# Contents

# 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 ) 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 and Layout.

# 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 FormatScript.

## 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 FormatScript.

# 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:

> 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.

# Files and Installation

You must have the following five files located in the specified directories:

| File | Location | Purpose |
|---|---|---|
| GSFDOC.DOC | In your DOT-Path or a separate gScript directory | This documentation. It is, in fact, a Word for Windows template containing several macros to run the demo and set defaults. |
| GSFDEMO.DOC | DOT-Path or a separate gScript directory | A sample screenplay used by the gScript Demo. |
| 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. |

How you get these files into the appropriate directories depends where you got gScript and how you feel about manually moving files around as opposed to automatic installations.

**If you got gScript on disk:**

There is a Windows setup application that will automatically copy the necessary files to their directories. It will also optionally create Program Manager icons for the documentation and demonstration files.

Simply insert the diskette in drive A: and enter **A:\SETUP** in the Run dialog box of either Program Manager or File Manager.

**If you got gScript from an on-line service as GSF29.EXE:**

You can either manually copy the five gScript files to the directories specified above, or you can use the supplied DOS batch file INSTALL.BAT.

The syntax for INSTALL.BAT is:

INSTALL <DOT-PATH> <WINDOWS-PATH>

You must supply both the DOT-Path and the Windows directory. For example:

INSTALL C:\WINWORD C:\WINDOWS

would copy GSCRIPT.DOT, GSFDEMO.DOC, and GSFDOC.DOC to C:\WINWORD, and copy GSCRIPT.INI and GSCRIPT.HLP to C:\WINDOWS.

*Note: If you are updating from a previous version of gScript you may want to copy your old version of GSCRIPT.DOT to another name (such as GSF20.DOT). Please see Updating Older gScript Documents for more information about ensuring that gScript 2.9 will work with your existing documents.*

*Also, be sure to remove older versions of the GSCRIPT.HLP file. In gScript 2.0 the help file was located in your Word directory.*

# 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*.

## Related Topic:

Creating a New Project

## Creating a New Project

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

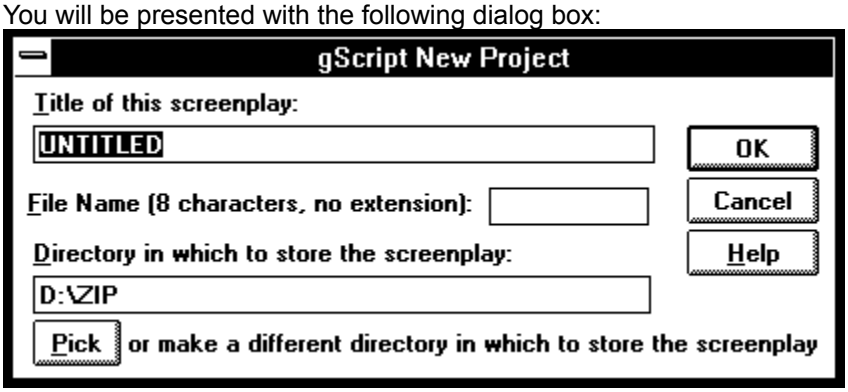
Select New from the File menu.

Highlight GSCRIPT in the list of templates

Press Enter

If you have not yet registered gScript you will be presented with a reminder dialog box. This reminder box will display under two circumstances: when you create a new screenplay and when you format an existing screenplay. It will not appear randomly when you are using the other macros and facilities in the package.

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.

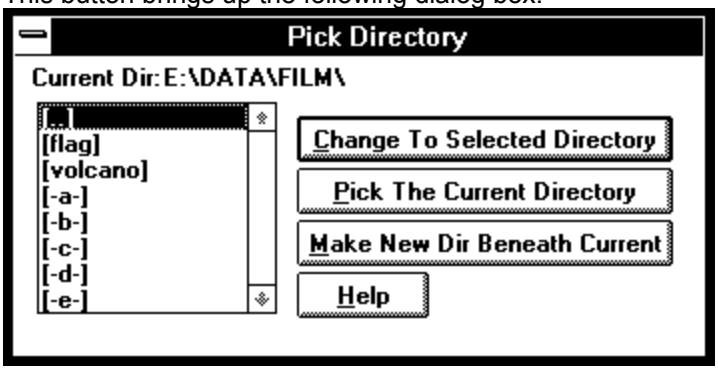
You can type in a different path. The only limitation is that the path must already exist.

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.

## *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 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 and ConvertThisPara.

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.

The following topics describe the pre-defined and built-in styles used by gScript.

## Related Topics:

Definitions and Usage
Styles and "follow" specifications
Styles and Line Spacing
Applying Styles
Headers and Footers
The Title Page

## 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<br>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.

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.

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.

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. Inlcudes 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

## 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.

## 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.

### Related Topics:

Header Information
Header Format
No Footer

### *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. Preceed 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 anominous 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.

### *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 accomodate the new strings. It is possible that a footer would intefere 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.

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.

## Related Topics:

Keyboard Customizations:
Menu Item Additions
The Toolbar in gScript
gScript's Use of Outline Levels
Glossaries/Abbreviations

## 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 |

## Related Topic:

Note on Modifying gScript's Key Assignments

## *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.

**Added to Insert**

**Title Page** - Runs CreateTitlePage,

**AB Scenes** - Runs InsertABScene,

**Omitted Scenes** - Runs InsertOmittedScene,

**Side By SideDialogue** - Inserts a two column table and the necessary styles to begin entering side-by-side dialogue.

**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.

**gScript Options** - Displays the Options menu.

**Format Script**... - Runs FormatScript on the finished draft.

**Scene Report** - Generates a scene report document containing a table of scenes, their page and their length.

**Remove Formatting** - Runs StripFormatting to remove the styles and strings inserted by FormatScript.

**Added to Help**

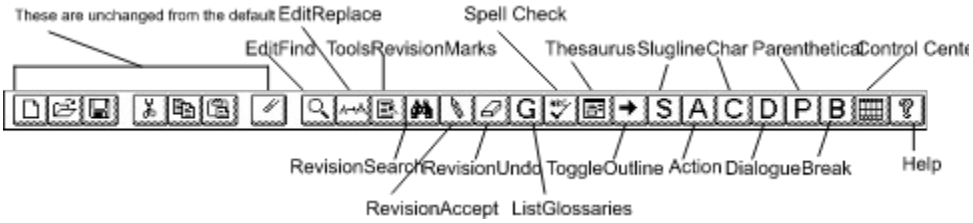
**gScript Help** - Loads the gScript Help file.

**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

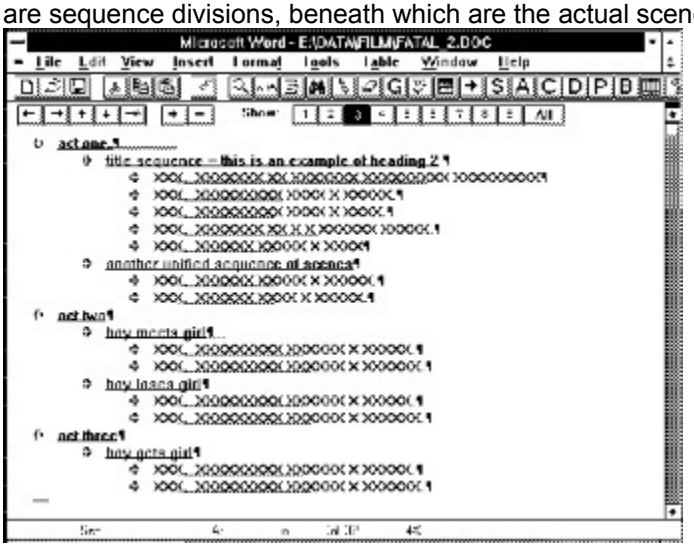
Below is a legend for the toolbar visible in any document based on GSCRIPT.DOT.

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.

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 defiined 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 preceeding 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<br>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.

In addition, SmartEnter provides a convenient alternative way to expand glossaries within paragraphs formated as Char.

**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-definied 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 whereever 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.

## 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.

### Related Topic:

Why {Seq Slug} instead of {Autonum}?

## *Why {Seq Slug} instead of {Autonum}?*

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 {autonum}. 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.

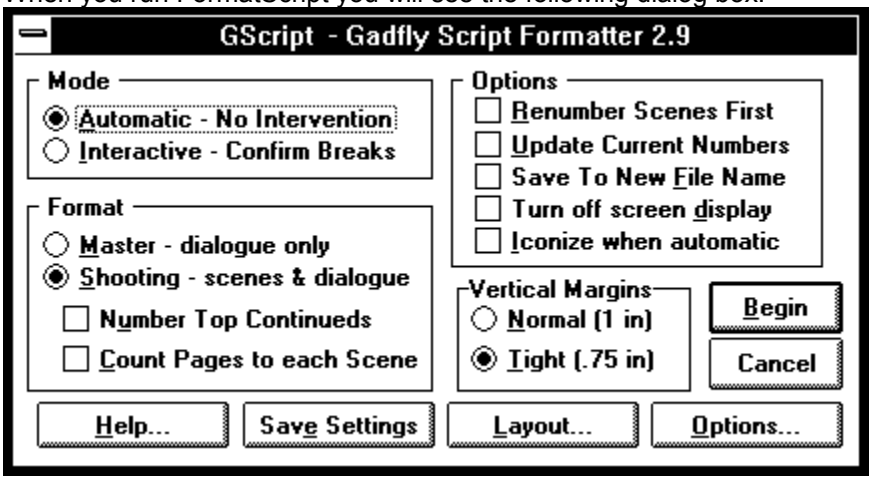
# FormatScript - Introduction

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:




## Related Topics:

Mode
Format
Options
Vertical Margins
PushButtons

## 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.

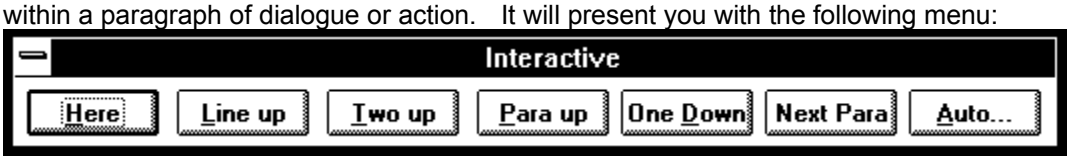
**Automatic - No Intervention**

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:** 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.

# 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.

# 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

## ApplyParen

Speedkey: Alt+P

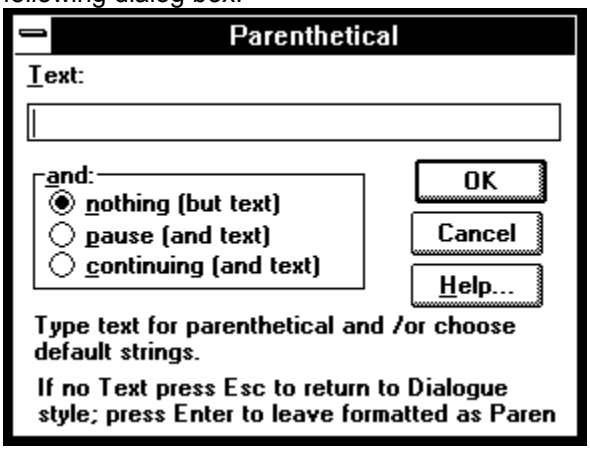
Menu: Format

There are six macros devoted to applying the paragraph styles associated with the most frequently used paragraph styles: ApplyChar, ApplyDialogue, ApplyHeading3, ApplyNormal, ApplyParen, and ApplyTransition. All but ApplyParen simply apply the named style. They are in macros so that they can be assigned to the toolbar and to the Alt+Key speedkeys.

ApplyParen, however, is a special case. It does more than simply apply the style Paren.

It's smart: You can execute ApplyParen while in a paragraph of Char or Dialogue and the macro will know that what you want to do is insert a parenthetical *after* the current paragraph.

It's smarter: After determining whether or not to insert a new paragraph, ApplyParen displays the following dialog box:




This dialog box allows you to enter parentheticals in three ways:

- typing directly into the Text edit box
- selecting pause or continue from the option buttons
- both typing in the Text box and selecting pause or continued

For instance, Alt+C with nothing in the text box would insert

(continuing)

Alt+P with "she collapses" in the Text box would result in:

(pause; she collapses)

ApplyParen automatically inserts the parentheses and inserts a new paragraph of Dialogue style. If the paragraph *following* the parenthetical is already Dialogue or DialogueItalic, ApplyParen does not insert a new paragraph.

If you execute ApplyParen on an existing parenthetical the text between the parenthesis is inserted into the Text edit box.

There are two ways to abort ApplyParen:

- Cancel (or Esc) returns the insertion point to its position prior to executing ApplyParen.
- Enter with a blank Text box and the nothing button selected leaves the insertion point in a new paragraph of parenthetical

**Note:** If you execute ApplyParen on a paragraph other than Char, Dialogue, DialogueItalic, or Paren an

error message will display on the status bar.

## AutoClose

This macro is executed every time you close a document based on GSCRIPT.DOT.

It checks to see if you have "Backup file on closing" set to yes (checked) in the Backup Options dialog box.  If so it executes BackupThisDoc.

## AutoNew

This macro will run whenever you create a new document based on GSCRIPT.DOT. It executes the macro gsAutoNew, which prompts the user for document information. See Creating a New Project.

## AutoOpen

This macro will run whenever you open a document based on GSCRIPT.DOT.

You can edit this macro to suit your own startup preferences.

As shipped this macro:

- Sets the view to Normal
- Sets the Zoom to "best fit"
- Turns off ViewFieldCodes
- Maximizes the Word Application
- Maximizes the opened document
- Returns to the last edit location

You can add to (or subtract from) this macro if there are any view preferences you wish to force into effect when you open a screenplay document.

What view settings you will want depends heavily upon your particular video setup (and your well-established habits). I offer these as a guideline, not as a necessity.

If you press Alt+O+M, select AutoOpen, and Alt+E (to Edit), you will see the following macro (egad, some of you will say; it ain't that scary. Really.):

```
Sub MAIN
REM The following changes various views in Word to prepare for editing
REM The settings uses reflect my own preferences.  To remove any of the
REM follwing simply preface that line with an apostrophe or REM
ViewNormal                          'Toggle to Normal View
ViewZoom .BestFit                   'Show text as large as will fit
ViewFieldCodes 0                    'Turn off Field Codes
If Not DocMaximize() Then DocMaximize       'Maximize the newly opened document
If Not AppMaximize() Then AppMaximize       'Maximize Word
If ExistingBookmark("\PrevSel1") Then EditGoTo "\PrevSel1"   'Return to the last edit location
End Sub
```

Perhaps the only line I would urge you to keep is the last. It moves to the place where you left off last time you closed the screenplay. I find that particularly useful.

All of the other settings have to do with varous view settings. You can experiment by placing a "comment" -- an apostrophe or the letters REM in front of the lines you might do without.

If you like these settings when working on a screenplay, but don't want them to be in effect for other documentss, you can edit the AutoClose macro to reset any view settings you change to your preference.

Here's where I can get in another plug for my book, Take Word for Windows to the Edge.

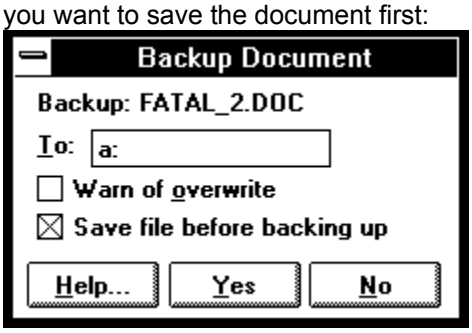
## BackupThisDoc

Speedkey:  None

Control Center:  Yes

Copies the currently active document to a specified backup directory (usually A:, but it could just as easily be a directory on your hard disk).

If the file has not been saved before executing this command you will see a check box option asking if you want to save the document first:



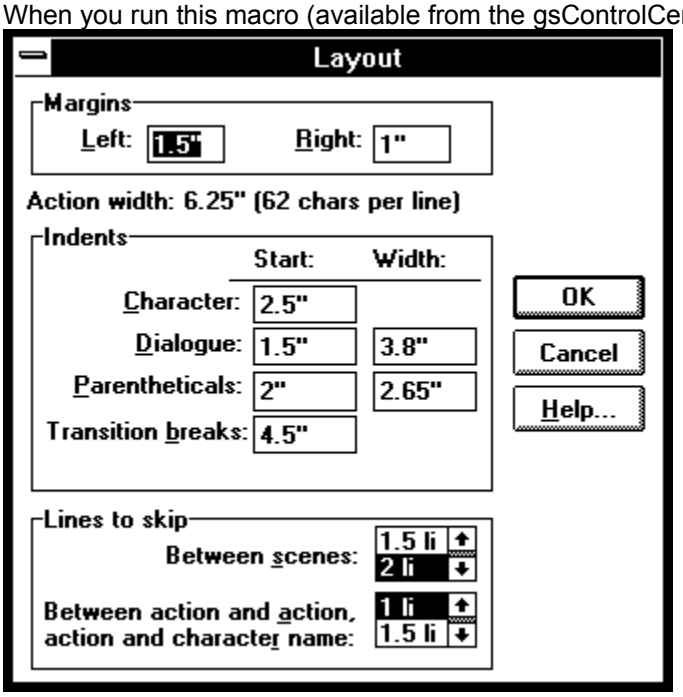
# ChangeLayout

Speed Key: None.

Control Center: Yes

Purpose: This macro allows you to change the number of lines skipped between scenes and between paragraphs of action or dialogue.

When you run this macro (available from the gsControlCenter), you will see the following dialog box.




**Indents**

This command makes it easier to change the indents and widths of the standard paragraphs. For one thing you can do it all at once. For another you don't have to fiddle with style definitions.

All measurements for starting positions are taken from the left margin, not from the left edge of the page. So, for instance, in the default GSCRIPT.DOT, which has a 1.5" left margin, Dialogue will start at approximately 3" from the left edge of the page (fudging allowed for the "unprintable region" of .25" necessary for most printers).

What this function is actually changing is the Left Indent (Start) and Right Indent (which rules the width) of the Normal, Dialogue, Char, Paren, and Transition styles.

**Lines to skip**

Since gScript removes all extra paragraphs, you should never try to increase the amount of space between scenes or paragraphs by inserting an extra carriage return. Instead, use ChangeLayout to modify the number of lines to place between the end of one scene and the next slugline and the number of lines to skip between action and action and action and character name.

What this function is actually changing is the Space Before setting for the styles heading 3 and Char.

**Margins**

This performs the same function as modifying Left and Right settings in FormatPageSetup. You cannot change the top and bottom margins for the compelling reason that FormatScript takes over those settings when it reformats your script.

*Note: You should make any layout changes* before *you run a screenplay through FormatScript. Since FormatScript calculates according to page breaks it requires that all settings that affect pagination have already been established. In fact, if you change any of the layout settings on an already formatted script, the chances are good that you will force repagination that results in stranded lines.*

## CheckForStranded

Speedkey:  Alt+Ctrl+C

Control Center:  Yes

This macro should be run *after* formatting a script.  It begins from the current insertion point and moves to the end of the document, stopping when it hits a page with less than 256 Characters (a good sign that something's amiss).

When it encounters such a "short page" it stops, beeps, and moves to the previous page.

You can then see if there are any paragraphs you can "fudge" using FudgeParaWider or SpaceBeforeDecrease in order to fit the stranded next page text on the current page.

See Stranded Continueds.

## ConvertStyles

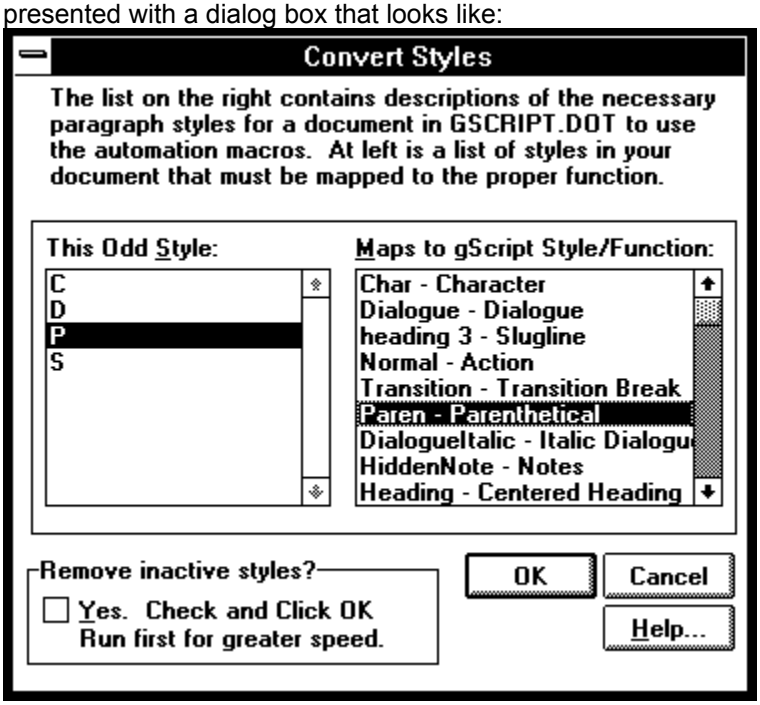
Speedkey: None

Control Center: Yes

Converts style names to those expected by the FormatScript.

This is particularly useful when you are converting a script already written in Word for Windows or Word for DOS.

If the screenplay is based on a template, with styles defined for the six screenplay paragraphs (Slugline, Dialogue, Parenthetical, Action, Character Name, and Transition), then simply run this macro. You will be presented with a dialog box that looks like:




The left hand list box contains all of the styles currently defined in the document. The right hand list box contains eight functions -- the six basic screenplay paragraph styles, plus Italic Dialogue and Notes.

You simply select a style in the left box and the function it corresponds to in the right box. In the above example, "C" would map to "Character", "D" would map to "Dialogue".

What the macro actually does is a search and replace for all paragraphs styled as "C" and changes them to "Char" style (Char is the name for the Character Name style).

The check box titled "Remove inactive styles?" performs a very interesting and useful function. Let's say you've imported a screenplay from Word for DOS. The resulting document contains all of the styles defined in the Word for DOS style sheet. Yet the document only uses a portion of those styles. Clicking on this option and then selecting OK searches through the list of available styles, checks to see if a style is actually used in the document, and deletes those that are not active.

This considerably shortens the list of styles displayed in FormatDefineStyles. And if it is run first, it speeds up the operation of ConvertStyles.

Hint for Word for DOS users: in earlier versions of Word for Windows the conversion of a Word for DOS style sheet mapped styles using the mnemonic key combination for the name in Word for Windows. For instance, a style accessed with Alt+D was copied into Word for Windows as a style *named* D. This always struck me as odd, and, thankfully, has been changed. Now the name of the new style is not based on the mnemonic, but on the style description.

So, if you want a Word for DOS screenplay style sheet to map accurately to Word for Windows, go into the Gallery and edit the styles so that the descriptions match the specifications found in the table on page 10.

## ConvertThisPara

Speedkey:  None

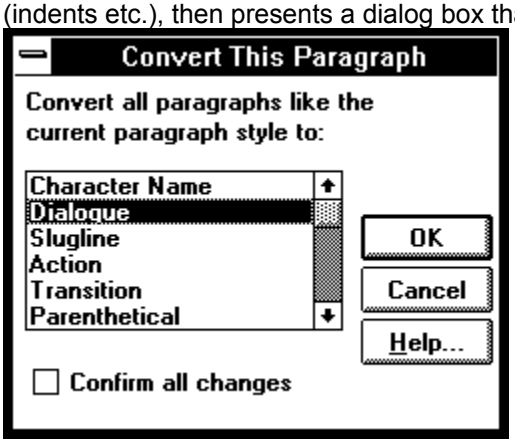
Control Center:  Yes

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.

What happens if you have directly formatted your screenplay using the ribbon and ruler?  The result would be that all paragraphs would be variations on the Normal style, just with different indents.

In this case, you can use the macro named ConvertThisPara.  What this macro does is look at the current paragraph (the one containing the insertion point), determines all of its specific characteristics (indents etc.), then presents a dialog box that looks like:




You simply select the function that all instances of this paragraph should map to, and the macro searches through the document for paragraphs with the same characteristics and formats them with the appropriate gScript style.

## CreateTitlePage

Speedkey: None

Menu: Insert

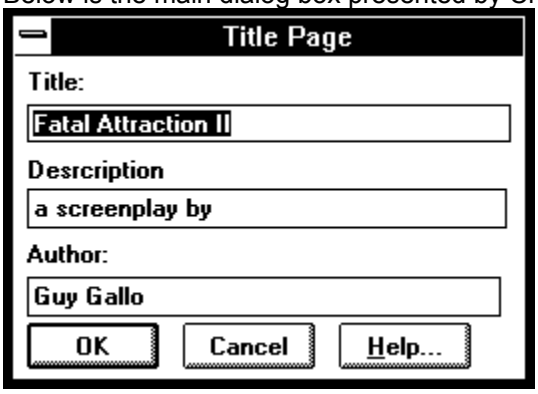
Control Center:  Yes

This macro has been significantly revised in version 2.9 of gScript.

When executed from the Insert menu or the Control Center CreateTitlePage first checks to see if there is an existing title page in the current document.  It does this by looking for a bookmark named TitlePage.

If there is an existing TitlePage, the insertion point will be left at the top of the document so that you can manually edit the title page.

Below is the main dialog box presented by CreateTitlePage




The title and author information in this dialog are taken from the FileSummaryInfo dialog box, which, under normal circumstances, you filled out when you first created the project.  You can, of course, edit the information for the title page.

If you change the title and/or author in CreateTitlePage, that information will be updated in FileSummaryInfo

If you cancel the dialog box CreateTitlePage will restore the previous title page (if there was one) and do nothing if there wasn't.

If you OK the dialog box CreateTitlePage does the following:

- Checks to see if the Author and Title information needs to be updated in FileSummaryInfo and does so if necessary
- Inserts a paragraph formatted as Title style
- Inserts the field {title} which will display the Title information from FileSummaryInfo
- Inserts a paragraph formatted as ByLine style
- Inserts the text entered into the Description edit box
- Inserts a paragraph formatted as TitleAuthor style
- Inserts the field {author}, which will display  the Author information from FileSummaryInfo
- Inserts a two column table which is absolutely positioned, using Frames, at the bottom of the page
- Formats the left hand cell as TableLeft and promptd for text
- Formats the right hand cell as TableRight and promptd for return address information

It is important to understand that this title page uses fields.  If you are completely unfamiliar with fields,

please review the User Reference Chapter 41.

Note: If you inserted a title page manually (that is, you deleted the default title page created when you create a new project **and** you typed a title page directly into the document rather than using this facility) it is possible that CreateTitlePage will not recognize the existing title page. The lesson here is: if you create a title page manually, cover it with a bookmark named TitlePage.

## EchoOnOff

This macro is a "library."  It is called by other macros to turn the screen display on and off.  It does nothing (but print a warning) if executed by itself.

If you run this macro under Windows 3.0 you may see a WordBasic error message.

Leave it alone.  Don't delete it.

## FormatScript

Speedkey: None

Menu:  Tools

Control Center:  Yes

Formats the screenplay.  This is the big cheese.

FormatScript - Introduction.

## FudgeAllParas

Speedkey: None

Control Center:  Yes

This is a new addition to gScript 2.9.

Word automatically wraps paragraphs as you type.  It is often the case that this will result in a last line consisting of a single, short word.  That doesn't bother everybody.  It sometimes bothers me.  Especially if I'm trying to conserve space and shorten the total page count of the screenplay.

This macro will go through your document and search for paragraphs that end with words of 4 characters or less.

You can change the number of characters FudgeAllParas will consider "too small" by changing the value in the "Wrap words that are X characters long" found on the Options dialog box.  See gsAdvancedOptions.

FudgeParaNarrower

FudgeParaWider

## FudgeParaNarrower

Speedkey: Alt+Ctrl+-

Increases the right indent for the current paragraph (and consequently makes the paragraph narrower).

## FudgeParaWider

Speedkey: Alt+Ctrl+=

Decreases the right indent for the current paragraph (and consequently makes the paragraph wider).

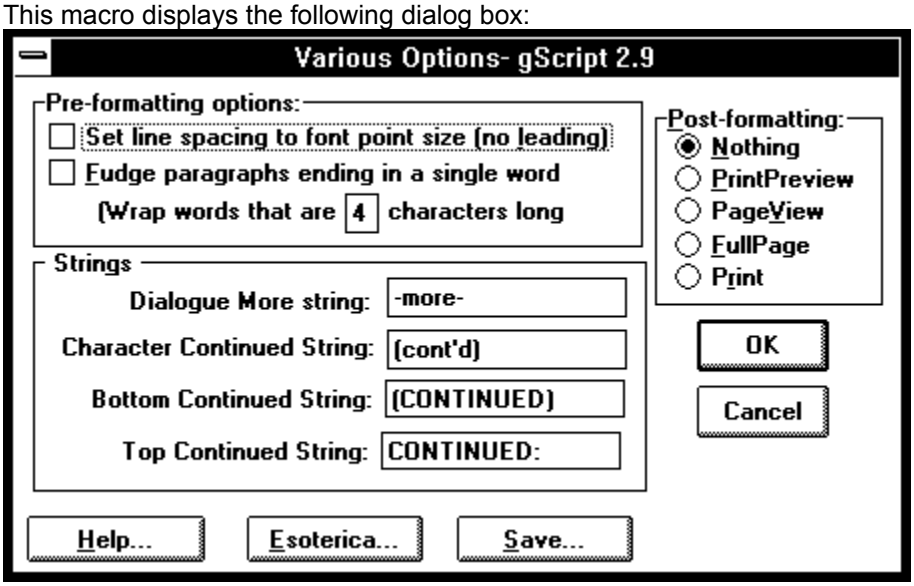
# gsAdvancedOptions

Speedkey:  None

Control Center:  Yes

Menu: Tools

This macro displays the following dialog box:




## Related Topics:

- Pre-formatting options
- Strings
- PostOptions
- Esoterica

## *Pre-formatting options*

**Set line spacing to font point size (no leading):**

When line spacing is set to Auto (in FormatStyles) Word cheats. A twelve point font actually has lines that are 13 points high. That extra one point is called "leading" in typesetting parlance.

If you want to squeeze, ever so slightly, the text in your screenplay, check this option.

In fact, with this option checked your screenplay will print more like it would on a typewriter.

**Fudge paragraphs ending in a single word**

When checked, the macro FudgeAllParas is executed before the formatting begins.

**Wrap word that are X characters long**

This setting changes the operation of FudgeAllParas. Legal entries are 2, 3, and 4.

## *Strings*

These are the specific strings that will be placed in your script at page breaks.

**Dialogue More String**

What will be inserted below a broken paragraph of dialogue::

```
                FRANK
We talked quite reasonably for a long
time.  I poured her one and me one.
And so I said to Gertrude, I can't
                --more--
```

**Character Continued String**

What will be inserted next to the character name on the top of the following page, when a page break is inserted into a paragraph of dialogue, e.g.:

```
                FRANK  (cont'd)
take this anymore.  Then she hauled
off and hit me with a skillet.
```

**Bottom Continued String**

What will be inserted at the bottom of the page when a scene continues:

```
(CONTINUED)
```

**Top Continued String**

What will be inserted at the top of the page when a scene continues:

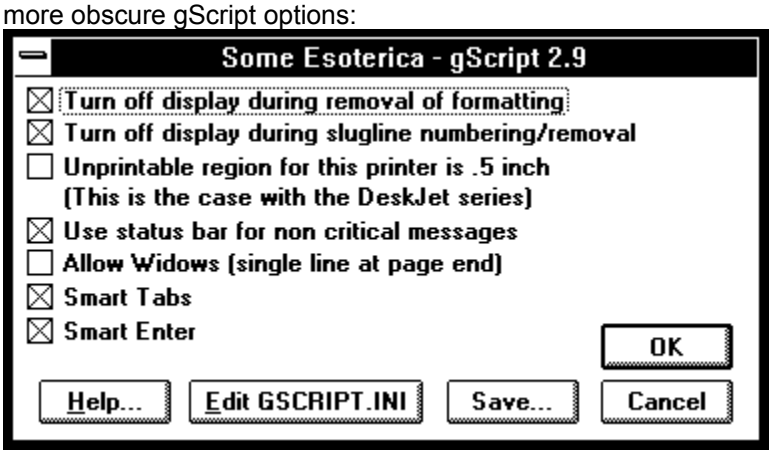
```
CONTINUED:
```

### *PostOptions*

This option box controls what will happen when FormatScript is complete.

## *Esoterica*

This button will display a second dialog box containing a series of check boxes that control some of the more obscure gScript options:




**Turn off display during removal of formatting**

Turns off the screen when executing StripFormatting. (This option is only available if you are running Windows 3.1.)

**Turn off display during slugline numbering/removal**

Turns off the screen when executing NumberAllSlugs.

**Unprintable region for this printer is .5 inch**

By default gScript assumes that the unprintable region is .25 inch. This is the case with most laser printers. However, some printers, notably the HP DeskJet, will not print outside of .5 inch from the paper edge.

If you have seen an error message stating that the margins are outside the printable region, check this box and try formatting again before printing.

**Use status bar for non-critical messages**

By default gScript uses message boxes to notify the user of certain things -- like the completion of a format run.

When this box is checked non-critical status messages are displayed at the bottom of the Word application.

**Allow Widows**

This option was added to give the user more flexibility in determining how many lines will appear on a page. When on a paragraph can be broken in such a way as to leave a single line at the bottom of a page.

By default this setting is off. I would, in fact, recommend that you leave it off.

**Smart Tabs**

When checked this enables the use of the Tab and Shift+Tab keys to move from style to style.

See: SmartTab

**Smart Enter**

When checked this enables SmartEnter. When SmartEnter is on, typing a glossary on a line of Char, followed by Enter, will expand the glossary and move to dialogue style with a single keystroke.

See: SmartEnter

## gsAutoNew

Speedkey: None

Control Center: No

This macro prompts you for title, file name, and directory when creating a new project. It is called by the AutoNew macro every time you base a document on GSCRIPT.DOT.

See Creating a New Project.

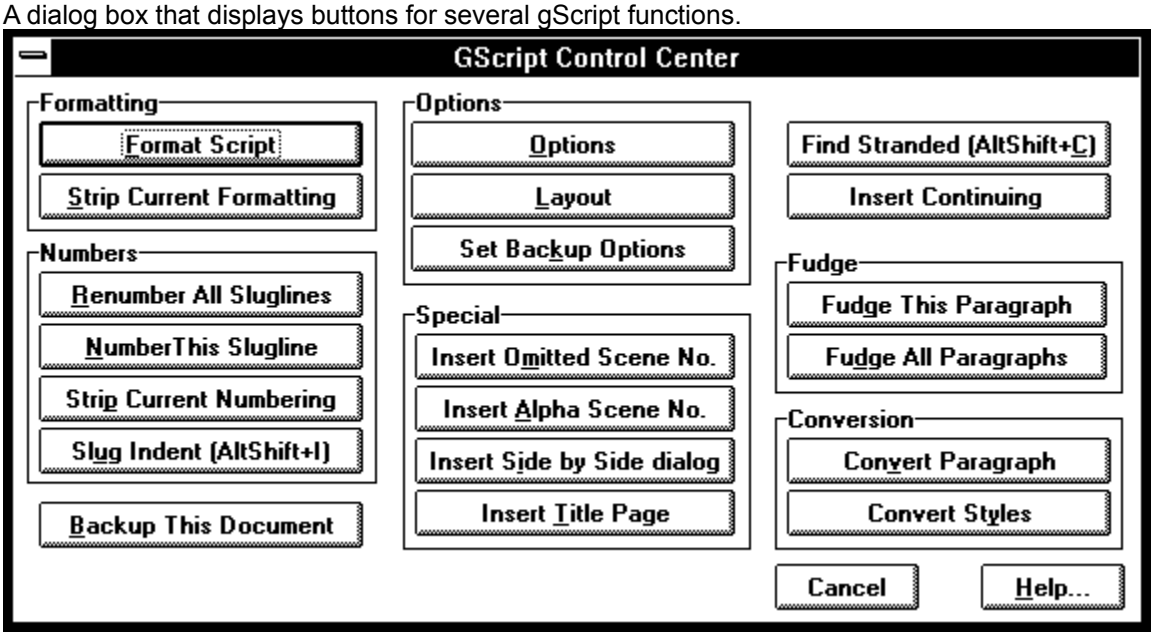
## gsControlCenter

Speedkey:  Ctrl+Shift+C

Menu: Tools

Toolbar : Yes

A dialog box that displays buttons for several gScript functions.




This is an unencrypted macro so you can customize it -- adding or removing or changing the speed keys to suit your preferences.

# gsLib

Speedkey:  None

Control Center:  No

This is a macro library used by many other macros in gScript.

Do not delete.

Executing this macro directly will simply display a message.

## InsertABScene

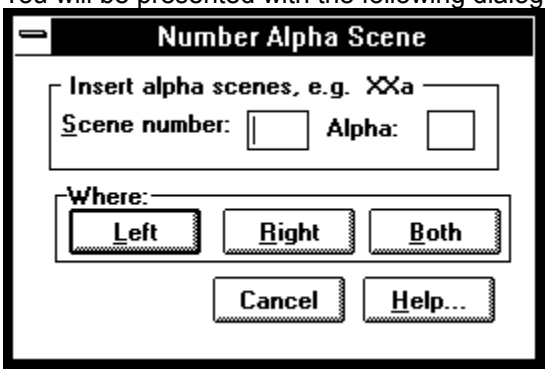
Control Center:  Yes

Menu:  Insert

Allows you to number a scene with an alphabetic modifier:  1a, 1b, 11a.

If you wish to number a scene with an a, b, c mark (rather than renumber the entire screenplay), simply run the macro InsertABScene (found on the Insert Menu).

You will be presented with the following dialog box:




Enter the scene number and which alpha increment to append.

You must currently be located on a heading 3 paragraph (a Slugline).  If you aren't, an error message is printed.

This command is intended for use *after* you have formatted a presentation script and do not wish to renumber the sluglines.  For that reason you might wish to freeze all fields by selecting the entire document and pressing Ctrl+Shift+F9.

## InsertContinuing

Speedkey: None

Control Center:  Yes

Menu:  None

This is a new feature.  A friend of mine submitted a script to a studio.  They had it re-typed.  And when they came upon a section such as

```
        FRANK
   One chimpanzee.

He lifts the knife high above his shoulder.

        FRANK
   Two chimpanzee.  Three...
```

They changed it to

```
        FRANK
   One chimpanzee.

He lifts the knife high above his shoulder.

        FRANK
     (Continuing)
   Two chimpanzee.  Three...
```

This added a total of 138 lines.  So I figured I should give my users a way to anticipate this additional length.  It goes through the entire document and checks to see if a single character's speech is separated by action description.  You will be prompted with a message box whether or not to insert a parenthetical continuing.

At the moment InsertContinuing isn't all that smart.  Some testers thought it overkill.  Other's wanted it to do more.  I'm thinking of allowing it to give you a choice:  add the parenthetical "continuing" or turn the action into a parenthetical itself (and remove the second Char paragraph).  Tell me what you think.

## InsertOmittedScene

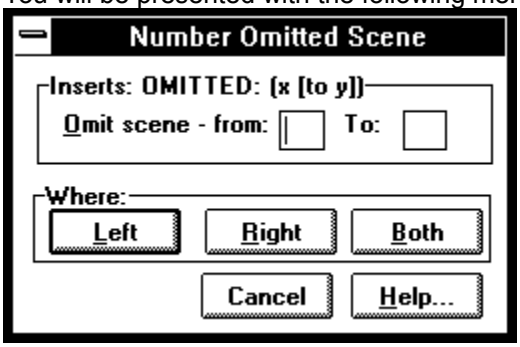
Control Center:  Yes

Menu:  Insert

Allows you to specify that a scene or range of scenes has been omitted.

If you are preparing a shooting script and wish to denote omitted scenes, run the macro InsertOmittedScene (found on the Insert menu).

You will be presented with the following menu:




Know that you must currently be located on a heading 3 paragraph (a Slugline).  If you aren't, an error message is printed.

This command is intended for use *after* you have formatted a presentation script and do not wish to renumber the sluglines.  For that reason you might wish to freeze all fields by selecting the entire document and pressing Ctrl+Shift+F9.

## ListGlossaries

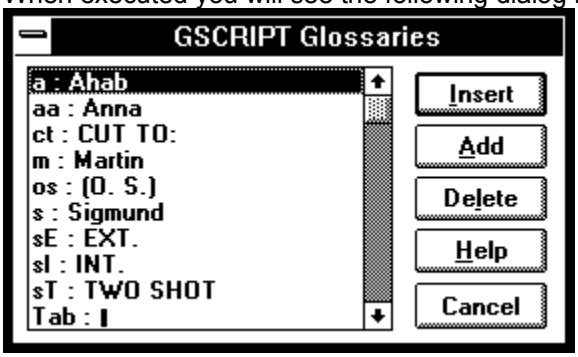
Speedkey: Alt+G

Control Center: No

Toolbar: Yes

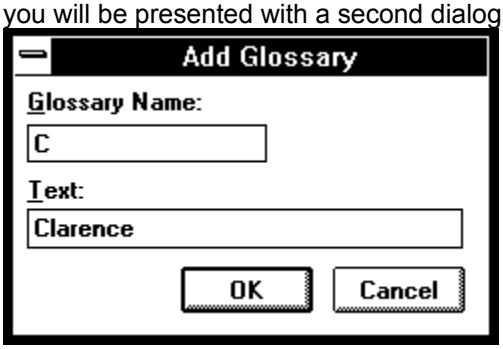
This macro replaces the built-in EditGlossary function. The major difference is that EditGlossary displays *all* glossary names, both global and local, that is glossaries from both NORMAL.DOT and GSCRIPT.DOT. ListGlossaries displays only those glossary entries contained in GSCRIPT.DOT.

When executed you will see the following dialog box:




The other difference concerns how you define a new glossary. With EditGlossary you must first select the text to store in a glossary, then execute EditGlossary (Glossary on the Edit menu or Ctrl+G in gScript) to define the abbreviation.

In ListGlossaries you do both on the same dialog box. When you select Add from the main dialog box you will be presented with a second dialog box:




The limitation is that you cannot include formatting instructions in the text stored in a glossary defined in this way. However, you aren't supposed to have fancy formatting in a screenplay.

If you do need italic or underline in a glossary, use EditGlossary to define.

You can also use the ListGlossaries function to store frequently used film abbreviations such as CUT TO: or INT.

Suggestion: you could segregate all of the "standard" abbreviations by prefacing thier mnemonics with a consistent character such as "s" for standard or "z" (simply becuase that puts all the standard abbreviations at the end of the list. So, CLOSE UP might be stored under sCU.

See Glossaries/Abbreviations for general comments about glossaries.

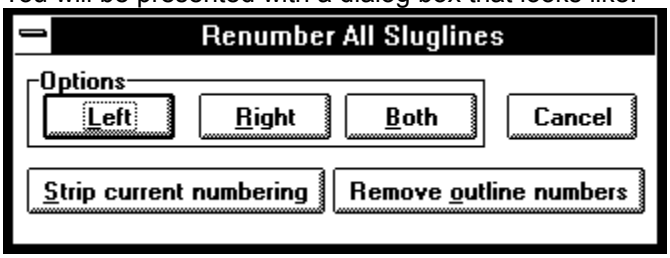
## NumberAllSlugs

Speedkey:  None

Control Center:  Yes

Inserts a {seq slug} field to number all slug lines.  Takes some time. You can turn of the display (speeding things marginally) by checking the appropriate box on the Esoterica sub-menu of gsAdvancedOptions. See gsAdvancedOptions.

You will be presented with a dialog box that looks like:




**Left**:  Places the scene number left of the slugline

**Right**:  Places the scene number right of the slugline

**Both**:  Places the scene number both right and left .

*Note that if you select either Left or Both then the left indent of heading 3 (Slugline) will automatically be changed to a "hanging indent" of -.5" so that the numbers will print in the "gutter" region of the page.*

### Related Topics:

Strip current numbering:
Remove Outline numbering:

### *Strip current numbering:*

This button is necessary only if you wish to remove all numbers before proceeding... It runs an independent macro named StripNumbers.

### *Remove Outline numbering:*

This button is only needed if you have previously numbered your screenplay using the built in outline numbering function of Word for Windows, found on the Utilities Renumber menu. It will first remove all the outline numbering before proceeding. (For a note on why this is necessary, and why gScript does not use the built in outline numbering, see the Technical Notes, "Styles and Line Spacing" on page 12.)

See "NumberThisSlug" on page 25.

## NumberThisSlug

Speed Key:  Alt+N

Control Center:  Yes

Numbers the current slug line.  If you enter this as you work on the script (and then update fields), it is unnecessary to NumberAllSlugs.

NumberThisSlug determines where to place the scene number by reading the Side keyword in GSCRIPT.INI.

## NumTopContinueds

A macro called by FormatScript to place scene length numbers at the top Continued.  Adds several minutes to the time needed to format a script.

This macro should not be called directly.

## ReactivateScreen

Speedkey: Alt+Ctrl+F10

Control Center: No

The ability to turn off the screen display during formatting, numbering scenes, or removing formatting brings with it a new danger. It is possible that for one reason or another an error will be encountered during the execution of these functions and the associated macro will fail before turning the screen display back on. If this should happen you might think your computer has frozen.

Most often it will only effect the current document. That is, you will still be able to move the mouse and select menu items. But PageUp/PageDown, for instance, will not update the screen.

This macro is included as a necessary precaution. If you are executing any command that turns off the screen and the above symptoms appears (usually after a warning beep), simply press Alt+Ctrl+F10 to reactive the disabled screen display.

## RemoveBlankParas

Speedkey - None

Control Center:  No

This macro simple checks to see if there are any blank paragraphs (doubled carriage returns) and if so, removes them.

If you need to run this macro you must select it from the list of template macros displayed when you select Macro from the Tools menu.

To execute this macro you must select it from the list of macros displayed when you choose Macro from the Tools menu.  See Accessing Macros for more details.

## ResetAllDefaults

Control Center:  No

Speedkey: None

This macro can be executed manually.  It resets the current template (whether it is GSCRIPT.DOT or a copy of GSCRIPT.DOT) to the shipping defaults for Menus, Keys, Toolbar, and GSCRIPT.INI.

It will completely remove any customizations you have added yourself and should be used with caution. The purpose is to provide a way to clear out any possible problem assignments and start "from scratch."

To execute this macro you must select it from the list of macros displayed when you choose Macro from the Tools menu.  See Accessing Macros for more details.

## RevisionAccept

Speedkey: Alt+Ctrl+A

This macro marks the currently selected revision as accepted.

There is no default speedkey (however you could assign one).

It is located on the Toolbar as Icon: 

## RevisionSearch

Speedkey: Alt+Ctrl+S

This macro searches for the next revision marked text.

It is located on the Toolbar as Icon: 

## RevisionUndo

Speedkey: Alt+Ctrl+U

This macro undoes the currently selected revision.

It is located on the Toolbar as Icon: 

## RunHelp

Speedkey: Alt+Ctrl+H

Control Center:  Yes

Toolbar:  Yes

Menu: Help

This macro loads the gScript help file.

## SceneReport

Speedkey: None

Control Center:  Yes

Menu: Tools

This macro will generate a new document containing a table of all the scenes in your screenplay.  The table will have the following form:

| No. | SlugLine | Length | Page |
|---|---|---|---|
| 1 | EXT.  DARK AND STORMY NIGHT | 1 | 1 |

The scene report file will be named with the same file name as your screenplay with the extension RPT. It will be saved into the same directory as your screenplay.

If you want to change the layout, fonts, orientation of the RPT file, simply load the document and manipulate it using Word's built-in functionality.  That is to say:  there are no options in gScript to change the way the table looks or special macros to modify the report.  But there is no need.  This is, after all, a word processor...

## SetBackupOptions

Speedkey: None

Control Center:  Yes

This macro displays the following dialog box:


This allows you to change the default settings for BackupThisDoc.

**Backup Destination**

Usually this would point to a floppy drive containing your backup diskette.

If you are changing the destination to a floppy drive, you must insert a diskette into the drive before you click OK or press Enter.

**Backup file at closing**

If  you check this option then every time you close a document based on GSCRIPT.DOT you will be prompted whether or not to backup to the destination path.

**Warn If Overwriting**

If you check this option you will be warned if a file of the same name already exists in the destination path.  By default this is un-checked.

## SideBySideDialogue

Speed Key:  none.

Menu: Insert

Inserts a two column table for side by side dialogue.

gScript contains macros and styles for easily inserting side by side dialogue.  The command is located on the Insert menu;  the styles associated with this command are:

| Style Name | Function | Speedkey |
|---|---|---|
| SbSChar | Side-by-side Character | (no key) |
| SbSDialogue | Side-by-side Dialogue | (no key) |

The command that inserts a table of two columns is located on the Insert menu.

Note:  Once you are in a table it's sometimes confusing how to get out.  Pressing Tab will simply insert another row.  You can either press the Down arrow key or press Alt+NumPad5 to select the table and then press the Right arrow key.

## SmartEnter

Speedkey:  Enter

Turned on and off in the Estoterica sub menu of gsAdvancedOptions:  See Esoterica.

This facility allows you to easily expand glossary entries.  Essentially the macro does the following:

If the current style is Char (for Character Name) then before inserting a new paragraph SmartEnter checks to see if the line contains a glossary abbreviation.  If it does, it expands the glossary.

If the line of Char contains only a period, SmartEnter runs ListGlossaries.

SmartEnter has no effect if the style is anything other than Char.

For example:

Create a glossary with abbreviation C for Carolyn.

1) On a blank paragraph type Alt+C for Char
2) Type C
3) Press Enter.
4) Type random dialogue
5) Press Enter again to move to the next Character name
6) Type a period
7) Press Enter

# SmartTab

Speedkey: Tab

Turned on and off in the Estoterica sub menu of gsAdvancedOptions: See Esoterica.

Allows you to use the tab key to move from style to style. What it does depends upon where the insertion point is located.

If you are in a non-blank paragraph when you press the Tab key, SmartTab will move to the end of the current line and insert a new paragraph before moving to the next style.

If the current line is empty, SmartTab will simply alter its format according to the following logic:

| Current Style | A tab will change to: |
|---|---|
| heading 1 (Slugline) | Normal |
| Action | Char |
| Char | Dialogue |
| Dialogue | Transition |
| Transition | heading 1 (Slugline) |

The tab key will act normally in headers and footers, footnotes and annotations, and in tables (so it works in paragraphs of side-by-side dialogue to move from one column to the next).

However, in all other styles the tab key is disabled. This should not be a problem when using gScript since the only tabs normally used in gScript are inserted automatically by NumberThisSlug or NumberAllSlugs when inserting a scene heading number.

The only one of the six standard "element" styles that is not included in the SmartTab funciton is Paren. This is because Paren is not as frequently entered as the other styles and the ApplyParen macro runs a dialog box (which makes the reversing with SmartTabShift rather awkward). In this version, until I get some feedback on SmartTabs, gScript will leave ApplyParen out of SmartTabs.

## Related Topic:

SmartTabShift

### *SmartTabShift*

This is a companion macro to SmartTab. SmartTab moves from style to style from left to right. That is, it moves through the list of styles in the logical order in which you would use them. SmartShiftTab, assigned to the Shift+Tab key when you turn SmartTabs on, moves from right to left.

SmartTabShift only works on blank paragraphs. It is, basically, a way to return to the previous style if you move too far using SmartTab.

| **Current Style** | **Shift+Tab will change to:** |
|---|---|
| heading 1 (Slugline) | Transition |
| Action | heading 3 |
| Char | Action |
| Dialogue | Char |
| Break | Dialogue |

## SpaceBeforeDecrease

Speedkey: Alt+Shift+-

Control Center: Yes

Decreases the space before a paragraph by 1 point each time it is executed.

## SpaceBeforeIncrease

Speedkey: Alt+Shift+=

Control Center:  Yes

Increases the space before a paragraph by 1 point each time it is executed.

## StripFormatting

Speedkey:  None

Control Center:  Yes

Menu: Tools

This is one of the great virtues of gScript.  It is a simple matter to remove all formatting instructions, page breaks, and additional strings inserted by running FormatScript in Automatic or Interactive mode.

After running StripFormatting you are left with a version of the document that is suitable for draft printing. There will be no inserted "Continueds" or "Mores", etc.

This is, I think, very useful, since you may wish to print "draft" copies of the screenplay as you work, and only FormatScript when a draft is completed.

## StripNumbers

Control Center:  Yes

If you entered slugline numbering as you wrote the script, then this is unnecessary.  If however, the script is not numbered, or not completely numbered (you just added a scene), then you should either run this macro from the Control Center or check the appropriate check box when you format the script.

## ToggleHidden

Speed Key:  Ctrl+Shift+H

Toggle between displaying and hiding text formatted as Hidden.

Since, by default, gScript uses hidden to format Outline Levels 1 and 2, this is a useful toggle to have installed.

Note:  There are two ways to display hidden text:

ToggleHidden -- changes the status ONLY of hidden text.

ShowAll -- this is a built in Word command, accessed by pressing Ctrl+Shift+8 (the thinking here is Shift+8 is the asterisk).

If ShowAll is active then toggling Hidden off will have no effect.  So:  use one or the other method.  Not both.

## ToggleOutline

Speed Key:  Ctrl+Shift+O

Toolbar:  Yes.

A small utility macro that will toggle in and out of Outline View.

## ToggleSlugIndent

Toggles the left indent of sluglines (heading 3) 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 numbers, it should be 0". This macro toggles between the two.

Speed Key: Alt+Ctrl+I

# 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, ActDivsion, 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

## Related Topic:

What's different in 2.9?

## 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

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

## Related Topics:

Title Page
Margins and Paragraph Indents
Header

## 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<br>Ctrl+Shift+<br>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 | |
| ***BackUpThisDoc*** | Backup the current document | Control Center |
| ***ChangeLayout*** | 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<br>Tools Menu |
| ***CheckForStranded*** | Moves through the formatted document | Alt+Ctrl+C |

| | | |
|---|---|---|
| | looking for stranded lines. | |
| ConvertStyles | Converts style names to those expected by the FormatScript. This is particularly useful when you are converting a script already written in Word for Windows or Word for DOS. | Control Center |
| 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 |
| ***EchoOnOff*** | 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<br>Tools Menu |
| ***FudgeAllParas*** | Reformats paragraphs that end with a line that consists of a single line | |
| ***FudgeParaNarrower*** | Decreases the width of the current paragraph by .1" with each execution. | Alt+- |
| ***FudgeParaWider*** | 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 | Ctrl+Shift+C |

| | | |
|---|---|---|
| | displays buttons for several gScript functions.<br>Unencypted | Toolbar<br>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 |
| ***InsertContinuing*** | 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 |
| ***ListGlossaries*** | Shows just the gScript glossaries; allows you to add and delete glossaries | Alt+G<br>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 | |
| ***ReActivateScreen*** | 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<br>Unencrypted | |
| RevisionAccept | Accepts the currently selected revised text | Alt+Ctrl+A |
| RevisionSearch | Searches for the next | Alt+Ctrl+S |

| | | |
|---|---|---|
| | instance of revised text | |
| RevisionUndo | Undoes the currently selected revised text | Alt+Ctrl+U |
| RunHelp | Launches WINHELP.EXE with GSCRIPT.HLP | Alt+Ctrl+H<br>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<br>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 | |
| ***SmartTabShift*** | Uses the Shift+Tab to move from one style to another | |
| | Unencrypted | |
| ***SpaceBeforeDecrease*** | Decreases the space beforea paragraph by 1 point with each execution | Alt+Shift+- |
| | Unencrypted | |
| ***SpaceBeforeIncrease*** | Increases the space before by 1 point with each execution | Alt+Shift++ |
| | Unencrypted | |
| 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 | Alt+Shift+I |

will print in the gutter. If you have sluglines numbered, then the indent should be -.5; if you have no 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 continueds. |

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 continueds.

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.

## Related Topics:


Finding stranded continueds
Making paragraphs wider
Decreasing Space Before a paragraph

## 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 Al+-

For instance, if a paragraph ends with a single word of three characters, pressing Alt+ twice should wraps 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 Al+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 thefollowing 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 |
| Registered=Not | Registration notice |
| 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

gScript is being distributed as ShareWare via electronic bulletin boards and on line services.

gScript 2.9 costs $75.00.  Upgrades from prior versions cost $20.00.

Any "slipstream" updates, will be free.  That is, until a major upgrade (not expected until version 3.0 of Word for Windows), all interim releases will be free.  There may, however, be a 2.9a to fix any (banish the thought) bugs or oversights.

Once you register I will send you a key to disable the single message box reminder which displays when you execute FormatScript.

# Support

There is no direct telephone support.

I will answer all on-line correspondence at the electronic addresses given below. If you do not have a modem or access to these electronic addresses, then write to the postal address given below, but be forewarned: you will get a much faster response over the wires than over land.

# Updates

Updates will be handled in the same manner: that is, they will be distributed via modem to these electronic services. Once you are registered for gScript, you are registered. Simply acquire the most current version from wherever you acquired the first version.

Notices will be mailed informing you of any maintenance releases or upgrades.

# Disks

It is possible to order gScript on disk when registering.

The shipping charge per disk is $10.00 (domestic), $15.00 (international).

The version of gScirpt distributed by disk is identical to that available via modem on CIS or BIX or public bulletin boards.  There is no added functionality.

It does, however, include a Windows setup application to copy the gScript files to the appropriate directory and to optionally create Program Manager icons for the documentation and demonstration documents.

Please be sure to specify disk size preference.

# 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.

# 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 released as ShareWare. You may evaluate it for a period of 30 days. At that point a registration fee of $75.00 must be paid if you continue to use this software. Consider, please, that the commercially available alternatives cost approximately four times as much.

Nothing in this version of gScript is disabled. There is a single reminder screen, which appears when you run the main formatting macro and when you create a new project. When you register, I will send you a key to disable this message.

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
```

Distinquished 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, ususally 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.