Slugline usually has the format:

INT. LOCATION - TIME

Distinguished from a "Headline" by the fact that in Shooting Script Mode a Slugline is numbered while a Headline is not.

Stranded Continueds

Sometimes, when formatting in Shooting mode, gScript will miscalculate where to place a break. The result is that a single line, usually the bottom continued, will be forced to the next page. It will be "stranded" all by its lonesome.

If you encounter this phenomenon, try the following: Uncheck `Allow Widows` in the `Esoterica` dialog box and Use 1" margins rather than .75".

Take Word for Windows to the Edge

My book on maximizing the power of Word for Windows. ISBN 1-56276-079-3. Published by Ziff-Davis Press, 1-800-688-0448 and your local computer bookstore.

Index

A

Abbreviations 8, 22
Act Numbers 16, 23
 ActDivision style 13, 16, 23

B

Backups 35
 Options 51

C

Carriage Returns 14, 24, 50
Colors 23
Cost 69

F

Field Codes 15, 16, 17, 40, 59
 Defined 73
 Locking 46
 Seq field 25
 StyleRef 16
Font 1
FormatScript 1, 3, 12, 25, 27, 30, 33
 Removing formatting 4, 54
 Style assumptions 11
Frozen screen 49

G

Glossaries 22, 47, 62
gScript 2.0 57

GSCRIPT.INI 5, 25, 31, 49, 50, 58, 66

H

Header 6
Hidden text 12, 22
 Toggle view 54
History 69

I

Iconizing Word 30
Installation 5

L

Layout 3, 11, 31, 51, 58
 Report file 51, 58
 Warning 31, 37

M

Margin
 Tweaking 64
Margins 31
 Vertical 31
Multi-tasking
 NOT 30
My book 74

N

Numbering Scenes
 Omitted 46
Numbering scenes 25, 48
 Alpha 45
 Removing 54

O

Options 2, 20, 28, 30, 42
 Formatting 2, 28, 30, 42
Outline 21, 55
 Possible bug 22
 Speedkeys 21
 Toggle view 21

R

Removing formatting 4, 54

Colors 23

Revision Marks 23, 50

S

Scene Report 51

SideBySideDialogue 52, 63

SmartEnter 44, 52

SmartTab 19, 44, 53, 63

Stranded continueds 28, 31, 60, 64, 65, 74

Styles

Act division 13

Action 12

Applying 14

Character name 11

Defined 11

Dialogue 11

Header 16

Heading 13

Headline 13, 25, 73

Line spacing 14

Merging 57

Parenthetical 11

Slugline 12, 55

Transition 12

Suggestions 70

Support 69

T

Template 1

ToolBar 20

U

Updates 69

V

View Settings

Initial 35