MPW QR4 Tools/Scripts
Release Notes

Tools

Several new tools appear in this release.

- **StreamEdit** is a scriptable text editor similar to the editor *sed* that is found in UNIX®.
- **CMarker** generates markers at function definitions in C++/ANSI C source files.
- **Get**, developed for “411” Help, is available for general use. See the MPW QR4 “411” Help Release Notes for information about “411”.

A number of improvements have been made to existing tools.

- **Choose** has two new options for prompting for passwords.
- **Compare** has had its line number limitation raised from 9999 to 65535.
- **DumpObj** has a slightly changed **-m** option, and new options **-mods** and **-co**.
- **FileDiv** has been enhanced to give the user the option of considering its input file as a stream of bytes.
- The option **-sort** has been added to **GetListItem**.
- The performance of **Search** has improved as the result of search algorithm and buffering changes. Four new options have been added.
- **Link** has a new data initialization routine to reduce the size of the compressed data image in the link output. Appropriate changes have been made to DumpCode for conformity with this change in Link. Options have been added to Link, one to accommodate a new run-time architecture (“32-Bit Everything”), the other to cause automatic generation of branch islands.
- **Link** and **Lib,** upon failure, will set that date of the output file to numeric zero (1 January 1904).

- **Make** has several new features. The default rule mechanism has been extended so that it may identify optional additional dependencies in addition to the single dependency permitted by the current mechanism. A default build rule has been added for C++. A new predefined variable, {Deps} stands for all of the dependencies of a target. A new predefined target, $OutOf Date can be used to force other targets to be rebuilt. A new option, **-y**, provides a limited form of the verbose option output.
- **ProcNames** has been enhanced to generate MPW Shell "mark" commands that set markers on all procedures and functions in an Object Pascal file. It has also been enhanced to process conditional compilation directives.
- **PasRef** has been enhanced to process conditional compilation directives. The maximum number of symbols it can handle has been raised from 5000 to 6000.
- **ResEqual** now looks at resource attribute flags.

△ A bug has been reported in **DumpCode.** Information will not be dumped correctly from a module that has been compiled and linked as "model far" and that contains any A5-relocatable information.

A partial work-around for this is to use the "-ri" option, which inhibits the dumping of relocation information. This will, at least, permit correct dumping of code. △

Backup

A bug that has now been fixed prevented the creation of inner folders when the when the `-t` option was used to limit recursive processing (`-r`) to files of a specific type.

Choose

The following options, that prompt for "secure" passwords with a dialog box, have been added:

```
-askpw #prompt for the server password.  Illegal in conjunction
    with -pw or -guest.

-askvp  # prompt for the volume password.  Illegal in conjunction
    with -vp or -guest.
```

CMarker

CMarker reads the specified C++ / ANSI C source file(s), syntax checks them and generates appropriate "Open" and "Mark" MPW commands, which, when executed, will mark the source file(s) at each function definition with the marker name being the name of the function. Its purpose is to aid in the marking of source files for use with the MPW "marker browser " capability. CMarker contains a full ANSI C preprocessor and provides options to mark include files, generate source listings (with or without showing macro expansions), run the preprocessor only, flag anachronisms, and syntax check C++ / ANSI C with or without Apple extensions.

(See Appendix B for the CMarker manual.)

Commando

The following bug has been fixed: conditionals in Commando resources were not always evaluated correctly when a set of dependencies was not a simple tree but a directed acyclic graph. An illustration of this would be two entries, B and C, both dependent on A, with D dependent on the expression (B or C).

Compare

The maximum number of lines in the files being compared has been raised. The limitation section of the manual should be amended to read: “The maximum number of lines in the text files read by Compare should be less than 65535.”

DumpCode

DumpCode now stops disassembling at the end of the resource (rather than skidding to a halt a few bytes beyond). It recognizes and dumps the new data initialization format used by Link.

DumpObj

The `-m` option has been modified such that the argument can now be either a module name or an entry point name. If it is the latter, the meaning is to dump the module that contains the named entry point. Any number of instances this option may now appear in a command line.

The `-mods` option prints a summary of the contents of an object file, including the name, size, scope and segment of each module and entry point. This means you can find out what's in an object file without dumping the entire file.

The `-co` option causes the generation of disassembled code only.

More information is being dumped (flags are printed in English, logical addresses are disassembled, and so forth).

Type data fixups are now printed correctly.

FileDiv

FileDiv now allows an input file to be viewed as containing an arbitrary byte stream in its data fork.

(See Appendix F for changes to the FileDiv manual.)

Get

Get is a tool for retrieving information from a data base indexed by a BTree. It will not only retrieve information, but will also create and update the index file when required.

Get is heavily oriented to the needs of the menu-driven “411”. Direct calls to it could be for the purpose of retrieving information from the “411” database in a different manner from that provided in “411”, or for accessing a database that is totally independent of “411”.

Information on how to construct database files can be found in the section entitled ‘Adding your own help to “411”’ in the MPW QR4 “411” Help Release Notes.

(See Appendix C for the Get manual.)

GetListItem

The option **-sort** has been added. Use of this option will cause the list in the dialogue box to appear in sorted order.

Lib

Lib now supports the `-mf` option to allocate MultiFinder temporary memory when the MPW Shell heap runs low. See the documentation (and warnings) about `-mf` in the Link manual page.

Link

Data Initialization

A new data initialization compression routine was added to Link to provide a more compact representation of the data image within link output. The routine originated from the need for better compression of C++ VTables. An associated _DATAINIT module exists within the 3.2 alpha Runtime.o library to expand the data image into the proper below-A5 world. The new routine may reduce the size of the %A5Init segment by up to 33%. Link will revert to using the original data initialization compression routine when older libraries are linked.

New Options

Two new options deal with removal of various 32K size limitations. For technical explanation, see the MPW QR4 Run-Time Architecture Release Notes.

Branch Islands

A new option has been defined to cause the automatic generation of branch islands to remove the 32K limit on segment size. The syntax is:

```
-br on      # generate branch islands where needed
-br off     # do not generate branch islands (default)
```

- The `-br on` option should not be used simultaneously with the `-model far` option.

△ **Important** Because of an outstanding bug, attempting to use the `-br on` option simultaneously with the `-sn` option may cause a crash. △

"32-Bit Everything"

A new option has been defined to accommodate the "32-Bit Everything" method of removing the 32K limit on segment size, jump table size, and the size of the global data area. They are:

```
-model near        # the default
-model far
```

If any of the code being linked was compiled with a "far" option, it is necessary to link with the option `-model far`.

New Keywords (-opt option)

The `-opt` option accepts two additional keywords:

`-opt names`	SelectorProcs are modules generated by the linker to be used at run-time for Object Pascal method dispatching. This keyword causes MacsBug symbols to be appended to SelectorProc modules so that the selector names are visible in MacsBug disassemblies. The MacsBug symbols take up space in the application, both on disk and in memory (e.g. 9K or more for a medium MacApp application).
`-opt info`	Write method table optimization information to diagnostics. This information was previously written to diagnostics using the '-p' option and was moved to the 'Info' subarg to reduce the size of the linker diagnostics when optimizing.

Object Pascal Optimization

Jump table entries for monomorphic methods are now being stripped. This is an intentional step of the optimizer, although it may present problems when trying to call an Object Pascal method from a C++ program. To work around the dispatching problem, use the `NoBypass` subargument to `-opt` to bypass monomorphic method optimization when optimizing.

Symbolics

The format of Sym files has changed to support Object Pascal and FORTRAN. Use the 1.3 SADE with the MPW QR4 Linker (you can't mix old and new Sym files or SADEs — please delete your existing Sym files). Contact DTS for the details of the Sym file and OMF changes.

Miscellaneous

The linker now returns an error when the size of the jump table has reached the maximum number of jump table entries.

Previously, when the `-ad` option was used, the linker would align every data module including the main data module, if one existed. This was unacceptable since MPW Pascal makes A5-relative references to data in the main data module assuming that it is located immediately below A5. The `-ad` option will now correctly align data while leaving the main data module immediately below A5.

There is now no limit to the size of global data that is compressed into the %A5Init data initialization segment. There was previously a limit of 32K.

When linking using the `-mf` option, if memory conditions continue to be tight, the flush command may be used to recover additional space from the shell's cache.

Various cosmetic changes have been made to the `-l` and `-map` output.

Make

The new version of Make for MPW QR4 includes several new features.

The default rule mechanism has been extended so it may identify optional additional dependencies in addition to the single dependency permitted by the current mechanism. This new capability has immediate applications to MacApp builds.

A default build rule has been added for C++ files, which are recognized by the suffix `.cp`.

Make supports a new predefined variable, {Deps}, which stands for all of the dependencies of a target.

Make also supports a new predefined target, $OutOfDate, which can be used to force other targets to be rebuilt.

A new option, **-y**, has been provided. This option tells why build rules were emitted, but in a less verbose form than the existing **-v** option. Namely, it does not emit messages about up-to-date targets.

Default Rules Extension

Default rules formerly had the following form:

```
.extension1 ƒ     .extension2
        <build rules>
```

This syntax has been extended as follows:

```
.extension1 ƒ     .extension2   [other dependencies ...]
        <build rules>
```

The other dependencies can be more file extensions and/or fixed file names. These dependencies are considered secondary or optional dependencies as is explained below.

The first extension on the right hand side of a default rule is treated as the **primary dependency** (or trigger dependency), that is, the file name created with this extension (i.e., "{depdir}{default}.extension2") must be valid in order for the default rule to be triggered or applied. To be *valid* a file name must either appear in the makefile or exist in the file system (or lead to a valid file name by further recursive applications of default rules). The existence or non-existence of files specified by secondary dependencies will have no effect on whether the rule is triggered, thus default rules are triggered just as they were before.

Secondary dependencies are optional; that is, none are required, and the indicated file (or files) need not be valid for the default rule to be applied. Secondary dependencies are only processed when a default rule is triggered by its primary dependency. When the secondary dependency is a fixed file name, a dependency will be added if the secondary dependency refers to a valid file name. Similarly, when the secondary dependency is a file extension (e.g., ".extension3") a dependency will be added if the file name implied by this extension (i.e., "{depdir}{default}.extension3") refers to a valid file name. If a file name specified by a secondary dependency is not valid then no dependency is added and the default rule is processed as usual.

Applications. This extension will be useful for MacApp, where typically a source file consists of a file with the interface and an include with the implementation. Now the dependency of the object file on both source files can be stated in a single rule, such as the one below:

```
.p.o  ƒ     .p    .p.inc1
      <normal Pascal build rule>
```

"Deps" Variable

The predefined "{Deps}" variable may be used in a target's build rules and represents all of the (first-level) dependencies of the target (as opposed to {NewerDeps} which represents only those dependencies which are newer than the target).

The {Deps} variable can be used to write a default rule for links or application builds where all of the dependency files will be linked together, such as:

```
.     ƒ     .o
      {Link}  {LinkOptions}  {Deps}  -o {TargDir}{Default}
```

"$OutOfDate" Target

The predefined "$OutOfDate" target is an artificial target which is always out of date and never needs to be rebuilt. One can force a target to be rebuilt by making it depend on $OutOfDate.

For example:

```
ForceRebuild =          # default variable definition (NOP)
foo          ƒ     {ForceRebuild}
       <build foo>
```

If Make is invoked normally the target "foo" will not be rebuilt; however, it will be rebuilt if invoked with the following command line:

```
Make  -d  ForceRebuild=$OutOfDate
```

This command line has the effect of overriding the vacuous definition of "ForceRebuild" in the makefile while giving foo a dependency on $OutOfDate which forces it to be rebuilt.

PasMat

The following bug, pertaining to the `j` formatting directive, has been fixed:

```
j=<width>[±]/<col1>[sd]/<col2>c
```

did not handle the case where `<col2>` could be 1.

PasRef

Options have been added to permit processing of conditional compilation directives. The limitation on the number of symbols has been raised from 5000 to 6000.

(See Appendix E for changes to the PasRef manual.)

ProcNames

ProcNames can be directed to generate MPW Shell “mark” commands that place markers on all procedures and functions in an Object Pascal file.

Options have been added to permit processing of conditional compilation directives.

(See Appendix D for changes to the ProcNames manual.)

ResEqual

ResEqual now compares resource attribute flags for equality and reports any differences found.

Rez

Avoid splitting expressions with #if or #elif commands, for example:

```
type 'TEST' {
      int;
};

#define big 5

resource 'TEST' (128) {
      10    // this will not work because the expression is split
#if big
      * 45
#endif
};
```

- Attention is directed to the material in the MPW 3.0 Reference Manual with respect to the arguments of the `-c` and `-t` options. These arguments are Rez expressions. If a special character within the expression requires that it be enclosed in single quotes, it is necessary to surround the quoted expression with double quotes. E.g. `-c " '§§§' ".`

Search

The following options have been added:

```
-b    #break File/Line output text into two lines

-nf   # write error message if pattern not found

-ns   # return 0 when pattern not found

-sf   # stop the search at the first successful match
```

Improvements in the search algorithm and in the buffering have greatly enhanced the performance.

Sort

A bug in Sort prevented the sorting of field expressions that include column offsets or counts. For example, a call of Sort with the option

```
-f 1.3+4
```

would fail to do any actual sorting. This bug has been fixed.

StreamEdit

StreamEdit is an non-interactive text editor similar in function to the Unix® tool *sed*. Providing scriptable text matching and editing operations, it is useful for making repetitive changes to files, for extracting information from text files, or as a filter.

StreamEdit takes a script and a set of input files (or standard input, if no input files are specified) and applies each statement in the script to each line of input, writing the output to standard output or the specified output file.

(See Appendix A for the StreamEdit manual.)

Scripts

CompareFiles

The CompareFiles command now has two more options for specifying screen size:

```
-TwoPage    # screen size for Apple Two-Page Monochrome Monitor
-Portrait   # screen size for Apple Macintosh Portrait Display
```

CreateMake

Enhancements to CreateMake are:

- A check box for providing symbolic information to the SADE debugger. This will apply the **-sym on** option to the generated command lines for compilation and linking. The check box is dimmed for desk accessories, because the method for building them produced by CreateMake is note compatible with SADE.
- Radio buttons for the options **-mc68020**, **-mc68881**, and **-elems881**.
- Provision for the building of objects of a new type: SIOW application. This type is described in the MPW QR4 SIOW Release Note.

Other changes are:

- CreateMake now creates correct Makefiles for building cdevs and most stand-alone code resources. Desk accessories remain a problem since there are several different methods for building the header at the start of a desk accessory. CreateMake will produce a correct makefile for one of the methods used in the Memory DA examples found in the CExamples and PExamples folders. This build technique is the same as that found in MPW 3.0, but is not compatible with SADE. For information see the Instructions file in the CExamples or PExamples folder. In conclusion, if you are building a desk accessory, you probably shouldn't be using CreateMake to produce the makefile.
- CreateMake recognizes that the functions of the libraries CRuntime.o and CInterfaces.o have now been absorbed into Runtime.o and Interfaces.o; it therefore no longer puts CRuntime.o and CInterfaces.o into the library list for linking.

- The compilation and linking command lines generated by CreateMake no longer have the **-w** option; warnings will be issued. The only exception is that the link command line for a tool has the **-d** option, thus suppressing the duplicate definition warnings that would normally be issued because of the use of Stubs.o.
- If any source file is written in Object Pascal or C++ (filenames with the extensions `.p` or `.cp`), the line `#"{Libraries}"ObjLib.o` will appear in the link.