New Technical Notes

Macintosh

Developer Support

WorldWide Overview Q&As

Macintosh Overview

Revised by: Developer Support Center May 1993
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 this month:
Changing DITL resource to adapt to language
Maintaining one application version with localized text
Macintosh 'vers' resource language field

Changing DITL resource to adapt to language

Date Written: 11/18/92
Last reviewed: 11/30/92

We'd like to maintain only one version of our globally distributed application, which would adapt to the language in use by changing DITL resource text items and menu titles and items. Does the Macintosh Operating System support this?

—

Currently the Macintosh Operating System doesn't inherently support localized resources for several languages, or choose the right language according to the localized version of the system. However, your approach of including all localized text items in the same application is absolutely feasible. Just include an option to let the user select the language—somewhere in Preferences, if not in a dedicated "Languages" menu—and design a numbering scheme for the resource IDs such that the resources to be loaded can be determined from the

language code.

It’s better to let the user choose the language, rather than derive it from the system. This provides for a choice in case the user lives in a multilingual region, or in case your application doesn’t include translations for the language of the user’s system.

Because menus, windows, and dialogs are displayed with the system font, this approach works only for languages supported by the system script.

Maintaining one application version with localized text
Date Written: 9/11/92
Last reviewed: 3/1/93

We need to localize our application for several international markets. Do you have any special tools or recommendations for us?

—

You can use a System 7.1 tool called AppleGlot (on the *Developer CD Series* disc) to localize text in your application. Once a file has been localized the first time, the tool can compare versions and copy over everything that has stayed the same (usually 99%) so that it can focus on the text that's different. It also creates a nice audit trail and is pretty easy to use. It should save you a lot of time.

To take full advantage of this tool, you need common code for all localized versions, which is what you're planning to do to avoid the mess of having multiple sources. Occasionally, your application might have features that make sense only on a particular script system; in that case, you can check for that configuration and enable those routines when appropriate. Once you have common source and tools that help localize your application, you can add auxiliary resources for various languages.

If you have only a small amount of text in your application, it makes sense to bundle everything together in one worldwide product. Apple's TrueType fonts, for example, have internal name tables with names and information such as copyright strings in about a dozen languages. Each string is tagged with a platform, script, and language. But if you have a fair amount of textual resources, it might make more sense to have optional files and resources that can be installed as needed.

Unless you intend to support every script and language, you'll probably want to have a set of resources for unavailable languages. You can pick whatever language you want for this other set (English is popular), but the trick is to use only 7-bit ASCII characters. All script systems use the same character codes for the range $00–$7F, which match ASCII. It's the 8-bit characters that differ radically. This means that text that includes characters like …, ™, ©, and • will not display properly on non-Roman script systems. Just substitute text such as . . ., tm, (c), and * for them. You can decide what's appropriate and necessary.

Another thing to consider is checking for and supporting secondary script systems in your application. The Macintosh Toolbox doesn't fully support secondary scripts such as Japanese menus on an English system, but your application can support secondary script data even with the current Toolbox limitations, by using styled text commands.

Macintosh 'vers' resource language field
Date Written: 9/11/92
Last reviewed: 11/24/92

How is the language portion of the 'vers' resource used by the system software, if at all? I’d like to use the 'vers' resource with my multilanguage application.

—

The language field in the 'vers' resource is part of the version ID to make it unique. The Language Manager, part of the Japanese Language Kit, is the first piece of Macintosh system software to reference the language of the 'vers' resource. It uses this information to switch system fonts, making it possible to display localized (such as Japanese) application resources. A language registration utility can be used by the user to override the 'vers' language.

See the Japanese Language Kit for more information.

Disabling multiscript keyboard menu items
Date Written: 8/11/92
Last reviewed: 11/24/92

When a user launches our application, which doesn't support multiple scripts, on System 7.1 with multiple scripts installed, the user is led to believe that multiple scripts work with our application because the Process Manager enables all items in the keyboard menu. How can we manually disable items in the keyboard menu so users won't be confused?

—

In System 7.0 and 7.1 an application can call KeyScript(smKeyDisableKybds) to disable keyboards which are not in the system script or Roman (see *Inside Macintosh* Volume VI, pages 14-43 and 14-44). As of System 7.1, this keyboard disable state is part of an application's context and gets updated when applications are activated and deactivated, so calling this from one application will not affect others.

Users don't have "multiscript" systems yet. (Multiscript means a system with a secondary non-Roman script installed.) Apple is releasing language modules to developers to help get MultiScript support into third-party products. At this point the Finder and desktop managers aren't multiscript-compatible. The Finder is aware of its limitations and does disable the non-Roman/non-Primary script keyboards. This may be an easy resolution for your application, too.

Human interface for wildcards and boolean operators
Date Written: 6/11/91
Last reviewed: 8/1/92

Is there a universally recognized wildcard character for the Macintosh, like the "*" in the MS-DOS world? Furthermore, for Boolean logic, should my application accept Pascal syntax (such as .NOT., .AND., .OR.), C syntax (such as !, &&, ||), or still another convention? My users aren't programming gurus.

—

First, see if there's a friendlier way to implement the wildcard's function. Take a look at System 7 Finder's Find command, for example. If you find wildcard use is necessary, "*" is common, though for file searching any character other than ":" can be used in an HFS filename.

As for Boolean operators, nonprogrammers prefer a syntax that matches English as closely as possible, so AND, OR, and NOT are better than their C counterparts. However, user testing

indicates that the most intuitive, user-friendly way to put Boolean search criteria on a command line is to bring up a dialog with pop-up menus used to form an English sentence describing the search (like System 7's Find). If you can make something like this work for your application, your nontechnical users will love you.

12 Golden Rules of worldwide design

Date Written: 3/1/91
Last reviewed: 4/1/91

I want my product to be usable around the world. Do you have a quick reference list of what I need to do so my Macintosh product can be localized easily?

—

Here's a summary list from Apple's International Software Support group:

(1) All text (including characters like '…') seen by the user must be in resources so they can be translated easily into other languages.
(2) Remember, characters can be 1 or 2 bytes. Be careful of assuming character lengths.
(3) Word order should be easy to change. Watch out for those strCat calls! Use ParamText.
(4) All graphics should be culturally neutral; obtain country feedback.
(5) All sorting and date, time, number formatting information should be coming from the international resources.
(6) All keyboard information should be coming from the system file via the keyboard resources.
(7) If you create a custom resource that contains text, you must provide a ResEdit TMPL or editor to ensure localizability.
(8) All text handling should be accomplished using TextEdit and/or the Script Manager.
(9) Make no assumptions about the character set. Use the International Utilities and Script Manager routines to find out information about a character.
(10) Keep in mind the different standards used around the world, such as for character sets, paper sizes, and ergonomics.
(11) HyperCard stack developers: The convention for indicating text in scripts which needs to be localized is to add a comment "--Δ" and an explanation:
(Δ=option j)
(12) Flexibility! Design in flexibility so your product can be localized in many different ways. For example, go ahead and set a default if you must (for example, a default keyboard layout), but allow the user to select other options if they are available.

International documents reference list

Date Written: 3/1/91
Last reviewed: 4/1/91

Human Interface Guidelines

Inside Macintosh:
“International Utilities,” *Inside Macintosh* Volume I
“Script Manager” and “International Utilities,” Volume V
“Worldwide Software Overview,” Volume VI
Technical Notes:
Macintosh Technical Note “Don’t Depend of Register A5 Within Trap Patches"
Macintosh Technical Note "Fond of FONDs"

Macintosh Technical Note “Using KanjiTalk”
Macintosh Technical Note “Changes in International Utilities and Resources”
Macintosh Technical Note “Key Mapping”
Macintosh Technical Note “Accessing the Script Manager Print Action Routine”
Macintosh Technical Note “Modifying the Standard String Comparison”
Macintosh Technical Note “How to Construct Word-Break Tables”
Macintosh Technical Note “TextEdit Record Size Limitations Revisited
Macintosh Technical Note “Script Manager’s Pixel2Char Routine”
Macintosh Technical Note “Fonts and the Script Manager”
Macintosh Technical Note “Script Manager Variables”
Macintosh Technical Note “International Cancelling”
Macintosh Technical Note “Script Manager 2.0 Date & Time Problems”
Software Development for International Markets (#A27G0016)
Worldwide Development: Guide to System Software (#M7047/A)
Script System Software and associated User Notes
Japanese Interface System (KanjiTalk)
Arabic Interface System
Hebrew Interface System
Chinese Interface System (HanziTalk)
Korean Interface System (HanguTalk)
HyperCard Stack Design Guidelines
You may also want to check out:
TextEdit Tech Notes

International VBL timing on the Macintosh
Date Written: 3/9/90
Last reviewed: 12/17/90

Do VBL tasks execute at 50 Hz in Europe?

—

Televisions use AC line frequency to determine the vertical retrace interval. The Macintosh does not. All Macintosh systems have their own internal 60.15 Hz clock for timing VBL tasks, regardless of whether that clock is actually connected to video circuitry. Tasks installed by _VInstall are always timed by this 60.15 Hz clock. Slotted Macintosh systems may have additional frequencies determined by the installed video card. Tasks installed by _SlotVInstall are timed by these card-specific clocks. In any case, VBL task frequency is not determined by the location of your power source.