Technical Note TN2053

Mac OS X 10.2

This Technote describes software development related changes provided in system software update Mac OS X 10.2.

This Technical Note was created for application developers interested in writing software that is compatible with Mac OS X 10.2. This list includes changes that affect API level programming and product testing, it is not intended to be an exhaustive list of all the changes in software updates.

CONTENTS

[Sep 04 2002]

Apple Help

Apple Help provides system-wide instructional help services for virtually all aspects of Mac OS. The Help Viewer is the main user interface component of the Apple Help technology.

- A problem where AHLookupAnchor could assume incorrect character encoding for the supplied anchor has been corrected. (r. 2953369).
- The new AppleScript commands 'go back', 'go forward', and 'go home' have been added to the Help Viewer's scripting dictionary. These commands have the same function as the buttons appearing in the Help Viewer's window.(r. 2654316). .
- It is now possible to use dynamically generated tables of contents without using frames (r. 2829108). .
- The Apple Help Dynamic Table of Contents feature no longer garbles Japanese AppleTitles when they are provided in a character encoding other than UTF-8 (r. 2859757). .
- Additional support for Internet-based help book content has been added (r. 2831085, 2431836, 2431821). .
- The table of contents template file can now be named either "toctmpl.htm" or "toctmpl.html"(r. 2829094). .
- AppleURLs can now contain partial URL references (r. 2861478). .

Back to top

AppleEvents

Apple events provide a simple inter-application and intra-application communications facility for Mac OS X applications.

- The routine AESend was returning the result code -600 (process not found) in cases where the target process died before returning a result from its handler. AESend has been changed to return error code -609 (connectionInvalid) in these cases. This change has the side effect of correcting a problem where AppleScript would attempt to relaunch the crashed application again when AESend returned -600. (r. 2681459).
- Coercing from a floating point descriptor record to an integer rounded differently depending on the target integer type: typeSInt16 and typeSInt32 used rint-like behavior, rounding to the nearest even integer but typeUInt32 and typeSInt64 always truncated. These coercions all now behave the same as rint. (r. 2635804).
- A problem where the keyEventSourceAttr attribute was being set to kAELocalProcess instead of kAESameProcess for events that an application targeted to itself have been corrected. Now, the keyEventSourceAttr attribute is set to kAESameProcess. (r. 2410753).
- The AppleEvent Manager has been rearchitected to take advantage of mach virtual memory APIs. While clients of the AppleEvent manager APIs should not see any functional difference, it is important to note that the new implementation may differ in behavior for undefined use. In particular, while dereferencing the dataHandle of an

AEDesc has worked in the past, beginning with 10.2 the opacity of the handle is enforced and you must use AEGetDescData/AEGetDescDataSize to extract the value from an AEDesc. AEGetDescData should not be used on a complex AEDesc like a list, record, or AppleEvent, however. Since the data is opaque, you should use the field accessors (AEGetNthDesc, AEGetParamDesc, etc) to extract data from complex descriptors. If you need to serialize a complex AEDesc, use AEFlattenDesc and AEUnflattenDesc. .

- Factored lists and records are deprecated and no longer supported in 10.2. The first parameters to AECreateList should be NULL and 0, ie: AECreateList(NULL, 0, false, &myList); AECreateList(NULL, 0, true, &myRecord); .
- AEGetArray and AEPutArray are unimplemented in 10.2 (r. 3025590) .
- Sending an AppleEvent from a thread other than the main thread, and specifying kAEWaitReply, requires that the outgoing event specify a reply mach port. For example: mach_port_t port; mach_port_allocate(mach_task_self(), MACH_PORT_RIGHT_RECEIVE, &port); AEPutAttributePtr(&myEvent, keyReplyPortAttr, typeMachPort, &port, sizeof(port)); AESendMessage(&event, &reply, kAEWaitReply, kAEDefaultTimeout); mach_port_destroy(port); See AE/AEMach.h for more information. .

Back to top

AppleScript

AppleScript is a component of the operating system that provides a scripting interface for users to automate actions on system and application data. Even more importantly, it allows users to access functionality of applications which would be difficult or impossible to access by hand.

- AppleScript was ignoring the text item delimiter when converting a list to a string if the text item delimiter was Unicode text (or, more generally, anything but a string). This problem has been corrected. (r. 2891903).
- A problem where coercing a list of items to Unicode text does not preserve the encoding of Unicode text items has been corrected. (r. 2883592).
- Coercions to 'short' such as 'set x to y as short' now work as expected - meaning coerce y to x as a short integer (a 16 bit integer). (r. 2863132).
- Added plural form of the "item" class name to Cocoa Scripting. (r. 2862538).
- A problem where disabling startup screen in stay-open applets was not working as expected has been corrected. (r. 2859959).
- A problem where the 'list folder' command would not accept a string as its parameter has been corrected. Now, both the commands 'list folder (path to scripting additions folder)' and 'list folder (path to scripting additions folder as string)' work as expected (r. 2856926).
- A problem where coercing a folder object to a POSIX path then back to a folder object could produce unexpected results has been corrected (r. 2856778).
- A problem where writing an empty string to a file was not being interpreted as a request to set the file position has been corrected (r. 2855993).
- Cocoa scripting correctly converts booleans it receives from AppleScript to boolean NSNumbers. However, when passing boolean NSNumbers back to AppleScript Cocoa was not converting them into AppleScript Boolean values. Cocoa now provides the correct coercion to AppleScript Boolean values when converting boolean NSNumbers (r. 2848562).
- The Cocoa equivalent for the AppleScript type 'msng' (missing value) is now NSNull (r. 2848555).
- NSAppleEventDescriptor now has the designated initializer -initWithAEDescNoCopy method (r. 2842470).
- An option was added to "do shell script" that allows you to turn off translation of line endings in the command output from LF and CRLF to single CRs (r. 2850410).
- A problem where files created by "open for access" did not have a type or creator has been corrected. Such files now have their type set to "TEXT' and their creator set to "ttxt" (r. 2839361).
- Fixed problem where Cocoa Scripting's copy command could copy objects in the wrong order (r. 2838757).
- Fixed problem in Cocoa scripting where records were not being passed to command handlers correctly. Rather than being passed as a dictionary with keys and values corresponding to those in the record, records were being passed as arrays with interlaced keys and values. Now, records are passed as dictionaries with key value pairs (r. 2837938).
- A problem where AppleScript's OSASetProperty call could become unstable following a call to OSAExecute has been corrected (rr. 2830152, 2786611). (r. 2827990)
- A problem where compiled scripts containing typeTargetID address descriptors could not be opened in Mac OS X has been corrected. This problem affected compiled scripts containing statements of the form 'tell app "foo" of machine "eppc://..."' (r. 2811892).
- The Cocoa scripting requirement that classes in scriptSuite actually exist in Objective-C has been relaxed. Now, if the scriptSuite class serves as a protocol, then there's no need for the corresponding Objective-C class to exist (r. 2807626).

- Fixed problem where Scriptable Cocoa applications would leak 2-6 NSTerminologyRegistry objects for each open document/close document pair of AppleScript commands (r. 2796255).
- The 'quoted form' property of strings now returns a valid quoted string suitable for passing to the 'do shell script' command. Previously, the resulting string was not encoded correctly (r. 2793920).
- A problem that could cause a crash in scriptable Cocoa applications that did not include a keyAEName code ('pnam') element in one of their class descriptions has been corrected. (r. 2785426).
- Relative specifiers inside of "whose" tests are now evaluated as expected. (r. 2784592).
- A problem related to the processing of queued events (events received during the execution of an Apple event handler) has been corrected. (r. 2973719).
- AppleScript enumerators in Cocoa now return the correct value when getting a subclass' property from a superclass object. (r. 2962920).
- UTF-16 to UTF-8 coercion now works as expected. (r. 2918203).
- A problem where Cocoa Scripting was mishandling commands of the form "make new <class> at end." has been corrected. (r. 2869492).
- A problem where "do shell script" could fail when with non-ASCII characters in paths has been corrected. (r. 2848082).
- A problem where an AppleScript attempting to send a restart command to a remote machine would cause the local machine running the script to restart has been corrected. (r. 2840060).
- A problem where Cocoa Scripting did not support copying/moving/creating one object on top of another object has been corrected. Now, it is possible to write the following sorts of scripts for Cocoa applications: tell application "TextEdit" to copy first word of text of front document to title of front (r. 2838745).
- AppleScript now supports typeUTF8Text = 'utf8' for UTF-8 encoded Unicode text. (r. 2835411).
- NSNumber<TypeCode> and NSData<...> are now valid scripting types in scripting suite command parameters and class property declarations for Cocoa applications. (r. 2776086).
- In Cocoa scripting, NSDictionarys returned to AppleScript will now be appropriately converted into Apple event records so they can be accessed as records in AppleScripts. (r. 2480527).
- In Cocoa scripting, your - (void) setX: (NSArray *)someX mutator will now be called when someX is an array. (r. 2470009).
- Cocoa scripting now has support for implicitly specified AppleScript properties. Now, the following sort of script works as expected: tell front document of application "TextEdit" to first word (r. 2243743).
- Fixed a problem where osascript would truncate its output if the resulting string returned by the AppleScript it called contained any null characters. Now the entire string is output as expected. (r. 2852985).

IMPORTANT:
The Info.plist key, NSAppleScriptEnabled = true, will soon be required before Launch Services will consider an application scriptable. Developers of scriptable applications should start using this Info.plist key now.

Launch Services lets you find out a number of things about an application, including whether or not it's scriptable. However, if you're a bundle application, you need to put NSAppleScriptEnabled = true in your Info.plist for Launch Services to consider you scriptable. AppleScript in Mac OS X 10.2 takes additional steps to determine if an application is scriptable, but in the future, it will rely on Launch Services exclusively. At that point, if your application is scriptable but lacks the NSAppleScriptEnabled key, it will not show up in lists of scriptable applications, such as the list displayed for the Script Editor's "open dictionary" command.

Back to top

Audio

This section discusses changes and new features in Mac OS X 10.1 related to audio input and output.

Audio HAL

The Audio Hardware Abstraction Layer (HAL) represents the lowest level of access to audio devices, as well as the general characteristics of the device.

- HAL now provides range information to clients so that they don't have to guess about which sample rates are actually supported by way of the new device property kAudioDevicePropertyAvailableNominalSampleRates. (r. 2832444).
- HAL now provides the properties kAudioDevicePropertySubVolumeScalar, kAudioDevicePropertySubVolumeDecibels, kAudioDevicePropertySubMute for controlling the level of iSub speakers. (r. 2829906).
- Added the device property kAudioDevicePropertyDeviceIsRunningSomewhere to AudioHardware.h . This new property can be used to find out whether or not any given device (input and/or output) is in use by some process on the system. (r. 2764738).
- Fixed an intermittent bug where the HAL could occasionally call a driver's performClientIO with invalid parameters. (r. 2974014).
- A new API for enabling/disabling streams by way of kAudioDevicePropertyIOProcStreamUsage. Now, if an IOAudioEngine has multiple IOAudioStreams, a client of the Audio HAL can indicate that it only wants to read or write a subset of those streams. (r. 2829849).
- Fixed a memory leak that could occur each time the physical format of a stream is changed. (r. 2802833).
- Fixed bug where the volume of the USB audio device was being reset to 0 while speakers are being plugged in. (r. 2779726).
- Reduced the probability of choppy sound occurring when running the HAL at buffer sizes lower than 48 frames per buffer. (r. 2728301).
- Added the new device properties kAudioDevicePropertyTransportType and kAudioStreamPropertyTerminalType that can be used to discover what the transport (USB, FireWire, etc) type is for a device and the kinds of functionality that are available for each stream. (r. 2724935).

Back to top

Audio Toolbox

Contains APIs for AUGraph and MusicToolbox. The AUGraph is a high-level representation of a set of AudioUnits, along with the connections between them. You can use these APIs to construct arbitrary signal paths through which audio may be processed, i.e., a modular routing system. The APIs deal with large numbers of AudioUnits and their relationships. The MusicPlayer APIs provide the services of a sequencing toolbox. This toolbox is where events can be collected into tracks, and tracks can be copied, pasted, and looped within a sequence.

- A problem where AUGraph calls could return the value -1 instead of a more informative result code when reporting error conditions has been corrected. (r. 2886642).
- Callbacks in audio headers are now defined using typedefs, which improves compatibility with C users (as opposed to C++ users). (r. 2781424).
- A problem where changing audio formats during audio playback could cause a kernel panic has been fixed. (r. 2762544).
- Built-in audio drivers are now able to report latency. (r. 2652057).
- The new /Library/Audio/Plugin directory has been added as a place to put MAS audio processing plugins. (r. 2935066).
- A number of factory preset properties for audio units were added to the headers in classinfo format. (r. 2934717).
- A memory leak in MusicSequenceLoadSMF was corrected. (r. 2848166).
- A problem where calling IOAudioLevelControl::create with a subtype of 0 would return an incorrect subtype has been fixed. (r. 2841871).
- Fixed FSSpec problems with AudioToolbox.framework function MusicSequenceSaveSMF. Now this function works as expected. (r. 2840253).
- A new createOutputSelector method has been added to IOAudioSelectorControl.h. (r. 2838912).
- A problem where software-controlled input gain only affected the right channel on certain machines has been fixed. (r. 2781825).
- AppleUSBAudio now includes AC-3 support. (r. 2764825).

Back to top

MIDI

This section discusses changes and new features in Mac OS X 10.2 related to MIDI.

- A problem in MIDIEndpointGetEntity() where it could return a successful status result in cases where it failed has been corrected. (r. 2969576).
- A number of bugs in the, SampleUSBDriver, shared source MIDI developers often use to build vendor-specific drivers have been corrected. (r. 2953808).
- kMIDIPropertySingleRealtimeEntity has been added. Some MIDI interfaces cannot route MIDI real time messages to individual outputs; they are broadcast. On such devices the inverse is usually also true -- incoming real time messages cannot be identified as originating from any particular source. When this property is set on a device driver, it specifies the zero-based index of the entity on which incoming real time messages from a device will appear to have originated from. (r. 2941877).
- In actual use, the second argument passed to the CoreMIDI driver's Monitor function did not match its declaration. The declaration has been updated so it is a pointer to a MIDIEndpointRef rather than the MIDIEndpointRef itself. (r. 2877457).
- A problem where CoreMIDI's use of CFRunLoop was allowing runloop sources (such as timers, etc.) to fire during an API call, causing re-entrancy issues, has been corrected. (r. 2852701).
- When using MIDIPacketListAdd() (in CoreMIDI) to construct a MIDIPacketList with one packet. If the MIDIPacketList's size is exactly large enough to contain this packet, MIDIPacketListAdd() adds a packet with length zero (instead of the length requested). This issue has been fixed and MIDIPacketListAdd() now works as expected. (r. 2824218).
- A problem where MIDISendSysex was leaking about 200 bytes per call has been corrected. (r. 2823042).
- A problem where calls to MIDISend with zero-length packets could hang the MIDI server task has been corrected. (r. 2812345).
- MIDIPacketListAdd will produce MIDI packets consisting of a SYS EX message followed by a channel message. The documentation for a MIDIPacket says "In the case of system-exclusive messages, a packet may only contain a single message, or portion of one, with no other MIDI events." This could cause problems for some drivers and/or the MIDIServer itself and cause them to drop data. This problem has been corrected. (r. 2809230).

Back to top

Sound Manager

The Sound Manager is the Carbon API for playing and recording sound.

- A problem where SysBeep did not work if the system output device did not have any mute controls has been corrected. (r. 2846418).

Back to top

Speech Recognition Manager

This manager provides speech recognition support in applications.

- A problem where SRGetIndexedItem() could erroneously generate the error kSRHasNoSubItems has been corrected. (r. 2959584).
- Speech Recognition would occasionally recognize and consume the listen key (ESC by default), even when using kSRNoFeedbackNoListenModes, preventing that key from reaching the application. We no longer respond to or consume the listen key in this situation. (r. 2902460).

Back to top

Text-to-Speech

The Text-to-Speech APIs provide facilities for converting text into audible speech.

- On occasion, the order of phonemes in a callback (as created using SetSpeechInfo()) would be incorrect after calling SpeakBuffer() (even though the spoken phoneme is in the correct order). We now return the phonemes in the correct order to the callback. (r. 2863646).
- When creating phonemes to precisely control text-to-speech pronunciation, using a comma (",") would incorrectly stop the speech. We now correctly pause and continue on a comma. (r. 2769766).
- If a developer calls SpeakBuffer with the controlFlag kPreflightThenPause, and never un-pauses the speech

channel, Speech Recognition and Talking Alert would stop working. This no longer occurs. (r. 2940266).

- A problem where, in rare circumstances, an application with many speech API calls in a row might encounter a race condition that resulted in an application crash has been corrected. (r. 2939995).

Back to top

BSD

This section describes changes and enhancements the BSD APIs provided in Mac OS X.

- The performance and stability of the tcpdump program has been improved to reduce the likelihood of crashing while monitoring network traffic. (r. 2936063).
- The ulimit routine for controlling process limits is now supported. (r. 2933221).
- The BSD lockf() routine has been added. (r. 2926584).
- Flat File host searches now read all the way through /etc/hosts. Now, if a host has multiple addresses (say, an IPv4 and IPv6 address or it has multiple interfaces), there will be multiple lines in the file for that host, and the search won't stop at the first hit. (r. 2922759).
- A problem where shmat would fail with a permission denied error when it was attempted by root has been corrected. (r. 2910913).
- The bzip2 command has been added to the BSD user commands. This tool handles .bz2 compression files. (r. 2903204).
- fs_usage was updated to allow reporting of physical disk I/O. There are 3 main categories of disk I/O that can be reported using fs_usage: Data, MetaData and Paging operations each of which is either synchronous (implied) or asynchronous (called out). These physical operations will not show up when tracing a single process (since they can't be attributed to any particular process). Using the -e option works around this. (r. 2891010).
- Fixed a bug that could cause getpwuid to return the wrong user when a NetInfo user and a local user had the same UID. This correction only applies to cases where the user logged in with the "short" login name (r. 2890314).
- Added the APIs getgrnam_r(), getgrgid_r(), getpwnam_r(), and getpwuid_r(). (r. 2885989).
- A declaration for the type 'struct timespec' has been added to the header file /usr/include/time.h. (r. 2883165).
- declarations and implementation has been provided for the Libc time functions timelocal, asctime, difftime, localtime, strftime, and strptime. (r. 2882786).
- the ld tool has been updated to produce an error message when an attempt to produce an object file with more than 255 sections is made. There is a limit of 255 sections in an object file. However, ld(1) can create object files with more than 255 sections, which creates unusable object files. We now generate an error message indicating "too many sections used, maximum is 255". (r. 2882553).
- fs_usage has been updated to report the following calls: mkfifo, chhdir, fchdir, utimes, futimes, chroot, undelete, delete, revoke, fsctl, load_shared_file, and copyfile (NOTE: Several of these commands do not have man pages at this time (r. 3035842)). (r. 2881912).
- The routines pread(), pwrite(), getsid(), and getpgid() are now exported from Libc. (r. 2880097).
- the BSD call new_system_shared_regions() is now exported from Libc. (r. 2870742).
- The routine posix_madvise() is now exported from Libc. (r. 2869420).
- Added support for the BSD calls: sigrelse() and sighold(). (r. 2867958).
- Updated printf so that it now supports C99 printf format modifiers. (r. 2867110).
- The a64l() routine is now exported from Libc. (r. 2866840).
- The thread safe implementation of strtok(), strtok_r(), is now exported from Libc. (r. 2866805).
- The thread safe implementation of rand(), rand _r(), is now exported from Libc. (r. 2866795).
- MAXLOGNAM is now set to MAXNAMLEN instead of the number 12. This change increases the maximum user name length. (r. 2863236).
- The routines strtoll() and strtoull() are now exported from Libc. (r. 2859956).
- The BSD 4.0 routine fflagstostr() is now exported from Libc. (r. 2857680).
- The BSD routines strlcat() and strlcpy() are now exported from Libc. (r. 2847856).
- The routine mkdtemp() is now exported from Libc. (r. 2846729).
- The routines basname() and dirname() are now exported from Libc. (r. 2843401).

- Cron jobs now use the new BSD program "periodic" for scheduling. (r. 2842289).
- Python 2.1.1 is now installed with Mac OS X. (r. 2839813).
- inetd is being replaced with xinetd. xinetd offers a much easier way of adding, deleting or modifying entries in the inetd daemon list. Rather than editing a single file, /etc/inetd.conf, a directory can have xinetd 'modules' dropped into it or removed in order to change the configuration, making it much easier to modify system behavior. In Mac OS X 10.2, inetd is still running for compatibility reasons (r. 2961142) with xinetd alongside it. This is a potential problem if people are trying to understand why services are running and they are looking in one place for a service being served when it is actually being served in the other place. Developers should move to using xinetd rather than inetd as inetd may not be supported in future releases. (r. 2838772).
- Added Pluggable Authentication Modules (PAM) to system. PAM is a mechanism for Unix utilities to do authentication in a flexible and extensible manner. As new authentication methods are adopted, it is simple to write a new PAM module for that method and have applications automatically become capable of dealing with it, all without recompilation or change to the applications themselves. (r. 2838768).
- The quotaoff is now installed among the BSD tools installed with Mac OS X. (r. 2836259).
- A prototype for getnameinfo() has been added to /usr/include/netdb.h. It was already exported from Libc, only the declaration was missing. (r. 2833476).
- Fixed a bug that could cause sscanf with %n to return strange values. (r. 2832277).
- Fixed a memory leak in printf that could occur when floating point values were provided as parameters. (r. 2831667).
- A problem where the routine sigpending() was not returning the pending signals in the sigset_t pointed to by its parameter has been corrected. (r. 2831405).
- The bash command shell is now installed with Mac OS X, and has replace zsh as the default /bin/sh (r. 2809843).
- Fixed a bug that would cause inet_aton() to return true if passed in a string starting with a space (a poorly formatted address string). (r. 2809345).
- The libedit library has been added to the system. (r. 2807693).
- Previously using tcpdump without arguments it didn't always find the default interface (sometimes found none). Updated default settings so default interface is always discovered. (r. 2797667).
- The routine scanf() now supports the "ll" (ell-ell) length modifier. (r. 2789072).
- Fixed getservbyname() and getservbyport so that they behave as the man page specifies. Previously, both of these calls returned the last matching protocol available. (r. 2785437).
- The command "chmod +/-t" now works as expected. (r. 2785169).
- A problem where the routine gethostbyname() would fail with an IP address if a reverse DNS entry could not be found has been corrected. Now, it returns the IP address. (r. 2771613).
- Fixed sscanf() now works as expected for reading long doubles. (r. 2757634).
- The routine remove() has been corrected so it can delete directories. Previously, remove would fail if asked to delete a directory. (r. 2757437).
- The routine pthread_kill() is now exported from Libc. (r. 2753869).
- The routine nanosleep() is now exported from Libc. (r. 2753843).
- A problem in setlocale() causing it to return incorrect results has been corrected. (r. 2752682).
- A problem where the command line tool m4 could hang under certain circumstances has been corrected. (r. 2749605).
- The routine asprintf is now exported from Libc. (r. 2727904).
- The Rand48() family of functions are now a part of stdlib. These routines include: drand48, erand48, lrand48, nrand48, mrand48, jrand48, srand48, seed48, and lcong48. (r. 2718058).
- The fcntl() options F_PREALLOCATE, F_SETSIZE, F_READBOOTSTRAP, F_WRITEBOOTSTRAP, F_LOG2PHYS now check for locking errors. (r. 2679948).
- The routine mkdtemp() is now exported from Libc. (r. 2613958).
- The gethostbyname() family of functions have been updated so they are now thread-safe. (r. 2238779).
- Removed libcurses from libSystem and replaced it with ncurses. Programs that uses curses functionality will need to be relinked with the new shared library (r. 2219086).
- The tool "automake" is now installed with Mac OS X. (r. 2942208).
- Fixed a bug that would cause mktemp() to underrun the template buffer and return an improperly formatted file name. (r. 2937492).

- readdir_r The thread safe version of the routine readdir(), readdir_r(), is now available. (r. 2930202).
- When working with .a files, ld(1) does not respect the -no_arch_warnings flag (as it does with .o files). We now correctly pay attention to this flag with .o files. (r. 2918296).
- The Terminal text encoding mode can now be changed to UTF-8 to better support international languages. (r. 2911610).
- The routine printf() now supports vector format specifiers. (r. 2911015).
- uudecode now will set execute bits on its output file if they are present in its uuencoded input. It will still not set sticky bits nor set-id bits. (r. 2910030).
- The Ruby scripting language is now installed with Mac OS X. (r. 2809964).
- ntpd v4.1.71 is now installed with Mac OS X. (r. 2809093).
- The routines inet_pton() and inet_ntop() have been added to Libc. (r. 2372291).
- The curses library has been updated to the newer ANSI compliant ncurses library, which supports color and other advanced text attributes as well as offering greatly increased compatibility with applications which rely on having a standards-compliant curses library. One side effect of this change is that ALL APPLICATIONS WHICH USE CURSES MUST BE RELINKED since ncurses is not backwards-compatible with the older version of curses that was shipped in 10.1. The libncurses library is also now available separately from libSystem in order to minimize the memory footprint of other applications which do not use curses, so applications using curses must explicitly link with -lncurses. .
- The default aliases and tab-expansion settings for tcsh have reverted to their factory defaults due to various complaints with the previous highly-customized set of defaults shipped with 10.1 which made using tcsh different under Mac OS X than on any other Unix platform. Users wishing to have some or all of these defaults back again are urged to investigate /usr/share/tcsh/examples/README.

Back to top

Carbon

Carbon is a set of programming interfaces for use in Mac OS applications, especially those being ported from Mac OS 9 to Mac OS X. Carbon includes about 70 percent of the legacy Mac OS APIs and a number of new APIs for Mac OS X.

Alias Manager

The Alias Manager is the part of the operating system that communicates with the file system to maintain alias records that are used to keep track of file and folder locations. The Alias Manager does not create Finder alias files; the Finder creates these files and stores alias records created by the Alias Manager in them.

- Aliases to mounted disk image volumes now reference the .img file. As a result, the disk image will be mounted when the alias is resolved with appropriate mount flags. (r. 2783427).
- Alias resolution now does a more thorough job of matching aliases, especially in cases where similar files can be found on different volumes with the same name. (r. 2776704).
- Two new APIs, FSMatchAlias and FSMatchAliasNoUI, have been introduced that provide FSRef based functionality similar in function to the older MatchAlias routine. (r. 2744821).
- A problem where ResolveAlias was incorrectly resolving minimal aliases on volumes that do not natively support FileIDs has been corrected. (r. 2988555).
- A new API, FSCopyAliasInfo, has been added that allows callers to retrieve detailed information from alias records. (r. 2714855).
- Alias resolution now begins with attempting to resolve the file path name first before attempting to resolve the alias using the file id. This is a fundamental change in the way aliases are resolved and it corrects a problem that would prevent aliases from working as expected on volumes restored from a back up copy. New flags have been added to specify the previous behavior (where the file id would be tried first). Pass kARMTryFileIDFirst to calls which accept a rulesMask or kResolveAliasTryFileIDFirst to calls which accept mountFlags to specify the file id first behavior. (r. 2683228).

Back to top

Appearance Manager

The Appearance Manager extends the facilities provided by the Control Manager, the Dialog Manager, the Menu Manager, and the Window Manager to provide a consistent look and feel for all on screen user interface elements in the Mac OS environment.

- The new constants kThemeSystemFontDetail, kThemeSystemFontDetailEmphazid, kThemeTextColorSystemDetail have been added in Apperance.h. (r. 2818428).
- The themeNoAppropriateBrushErr (-30568) is now defined in MacErrors.h and can be return by SetThemeTextColorForWindow (see comments in Appearance.h for more details). (r. 2808256).
- The new constants kThemeBrushPrimaryHighlightColor and kThemeBrushSecondaryHighlightColor are now available in Appearance.h and should be used as seen in the column view of the Finder. (r. 2763475).
- Performance of Unicode text drawing in items displaying text has been improved. (r. 2704870).
- A new constant, kThemeToolbarFont, has been added to the ThemeFontIDs to support the new Toolbar object. (r. 2822282).
- A problem in Mac OS X 10.1 where drawing with a pattern selected by calling ApplyThemeBackground(kThemeBackgroundWindowHeader ...) did not produce the same results as the call DrawThemeWindowHeader() has been corrected. (r. 2760583).

Back to top

Component Manager

The Component Manager provides facilities for tracking and calling code modules that provide a defined set of services to one or more clients (components).

- In Mac OS X 10.1, the component manager would sometimes crash if a component call was made on an invalid component (such as a component that had been closed). This bug has been fixed. (r. 2772868).

Back to top

Control Manager

The Control Manager provides facilities for drawing and processing user interaction with on screen items inside of windows.

- A problem where the Mach-O implementation of IsAutomaticControlDragTrackingEnabledForWindow was returning an incorrect value has been corrected. This problem could cause unexpected results in drag and drop operations after calls to SetAutomaticControlDragTrackingEnabledForWindow (r. 2447465). (r. 2764233).
- A problem where the minimum width in specified for a list view column could be ignored, or modified to a larger value, during a live resize operation has been corrected. (r. 2761051).
- A new constant, kControlEditTextSingleLineTag, has been added to ControlDefinitions.h for use with the Unicode Text Control. This control data tag allows you to turn on or off single line editing. (r. 2733779).
- GetControlData(..kControlStaticTextTextTag..) will now works correctly even if the text was set using SetControlData(..kControlStaticTextCFStringTag..). (r. 2700210).
- Static Text Controls now truncate text properly when the kControlStaticTextTruncTag is used. (r. 2661131).
- A problem where, under certain conditions, calling SetDataBrowserTableViewColumnPosition could swap the column widths of the columns being moved has been corrected. (r. 2658667).
- A problem where moving the column containing the disclosure triangles could cause the data browser to crash has been corrected. (r. 2655765).
- A problem where calling SetDataBrowserListViewDisclosureColumn could change the column width has been corrected. (r. 2652930).
- A problem where calling ExecuteDataBrowserEditCommand(macControl, kDataBrowserEditMsgCopy) command would not copy the text of the selected items to the scrap has been corrected. (r. 2511863).
- Problems in the Control Manager's handling if IconRefs have been corrected. This was a retain/release problem with icon references that could cause the inadvertent disappearance of icons. (r. 2979503).
- Disposing a control that is the default or cancel button in a window now clears properly and the window no longer attempts to use it. (r. 2954785).
- A problem in the data browser related to column widths of non-resizable columns used to display an hierarchical lists has been corrected. Here, disclosing hierarchical items would not widen columns while collapsing them would make the same columns smaller. (r. 2814835).

- A problem where moving an EditUnicodeText control between windows with EmbedControl would crash has been corrected. Here, the control was keeping some internal fields pointing to the original window. This has been fixed. (r. 2795838).

- FindControlUnderMouse now accepts a NULL value for the outPart parameter. This is useful for cases where your program is only interested in finding the control under the mouse, but not the part code describing the part of the control under the mouse. (r. 2879400).

Back to top

Date & Time Utilities

The Date & Time Utilities provide facilities for accessing the clock chip and APIs for converting dates between various formats.

- Fixed an obscure conversion problem with StringToDate() where it could fail on the date of January 1 of any year where (year mod 400) equals 101. (r. 2860198).

- Added support for time zone support for cities in Portugal, Croatia, and Slovenia. (r.2946953) (r. 2928698).

Back to top

Dialog Manager

The Dialog Manager manages user interactions with windows containing controls arranged and positioned using dialog layout templates.

- RunStandardAlert disposes of the dialog before returning. More comments have been added in Dialogs.h for RunStandardAlert and CreateStandardAlert to clarify the allocation/deallocation issues. (r. 2828656).

- Problems where the DITL accessor routines could crash if passed NULL dialog item list handles have been corrected. (r. 2603639).

- Memory leaks that could occur when using GetNewDialog or DisposeDialog with sheets have been corrected. (r. 2967411).

- A problem where GetDialogItem could alter the font information in the active window port, rather than the explicit dialog port, has been corrected. (r. 2938978).

Back to top

Display Manager

The Display Manager allows users to dynamically change the arrangement and display modes of the monitors attached to their computers.

- Problems with calling DMRegisterExtendedNotifyProc with kFullDependencyNotify have been corrected. (r. 2788058).

- Added new Display Manager notifications for display sleep and wake. Check gestaltDisplayMgrSleepNotifies to make sure it is available and see Displays.h for kDMNotifyDisplayWillSleep and kDMNotifyDisplayDidWake. (r. 2806290).

Back to top

Drag Manager

The Drag Manager supports drag-and-drop operations between windows and applications.

- A new constant, kDragStandardDropLocationTrash, has been added in Drag.h. When used with the new API, SetStandardDropLocation, an alias to the Trash will be set as the drop location (r. 2916609). (r. 2829056).

- Calling SetDragImage() can be called multiple times during a Drag to update the drag image. Unfortunately, there was a problem where the image flags parameter was being ignored after the first call to SetDragImage(). Now, the flags parameter is observed on subsequent calls to SetDragImage. This allows callers to turn on and off the drag region during a drag by changing the kDragRegionAndImage flag. (r. 2641813).

- A problem where the region offset of the masking region was being ignored by SetDragImage has been corrected. (r. 2640917).

- A new API, SetDragImageWithCGImage, has been added to Drag.h to enable callers to specify a CGImage as a drag image. (r. 2916609).

Back to top

Event Manager

The Event Manager manages the delivery of information about system operations and user interaction to applications.

- A problem where an activate events were not being generated properly between calls to BeginFullScreen() and EndFullScreen() has been corrected. (r. 2873868).
- A new constant, kMouseTrackingMouseMoved, has been defined as one of the possible result codes returned by TrackMouseLocation/TrackMouseRegion. Previously, calling TrackMouseLocation/TrackMouseRegion while the mouse button did not work. Now, if your application calls TrackMouseLocation/TrackMouseRegion while the mouse button is up, kMouseTrackingMouseMoved will be called when the mouse is moved. (r. 2792773).
- New Mouse Tracking Region APIs are now available (see Carbon/CarbonEvents.h). (r. 2716688).
- Mouse delta values provided in mouse moved events are now calculated correctly. (r. 2673343).
- A problem where an application using the classic event model would not receive an osEvt (mouse-moved) event when it was switched into the foreground has been fixed. (r. 2671602).
- SendEventToEventTarget now returns paramErr for an invalid target. (r. 2976195).
- Floating utility windows can now receive kEventMouseWheelMoved events. (r. 2751193).
- Events for 'application hidden' and 'application shown' have been added to Carbon Events. (r. 2693184).

Back to top

File Manager

The File Manager provides APIs for storing and retrieving disk-based information.

- A problem where the containerChanged parameter to FSGetCatalogInfoBulk was being ignored (and not set to any value) has been corrected. In Mac OS X 10.2, it is always set to false. (r. 2874842).
- In Mac OS X 10.1, attempts to iterate through the list of mounted volumes before the first call to the event loop would return a list of local volumes and the volume on which the application is running would be returned - remote volumes would not be included in the list of volumes returned. This problem has been corrected. (r. 2895905). (r. 2759485).
- With Mac OS X 10.2, when one process opens a file with write access, it will have exclusive write access. If another application tries to open the same file with write access it will fail. See Technical Note TN2037 for further information. (r. 2663613).
- New volume signature word has been added for FTP servers mounted as volumes. Here's a list of common volume sigwords currently in use: 0x4B47 = FTP, 0x4341 = webDAV, kHFSPlusSigWord = HFS+, kHFSSigWord = HFS, 0xd2d7 = MFS, 0x4B48 = UFS, and 0x4E4A = NFS. (r. 2865998).
- A new set of APIs for mounting, un-mounting and ejecting volumes has been added to the File Manager. The routines allow for both synchronous and asynchronous operations. The routines are FSMountServerVolumeSync, FSMountLocalVolumeSync, FSUnmountVolumeSync and FSEjectVolumeSync (replace Sync with Async for the synchronous variants). (r. 2788787).
- Hard link support in has been added to the File Manager. The new kFSNodeHardLinkBit bit has been added to Files.h that allows callers to determine if a file system reference is actually a hard link. If this bit is set, then callers can use stat(2) to find out more information about the file. (r. 2719873).

Back to top

Folder Manager

The Folder Manager provides facilities for locating "special" folders (for example, the Extensions folder) without relying on the names of those folders. This aids developers in application localization.

- Added the new FindFolder selector constant kManagedItemsFolderType defined that allows callers to locate the user's "Managed Items" folder. (r. 2871449).
- A problem where calls to FindFolder with create flag set was not setting the correct permissions on the Temporary

Items folder has been corrected. Now, the folder is created with "world privileges" with the sticky bit set. (r. 2847142).

- A new FindFolder selector constant, kKeyboardLayoutsFolderType, was added in Folders.h to allow callers to locate the user-installible keyboard layouts folder. (r. 2842566).
- A new FindFolder selector constant, kFindByContentIndexesFolderType, was added that allows callers to find the find by content folder. (r. 2826947).
- A problem where calls to asking it to create the kInstallerLogsFolderType folder (/Library/Receipts/) would always fail with a error of type -43 has been corrected. Now, when this folder is not present, it will be created as expected. (r. 2656007).

Back to top

Gestalt Manager

The Gestalt Manager provides a simple, efficient way for your application to determine the user's hardware and software configuration at runtime so that you can fully exploit the available features or inform the user if a necessary feature is missing.

- The Gestalt selector gestaltCarbonVersion returns version 1.6 when called in Mac OS X 10.2. (r. 2870430).

Back to top

Help Tags

Help Tags are the primary method for providing context sensitive help in Mac OS X. They are the Mac OS X replacement for the older Balloon Help technology.

- The help tag code for menu titles used to check whether the point was inside the absHotRect before displaying the help tag. Now, when checking for a menu title help tag, the absHotRect is now ignored; the menu title's actual bounds is always used. This allows a menu title content callback to return an empty absHotRect. Also, after calling a content callback for a control, window, or menu item, the code now checks for an empty absHotRect and normalize the hot rect to the current global position of the object. This allows content callbacks for these objects to return empty absHotRects. Previously, they had to determine the actual current global coordinates of the object and put that into the absHotRect for the help tag to display. Finally, another minor change to the code now checks whether the mouse is inside the absHotRect before displaying a menu item help tag. This generally will have no effect, but it's meant to allow a custom MDEF that only wants to display help tags over a subset of its content area to customize the hot rect. (r. 2833495).
- In Help, a HelpContentRec content type was added to support a localized string. Newly added is kHMCFStringLocalizedContent, which uses the tagCFString field of the HMHelpContentRec. The CFString is interpreted as the name of a localized string in the application's bundle. CFCopyLocalizedString is used to load the string, and it's released when the system is done with it. Also fixed was a long-standing bug where the content callback was not being called with a kHMDisposeContent request when the content type was CFStringRef. This could lead to memory leaks of CFStringRefs. (r. 2806155).
- It is now possible to break the text displayed in help tags into multiple lines of text by including line feed characters in the text. (r. 2696185).
- It is now possible to create help tags for utility windows. (r. 2526126).

Back to top

Icon Services

Icon Services provides fast and efficient facilities for retrieval of appropriate icon information used to represent files, directories, or other commonly used icons (for example, the caution alert icon, the note alert icon, and the help icon).

- GetIconRefFromFile now badges icons returned for alias files as expected. (r. 2663503).
- A problem where SetIconFamilyData could fails with memFullErr (-108) when given valid parameters has been corrected. (r. 2952322).
- GetIconRefFromFile now supports custom badge resources. .
- GetIconRef now supports creator/types that have been registered with LaunchServices. .
- Some IconServices APIs are now thread safe (see APIs comments in Icons.h). .
- IconServices supports custom media icons via IOMediaIcon property. See Mass Storage release notes for details. .

Back to top

List Manager

The List Manager provides a standard user interface for drawing and managing user interaction with lists of items.

- References to the type ListRef have been removed from Lists.h. All of these references have been replaced with ListHandle (r. 2837663). .

Back to top

Menu Manager

The Menu Manager is the part of the operating system responsible for both drawing the menu bar, and drawing menus and pop-up menus on the screen while the mouse is being held down.

- The Menu Manager now invalidates the command key cache whenever hierarchical menus are inserted or removed. This change clears out any stale command key/menu associations left in the cache after the active menu set is changed. (r. 2866303).
- The new constants kMenuAttrHidden and kMenuItemAttrIncludeInCmdKeyMatching have been added to the added as possible Menu Item attributes. (r. 2835332).
- Implemented support for a MenuContext parameter in menu-originated command events so that the handler can determine from where the command came from. (r. 2790403).
- Removed the limit of 5 hierarchical menus on-screen at once. (r. 2456159).

Back to top

Navigation Services

Navigation Services provides new Open and Save dialogs, allowing users to locate and select files and other resources.

- NavCustomControl/kNavCtlSelectCustomType now works correctly instead of returning error -5697 (kNavCustomControlMessageFailedErr). (r. 2763191).
- You can now pre-select the default location in Navigation Services in the same way it used to work on Mac OS 9. (r. 2759311).

Back to top

PLStringFuncs

PLStringFuncs provides a number of Pascal-string APIs that are similar to the standard C library string routines.

- The routine PLstrcat(a,b) in CarbonCore was broken if length of a + b > 255 characters. It now does proper range checking. (r. 2865047).

Back to top

Process Manager

The Process Manager provides a set of APIs allowing processes to find out information about other processes registered with the Process Manager.

- Calling GetNextProcess or GetProcessInformation with an invalid ProcessSerialNumber now properly returns procNotFound instead of noErr. (r. 2971688).

Back to top

Resource Manager

The Resource Manager provides a data file format for storing and retrieving information used during application and operating system run time.

- The CSResourcesFileMapped info.plist key is now respected for single binary applications. (r. 2704627).
- FSCreateResourceFork API was added to create a resource fork given an FSRef. (r. 2650255).

Back to top

Window Manager

The Window Manager provides facilities for drawing and maintaining windows on the screen.

- When using GetWindowIdealUserState() or ZoomWindowIdeal() on an un-initialized window, we incorrectly returned an empty rect (0,0,0,0). We now return the window's global port bounds instead. (r. 2804328).
- In some circumstances, clicking on a window with an active sheet brought the window to the front, but didn't activate it (clicking on the sheet would activate the window). We now activate the window regardless of where the window was clicked. (r. 2799575).
- On floating utility windows with no title-bar widgets (i.e. with window attributes set to kWindowNoAttributes), long titles would create a rectangle off the edge of the window (instead of truncating the title). We now correctly truncate titles to the width of the title-bar. (r. 2781092).
- Using SetCursor() from a background application with a utility window had no effect. SetCursor() now works for both foreground and background applications over utility windows. (r. 2779450).
- Previously, adding windows to the standard Window menu required creating the menu before creating any windows. We now automatically add existing windows that have the InWindowMenu attribute to the Window menu when that menu is created. (r. 2754926).
- In some situations involving floating windows, system-generated modal dialogs might appear in incorrect locations (centered on a floating window, generally). We now correctly position modal dialog boxes on the front-most, non-floating window. (r. 2676193).
- If an application is hidden, it was still possible to show a window (using ShowWindow(), e.g.) for that application. We now correctly prevent a window from being shown if an application is hidden. (r. 2494500).

Back to top

Classic Runtime

The Mac OS X Classic Runtime environment provides a Mac OS 9 compatibility runtime environment for legacy applications and software.

- A problem where launching classic could trash the contents of the clipboard has been corrected. Affects Mac OS X 10.1.2 and later. (r. 2835692).
- A number of issues related to transferring the contents of the clipboard to and from the classic environment have been corrected. related to 2763740, 2395383. (r. 2773083).
- Fixed a problem whereby if a KEXT did a programmatic setting of ClassicMustSeize on a USB device in the IORegistry, it could crash the Classic Environment. (r. 2911540).
- Enhancements were made for Classic's recognition of NFS volumes. (r. 2897832).
- GetProcessInformation will now return correct type and creator information for Classic Desk Accessories. (r. 2893023).
- The Classic File Manager now calls through the Carbon File Manager. One of the benefits of this is to allow Classic applications access to WebDAV, DOS FAT, SMB, UFS, etc. volume formats supported by the Carbon File Manager. (r. 2842657) (r. 2884689).
- Classic can now handle DebugStr being called from an MP task. (r. 2866846).
- A problem where AltiVec registers would be trashed by calls to WaitNextEvent has been corrected. (r. 2768550).
- Fixed a problem where a USB composite device - a headset with HID controls - would keep Classic from starting. Besides fixing the specific problem, which was an internal bug in Classic, a change was made to Classic USB policy for MacOS X 10.2 and later. The new policy is that if a USB device has multiple interfaces, some of which are of the class that they might have been opened by Classic to be visible to the emulated MacOS 9 environment, yet others of which are not and thus might successfully match to a Mac OS X driver, then the entire device is not opened by Classic and thus will only function in the Mac OS X environment. In other words, "split cases" where a USB device has some interfaces in use by Classic and some by "X" drivers, will no longer be allowed. For additional information on which devices are seized by Classic, search for the document "Classic USB Device Arbitration" on developer.apple.com and in the USB SDK; it has been updated for Jaguar. (r. 2761572).

- Calling LMGetBootDrive() from classic now returns the correct volume reference number. (r. 2756118).
- It is now possible for classic applications to extract the process serial number from an Apple event they received from a Mac OS X application, save that process serial number, and later use it to send an Apple event to that application. (r. 2392561).
- A problem where Application Died events were not being sent for Classic applications that had unexpectedly quit has been corrected. (r. 2922254).
- When called from a Classic application, PBHGetDirAccess was failing with a paramErr (-50). This has been corrected. (r. 2908703).
- A problem where Gestalt() was returning the wrong system version - 9.2 rather than 9.2.1 - when called in a classic application running in the Classic environment in Mac OS X 10.1 has been corrected. (r. 2771448).
- Better Apple Event error checking and handling has been added to Classic Support's Apple Event handlers. This helps prevent possible crashes inside Classic if an error occurred while an Apple Event was being processed. (r. 2749114).
- A problem where applications in the Classic environment could not connect to localhost tcp servers running in Mac OS X has been fixed. (r. 2831535).
- A problem where PAP spoolers running in the Classic environment could not be seen over AppleTalk by PrintCenter has been fixed. (r. 2720750).

Back to top

Cocoa

A set of object-oriented frameworks that support rapid development and high productivity, using a dynamic runtime and rich object hierarchy. Please refer to file:///Developer/Documentation/ReleaseNotes/AppKit.html on your 10.2 system for detailed information on changes in Cocoa in 10.2.

AppKit

AppKit is a framework that provides graphical objects with default behaviors for rapid application development, including windows, panels, buttons, menus, scrollers, text fields, and more.

- The declaration of NSCopyBitmapFromGState() listed in NSGraphics.h was removed. (r. 2781664).
- NSColor's +colorWithPatternImage method now returns an auto-released object, where before it did not (r.2797049) .

Back to top

AppKitJava

AppKitJava refers to the Java-language-specific portion of AppKit.

- -outlineView:itemForPersistentObject: now has a Java based counterpart named outlineViewItemForPersistentObject. As well, -outlineView:persistentObjectForItem: now has the Java based counterpart outlineViewPersistentObjectForItem. (r. 2837824).
- Cocoa-Java's NSSavePanel didn't have a way to set wether or not the extension is shown. Now, the following APIs can be used to control extension hiding: setCanSelectHiddenExtension, isExtensionHidden, and setExtensionHidden. (r. 2830102).

Back to top

Data Management

Data Management involves the various object classes, including strings, dates and times, collections, numbers, and the undo architecture that Foundation provides.

- NSString's -stringWithContentsOfURL: no longer leaks memory. (r. 2916292).
- NSMutableString's -deleteCharactersInRange: now throws an exception if the range is invalid for the calling

string. (r. 2877676).

- NSMutableString's -setString: method now raises an exception when passed nil. (r. 2860954).
- NSString's -stringByAppendingString method now raises an exception if you pass a nil pointer. (r. 2838156).
- NSString's -hasPrefix: and -hasSuffix: methods now raise exceptions in cases where they would crash before when passed nil. (r. 2835658).
- Out-of -bounds accesses in NSString's -characterAtIndex: and -getCharacters:range: will now cause exceptions (r. 2789384).
- NSMutableString now has a method, -replaceOccurrencesOfString:withString:options:range: to replace all occurrences of a target string with another. It does this in a way that is potentially a lot more efficient than multiple calls to find and replace.

Back to top

Drawing and Imaging

The portion of AppKit devoted to graphics, views, printing, and OpenGL.

- A problem where some black and white GIF's didn't render correctly has been corrected. (r. 2973862).
- Asking for a cached image representation deeper than the screen now returns the requested depth. (r. 2972400).
- Three new alternate selection colors have been made available in Interface Builder. (r. 2956191).
- NSView's -inLiveResize method now always returns the correct value, instead of getting "stuck" at YES. (r. 2950733).
- In a Cocoa window, springs now work correctly when you have a control that you want to maintain a constant width and distance from the right-hand side of the window. (r. 2948689).
- New methods for alternate selection colors are publicly available. (r. 2939444).
- NSEPSImageRep now claims all EPS file types including 'TEXT', EPS, EPI, epsf, epsi, EPSF, and EPSI. (r. 2889834).
- A problem where using [NSPatternColor isEqual] against objects of the same type could throw an exception has been fixed. (r. 2838383).
- Cocoa's NSSplitView now returns proper values from -isSubviewCollapsed: instead of returning false positives under some circumstances. (r. 2620918).
- Cocoa views and controls now move properly in response to springs and struts when their superview is resized, without round off errors creeping in. (r. 2173955).

Back to top

Events and Other Input

The portion of AppKit relating to event handling and user event processing.

- Cocoa applications with dock menu items now pass back the correct sender. (r. 2751274).

Back to top

File, Resource, and Process Management

AppKit classes involved in working with the variety of components needed during program execution, including file management, images, code, and system information.

- The NSFileManager APIs now provide access to file creation dates. (r. 2814464).
- The NSFileManager APIs are now able to report additional file properties based on flags returned by the BSD stat call. For example, the IMMUTABLE bit (uchg), which maps to Finder's "islocked" bit was not available through the NSFileManager APIs prior to this change. (r. 2750685).

Back to top

Foundation

The Foundation framework defines a base layer of Objective-C classes. In addition to providing a set of useful primitive object classes, it provides support for Unicode strings, object persistence, file system functions, scripting, and more.

- The Documents directory was added to the list of NSSearchPathDirectories for use in NSSearchPathForDirectoriesInDomains. (r. 2925120).
- +[NSHost currentHost] no longer crashes if configd isn't running. (r. 2913313).
- NSTasks launched from a non-main thread no longer hangs in -waitUntilExit. (r. 2910194).
- NSHost's +hostWithName: now treats the empty string the same as nil. (r. 2903003).
- If the server dies while the client is waiting for a Distributed Objects reply, the client now gets an exception almost immediately, instead of waiting for a timeout, etc. (r. 2902227).
- A number of problems related to using pthreads (as opposed to threads created with NSThread) have been resolved in AppKit. (r. 2895833).
- For applications compiled against Mac OS X version 10.2, NSCalendarDate's -classForCoder now returns -[self class] instead of +[NSCalendarDate class] so that subclasses don't have to override this method. (r. 2867607).
- NSString's -writeToFile:atomically: no longer fails when using a relative path based in /tmp. (r. 2856590).
- Sending NSData with Distributed Objects no longer leaks for lengths greater than 4095 bytes. (r. 2803364).
- NSRunLoop now implements -description. (r. 2803315).
- NSClassFromString() and NSStringFromClass() no longer crash when passed nil; they now return nil instead. (r. 2802590).
- -[NSMutableArray removeObjectsFromIndices:numIndices:] no longer raises spurious exceptions. (r. 2800991).
- [[NSHost currentHost] addresses] now returns a complete list of IP addresses when connected through DHCP, instead of only 127.0.0.1 - localhost. (r. 2665193).
- NSInvocation no longer coerces negative values for char or short arguments to unsigned (positive) values. (r. 2545453).
- [[NSProcessInfo processInfo] arguments] no longer fails when one of the argument strings contains high-ASCII characters. (r. 2474167).
- [NSThread description] now returns useful information about the thread in question, not the currently running thread. (r. 2309712).
- NSTask now generates a more complete error message if it cannot exec its task. (r. 2205786).
- -[NSSocketPort scheduleInRunLoop:forMode:] could exit a critical region without unlocking its lock under some conditions. This has been fixed. (r. 2983123).
- NSMethodInvocation no longer gets confused in Distributed Objects calls to methods that have long long int arguments. (r. 2940584).
- NSHost's +currentHost method no longer initiates a dialup connection. (r. 2914589).
- -[NSNumberFormatter initWithCoder:] no longer leaks two NSDecimalNumbers. (r. 2884200).
- +[NSCalendarDate dateWithString:] now correctly returns an NSCalendarDate object. (r. 2874746).
- -[NSDate isEqualToDate:] now raises an exception if you compare your date with nil. (r. 2873711).
- A memory leak in NSCountedSet has been corrected. (r. 2853676).
- Applications no longer crash when mapping more than 4GB of data into memory using NSData. (r. 2810307).
- -[NSMutableArray replaceObjectsInRange:withObjectsFromArray:range:] now works even if the other array is equal to the receiver. (r. 2803598).
- -[NSDictionary description] will no longer raise an exception if the dictionary's keys are of different classes, as it would in some cases before. (r. 2699692).
- NSNumber/CFNumber now work better together when doing comparisons and equality tests. (r. 2546052).

Back to top

FoundationJava

The Java-language-specific portion of the Foundation Cocoa framework.

- None of the NSFile* constants, like NSFileHFSCreatorCode, NSFileHFSTypeCode, NSFileExtensionHidden, etc.

existed on the Java side in Cocoa-Java AppKit; they have now been added. (r. 2830098).

- NSScriptObjectSpecifier had a Misspelled field with the name InvlaidIndexSpecifierError. This field has been renamed as InvalidIndexSpecifierError. (r. 2647563).

Back to top

Interapplication Communication

The portion of Cocoa providing functionality in the areas of distributed objects, system services, OS services, networking, and rendezvous technologies.

- A problem where dragging images to the NSPasteboard could leak some data into a private system memory area has been corrected. (r. 2921843).
- A problem in _NSLookupPBS() where spurious NSPortTimeoutException exceptions could be generated when it was called from an application running as part of the login sequence has been corrected. (r. 2863180).
- Moving a file to the trash with NSWorkspace now updates the Finder and succeeds even if a file with the same name is already in the trash. (r. 2853822).
- NSWorkspaceDidTerminateApplicationNotification is now sent even if the app that is terminating was already running when you started to observe the notification. (r. 2841095).
- Carbon 'url ' drag flavor is now mapped to a NSURLPboardType. When both a 'furl' and 'url ' are present in a drag (which are both convertible to NSURLPboardType) AppKit will now choose to convert the 'furl' first. (r. 2794628).
- NSWorkspace now returns correct application names in notifications when the application names contain "." or other special characters. (r. 2671677).
- A problem where large cut and paste operations of formatted text from Cocoa applications could lose style information has been corrected. (r. 2630845).
- A problem where NSWindow's dragImage was sometimes being drawn in the wrong location has been corrected. (r. 2796124).
- A problem where private drags initiated from Cocoa applications were being passed through to carbon applications has been corrected. (r. 2627542).

Back to top

Multimedia

The portion of AppKit providing sound and video support.

- An application with an NSMovieView with autoresizing turned on would sometimes display garbage or black bars over the controller during window resizing. We now correctly redraw the region during live resize. (r. 2784602).

Back to top

Services (AppKit)

Services give applications an open-ended way to extend each other's functionality by allowing applications to provide services to other applications, and access functionality provided by other applications.

- Third-party spell checkers now launch automatically (r. 2766130).
- Non-packaged Carbon applications can now provide Services. (r. 2761555).

Back to top

Text Handling

One of the portions of AppKit dealing with text.

- Support for bi-directional layout added to support those languages that require it (Arabic and Hebrew). (r. 2822201).

- Implementation of NSRightTabStopType, NSCenterTabStopType, and NSDecimalTabStopType have been provided for the declarations of these routines in AppKit/NSParagraphStyle.h. (r. 2116892).
- A problem where NSAttributedString's -size method would fail to take into account the height of fonts occurring later in the string has been corrected. (r. 2824091).
- Fixed a bug where NSString sizeWithAttributes could return imprecise values when used with NSCenterTextAlignment. (r. 2727150).
- Cocoa text now uses the same line layout engine used by ATSUI, leading to greater consistency between Cocoa and Carbon, as well as support for complex writing systems. (r. 2570275).

Back to top

Threading Support (AppKit)

Threading support in AppKit.

- NSWindows can now be created in threads other than the main thread. (r. 2887016).

Back to top

User Interface Elements

AppKit provides a rich selection of controls and user interface elements, and the classes used to control them.

- NSTextViews that can't import graphics are now able to export them (for copy/paste, etc.), i.e. if they contain attachments added programmatically. (r. 2979744).
- NSProgressIndicator's -sizeToFit method now respects the NSControlSize setting for the control. (r. 2963039).
- setImage: is now declared in NSBrowserCell.h. (r. 2918420).
- NSTableView's -setIndicatorImage: now retains the image passed to it, so that if it's given an autoreleased image (with a retain count of 1), a crash won't occur. (r. 2891170).
- In some situations, NSTableView's -selectedRow method used to return 0 instead of -1 when there were no rows in the table. This has been fixed. (r. 2889283).
- Cocoa-Java's NSTableView class now has methods for setting and getting the indicator image for a table column, as well as highlighting a given table column. (r. 2885558).
- Drawing problems associated with drawers and deferred windows have been corrected. Drawers were not being displayed as expected. The work around for this problem, that is no longer needed, was to tell the drawer to close and open again after the deferred window is visible. (r. 2884722).
- Users can now use cmd-shift-` to send keyboard focus to a drawer. Previously, it was only possible to do so by clicking on a drawer. (r. 2872115).
- Carbon windows (from Java or QuickTime) in a Cocoa app now get events properly. (r. 2865112).
- NSDrawers now attach properly to NSWindows created from Carbon windows (via -initWithWindowRef:). (r. 2860068).
- an empty NSTableView's -selectedRowEnumerator used to return an enumerator with 1 element. This has been fixed. It now correctly returns an 'empty' enumerator object. (r. 2854415).
- NSTextView's -selectionRangeForProposedRange:granularity: now returns the last word instead of raising an exception when the granularity is NSSelectByWord and the range's location is equal to the length of the underlying string and the length is 0 (i.e., an insertion point at the end). (r. 2845418).
- A bug in Cocoa applications where the parent window did not always become key when its sheet was dismissed has been fixed. (r. 2800050).
- Carbon WindowRefs for Cocoa NSWindows (created using -initWithWindowRef:) now have their regions maintained correctly as the Cocoa window resizes and moves on the screen. (r. 2769632).
- NSOutlineView has had its row limit of 32767 children in an item removed. (r. 2761925).
- A problem where a drawer associated with a utility window overlapping another utility window could be displayed with the wrong window has been corrected (r. 2659187).
- Tooltip rects are now clipped to the visible frame of their view's parent scroll view. (r. 2565501).
- Calling NSButton setKeyEquivalent will now cause the default button to be displayed in the "default" key manner. (r. 2450365).

- In some situations, NSTableView used to incorrectly call the willDisplayCell: delegate method before setting the cells object value with setObjectValue:. This has been fixed. NSTableView should now always set the object value of the cell first before sending it's delegate the willDisplayCell: delegate message. (r. 2362114).

- In some cases, AppKit did not send a mouseExited event the first time the mouse was moved outside a tracking rect. This problem has been corrected. (r. 2480290).

Back to top

Core Foundation

Core Foundation is a set of APIs used as a common data and service abstraction layer for other high level software facilities in Mac OS X. Please refer to CoreFoundation release notes (file:///Developer/Documentation/ReleaseNotes/CoreFoundation.html on your 10.2 system) for more details.

CFPreferences

Preference Services allows you to store values that are associated with a key that can later be used to look up the preference value when you need it. Key/value pairs are assigned a scope using a combination of user name, application ID, and host (computer) name.

- In order to better support operation in a managed environment, a new API has been added to CFPreferences: Boolean CFPreferencesAppValueIsForced(CFStringRef key, CFStringRef applicationID); Developers may call this to determine the user's ability to change the value for a given preference key. If a key is "forced" then developers should disable UI appropriate to changing the value. Previously, there was no mechanism to determine if a value was forced for a particular key. (r. 2889315).

Back to top

CFStream

CFReadStream and CFWriteStream are abstractions of one-way byte streams that are used much as file descriptors are used in POSIX. They can be created to or from files, sockets, or memory; CFNetwork defines other useful kinds of streams.

- Async notification of dynamic store changes has been added, meaning that an event-based socket stream that has a link go dead will receive an error event indicating a dead connection. All socket streams by default are opted into the auto-erroring. (r. 2705758).

- Property names and values have been added to CFNetwork/CFSocketStream.h for manipulating SOCKS proxies. These include support for SOCKS4 and SOCKS5. (r. 2705566).

- CFReadStreams and CFWriteStreams created to/from a host no longer block in open looking up the host's IP address; instead, the lookup is done asynchronously, and the call to CFReadStreamOpen (or CFWriteStreamOpen) returns immediately. (r. 2683774).

- Socket reading I/O performance has been improved. (r. 2938957).

Back to top

CFString

String Services is an API that provides a suite of efficient string-manipulation and string-conversion routines.

- A problem where CFStringGetBytes would ignore its range parameter when asked to retrieve a range of bytes from a Unicode (kCFStringEncodingUnicode) string has been corrected. (r. 2812392).

- CFStringCreateWithCString now does better error checking and will return NULL if passed an improperly encoded UTF-8 C string. For example, if you provide a C style string and indicate that the format of the string is kCFStringEncodingUTF8 but the string contains non-UTF-8 characters, then CFStringCreateWithCString will return NULL. (r. 2980386).

Back to top

CFXMLParser

Core Foundation's XML parser for reading XML documents.

- CFXMLParser now handles UTF-8 documents with a byte order mark (BOM). (r. 2826988).
- CFXMLTreeCreateXMLData would sometimes put extra spaces in the data when using CFXMLTreeCreateFromData and/or CFXMLTreeCreateXMLData. This problem was fixed. (r. 2453608).
- The XML parser didn't properly handle real numbers +Inf, -Inf and NaN in plists. We now correctly handle such numbers. (r. 2740510).

Back to top

Core Foundation (general)

Core Foundation is a set of APIs used as a common data and service abstraction layer for other high level software facilities in Mac OS X.

- Files.h is no longer included in CoreFoundation.h or CFURL.h. Now, if you were including CoreFoundation.h and you used any of the declarations in Files.h, you will need to explicitly #include <Files.h>. (r. 2868727).
- There are several changes to CFNumbers handling of NaN and infinities: (a) CFEqual() would return false on two different NaN CFNumbers as the parameters, but return true with the same NaN CFNumber as both parameters.: NaNs might be logically equal, but not mathematically equal. To eliminate this inconsistency, we make CFNumbers with NaNs equal to each other. This change will only affect applications linked on Jaguar or later. (b)Added detail to the debug description of CFNumbers. We now include more precision; distinguish zero from negative zero; always include sign on values; and more). This change will affect all applications. (c) Changed the formatting description produced by CFNumber to: include more precision on doubles and floats; report special values as "nan", "+infinity", and "-infinity"; remove trailing zeros on the value; and report small values (under 1.0x10^-3) in scientific notation. This change will only affect applications linked on Jaguar or later. (d) The value of CFNumberCreate() with float and double values is checked against NaN and the infinities, and the global value is returned if the value matches. This change will affect all applications. (e) CFNumberCompare() has been corrected to match the documentation in the header comment regarding NaN, infinities and zeros. This change will only affect applications linked on Jaguar or later. (r. 2867452).
- Using CFPropertyListCreateXML to generate XML from a CFDictionaryRef would cause a crash if any dictionary key was something other than a CFString. We no longer crash in this situation. (r. 2830305).
- Using CFStringFindWithOptions() to check for "Ends With" would sometimes return false positives. We now return correct data in all cases. (r. 2813801).
- In some Carbon applications, a slash ("/") in a path name would crash when loading a nib file. This on longer occurs. (r. 2787552).
- In rare cases where an XML file size was exactly a multiple of a page size, calling CFPropertyListCreateFromXMLData with that XML file would cause an application to crash. We no longer crash in this situation. (r. 2765819).
- The error messages returned by CFPropertyListCreateFromXMLData have been improved. (r. 2722422).
- Removing a CFTree created using CFTreeInsertSibling would inadvertently release more memory than was necessary, possibly leading to an application crash. We now correctly release memory in this situation.(r.2747173). (r. 2387606).
- CFLog messages are now time stamped. (r. 2956225).
- The new Core Foundation data type CFNull has been added for symmetry with the Cocoa NSNull data type. Now NULL (nonexistant) values can be represented in CFDictionaries. (r. 2828586).
- The CFPropertyList parser has been updated to properly treat """ as a double quote. (r. 2738226).
- CFMutableDataRef data pointers are now 16 byte aligned. (r. 2701601).
- CFTreeSetContext appeared in the headers but was not implemented. We now implement this call. (r. 2570830). .
- Please refer to CoreFoundation release notes (file:///Developer/Documentation/ReleaseNotes/CoreFoundation.html on your 10.2 system) for a lot more information.
- Under Mac OS X 10.2, the function CFReadStreamCreateForStreamedHTTPRequest will not work if the requestBody argument is non-NULL; instead, it ignores the requestBody provided and always sends a zero-length request.

Back to top

Core OS

The Core OS incorporates essential services that are generally devoid of a user interface. These facilities range from memory management to process management.

- Kernel panic information is now recorded in NVRAM and at the next system startup that information is written to the /Library/Logs/panic.log. This allows panics to be reported without the inconvenience of having to write down the information or copy the information from the screen by some other means. (r. 2795225).
- Panic Text now displays the HW and IP Addr. (r. 2774419).
- The CALL_EXTERN_AGAIN macro grew the stack by 56 bytes, which was not 16 byte aligned. This has been fixed. (r. 2226272).
- Multiple TRUE/FALSE definitions no longer invalidate precompiled headers. (r. 2219860).
- All storage support (SCSI) is now being done through the IOSCSIArchitectureModelFamily and relevant pieces. Thus the IOSCSIDrive.kext and other similar kexts were removed from the system. (r. 2824289).
- CGSGetNextEventRecord would at times return a garbage event type. That was the event type values returned by this API does not match the actual event. This problem was resolved. (r. 2782314).

Back to top

CUPS

The Common Unix Printing System.

- CUPS has been integrated into Mac OS X. (r. 2849589).

Back to top

CVS

The Concurrent Versions System.

- The CVS command line tool now supports the -R option, which allows you to check out from a read-only repository (such as a CD-ROM). (r. 2919717).
- CVS's unwrap facility now updates the modification time of untar'd files. (r. 2938582).

Back to top

Disk Arbitration

The DiskArbitration Server tracks and provides notification services announcing new disks.

- The Disk Arbitration routine DiskArb_EjectKeyPressed has now been deprecated. The framework still exports the function, but it does nothing. This function is in an Apple-private framework and third party developers should not be calling it. (r. 2880342).
- Supplemented the "disktool" command line tool (which was always intended as a test tool) with a more user-oriented "diskutil" command line tool. (r. 2817377).
- The "disktool" command line utility no longer remounts all partitions on a drive when you request a particular partition (for example, "disktool -m disk1s12"). (r. 2817376).
- Disk Arbitration now wakes up the monitor when the user inserts a disk. (r. 2788175).
- The disktool command line utility now quits after mounting volumes with the "-m" option. (r. 2780308).
- Disk Arbitration now respects information in "/etc/fstab". See the man page, autodiskmount(8), for details. (r. 2383037).

Back to top

Dock

The Dock provides a convenient graphical interface for launching applications and organizing frequently used items.

- A problem where an application's dock tile image was not being set to its original image when the application quit (after it was set to a custom image by a call to SetApplicationDockTileImage) has been corrected. Now, the image is

restored to its original image as expected. (r. 2738381).

Back to top

File Systems

This section discusses changes and improvements in the file systems installed with Mac OS X 10.2.

AFP

Apple's implementation of AFP (AppleTalk Filing Protocol) over TCP server that allows AppleShare clients to connect and access files.

- Improved the security of the password changing mechanism. (r. 2818447).
- Previously, if you connected to the server using the admistrator's long name, you would not see the user's home directory. This has been fixed. (r. 2800012).
- Revised the AFP protocol to support more than 16 bits worth of items in a folder. (r. 2757392).
- Due to problems displaying UTF-8 messages (such as disconnect messages) on the client, the server now only sends messages in old-style script encodings. (r. 2971846).
- Personal File Sharing now supports the ability to turn off guest access via NetInfo--there is no user interface for this. (r. 2949059).
- Fixed problem where the AFP client was always using the ByteRangeLock call instead of ByteRangeLockExt even on AFP 3.0 or newer servers. (r. 2936968).
- Copying a file from a Mac OS X AFP server to a Mac OS 9.x AFP server while the current time zone is observing Daylight Saving Time could cause the destination copy to have its modification time set off by one hour. This could happen if the file being copied had a modification time when DST was not in effect. This problem has now been fixed. (r. 2892452).
- mount_afp no longer requires a password embedded on the command line. Instead, it accepts an afp:// URL and prompts separately for the password. (r. 2886847).
- Fixed a problem where PBGetCatInfo could return the wrong number of files in the root directory of an AFP volume. (r. 2883190).
- The modification time of a file on an AFP server was reported relative to the clock on the client system. This has been changed to use the time reported on the server. This prevents the modification time from changing simply because of clock skew between the client and server. (r. 2877917).
- Fixed a problem where certain folders could not be copied from a Mac OS 9 AFP server to a client Mac OS X system with the primary language set to Simplified Chinese. (r. 2872906).
- The API function AFPSortSharedVolumes was not being properly exported, causing link errors if that function was referenced. (r. 2858101).
- The AFP client now logs various events to the system log file in addition to providing user notification of the events. These events are: umount failures, idle disconnects, reconnect attempts, and whether session will show true vs. mapped permissions. (r. 2848604).
- Fixed a problem where FSGetCatalogInfoBulk would return incorrect permissions on mounted AppleShare volumes while FSGetCatalogInfo and lstat would return the correct values. (r. 2832434).
- The AFP server now handles non-ASCII characters in file names better. (r. 2803650).
- The AFP server now correctly handles UFS boot volumes with traditional Mac OS clients. (r. 2797781).
- Fixed a problem where directory enumeration using FSGetCatalogInfoBulk could return extra non-existent files when called to enumerate files on an AFP volume. (r. 2795825).
- Doing a CatSearch on the AFP server no longer blocks other clients from doing work for long periods of time. (r. 2691585).
- The AFP 3.0 protocol was extended to require a UTF-8 server name in addition to the local encoded server name be sent in the serverinfo packet. (r. 2675317).

Back to top

AFP Server

Implements an AFP (AppleTalk Filing Protocol) over TCP server that allows AppleShare clients to connect and access files.

- The AFP server now handles file EOF better, working around another bug (r. 2905716). The upshot is that you can now use "vi" to edit files on an AFP volume. (r. 2903677).
- In the AFP server, we now correctly recognize certain AFP commands as user activity. This prevents an idle timeout when copying large files up to the server. (r. 2898895).
- The AFP server now handles large (greater than the old AppleTalk maximum of approximately 576 bytes) command packets. Previously the only large command packets that were handled correctly were writes. (r. 2897405).
- The AFP server now correctly handles the SIGTERM signal sent to the parent (all of its children terminate and then it terminates) and the SIGHUP signal sent to the child (the child re-reads its preferences). (r. 2885293).
- Searching for files by date on the AFP server now works correctly. Previously it would return seemingly random results. (r. 2878165).
- You can no longer share two different share points (AFP volumes) whose names differ only in case. (r. 2870015).
- The AFP server now has the ability to deny guest access to a share point, even if the contents of the share point would otherwise be accessible by guest users. (r. 2865398).
- The server now supports the AFP 3.1 protocol extension that allows UNIX-to-UNIX implementations of AFP to not translate user and group IDs. See the discussion of kNoNetworkUserIDs in the AFP 3.1 protocol specification for more details. (r. 2848567).
- By default the server no longer allows clients logged in as an admin user to access files that are only readable by root. (r. 2838703).
- Previously a bug in the server caused it to disconnect sleeping clients. This has been rectified. (r. 2835079).
- Changed the server to allow clients to create symlinks properly. (r. 2827876).
- If the AFP server is on and you insert an audio CD, the CD now appears in the server's list of share points. (r. 2796852).
- By default the AFP server now allows the client to reconnect if they were disconnected on idle or server shut down. (r. 2793339).
- The AFP server now behaves better if the user unmounts a volume that is in use by one of its clients. (r. 2768585).
- The AFP server now respects the Max Reply Size field of the FPEnumerateExt request. (r. 2739396).
- The AFP server's protocol version string is now "AFP3.1". (r. 2651749).
- The AFP server now works as expected when booted from UFS. (r. 2625961).
- The AFP server now correctly handles disk quotas on the volumes it is serving. Operations exceeding the disk quota will be fail with kFPDiskQuotaExceeded (-5047). (r. 2976212).
- The AFP server no longer lets you set the EOF of a file if growing the file would exceed the free disk space. (r. 2969704).
- When operating in its default permissions model, the AFP server no longer allows non-root clients to set the ownership of items to root. (r. 2967478).
- The AFP server now implements the DHX2 UAM. (r. 2933358).
- The AFP server now correctly handles range locks beyond the EOF. (r. 2922663).
- The server now works correctly with paths longer than 128 characters. Previously a sign extension bug made it impossible to access files buried deep on the server. (r. 2895343).
- The owner of a file on an AFP server previously could call chown on a file and change the file's owner without first gaining root privileges. This is no longer allowed. (r. 2866129).
- The AFP server now correctly sets the ownership of resource fork files when it's serving from UFS. Previously these files were owned by root, which made it impossible for the user to access them when logged into the server directly. (r. 2820795).
- The AFP server now supports Kerberos authentication.(r. 2850300). .
- The AFP server now supports byte range locking by calling through to the underlying BSD byte range lock support. Along with other changes in Mac OS X 10.2, this significantly improves support for exclusive file access. .
- Changes to the AFP server to support empty passwords better. (r. 2960930).

Back to top

HFS

The Hierarchical File System.

- A problem where certain erroneous calls to getattrlist could cause a kernel panic has been corrected. (r. 2823972).
- FSGetCatalogInfo was returning inconsistent results. Some files that were not locked were showing up as locked. The list of locked files would also vary between runs. This problem was fixed. (r. 2797739).
- A problem where PBCatSearch could fail to find some newly created file system items if called immediately after their creation has been corrected. (r. 2941532).
- A problem where dates were not being adjusted correctly for daylight savings time on HFS standard formatted volumes has been corrected. (r. 2884313).
- Previously if any space is allocated to a file on an HFS/HFS+ volume but no data is written to the newly allocated space, this space is not reclaimed as it ordinarily would be at the time of the close() call. This problem was fixed and now there is no space leakage if a file is allocated and then closed without adding any data. (r. 2760998).

Back to top

MS-DOS

Volume format used by MS-DOS and Microsoft Windows.

- Fixed a bug where ejecting a MS-DOS volume without unmounting it first could hang the system. (r. 2894402).
- Corrected problems preventing Mac OS X from recognizing some FAT32 FireWire hard drives. (r. 2893955).
- A problem where fsck_msdos did not properly handle devices with sector sizes greater than 512 bytes has been corrected. (r. 2883078).
- Fixed a problem with certain invalid long names on MS-DOS disks. The problem happened when long names were a multiple of 13 characters and contained an extra (invalid) entry with only a NULL terminator and padding. (r. 2865792).
- A problem where FSpMakeFSRef() would fail for some Japanese file names on MS-DOS formatted volumes has been corrected. (r. 2765643).
- A problem where FSCreateFileUnicode() could fail when asked to create a file with a nine character file name on a MS-DOS formatted volume has been corrected. (r. 2754174).
- Problems preventing FSpSetFLock from working as expected on MS-DOS formatted volumes have been corrected. In the past, FSpSetFLock would return noErr without actually locking the file. (r. 2759105).

Back to top

NFS

Network Filesystem, originally developed by Sun Microsystems, used in UNIX environments for sharing volumes over TCP/IP networks.

- A number of performance and stability enhancements have been incorporated into the implementation of NFS provided with Mac OS X 10.2 (r. 2465832) (r. 2827760).
- Problems preventing the use of mmap() for memory mapped files over NFS have been corrected. (r. 2472566).

Back to top

SMB

Server Message Block, Microsoft's file sharing protocol for Windows.

- Socket read size was reduced from the full packet size to improve performance with remote clients (such as Windows, Linux and BSD). (r. 2917352).
- SMB now ignores windows shares ending in a dollar sign (e.g. c$) as these are administrative shares not intended for ordinary use. This is also windows client behavior for non-administrators. (r. 2899088).
- Problems that could occur when reading more than 128K at a time from a file on a SMB share have been corrected.

(r. 2797032).

- A problem where the NSL browser could fail with SMB servers having names with no corresponding DNS host name has been corrected. (r. 2787668).
- SMB browsing is only supported on the local subnet. .

Back to top

UDF

UDF is a file system plug-in that allows access to Universal Disk Format volumes. This format is used for consumer DVD media and can also be used as a multi-platform disk format.

- Corrected a problem with UDF's conversion of on-disk ISO Latin-1 names to Unicode. The symptom was that a file with ISO Latin-1 characters in its name would not show up properly (or at all). (r. 2835582).
- UDF now supports read/write access to UDF formatted DVD-RAM disks (r. 2477531).
- The UDF file system is now capable of handling files larger than 2 GB. (r. 2944397).
- UDF not returns the VOL_CAP_FMT_PERSISTENTOBJECTIDS flag from getattrlist appropriately, depending on the UDF volume format. UDF formats prior to 2.0 didn't support persistent object IDs. This problem was causing aliases to UDF volumes to not work correctly. (r. 2762832).

Back to top

UFS

The Unix File System.

- A problem where the disk cache for UFS volumes was not being flushed before shutdown or restart has been corrected. (r. 2908945).

Back to top

WebDAV FS

WebDAV stands for "Web-based Distributed Authoring and Versioning". It is a set of extensions to the HTTP protocol which allows users to collaboratively edit and manage files on remote web servers.

- Problems using WebDAV FS to mount a volume on a port other than the default http port (port = 80) have been corrected. (r. 2777825). .
- Problems with DNS lookups via Airport have been corrected (r. 2780350). .
- Interoperability with non-iDisk WebDAV servers has been improved. HTTP header lines up to 4096 characters (1024 characters in 10.1.5) are now accepted (rr. 2788583, 2944883). Chunked Transfer Coding is now handled correctly in all known cases (r. 2868057, 2872308). Digest challenges with zero-length domain directives are now accepted (r. 2804676). Connection headers with the close token are now handled (2921033). DAV headers with extend text are now correctly parsed (r. 2891691). All Informational 1xx HTTP status codes are now handled (r. 2951626). .
- Performance has been improved. WebDAV FS sends fewer requests to the server (r. 2858454, 2919924). Latency of the first read or write after opening a file has been reduced (r. 2864082, 2879261). WebDAV FS's use of sockets has been optimized (r. 2929972, 2919564). WebDAV FS now asks the server for only the properties it needs which reduces the amount of data the server has to return (r. 2937458). WebDAV FS uses the buffer cache more effeciently (r. 2895257, 2853856). The use of temporary files for most requests has been eliminated (r. 2944883). .
- Memory leaks have been eliminated and crashing bugs have been fixed (rr. 2874995, 2879528, 2806688, 2958029, 2963540, 2771229, 2989544, 3000493, 3002584). .
- WebDAV FS's version is now included in its User-Agent request-header field so servers can identify specific versions of WebDAV FS (rr. 2797472, 2891668, 2881656). .
- The directory link count of directories on WebDAV volumes is now set to 1 indicating that WebDAV FS does not support link counts. This allows the find command (and other FTS(3) based software) to work on WebDAV volumes (r. 2546611). .
- Access to URLs longer than 255 characters, and up to 1024 characters beyond the mount URL, are now supported (r. 2755598). .

- The same URL cannot be used to mount multiple copies of a WebDAV volume (r. 2884337). .
- The WebDAV authentication dialog now shows both the URL and the realm string making it easier for the user to know why they are being asked for authentication information (r. 2788480). .
- WebDAV FS now warns the user if a server asking for authentication only supports the non-secure Basic authentication scheme (r. 2904390). .
- When a WebDAV server cannot be reached, the timeout before an error is returned has been reduced from up to 10 minutes to 2 minutes maximum (r. 2517387). .
- The source code for the webdavfs project has been cleaned up. All warnings from gcc3 and kextload validation failures have been eliminated (r. 2911492, 2917367, 2796114, 2919408). .

Back to top

Graphics

Routines for drawing and displaying graphical information on the screen are discussed in this section.

ATS (Apple Type Services)

ATS provides APIs that make it possible for Mac OS applications to manage and access fonts, and draw glyphs to the screen.

- The new routine GetFSRefForFileToken can be used to retrieve a FSRef from a fileToken. (r. 2894186).
- Fixed a problem that could cause FontMetrics to crash when used with some older Mac OS 9 suitcase fonts. (r. 2871889).
- A problem where the Kotoeri character palette could not represent Gaiji characters (SJIS: over 0xF040, Unicode: over u+E000) has been corrected. (r. 2831210).
- True Type font files with the extension '.ttf' are now recognized in Mac OS X. (r. 2823850).
- Problems where the routines ATSGlyphGetIdealMetrics and ATSGlyphGetScreenMetrics would return incorrect error codes have been corrected. (r. 2791182).
- Already created instances of Multiple Master fonts are now supported. The creation of new instances is not supported and requires the use of products from Adobe under Classic.(r. 2618745) (r. 2739365).
- Fixed a crash when the invalid value NULL was passed as the text encoding parameter to FMGetFontFamilyTextEncoding. (r. 2677209).
- Added hooks for 3rd party font activation. See "Font query message hooks" in ATSFont.h for more information. (r. 2667733).
- Calling ATSFontFamilyGetName with the name parameter set to NULL now returns paramErr instead of crashing. (r. 2621484).
- ATSFontGetFileSpecification now returns paramErr if the oFile parameter is NULL. (r. 2610513).
- System-critical fonts can no longer be deactivated by non-system Font Manager clients. (r. 2599690).
- Finding all the fonts within a particular domain (using FMCreateFontFamilyIterator or FMCreateFontIterator, e.g.) would fail if fonts were in nested directories. We now properly iterate though all subdirectories by default. (r. 2897297).
- In some instances, the directory returned from FMGetFontTableDirectory was an incorrect font table for LWFN fonts. We now indicate there are no entries in the directory in these cases. (r. 2848718).
- Creating or using an OTF(CID) format font which contains more than 32676 glyphs would sometimes crash an application. We now allow over 32676 glyphs. (r. 2821918).
- In rare cases, ATSFontFindFromName() returns no matches, even when using the CFStringRef returned by ATSFontGetName(). We now correctly return all fonts. (r. 2764305).

Back to top

ATSUI (Apple Type Services for Unicode Imaging)

ATSUI is core technology for Unicode text drawing and editing on OS X. It is fully multilingual and provides basic services as well as high-end typographical control. It can be used with Quartz or Quickdraw based applications.

- kATSLineDisableAllBidirectionalReordering has been removed and kATSLineDisableAllCharacterMorphing has been renamed to kATSLineDisableAllGlyphMorphing. (r. 2925657).
- The original API ATSUMeasureText will still function fine though each process that calls it will get a single syslog stating that it is deprecated and that they should use ATSUGetUnjustifiedBounds instead. (r. 2918417).
- ATSUI will now display the specified glyph when kATSUGlyphInfoTag attribute exists and if the font is not consistent with the glyph attribute, it will ignore the glyph attribute. (r. 2883308).
- Developers now have direct access to the ATSUI glyph rec array and are able to manage the steps of line layout directly, altering bidi, glyph metamorphosis, and justification as needed. (r. 2853741).
- Multi-run ATSUTextLayout objects now transfer styles into the ATSUGlyphInfoArray correctly. (r. 2850959).
- A new API ATSUHighlightInactiveText is now available. This API allows callers to have better control over the appearance of selected text as it is drawn when the window containing the text is inactive. (r. 2842617).
- ATSUStyles can now have their ascent and descent values changed using the kATSUAscentTag, kATSUDescentTag, and additionally the leading can also be set using the kATSULeadingTag. (r. 2804497).
- ATSUI now has layout control facilities to override user anti-aliasing preference. To control the anti-aliasing for an entire layout or line you can use one of the Layout options: kATSLineApplyAntiAliasing or kATSLineNoAntiAliasing. (r. 2754392).
- The style's leading is now retrieved from the ATSUStyle using the ATSUGetAttribute with the kATSULeadingTag. If the ATSUI client did not set the leading (which can now be specified using this tag), the ATSUGetAttribute will return kATSUNotSetErr and the font's leading value. (r. 2621797).
- Added support for RGB-Alpha using the CGColorSpaceRef in ATSUStyle records. (r. 2951269).
- A problem where ATSUClearLayoutControls was not returning the error code kATSUInvalidAttributeTagErr in cases where it should have has been corrected. (r. 2927064).
- The new kATSLineUseDeviceMetrics option has been added to allow layout and line breaking in device space. (r. 2917413).
- The kATSUFontMatrixTag style transform matrix is now implemented. (r. 2911269).

Back to top

ColorSync

ColorSync provides system-level color management that enables publishing software to achieve repeatable and consistent color on-screen, in print, and for electronic delivery.

- When update a procedure based profile, the procedure profile proc is called with the cmWriteAccess selector a number of times to write out the data from the profile. If the function that handles the cmWriteAccess (or cmWriteSpool) command cannot write out all the bytes requested, then the procedure profile proc is called again with the cmWriteAccess (or cmWriteSpool) selector to write out the remaining bytes. There was a problem in the update code where the calls to write additional bytes after the first call would always be passed the address of the beginning of the buffer rather than the beginning of the bytes needing to be written. This problem has been corrected.(r. 2892979) (r. 2928633).
- The NCMUnflattenProfile call will now unflatten the data into a profile correctly if the number of bytes that are sent back by the call back proc are less than the number of bytes requested by the call back proc. (r. 2910239).
- Fixed bug where calling CMSetProfileElement to set tags that are obtained from an existing profile into a newly created profile would result in error -50 for certain tags from certain profiles. (r. 2906498).
- NCWNewLinkProfile no longer returns error -50 when used with big (>100kb) profiles. (r. 2897811).
- Fixed bug where no error is returned for attempts to register a profile for a device at a scope outside the user's domain. (r. 2890764).
- Fixed a bug where callback proc of CMIterateColorDevices was not being called for registered devices of different scopes. (r. 2858107).
- Fixed bug where a display device would not be returned by CMIterateColorDevices unless it was registered with the scope of kCFPreferencesAnyUser, kCFPreferencesCurrentHost. Now CMIterateColorDevices will return display devices registered with the scope of kCFPreferencesCurrentUser, kCFPreferencesCurrentHost. (r. 2858100).
- A problem where CMIterateColorDevices would always return the last device has been corrected. Now, CMIterateColorDevices will return the correct device if more than one device is registered. (r. 2858094).
- Fixed bug where CMSetDeviceProfiles returns a paremeter error (paramErr = -50) for a device registered with the scope of kCFPreferencesCurrentUser or kCFPreferencesCurrentHost that has two registered factory profiles. (r. 2858093).
- Fixed NCMUnflattenProfile so that it will no longer return error code -4205 when asked to unflatten a profile to a

file based profile. (r. 2858082).

- A problem where the count field of the CMDeviceProfileArray was not being reset to zero by calls to CMGetDeviceFactoryProfiles for devices containing no profiles has been corrected. (r. 2858075).
- Notifications of type kCMDeviceRegisteredNotification now occur when calls to CMRegisterColorDevice are made to register a new device. (r. 2858059).
- A problem in CMUpdateProfile where it could return the error code -17w when asked to update a procedure based profile has been corrected. (r. 2857996).
- Fixed a bug in CMCopyProfile where it could return the error -48 when asked to copy a path based profile to a network volume. (r. 2857909).
- A problem where attempting to create a new profile on a network volume could fail and return the result code -48 has been corrected. (r. 2857277).
- A problem where under some circumstances calling CMCopyProfile to copy a profile from one file to another could result in an invalid profile has been corrected. (r. 2855010).
- CMUpdateProfile no longer returns error -4205 when asked to update a buffer based profile. (r. 2854781).
- A problem where CMIterateColorDevices would return the error code -4227 when a NULL callback procedure pointer was provided has been corrected. (r. 2854778).
- Calls to CMSetDeviceDefaultProfileID specifying the id of a profile that is not registered will now return a parameter error (paramErr = -50). (r. 2838339).
- Fixed a crash that could occur when matching colors to a bitmap or matching colors to a Lab profile. (r. 2772363).
- Correct dictionary of names now passed in to callback of CMIterateDeviceProfiles. (r. 2768385).
- The dictionary of names passed in to the profileNames field of the CMDeviceProfileInfo parameter of the CMIterateDeviceProfiles callback proc are no longer set to NULL when calling the CMIterateDeviceProfiles call with flags set to cmIterateCustomDeviceProfiles. (r. 2768351).
- The routine CMIterateDeviceProfiles was leaking at least 64 bytes of memory each time it is called. This memory leak has been removed. (r. 2759292).
- If a device class has multiple devices registered for it, and the default device is unregistered, a new default device is now correctly set. (r. 2751923).
- In the past, if a device class has devices registered for it with scope of kCFPreferencesCurrentUser, kCFPreferencesAnyHost, or kCFPreferencesAnyUser, kCFPreferencesCurrentHost, but does not have devices registered for it with the scope of kCFPreferencesCurrentUser, kCFPreferencesCurrentHost, then no defacto default device would be set for it. Now, the defacto default device will be set for a class if class has no devices set as such. (r. 2747577).
- CMIterateDeviceProfiles no longer crashes when called with the proc parameter set to NULL. (r. 2743114).
- Fixed CMGetDeviceFactoryProfiles to return the newly updated default profile id in the defaultProfID parameter. (r. 2743028).
- Fixed CMSetDeviceFactoryProfiles to return an error if you pass in illegal values in the dataVersion field and the reserved fields of CMDeviceProfileInfo. (r. 2699519).
- Error is now returned from CMSetDeviceProfiles if either the data version or the reserved fields contain illegal values. (r. 2966779).
- Fixed problem where switching between the different ColorSync Profiles wouldn't take effect for certain displays. (r. 2859752).
- Added new device notification kCMPrefsChangeDeviceNotification for changes in ColorSync Prefs panel. (r. 2853393).
- A problem where the resulting image would be drawn incorrectly when CMMatchImage was used with very wide JPEG files (over 4100 pixels wide) has been corrected. (r. 2838201).
- Fixed crash in CMIterateColorSyncFolder for profiles with Unicode text tag of 'mluc' but not having a language field of 'en'. (r. 2836866).
- A memory leak in CMGetDefaultDevice has been corrected. (r. 2802812).
- It is now possible to proof with a named color space profile in the middle of the proof sequence. (r. 2744111).
- A number of memory leaks have been corrected. (r. 2935758).

Back to top

ImageCapture

The Image Capture architecture provides a common API for discovering digital imaging devices and transferring images from them.

- The TWAIN consortium's libraries have been added to the system in the form of a framework and may be accessed from applications. (r. 2849393).

Back to top

OpenGL / OpenGL Carbon / AGL

OpenGL is a vendor-neutral, multi-platform graphics standard with broad industry support. OpenGL APIs allow you to develop portable, interactive 2D and 3D graphics applications.

- A problem where aglDescribePixelFormat could return incorrect results has been corrected. (r. 2879045).
- A problem where glutHideWindow() was not working correctly in Mac OS X 10.1 has been corrected. (r. 2842975).
- GLUT implementation is now compliant with the GLUT 3 specification. (r. 2781483).
- A problem where the Mac OS X implementation of aglGetInteger() was returning incorrect values when called with the AGL_BUFFER_RECT parameter has been corrected. (r. 2765920).
- A crashing bug in the routine glutDestroyWindow() has been corrected. (r. 2765003).
- OpenGL multi-texture coordinate clipping now works as expected when texture unit zero is disabled. (r. 2879620).
- A problem where calling aglGetCurrentContext() before any GL contexts have been created would crash (rather than returning NULL) has been corrected. (r. 2876559).
- A problem that surfaced in Mac OS X 10.1.3 where glDrawElements could fail if the indices provided where not aligned on a word boundary. This problem has been corrected in Mac OS X 10.2. (r. 2869834).
- Both the vertex array object and vertex array range extensions are now supported in Mac OS X 10.2. (r. 2809428).
- Dependent texturing is supported in Mac OS X 10.2 through the use of Shader Builder. (r. 2808977).
- GL_TEXTURE_BASE_LEVEL cosmetic issue Changing the texture base level from 0 will now work correctly on NVIDIA hardware .

Back to top

Printing

This section discusses new features, bug fixes, and new APIs in the Mac OS X 10.2 printing services.

- Two new constants (kSyncTicketFromPane and kSyncPaneFromTicket) have been added to the Printing dialog extension API constants to this API for interpreting the Sync function's Boolean parameter. These constants are defined in the file PMPrintingDialogExtensions.h. (r. 2891441).
- makequeues now supports custom IOMs.

  Automatic creation of USB printer queues is supported for a broader range of printers. - Olav Andrade (r. 2866760).
- A new API, PMPrinterGetState, that allows callers to check the state of a printer queue has been added. (r. 2866758).
- New API, PMSessionEnablePrinterPresets, has been added that allows callers to enable printing presets in the Print dialog. (r. 2849635).
- Problems where PICTs containing fractionally rotated text or graphics did not print as expected have been corrected. (r. 2843826).
- PMPrinterGetResolution now returns min/max range independent of the current printer module settings. (r. 2843176).
- All printing routines that return data will now return NULL if there is an error. (r. 2843168).
- Problems with printing some PICTs with embeded ColorSync source profiles have been corrected. (r. 2839776).
- A new API was added, PMPrinterGetMakeAndModelName, that allows callers to retrieve information about the make, model and name of a printer. (r. 2811370).
- A new attribute has been added, kUSBIOMSetAltInterface, that allows Printer Modules to select an alternate USB interface. (r. 2811344).

- In the past, Termiate routines in PDEs were not being called if the Initialize routine failed. Now, Terminate routines will be called even if the Initialize routine fails. (r. 2797592).
- PMRelease now correctly handles code that incorrectly tries to release objects obtained via "get" calls like PMContextGetCurrentPrinter. (r. 2793493).
- New APIs to get a ticket and template from a session have been added. These APIs are PMSessionGetTicketFromSession and PMSessionGetTemplateFromSession. (r. 2788795).
- A new API has been added, PMPrinterIsPostScriptCapable, that callers can use to determine if a given printer is a PostScript printer. (r. 2776545).
- Improved handling of NSPrintInfo attributes. A number of new NSPrintInfo attributes are now supported including NSPrintPagesPerSheet, NSPrintReversePageOrder, NSPrintJobDisposition, NSPrintSavePath, and others (r. 2574589). (r. 2773511).
- New APIs have been added to NSPrinter.h and NSPrintInfo.h that allow callers to determine the imageable area of printed pages. (r. 2766258).
- Normally, when a PDE returns kPMDontSwitchPDEError from its Sync function, switching panels, Preview, and Print should be disabled, but, under in some cases the kPMDontSwitchPDEError result code was being ignored. These problems have been corrected. (r. 2764767).
- A problem where PMSessionSetDestination was returning kPMGeneralError rather than using the default format when the destFormat parameter was set to NULL has been corrected. (r. 2741954).
- New Carbon APIs have been added that allow callers to have greater control the page Collation and the ability to set the Collated state for print jobs. (r. 2715677).
- PDEs can now send a Carbon Event to their window to change size after their Prologue has completed. (r. 2679919).
- Embedded PMPageFormat and PMPrintSettings objects are now properly encoded into NSPrintInfo objects. (r. 2555068).
- -[NSPrinter printerNames] now returns the list of available printers (added via Print Center) instead of the list of possible NetInfo printers as in 10.1. (r. 2960293).
- A number of new PMPrinter APIs that allow for some printing administrative functionality have been added to PMPrinter.h. (r. 2932965).
- The Print dialog no longer changes printers if a PDE returns an error due to invalid page ranges, etc. (r. 2894025).
- Fixed a memory leak in NSPrintPanel's setAccessoryView. (r. 2893943).
- Print Center now sends correctly formed SendData requests while auto configuring printers via PAP. (r. 2889207).
- A new API, PMPrinterGetDeviceURI, has been added that allows callers to find out a printer's network address. (r. 2874521).
- Improved handling of PostScript query responses that contain DOS line endings. (r. 2786812).
- Added support for user-specified custom paper sizes. (r. 2713854).
- Mac OS X 10.2 ships with a USB Printer Class driver. (r. 2431394).
- A problem where PostScript printing of PDFs using Separation or DeviceN color spaces could fail over a seven bit communications channel has been corrected. (r. 2825012).
- Corrected a problem that could cause a crash when printing images whose color elements are stored in planar (not in mesh) style. (r. 2853653).

Back to top

Quartz 2D

Quartz 2D is a feature-rich, two-dimensional drawing engine that is accessible from all Mac OS X application environments outside of the kernel. The Quartz 2D application programming interface (API) is easy to use and gives you access to powerful features such as Bezier curves, path-based drawing, transparency, and advanced color management.

- New server side gamma fade APIs are now exported from CoreGraphics/CGDisplayFade.h. (r. 2882664).
- The PNG Data Provider is now available for application's use through the API : CGImageCreateWithPNGDataProvider in CoreGraphics/CGImage.h (r. 2882655).
- New APIs have been introduced for Axial and Radial gradients. The new APIs are contained in CoreGraphics/CGShading.h (r. 2882637).

- A new API, CGContextSetPatternPhase, has been added that allows callers adjust the alignment a pattern when it is drawn. (r. 2846854).
- Fixed bug where QDPictDrawToCGContext draws nothing in millions of color mode. (r. 2837169).
- Cmd-Brightness and mirroring reconfiguration trigger should do nothing if a full screen app is running

Put support in IOGraphics so CG can tell the kernel side the display is captured so that Cmd-Brightness and mirroring reconfiguration trigger does nothing if a full screen app is running. - Steve Martin (r. 2830991).
- WindowServer crashed randomly. Fixed a corrupted region object to address a bug where WindowServer crashes due to a memory smasher. (r. 2829598).
- A problem where CGPostKeyboardEvent for caps lock key could alter the reported state of the shift key has been corrected. (r. 2742243).
- Problems with rendering images taller than 32767 pixels or wider than 32767 pixels have been corrected. (r. 2455072).
- A problem where rotated patterns were not being drawn as expected has been corrected. (r. 2976284).
- Added a convenience API, CGColorSpaceCreateWithName, that allows callers to retrieve ColorSync information directly from CoreGraphics. (r. 2953354).
- It is now possible to switch out the active application when the menu bar is hidden. (r. 2950690).
- A new API was added to CoreGraphics/CGContext.h, CGContextSetShouldSmoothFonts, that allows callers to disable text anti-aliasing. (r. 2940365).
- Problems rendering some CGImageRefs with custom decode arrays have been corrected. (r. 2936294).
- CFDictionarys returned by CGDisplayAvailableModes now include the number of bytes per scan line that can be accessed using the kCGDisplayBytesPerRow key value. (r. 2893175).
- Quartz 2D now respects embedded vector clip paths found in jpeg files when it is rendering them. (r. 2879701).
- A memory leak of about 46 bytes that could occur when releasing a CGBitmapContext has been corrected. (r. 2875555).
- Problems with displaying PDF files with Device CMYK colors or images have been corrected. (r. 2837498).
- Problems causing CGRectDivide to produce erroneous results have been corrected. (r. 2791321).
- CGPostKeyboardEvent fixed to work for help key (key code 0x72). (r. 2784337).
- A problem where CGContextStrokePath could hang when adding a zero length stroke has been corrected. (r. 2767903).
- Cases where calling CGPostKeyboardEvent from Cocoa applications would fail with certain key codes have been corrected. (r. 2689300).
- Problems where the CGRemoteAccess event APIs were not preventing display sleep have been corrected. (r. 2667840).

Back to top

QuickDraw

QuickDraw is a part of the Mac OS used for drawing and displaying graphical information on the screen and other raster devices.

- Fixed problem where calling DrawPicture to draw a picture to a GWorld could result in some parts of the image being clipped incorrectly for some bit depth combinations. (r. 2896796).
- Routines have been added to QuickDraw.h that allow callers to temporarily disable region and polygon recording. (r. 2873386).
- A problem where calls to FrameRoundedRect with invalid or empty rectangles could result in drawing operations has been corrected. (r. 2868895).
- A problem where the default picture data recording routine could attempt to access a NULL pointer has been corrected. (r. 2868114).
- A number of new Local/Global coordinate transformation routines have been added. These include QDLocalToGlobalPoint, QDGlobalToLocalPoint, QDLocalToGlobalRect, QDGlobalToLocalRect, QDLocalToGlobalRegion, and QDGlobalToLocalRegion. (r. 2835945).
- A problem when FillCRect would not draw anything when drawing misaligned 32 bit patterns has been corrected. (r. 2833800).

- A problem where some poorly formatted pictures could hang the picture parsing code has been corrected. (r. 2824979).
- A problem that occurred Mac OS X 10.0 and Mac OS X 10.1 in where calls to NewGDevice could switch the current GDevice has been corrected. Now, calls to NewGDevice no longer switch the current GDevice. (r. 2814795).
- The new APIs, QDSwapTextFlags and QDSwapPortTextFlags, have been added to QuickDraw.h that allow callers to ask QuickDraw to use Quartz 2D for drawing anti-aliasing text. (r. 2810408).
- Problems in PaintPoly that could cause it to draw incorrectly when drawing some self-crossing polygons have been corrected. (r. 2807382).
- To simplify debugging, attempts to call QuickDraw to do any drawing between calls to QDBeginCGContext and QDEndCGContext are now logged to the console as "Ignoring QuickDraw drawing between QDBeginCGContext and QDEndCGContext". (r. 2802466).
- A problem that occurred in Mac OS X 10.1 where calls to CopyBits could write beyond the end of the destination PixMap's raster array (and cause heap corruption) when the destination's rowBytes value was not an even multiple of 4 has been corrected. (r. 2792873).
- Some new Cursor APIs have been added to QuickDraw.h that allow callers to specify larger cursors. These APIs includeQDIsNamedPixMapCursorRegistered, QDRegisterNamedPixMapCursor, QDUnregisterNamedPixMapCursor, QDSetNamedPixMapCursor, and QDSetCursorScale. (r. 2786270).
- A problem where calls to CopyBits using the transparent copy mode could result in a black vertical line on the edge of the destination rectangle have been corrected. (r. 2780384).
- A problem where DrawPicture could read past the end past the end of the picture data in memory when drawing large pictures has been corrected. (r. 2735743).
- Fixed CopyDeepMask now works as expected when the mask is 2, 4, or 8 bits deep. (r. 2557268).
- CreateNewPort can now be called from a daemon process. (r. 2519829).
- Error checking for Region size overflow during UnionRgn, SectRgn, or DiffRgn calculations has been improved. Now, when an overflow is detected, QDerror will return a non zero result code and the resulting region will be set to an empty region. (r. 2377836).
- A memory leak in SetPortPenPixPat has been corrected. (r. 2975566).
- A problem where lnitGDevice was not always setting the gdType field in GDHandles reliably has been corrected. This problem was present in all versions of lnitGDevice until now. (r. 2964709).
- Some drawing problems with CopyDeepMask have been corrected (r. 2867617). As well, a problem where CopyDeepMask could crash if the maskRect was different than either the src or dst rectangle has been corrected. (r. 2908766).
- Previously, calls to CopyBits with a 1, 2, or 4 bit source and a 32 bit destination would only set the alpha channel to 255 if the destination was a window buffer. Now, the alpha channel will be set to 255 when the destination is an off-screen GWorld as well. (r. 2852806).
- A problem where calls made to SetCPixel during printing could crash has been corrected. (r. 2840205).
- A problem where calls to GetPictInfo() could call the get picture data QDProc with a zero byte count has been corrected. (r. 2840062).
- A problem that appeared in Mac OS X 10.1.2 where GetPictInfo could hang when called with a JPEG compressed picture has been corrected. (r. 2833865).
- A problem where calls to HasDepth for the main device were returning false for all bit depths after the main device's depth was switched to 8-bit has been corrected. (r. 2799168).
- A number of problems in the drawing produced by calls to CopyMask have been corrected. These included clipping issues (r. 2779309), mask width calculations (r. 2585796), the appearance of diagonal lines (r. 2637002) (r. 2637002).
- Large cursor support, for cursors up to 64 by 64 pixels, has been added to the QuickDraw APIs. (r. 2827587).

Back to top

QuickDraw Text

QuickDraw Text is the part of the Mac OS used for drawing and displaying textual information on the screen and other raster devices.

- A new API, QDSwapPortTextFlags, has been added that allows callers to set the QD text drawing flags on a per-port basis. (r. 2888518).
- Improved QuickDraw rendering of Japanese anti-aliased text. (r. 2489470).

Back to top

Hardware/Devices

This section discusses changes and new features for device driver writers provided in Mac OS X 10.2.

ATA

The Apple's support for connected ATA devices.

- Large ATA hard drives (over 137 GB - 48-bit LBA addressing) are now supported on systems with large ATA drive support in the boot ROM. All changes are documented in the relevant header files in IOATAFamily. .

Back to top

Ethernet

Apple's ethernet software provides low level networking connectivity by controlling Apple ethernet hardware supplied by Apple with Apple hardware.

- Implemented support to force the UniNEnet driver to disable auto-negotiation mode and to force a specific media type and speed with the ifconfig tool. For example, the following terminal commands can be used: % ifconfig en0 media 10baseT/TP % ifconfig en0 mediaopt half-duplex (r. 2798973).
- The media and mediaopt settings are only supported on computers with gmac+ Ethernet chipsets.
- Cases where some third party ethernet adapters were sometimes mistaken for the primary ethernet interface on certain machines in 10.1.x have been corrected. (r. 2764618).

Back to top

FireWire

Apple's support for the IEEE 1394 High Performance Serial Bus standard.

- A new API, RemoveIsochCallbackDispatcherFromRunLoop, has been added that allows callers to remove any IsochRunLoopSources they have installed using AddIsochCallbackDispatchToRunLoop. (r. 2795841).
- The routines AddCallbackDispatcherToRunLoop and AddIsochCallbackDispatcherToRunLoop have been updated to allow callers to specify valid runloop modes for the callback routines they are being used to install. (r. 2738757).
- The IOKit FireWire headers have been updated so they can be used with standard C. (r. 2737041).
- Fixed a problem with isoch reception on PCI-Lynx machines (i.e. B&W G3, original G4). (r. 2851043).
- The timout period for AVC devices to respond after a bus reset has been increased to accommodate slower devices. (r. 2842540).
- Add support in FireWire DV for DVCPRO CIP headers. (r. 2709090).
- Improved compatibility with a number of third party FireWire cards. (r. 2923203).

Back to top

HID Support

The Human Interface Device (HID) class is one of several device classes described by the USB (Universal Serial Bus) architecture. The HID class consists primarily of devices humans use to control a computer system's operations.

- A problem where getElementValue() would not query a device (if the value being requested had not been included in a previous input report provided by the device) has been corrected. (r. 2883840).
- The HID Manager now publishes unit and unit exponent properties for each hid element. (r. 2883290).
- Added support for scroll wheel acceleration. (r. 2880436).

- The HID Manager now supports Mouse devices. (r. 2770086).
- The HID manager now supports Keyboard devices. .
- The HID manager now supports non-USB mouse and keyboard devices. (r. 2861323).
- A new method, IOHIDDevice::updateElementValue, has been added to ensure values return for feature element inquiries are up to date. It is called when ever a user client attempts to access a value for a feature element. The new method calls to IOHIDDevice::getReport to poll the device for the report values. Then it calls IOHIDDevice::handleReport with the descriptor that was filled in with getReport. (r. 2731494).
- A problem where IOMatchingDictionaries ware returning all HID devices rather than just matching ones has been corrected. Now only matching devices are returned. (r. 2690962).
- Added horizontal scroll wheel support. (r. 2660532).

Back to top

IOKit

Apple's object-oriented I/O development model. The I/O Kit provides a framework for simplified driver development, supporting many families of devices.

- A memory leak in IOCreatePlugInInterfaceForService where one or more CFArrays were not being released has been corrected (r. 2927887).
- Fixed the IODataQueueDequeue call so that the dataSize parameter is updated properly, even when the data pointer is null. (r. 2926620).
- Both IOGeneralMemoryDescriptor and IOSubMemoryDescriptor now have a serialize method so they can display themselves in ioreg output. (r. 2916500).
- The ioreg tool now has an option for printing numbers in hex. (r. 2916477).
- IOPCIBridge now conforms to the PCI spec which says that an I/O memory range read from the PCI config register may have their 16 upper bits set to 0, in which case they should be ignored. Previously IOPCIBridge assumed that all 32 bits of an I/O range were valid. (r. 2909765).
- Memory descriptor redirect (protected from access during sleep) now works for non-block-mapped mappings (< 32K). (r. 2893715).
- Fixed problem where IOSharedInterruptController could become permanently disabled on MP systems. (r. 2875529).
- Fixed a problem with IODataQueue where an entry that just fit at the end of the queue memory buffer would instead be added at the start of the buffer. This would cause the user client code to return the wrong entry. (r. 2860701).
- The routine IOSleep() is now uninterruptible. (r. 2857482).
- The ioreg and IORegistryExplorer tools fixed a problem that didn't show @location for objects that are part of an alias on the device tree plane. With the addition of IORegistryEntryGetLocationInPlane() API call, the tools now work correctly. (r. 2856361).
- Added the ability to describe a parent object in a matching dictionary. (r. 2845592).
- Fixed a problem with interruptible IOCommandGate::commandSleep() that could corrupt the driver's work loop. (r. 2789143).
- A mechanism was added to track the number of OSContainers an object is in. If the retain count goes less than the container count the system will panic. This is to track cases where drivers are releasing too many object references thus corrupting the I/O Registry. (r. 2781013).
- Changed IOMemoryDescriptor::getSourceSegment() to no longer ignore errors from IOMemoryDescriptor::prepare(). This avoids the rare case where callers would assume that the memory was prepared when it actually wasn't. (r. 2757553).
- Added new API IOBufferMemoryDescriptor::inTaskWithOptions.This method allocates a memory buffer with a given size and alignment in the task's address space specified, and returns a memory descriptor instance representing the memory. Pageability and sharing are specified with option bits. This API is useful for graphics accelerators drivers, for example. (r. 2950553).
- The constant kIOMasterPortDefault may now be passed in IOKit APIs. When the kIOMasterPortDefault constant is used, the IOKit APIs will look up the default master port on behalf of the caller. (r. 2920131).
- A new API, IOServiceGetMatchingService, has been added that allows callers to create simpler code for locating services they require. This new call is a combination of IOServiceGetMatchingServices() and IOIteratorNext() that saves callers from having to create iterator and retrieve the first object from it when it is known that there is only one match. (r. 2920026).

Back to top

Mass Storage

Drivers for storage devices such as hard disks and CD-RW drives.

- Fixed a problem that occurs when editing content on a disk node that (a) has content that some content driver wishes to probe, and (b) whose driver is not loaded at the time it is needed to probe. (r. 2913406).
- Fixed a bug that caused premature closure of the IOBlockStrorage driver before all of its the completion routines had finished executing. (r. 2886160).
- Added support in Storage Family to accommodate incrementally recordable file systems. For instance, udf on CD-R. (r. 2885228).
- IOBlockStorageDevice now supports 64-bit block numbers. (r. 2826537).
- The new IOMedia object property kIOMediaRemovableKey and IOBlockStorageDevice object property kIOPropertyProtocolCharacteristicsKey { kIOPropertyPhysicalInterconnectLocationKey } are now available in IOMedia.h, IOStorageDeviceCharacteristics.h. These properties are used to describe objects as either "removable" and "detachable". (r. 2751422).
- The property IOMediaIcon was added to IOMedia objects to support custom media icons. As well the new KextManagerCreateURLForBundleIdentifier() client function has been added for getting path to a loaded bundle so a program can get at the bundle and its info dictionary. (r. 2566007).

Back to top

Power Manager

The Power Manager is the part of the Mac OS that controls power to the internal hardware devices of Macintosh PowerBook computers.

- Uninterruptible power supplies are now supported. A basic driver now exists for UPSes conforming to the USB Power Class specification. The default behavior is for the system to be shut down via /sbin/halt when the UPS battery is drained below 20% of its capacity. Developers can replace this behavior with their own UPS monitor process.(r. 2863893). (r. 2972495).

Back to top

SCSI Architecture Model (SAM)

Support for devices conforming to the SCSI Architecture Model. This includes ATAPI, FireWire, and some USB storage devices.

- Added a new API, SetAutoSenseDataBuffer, to SCSITaskInterface to allow retrieving sense data longer than the minimum eighteen bytes. (r. 2972370).
- The synchronize cache command is now sent to devices which have a write cache. The write cache is enabled at startup and then disabled at shutdown. (r. 2957580).
- Added kIOMessageTrayStateHasChanged and kIOMessageMediaAccessChange notifications to SCSITask User Client. (r. 2928061).
- A memory leak in SCSITaskUserClient has been corrected. (r. 2767519).
- SAM commands now have reasonable timeouts to prevent ill-behaved devices from hanging the system indefinitely. (r. 2760367).
- The SCSI Protocol Layer now supports the required Management functions as defined by the SCSI Architecture Model. These functions include Target Reset, Logical Unit Reset, Abort Task, etc. (r. 2587364).

Back to top

USB Support

A set of APIs for communicating with USB devices connected to your Macintosh.

- Fixed a problem with ResetPipe which did not work properly with Isoc pipes where the result was such that USB did not function properly after an attempt to use one was made. (r. 2912416).

- Implemented the following changes to USB isochronous transfers: (a) If we get an DATAOVERRUN because the whole Transfer Descriptor was late, we change the error to kIOReturnIsoTooOld. This will allow a client to realize that all the frames in the TD will have a kIOUSBNotSent2Err so it does not have to parse all the frames in the TD. And, (b) When setting the aggregateStatus (status that will be returned if the completion had no error but frames in the TD did have errors), assign a low priority to kIOReturnUnderrun. We will only return kIOReturnUnderrun if there is NO other error in any of the frames. The underrun condition is fairly common for isoc transactions and not as useful as more important errors. (r. 2884265).

- Fixed the SetAlternateSettings function to return a failure result if there is insufficient bandwidth. (r. 2880635).

- Fixed a problem with the IOUSBInterfaceUserClient such that in an asynchronous call, the user client object was being released prior to the completion callback. (r. 2874390).

- IOUSBHIDDriver now subclasses setReport. The original implementation supported getReport only. This makes it possible to use the standard HID Manager calls to send data to a HID device. (r. 2872057).

- A potential problem in the routine IOUSBInterface::handleOpen that could cause it to fail in circumstances where it was okay to proceed has been fixed. (r. 2869674).

- Implemented a session ID property for an IOUSBDevice to use in tracking a device. The session ID property changes each time the device is re-connected. (r. 2858029).

- Converted the AppleUSBMouse driver to subclass from IOHIDFamily. You can now see the Apple USB Mouse device from the HID Explorer sample code. (r. 2855953).

- Implemented support for ClassicMustSeize and ClassicMustNotSeize at the USB Interface level. This functionality was already supported, since Mac OS X 10.1, at the USB Device level. This is discussed further in the USB SDK's document User Mode USB Arbitration. (r. 2853912).

- Implemented a fix with USB so that the Isochronous error codes are correctly returned to the caller. Previously, the return errors were being shifted 1 extra bit resulting in incorrect results. (r. 2850929).

- Fixed the GetConfiguration call so that it would not overwrite memory on return. If you call the "IOReturn (*GetConfiguration)(void *self, UInt8 *configNum)" function passing in a buffer of memory instead of a single UInt8 byte's address, the first byte "configNum" is correctly set, however, the 3 bytes following this location would be overwritten. (r. 2842636).

- Fixed bug that caused delays with certain USB input devices using universal HID driver. (r. 2830705).

- Implemented USB suspend/resume support. (r. 2778595).

- Fixed a problem with data corruption in reading audio tracks from some USB Mass Storage devices. (r. 2771845).

- Fixed a problem where connecting a USB hub with multiple USB keyboards attached would result in some of the keyboards failing to enumerate. (r. 2763848).

- Fixed the IOUSBDevice::GetFullConfigurationDescriptor call so that the call is reentrant. (r. 2756249).

- Fixed the SetReport call in the IOUSBHIDDriver so that it will set report values on a specified HID device. (r. 2588675).

- Fixed the USB Mass Storage Class driver so that when media in a manual eject USB floppy drive is removed, the volume will always be unmounted from the desktop. (r. 2917459).

- Fixed conditions where USB requests to read and write on stalled pipes get queued on the endpoint. USB requests now return immediate errors in this condition. (r. 2905718).

- Fixed to USB Mass Storage driver to address a panic condition which occurred with a number of USB Mass Storage devices when they were hot unplugged. (r. 2826979).

- Fixed bug where the IOUSBHIDGetReport call in IOUSBHIDDriver could return invalid information. (r. 2588665).

- So clients can specify a request count when calling the IOUSBPipe read and write methods, the GetReqCount() method has been added. This allows clients to explicitly specify the number of bytes to transfer rather than requiring the use of the getLength method from the IOMemoryDescriptor to determine the number of bytes to transfer. (r. 2465039).

- Fixed the Isochronous code in the IOUSBFamily to return IOReturn values in the frStatus fields of an isochronous frame structure, instead of the hardware status codes which were previously being returned (r. 2809227). .

Back to top

Java

This section discusses changes and fixes present in the Java implementation provided in Mac OS X 10.2.

Bridge Technology

Java Bridge Technology allows Java classes to subclass Objective-C classes and reference Objective-C classes.

- A problem where Cocoa Java programs invoked from Java failed to exit has been corrected. (r. 2856185).
- Java Bridge didn't handle calls from Objective-C to Java properly when the function had a float parameter, it now does. (r. 2714366).

Back to top

Embedding

Java Embedding is a framework that allows Java Applets to be embedded inside of Carbon applications.

- Drawing with Swing from two threads could cause a hang in some applications. This has been fixed. (r. 2900957).
- A problem where some Java plugins could crash while being loaded has been corrected. (r. 2922549).

Back to top

Java Classes

The Java Classes component includes all parts of Apple's Java implementation beyond the virtual machine: AWT/Swing, tools, Apple APIs, hardware acceleration, etc.

- Some Java applications packaged on 10.1.x stated they required a newer version of Mac OS X to run when launched in 10.1.0. This problem has been corrected and these applications can be run in Mac OS X 10.2. (r. 2901962).
- The hwaccellist has been updated to be an opt out list instead of an opt in list. (r. 2870062).
- The Policy tool didn't allow saving of files with a dot (.) in front of the name as expected by java, it now does. (r. 2863801).
- com.apple.mrj.dnd.OutgoingDrag has been updated to use FSRefs for better interoperability with the Mac OS X file system. (r. 2855255).
- Calling Paper.setImageableX() (and Y()) or setImageableArea() now work properly for setting custom page margins. (r. 2851380).
- A race condition that could occur while attempting to use multiple signed Applets at the same time has been fixed. (r. 2847933).
- OpenGL glyphvector drawing is now implemented. We also now correctly determine visual boundaries of primitives we draw using fall backs, so that they no longer get clipped when blit back to the screen. (r. 2805509).
- With OpenGL acceleration on, the Mix.BezierScroller demo in Java2Demo had a bad frame, this has been fixed. (r. 2804556).
- A problem where a JavaApplications would hang if it tried to activate multiple JDialogs has been corrected (r. 2804122).
- A problem where Java applications launched with an open document Apple Event would not receive the event has been fixed. Now, the event is queued until the application is ready to process it. (r. 2784162).
- A problem where modifier keys were being dropped for command events inside of java.awt has been corrected. (r. 2720644).
- A problem in awt.graphics where attempting to set the clipping region to an empty area could cause an error has been corrected. (r. 2693942).
- A problem where Java frames were not displaying their appropriate anchors has been corrected. (r. 2559558).
- A problem where some applets failed to load because of untrusted server cert chain exceptions has been corrected by updating JSSE certificate authority certificates. (r. 2970172).
- Some miscalculations in the offset of a scrollpane's content when more than one scroll bar was being used have been corrected. (r. 2962419).
- A problem where scaling a jpg on the fly to fit inside of part of a drawable surface did not produce any visible image has been corrected. This problem appeared with Java 1.3.1 update 1. (r. 2891548).

- The value returned by The MouseEvent.getY() value was being calculated incorrectly in cases where Frame.setLocation(x, y) set the y location between -1 and -20. This problem has been corrected. (r. 2883773).
- A problem where the pageDialog function always returned the same PageFormat object provided as a parameter rather than an updated PageFormat object has been corrected. This problem appeared with Java 1.3.1 Update 1. (r. 2869662).
- A problem where, under certain conditions, swing dialogs were not being updated correctly has been corrected. This problem appeared with Java 1.3.1 Update 1. (r. 2869281).
- A problem where it was not possible to drag windows above the system menu bar on multiple monitor systems has been corrected. This problem appeared with Java 1.3.1 Update 1. (r. 2868758).
- A problem where Java Swing Applets did not reload correctly in some internet browsers has been corrected. This problem appeared with Mac OS X 10.1.2. (r. 2846360).
- A problem where Client drag images were ignored has been corrected. (r. 2783811).
- A problem where multiple help menus could be inadvertently created has been corrected. (r. 2755635).
- Tidied up Java Web Start's interaction with the dock. Radically reduced the number of different icons shown in the dock as a Java Web Start application starts up. (r. 2751985).
- The MRJOpenApplicationHandler was not being called on Mac OS X when the openDocument event caused the application to be launched, it only worked when the application was already running. This has been fixed, so that the openDocument event is delivered even if the application was not running when the document was dropped on it. (r. 2720352).
- The MRJQuitHandler method was being called twice if there was also an MRJAboutHandler. Now, it is only called once. (r. 2680426).
- A problem where displaying a dialog inside of the MRJQuitHandler method would hang an application has been corrected. (r. 2591363).
- The MRJToolkit has been updated to support the standard application menu operations including quitting the application, bringing up the About Box, and bringing up the Preferences dialog. (r. 2511089).
- A race condition in Thread.stop() has been corrected. (r. 2870719).

Back to top

Java Deprecations

This section lists Apple Java APIs that have been deprecated as of Mac OS X 10.2 Jaguar. The APIs have been flagged as deprecated and will trigger a deprecation warning at compile time.

- MRJFileUtils has been deprecated. (r. 2964506).
- MRJ Drag and Drop is now deprecated. Use the pure Java 2 Drag and Drop api's instead. (r. 2889765).
- MRJ DataTransfer (com.apple.mrj.datatransfer) is deprecated in Jaguar. (r. 2905002).
- MRJ console (com.apple.mrj.console) has been deprecated in Jaguar. (r. 2905006).

Back to top

Java VM

Java Virtual Machine used in Mac OS X.

- HotSpot used to ignore the ThreadStackSize=n JVM setting, now it handles it properly. (r. 2913774).
- JVM used to truncate an invoking thread's stack size to 512K, it now leaves the invoking stack size alone. (r. 2941848).
- The Java SDK now contains all Sun changes for 1.3.1_03. (r. 2895871).

Back to top

Launch Services

Launch Services provides APIs applications can use for launching other applications.

- LaunchServices is now thread safe. (r. 2896505).
- Local applications on hard drives are now launched before identical copies of the same application found on CD drives. (r. 2815028).
- A problem where local programs were not being given preference at document-opening time over server application has been corrected. Now, a local copy of an app takes priority over a copy with the same version number located on a network volume. A higher version number will trump the local copy, but in that case you can set a preference for the local copy using the Finder's Info Window. (r. 2645865).
- The kLSLaunchDontSwitch flag is now respected when using LSOpenFromURLSpec. Previously, this flag was ignored. (r. 2755764).
- MIME info can now appear in application plists along side CFBundleTypeExtensions and CFBundleTypeOSTypes to identify files an application is capable of displaying or editing by MIME type. The MIME key is CFBundleTypeMIMETypes. (r. 2722760).
- The new LaunchServices APIs, LSCopyKindStringForTypeInfo() and LSCopyKindStringForMIMEType(), were added to allow callers to retrieve kind strings of non-existent files. (r. 2696767).
- Previous 1K restriction to the length of URLs that can be passed to Launch Services have been removed. Now URLs can be of any size. .

Back to top

Mach Kernel

The complete Mac OS X core operating system environment that includes Mach, BSD, the I/O Kit, file systems, and networking components.

- Fixed a subtle bug in IOCatalogue where, if IOCatalogue was asked to add a new personality, it would compare the personality against all those it knows, but would check only the keys of the dictionaries already present. This could lead to a case where adding a new personality that is a superset of an existing one would cause the existing one to be dropped as a duplicate. (r. 2928968).
- When Safe Booting (shift key down), ignore I/O Kit personalities with the IOKitDebug property. This is to allow developers to bypass loading a panic()ing boot-time kext. (r. 2864324). (r. 2920410).
- sprintf and snprintf now support quad-ints (%qd, %qu, %qx, etc). (r. 2917340).
- kextload now does full authentication by default, and only allows the invoker to skip authentication if the kext is not being loaded (as with -n). (r. 2906248).
- New commands kextload, kextunload, and kextcache obsolete kmodload/kmodsyms, kmodunload, kmodsyms, and mkextcache.(r. 2875218, 2875215, 2899022) (r. 2898582).
- Kext loading now passes through dependencies on kexts with no binary directly to the next kext that does have a binary. Arbitrary levels of aliasing through codeless kexts are supported. (NOTE: This only works for run-time kext loading. The boot-time loader does not yet support this feature.) (r. 2894843). (r. 2894843).
- If kextd or kextload is launched in safe boot mode (shift key down), it will not use the info dictionary cache and will instead re-scan the kexts themselves. (r. 2887998).
- dyld now correctly handles unknown load commands with the LC_REQ_DYLD bit set. (r. 2882932).
- A problem where the routine thread_sleep_mutex() was returning an invalid value has been corrected. (r. 2863728).
- The criteria for kernel preemption has been expanded to tasks running in the application band, and those member threads with +importance that are not timeshared. (r. 2862611).
- A race condition inside of the routine wait_queue_remove() that could cause unpredictable behavior for threads in an un-interruptible wait state has been corrected. (r. 2857550).
- The kextload command now allows more than one kext argument to be specified. (WARNING: Utilizing this feature may make error diagnostics difficult as any returned error exit status is ambiguous for multiple kexts.) (r. 2854652).
- The maximum limit for malloc was too small. This has been increased to a limit of 2.5 GB for the text, data, and heap and the default stack can be up to 509 MB (r. 2848946). (r. 2848945).
- In order to reduce application launch times, the kernel now maintains information about the working set of an application between launches (in "/var/vm/app_profile"). Pre-heat files are meant to be transparent to the user; however, developers who are constantly re-working their applications may find that their pre-heat files are getting large. The files may become clogged with out-of-date profiles on applications who's versions have changed. As a result, developers may find that it is good to clear out the old pre-heat files on test machines once in a while. To do this, become super-user and do a rm -r /private/var/vm/app_profile and then reboot. app_profile is the directory which contains the profile files. The directory is automatically re-created on reboot. (r. 2847332).

- As xnu no longer builds with PBWO and the presence of the PB.project file was confusing people into thinking that it was, we've removed this file. (r. 2841647).
- A problem where terminating a mach thread from another mach thread could leave a thread running but marked as ABORTED with thread-self ports disabled has been corrected. This problem could only occur on MP machines. (r. 2840580).
- A problem in the processor_set_threads() routine that could cause can cause thread reference underflows or kernel data corruption and/or panics has been corrected. (r. 2838105).
- The version number of com.apple.kernel, com.apple.kernel.bsd, com.apple.kernel.iokit, com.apple.kernel.libkern, and com.apple.kernel.mach has been incremented to 6.0. (r. 2796646).
- The routine KUNCUserNotificationDisplayFromBundle call now uses a "%@" character sequence to separate out words to add to the dialog it presents. (r. 2795373).
- Fixed a long-standing problem where a dependency version incompatibility would report the requested version of the dependency instead of its actual version. (r. 2789001).
- All non-Objective-C mach headers have been modified to use #include instead of #import for better compatibility with C programs. (r. 2778821).
- A problem where the Velocity Engine and floating point contexts could be lost when a machine was put to sleep has been corrected. (r. 2777855).
- The processor load averages reported by systctl() (and, therefore, by "w", "uptime" and friends) have been corrected so they have meaning similar to the values reported on other Unix systems. (r. 2752471).
- kernel malloc has been updated to return a pointer to a zero length block of memory when called with a zero argument. Prior to this change kernel malloc would return NULL if asked to create a zero length block. (r. 2713055).
- The kextload command now correctly returns a nonzero result for some error conditions that used to return zero, such as if the binary to load was absent. (r. 2712179). (r. 2712179).
- The kextload command now does more extensive error checking when parsing kext plists. (r. 2679397).
- System V Semaphores for xnu are installed with Mac OS X 10.2. (r. 2676742).
- It is now possible to enable hardware floating point exceptions. (r. 2656343).
- Panic dumps to the screen are now a boot-time option. To turn off the dialog and get the old textual presentation of info on the screen, set boot-args to debug=0x100. (r. 2653104).
- The panic dump no longer includes dependency info for in-kernel kernel modules such as com.apple.kernel.iokit. This information added no real benefit and simply made panic dumps longer. (r. 2641919).
- Previously, the CFBundleIdentifier property was required in all IOKitPersonalities. In most cases that property would simply refer to the KEXTs own bundle. Now, a personality without a CFBundleIdentifier property is assumed to be referring to its own bundle. (r. 2604841).
- Incompatibilities between the headers <sys/systm.h> and <kern/sched_prim.h> that prevented them from being included in the same kext source file have been corrected. (r. 2578701).
- Kernel panics are now displayed in a dialog box. It is possible for developers to turn on the older display when it is needed (see 2653104 above). (r. 2512985).
- A problem where a kernel hang could occur when mapping files has been corrected. (r. 2477145).
- Many third party kexts do not load in Jaguar due to strict permissions checking. Now, all kext bundles must have owner and group root:wheel with permissions 755 for folders and 644 for files in order to load. If a kext fails this check, the user is presented with a panel informing them that the kext could be a security hole, and offering them the option to allow it to load anyway, to fix its ownership and permissions so it will load from that point on, or to disallow the load. (r. 2867143).
- CFBundlePointerFromFunctionName was crashing when the named function was not present. This has been fixed. (r. 2857197).
- The new kext management code produces better diagnostic messages if a kext cannot be loaded. (r. 2803079).
- Loading too many boot-time kexts built with full symbols can cause the secondary loader (BootX) to run out of memory. The result is the broken System Folder icon at boot time. Future releases of Mac OS X after 10.2 may allocate more memory for BootX. However, it's never necessary for debugging purposes to run a kext with full symbols. The symbols only need to be visible to the debugging system, not the target system.

See previous comment submitted.(r. 2941822). .
- Fixed a bug where Kernel panic could occur when the vmmStopVM kernel API was called. (r. 2837235).
- Fixed a bug where a funnel lock was being held during vm_map_remove. This call can block and cause the mouse to be "skittery" when relinquishing huge number of pages. (r. 2826784).

Back to top

Math & Logical Utils

The mathematical libraries included with Mac OS X.

- A complete BLAS implementation ships with Mac OS X 10.2 (Complete in 10.2). The Basic Linear Algebra Subroutines (BLAS) are high quality leaf routines for performing basic vector and matrix operations. Level 1 BLAS consists of vector-vector operations, Level 2 BLAS consists of matrix-vector operations, and Level 3 BLAS have matrix-matrix operations. The efficiency, portability, and the wide adoption of the BLAS have made them common place in the development of high quality linear algebra software such as LAPACK or other technologies requiring fast vector and matrix calculations. This BLAS implementation covers vector and scalar, complex and real, single and double precision. For more information refer to <http://www.netlib.org/blas/faq.html>

  The implementation of the BLAS in Mac OS X 10.2 is based on ATLAS and has benefited from additional hand tuning whenever possible. Additionally, the matrix operations follow a proprietary block algorithm for small to medium size dimensions. This method improves the speed of functions such as SGEMM on the vector engine and keeps bit per bit compatibility with its scalar counterpart.

- The LAPACK library ships with Mac OS X 10.2 (New in 10.2). LAPACK provides routines for solving systems of simultaneous linear equations, least-squares solutions of linear systems of equations, eigenvalue problems, and singular value problems. The associated matrix factorizations (LU, Cholesky, QR, SVD, Schur, generalized Schur) are also provided, as are related computations such as reordering of the Schur factorizations and estimating condition numbers. Dense and banded matrices are handled, but not general sparse matrices. In all areas, similar functionality is provided for real and complex matrices, in both single and double precision. LAPACK in vecLib makes full use of the hand tuned BLAS and fully benefits from its performance. This LAPACK implementation covers vector and scalar, complex and real, single and double precision. For more information refer to <http://www.netlib.org/lapack/index.html>. Note that vecLib's LAPACK was built using the FORTRAN to C converter called f2c. As a result, users must be aware that: (a) ALL arguments must be passed by reference. This includes all scalar arguments such as matrix dimension M and N, and (b) there is a difference in the definition of a two-dimensional array in Fortran and C. For more information refer to <http://www.netlib.org/clapack/readme>.

- The FixMath routines have been optimized and their rounding operations have been corrected to be in sync with previous versions of FixMath (Improved implementation in 10.2). The algorithms in FixMath have been revamped to give the same answers in jaguar as in Mac OS 9. All rounding mode sensitivities have been corrected and major performance improvements to all functions have been made.

- The libm library is now standard compliant (Standard compliant in 10.2). The new math library in jaguar is now IEEE-754 and C99 compliant in double precision. In addition, the new libm is faster than MathLib found in Mac OS 9 and faster than libm in Mac OS 10.1.x.

- SIGFPE (New in 10.2) Jaguar floating-point environment now fully supports IEEE-754 style exception halts. The following example illustrates its use:

```
#include <math.h>
#include <fenv.h>
#include <signal.h>
#include <ucontext.h>
#include <mach/thread_status.h>

#define fegetenvd(x) asm volatile("mffs %0" : "=f" (x));
#define fesetenvd(x) asm volatile("mtfsf 255,%0" : : "f" (x));

enum {
  FE_ENABLE_INEXACT    = 0x00000008,
  FE_ENABLE_DIVBYZERO  = 0x00000010,
  FE_ENABLE_UNDERFLOW  = 0x00000020,
  FE_ENABLE_OVERFLOW   = 0x00000040,
  FE_ENABLE_INVALID    = 0x00000080,
  FE_ENABLE_ALL_EXCEPT = 0x000000F8
};

typedef union {
  struct {
    unsigned long hi;
    unsigned long lo;
  } i;
  double d;
} hexdouble;

void myHandler(sig, sip, scp)
  int sig;
  siginfo_t *sip;
  struct ucontext *scp;
  {
    hexdouble t;
    ppc_float_state_t *fs;
    ppc_thread_state_t *ss;
```

```
    fs = &scp->uc_mcontext->fs;
    ss = &scp->uc_mcontext->ss;

    printf("SIGFPE taken at 0x%x invokes myHandler, fpscr = %08X\n",
            sip->si_addr, fs->fpscr);

      /* Re-arms interrupts when this state is restored */
    fs->fpscr &= FE_ENABLE_ALL_EXCEPT;
      /* Advances the PC when this state is restored */
    ss->srr0 += 4;
    printf("fpscr = %08X\n", fs->fpscr);
}


static struct sigaction act = { myHandler, (sigset_t)0, SA_SIGINFO };

main ()
{
    float s;
    hexdouble t;

    fegetenvd(t.d);
        /* Enable hardware trapping for all exceptions */
    t.i.lo |= FE_ENABLE_ALL_EXCEPT;
    fesetenvd(t.d);
        /* Set handler */
    if (sigaction(SIGFPE, &act, (struct sigaction *)0) != 0) {
        perror("Yikes");
        exit(-1);
    }
    fegetenvd(t.d);

        /* Overflow folded out by compiler */
    s = HUGE_VALF * HUGE_VALF;
        /* Inf/Inf raises invalid operation */
    s = s / (1.0 + s);

    fegetenvd(t.d);

        /* verify that the compiler is doing
        the right thing */
    printf("In main(1), s computed as: %e ,
        fpscr = %08X\n", s, t.i.lo);

        /* Overflow folded out by compiler */
    s = HUGE_VALF * HUGE_VALF;
        /* Inf/Inf raises invalid operation */
    s = s / (2.0 + s);

    fegetenvd(t.d);

        /* verify that the compiler is doing
        the right thing */
    printf("In main(2), s computed as: %e , fpscr = %08X\n",
        s, t.i.lo);
}
```

- Rounding control (changed in 10.2) The return values of the floating point environment routines fesetround(), feholdexcept() have been changed and corrected. These routines in Mac OS 10.1.x and earlier returned boolean status opposite to that specified by C99. Jaguar libm corrects this and brings them into agreement with the ISO/IEC 9899:1999 (C99) specification.

- The gamma function has been changed in Mac OS X 10.2 (Changed in 10.2). Historically, the MacOS 9 MathLib (and CarbonLib) have exported a routine named "gamma()" that approximates the Gamma function. The many variants of Unix (BSD, Posix, etc.) have offered routines named "lgamma()" and "gamma()" both approximating the log of the absolute value of the Gamma function (N.B. not Gamma in its natural scale!) C99 attempts to untangle this by:
    1. introducing `tgamma()` to approximate the Gamma function in its natural scale called "true gamma"
    2. leaving `lgamma()` in place to approximate ln (| Gamma |).
    3. deprecating use of `gamma()`.

  We have chosen (1) and (2), but we also preserve the MacOS 9's MathLib/CarbonLib semantics for "gamma()" as an

approximation to the Gamma function in its natural scale. This ensures source and binary compatability for Carbon applications and only inconveniences legacy Unix applications that refer to "gamma()" and expect to obtain ln (| Gamma |). See Technical Q&A QA1143 for more information.

- Quiet and signaling NaNs have been changed in Mac OS X 10.2 (Changed in 10.2). FP_SNAN, FP_QNAN are deprecated in the final C99 spec. The new <math.h> defines them to be the standard FP_NAN.

Back to top

MultiProcessing

Facilities for symetric multiprocessing built in to Mac OS X.

- pthread_exit() from main() wasn't waiting for other threads to finish. This has been fixed. (r. 2888517).
- pthread_setcancelstate, pthread_setcanceltype returned wrong flags. This has been fixed. (r. 2653228).
- The pthread_setcancelstate and pthread_setcanceltype API's would crash if the second paramater was NULL. This has been fixed. (r. 2458277).

Back to top

Networking

This section discusses changes and new features in the networking services provided in Mac OS X 10.2.

AppleTalk

AppleTalk is a built-in network protocol that supports communications over AppleTalk networks.

- Fixed a problem where names written to the /etc/hostconfig file could include unreadable text for the APPLETALK_HOSTNAME entry. Now, an alternate encoding scheme is used. (r. 2911094).
- Modified the NBP tuple unpacking code to fix a potential crashing problem which could result if the code read past the end of the reply buffer. (r. 2903925).
- When you enable AppleTalk and apply the change, if there is an AppleTalk zone name, the name will appear in the popup. (r. 2886131).
- Fixed a problem with the ADSP in the kernel such that a heavily loaded Mac Manager server (250+ clients) would panic the system. (r. 2822909).
- Modified checkATStack function so that the RUNNING result is only returned when the AppleTalk stack has completed loading. Previously, the checkATStack function would return the RUNNING result while the AppleTalk stack was still being loaded. (r. 2806488).
- Fixed the AppleTalk Framework header file at_proto.h to include the "C" wrapper for use with C++ code. (r. 2776643).
- Fixed a bug in AppleTalk routing which caused a multihomed system to use the wrong source address when sending packets out the non-default interface. (r. 2741178).
- Fixed the problem that an AFP Server could not be seen on any active connection on a 4-port Ethernet card. (r. 2672538).
- Fixed a problem such that changes in AppleTalk state were not being reported. (r. 2561461).
- Reduced the amount of memory consumed by the atp_open call. (r. 2882827).

Back to top

BSD Networking APIs.

The BSD networking APIs included with Mac OS X.

- ftpd has been updated to lukemftpd 1.1. (r. 2889211).
- Correction to handling of IP_MULTICAST_IF socket option. (r. 2847168).

- Removed uucpd from Mac OS X since it was not used. (r. 2780797).
- Fixed the Ethernet demultiplexor to recognize the DLIL_DESC_SNAP socket option to allow a raw Ethernet endpoint to process incoming Ethernet SNAP packets. (r. 2953085).
- Fixed the Ethernet demultiplexor to handle incoming SAP and SNAP packets with the size field set for 1500. This problem only affected raw Ethernet sockets. (r. 2952858).
- Implemented support for an application to send and receive ICMP echo packets for a normal user with no administrative privileges. Instead of opening a raw IP socket like this: socket(PF_INET, SOCK_RAW, IPPROTO_ICMP), you can now open a non-privileged datagram oriented socket: socket(PF_INET, SOCK_DGRAM, IPPROTO_ICMP). (r. 2655317).

Back to top

CFNetwork

CFNetwork provides various network facilities using Core Foundation data types and calling conventions. Most notably, it provides facilities for building and processing HTTP requests, building and using various kinds of direct socket connections, and advertising or discovering network services shared via Rendezvous. It is available as part of the Core Services framework.

- Added a comment to "CFSocketStream.h" to clarify that the TLS/SSL setting can be negotiated upwards after a stream has been opened. (r. 2915430).
- CFReadStreamOpen no longer fails if the first character of the host name is a digit (for example, "3ps.go.com"). (r. 2891527).
- Added support for SOCK v5 firewalls. (r. 2833743).
- Added support for SOCK v4 firewalls. (r. 2833740).
- Added CFNetwork APIs to browse for and register services using Rendezvous. See "CFNetwork/CFNetServices.h" for more information. (r. 2833543).
- Previously CFNetwork would always send the port number in the "Host" header when requesting a URL. That caused problems for some servers. Now CFNetwork only puts the port number in the "Host" header if it's not 80. (r. 2825110).
- An HTTP stream can now accept its payload data from a stream. (r. 2706564).
- Added support for persistent HTTP connections. (r. 2705578).
- Added support for HTTP digest authentication. (r. 2705542).
- Fixed a bug related to asynchronous downloads. CFNetwork was occasionally signalling that bytes were available when they weren't. This happened most often when the client calls CFReadStreamRead multiple times in response to a single signal of kCFStreamEventHasBytesAvailable. (r. 2973276).
- CFNetwork now returns an error, as opposed to crashing, if you pass it a request with high-bit set characters. (r. 2969566).
- Fixed a problem where passing a bad host name to CFNetwork would sometimes cause it to crash. (r. 2923325).
- Fixed a bug that caused CFReadStreamCopyProperty to return NULL intermittently when asking for the property kCFStreamPropertyHTTPResponseHeader. (r. 2916653).
- Fixed a bug that caused CFReadStreamRead to hang. (r. 2915455).
- HTTP headers transmitted at the end of a chunked response are now properly picked up and added to the response headers; before, they were misparsed and could cause crashes. (r. 2912738).
- Fixed an incompatibility with web servers that expect three CRLF pairs after a request whose Content-Length header is 0. (r. 2896595).
- Fixed a problem where CFNetwork would consider %00 in the host name portion of the URL to be the end of the URL. (r. 2890529).
- Fixed an HTTP/1.1 problem where the server responds with a 100-continue response, but CFNetwork would not continue. This is most common when sending POST requests. (r. 2851403).
- Fixed a problem that could cause the kCFStreamEventHasBytesAvailable event to be delivered after the kCFStreamEventEndEncountered event. (r. 2840545).

Back to top

DHCP and BOOTP

DHCP and BOOTP servers provide automated network address configuration services. This section discusses changes in Mac OS X's facilities for accessing DHCP and BOOTP servers.

- Changes have been implemented to support interfaces being removed from the network. (r. 2879871).
- DHCP now correctly sets the "secs" field of request packets to the number of seconds since DHCP was started. Previously that field was erroneously always set to 0, which caused problems for some DHCP servers. (r. 2857975).
- The DHCP client now asks for the LDAP configuration option (95). (r. 2823022).
- Changes to support single-link multihoming (multiple network configurations using the same network card). (r. 2821104).
- It is now possible to configure the DHCP server (via properties in the "/config/dhcp" NetInfo node) to hand out addresses only to clients with known MAC addresses. (r. 2804638).
- DHCP now exports a public API for requesting non-standard DHCP options and for accessing the lease time. See <SystemConfiguration/DHCPClientPreferences.h> and <SystemConfiguration/SCDynamicStoreCopyDHCPInfo.h> for more details. (r. 2794670).
- The DHCP client now tries harder to get an address, which improves compatibility with spanning tree switches. (r. 2785257).
- When configured to use DHCP, the system now allocates a link local address quicker and then continues to probe for a DHCP server. This allows Rendezvous to function sooner after waking up a machine on a network without a DHCP server. (r. 2901898).
- The DHCP client now does not pass a hostname option to the server if the computer's name contains characters that would be invalid according to the DHCP specification. (r. 2822711).
- The DHCP server now supports multiple boot images. (r. 2477608).

Back to top

Internet Config

Internet Config provides an API to access and manage Internet-related preferences.

- Internet Config has been moved out of Carbon and into the new HIServices framework (a sub-framework of the Application Services framework). This makes it easier to access from non-Carbon applications. (r. 2786960).
- Internet Config now correctly handles URLs with pound ('#') characters in them. (r. 2783916).
- Calling ICEditPreferences with a proxy key (for example, kICUseFTPProxy) now opens the Network preferences panel, where you can edit the proxy setting. Previously it would open the Internet preferences panel. (r. 2779246).
- Updated the IC mappings to map .jar files to JAR Launcher. (r. 2747714).
- As the system currently provides no user interface to edit file mappings, calling ICEditPreferences with the kICMapping key now returns an error. (r. 2940580).
- Fixed a problem where, if you requested a valid helper application preference, Internet Config would return the correct information but return an error result of icPrefNotFoundErr rather than noErr. (r. 2919430).
- IC now handles arbitrary sized URLs passed to ICLaunchURL. This, combined with a fix to the "open location" scripting addition, means that you can now launch URLs longer than 255 characters from scripts. (r. 2871281).
- IC now correctly handles user data added to kICMapping preference entries. (r. 2690150).

Back to top

NSL

The Network Services Location (NSL) Manager provides a protocol-independent way for applications to discover network services that are available in the local network.

- NSLPrepareRequest now allows you to pass in NULL for the contextPtr. (r. 2886211).
- The Connect To Server dialog now displays UTF-8 strings containing Japanese characters correctly. (r. 2794449).
- Now, NSL will launch Open Directory if it is not running when the Connect To Server dialog box is involked.(r. 2783362). .

- NSL only supports browsing Rendezvous services through its high level GUI API (NSLStandardGetURL). A user selecting a service located via Rendezvous will not show an IP address or URL. The URL gets generated after the item is selected and the dialog is dismissed..
- The Service Discovery plug-ins in Open Directory (Rendezvous, SLP, AppleTalk, SMB) are initialized on demand. This can cause a delay seeing the Neighborhoods (Scopes, Workgroups, Zones) the first time Connect To Server is invoked after a restart. .
- NSL now has a concept of a "Simple Network". If the set of neighborhoods found is equal to the set of "default" neighborhoods, NSL will return 0 neighborhoods. Searching for services in the "default" will show a single list of services from all service discovery protocols. This can be seen in Connect To Server when switching between a network that has more than one Scope, Zone, or Workgroup, to a simple network that is unmanaged.

Back to top

OpenTransport

OpenTransport is a set of APIs used by CFM-Carbon applications to access TCP/IP services.

- Fixed Open Transport PPP so that asynchronous PPP endpoints which register to receive notification of PPP events, receive the standard notification events. (r. 2756632).
- Implemented a fix for an asynchronous OTLookupName call so that the call get cleaned up properly if the mapper provider is closed before completing the lookup. (r. 2652603).
- Changes in the system's IP network configuration now close inet providers and generate kOTProviderIsClosed events on their notifiers. When that happens, the provider is already internally closed and OTCloseProvider is the only legal API call that can be made on it. (r. 2446614).
- Fixed a memory leak with the OTConnect call as implemented in Carbon Open Transport. (r. 2897246).
- Fixed the OTInetAddressToName call so that it would not used cached data from a previous OTInetStringToAddress call. A previous OTInetStringToAddress might resolve a name which points to multiple hosts with unique IP addresses and host names. (r. 2875786).
- Fixed a bug where calling OTIoctl(..., I_OTConnect, ...) to establish a PPP connection could crash the system. (r. 2751639).
- The Jaguar WWDC seed contained a version of OpenTransport that was a bit too aggressive in closing providers when a configuration change occurred. The GM version of Jaguar takes an approach that is friendlier to the multihomed environment of OS X. When a configuration change occurs, OpenTransport will only close inet endpoints that were bound to an address that is no longer present on the system.
- Fixed a problem in which multiple calls to OTRcvUData for the same packet would corrupt the TUnitData structure.

Back to top

SLP

Service Location Protocol (SLP) is the protocol used by OS 9 and OS X to discover certain services on IP networks. Apple now recommends that developers should use Rendezvous instead of SLP.

- After switching locations, all services that were registered via SLP are deregistered. It is up to the registering application to make sure that they re-register themselves after such an occurrence. (r. 2896398).
- SLP now supports more than 2K of scope information. This was done by using the field in the standard SLP header for marking the packet as having overflow data that needs to be re-requested by the client with a TCP connection. (r. 2773380).
- On Mac OS X Server, the SLP regfile is now deleted on each reboot to keep old registrations from persisting. (r. 2718327). (r. 2772412).
- Unplugging the net cable while NSL is doing a search impedes searches, this has been fixed. (r. 2739157).
- Improved error reporting for SLP when networking is down. (r. 2732590).
- A problem where a crash could occur when repeatedly running a service search and cancelling it before it had completed has been corrected. (r. 2600622).
- A number of memory leaks and crashing bugs have been corrected. (r. 2969856).
- A problem where the routine dsGetRecordList() could return erroneous information when called with a match type of eDSExact has been corrected. (r. 2949248).

- The routine dsGetRecordList() was returning duplicate results in most cases. This problem has been corrected. (r. 2949230).

Back to top

SNMP

Simple Network Management Protocol (SNMP) for Mac OS X.

- SNMP support has been added to Mac OS X 10.2. (r. 2561896).

Back to top

URL Access

The URL Access Manager provides application support for downloading data from or uploading data to a Universal Resource Locator (URL), with support for automatic decompression of compressed files.

- URL Access now deletes partially uploaded files when the transfer fails. (r. 2891692).
- Fixed a bug that caused downloads to fail when encountering a redirection. (r. 2880313).
- Fixed a bug that caused URL Access to return an impossible error (kURLExtensionFailureError). (r. 2858557).
- Calling URLGetError() now returns the correct error code instead of always returning 404. (r. 2817022).
- After downloading a folder using URLDownload, caling URLGetCurrentState() now returns the correct value instead of 0. (r. 2804425).
- When specifying kURLDisplayAuthFlag, URLDownload will now only prompt you for a password if necessary. (r. 2800914).
- URL Access can now FTP download files with special characters in their names. (r. 2939783).
- Fixed a bug that would cause URLDownload to return -1 when posting to a certain HTTPS server. (r. 2906557).
- Fixed a crashing bug when downloading folders using URLDownload. (r. 2892959).
- Increased the number of allowable redirections from 1 to 4. (r. 2868534).
- Fixed a bug that would cause URL Access transactions to never complete because server was returning "100 Continue" responses. (r. 2855680).
- Fixed bug that caused URLNewReference to return +1 if URL had illegal characters. (r. 2843098).
- URL Access now sends notifications for kURLResourceFoundEvent, kURLUploadingEvent, and kURLDownloadingEvent. (r. 2813269).
- URLGetProperty for kURLResourceSize now returns the correct value instead of 0. (r. 2783715).
- Fixed a bug that caused FTP uploads to leave 0 length files when the disk was full. (r. 2752669).
- URL Access now properly handles "100 Continue" replies from HTTP 1.1 servers. (r. 2872047).

Back to top

Open Directory

Open Directory provides an abstraction layer designed to isolate clients of the Open Directory programming interface from the actual implementation of a directory system.

- Support for password server authentication methods has been added. (r. 2932522).
- Added Open Directory support for managed client types. (r. 2878911).
- Added LDAPv3 Open Directory plug-in with read/write support. (r. 2878721).
- A new error code, eDSSchemaError, has been assimilated into the Directory Services APIs. This error is returned when a Directory Services operation is attempted that violates the schema of the directory service being accessed. (r. 2871811).
- Previously, calls dsGetRecordList() with more than one recName could fail under certain conditions by only providing information in the first record when more information was available. This error has been corrected. (r.

2863112).

- These methods are supported with either Password Server or the legacy Authentication Manager ("tim"). (r. 2861932).
- Added a new BSD Configuration File Open Directory plug-in. (r. 2852585).
- A problem where the dsAddAttributeValue routine would not accept a NULL tDataNodePtr parameter has been corrected. Now, a NULL tDataNodePtr means 'add an attribute without an initial value'. (This works on NetInfo, but may not be supported on some directory systems, such as LDAP.) (r. 2852078).
- The new kDS1AttrPicture standard type constant has been added to directory services. This constant can be used to access the picture attribute associated with each user. (r. 2846085).
- The new Open Directory standard type constants: kDSNAttrPort and kDSNAttrLocation were added to the Open Directory constant database to support NSL/Rendezvous lookup. (r. 2829443).
- Previously Open Directory required the calling process to be running as root before any calls could be made to a node that modified a directory. For example, dsSetRecordName() and dsCreateRecordAndOpen() would unconditionally fail with a eDSPermissionError if the calling process was not running as root. This restriction has been relaxed so that non-root processes can authenticate to a node reference using dsDoDirNodeAuth in order to make changes. (r. 2778611).
- Problems where some Open Directory lookup operations could fail for user names with some combinations of characters (usually underscores mixed with brackets) in their names have been corrected. (r. 2762543).
- The Open Directory daemon now registers a key inside of the system configuration framework to indicate a restart of the daemon was experienced. This key is only temporary though and is intended to be used by configd. (r. 2759398).
- GetDirNodeInfo now checks for buffers passed which are too small (similar to what GetRecordList does). When a too small buffer is detected a eDSBufferTooSmall error is returned. (r. 2975226).
- A new attribute, kDS1AttrTypeAdminLimits, was added to support Workgroup Manager admin limits. (r. 2954158).
- Added new error codes to support password policy features of the Password Server: eDSAuthQualityCheckFailed, eDSAuthPasswordTooShort, eDSAuthPasswordTooLong, eDSAuthPasswordNeedsLetter, and eDSAuthPasswordNeedsDigi (r. 2951855).
- Added constants, kDS1AttrPresetUserIsAdmin and kDS1AttrHomeLocOwner, to describe new attributes. These attributes describe if a preset user is admin (or not) and the owner associated with a workgroup's home directory. (r. 2945978).
- Support for Password Server was added to both the NetInfo and the LDAPv3 plug-ins. .

Back to top

QuickTime

QuickTime provides various multimedia services for the Mac OS, including the ability to display movies and facilities for the translation and display of various audio and visual data file formats.

- Mac OS X 10.2 ships with QuickTime 6. New features, changes, and bug fixes in QuickTime 6 are documented in the QuickTime 6 documentation. .

Back to top

Security

This section discusses changes and new features in the security services provided in Mac OS X 10.2.

Kerberos

Kerberos is a networking protocol designed to allow strong client/server authentication.

- Support for client commands: kinit, klist, kdestroy, kpasswd was added to system.
- The Kerberos library provided with Mac OS X 10.2 has been simplified so it is more convenient for programmers to use. (r. 2829684).
- The Kerberos ticket manager is installed with Mac OS X 10.2. (r. 2949384).
- A problem where attempting to use a Kerberos command line tool when no preference file existed could create an

empty preferences file has been corrected. (r. 2824235).

- A problem where the kinit tool did not do any error checking on the format of the renew parameter has been corrected. (r. 2816142).

Back to top

Security Framework

The security framework exports a security APIs for use with Mac OS X.

- Add a function to the TP module to enable developers and sysadmins to generate Cert Signing Requests. (r. 2909233).
- CSP, TP, CL modules are now thread safe. (r. 2897036).
- Added SSLSetCertificate() for server side SSL support. (r. 2886598).
- Added expanded per-cert error reporting to TP Policy verification. The evidence chain returned by CSSM_TP_CertGroupVerify now has policy-specific error codes reported in the returned evidence chain. (r. 2878647).
- The Keychain database schema was updated to facilitate storage of certificates per the CDSA spec. (r. 2858506).
- Added SecureTransport APIs. (r. 2855500).
- Fixed a problem where KCAddGenericPassword would return a CDSA error code when a write to the KeyChain file failed because of insufficient permissions. Now the OSErr wrPermErr is returned instead. (r. 2775734).
- Keychains now support storing public as well as protected items. This is part of allowing certificates to be stored in keychains. (r. 2304871).
- Additional X509 root certificates have been included. (r. 2973839).
- The trusted anchor certificates are now stored in a (root write-protected) keychain file. This makes the set of anchor certs easier to update by system administrators and via Software Update. (r. 2950315).
- New tool /usr/bin/certtool added to: (a) Create a key pair and a self-signed root cert, add them to a keychain. Used for early debugging of server-side code. (b) Create a key pair and an associated Cert Slgning Request (CSR). A CSR is the means by which one requests a cert authority (CA) such as Verisign to provide a "real" cert which is signed by a recognized CA. (c) Import a cert obtained from the CA to whom one sent the CSR generated in step 2, add it to a keychain, and bind the cert with the keypair generated in step 2. This is the "real world" procedure for obtaining an SSL cert for a server which uses SecureTransport. (r. 2911384).
- CSSMOID_SHA1 has been added to oidsalg.h. This is a common industry-standard value but not required by the CDSA spec. (r. 2901582).
- Brought the names of the fields in the cssm_context structure in cssmtype.h up to date with the CDSA 2.0 spec. This does not affect binary compatibility. (r. 2900877).
- Added the constant CSSMERR_DL_INVALID_DL_HANDLE which was missing from cssmerr.h. (r. 2900467).
- Added the constant CSSMERR_CSSM_FUNCTION_NOT_IMPLEMENTED which was missing from cssmerr.h. (r. 2900465).
- gGuidCssm was declared external in Security/cssmapple.h, but was not actually exported. This symbol is now exported correctly. (r. 2892955).
- Fixed a memory leak when calling KCFindInternetPassword. (r. 2843883).
- Added support for smart cards that comply with the JavaCard 2.1 standard. (r. 2824101).
- New Security APIs added to simplify use of the underlying Common Data Security Architecture (CDSA). (r. 2887937).
- New APIs were added to Security.framework for working with X.509 certificates.(r. 2887929) (r. 2892437).
- Added certificate support APIs to Security.framework.(r.2846641). (r. 2866950).
- The list of trusted root certificates is now user modifiable, so that additional roots can be added, or existing roots marked as not trusted. (r. 2824109).
- Checking compliance with the CDSA 2.1 specification by running through the CDSA conformance test suite developed by UniSoft. (r. 2824077).
- Fixed memory leaks when calling KCFindGenericPassword and KCFindAppleSharePassword. (r. 2953620).
- Data types in the layered services of Security.framework (such as KCRef) are now true Core Foundation objects. (r. 2904268).

- If the default keychain has been deleted but other keychains exist, the create/select keychain dialog is now presented instead of the create keychain dialog so the user can choose an alternate keychain. (r. 2853839).
- The Carbon SecurityHI framework now uses the Cocoa SecurityHICocoa framework. This takes advantage of the ability to mix Carbon and Cocoa UI in Jaguar. (r. 2851842).
- Fixed a problem where retrieving a 'cusi' (has custom icon) attribute would return the wrong value. (r. 2748773).
- Fixed a problem where NSKeyChainItem could return a bogus value for a numeric attribute stored in a keychain. .
- A no-pad option was added for raw RSA encryption and signing. Before, PKCS1-style padding was assumed. (r. 2868328).
- Basic validity checking of incoming KeyAttr flags (RETURN_DATA, ALWAYS_SENSITIVE, NEVER_EXTRACTABLE) is now done at this level rather than by the CSP. (r. 2879872).
- CSSM_DigestDataClone has now been implemented. (r. 2890980).
- Fixed a bug that would cause AuthorizationExecuteWithPrivileges to fail if run more than once in the same login session. (r. 2824995).

Back to top

Text

This section discusses changes and new features in the text handling facilities provided in Mac OS X 10.2.

Multilingual Text Editor

MultiLingual Text Editing (MLTE) supplies a C language interface for creating and editing Unicode text documents. Although MLTE may appear to be functionally similar to TextEdit in many ways, MLTE provides many features that are not provided in TextEdit.

- Mac OS X 10.2 includes MLTE version 1.5. (r. 2908662).
- Mouse wheel events are now supported by the MLTE APIs. (r. 2526078). (r. 2875253).
- Improved performance of TXNObjects in CG transformations. (r. 2864521).
- Added support in MLTE to honor the "scroll to here" setting. (r. 2822866).
- New API TXNScroll added to MLTE (MacTextEditor.h). (r. 2822193).
- MLTE now supports printing with a CGContext. (r. 2818402).
- TXNAdjustCursor now behaves correctly if the second parameter, ioCursorRgn, is NULL. (r. 2812991).
- A problem where Drag scrolling would only occur while the mouse is moving has been corrected. (r. 2763883).
- A problem where the grow box area wasn't appearing in an MLTE pane object when it was requested, by way of the kTXNDrawGrowIconMask, has been corrected. (r. 2677645).
- The new TXNControlTag, kTXNVisibilityTag, has been made available in MacTextEditor.h for controlling the drawing state of MLTE objects. Under normal circumstances, each time some type attributes are set, the TXNObject will redraw. However, with this tag set, MLTE panes will no longer draw every time a type attribute is set. Setting this tag will not cause a refresh of the TXNObject pane, and unsetting this tag will not force an update. (r. 2936678).
- The new kTXNSingleLevelUndo tag has been added to MacTextEditor.h. This tag allows a choice for MLTE clients between supporting a relatively expensive multiple undo or a simple single undo state. (r. 2930283).
- URL support has been added to MLTE. (r. 2927249).
- MLTE now fully supports monostyled text with kTXNMonostyledTextMask (r. 2689984).

Back to top

Text Encoding Converter

The Text Encoding Converter enables the conversion of text data from one encoding to another (e.g., Mac OS Roman to Unicode).

- The Eject Symbol is now mapped to 0x8C as the char code in the MacKeyboard encoding and to the Corporate-Use-Area Unicode code point U+F804. (r. 2781691).
- Improvements to Text Encoding Conversion between Macintosh Korean and Unicode 2.1 or 3.0. (r. 2765765).
- TECGetMailTextEncodings and TECCountMailTextEncodings now supports Tier 3 languages. (r. 2969270).
- TEC support added for the following DOS encodings: kTextEncodingDOSGreek, kTextEncodingDOSBalticRim, kTextEncodingDOSLatin2, kTextEncodingDOSTurkish, kTextEncodingDOSIcelandic, and kTextEncodingDOSRussian. (r. 2842283).
- The new APIs GetTextEncodingFromScriptInfo and GetScriptInfoFromTextEncoding have been added to TEC. These APIs replace UpgradeScriptInfoToTextEncoding and RevertTextEncodingToScriptInfo where you pass NULL as the font name. (r. 2588276).
- Added support for converting to canonically composed Unicode (kUnicodeCanonicalCompVariant) and to two HFS Plus-specific variants (kUnicodeHFSPlusDecompVariant and kUnicodeHFSPlusCompVariant). kUnicodeHFSPlusDecompVariant is now the recommended way for requesting decompositions with HFS Plus exclusions, instead of using kUnicodeUseHFSPlusMapping.(r. 2715044). .
- Output of unnecessary direction overrides has been reduced when converting from MacArabic and MacHebrew to Unicode. (rr. 2544128, 2972010, 2967339).

Back to top

Text Services Manager

The Text Services Manager provides facilities for applications to communicate with various text processing utilities that provide services such as special text input methods, spell checking, hyphenation, etc.

- TSM support for Glyph Input has been added. (r. 2883312).
- Additional support for general keyboard navigation has been added. (r. 2858720).
- Improved performance/robustness of input methods and application switching. (r. 2825097).

Back to top

TextEdit

TextEdit is a simple text-processing service that provides basic text-handling for small amounts of text. It was introduced with the original version of Macintosh system software. An enhanced text-processing service, the Multilingual Text Engine (MLTE), was introduced in Mac OS 9. Most TextEdit functions are not recommended for use on Mac OS X, so you should use MLTE instead of TextEdit.

- A problem where auto-scrolling would stop if the mouse were moved outside of the window containing the text edit field has been corrected. (r. 2836264).
- Fixed a problem with TextEdit where 2-byte text would not have the gray baseline displayed to help distinguish it from unconfirmed text. (r. 2793871).

Back to top

Unicode Utilities

The Unicode Utilities provides a collection of Unicode text handling routines.

- Calling UCCompareText() with kUCCollateDigitsAsNumberMask no longer reads text off the end of a block. (r. 2879176).
- The routine UCGetCollationKey now returns actualKeySize with noErr instead of paramErr when collationKey is NULL. (r. 2843282).
- Changed UCKeyTranslate to return noErr and zero length string instead of param err when the key code is out of range. (r. 2841440).
- The routine LocaleGetRegionLanguageName no longer returns an empty string. (r. 2920226).
- UCGetCollationKey now produces smaller collation keys whenever possible. NOTE: the collation keys produced by UCGetCollationKey() are not guaranteed to be compatible accross system versions. (r. 2843266).

Back to top

Downloadables

Acrobat version of this Note (460K) Download

Back to top
