T E C H N O T E 1078

Solutions for Finding Missing MPW Libraries and Tools

CONTENTS

The Macintosh programming landscape is sometimes subject to change without notice. Libraries that support various MPW runtime architecture, for example, may inexplicably shift: entry points (functions) or global variables that appeared in one library may have moved to a another without your knowledge. If your application code hasn't changed, or changed all that much, you may be somewhat confused. Typically, you'll have to embark on a search of all libraries not currently included in your link. Another scenario is that you encounter some new header files and can't find the correct libraries to resolve any undefined symbols that you may encounter.

This Note addresses how, using MPW, you can go about resolving undefined symbols found while linking. It also demonstrates the use of several MPW Shell scripts working together to automate this process.

This Note is of general interest to all developers involved in Macintosh programming using C or C++.

Displaying Obsolete Libraries and Tools

Sometimes, when a library has become obsolete, a text file of the same name (called a knockout file) with a special creation date (for ETO #21 that date is 1/1/98) replaces the obsolete library. This often causes an error to be generated at link time. The Link error may state something along the following lines:

```
Can't open object file for input. (Error 32)
HD:Development:Interfaces&Libraries:Libraries:Libraries:Runtime.o is not an object file.
```

Knockout files are also used to replace MPW Tools. The difference with a MPW Tools knockout file is that when you try to run a tool that has been replaced by a knockout file, you will get a message printed to your screen explaining the status of that tool.

If you have your Desktop configured using the by Icon or by Small Icon View items, you'll find that the offending file is actually a text file. Often, though, some other option is selected in the Finder's View menu, thereby obscuring the fact that the offending file is actually a text file.

Using the DisplayObsolete Script

Searching through all of the object files can be quite tedious. I have constructed a script, called '**DisplayObsolete**', that flags all obsolete libraries and MPW tools. Currently, there are six folders in the Interfaces&Libraries:Libraries folder, 'DisplayObsolete' will go through each of those folders and the MPW Tools folder flagging all text files created on 1/1/98 as obsolete. The output is placed in a file called Obsolete.index in the MPW folder. Listing 1 shows the DisplayObsolete script.

Listing 1 - The DisplayObsolete script

```
Echo "The following file(s) are now obsolete." > "{MPW}"Obsolete.temp0;
Echo "*" >> "{MPW}"Obsolete.temp0;
Echo "*" >> "{MPW}"Obsolete.temp0;
Echo "*" >> "{MPW}"Obsolete.temp0;

If `Exists "{MPW}:Interfaces&Libraries:"`
        Set InterfacesandLibrariesLocation "{MPW}:Interfaces&Libraries:"
Else
        Set InterfacesandLibrariesLocation "{MPW}"
End

Set CurrentDirectory `Directory`

For j in `Files -f -d "{InterfacesandLibrariesLocation}Libraries:"`
        Set Count 0
        Set FieldCount 0
        Directory "{j}"
        For i in `Files -n -o -t TEXT -x d "{j}"`
                If {FieldCount} == 0
                        Evaluate Exit = 0
                        Search /obsolete/ "{i}" -q >> Dev:Null
                        If {Status} == 0
                                Echo {i} "is now obsolete." >> "{MPW}"Obsolete.temp0;
                                Evaluate Count += 1
                        End;
                End;
                Evaluate FieldCount += 1
                If {FieldCount} == 4
                        Evaluate FieldCount = 0
                End
        End;
        If {Count} == 0
```

```
                Echo "dnThere are no obsolete files in" "{j}dndn" >> "{MPW}"Obsolete.temp0;
        Else If {Count} == 1
                Echo "dnThere is one obsolete file in " "{j}dndn" >> "{MPW}"Obsolete.temp0;
        Else
                Echo "dnIn" "{j}" "there are" {Count} "obsolete filesdndn" >> "{MPW}"Obsolete.temp0;
        End;
End;

Set Count 0
Set FieldCount 0
Directory "{MPW}Tools:"
For i in `Files -n -o -t TEXT -x d "{MPW}Tools:"`
        If {FieldCount} == 0
                Evaluate Exit = 0
                Search /MPW Archive Read Me/ "{i}" -q >> Dev:Null
                If {Status} == 0
                        Echo {i} "is now obsolete." >> "{MPW}"Obsolete.temp2;
                        Evaluate Count += 1
                Else
                        Search /no longer supported/ "{i}" -q >> Dev:Null
                        If {Status} == 0
                                Echo {i} "is now obsolete." >> "{MPW}"Obsolete.temp2;
                                Evaluate Count += 1
                        Else
                                Search /obsolete/ "{i}" -q >> Dev:Null
                                If {Status} == 0
                                        Echo {i} "is now obsolete." >> "{MPW}"Obsolete.temp2;
                                        Evaluate Count += 1
                                End;
                        End;
                End;
        End;
        Evaluate FieldCount += 1
        If {FieldCount} == 4
                Evaluate FieldCount = 0
        End;
End;

If {Count} == 0
        Echo "dnThere are no obsolete tools in" "{MPW}Tools:dndn" >> "{MPW}"Obsolete.temp2;
Else If {Count} == 1
        Echo "dnThere is one obsolete tool in " "{MPW}Tools:dndn" >> "{MPW}"Obsolete.temp2;
Else
        Echo "dnIn" "{MPW}Tools:" "there are" {Count} "obsolete toolsdndn" >> "{MPW}"Obsolete.temp2;
End;

If {Count} > 0
        Echo "dnThe following tool(s) are now obsolete, but may be found in E.T.O.:Past&Future:Archive" >> "{MPW}"Obsolete.temp1;
        Echo " " >> "{MPW}"Obsolete.temp1;
        Echo "The 'MPW Archive' folder is a repository for MPW tools and related items which are no" >> "{MPW}"Obsolete.temp1;
        Echo "longer supported and whose use is no longer recommended. While the intent is that" >> "{MPW}"Obsolete.temp1;
        Echo "someday these tools will disappear from ETO altogether, it is recognized that some" >> "{MPW}"Obsolete.temp1;
        Echo "developers may need more time before they are ready to transition to newer tools." >> "{MPW}"Obsolete.temp1;
        Echo " " >> "{MPW}"Obsolete.temp1;
```

```
End;

Catenate "{MPW}"Obsolete.temp0 "{MPW}"Obsolete.temp1 "{MPW}"Obsolete.temp2 > "{MPW}"Obsolete.index
Delete -y "{MPW}"Obsolete.temp0 "{MPW}"Obsolete.temp1 "{MPW}"Obsolete.temp2

Echo "dn" >> "{MPW}"Obsolete.index;
Echo "=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=" >> "{MPW}"Obsolete.index;
Echo "=- An obsoleted file may contain information about which object file/tool replaces it. -=" >> "{MPW}"Obsolete.index;
Echo "=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=" >> "{MPW}"Obsolete.index;

Directory "{CurrentDirectory}"
```

Table 1 lists all tools that are obsolete on ETO #21, and the new tools that replace them.

Table 1 - Obsolete Tools

	Obsolete	**New**
Tools	C	SC
	CFront	SCpp
	Pascal	Currently no replacement
	PasMat	Currently no replacement
	PasRef	Currently no replacement
	PPCC	MrC / MrCpp

Table 2 lists all libraries that are obsolete on ETO #21, and the new libraries that replace them.

Table 2 - Obsolete Libraries

	Obsolete	New
Library	Complex.o	Currently no replacement
	Complex881.o	Currently no replacement
	CPlusLib881.o	CPlusLib.o and IOStreams.o
	CPlusOldStreams881.o	CPlusLib.o and IOStreams881.o
	CPlusOStreams.o	CPlusLib.o and IOStreams.o
	CSANELib.o	{Libraries}MathLib.o
	CSANELib881.o	{Libraries}MathLib881.o
	Math.o	{Libraries}MathLib.o
	Math881.o	{Libraries}MathLib881.o
	CPlusLib.o	MrCPlisLib.o and MrCIOStreams.o
	MathLib.xcoff	{SharedLibraries}MathLib.o
	ObjectSupportLib.xcoff	{SharedLibraries}ObjectSupportLib
	StdCLib.xcoff	{SharedLibraries}StdCLib

Search the Static QuickView Database

A new feature in the Toolbox Assistant, implemented in ETO #21, is a database of all the symbols in all the various libraries. The name of this database file is 'LibFuncRef.qv'. To access the LibFuncRef.qv database, you perform the following:

1. Select the Home Page item from the MPW Info menu.
2. Select the Toolbox Assistant from the Home Page.
3. Press the Index... button at the top left of the Toolbox Assistant.
4. Select LibFuncRef.qv from the Which Index pop-up menu.
5. Once in the LibFuncRef.qv database, enter the desired symbol name.

Using QuickView, this database may be used to quickly search for all missing symbols. LibFuncRef.qv presents a static view of all the libraries shipped on a given ETO. If you get new libraries from another source (for example, the Mac OS SDK), you may need to create your own database of functions and symbols.

Using MPW Tools to Search Library Contents

While symbols may move from one library to another, they don't generally disappear. MPW provides several tools to help you search the contents of libraries.

What you need to do is the following:

1. Dump the contents of each library.
2. Search for the symbols you want to find.

Use DumpObj, DumpPEF, and DumpXCOFF to dump the contents of libraries. Table 3 shows you the correct tools to use with each library.

Table 3 - MPW tools for dumping libraries

MPW Tool	Library Suffix	Type
DumpPEF	none	'shib'
DumpObj	.o	'OBJ'
DumpXCOFF	.o or .xcoff	'XCOF'

Once the contents of a library has dumped to a text file, you can use standard MPW searching tools to locate the symbols that you need to find.

Dumping the Contents

There are six scripts to dump the contents of the six folders in the Libraries sub-folder of the Interfaces&Libraries folder, respectively. There is also one script that dumps all the libraries into a single file. The following is the list of scripts to dump the various libraries:

1. DumpAllLibraries
2. DumpCFM68KLibraries
3. DumpCLibraries
4. DumpLibraries
5. DumpPLibraries
6. DumpPPCLibraries
7. DumpSharedLibraries

The sorted contents of each of these six folders in the Interfaces&Libraries:Libraries folder is dumped into a file bearing the name of the library and the suffix .index and placed in the MPW folder. Finally, a single file called AllLibraries.index is created that contains the sorted and merged contents of the previous six .index files. Here is an example of two scripts that are used to dump the contents of libraries:

Listing 2. The DumpCLibraries Script

```
If "`Exists -d "{CLibraries}"`"
        For i in `files -f -t 'OBJ ' "{CLibraries}"`;
                DumpObj -n "{i}" | streamedit -e "1 delete; /(=)(R)1/  replace /[ ]*(=)(R)1/ (R)1'dt{i}'" >> "{MPW}"CLibraries.index
        End;
        sort -l -unique "{MPW}"CLibraries.index -o "{MPW}"CLibraries.index
End;
```

Listing 3. The DumpAllLibraries Script

```
DisplayObsolete;
DumpCFM68KLibraries;
DumpCLibraries;
DumpLibraries;
DumpPLibraries;
DumpPPCLibraries;
DumpSharedLibraries;

sort -l -unique "{MPW}"SharedLibraries.index d
                                "{MPW}"Libraries.index d
                                "{MPW}"CLibraries.index d
                                "{MPW}"PPCLibraries.index d
                                "{MPW}"PLibraries.index d
                                "{MPW}"CFM68KLibraries.index -merge >> "{MPW}"AllLibraries.index
```

Searching the Contents

After the contents of a library is dumped, you may search it for the function or variable that you are looking for. There are several way to search for a function or variable in the created index of a library. You may open the file with your favorite text editor and use its search facility. Or, you may use the MPW Shell search command. As an alternative, I have created menu items that will search each of the index files dumped.

Using the Startup*Utilities Script

You can use a startup script called Startup*Utilities that builds the Info menu. The user can search the various libraries by selecting different Info menu items. Listing 3 shows you the Startup*Utilities script.

To make it easier to user these scripts, I have added a menu item to the MPW Shell that allows you to select the library that you want to search. To add this menu to your MPW Shell, simply copy/move the file Startup*Utilities into your MPW Startup Items folder. Startup*Utilities then builds the Info menu for you each time the MPW Shell is launched. You can then search the various libraries by selecting different Info menu items. Here is the listing of the Startup*Utilities script:

Listing 4. The Startup*Utilities Script

```
# Add Info menu

AddMenu Info   '(-'                     ''

AddMenu Info   'Create Library Indices/2'  '"{MPW}Scripts:DumpAllLibraries"'

AddMenu Info   '(-'                     ''

AddMenu Info  'Display Obsolete Libraries/3'  'CheckForObsoleteIndex; d
              Open "{MPW}Obsolete.index";'

AddMenu Info  'Search Libraries Index/4'  'CheckForLibrariesIndex; d
              (search /"`Request "Search for..."`"/ -b "{MPW}Libraries.index" >> "{Worksheet}") >=>= Dev:Null';

AddMenu Info  'Search CLibraries Index/5'  'CheckForCLibrariesIndex; d
              (search /"`Request "Search for..."`"/ -b
              "{MPW}CLibraries.index" >> "{Worksheet}") >=>= Dev:Null';

AddMenu Info   'Search SharedLibraries Index/6'  'CheckForSharedLibrariesIndex; d
              (search /"`Request "Search for..."`"/ -b
              "{MPW}SharedLibraries.index" >> "{Worksheet}") >=>= Dev:Null';

AddMenu Info   'Search CFM68KLibraries Index/7'  'CheckForCFM68KLibrariesIndex; d
              (search /"`Request "Search for..."`"/ -b
              "{MPW}CFM68KLibraries.index" >> "{Worksheet}") >=>= Dev:Null';

AddMenu Info   'Search PPCLibraries Index/8'  'CheckForPPCLibrariesIndex; d
              (search /"`Request "Search for..."`"/ -b
              "{MPW}PPCLibraries.index" >> "{Worksheet}") >=>= Dev:Null';

AddMenu Info   'Search All Libraries Index/9'  'CheckForAllLibrariesIndex; d
              (search /"`Request "Search for..."`"/ -b
              "{MPW}AllLibraries.index" >> "{Worksheet}") >=>= Dev:Null';

AddMenu Info   'Search PLibraries Index/0'  'CheckForPLibrariesIndex; d
              (search /"`Request "Search for..."`"/ -b
              "{MPW}PLibraries.index" >> "{Worksheet}") >=>= Dev:Null';

AddMenu Info   '(-'                           ''

AddMenu Info   'Remove All Indices'           'Delete -i "{MPW}Obsolete.index"; d
                                               Delete -i "{MPW}Libraries.index"; d
                                               Delete -i "{MPW}CLibraries.index"; d
                                               Delete -i "{MPW}SharedLibraries.index"; d
                                               Delete -i "{MPW}CFM68KLibraries.index"; d
                                               Delete -i "{MPW}PPCLibraries.index"; d
                                               Delete -i "{MPW}AllLibraries.index"; d
                                               Delete -i "{MPW}PLibraries.index";'

AddMenu Info   'DeInstall Info'                      'Delete -i "{MPW}Scripts:CheckForAllLibrariesIndex"; d
                                               Delete -i "{MPW}Scripts:CheckForCFM68KLibrariesIndex"; d
                                               Delete -i "{MPW}Scripts:CheckForCLibrariesIndex"; d
                                               Delete -i "{MPW}Scripts:CheckForLibrariesIndex"; d
                                               Delete -i "{MPW}Scripts:CheckForObsoleteIndex"; d
```

```
                                        Delete -i "{MPW}Scripts:CheckForPLibrariesIndex"; d
                                        Delete -i "{MPW}Scripts:CheckForPPCLibrariesIndex"; d
                                        Delete -i "{MPW}Scripts:CheckForSharedLibrariesIndex" d
                                        Delete -i "{MPW}Scripts:DisplayObsolete"; d
                                        Delete -i "{MPW}Scripts:DumpAllLibraries"; d
                                        Delete -i "{MPW}Scripts:DumpCFM68KLibraries"; d
                                        Delete -i "{MPW}Scripts:DumpCLibraries"; d
                                        Delete -i "{MPW}Scripts:DumpLibraries"; d
                                        Delete -i "{MPW}Scripts:DumpPLibraries"; d
                                        Delete -i "{MPW}Scripts:DumpPPCLibraries"; d
                                        Delete -i "{MPW}DumpSCLibraries"; d
                                        Delete -i "{MPW}Scripts:DumpSharedLibraries" d
                                        Delete -i "{MPW}Startup Items:Startup*Utilities"'

AddMenu Info   'Remove Info Menu'              'DeleteMenu Info'
```

Using the InstallUtilities Script

To further simplify the use of these scripts, you can use the installation script called InstallUtilities. This script places the necessary scripts in the correct folders and adds several items to the Info menu to manage the use of the installed scripts. To use the InstallUtilities script:

1. Set your directory to the 'LocateLibs&Tools' script folder.
2. Execute the InstallUtilities script.

Listing 5 shows you the InstallUtilities script.

Listing 5 - The InstallUtilities Script

```
Duplicate       CheckForAllLibrariesIndex d
                       CheckForCFM68KLibrariesIndex d
                       CheckForCLibrariesIndex d
                       CheckForLibrariesIndex d
                       CheckForObsoleteIndex d
                       CheckForPLibrariesIndex d
                       CheckForPPCLibrariesIndex d
                       CheckForSharedLibrariesIndex d
                       DisplayObsolete d
                       DumpAllLibraries d
                       DumpCFM68KLibraries d
                       DumpCLibraries d
                       DumpLibraries d
                       DumpPLibraries d
                       DumpPPCLibraries d
                       DumpSharedLibraries "{MPW}Scripts:" -y

Duplicate       Startup*Utilities "{MPW}Startup Items:" -y

"{MPW}Startup Items:"Startup*Utilities
```

Summary

If you encounter an undefined symbol error message from the linker, what you need to do is dump the contents of all libraries, using the MPW tools DumpObj, DumpPEF, or DumpXCOFF, and search for the missing symbol. Because this may prove to be quite tedious to perform by hand, a better alternative is to use the QuickView database or generate your own database with the scripts provided here and search the resulting text files.

Further References

MPW Command Reference

Downloadables

 Acrobat version of this Note (K)

 AppleWorks version of this Note (26K)

 Binhexed MPW Scripts Folder (46K)

To contact us, please use the Contact Us page.
Updated: 7-Nov-96