# 11. SYSTEM CARDS

Cards can be broadly divided into two categories. System cards are cards where the contents are built by the system for you. User cards are those cards for which you create the contents. This chapter discusses system cards.

This chapter explains how to use:

- Search cards
- LinkIndex cards
- Document cards
- Browser cards

## Search Cards

Search cards search through all boxes and cards looking for titles containing a specified string of characters. A new card is created containing links to these cards. To create a search card, point to New Cards entry in the banner with the middle button. Then slide the mouse to the Search entry, as shown in Figure 11-1.

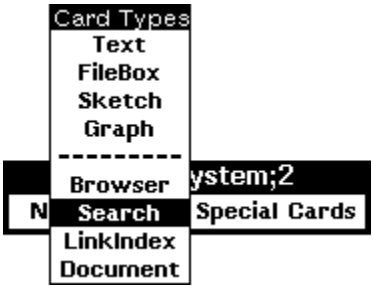


Figure 11-1. **Search Card** Menu Option

The search card also contains the day and time the list was compiled. Search is case sensitive so the string must be typed exactly as it appears in the title. For example, the Search card below was created by answering the prompt for a search string with **Creat**.

Search strings allow wild card characters.

* Matches any number of characters.

? Matches one character.

If you enter a string without wild cards, a "*" will be inserted at each end. If you enter any wild cards, you must enter a "*" at each end of the search string if you want an internal match. For example, a search of "xxawczz" with "a?c" fails, but "*a?c*" succeeds. Note that "awc" would also succeed.

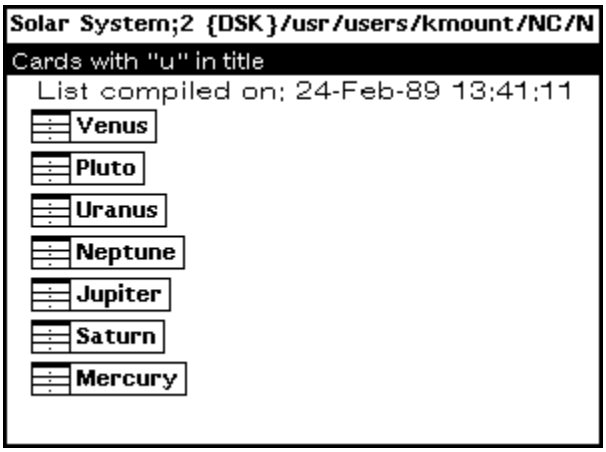


Figure 11-2. Search Card Example

Note: This list is not updated as new cards are entered into the NoteFile. The date and time the list was compiled can be used to help determine if the list in the card is out-of-date.

## LinkIndex Cards

LinkIndex cards build sorted lists of all of the cards in the Notefile connected by the specified link type(s). A LinkIndex card helps clarify the link-based relationship taxonomy which has been created within a Notefile. As for search cards, link index cards are created using the middle button menu of the New Cards entry in the notefile banner, as shown in Figure 11-3.

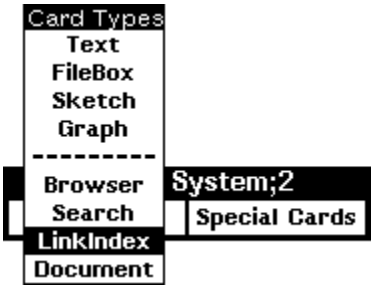


Figure 11-3. LinkIndex Card Menu Option

The following is an example of a Link Index card.

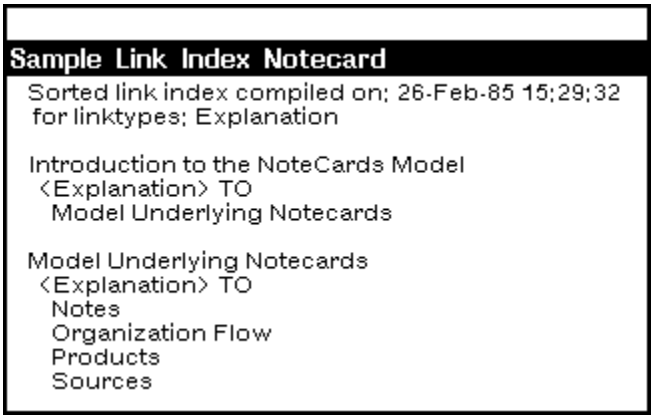


Figure 11-4. LinkIndex Card Example

After creating a link index card, a list of Link Types is displayed from which to select the link type with which the index will be built. Both links to cards/boxes and links from cards/boxes may be selected for tracking. In the example above, Explanation was selected. The prompt window also asks whether back links should be built to the cards or boxes connected by the specified links. Back links are useful if the link index is being used to collect cards for later modification of link types. The example shown above was compiled with the back links option off. Then **Done** was moused.

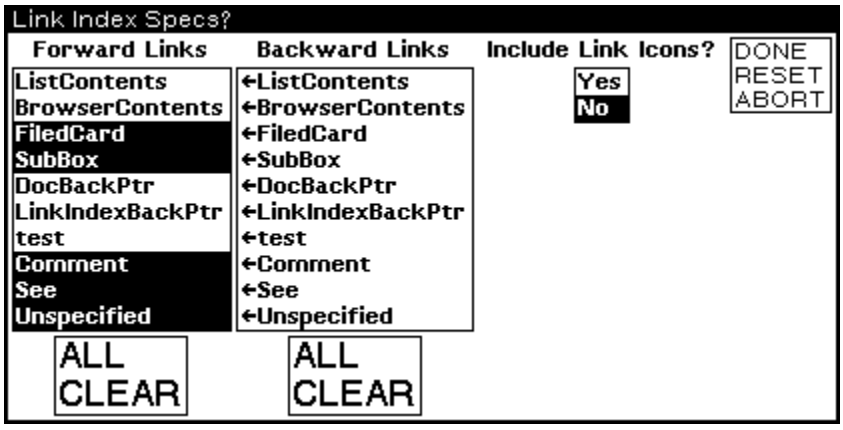


Figure 11-5. **Link Index Specs?** Menu

After the prompt is answered, link index compiles the set of cards/boxes linked by the chosen link types and displays the list in the Link Index card, sorted alphabetically. This new card also contains the day and time the list was compiled.

Note: This list is not updated as new cards are entered into the NoteFile. The date and time the list was compiled can be used to help determine if the list in the card is out-of-date.

## Document Cards

creates a card that contains textual information collected from all descendant cards and/or boxes from a specified FileBox or NoteCard. Once Document is selected (again using the middle buttton of the **New Cards** entry in the Notefile banner), the user is prompted for a starting box or card. After this selection, a sub-menu of Make Document parameters is displayed.

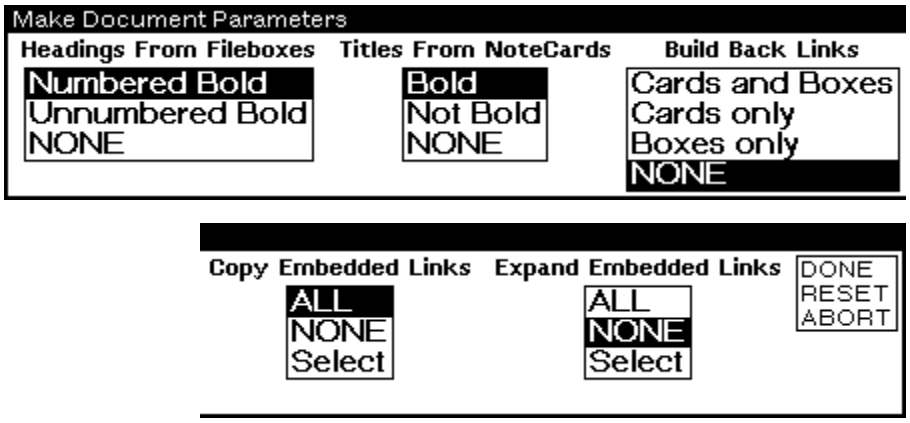


Figure 11-6. **Make Document Parameters** Menu

These parameters are set to default values displayed to the right of each parameter. It is possible to change a value by placing the cursor over the parameter and depressing the left mouse button. This pops up a menu of values. Select a value from this menu. The parameters and their choice of values are described below.

## HeadingsFromFileBoxes

Creates section headings and subheadings for the document according to the option selected.

**NumberedBold** includes numbered headings, in bold, of FileBox descendants of the originating FileBox. Each heading is numbered in standard section heading form. For example, "1.3" would be assigned to the third sub-filebox of the first sub-filebox of the originating FileBox.

**UnnumberedBold** includes unnumbered headings, in bold, of FileBox descendants of the originating FileBox.

**NONE** excludes headings of FileBoxes including the originating FileBox.

## TitlesFromNoteCards

Enables the document card to be generated with paragraph-level labeling according to the option selected.

**Bold** includes titles, in bold, of all card descendants of the originating card.

**NotBold** includes titles, in regular type, of all card descendants of the originating card.

**NONE** excludes all titles of cards including the originating card (if starting from a NoteCard).

## BuildBacklinks

Builds back links according to the option selected. Back links to cards allow the user to easily retrieve a card for text revision; back links to boxes facilitate structural revision of a document. In the sample **MakeDocument Parameters** menu shown above, the BuildBacklinks parameter is set to ToCards to facilitate the text editing process. When a final version of the document card has been generated, the user may want to set this parameter to NONE to omit the back links from a hardcopy.

**Cards and Boxes** builds links to each NoteCard and FileBox descendant of the originating card or box. The link icon, displayed as a small notecard, is inserted next to the title of the card/box the link refers to.

**Cards only** includes link icons only to NoteCard descendants of the originating card or box. The link icon, displayed as a small notecard, is inserted next to the title of the card the link refers to.

**Boxes only** includes link icons only to FileBox descendants of the originating or box. The link icon, displayed as a small notecard, is inserted next to the title of the box the link refers to.

**NONE** back links are not built for the Document card.

## CopyEmbeddedLinks

Allows the user to retain or omit embedded links in descendant cards.

**ALL** includes any links appearing in descendant NoteCards.

**NONE** omits any links appearing in descendant NoteCards.

**Select** enables the user to select link types to be included in the Document card. When select is designated, a list of Link Types pops up from which the user selects the link types to include.

## ExpandEmbeddedLinks

Allows the user to replace embedded links in descendant cards with their text. Note that cards that appear multiple places in the structure defined by the descendant cards are only expanded once, thus avoiding unneccessary duplication of text. In the sample MakeDocument Parameters menu shown above, the Select option was chosen, then the Capital link type was selected off of the menu listing link types. Therefore, any Capital links encountered while building the document will cause the text of the cards at the other end of the link to be embedded in the document at the link's location.

**ALL** for any link appearing in the text of a descendant card, the link is replaced by the text of the card linked to.

**NONE** leaves embedded links alone.

**Select** expands embedded links selectively, depending on the link type.

## Browser Cards

A card that presents a view of cards/boxes and the links between cards/boxes in a specified portion of the current NoteFile. Given a starting card or box and a set of link types, Browser will create a graph structure showing all NoteCards and FileBoxes linked to and/or from the initial card/box by recursively following all links of the types specified.

This graph structure displays both the specified set of cards/boxes, and the links between the cards/boxes in the set. The following is an example of a Browser card:

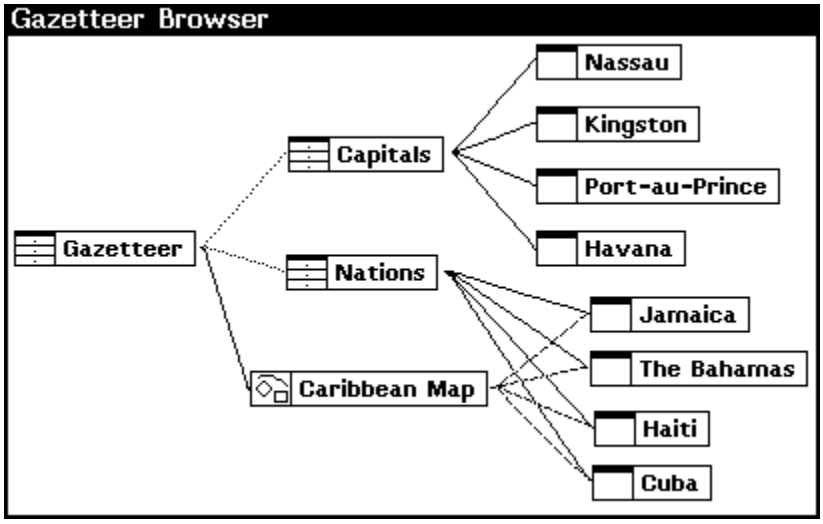


Figure 11-7. Browser Card

Before selecting Browser, the starting card/box or a link icon referring to the card/box must be visible on the screen to allow selection with the mouse. The Browser selection is started also from the **New Cards** entry of the Notefile banner with the middle button. After creating a blank browser card, a prompt window will be displayed above the card asking the user to designate a starting card/box. The user may select the initial card/box or abort the browser by selecting **Cancel**. In the above example, the box chosen was Gazetteer.

After designating the starting card/box, a list of Link Types is displayed from which to select the types of links the browser should follow. Both links to cards/boxes and links from cards/boxes may be selected for tracking. You may select the types and **Done** or abort the browse by selecting **Cancel**. In the above example, the link types chosen were FiledCard, SubBox, and Comment. Once the link types have been determined, the browser compiles the set of cards/boxes and displays it in the Browser card.

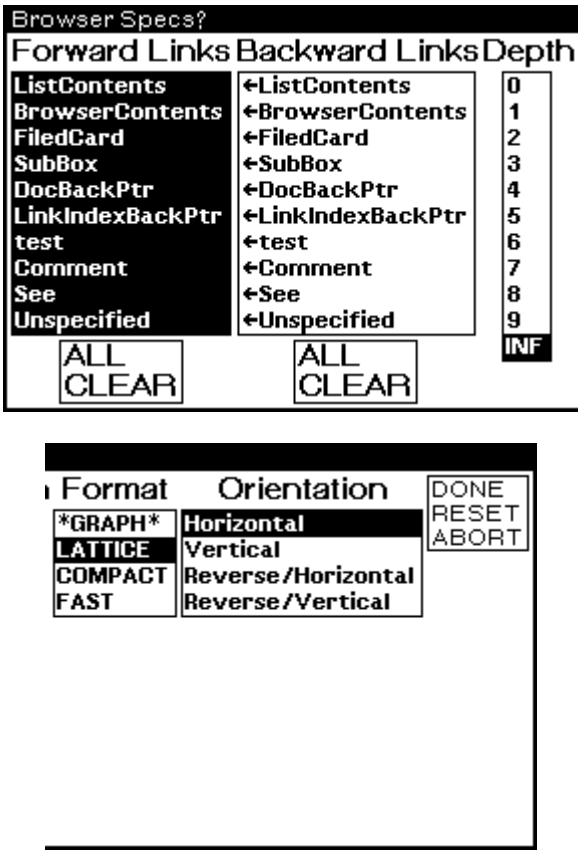


Figure 11-8. **Browser Specs?** Menu

The links can be drawn in one of two ways, depending on a parameter value set in the **Change Parameters** command from the **Main Menu**. The first method shows the links as solid lines; the second method uses dashed lines. With the second method, each link included in the Browser is assigned a different style of dashing. There are six styles of dashing; therefore, dashing for links after the sixth type is of the same style. The Browser card shown above was created with the LinkDashingInBrowsers parameter set to `Yes`.

A legend is attached to the upper right-hand border of every Browser. This legend displays each type of link present, and, if the links are dashed, the legend displays the particular pattern of dashing. For the Browser card shown above, the legend appears as follows:

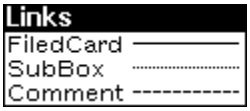


Figure 11-9. Browser Card Legend

The characteristics of browser cards and graph cards are identical except for the single-item menu that is displayed by pointing to the title bar with the cursor and pressing the middle mouse button. (See the Graph Editing Menu section.) The commands in this menu follows:

## Recompute Browser

**Recompute Browser** modifies the browser card to reflect the current NoteFile starting from the same card or box and following the same link types as originally specified.

**Warning:** Nodes and links added or deleted through the **Graph Editing Menu** commands **Add Node**, **Add Link**, or **Delete Link** are removed when **Recompute Browser** is executed.

## Multiple Roots

Browsers can contain multiple roots, in which case the graph will be laid out as a forest.

## Dashed Links

Dashed browser links should be used carefully because of speed considerations. There are currently nine different dashing styles possible. If a browser contains instances of more than nine different link types, then the last dashing style will be used repeatedly for each link type beyond the ninth. Link dashing is a user-settable option in the GlobalParameters menu (see Section #). In Figure # the links legend is attached to the upper right corner of the browser.

## Arrowheads

Arrowheads can be drawn on browser links. These show the direction of the notecards link being represented in the browser. This is a user-settable parameter in the GlobalParameters menu with possible values AtMidpoint, AtEndpoint, or None. If AtMidpoint or AtEndpoint is specified, then arrowheads will be drawn at link midpoints or endpoints, respectively. However, in either case, if two browser nodes are connected by more than one link, then any arrowheads for those links will appear at the midpoints (so as not to overlap).

## Browser Specs

Link types is one of a number of browser specs, as already mentioned. Also included are browser depth, format, and orientation. These are accessible through a BrowserSpecs stylesheet, a collection of five menus. The browser specs stylesheet is shown in Figure #.For general details on the stylesheet interface see Section #. In this case, the forward and backward link types menus are multi-selectable, that is, more than one entry can be chosen. The other three menus are used to make single selections.

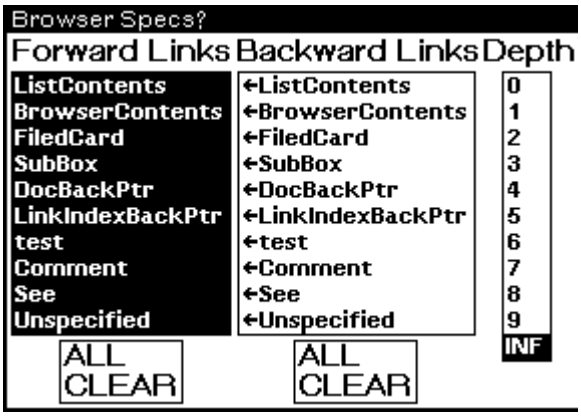


Figure 11-10. Link Types Selection Section of **Browser Specs?** Menu

Forward and backward link types function as explained in the chapter on Links. That is, the browser will contain only nodes for cards reachable from the root cards by following forward links in "line of direction" or backward links in "reverse line of direction."Browser depth is chosen from a menu containing entries for the integers 0 through 9 and INF (or infinite depth). The default is INF, meaning that the browser will not be cut off until there are no more links to follow from leaf nodes. Choosing depth 0 means that only the root nodes will appear (and no links).

Browser format is one of *GRAPH*, LATTICE, COMPACT, or FAST. The latter three are provided by the grapher package and correspond to lattice, compact forest and fast forest, respectively. COMPACT and FAST generate virtual nodes (in double boxes) whenever two or more links would be drawn to the same node. LATTICE only generates virtual nodes when a cycle exists in the graph. *GRAPH* is a format that never generates virtual nodes. The drawback to using *GRAPH* is that a cycle can cause lines to be drawn that cross boxes or overlap other lines. Thus you may have to move nodes around for legibility after computing the browser. The default is LATTICE.

Browser orientation is one of Horizontal, Vertical, Reverse/Horizontal, or Reverse/Vertical. These specify whether the graph is layed out left-to-right, top-to-bottom, right-to-left, or bottom-to-top, respectively. The default is Horizontal.

## Middle Button Title Bar Menu Options

The options to the middle button menu invoked from a browser's title bar are RecomputeBrowser, RelayoutGraph, ReconnectNodes, UnconnectNodes, ExpandBrowserNode, GraphEditMenu, ChangeBrowserSpecs , ChangeBrowserRoots, BrowserOverviewWin and ChangeOverviewSpecs.

**RecomputeBrowser** causes the current contents of the browser to be thrown away and recomputed. You can optionally specify a new set of root nodes.

**RelayoutGraph** does not rebuild the graph, but rather causes the nodes and links of the graph to be repositioned on the screen. This will destroy any work you have done moving nodes within the graph.

**ReconnectNodes** first causes any link edges in the graph to be erased. (Note, however, that non-link edges, those created by "AddEdge" as described below, are ignored.) Then, each node in the graph is connected to every other node in the graph for which there is a link between them having one of the currently selected link types. This can be useful for several reasons:

1. When the linking structure between cards has changed, but the current browser layout needs to be preserved.

2. When some browser nodes need to be moved, but dragging the connected links is too slow. In this case, do UnconnectNodes followed by ReconnectNodes (after you've moved the nodes around).

3. When special browser layouts are desired. For example, suppose you like the layout that Grapher gives you when certain links are left out or when you limit the depth. Then calling ReconnectNodes will fill in the missing links without affecting the graph's layout.

**UnconnectNodes** erases all edges in the browser. This is useful for positioning a browser's nodes before invoking ReconnectNodes.

**ExpandBrowserNode** allows you to enlarge the graph under a given node. After selecting a node, you're asked for a depth (defaults to 1). The graph is then expanded under the selected node to the given depth, following any currently selected links. Note that ExpandBrowserNode calls LAYOUTGRAPH so any existing special node arrangements will be lost.

**GraphEditMenu** brings up the graph editing menu. See the description below.

**ChangeBrowserSpecs** brings up the BrowserSpecs stylesheet to allow you to change any of the browser specs. These changes will be noticed at the next RecomputeBrowser, ExpandBrowserNode, etc.

**ChangeBrowserRoots** allows to change the roots for the browser card. These changes are noticeable at the next RecomputeBrowser, ExpandBrowserNode, etc.

**BrowserOverviewWin** causes an overview window to be created and attached to the upper left corner of the Browser. If this Browser has never had an overview window, then the overview window has the default height and width. If there previously was an overview window that has since been closed, then its height and width are reused.

**ChangeOverviewSpecs** brings up the Browser overview stylesheet. The stylesheet contains three submenus. The first two specify where to attach the overview window and the third specifies the mode of operation.

## Editing the Browser Manually and "Structure Editing"

Browsers are first useful as static views of a portion of a notefile. But, they can also serve as "reach-through" ports, allowing changes to be made to the notefile. This capability is handy when you want to create structure, but not content. For example, when brainstorming ideas for a paper, you might have card titles and links in mind, but want to put off the job of filling in each card's text till later.

A browser consists of **nodes** and connections (or lines). These usually correspond to cards and links in the notefile. However, it is also possible to annotate a browser with **labels** which are pieces of text representing nothing in the notefile (at least not as far as NoteCards is concerned). Edges can also be created joining labels and/or normal nodes which do not correspond to links in the notefile. These are called **non-link edges**.

Creation and deletion of nodes and edges in the browser is accomplished using the GraphEditMenu. (In Figure #, this menu is attached to the lower right corner of the browser.) This menu can be obtained either by right-buttoning in the browser window or by choosing GraphEditMenu from the title bar middle button menu.

The options in the GraphEditMenu can be used to create either nodes representing existing cards in the notefile, new nodes and corresponding cards, or browser labels with no corresponding card. There are also operations for creating browser edges and

accompanying links as well as edges without notefile analogues. Edges and nodes can be deleted in two ways: either such that the corresponding link or card is deleted as well as the node or edge, or just such that the node or edge is removed from the browser. These options are described further below.

---

**Create Card** causes a new card to be created in the current Notefile and a corresponding node for it to be included in the browser. You're asked for the type of the new card, its title, and where to position the node representing it.

**Insert Card** adds an existing card to the browser. You are asked to point to a card (title bar or link icon) on the screen that this node is to represent and then to position the node.

**Delete Card** causes a card to be deleted and its corresponding node in the browser to be removed. You are asked first to choose the node representing the card to be deleted and then to confirm the removal of the node (type "y" to confirm) and the deletion of the card. If the selected node is one of a set of virtual nodes (double boxed), then all nodes in the set (i.e. representing the given card) are removed.

---

**Create Link** causes a new link to be created between two existing cards and a corresponding edge to be drawn in the browser. (We call such an edge representing a Notecards link, a "link edge." See AddEdge below for creating non-link edges.) You're asked for the "From" and "To" nodes in the browser corresponding to the cards to be linked as well as a link type. The link icon for the new link is positioned at the cursor point in the From card if the card has text substance and an open window. Text cards with closed windows have links inserted at the start of the text stream. Otherwise, the new link is a global link. You can have multiple link edges between pairs of cards. In this case the edges are displayed in a spline or "flower" arrangement.

**Delete Link** causes a link in the Notefile to be deleted and the corresponding edge in the browser to be removed. You first pick the "From" and "To" nodes corresponding to the source and destination ends of the link respectively. Then, if there is only one link between those two cards, the link is deleted after user confirms. If there are multiple links between the two cards, then the user chooses from a menu of link types.

---

**Move Node** allows you to change the position of any node, rubber banding any edges pointing to it. You're asked to point to the node by left-buttoning, and holding down the left button, drag the node to its new position.

**Remove Node** removes a node from the browser. It does not delete the card (if any) that the node represents. Edges into and out of the node are also removed. If the selected node is one of a set of virtual nodes representing the same card, then you will be told how many nodes will be removed with this one and will be asked to confirm. The only way to remove only one node of a set of virtual nodes, is to first manually remove edges into and out of it using RemoveEdge. Then RemoveNode can be used to remove only the one virtual node.

**Connect Nodes** draws a line between two nodes in the browser. This edge does not correspond to a real link in the Notefile. To avoid confusion, it is best to have the arrowheads option on (see Section #) in this case, since edges formed by AddEdge do not have arrowheads (or dashing). Only one such edge is allowed between any two nodes

and none if there are already link edges between the nodes. Thus doing CreateLink&Edge will remove any existing non-link edge.

**Disconnect Nodes** removes an edge from the browser. It does not delete the link (if any) that the edge represents. The user is asked to select the "From" and "To" nodes of the edge.

---

**Add Label** puts a "label node" into the browser that does not represent a Notecard. You are prompted for a string forming the node's label and then must position the label node. This node is not boxed. (But note that "virtual" label nodes can be boxed and thus can be confused with non-virtual regular nodes.)

**Smaller Label** is used to decrease the font size of label nodes. Note that it does not work for regular non-label nodes.

**Larger Label** is used to increase the font size of label nodes.

**Toggle Shade** toggles the shade of a node between black-on-white and white-on-black. This can only be performed on label nodes (not on nodes representing Notecards).

---

**FIX MENU** causes the GraphEditMenu to be affixed to the lower right edge of the browser window. Note that this does not prevent you from obtaining the menu via right button inside the window.

## Overview Windows on Browser Cards

A Browser overview is a small window attached to a Browser that contains a shrunken view of the entire Browser graph (not just what currently appears in the Browser's window). Within this overview is a wire frame representing that portion of the Browser graph that is currently displayed in the Browser window. The wire frame can be moved in the overview window to effect large scale scrolling in the Browser window. Likewise, scrolling the Browser window moves the wire frame in the overview window.

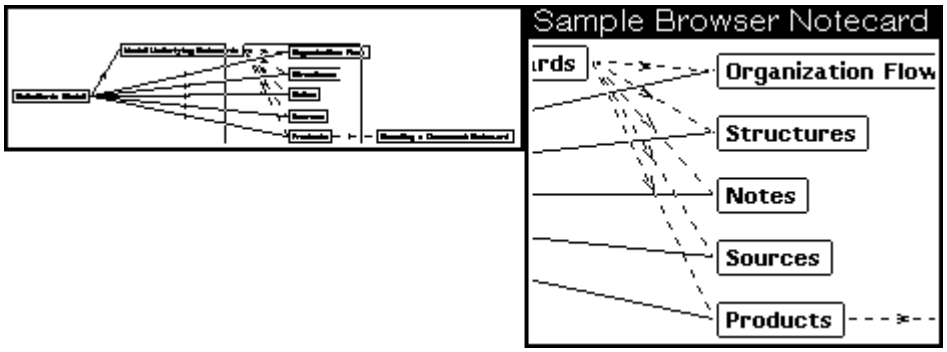


Figure 11-11. Browser Card with Overview Window

The details of creating and manipulating overview windows follow.

### Creating an Overview Window

There is a menu item **Browser Overview Win** on the Browser middle button menu. Clicking on this causes an overview window to be created and attached to the upper left

corner of the Browser. If this Browser has never had an overview window, then the overview window will have the default height and width. If there previously was an overview window that has since been closed, then its height and width will be reused.

### Reshaping the Overview Window

If you reshape the overview window, then its contents are expanded or shrunk to "fill the container," and then the overview window is reattached to the Browser.

### Scrolling and the Wire Frame

In the overview window there is a wire frame whose location and size in the overview match the scrolled position of the Browser window relative to the entire graph. In effect, the wire frame provides a 2-dimensional scroll bar. Its position and size are dynamically updated as the Browser window is scrolled and reshaped.

Furthermore, the wire frame can be moved in the overview window to effect large scale scrolling of the Browser window. This is done by clicking the mouse inside the overview window. The cursor is "pulled" to the nearest corner of the wire frame, and as long as the button is held down, you can drag the wire frame to a new position. If the button is let up with the wire frame outside the overview window, then the scrolling operation is cancelled.

### Recomputing the Overview Contents

For large graphs, the operation of creating the overview contents can be expensive. Thus if the Browser's contents change, the overview window doesn't update automatically. In order to recompute the overview window based on current Browser contents, hit Redisplay from the right button menu of the overview window.

### Browser Overview Stylesheet

Certain parameters governing the operation of Browser overviews can be changed via the Browser overview stylesheet. To bring up the Browser overview stylesheet, click on **Change Overview Specs**, from the Browser title bar middle button menu. The stylesheet contains three submenus. The first two specify where to attach the overview window and the third specifies the mode of operation.

The position at which the Browser overview is attached can be selected by specifying values from two menus: the edge along which to attach (top, bottom, right, left), and the position along that edge (top/right, center, left/bottom). [Currently center position only works for top and bottom edges.]

The third submenu lets you select the mode. This can have one of three values:

**Compress Overview Win**: This forces the overview window to be just big enough to fit the overview contents plus the wire frame. This means that when you reshape, the window will be compressed along one dimension to fit the overview.

**Expand Overview**: This doesn't change the window size, but rather increases the size of the overview graph to fit the window shape you choose. This is done by scaling up in the dimension needed to fill the window. (Scaling is only done to node position, NOT node size. So what you're really getting is stretching out of your links.)

**Neither**: The old style. That is, the window is whatever shape you make it and the overview contents are in the same aspect ratio as Browser contents. This is the default.

## Tidbits

When the ratio between overview window size and graph size gets smaller than a threshhold value (currently .3), nodes in the overview switch from being shrunken but readable to being unreadable shaded boxes.

Use the wire frame in the Browser overview to scroll the Browser beyond its displayed region. In fact, you can even pull the wire frame partially outside the overview window and the Browser will scroll appropriately. (Note that if you do such a thing, you may want to put a label in that new area or move a node over there or something to make your next Redisplay of the overview window show the increased area.) As mentioned above, if you pull the wire frame totally outside the overview window, you've done a no-op.

[This page intentionally left blank]