Splat!

Version 1.0

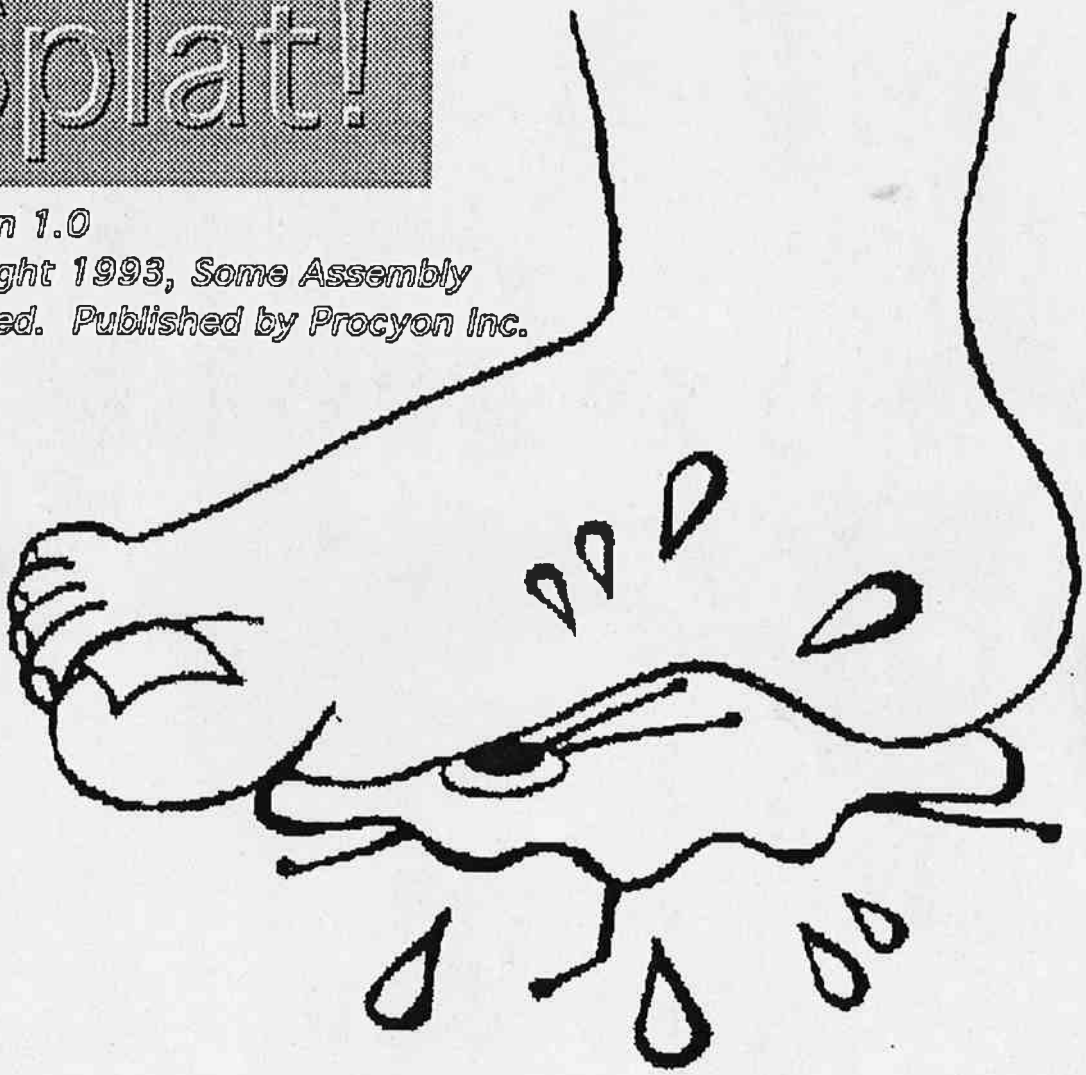

TABLE OF CONTENTS

Chapter 1: INTRODUCTION

While working on another program I was writing in ORCA/C for the IIgs, I was finding it increasingly difficult to debug this large C program with a machine-level debugger such as **GSBug**. I really needed a source-level debugger, but the only one available, Prizm, was, at the time, terribly buggy and not very good. So, **Splat!** was created to answer this need for a good source-level debugger for the IIgs. Originally, I was just going to whip together something simple for myself, but when Mike Westerfield at The ByteWorks showed interest in the program (though he eventually went with his own), I decided to go all out and make something truly impressive. This is when the text-based desktop-like interface for the program was conceived.

The complications of coordinating a desktop-based debugger working with another desktop program at the same time led me to decide that a text-based debugger would be easier to write. Also, I thought this would allow me to debug text-based programs more easily. However, the final product (or rather the one you have in your hands, as I don't think this program will ever really be *done*), is far from being a simple program. There are over 30,000 lines of code here, *all* of which is assembly code and macros. Actually, quite an extensive macro system was created (which help me greatly reduce the number of lines of code) to extend the assembler and fill in some of the important features missing in the ORCA/M assembler.

Splat! works with any language that supports the ORCA debugging information format, such as ORCA/C or ORCA/Pascal. The ORCA/M assembler, does not support source-level debugging, and **Splat!** does not (yet) handle machine-level debugging. However, it can be used in conjunction with Apple Computer's machine-level debugger, GSBug, to provide you with the best of both worlds - source-level and machine-level control.

Splat! requires roughly 128K or more memory to run comfortably, depending on the size of the program you are debugging, above what you would normally need. Also, you should be running System Software 5.0.4 or later (System Software 6.0 or later for use with GNO/ME).

This manual assumes that the reader has at least a good working knowledge of how to use their IIgs system, and that you know, or at least have a good book on, the programming language you are using. This manual does not make any attempt to teach you a programming language, nor does it discuss how to use your chosen operating shell, apart from describing how to invoke the debugger and how to prepare your programs for use with the debugger. Any technical details given are provided only for those who need them or just want to know. If you do not understand these details, or do not expect you will need to know about them, feel free to ignore them.

Also, those of you who are familiar with the graphical "desktop" interface used on the IIgs and Macintosh should be able to get into the program very quickly, and can probably just skim over a good deal of the manual. **Splat!**, though it runs on the text screen, simulates the desktop interface using the special MouseText characters available on the IIgs. With the exception of a menu bar, and the lack of mouse control, this interface is virtually identical to that available through the standard Toolbox functions. Additionally, all of the program's functions are accessible from the keyboard for those of you who prefer this method of input. At this time, this is the only form of input available, though I hope to add mouse control as well in the future.

There are a great many features this debugger *could* have had in it and many features that I would like to have included, but priorities had to be set and some features had to be set aside for possible inclusion in future versions. As I have said, this program will likely continue to grow over time, and I am always interested in hearing users' comments, criticisms and suggestions to help me to improve **Splat!** and to help me to decide which new features should be added first.

Included on your program disk are several addendum files that include up-to-date information about the debugger. The "Release.Notes" file contains information on new features added since this manual was printed, as well as anything additional that we forgot to put into this manual in time. This file will also alert you to known problems with the supported compilers and environments. Please read this over before you install or run the program. The "Bugs" file lists a history of all of the known bugs and tells you when they were fixed or if they are still outstanding. Finally, the "Future.Thoughts" file lists some of the possible features that might be added to the debugger in the future. In your feedback, you can put in your votes for the things you would most like to see, or add new ideas to the list if what you want is not listed there.

You can contact me at the address and phone number listed below, or via electronic-mail from the Internet, America Online, or AppleLink at the address also listed below. If this fails, you can also get a message to me through Procyon, whose address is also given below, following mine.

Finally, I'd like to thank Jawaid Bazyar for helping me get this program to you and for providing lots of helpful information along the way. Also, David Empson for directing me on how to use SANE, and Mike Westerfield for getting me started on the project. Finally, a *big* thanks to all of the beta testers for this project, who helped me track down many errors quickly and who provided me with plenty of good ideas on improving the program. Thanks for your time and patience.

Splat! was written by:
Michael Hackett
Some Assembly Required
102-234 Oakdale Avenue
St. Catharines, ON L2P 2K6
Canada
(905) 687-1657

email:
Internet, America Online,
AppleLink:
`Mike.Hackett@uwo.ca`

and is published by:

Procyon Enterprises, Inc.
PO Box 620334
Littleton, CO 80162-0334 USA

(303) 781-3273

email:
Internet: `bazyar@cs.uiuc.edu`
America Online: `GNO Jawaid`
GEnie: `Procyon.Inc`
Delphi: `JawaidB`

Manual written by Mike Hackett, edited by Jawaid Bazyar
Cover art of one dead bug and program icons by Lynn Hackett

Chapter 2: INSTALLATION

There are two complete versions of the debugger included in this package: a shell version and an INIT version. Both are functionally identical but each version has its advantages and disadvantages. To avoid placing any unnecessary limitations on the use of the debugger, I created both. The INIT version will likely be the more popular of the two, but the shell version is there if your situation warrants it.

The main advantage of using the INIT version is that once it is placed in the "System.Setup" folder of your boot disk or partition, it is always available no matter where you are. The shell version, as the name suggests, can only be invoked from within the ORCA, APW, or GNO shells (or some other compatible shell). Since most program development will normally take place within one of these shells, this is not often such a great inconvenience, but using the INIT version will prevent you from making the very common error of executing a program that has been compiled with debugging codes without one of the versions of the debugger running. Doing so will invariably cause the program to crash and force you to reboot the machine. In addition, with the INIT version, you can debug other INITs (both permanent and temporary), as well as CDAs, NDAs, and Control Panels (CDevs).

On the other hand, if you are somewhat cramped for memory or use the debugger only occasionally, using the shell version will leave some extra memory free that would otherwise be permanently occupied by the debugger code, even when dormant (The debugger is about 80K in size, plus 1.5K stack space). The shell version still requires the same amount of memory be available, but will only need it when you are actually using the debugger. You will just need to be careful that you do not forget to execute all programs containing debugging information explicitly through the debugger shell command, to avoid crashing the computer. Note that programs without debugging information can be run through the debugger command with no ill effects (other than that there will be less memory available), though they will not cause the debugger to become active. If the INIT version is already installed when the shell version is run, the shell version will abort and report the error. You can then reexecute the program you wish to debug *without* the debugger shell command.

Now to the details of installation. There are Installer scripts included for those of you who prefer to use them, but we will also give you details here on how to install the necessary files by hand. This may be a handy reference even to those who use the Installer scripts, just to let you know what the Installer actually did.

Firstly, as always, you should make a backup copy of the program disk, even if you are just going to be installing the program onto a hard drive and then putting the floppy away. There are so many ways for accidents to happen, so it is better to take this extra step and be safe than to damage your original program disk. Then, store the original disk in a different location from the backup disk, to protect it from stray magnets, coffee/pop spills, and other such hazards.

If you want to use the Installer scripts, insert your backup of the Splat! program disk and launch the Installer program. Click on the "Customize" button at the opening window. You will then be presented with a list of installation options. Which ones you will select depends on which version you wish to install and which shell environment you are using, if any. The chart below lists which script or scripts you need to use for your particular setup.

INIT version with no shell:
    Splat INIT

INIT version with ORCA shell:
    Splat INIT
    Splat INIT utilities for ORCA

INIT version with GNO/ME shell:
    Splat INIT
    Splat INIT utilities for GNO

Shell version with ORCA shell:
    Splat shell version for ORCA

Shell version with GNO/ME shell:
    Splat shell version for GNO

The destination for installation will vary depending on the script, so you should install only one script at a time, just to be safe. Each script tells you what the destination should be in its "Help" window - select the script and click on the "Help" button to view this.

The "Splat INIT" script should be run with your startup volume as the destination, while all of the ORCA-specific scripts should be run with your ORCA Utilities directory as the destination, and the destination directory for the GNO-specific scripts should be any directory that is in your GNO/ME search path, most likely the one in which other shell utilities are kept.

With some of the shell-specific scripts, there may be additional work that you will have to perform yourself, as the Installer program is fairly limited in what it can do. For example, it does not have the capability to modify an existing file, which is what is often needed here. When this is necessary, the script will always display its "Help" window whenever you try to "Install" it, where the additional instructions will be given. Write them down so that you will remember exactly what to do when you quit the Installer. Please make sure that you complete these instructions, as the debugger or some support utility may fail to function correctly if you do not.

You can, of course, install more than one version and choosing both the ORCA and GNO scripts for either version, even if you install them in the same directory, will not have any harmful effects. In fact the scripts for the two are very similar, but have been set up in this manner to make it easier for the user. For details on what each script actually does, continue reading the sections below. If you don't really care, at least at the moment, and you have followed all of the instructions in the Installer help windows for the scripts you have installed, then you can skip the remainder of this section and begin using the program.

If you prefer to do things by hand, or just want to know what is happening behind the scenes, you can read one or both of the following subsections for the two forms of the debugger. If you have problems after the installation, the following information may also help you to diagnose the problem and to find a cure.

INIT Version:

To install the INIT version, first copy the "**Splat.INIT**" file from the (backup) program disk into the "System.Setup" folder inside the "System" folder on your boot disk or partition. The only consideration you should make here is that the debugger must be loaded before any INITs that you might want to debug with it. So, it is probably safest to make sure it is one of the first files in the "System.Setup" folder, *after* "Tool.Setup". It is very important that "Tool.Setup" be the first INIT loaded during bootup. If you are also using GSBug in conjunction with Splat!, you can install them in any relative order, but again with GSBug, it should be one of the first INITs loaded. If you are unsure how to rearrange the order of the files in a subdirectory (folder), one way is to move all of the files out to another area and then move them back in one-by-one. Note, that using the Finder to move the files to the desktop and then back will *not* work! You must move the files to another folder or disk to remove the file's entry from the original directory.

If at any time you wish to temporarily disable the loading of Splat!, you can use the Finder (or some other file utility) to check the Inactive box (in the "Icon Info" dialog) for the "Splat.INIT" file. To remove the debugger permanently, simply delete the "Splat.INIT" file, or use the "Remove" operation in the Installer. If Splat! is still in memory at this point, you should restart (reboot) your machine as soon as possible to completely purge the program from your machine.

Additionally, there is a small shell utility and a CDA (Classic Desk Accessory) that allow you to enable and disable the debugger during a session. The shell utility, "**DebugSet**", must be placed in your ORCA/APW Utilities directory, or anywhere in your search path if you are using GNO/ME. Also, if you are using ORCA or APW, you must add the following line to your "SYSCMND" file (see your ORCA or APW manual for information about this file).

```
DEBUGSET          *U
```

Important: This line must be added whether or not you have used the Installer scripts! There is currently no way for an Installer script to modify an existing file, and since each user may have configured this file differently, we cannot simply replace the existing file.

The syntax of this command is:

```
debugset [on|off]
```

That is, to turn off the debugger, you issue the command "`debugset on`"; to turn it off, use "`debugset off`". Simply issuing "`debugset`" with no parameters will display the current setting of the debugger flag. If the INIT version is not installed, the program will simply report that and exit without any further action. Provided your shell allows aliases to be set up (and all of the ORCA, APW, and GNO/ME shells do), you can easily create aliases for turning the debugger on and off, if you like.

To use the CDA, copy the file "DebugSet.CDA" to the "Desk.Accs" folder inside the "System" folder on your boot disk or partition. It will show up as "Splat! DebugSet" in the CDA menu. (To access the CDA menu, press Apple-Control-Esc at any time - it will immediately appear unless an important operation is in progress, in which case, the menu will appear when the operation is finished.) To open the CDA, move the cursor bar to "Splat! DebugSet" with the up- and down-arrow keys, and then hit <Return>. A message will appear on the screen informing you of the current setting of the debugger flag, and then it will ask you if you wish to change the setting. Hit <Return> to accept the change, or press <Esc> to cancel the operation. If the Splat! INIT is not installed at the time you open the CDA,

the situation will be reported and you will be asked to press any key to return to the menu.

When Splat! is turned off, programs with debugging codes will not cause Splat! to become "active" - that is, the full debugging environment will not be started up, saving a great deal of overhead and allowing your program to execute much faster; faster than even the debugger's fastest execution mode. Splat! is still interpreting the debugging codes, but just skips over the information, allowing your programs to execute normally, whether or not they have been compiled with debugger information included.

Finally, there is one additional file, "**FixSplat**", that GNO/ME v1.0 users need in order to make the debugger operate correctly under this environment. This small program reconnects the debugger to the necessary system vectors after the GNO/ME kernel has disconnected it during startup. Copy this file into your utilities directory with "**DebugSet**", or into some other directory in your search path. If you are using GNO/ME v2.0 or later, you will not need this program.

You will also likely want to place an additional line that executes "FixSplat" somewhere in your "gshrc" file so that the debugger is reconnected automatically every time you start GNO/ME. FixSplat works fine even when the INIT version has not been loaded. It first checks to see if the Splat! INIT has actually been loaded before taking any actions, and does nothing if it does not find the debugger. When it is run, it normally prints a one line message indicating its action, however this can be disabled by using "FixSplat -s" if desired. "FixSplat" need only be run once after each time GNO/ME has started up, and should never be run from outside GNO/ME.

(For those of you interested in the technical details, the debugger INIT posts a MessageCenter message for use by these utilities. This is what FixSplat and the other utilities check for and use. The format of this message is not fixed and may change in future versions so do not count on its format. This message is for our use only!)

Remember, you have to reboot the system before any changes to files in the system folder will take proper effect. In this case, this means that installing the debugger INIT file does not cause the debugger to be loaded immediately. It will be loaded only during GS/OS startup. Similarly, deleting the INIT does not remove the debugger from memory until a reboot. You *can* use an Apple utility called "IR" (for "Init Restarter"), which is a Finder Extension that allows you to load and remove INITs and DAs without resetting the computer, to install the debugger from the Finder, however, it is not safe to remove Splat! with this program, and Splat! will report an error message if you attempt to load it when it is already installed.

Shell Version:

Installing the shell version of Splat! requires only slightly more effort. First you must copy the "**Splat**" program file from the program disk to your shell's utilities folder. Where this is depends on which shell you are using and how you have your setup configured.

If you are using the ORCA or APW shells, this would normally be the "Utilities" subdirectory in the same directory as the shell program itself. However, you can change the location at which the shell looks for utilities by changing the utilities prefix (17 for ORCA 2.0 and greater; 6 for APW and earlier ORCA shell versions). Also, you must add the following new line to your "SYSCMND" file (see your shell reference manual for more information about this file):

```
SPLAT              *U
```

If you are using GNO/ME, you can place the program file anywhere in your search path. However, if you have both ORCA and GNO/ME installed on the same volume and your GNO/ME search path includes the ORCA Utilities directory, then this directory would be the most logical choice for the Splat file. For more information on search paths, see the GNO/ME Shell Reference.