EARLY MACINTOSH TECHNICAL INFORMATION

APPLE MACINTOSH APPLICATION DEVELOPMENT PROCESS

COMMENT

3RD PARTY MACINTOSH DEVELOPMENT NOTES

AUTHOR

DAVID T CRAIG

DATE

20 JULY 1984

SOURCE

DAVID T CRAIG • JANUARY 2004

Apple Macintosh Application Developement Process

+--+
| Property of P^3, Inc. |
| Created by David T. Craig [20-Jul-1984] |
+--+

INTRODUCTION

This documentation describes the developement process of computer software applications for the Apple Macintosh (henseforth called 'Mac') computer. Currently developement can be done using either Pascal or 68000 Assembly Language. For this documention only the Pascal usage is described. Currently applications can only be developed with the Apple Lisa 2.0 computer (but in the future Mac-based developement programs will appear). Lisa 2.0 developement is only discussed here in comparison to the older developement system called the Lisa 1.0.

USER REQUIREMENTS

This documentation assumes you are familiar with the Apple Macintosh and Lisa computer systems. If you are not, then refer to the owner manuals of these machines before progressing further. A decent understanding of UCSD Pascal (Standard Pascal will do fine also) would be nice too. If you are not interested in the technical side of software developement, then only the next section (Neccessary Supplies) will be of direct interest and/or use.

NECCESSARY SUPPLIES

Developement can only occur if the correct hardware, software and manuals are purchased. Mac developement requires the following items:

[Hardware]

1) Apple Lisa 2.0 computer with 1M byte of RAM
2) Apple 5M byte Profile hard disk
3) Apple Macintosh computer
4) Apple ImageWriter dot-matrix printer with cable

[SoftWare]

5) Apple Lisa Pascal Workshop
6) Apple Macintosh Workshop Suppliment
7) Apple MacWorks

[Manuals]

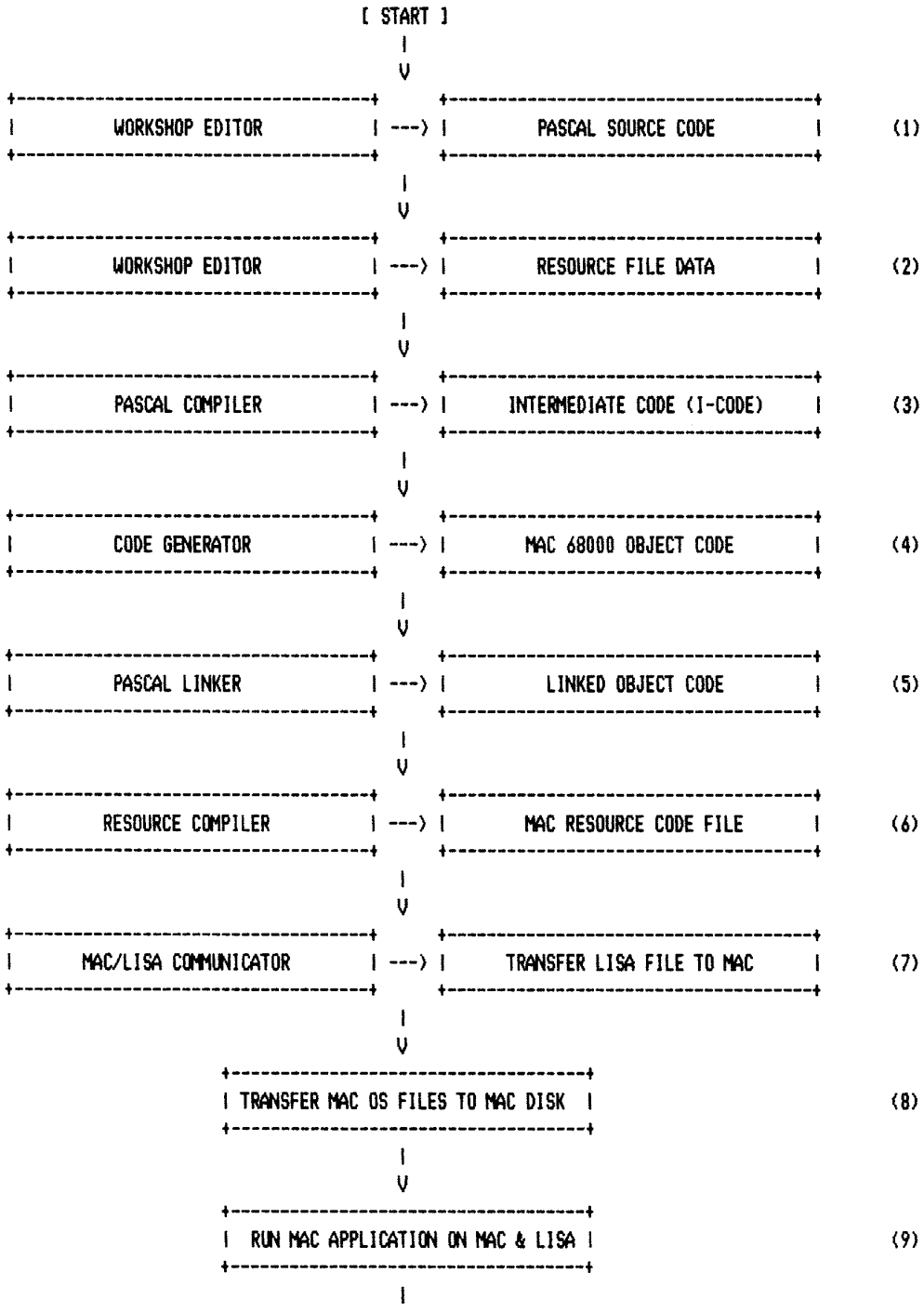
8) Inside Macintosh

Note: Future versions of much of the above will be developed by Apple. At that future time some of the above items might change or disappear totally. For example, Apple is working on a 70M hard disk system called the Priam & will release very soon a 10M hard disk (might replace the Profile). A laser printer is in the works (say bye to the ImageWriter) and a slow color printer, the Scribe, is out now. Other

developement items are currently available, will be out soon or are only ideas in Apple minds.

DEVELOPEMENT PROCESS

Developing Mac applications on the Lisa 2.0 requires creating the Pascal source code on the Lisa 2.0, converting the source code to object code on the Lisa 2.0 and transfering this object code to the Mac by way of the Lisa 2.0's built-in 3.5" floppy disk drive. This developement process can be broken down into nine (9) steps as seen below:



V
[END]

DEVELOPEMENT STEP DESCRIPTIONS

Each of the nine (9) developement steps will now be explained fully. For additional understanding refer to the sample Lisa console outputs as provided after this section. These outputs show the steps and their actions on an actual Mac application called 'Edit'. 'Edit' is a very simple Mac word processor as developed by Apple Macintosh User Education for teaching application developement.

In the below steps reference is made many times to the 'UCSD Pascal main command line'. This is a list of single words displayed at the top of the Lisa screen which allow you to access the various programs you will need. This 'command line' looks like the following:

(V2.0) WORKSHOP: FILE-MGR, SYSTEM-MGR, Edit, Run, Pascal, Basic, Cobol, Quit, ?

The command line is not all displayed due to the length of the Lisa's screen. To see the additonal commands press the '?' key. This generates at the top of the screen:

Assemble, Debug, Generate, MakeBackground, Link, TransferProgram

[STEP 1] WORKSHOP EDITOR ---) PASCAL SOURCE CODE

The first action that is required is the creation of the text Pascal source code on the Lisa 2.0. This is achieved by using the WorkShop editor program. The editor is accessed by way of the main UCSD Pascal command line by pressing 'E' on the keyboard. The editor is named EDITOR.OBJ on the Profile.

[STEP 2] WORKSHOP EDITOR ---) RESOURCE FILE DATA

This step can actually occur anywhere before step 6 below, but is described 2nd for logic's sake. Within this step the WorkShop Editor is used to create the resource file for your application. For further information about these files refer to 'Inside Macintosh'.

[STEP 3] PASCAL COMPILER ---) INTERMEDIATE CODE (I-CODE)

The Pascal compiler converts the text Pascal source code (from step 1) into Intermediate code. This Intermediate code (refered to as I-Code) is a pseudo-language which will be further transformed by the code generator of step 4. The Lisa Pascal compiler is accessed by pressing the 'P' key on the Lisa keyboard in responce to the UCSD Pascal main command line. The Pascal compiler is named PASCAL.OBJ on the Profile.

[STEP 4] CODE GENERATOR ---) MAC 68000 OBJECT CODE

The code generator transforms I-code files (from the compiler) into 68000 object code files. The reason for this step is because the Mac (and Lisa) only understand 68000 code, not I-code or Pascal. Access to the code generator is by pressing 'G' in responce to the main Pascal command line. The code generator can generate 2 different types of 68000 code; one for the Lisa, the other for the Mac. To specify your desires enter '$M+' for the Mac and nothing for the Lisa. See the below sample to see exactly how this is done. The code generator is called CODE.OBJ on the Profile.

[STEP 5] PASCAL LINKER ---) LINKED OBJECT CODE

This step takes the resultant 68000 object code file from the code generator of step 4 and combines or 'links' the specified files to it. The files that will be linked are the Mac operating system files such as Quick Draw, the Menu Manager, etc... After this step you will have an object code file that will

almost run on your Mac. The linker is accessed by way of the main Pascal command line by pressing the 'L' key. For Mac file linkage you must tell the linker this by giving it a special option command. To do this type a question mark, '?', in responce to its prompt & enter 'X+'. Press (Return) to return to the normal linker prompt line. The linker is named LINKER.OBJ on the Profile.

[STEP 6] RESOURCE COMPILER ---) MAC RESOURCE CODE FILE

The resource compiler transforms a text file on the Lisa into a special file called a 'Resource' file for use with step 7. This resource file is actualy a combination of the resource data file of step 2 and the 68000 object code as generated by the linker of step 5. To access the resource compiler press 'R' from the main Pascal command line and enter the word 'RMAKER' into the input statement. The resource compiler is named RMAKER.OBJ on the Profile.

[STEP 7] MAC/LISA COMMUNICATOR ---) TRANSFER LISA FILE TO MAC

Up to this step all of the files that you have worked with are on the Lisa 2.0 (most likely on the Profile hard disk). Now you must transfer the resultant file of step 6, the Mac resource code file, to a Mac floppy disk. This is achieved by way of a program called MacCom (Mac Communications) and the built-in 3.5" floppy disk drive of the Lisa 2.0. To access MacCom type a 'R' in responce to the Pascal main command line and type the word 'MACCOM'. This file is named MACCOM.OBJ on the Profile.

[STEP 8] TRANSFER MAC OS FILES TO MAC DISK

Transfering Mac OS (Operating System) files to the Mac is done on the Mac by way of the Mac Finder which comes with every Mac application. For an application to work on the Mac it needs several files from Apple. These are the ScrapBook, ClipBoard, NotePad, ImageWriter, Finder, System and Desktop. These are normally contained within the System Folder of every Mac disk. For further information on how to achieve this step physically see the Mac owner manual.

[STEP 9] RUN MAC APPLICATION ON MAC & LISA

At last, the final step. Within this test you do the most important action of all developement. You execute your program to test if it works on the Mac and the Lisa 2.0. For the Mac just insert your disk into the Mac, turn on the machine and execute the program. For the Lisa 2.0 (not the Lisa 1.0) you must have the program called MacWorks. It allows the Lisa 2.0 to emulate a Mac. The reason for testing the Mac program on the Lisa is because MacWorks & the Lisa 2.0 provide a very nice market for your hard developement work: why waste it?

MACINTOSH DEVELOPEMENT UTILITIES

Several sources for utilities of value for Mac developement are available. Within the Pascal Workshop exist several excellent ones. The SXRef utility produces a cross reference of all the different words within a text (normally Pascal source) file. Others compare files for differences, search for text strings within files, split or combine several files, etc... These utilities are mainly for assistance with your Lisa Pascal program. But others exist for the Mac specifically. These reside on the Mac disks that are part of the Lisa Pascal Suppliment package. These utilities are: an Icon/Cursor Editor, Resource Mover, Set File, Disk Utility, Boot Configure, Examine File, Font Mover, Font Editor, Hex Dump, Printer Utility and an Alert/Dialog Editor. Some of these Mac utilites are very esoteric, while others are a dream. Most of the Mac utilities have no documentation (they themselves are the documentation).

MAKING DEVELOPEMENT EASIER

There exist several ways to make Mac developement easier & less time consuming. The first way concerns shorting the time & work needed to convert a Lisa Pascal source program to a Mac format disk. This covers developement steps 1-7. The method is the use of Exec files as explained within the Lisa WorkShop manual. This alone is terrific since its very convienent. The use of Lisa & Mac utilities helps alot too. One of the

best places for assistance is the various sample Mac programs as provided by Mac User Education. With the Lisa Pascal Suppliment come several very good & short Mac applications complete with some instructions & source code on disk. These applications are: Edit (a very simple & small word processor), Grow (a little better word processor), File (a fairly detailed word processor with many gee-wiz features), Sound Lab (a short demonstration of the Macs 4 wave sound capabilites), ADeskAcc (a sample desk accesory written though in 68000 Assembly Language) & Boxes (a neat demonstration of the use of Apple's 3-D to 2-D mapping graphics conversion program).

Another simple method to easying the work load is to logically organize the information present on the Profile into 2 sections: Lisa files & Mac files. Within the Lisa files no changes occur with their file names due to the fact that many of the Lisa's files must have their original name. An example of this is the file SYSTEM.OS (the Lisa Operating System). But the Mac files are fair game! I have organized our Profile Mac files into the following arrangement:

```
MAC/Obj/      : Object interface & unit files for the ToolBox
MAC/Samp/     : Sample Mac application source programs
MAC/Util/     : Utility programs for the Mac on the Lisa
MAC/Intf/     : Text interfaces for the ToolBox
MAC/P3/       : P*3, Inc. program files (source, object & resource)
MAC/Lett/     : Letter correspondices to other people & companies
MAC/Doc/      : Lisa or Mac documentation as created by P*3, Inc.
```

With the above beginning file name organization comes another good method. This concerns adding single characters to the end of the file name (but before the file type suffix). I have gone with Apple's method of placing nothing after a Pascal source file, adding a 'R' if its a resource text file and an 'X' if its an Exec file. For example:

```
MAC/Samp/Boxes.TEXT   : A Pascal source file
MAC/SAmp/BoxesR.TEXT  : A resource file
MAC/Samp/BoxesX.TEXT  : An Exec file
```

Files must also have a suffix string describing what type of file it is. Apple has chosen the following:

```
TEXT : A file consisting of Ascii textual data    [ MAC/Samp/Boxes.TEXT ]
OBJ  : An object code file                        [ MAC/Samp/Boxes.OBJ  ]
I    : An intermediate code file                  [ MAC/Samp/Boxes.I    ]
RSRC : A compiled resource file                   [ MAC/Samp/Boxes.RSRC ]
```

To further assist file name understanding the object output for the Linker should have an 'L' (for Link file) immediatly after the file name to show that this is the result of the Linker. As an example:

```
MAC/Samp/EditL.OBJ
```

For a physical example of the above file name conventions the file directory for all the Mac files on P*3's Profile is listed below:

Filename	Size	Psize	Last-Mod-Date	Creation-Date	Attr
MAC/Doc/Lisa2MacDevel.TEXT	29696	58	07/20/84-22:27	07/20/84-22:27	
MAC/Intf/Elems.Text	4096	8	06/02/82-22:06	06/02/82-22:06	L
MAC/Intf/Graf3D.Text	4096	8	03/01/84-10:42	03/01/84-10:42	L
MAC/Intf/MacPrint.Text	21504	42	07/08/84-12:23	07/08/84-12:22	L
MAC/Intf/OSIntf.Text	28672	56	07/08/84-12:35	07/08/84-12:35	L
MAC/Intf/PackIntf.Text	9216	18	03/06/84-16:22	03/06/84-16:22	L
MAC/Intf/QuickDraw.Text	17408	34	07/08/84-12:47	07/08/84-12:47	L
MAC/Intf/QuickDraw2.Text	4096	8	02/15/84-08:46	02/15/84-08:46	L

```
MAC/Intf/Sane.Text          9216   18  02/20/84-15:41  02/20/84-15:41  L
MAC/Intf/ToolIntf.Text     36864   72  07/08/84-13:04  07/08/84-13:04  L
MAC/Lett/Apple.TEXT         4096    8  07/20/84-11:05  07/20/84-11:05
MAC/Obj/DMgrUtil.Obj         512    1  09/30/83-19:31  09/30/83-19:31  L
MAC/Obj/Elems.Obj           2560    5  06/02/82-22:40  06/02/82-22:40  L
MAC/Obj/ElemsAsm.Obj        1024    2  02/20/84-15:54  02/20/84-15:54  L
MAC/Obj/Graf3D.Obj         13824   27  03/01/84-10:44  03/01/84-10:44  L
MAC/Obj/MacPasLib.Obj      10752   21  03/05/84-13:08  03/05/84-13:06  L
MAC/Obj/MacPrint.Obj       18944   37  02/27/84-16:36  02/27/84-16:36  L
MAC/Obj/OSIntf.Obj         24064   47  03/11/84-18:20  03/11/84-18:20  L
MAC/Obj/OSTraps.Obj        12288   24  03/21/84-09:07  03/21/84-09:05  L
MAC/Obj/PackIntf.Obj        7680   15  03/11/84-18:22  03/11/84-18:22  L
MAC/Obj/PackTraps.Obj       2048    4  03/11/84-18:57  03/11/84-18:56  L
MAC/Obj/PrLink.Obj          1024    2  02/20/84-15:55  02/20/84-15:55  L
MAC/Obj/QuickDraw.Obj      16896   33  03/20/84-14:22  03/20/84-14:22  L
MAC/Obj/Sane.Obj           16896   33  02/20/84-15:55  02/20/84-15:55  L
MAC/Obj/SaneAsm.Obj         2560    5  02/20/84-15:55  02/20/84-15:55  L
MAC/Obj/StdFile.Obj         3584    7  02/27/84-15:29  02/27/84-15:29  L
MAC/Obj/StdFile68K.Obj       512    1  02/20/84-15:55  02/20/84-15:55  L
MAC/Obj/ToolIntf.Obj       35328   69  03/20/84-14:24  03/20/84-14:24  L
MAC/Obj/ToolTraps.Obj       2048    4  03/11/84-18:55  03/11/84-18:55  L
MAC/P3/Investor.A.TEXT     14336   28  07/19/84-22:25  07/19/84-22:25
MAC/Samp/ADeskAcc.Text     18432   36  02/20/84-16:10  02/20/84-16:10  L
MAC/Samp/Boxes.Text         7168   14  03/02/84-09:18  03/02/84-09:16  L
MAC/Samp/BoxesR.Text        2048    4  03/14/84-21:10  03/14/84-21:10  L
MAC/Samp/BoxesX.Text        2048    4  03/14/84-21:34  03/14/84-21:34  L
MAC/Samp/Edit.TEXT          6144   12  07/15/84-14:49  07/15/84-14:49  L
MAC/Samp/EditR.Text         2048    4  03/14/84-21:09  03/14/84-21:09  L
MAC/Samp/Exec.Text          6144   12  07/08/84-13:35  07/08/84-13:35  L
MAC/Samp/File.Text         73728  144  07/08/84-13:48  07/08/84-13:48  L
MAC/Samp/FileAsm.Text       8192   16  07/08/84-14:26  07/08/84-14:26  L
MAC/Samp/FileR.Text         6144   12  03/11/84-21:03  03/11/84-21:03  L
MAC/Samp/Grow.Text         10240   20  03/02/84-09:18  03/02/84-09:18  L
MAC/Samp/GrowR.Text         2048    4  03/16/84-11:22  03/16/84-11:22  L
MAC/Samp/SoundLab.Text     19456   38  07/08/84-14:42  07/08/84-14:41  L
MAC/Samp/SoundLabR.Text     2048    4  03/16/84-11:31  03/16/84-11:31  L
MAC/Util/Mac.Boot           2560    5  03/16/84-10:12  03/16/84-10:12  L
MAC/Util/MacCom.OBJ        25088   49  03/22/84-10:42  03/22/84-10:41  L
MAC/Util/RMaker.OBJ        27648   54  03/08/84-12:32  03/08/84-12:32  L
```

LISA OUTPUTS OF A SAMPLE PROGRAM DEVELOPEMENT

The below items are direct screen outputs of most of the developement steps. They provide a concrete example of what to do and expect from your Lisa.

```
+-------------------------------------------------------------------------------------------+
|                  M A C I N T O S H   P A S C A L   P R O G R A M                          |
+-------------------------------------------------------------------------------------------+

{????????????????????????????????????????????????????????????????????????????????????????
 ??                        E      D      I      T                                        ??
 ????????????????????????????????????????????????????????????????????????????????????????}

  {-------------------------------------------------------------------}
  {A small sample application written in Pascal by Macintosh User Education}
  {-------------------------------------------------------------------}
```

```
PROGRAM Edit;

   {$L-} {Turn output listing OFF}

   USES {$U-} {Turn Lisa Libraries OFF}

      {$U MAC/Obj/QuickDraw} QuickDraw,
      {$U MAC/Obj/OSIntf   } OSIntf,
      {$U MAC/Obj/ToolIntf } ToolIntf;

   {$L+} {Turn output listing ON}

   CONST
      lastMenu  = 3;   { number of menus                  }
      appleMenu = 1;   { menu ID for desk accessory menu }
      fileMenu  = 256; { menu ID for File menu            }
      editMenu  = 257; { menu ID for Edit menu            }

   VAR
      myMenus                  : ARRAY [1..lastMenu] OF MenuHandle;
      screenRect,dragRect,pRect: Rect;
      doneFlag,temp            : BOOLEAN;
      myEvent                  : EventRecord;
      code,refNum              : INTEGER;
      wRecord                  : WindowRecord;
      myWindow,whichWindow     : WindowPtr;
      theMenu,theItem          : INTEGER;
      hTE                      : TEHandle;

{ÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐ}
   PROCEDURE SetUpMenus;
{ÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐ}

                  { Once-only initialization for menus }

      VAR i         : INTEGER;
          appleTitle: STRING[1];

      BEGIN
         InitMenus; { initialize Menu Manager }
         appleTitle := ' '; appleTitle[1] := CHR(appleSymbol);
         myMenus[1] := NewMenu(appleMenu,appleTitle);
         AddResMenu(myMenus[1],'DRVR'); { desk accessories }
         myMenus[2] := GetMenu(fileMenu);
         myMenus[3] := GetMenu(editMenu);
         FOR i := 1 TO lastMenu DO
           InsertMenu(myMenus[i],0);
         DrawMenuBar;
      END; { of SetUpMenus }

{ÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐ}
   PROCEDURE DoCommand(mResult: LongInt);
{ÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐÐ}

      VAR name: STR255;
```

```
      BEGIN
        theMenu := HiWord(mResult); theItem := LoWord(mResult);
        CASE theMenu OF
          appleMenu:
            BEGIN
              GetItem(myMenus[1],theItem,name);
              refNum := OpenDeskAcc(name);
            END;
          fileMenu:
            BEGIN
              doneFlag := TRUE; { Quit }
            END;
          editMenu:
            BEGIN
              IF NOT SystemEdit(theItem-1)
                THEN
                  BEGIN
                    SetPort(myWindow);
                    CASE theItem OF
                      1: TECut(hTE);
                      2: TECopy(hTE);
                      3: TEPaste(hTE);
                    END; { of item case }
                  END;
            END; { of editMenu }
        END; { of menu case }
        HiliteMenu(0);
      END; { of DoCommand }

{@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@}
{                        T H E   M A I N   E V E N T                              }
{@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@}

    BEGIN { main program }
      InitGraf(@thePort);
      InitFonts;
      FlushEvents(everyEvent,0);
      InitWindows;
      SetUpMenus;
      TEInit;
      InitDialogs(NIL);
      InitCursor;
      screenRect := screenBits.bounds;
      SetRect(dragRect,4,24,screenRect.right-4,screenRect.bottom-4);
      doneFlag := FALSE;
      myWindow := GetNewWindow(256,@wRecord,POINTER(-1));
      SetPort(myWindow);
      pRect := thePort^.portRect;
      InsetRect(pRect,4,0);
      hTE := TENew(pRect,pRect);
      REPEAT
        SystemTask;
        TEIdle(hTE);
        temp := GetNextEvent(everyEvent,myEvent);
        CASE myEvent.what OF
```

```
            mouseDown:
               BEGIN
                 code := FindWindow(myEvent.where,whichWindow);
                 CASE code OF
                   inMenuBar       : DoCommand(MenuSelect(myEvent.where));
                   inSysWindow     : SystemClick(myEvent,whichWindow);
                   inDrag          : DragWindow(whichWindow,myEvent.where,dragRect);
                   inGrow,inContent:
                     BEGIN
                       IF whichWindow<>FrontWindow
                         THEN
                           SelectWindow(whichWindow)
                         ELSE
                           BEGIN
                             GlobalToLocal(myEvent.where);
                             TEClick(myEvent.where,FALSE,hTE);
                           END;
                     END;
                 END; { of code case }
               END; { of mouseDown }
            keyDown,autoKey:
               IF myWindow=FrontWindow
                 THEN
                   TEKey(CHR(myEvent.message MOD 256),hTE);
            activateEvt:
               IF ODD(myEvent.modifiers) { window is becoming active }
                 THEN
                   TEActivate(hTE)
                 ELSE
                   TEDeactivate(hTE);
            updateEvt:
               BEGIN
                 SetPort(myWindow);
                 BeginUpdate(myWindow);
                 TEUpdate(thePort^.portRect,hTE);
                 EndUpdate(myWindow);
               END; { of updateEvt }
          END; { of event case }
        UNTIL doneFlag;
      END. { MAIN }
{??????????????????????????????????????????????????????????????????????????????
 ??    F             I             N             I             S    ??
 ??????????????????????????????????????????????????????????????????????????????}

        +-------------------------------------------------------------------------------+
        |                    R E S O U R C E    T E X T    F I L E                     |
        +-------------------------------------------------------------------------------+

*  EditResDef -- Resource input for small sample application
*               Written by Macintosh User Education

MAC/Samp/Edit.Rsrc

Type MENU
  ,256
  File
```

```
    Quit

  ,257
  Edit
    Cut
    Copy
    Paste

Type WIND
  ,256
  A Sample
  50 40 300 450
  Visible NoGoAway
  0
  0

Type EDIT = STR
,0
Edit Version 1.0    - 12 December 83

Type CODE
 MAC/Samp/EditL,0

       +-----------------------------------------------------------------------------------------+
       |                                C O M P I L A T I O N                                    |
       +-----------------------------------------------------------------------------------------+

(V2.0) WORKSHOP: FILE-MGR, SYSTEM-MGR, Edit, Run, Pascal, Basic, Cobol, Quit, ?P

Lisa Pascal Compiler V1.153 (22-Mar-84)                      17:32:02 20-Jul-84
(c)1981 SVS, Inc.  (c)1983, 1984 Apple Computer, Inc.

Input file - [.TEXT] MAC/Samp/Edit
List file - [.TEXT]
I-code file - [MAC/Samp/Edit] [.I]

[193688 words]    SETUPMEN
[193672 words]    DOCOMMAN
[193696 words] EDIT

Elapsed time: 38.706 seconds.

Compilation complete - no errors found.  2052 lines.

       +-----------------------------------------------------------------------------------------+
       |                    6 8 0 0 0    C O D E    G E N E R A T I O N                         |
       +-----------------------------------------------------------------------------------------+

(V2.0) WORKSHOP: FILE-MGR, SYSTEM-MGR, Edit, Run, Pascal, Basic, Cobol, Quit, ?G

Lisa Pascal MC68000 Code Generator V2.57 (22-Mar-84)         12:05:43 19-Jul-84
(c)1981 SVS, Inc.  (c)1983, 1984 Apple Computer, Inc.

Input file - [.I] $M+
Input file - [.I] MAC/Samp/Edit
Output file - [MAC/Samp/Edit] [.OBJ]
```

```
SETUPMEN - SETUPMEN    Code size =    142
DOCOMMAN - DOCOMMAN    Code size =    182
EDIT     - EDIT        Code size =    494

Elapsed time: 4.302 seconds.

MACINTOSH code generated.

Total code size = 818
```

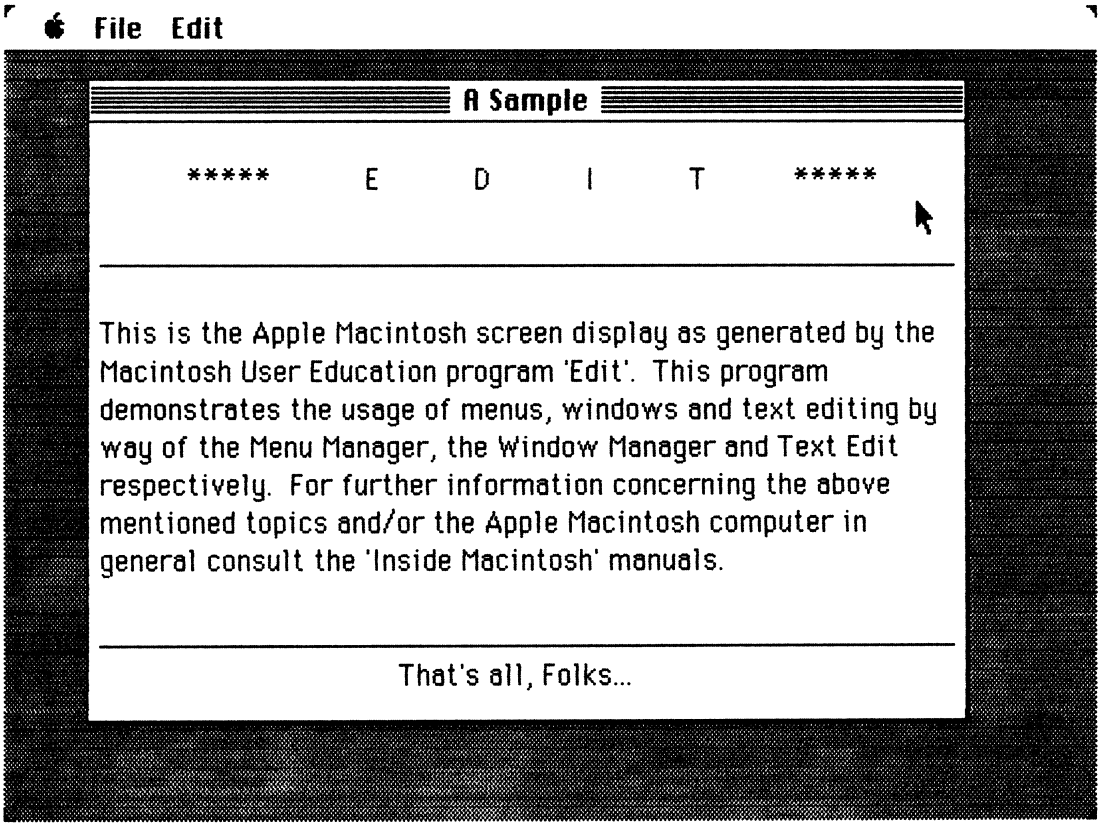
```
+-----------------------------------------------------------------------------------------+
|                                        L I N K I N G                                    |
+-----------------------------------------------------------------------------------------+
```

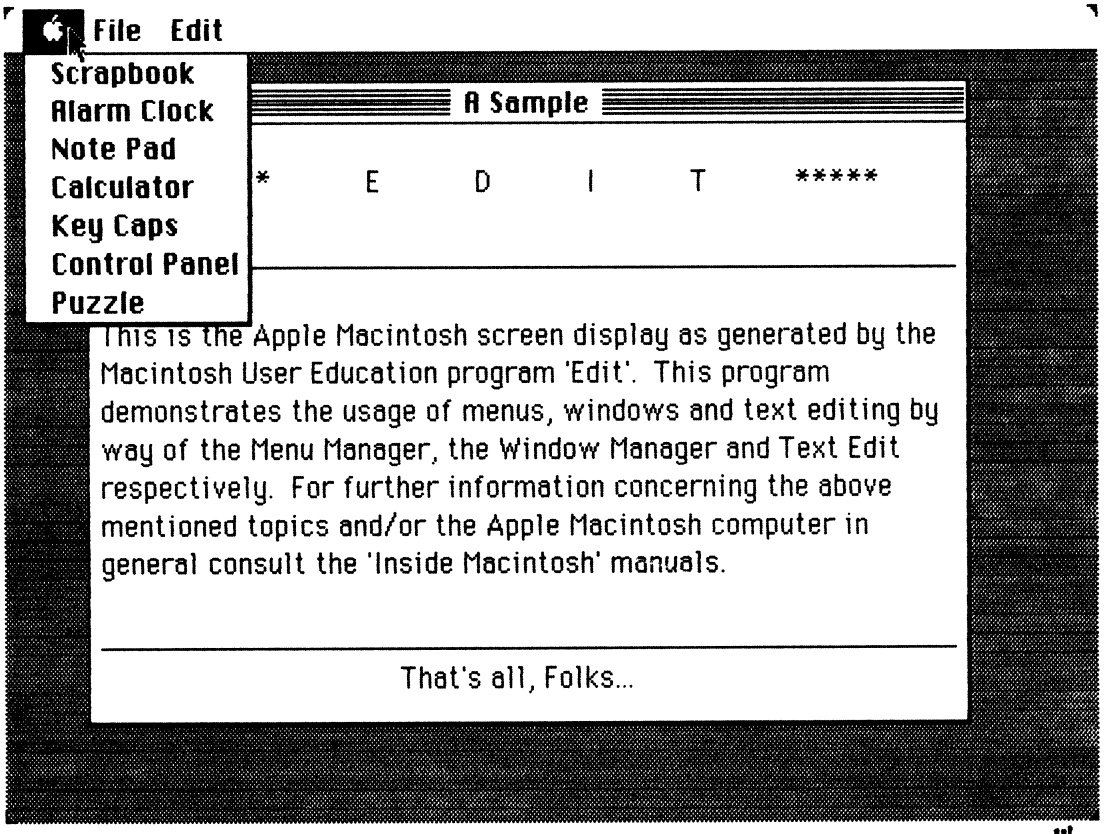
```
(V2.0) WORKSHOP: FILE-MGR, SYSTEM-MGR, Edit, Run, Pascal, Basic, Cobol, Quit, ?L

Linker - PASCAL Program/Intrinsic Library Linker           24-Mar-83
Copyright 1983, Apple Computer, Inc.

Beginning memory -   239192
After static allocation, memory -    83007
Input file [.OBJ] - ?
Options ? X+
Options ?
Input file [.OBJ] - MAC/Samp/Edit
Input file [.OBJ] - MAC/OBJ/MACPASLIB
Input file [.OBJ] - MAC/OBJ/OSTRAPS
Input file [.OBJ] - MAC/OBJ/QUICKDRAW
Input file [.OBJ] - MAC/OBJ/TOOLTRAPS
Input file [.OBJ] -
Listing file [-CONSOLE] - [.TEXT]
Output file [.OBJ] - [redacted] MAC/Samp/EditL
Reading file: MAC/Samp/Edit.OBJ
Reading file: MAC/OBJ/MACPASLIB.OBJ
Reading file: MAC/OBJ/OSTRAPS.OBJ
Reading file: MAC/OBJ/QUICKDRAW.OBJ
Reading file: MAC/OBJ/TOOLTRAPS.OBJ
Read     5 files, max =   100
         2 segments, max =   128
       235 modules, max =  1450
       269 entries, max =  2000
        83 ref. lists, max =  8000
        97 references, max = 16000
Linking Main Program.
Global area: QUICKDRA at $00E6
Active: 13 of 235 read.
Visible: 1 of 269 read.
Global data: $0000E6
Common data: $0000CE
Linking segment #: 0 :          file (JT) seg: 1 size:      950
  Beginning memory -    82691
  Ending memory -    82600
   0 Errors detected.

The output is executable.
```

```
Elapsed time: 169 and 665/1000  seconds.
That's all Folks !!! . . .

       +-------------------------------------------------------------------------------------+
       |                  T H E    R E S O U R C E    C O M P I L E R                        |
       +-------------------------------------------------------------------------------------+

(V2.0) WORKSHOP: FILE-MGR, SYSTEM-MGR, Edit, Run, Pascal, Basic, Cobol, Quit, ?R

Run what program? MAC/Util/RMaker

MacIntosh Resource Compiler Version 5.0          March 8, 1984
Copyright 1983, 1984, Apple Computer, Inc.
  << NOTE -- new resource file format!!! >>
  << NOTE -- automatically handles Workshop MACExecutible !!! >>
Runs under both the Lisa workshop and the Monitor. (Dec 6 1983)

Input file [sysResDef] [.TEXT] - MAC/Samp/EditR

Scanning resource definition file MAC/Samp/EditR.TEXT



The output file is MAC/Samp/Edit.Rsrc.


The jump table size is       8 bytes long.

Type CODE has        2 objects.
Type EDIT has        1 objects.
Type WIND has        1 objects.
Type MENU has        2 objects.

There are  1280 bytes of resource data.
The map is  134 bytes long.

starting pass 2...

       +-------------------------------------------------------------------------------------+
       |              M A C C O M   :   M A C / L I S A    T R A N S F E R                    |
       +-------------------------------------------------------------------------------------+

(V2.0) WORKSHOP: FILE-MGR, SYSTEM-MGR, Edit, Run, Pascal, Basic, Cobol, Quit, ?R

Run what program? MAC/Util/MacCom

MacCom: Delete, Eject, Help, Init, Lisa->Mac, Mac->Lisa, Names, B,F, Quit: H

'Delete' deletes files from a Mac disk.
'Eject' ejects the Mac disk.
 You are using Help now!
'Init' initializes the Mac disk.
'Lisa->Mac' sends files from your Lisa to your Mac.
'Mac->Lisa' sends files from your Mac to your Lisa.
'Names' lists the names on a Mac disk.
'B' allows you to write just the boot blocks on a Mac disk.
```

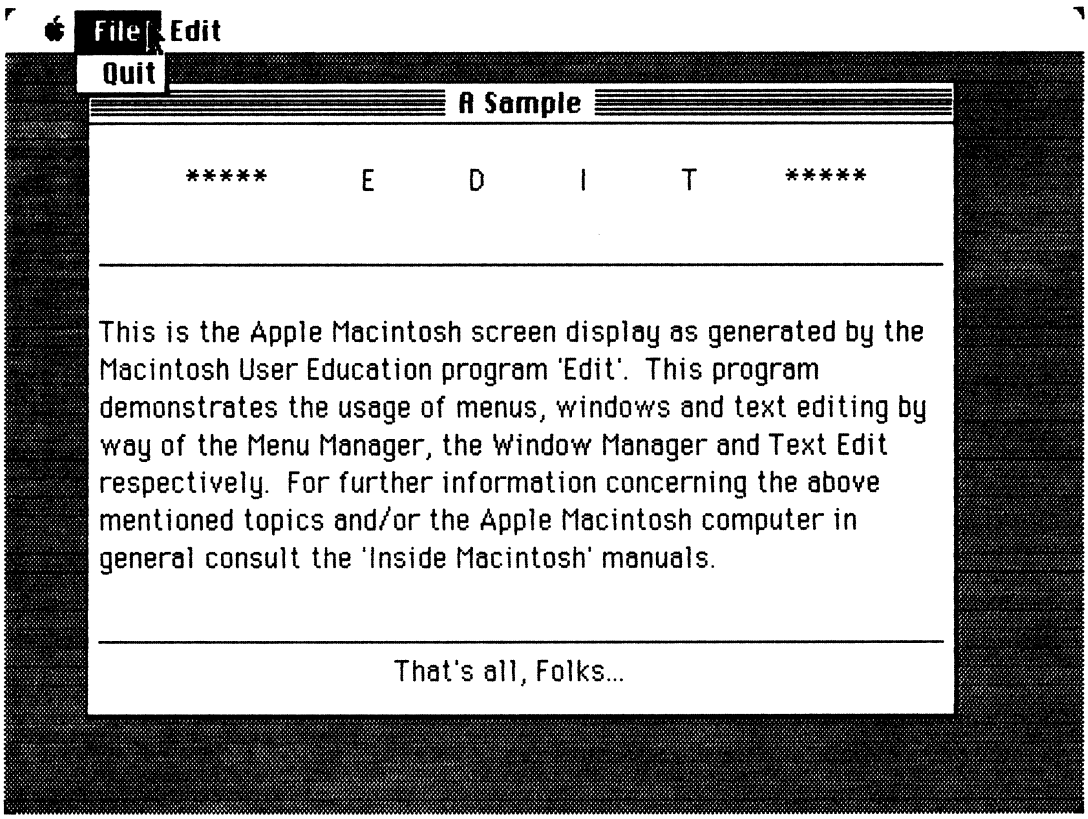
```
'F' is for entering your own finder info data.
'X' allows you to display debugging information.

 Files are specified with the usual '?', '=', pattern matching
 characters.  A '(' prefix gets command input from a file.  This
 can recurse.  A '.RSRC' suffix means this file will be output
 as a resource file.

MacCom: Delete, Eject, Help, Init, Lisa->Mac, Mac->Lisa, Names, B,F, Quit: L

Lisa files to write to Mac disk? MAC/Samp/Edit.RSRC

Read directory on Mac disk: MacVol

Copy to what Mac file? Edit
Type? [APPL]
Creator? [????] DAVE
Set the Bundle Bit? (Y or N)  [No] Y
MAC/Samp/Edit.RSRC                copied to Edit

MacCom: Delete, Eject, Help, Init, Lisa->Mac, Mac->Lisa, Names, B,F, Quit: Q
```

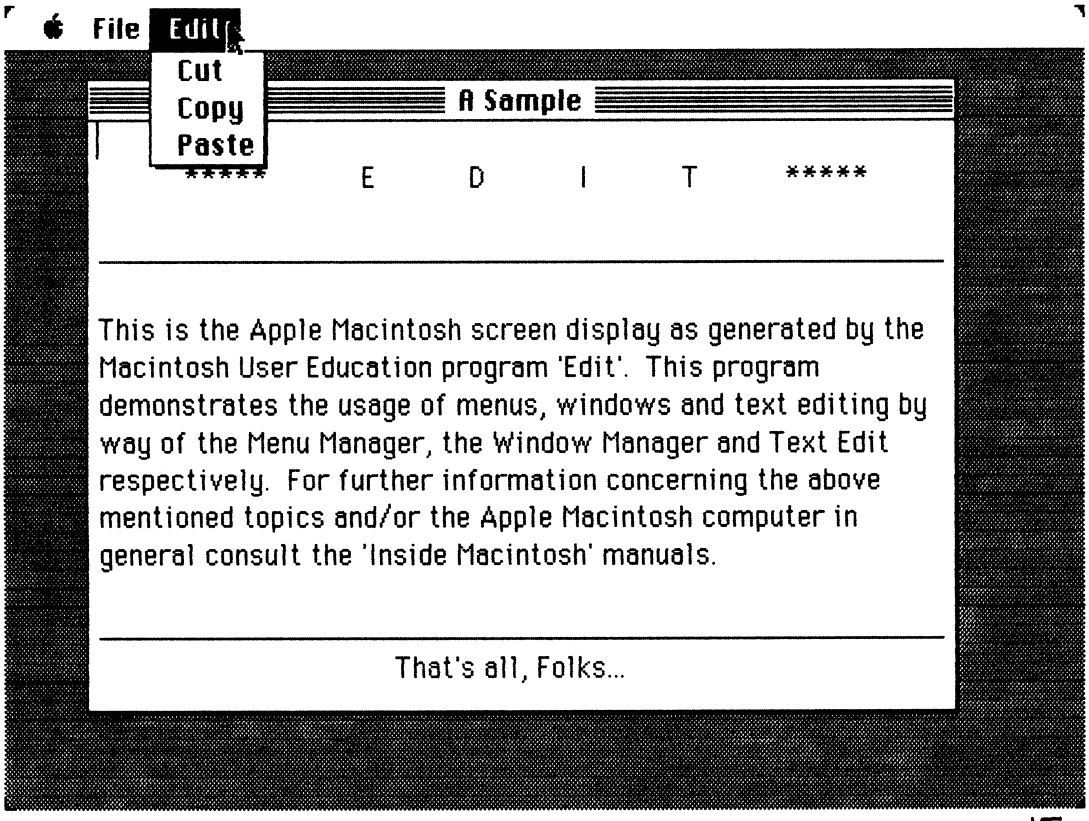
```
**********************************************************************************************************
*                      F           I           N           I           S                      *
**********************************************************************************************************
```


File Edit
A Sample
***** E D I T *****
This is the Apple Macintosh screen display as generated by the Macintosh User Education program 'Edit'. This program demonstrates the usage of menus, windows and text editing by way of the Menu Manager, the Window Manager and Text Edit respectively. For further information concerning the above mentioned topics and/or the Apple Macintosh computer in general consult the 'Inside Macintosh' manuals.
That's all, Folks...

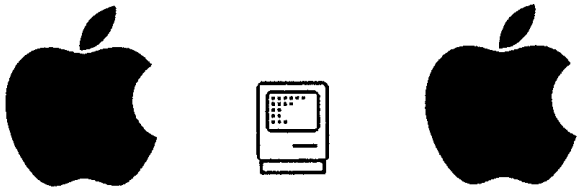

File Edit
Scrapbook
Alarm Clock
Note Pad
Calculator
Key Caps
Control Panel
Puzzle
A Sample
* E D I T *****
This is the Apple Macintosh screen display as generated by the Macintosh User Education program 'Edit'. This program demonstrates the usage of menus, windows and text editing by way of the Menu Manager, the Window Manager and Text Edit respectively. For further information concerning the above mentioned topics and/or the Apple Macintosh computer in general consult the 'Inside Macintosh' manuals.
That's all, Folks...

14


File
Edit
Quit
A Sample
***** E D I T *****
This is the Apple Macintosh screen display as generated by the Macintosh User Education program 'Edit'. This program demonstrates the usage of menus, windows and text editing by way of the Menu Manager, the Window Manager and Text Edit respectively. For further information concerning the above mentioned topics and/or the Apple Macintosh computer in general consult the 'Inside Macintosh' manuals.
That's all, Folks...


File
Edit
Cut
Copy
Paste
A Sample
***** E D I T *****
This is the Apple Macintosh screen display as generated by the Macintosh User Education program 'Edit'. This program demonstrates the usage of menus, windows and text editing by way of the Menu Manager, the Window Manager and Text Edit respectively. For further information concerning the above mentioned topics and/or the Apple Macintosh computer in general consult the 'Inside Macintosh' manuals.
That's all, Folks...

15



END OF DOCUMENT