New Technical Notes

Macintosh

Developer Support

Communications Toolbox Overview Q&As

Communications **M.CM.CTBOv.Q&As**

Revised by: Developer Support Center October 1992
Written by: Developer Support Center October 1990

This Technical Note contains a collection of Q&As relating to a specific topic—questions you’ve sent the Developer Support Center (DSC) along with answers from the DSC engineers. While DSC engineers have checked the Q&A content for accuracy, the Q&A Technical Notes don’t have the editing and organization of other Technical Notes. The Q&A function is to get new technical information and updates to you quickly, saving the polish for when the information migrates into reference manuals.

Q&As are now included with Technical Notes to make access to technical updates easier for you. If you have comments or suggestions about Q&A content or distribution, please let us know by sending an AppleLink to DEVFEEDBACK. Apple Partners may send technical questions about Q&A content to DEVSUPPORT for resolution.

New Q&As and Q&As revised this month are marked with a bar in the side margin.

How to obtain Macintosh Communication Toolbox

Written: 4/29/91
Last reviewed: 5/22/91

Does System 7.0 ship with the Macintosh Communications Toolbox?

___

Though the Communications Toolbox is a part of System 7.0, the Communications Tools are not. They are, however, available from APDA as the following products:

- Macintosh Communications Tools Basic Connectivity Set, Version 1.0 (part # M0379LL/B)
- LAT: Local Access Transport Connection Tool, Version 1.0 (M0800LL/A)
- Serial NuBus™ Tool, Version 1.0F11 (M0950LL/A)
- VT320 Terminal Emulation Tool, Version 1.0 (M0801LL/A)

You can license the above tools through Software Licensing. If you need the Communications Tools (or any other connection tool) to run a third-party product, the software probably is provided as part of the third-party product.The specific tools you need depend on the third-party product.

Macintosh Communications ToolBox documentation

Written: 8/26/91
Last reviewed: 9/17/91

Can you point me to documentation on the Macintosh Communications Toolbox?

___

Inside the Macintosh Communications Toolbox (Addison-Wesley, APDA #M1215LL/A, $24.95) is a nice, comprehensive manual, detailing how to write programs that use the Communications Toolbox. In addition, the documents for the newly updated Macintosh Communications Toolbox 1.1 APDA packages are useful supplements to the Addison-Wesley reference.

Macintosh Communications Toolbox Reference changes

Written: 8/8/91
Last reviewed: 8/13/91

Is the new Addison-Wesley Macintosh Communications Toolbox Reference significantly different from the MacintoshCommunications Toolbox 1.0 reference manual that’s been available from APDA for a while?

___

The Addison-Wesley manual has more information regarding extensions to the File Transfer Manager. It also has a description of the scripting interface to the tools and the Basic Connectivity Set, as well as having some of the text reworked. If you’re unsure about buying it, try taking a look at it first at a computer bookstore.

Macintosh Communications Toolbox manual init correction

Written: 8/30/91
Last reviewed: 8/30/91

The latest Macintosh Communications Toolbox manual says InitCTBUtilities should be called after InitCRM (page 192), but it should be the other way around. Call InitCTBUtilities before InitCRM, as do SurferPlus and Sessions.

System 7 initializes all appropriate CTB utilities itself, so it shouldn’t matter which order you call these routines if you’re using System 7, but it does matter for earlier versions of the operating system.

Systems 6 & 7 Macintosh Comm Toolbox (CTB) installation

Written: 8/26/91
Last reviewed: 8/30/91

How do users install the Macintosh Communications Toolbox (CTB)?

___

The Communications Toolbox consists of two parts: the CTB managers and the CTB tools. The installation procedure for CTB tools is different between System 6 and System 7. Under System 6 the CTB tools are dragged from the Basic Connectivity Set disk to the Communications Folder in the System Folder on the hard disk. Under System 7 the CTB tools are dragged from the Basic Connectivity Set disk to the Extensions folder in the System Folder on the hard disk. Of course, because of the way System 7 works, CTB tools can simply be dragged to the System Folder and the Finder will automatically move them to the Extensions folder where they belong.

No installation of the CTB managers is required under System 7, since System 7 includes the Communications Toolbox as part of the system. Users running System 6.0.5 should use the Installer and Install script on the Communications 1 disk to install the CTB managers and other resources onto their hard disks. Users running System 6.0.7 should use the Installer and Install script on the Network Products Installer disk, which is part of the System 6.0.7 set users receive with their Macintosh systems. The Network Products Installer disk does not contain the CTB managers and other resources, but it prompts the user to insert the Communications 1 disk during the installation procedure.

The Basic Connectivity Set and Communications 1 disks should be shipped with your CTB-aware product. Contact Apple's Software Licensing group (AppleLink SW.LICENSE) for a licensing agreement to ship the disks.

How to "de-install" Communications Toolbox under System 6.0.x

Written: 1/17/91
Last reviewed: 1/17/91

I have installed the Macintosh Communications Toolbox (CTB) under System 6.0.7. What do I need to do to "de-install" it?

___

As you launch the Comm Toolbox Installer, hold down the option key. When the installer window comes up, you can toggle back and forth between the install and remove options using the option key.

OK to use Communications Toolbox in DRVRs, cdevs, and INITs

Written: 11/6/90
Last reviewed: 12/19/90

Can the Macintosh Communication Toolbox (CTB) be used in a DA? InitCM has to be called and the manual says to only call it one time (implies a crash if it is called twice). Is there any way to know if CM has already been INITed?

___

Yes, it is all right to call CTB Initialization routines from CODE Resources. This includes DRVR, cdev, and INIT. The CTB Reference Manual was not clear about this.

Macintosh Communications ToolBox (CTB) initialization

Written: 9/24/91

Last reviewed: 9/24/91

Is there a way to tell if Macintosh Communications ToolBox (CTB) is already initialized? The CTB documentation says to make initialization calls once and only once.

___

The CTB documentation isn’t really true in that regard. It was originally thought that you couldn’t re-initialize the CTB, but that was changed, and it is safe to do so.

Macintosh CTB initialization calls from nonapplications

Written: 4/4/91
Last reviewed: 6/7/91

What are the effects of InitCM, InitCRM, InitCRMUtilities, and similar calls to the Macintosh Communications Tools by nonapplications such as INITs or background processes? What effect do multiple INIT calls have on Communications Tools operations and any existing connections? What do these calls do, and when (and how many times) is it safe to call them?

__

All these initialization calls deal with the process of loading the 'cmtb' resource into the system heap, putting entries into the Communications Toolbox (CTB) dispatch table with references to the current heap zone, and setting up various things having to do with resource management for the operation of the CTB. You can call them over and over again without damaging existing information.

With the perspective provided by that information, you can see that a synergy exists between operations of the CTB and the Resource Manager. The CTB needs to be in the appropriate heap zone when a ConnHandle is generated or referenced for it to operate as desired. If you’re writing a background process that maintains its own persistent heap zone, you shouldn’t have any difficulties using the CTB.

If you’re writing an INIT, matters are more complex. You must SetZone(SystemZone) while performing all CTB operations (including initialization, CMGetProcID, and CMNew) to be sure the heap zone will persist after the INIT has completed. Possibly the reason for doing this at INIT time is to create ConnHandles that will persist for CTB access at a later time. It seems reasonable that ConnHandles thus created require a SetZone(SystemZone) by any routine that needs to access them for Read/Write/Status/Delete operations.

Customizing a modem’s initialization string

Written: 8/30/91
Last reviewed: 8/30/91

How do you customize the intialization string of a modem?

If the user holds down the option key while clicking on the Modem Type Popup menu in the lower left corner of the “Apple Modem Tool” dialog box, then an additional menu item appears. This item is called “custom,” and when selected, it brings up another dialog that contains the initialization string of whichever modem type (such as Hayes or Apple Personal) was LAST selected. You can then customize the intialization string to whatever you want it to be. In other words, select the modem, use the option-select method to get to custom, and then edit the init string.

Trapping the "connect" message returned to CTB AMT

Written: 8/15/91
Last reviewed: 8/15/91

When I use the Macintosh Communications Toolbox (CTB) Apple Modem Tool (AMT) to dial and connect to Telenet or CompuServe, I never see the "connect" message returned by the network. Does the tool remove the "connect" message? If so, how can I trap on the "connect" message?

___

The Communications ToolBox (CTB) Apple Modem Tool traps the keywords, such as "Connect," and does not pass this information on to the application. You'll need to set a search stream using CMAddSearch.

Mac Communications Toolbox (CTB) theEnvirons initialization

Written: 10/1/91
Last reviewed: 10/1/91

The Macintosh Communications Toolbox (CTB) Connection Manager and Terminal Manager each support a "get environment" function in order that the application can obtain the current environment of the connection or terminal record. They are declared as follows in the Pascal interface files:

```
FUNCTION  CMGetConnEnvirons(hConn: ConnHandle;
                           VAR theEnvirons: ConnEnvironRec): CMErr;

FUNCTION  TMGetTermEnvirons(hTerm: TermHandle;
                           VAR theEnvirons: TermEnvironRec): TMErr;
```

and in the C header files:

```
pascal CMErr CMGetConnEnvirons(ConnHandle hConn,
                              ConnEnvironRec *theEnvirons);

pascal TMErr TMGetTermEnvirons(TermHandle hTerm,
                              TermEnvironRec *theEnvirons);.
```

"Inside the Macintosh Communications Toolbox" does not make it clear that the respective theEnvirons parameter blocks used for these two functions require initialization for the functions to operate correctly. For CMGetConnEnvirons, theEnvirons.version should be initialized with the constant curConnEnvRecVers; similarly, TMGetTermEnvirons needs theEnvirons.version initialized with the constant curTermEnvRecVers. These constants are defined in Connections.p and Terminals.p (Connections.h and Terminals.h for C).

Initializing theEnvirons properly will prevent error result codes envBadVers or envVersTooBig. On return with cmNoErr or tmNoErr results, CMGetConnEnvirons and TMGetTermEnvirons will have filled in theEnvirons correctly for the version you specified and will update theEnvirons.version with the current revision level of the environment function call.

Macintosh Communications Toolbox resource ID conflicts

Written: 8/15/91
Last reviewed: 10/8/91

MPW’s Simple I/O Window (SIOW) package is incompatible with certain Macintosh Communications Toolbox (CTB) Tools because of difficulties with CMGetConfig and CMChoose. The CTB and Tools do a somewhat imperfect job of looking in the correct resource map for string components used in CMGetConfig; the application-defined 'STR ' IDs override the correct Tool defined IDs. CMChoose has a similar problem with 'CNTL' resource conflicts.

This problem is not confined to SIOW; it can happen for any application. In the case of SIOW, it is particularly frustrating since SIOW is a precompiled library and you can’t simply change resource IDs to ones that don’t conflict. Also, you cannot be certain that you won’t ever conflict in this manner for any arbitrary Tool and application.

There is a workaround to this situation which should be used for all applications, not just SIOW-based apps. The CTB makes sure that resource maps for Tools are searched after the system resource map—that is, instead of the resource chain looking like this:

ROM (resource lookup progresses from the bottom up)
System
Application
Document
Tool

it looks like this:

ROM
Tool
System
Application
Document.

The workaround is to save the current resource file reference, set the current resource file to the System file, and return to the proper resource file after completing CMGetConfig or CMChoose. Here is a C code fragment illustrating this technique:

```
{  ...
  short saveResFile;
  Point displayPt;
  Ptr cfgString;

  ...
  // save the current resource file
  saveResFile = CurResFile();
  // set resource file to the System file
  UseResFile(0);

  // call CMChoose to configure the Tool
  err = CMChoose(&connectionHdl,displayPt,NULL);
  // call CMGetConfig to get the configuration string
  cfgString = CMGetConfig(connectionHdl);

  // restore proper resource file
  UseResFile(saveResFile);
  ...
}
```

and again in Pascal:

```
VAR
  ...
  saveResFile : INTEGER;
  displayPt : Point;
  cfgString : Ptr;

BEGIN
  ...
  {save the current resource file}
  saveResFile := CurResFile;
  {set resource file to the System file}
  UseResFile(0);

  {call CMChoose to configure the Tool}
  err := CMChoose(connectionHdl,displayPt,NIL);
  {call CMGetConfig to get the configuration string}
  cfgString := CMGetConfig(connectionHdl);

  {restore proper resource file}
  UseResFile(saveResFile);
  ...
END;
```

Macintosh CTB CRMSerialRecord documentation correction

Written: 10/8/91
Last reviewed: 10/8/91

On page 183 of Inside the Macintosh Communications Toolbox, the CRMSerialRecord data structure contains two fields, inputDriverName and outputDriverName, declared as type StringHandle. These two fields are erroneously described as “pointer to Pascal string” in the documentation further down the page. The correct declaration is type StringHandle.

Installer 3.2 and Macintosh Communications Toolbox

Written: 8/29/91
Last reviewed: 9/24/91

How do we make the Macintosh Communications Toolbox (CTB) installation script work with Installer 3.2 for users running System 6.x?

___

The current installer script for the CTB is not compatible with Installer 3.2, nor will it be in the near future. Installer 3.2 was designed to be compatible with System 7.0. Because the CTB is included with System 7.0, and Systems 6.0.5 and 6.0.7 come with the installation disk necessary to install the CTB, there’s no need to update the script. There isn’t really any Apple-approved way to make the script Installer 3.2-compatible.

Your users who are still using System 6.0.7 (or earlier) need to install the CTB separately from your product. They should have the installation disks necessary to install the CTB on their system. For System 6.0.5, they’ll need to use the Installer and Install script on the Communications 1 disk to install the CTB managers and other resources onto their hard disks. Users running System 6.0.7 should use the Installer and Install script on the Network Products Installer disk, which is part of the System 6.0.7 set users receive with their Macintosh systems. The Network Products Installer disk does not contain the CTB managers and other resources, but it prompts the user to insert the Communications 1 disk during the installation procedure.

The Basic Connectivity Set and Communications 1 disks should be shipped with your CTB-aware product. Contact Apple’s Software Licensing group (SW.LICENSE on AppleLink) for a licensing agreement to ship the disks. This will also help ensure that the users are using the version of the tool(s) that is most compatible with your product.

Remember to use Gestalt to make sure that you have the Communications ToolBox available, and check to make sure that the tools you need are available as well. For specific information on using Gestalt, please refer to Inside Macintosh Volume VI and the Macintosh Technical Note “_Gestalt and _SysEnvirons-A Never Ending Story.”

Macintosh System 6 and counting number of dialog items
Written: 4/22/92
Last reviewed: 7/13/92

How do I count the number of items in a dialog without System 7’s CountDITL? My solutions are either messy or dangerous: (1) Fiddle with the dialog’s item list, (2) Try to find out which DITL the dialog used and read the count from the DITL resource, or (3) Repeatedly call GetDItem until garbage is returned :-(

___

It is possible, using the Macintosh Communications Toolbox (and System 6.0.4 or later), to use the CountDITL function since it is included as a part of the CTB. It’s also possible, as you found, to use the first two bytes of the DITL resource to get the number of items in the item list (see Inside Macintosh Volume I, page 427). If the handle to your DITL resource is defined as ditlHandl, you can get at the number of items as follows:

```
short    **ditlHandl;

ditlHandl = (short **)ditlRez;
itemcount = (**ditlHandl) + 1;
```