WhiteboardDoc.tioga
Last Edited by: Donahue.pa on June 5, 1984 9:35:33 am PDT
Last Edited by: Subhana, June 6, 1984 2:09:37 pm PDT
Last Edited by: Widom, August 25, 1984 0:47:32 am PDT
Last Edited by: Donahue, June 13, 1986 9:43:44 am PDT
Jack Kent April 28, 1987 8:36:32 am PDT
Polle Zellweger (PTZ) July 15, 1986 4:02:57 pm PDT

Whiteboards

Jim Donahue

Release as [Cedar]<CedarChest6.0>Documentation>WhiteboardDoc.Tioga

XEROX

Xerox Corporation
Palo Alto Research Center
3333 Coyote Hill Road
Palo Alto, California 94304

For Internal Xerox Use Only

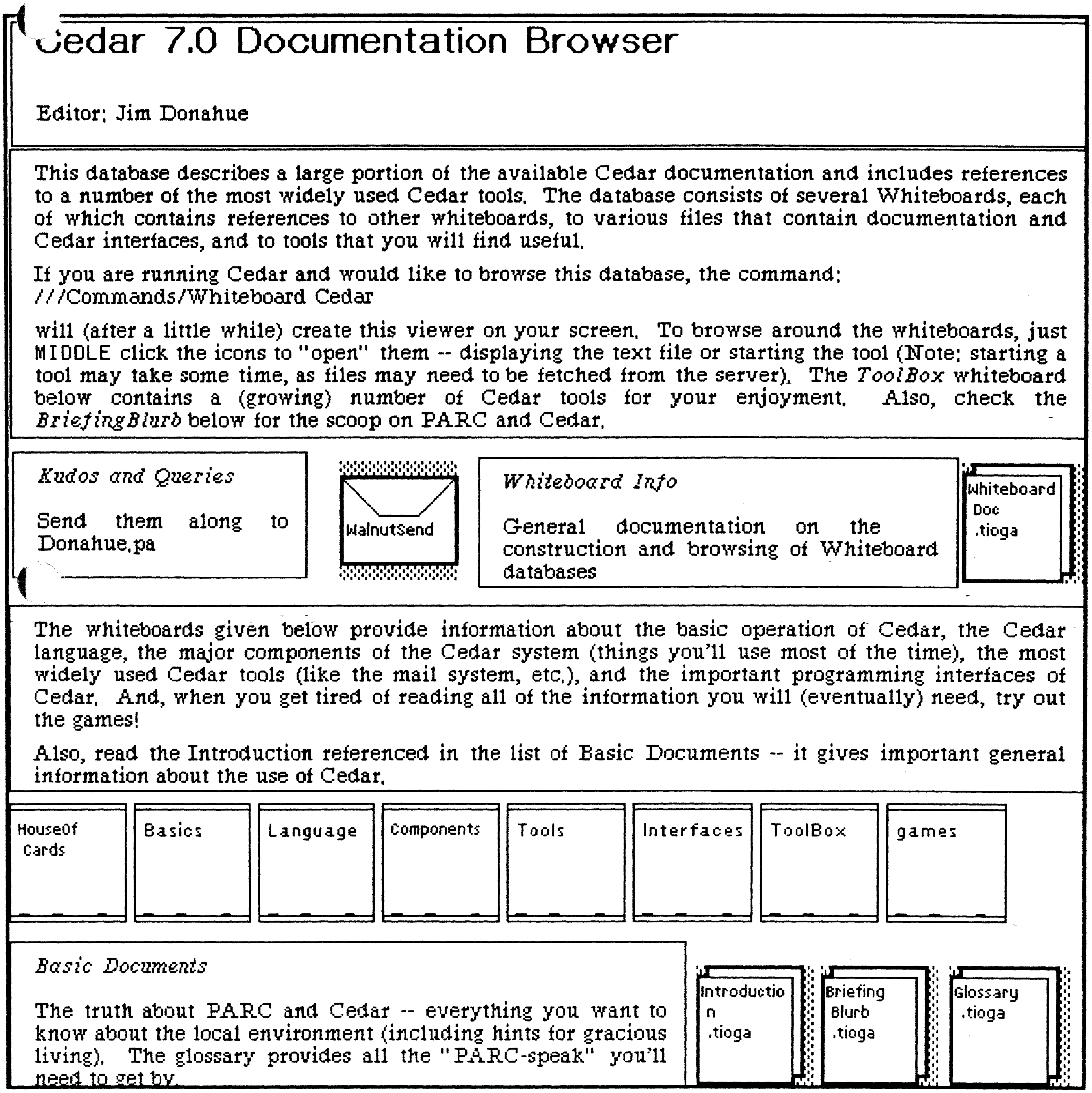
Introduction

The noun "Whiteboard" refers both to an object found on PARC office walls and to a particular kind of Cypress database. A Whiteboard database contains information about the positions of "text boxes" (filled with Tioga text) and Cedar icons that will be displayed on a Whiteboard viewer: the whiteboard code allows a user to move icons and text boxes in a viewer and save the resulting arrangement permanently in a database. The "Cedar Whiteboards" are a whiteboard database containing references to both documentation on the Cedar system and a large (and growing) collection of Cedar tools (so that you don't have to know that to read your mail, you need to run a program called Walnut). John Maxwell had the initial idea for a whiteboard database; Jim Donahue redesigned both the implementation and database schema and put together the Cedar Whiteboards; Jennifer Widom did a large part of the reimplementation.

Below we give a discussion of what a whiteboard database is, discuss creating and editing your own whiteboards, and then describe how you can browse the Cedar Whiteboards.

What a Whiteboard Is

A whiteboard database records the spatial arrangement of icons and text boxes in a whiteboard viewer. The text boxes can contain arbitrary Tioga text and the icons can be any of the normal Cedar icons you find on the screen, ie., they can be icons referencing text files, other whiteboards, entities in other databases (like Walnut message sets) and Cedar tools (like Walnut or the EditTool). The whiteboard browser is responsible for storing the arrangement of a whiteboard in a database and then later reconstructing the viewer from the stored information. Additionally, the browser implements the semantics of "opening" an icon on a whiteboard: this may involve opening a file (for text file icons), performing database manipulations (for opening other whiteboards) or loading a program (for tool icons). Below is a picture of the Whiteboard named Cedar that is the root of the Cedar Whiteboards.



Cedar 7.0 Documentation Browser

Editor: Jim Donahue

This database describes a large portion of the available Cedar documentation and includes references to a number of the most widely used Cedar tools. The database consists of several Whiteboards, each of which contains references to other whiteboards, to various files that contain documentation and Cedar interfaces, and to tools that you will find useful.

If you are running Cedar and would like to browse this database, the command:
///Commands/Whiteboard Cedar

will (after a little while) create this viewer on your screen. To browse around the whiteboards, just MIDDLE click the icons to "open" them -- displaying the text file or starting the tool (Note: starting a tool may take some time, as files may need to be fetched from the server). The *ToolBox* whiteboard below contains a (growing) number of Cedar tools for your enjoyment. Also, check the *BriefingBlurb* below for the scoop on PARC and Cedar.

Kudos and Queries

Send them along to Donahue.pa

Whiteboard Info

General documentation on the construction and browsing of Whiteboard databases

The whiteboards given below provide information about the basic operation of Cedar, the Cedar language, the major components of the Cedar system (things you'll use most of the time), the most widely used Cedar tools (like the mail system, etc.), and the important programming interfaces of Cedar. And, when you get tired of reading all of the information you will (eventually) need, try out the games!

Also, read the Introduction referenced in the list of Basic Documents -- it gives important general information about the use of Cedar.

Basic Documents

The truth about PARC and Cedar -- everything you want to know about the local environment (including hints for gracious living). The glossary provides all the "PARC-speak" you'll need to get by.

The Cedar Documentation Whiteboard

Operations on Whiteboards

Commands for examining whiteboards

A Whiteboard may be displayed using the Whiteboard command registered with the Commander:

Whiteboard name (displays the named whiteboard).

The whiteboard is looked up in the Cypress database named by a previous WBOpen command.

Whiteboards may be manipulated using the menu at the top of a whiteboard window, or by using mouse buttons with special meanings within the whiteboard itself. For the purposes of browsing, only middle-button selection within the window itself will be needed. Middle-button selection means "open": whenever the user points at an icon on a whiteboard and presses the middle button, a new window is opened on the screen displaying the entity represented by that icon.

Use middle-button selection to examine items on the Cedar whiteboard. Note that documents and programs open Tioga windows or start programs, respectively. Documents appear as the familiar Cedar icons for ducments, and programs as their normal icons as well. Whiteboards may be nested inside whiteboards: they appear as icons resembling phyusical whiteboards. Try browsing through some of these. Each time you select an icon, a new window will be created on the screen. You will require only the middle mouse button for this browsing. For browsing of the network structure of the whiteboards, try middle-button selection of a whiteboard while holding down the control key. All of the whiteboards contained in the selected whiteboard will be iconically displayed, with lines drawn to illustrate the structure. These new whiteboards can be opened or similarly expanded.

Commands for editing whiteboards

The Cedar Whiteboards are readOnly for almost everyone, so you won't be able to edit them (you can however, move things around on a whiteboard -- you just won't be able to make any permanent changes). If you plan to create or edit your own whiteboards, you will need to know about more than just browsing with middle-button selection.

The meaning of the menu buttons on a Whiteboard are as follows:

Freeze: Retains this whiteboard on the screen, rather than replacing it with another whiteboard, when middle-button selection would normally display something else in it.

Reset: Restores a whiteboard to its state before any edits were made. This may take some time, since a Reset forces Cypress to discard any pages of the database that it may have cached (to guarantee that the display is consistent with the most recent contents of the database).

AddSelected: You select an icon somewhere at the bottom of the Cedar Viewers screen for this command. The selected entity is added to the whiteboard. It may already be a database entity (e.g. another whiteboard entity), or it may be a Tioga document or Cedar program (in which case a database entity may be created to represent it).

Note: Being able to put tools on a whiteboard means that you can add things like SIL files or Press files to a whiteboard display, since the whiteboards code remembers not only the name of the tool but also any arguments used to create the particular tool viewer that was added to a whiteboard. For instance, try displaying a SIL file using Cedar SIL and then adding the SIL viewer to a whiteboard. This places on the whiteboard an icon that will load SIL (if it is not already loaded) and will create a SIL viewer displaying the same SIL file. Just remember, if you do this on a public whiteboard to make sure that the file being displayed is in a public place! (The same trick works using PreView to display Press files.)

AddCommandFile: You can add command files to a whiteboard by selecting a file viewer and applying this menu button. The command file will be added using a Typescript icon (suggesting its use); when opening the icon a CommandTool will be started to perform the operations specified in the file.

Erase: Erases the entire whiteboard, but no need to panic ... this edit isn't commited to the database until you actually Save the whiteboard.

HELP: Adds a text box to the whiteboard which contains the instructions for editing the whiteboard using the mouse.

Grid: You may want the icons and text boxes on the whiteboard to line up either vertically or horizontally. The Grid menu item lets you set the size of a grid (in pixels) on which to place viewers on a whiteboard. The default grid size is 1; it can be doubled (up to a maximum of 32) by left-clicking the menu item and halved by right-clicking the menu item. Each time the grid is changed, the whiteboard will be redisplayed. Moreover, new items added to the whiteboard will only be allowed to settle on points on the grid (for icons, this means that the top left corner will be on a grid point, for text boxes that all of the corners will be on grid points).

Save: Stores any edits to the whiteboard in the database; removes "[New Version]" at top of window. An attempt to store changes for a whiteboard may fail if the server has crashed, the database is readonly (you don't have permission to change it), the transaction aborted (this is quite unlikely), or someone else has updated the whiteboard since you began editing it. If this later case arises, you will be notified with the message *Can't Save -- whiteboard has wrong version* in the message window; the contents of the whiteboard will not be altered, so that you can save the contents of any text boxes that you have edited; you must do a Reset before editing, however, to retrieve the new contents of the whiteboard.

In addition to the commands at the top of a Whiteboard window, mouse button presses within the window area itself have specific effects on the entity the mouse is pointing at. Because mouse selections have their Tioga meanings when the cursor is *inside* text boxes, you must select just *outside* to get these effects on text boxes. The mouse buttons and their effects are:

Left: Selects and moves the icon for the entity (or a text box) around on the whiteboard.

Shift Left: Adds a new text box to the whiteboard at the location of the cursor. The text box can then be grown, moved, or typed into.

Control Shift Left: If the current selection is in a text box on a whiteboard, copy the text box at the location of the cursor. Both the size and contents of the new box are taken from the old one.

Control Left: Deletes the entity from this whiteboard.

Middle : Opens a displayer on the entity (or activates a tool).

Shift Middle : Opens a full-sized displayer on the entity

Control Middle : Expands the selected whiteboard. All whiteboards contained in the selected whiteboard are displayed, with lines drawn to illustrate the network structure. The added whiteboards can be opened, moved, deleted, etc... . but are not saved as permanent edits.

Right: Grows (or shrinks) a text box.

Finally, there are a few command-tool commands available for manipulating whiteboards. They are:

WBOpen <filename>: Opens the whiteboard database specified by the file name (this must be a full path name, including server). If no file name is specified, then the name of the previously opened whiteboard segment is used. If you already have a whiteboard database open, the old one will be closed and all whiteboard viewers on the screen will be destroyed.

WBClose: Closes the open whiteboard database. (Note: this does not destroy any whiteboard viewers currently on the screen; however, you won't be able to perform any operations on them.)

Whiteboard <name> or *WBDisplay <name>*: Opens the named whiteboard. If Whiteboards is not already running, the default Whiteboard.load file will first perform a WBOpen of the database named in the file "Whiteboard.DefaultDatabase."

WBCreate <name>: Creates a new whiteboard with the given name in the currently open database.

WBDestroy <name>: Destroys the named whiteboard, erasing it entirely from the database (be careful!).

WBCopy <from> <to>: Copies the whiteboard <from> to the whiteboard <to>. If <to> already exists in the database it is erased before the copy takes place; otherwise a new whiteboard is created.

WBDump <name>: Dumps the current content of the database to the named file. If no file is given, "Whiteboard.dump" is used. The dump file format is described below.

WBLoad <name>: Loads the database from the contents of the specified dump file. Again, if no file is given, "Whiteboard.dump" is used.

WBList <pattern>: Lists all of the whiteboards in the database whose names match the given pattern; the listing also gives the time of the last recorded update to each whiteboard.

WBCedar: Opens the Cedar whiteboards (see below).

WBCSL-Notebook: Opens the CSL-Notebook whiteboards. These whiteboards organize CSL documents that are not directly a part of Cedar. For more information, see [Indigo]<CSL-Notebook>Documentation>CSL-NotebookDoc.tioga.

If you forget what the commands are, use the command "help WB*".

Whiteboards in a Checkpoint

It is alright to put whiteboards in your checkpoint. The checkpoint procedure for whiteboards performs the equivalent of the WBClose command, which will close the connection to the database and leave any currently displayed whiteboards on the screen. When coming back from a rollback, you will need to perform a "WBOpen" to resume use of the whiteboard database you were last connected to. (Note: this restart may take some time as connections may be opened to an icon database as well -- it has finished after all of the whiteboards on the screen have been redisplayed.)

Whiteboard Dump Files

Dump files provide a convenient way to copy entire collections of whiteboards from one database to another or to make backups. The dump file format is intended to be relatively easy to hand edit: its syntax is given by the following grammar:

DumpFile ::= Whiteboard*

A Dump file is a sequence of whiteboard descriptions

Whiteboard ::= WHITEBOARD Name Item* END

Each whiteboard decriptions begins with the keyword WHITEBOARD, gives the name of the whiteboard and the list of items on it and ends with the keyword END

Item ::= NoteItem | IconItem

NoteItem ::= NOTE x y w h content formatting

The x, y, w, and h integers give the position and size of the box; the content and formatting Rope literals give the content and formatting for the note. (IO.GetInt is used to read x, y, w, and h; IO.GetRopeLiteral is used for content and formatting.)

IconItem ::= ICON x y name label type iconName argument

X and y are integers that give the position of the icon; name, label, type, iconName, argument are Rope literals that give the information necessary to display and open the icon. (Type will be one of "Text", "Tool", "ToolRope", "Whiteboard", or "Entity")

Browsing the Cedar Whiteboards

You can begin to browse the Cedar Whiteboards by using the command:

WBCedar

[Note: if you are on a public machine (or a non-rogue private machine) and you have not previously opened any other whiteboards, the command *Whiteboard Cedar* will also open the Cedar whiteboards.]

The Cedar whiteboard is the "root" of the documentation; it contains some basic information about CSL and Cedar and also includes icons for a collection of additional whiteboards that give more specific information on various aspects of the system. Here is a brief summary of the contents of the whiteboards referenced from the root (pictures of these whiteboards can be found at the end of this document):

HouseOfCards: A picture of the "James Morris Memorial House of Cards", complements of Polle Zellweger.

Basics: Information about user profiles, the set of basic DF files in Cedar, the CommandTool and (perhaps most importantly) how to boot a Dorado.

Language: An overview of the Cedar language, with pointers to documentation giving the complete Cedar language syntax, some suggestions for Cedar program style, and a collection of example programs.

Components: Information about some of the more important components of Cedar: Tioga, Viewers, the CommandTool (from the programmer's viewpoint) and Cypress. This also contains a reference to the Cedar Catalog.

Tools: The most widely used tools are described here, including Walnut, the DFTool, the EditTool, Chat, TSetter and Remember.

Interfaces: The most widely used interfaces in Cedar include Rope, Viewers, and IO (both the IO and FS interfaces).

ToolBox: A large (and growing) collection of Cedar tools can be started here just by opening the appropriate icon (but it wouldn't hurt to read the documentation for the tools first!).

Games: When you tire of doing serious work, there is a small collection of Cedar games to play: Mazewar, Football and Tank.

Many of these whiteboards contain references to other whiteboards, either to go more deeply into the structure (for instance, on the ToolBox whiteboard is a PerformanceTool whiteboard, which contains a collection of tools related to measuring performance) or to permit browsing in a less hierarchical fashion (the Cedar whiteboard can be found on many of the other whiteboards in the database, making it pretty easy to find your way back to the root).

The structure of this database and its contents should change over time -- this is just the beginning of an experiment! If you have comments on the organization of the database or on missing (or just wrong) contents, please send them along to Donahue.pa. If you're interested in either massaging the database or writing new applications code to improve the utility of the system, I would be especially interested in talking to you.

Browsing the CSL-Notebook Whiteboards

Another important collection of whiteboards is the CSL-Notebook whiteboards, which can be opened by issuing the command

WBCSL-Notebook

These whiteboards organize CSL documents that are not directly a part of Cedar. For more information, see [Indigo]<CSL-Notebook>Documentation>CSL-NotebookDoc.tioga.

User Profile Options

There are no user profile options for whiteboards. The default database used by Whiteboard.load is taken from a file named "Whiteboard.DefaultDatabase" -- you can change this file to change the default whiteboard database to be opened.