Appendix A: CDE Application Programming Interfaces

This appendix lists the CDE libraries and header files that contain the CDE APIs.

Desktop Services Library (libDtSvc)

Desktop Initialization APIs. The desktop services library must be initialized with DtAppInitialize() or DtInitialize() before an application can use the APIs for action invocation, data typing, drag and drop, screen saving, session management, or workspace management.

- #include <Dt/Dt.h>
- Dt(5): Miscellaneous desktop definitions.
- DtAppInitialize(3), DtInitialize(3): Desktop services library initialization functions.

Action Invocation APIs. These APIs provide applications access to the desktop action database to query action attributes and to invoke actions. The DtActionInvoke() function selects the correct desktop action to invoke based on its arguments and actions in the database. The DtDbLoad() and DtDbReloadNotify() functions apply to the shared database for actions and data types.

- #include <Dt/Action.h>
- DtAction(5): Action service definitions
- DtActionCallbackProc(3): Notifies application that the status of an action has changed
- DtActionDescription(3): Obtains the descriptive text for a given action
- DtActionExists(3): Determines if a string corresponds to an action name
- DtActionIcon(3): Gets the icon information for an action
- DtActionInvoke(3): Invokes a CDE action
- DtActionLabel(3): Gets the localizable label string for an action
- DtDbLoad(3): Loads the actions and data types database
- DtDbReloadNotify(3): Registers callbacks for changes to actions and data types database.

Data Typing APIs. Data typing APIs provide applications with access to the desktop data type database to query data type attributes and to determine the data type of files, buffers, and data.

- #include <Dt/Dts.h>
- DtDts(5): Provides data typing definitions
- DtDtsBufferToAttributeList(3): Gets a list of data attributes for a byte stream
- DtDtsBufferToAttributeValue(3): Gets a single data attribute value for a byte stream
- DtDtsBufferToDataType(3): Gets the data type for a byte stream
- DtDtsDataToDataType(3): Gets the data type for a set of data
- DtDtsDataTypeIsAction(3): Determines if the data type is an action
- DtDtsDataTypeNames(3): Gets a list of available data types
- DtDtsDataTypeToAttributeList(3): Gets a list of attributes for a data type
- DtDtsDataTypeToAttributeValue(3): Gets an attribute value for a specified data type
- DtDtsFileToAttributeList(3): Gets a list of attributes for a file
- DtDtsFileToAttributeValue(3): Gets a specified attribute value for a file
- DtDtsFileToDataType(3): Gets a data type for a file
- DtDtsFindAttribute(3): Gets a specified list of data types
- DtDtsFreeAttributeList(3): Frees a list of data attributes
- DtDtsFreeAttributeValue(3): Frees a data attribute value
- DtDtsFreeDataType(3): Frees a data type pointer to memory
- DtDtsFreeDataTypeNames(3): Frees a list of data type names
- DtDtsIsTrue(3): Returns a Boolean value associated with a string
- DtDtsLoadDataTypes(3): Loads and initializes the data types database
- DtDtsRelease(3): Frees memory associated with the data types database
- DtDtsSetDataType(3): Sets the data type of a directory.

Drag and Drop APIs. The drag and drop APIs are a convenience and policy layer on top of Motif 1.2 drag and drop. The drag and drop APIs manage the configuration and appearance of drag icons, define a transfer protocol for buffers, enable animation upon drop, provide enumeration of targets for text and file transfers, allow dual registration of text widgets for text and other data, and provide prioritized drop formats.

- #include <Dt/Dnd.h>
- DtDnd(5): Provides drag and drop definitions
- DtDndCreateSourceIcon(3): Creates a drag source icon
- DtDndDragStart(3): Initiates a drag
- DtDndDropRegister(3): Specifies a drop site
- DtDndDropUnregister(3): Deactivates a drop site.

Screen Saver APIs.

- #include <Dt/Saver.h>
- DtSaver(5): Provides screen saver definitions
- DtSaverGetWindows(3): Gets the list of windows for drawing by a screen saver application.

Session Management APIs.

- #include <Dt/Session.h>
- DtSession(5): Provides session management services definitions
- DtSessionRestorePath(3): Gets a path name for the application's state information file
- DtSessionSavePath(3): Gets a path name for saving application state information.

Workspace Management APIs. The workspace management APIs provide functions to access and modify workspace attributes and to request notification of workspace changes.

- #include <Dt/Wsm.h>
- DtWsm(5): Workspace manager definitions
- DtWsmAddCurrentWorkspaceCallback(3): Adds a callback to be called when the current workspace changes
- DtWsmAddWorkspaceFunctions(3): Adds workspace functions for a window
- DtWsmAddWorkspaceModifiedCallback(3): Adds a callback to be called when any workspace is changed
- DtWsmFreeWorkspaceInfo(3): Frees workspace information
- DtWsmGetCurrentBackdropWindow(3): Gets the backdrop window for the current workspace
- DtWsmGetCurrentWorkspace(3): Gets the current workspace
- DtWsmGetWorkspaceInfo(3): Gets detailed workspace information
- DtWsmGetWorkspaceList(3): Gets the names of the currently defined workspaces
- DtWsmGetWorkspacesOccupied(3): Gets the workspaces in which a window resides
- DtWsmOccupyAllWorkspaces(3): Puts a window into all workspaces
- DtWsmRemoveWorkspaceCallback(3): Removes a workspace callback
- DtWsmRemoveWorkspaceFunctions(3): Removes a window's workspace function
- DtWsmSetCurrentWorkspace(3): Sets the current workspace
- DtWsmSetWorkspacesOccupied(3): Sets the workspaces in which a window resides.

Help Widget Library (libDtHelp)

Help Utility APIs. These APIs are used to manage application help.

- #include <Dt/Help.h>
- DtHelp(5): Help services definitions
- DtHelpReturnSelectedWidgetId(3): Selects a widget or gadget
- DtHelpSetCatalogName(3): Assigns the name of the message catalog to use for help services

HelpDialog Widget API. The DtHelpDialog widget provides users with the functionality for viewing and navigating structured online information (CDE help volumes). This functionality includes text and graphics rendering, embedded hypertext links, and various navigation methods to move through online help information. The widget supports rendering of CDE help volumes, system manual pages, text files, and character string values.

- #include <Dt/HelpDialog.h>
- DtHelpDialog(5): DtHelpDialog definitions
- DtCreateHelpDialog(3): Creates a general DtHelpDialog widget
- DtHelpDialog(3): The DtHelpDialog widget class.

HelpQuickDialog Widget APIs. The DtHelpQuickDialog widget provides users with the same functionality as the DtHelpDialog widget. The difference here is that the functionality is for the quick dialog widget.

- #include <Dt/HelpQuickD.h>
- DtHelpQuickD(5): DtHelpQuickDialog definitions
- DtCreateHelpQuickDialog(3): Creates a DtHelpQuickDialog widget
- DtHelpQuickDialog(3): The DtHelpQuickDialog widget class
- DtHelpQuickDialogGetChild(3): Gets the child of a DtHelpQuickDialog widget.

Terminal Widget Library (libDtTerm)

Terminal Widget APIs. The DtTerm widget provides the core set of functionality needed to emulate an ANSI X3.64-1979- and ISO 6429:1992(E)-style terminal, such as the DEC VT220. This functionality includes text rendering, scrolling, margin and tab support, escape sequence parsing, and the low-level, operating-system-specific interface required to allocate and configure a pty or streams pseudoterminal device and write to the system's utmp device.

- #include <Dt/Term.h>
- DtTerm(5): DtTerm widget definitions
- DtCreateTerm(3): Creates a DtTerm widget
- DtTerm(3): DtTerm widget class
- DtTermDisplaySend(3): Sends data to a DtTerm widget's display
- DtTermInitialize(3): Prevents accelerators from being installed on a DtTerm widget
- DtTermSubprocReap(3): Allows a DtTerm widget to clean up after subprocess termination
- DtTermSubprocSend(3): Sends data to a DtTerm widget's subprocess.

Desktop Widget Library (libDtWidget)

Editor Widget APIs. The DtEditor widget supports creating and editing text files. It gives applications running in the desktop environment a consistent method for editing text data. The widget consists of a scrolled edit window for text, dialogs for finding and changing text, and an optional status line. Editing operations include finding and changing text, simple formatting, spell checking, and undoing the previous editing operation.

- #include <Dt/Editor.h>
- DtEditor(5): Editor widget definitions
- DtCreateEditor(3): Creates a DtEditor widget
- DtEditor(3): DtEditor widget class
- DtEditorAppend(3): Appends data to a DtEditor widget
- DtEditorAppendFromFile(3): Appends data from a file into a DtEditor widget
- DtEditorChange(3): Changes one or all occurrences of a string in a DtEditor widget
- DtEditorCheckForUnsavedChanges(3): Reports whether text has been edited
- DtEditorClearSelection(3): Clears the primary selection in a DtEditor widget
- DtEditorCopyToClipboard(3): Copies the primary selection in a DtEditor widget to the clipboard
- DtEditorCutToClipboard(3): Copies the primary selection in a DtEditor widget to the clipboard and deletes the selected text
- DtEditorDeleteSelection(3): Deletes the primary selection in the DtEditor widget
- DtEditorDeselect(3): Deselects the current selection in a DtEditor widget
- DtEditorDisableRedisplay(3): Temporarily prevents visual update of a DtEditor widget
- DtEditorEnableRedisplay(3): Forces the visual update of a DtEditor widget
- DtEditorFind(3): Searches for the next occurrence of a string in a DtEditor widget
- DtEditorFormat(3): Formats all or part of the contents of a DtEditor widget
- DtEditorGetContents(3): Retrieves the contents of a DtEditor widget
- DtEditorGetInsertionPosition(3): Retrieves the position of the insert cursor in a DtEditor widget
- DtEditorGetLastPosition(3): Retrieves the position of the last character in a DtEditor widget
- DtEditorGetMessageTextFieldID(3): Retrieves the widget ID of the message text field in the DtEditor status line
- DtEditorGetSizeHints(3): Retrieves sizing information from a DtEditor widget
- DtEditorGoToLine(3): Moves the insert cursor for a DtEditor widget to a specified line
- DtEditorInsert(3): Inserts data into a DtEditor widget
- DtEditorInsertFromFile(3): Inserts data from a file into a DtEditor widget
- DtEditorInvokeFindChangeDialog(3): Displays the DtEditor widget dialog for searching and replacing text
- DtEditorInvokeFormatDialog(3): Displays the DtEditor widget dialog for choosing formatting options
- DtEditorInvokeSpellDialog(3): Displays the DtEditor widget dialog for checking text for spelling errors
- DtEditorPasteFromClipboard(3): Inserts the clipboard selection into a DtEditor widget
- DtEditorReplace(3): Replaces a portion of the contents of a DtEditor widget
- DtEditorReplaceFromFile(3): Replaces a portion of the contents of a DtEditor widget with the contents of a file
- DtEditorReset(3): Resets a DtEditor widget to its default state
- DtEditorSaveContentsToFile(3): Saves the contents of a DtEditor widget to a file
- DtEditorSelectAll(3): Selects all the text in a DtEditor widget
- DtEditorSetContents(3): Places data into a DtEditor widget
- DtEditorSetContentsFromFile(3): Loads data from a file into a DtEditor widget
- DtEditorSetInsertionPosition(3): Sets the position of the insert cursor in a DtEditor widget
- DtEditorTraverseToEditor(3): Sets keyboard traversal to the edit window of a DtEditor widget
- DtEditorUndoEdit(3): Undos the last edit made to the text in a DtEditor widget.

ComboBox Widget APIs. The DtComboBox widget is a combination of a TextField and a List widget that provides a list of valid choices for the TextField. Selecting an item from this list automatically fills in the TextField with that list item.

- #include <Dt/ComboBox.h>
- DtComboBox(5): DtComboBox widget definitions
- DtCreateComboBox(3): Creates a DtComboBox widget
- DtComboBox(3): DtComboBox widget class
- DtComboBoxAddItem(3): Adds an item to the ComboBox widget
- DtComboBoxDeletePos(3): Deletes a DtComboBox item
- DtComboBoxSelectItem(3): Selects a DtComboBox item
- DtComboBoxSetItem(3): Sets an item in the DtComboBox list.

MenuButton Widget APIs. The DtMenuButton widget is a command widget that provides the menu cascading functionality of an XmCascadeButton widget. DtMenuButton can only be instantiated outside a menu pane.

- #include <Dt/MenuButton.h>
- DtMenuButton(5): DtMenuButton widget definitions
- DtCreateMenuButton(3): Creates a DtMenuButton widget
- DtMenuButton(3): DtMenuButton widget class

SpinBox Widget APIs. The DtSpinBox widget is a user interface control for incrementing and decrementing an associated TextField. For example, it can be used to cycle through the months of the year or days of the month.

- #include <Dt/SpinBox.h>
- DtSpinBox(5): DtSpinBox widget definitions
- DtCreateSpinBox(3): Creates a DtSpinBox widget
- DtSpinBox(3): DtSpinBox widget class
- DtSpinBoxAddItem(3): Adds an item to the DtSpinBox
- DtSpinBoxDeletePos(3): Deletes a DtSpinBox item
- DtSpinBoxSetItem(3): Sets an item in the DtSpinBox list.

Calendar Library (libcsa)

Calendar APIs. The Calendar APIs include functions for inserting, deleting, and modifying entries, functions for browsing and finding entries, and functions for calendar administration.

- #include <csa/csa.h>
- csacsa(5): Calendar and appointment services definitions
- csa_add_calendar(3): Adds a calendar to the calendar service
- csa_add_entry(3): Adds an entry to the specified calendar
- csa_call_callbacks(3): Forces the invocation of the callback functions associated with the specified callback lists
- csa_delete_calendar(3): Deletes a calendar from the calendar service
- csa_delete_entry(3): Deletes an entry from a calendar
- csa_free(3): Frees memory allocated by the calendar service
- csa_free_time_search(3): Searches one or more calendars for available free time
- csa_list_calendar_attributes(3): Lists the names of the calendar attributes associated with a calendar
- csa_list_calendars(3): Lists the calendars supported by a calendar service
- csa_list_entries(3): Lists the calendar entries that match all the attribute search criteria
- csa_list_entry_attributes(3): Lists the names of the entry attributes associated with the specified entry
- csa_list_entry_sequence(3): Lists the recurring calendar entries that are associated with a calendar entry
- csa_logoff(3): Terminates a session with a calendar
- csa_logon(3): Logs on to the calendar service and establishes a session with a calendar
- csa_look_up(3): Looks up calendar information
- csa_query_configuration(3): Determines information about the installed CSA configuration
- csa_read_calendar_attributes(3): Reads and returns the calendar attribute values for a calendar
- csa_read_entry_attributes(3): Reads and returns the calendar entry attribute values for a specified calendar entry
- csa_read_next_reminder(3): Reads the next reminder of the given type in the specified calendar relative to a given time
- csa_register_callback(3): Registers the callback functions to be invoked when the specified type of update occurs in the calendar
- csa_restore(3): Restores calendar entries from an archive file
- csa_save(3): Saves calendar entries into an archive file
- csa_unregister_callback(3): Unregisters the specified callback functions
- csa_update_calendar_attributes(3): Updates the calendar attributes values for a calendar
- csa_update_entry_attributes(3): Updates the calendar entry attributes
- csa_x_process_updates(3): Invokes a calendar application's calendar event handler.

ToolTalk Messaging Library (libtt)

ToolTalk Messaging API. This API provides functions for managing all aspects of ToolTalk messaging.

- #include <Tt/tt_c.h>
- Tttt_c(5): ToolTalk messaging definitions.

ToolTalk Toolkit APIs. The ToolTalk toolkit APIs are a set of higher-level interfaces to the ToolTalk messaging APIs. The ToolTalk toolkit APIs facilitate use of the desktop message set and the media exchange message set.

- #include <Tt/tttk.h>
- Ttttk(5): ToolTalk toolkit definitions
- ttdt_Get_Modified(3): Asks if any ToolTalk client has changes pending on a file
- ttdt_Revert(3): Requests a ToolTalk client to revert a file
- ttdt_Save(3): Requests a ToolTalk client to save a file
- ttdt_close(3): Destroys a ToolTalk communication endpoint
- ttdt_file_event(3): Uses ToolTalk to announce an event about a file
- ttdt_file_join(3): Registers to observe ToolTalk events on a file
- ttdt_file_notice(3): Creates and sends a standard ToolTalk notice about a file
- ttdt_file_quit(3): Unregisters interest in ToolTalk events about a file
- ttdt_file_request(3): Creates and sends a standard ToolTalk request about a file
- ttdt_message_accept(3): Accepts a contract to handle a ToolTalk request
- ttdt_open(3): Creates a ToolTalk communication endpoint
- ttdt_sender_imprint_on(3): Acts like a child of the specified tool
- ttdt_session_join(3): Joins a ToolTalk session
- ttdt_session_quit(3): Quits a ToolTalk session
- ttdt_subcontract_manage(3): Manages an outstanding request.

Motif Toolkit Libraries (libXm, libMrm, libUil)

Motif Widget API. The CDE Motif Widget API (Xm) consists of the Motif 1.2 widget library (libXm) with enhancements to existing functionality and bug fixes. The CDE Motif widget API maintains source compatibility and binary compatibility with Motif 1.2 applications.

- #include <Xm/XmAll.h>

Motif Resource Manager API. The Motif resource manager API (Mrm) creates widgets based on definitions contained in user interface definition files created by the user interface language (UIL) compiler. The Motif resource manager interprets the output of the UIL compiler and generates the appropriate argument lists for widget creation functions.

- #include <Mrm/MrmAppl.h>
- #include <Mrm/MrmDecls.h>
- #include <Mrm/MrmPublic.h>

Motif User Interface Language (UIL) API. The Motif UIL is a specification language for describing the initial user interface of a Motif application.

- #include <uil/Uil.h>
- #include <uil/UilDBDef.h>
- #include <uil/UilSymDef.h>
- #include <uil/UilSymGI.h>

▶ Go back to Article 1

▶ Go to Article 2

▶ Go to Table of Contents

▶ Go to HP Journal Home Page