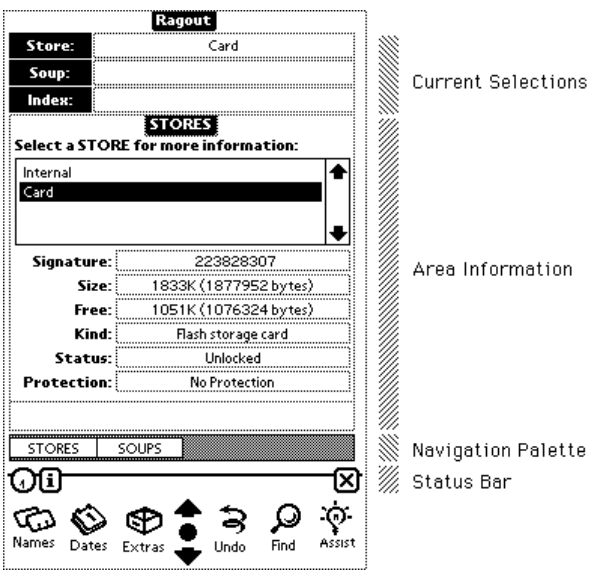
INTRODUCTION

Ragout is a soup browser and editor that lets Newton developers, power users, and consultants inspect and modify soups on the fly. Note that Ragout has several powerful features that can change the contents of a soup. You need to use these with caution. If you are not familiar with soups, we strongly suggest you read "A Brief Introduction to Soups" in Appendix A of this manual. For more detailed information, you should also read Chapter 7 of the *Newton Programmer's Guide*.

Developers find themselves performing different types of roles during the software development cycle. Early in a project you may play the role of a consultant that recommends and advises a client on what is possible in a particular time frame. These early project discussions often determine the scope and feasibility of a project. This is the time when it's appropriate to set the right expectations. Ragout is a tool that can help you obtain real facts concerning critical soup issues such as size and speed.

During the development of a project, Ragout's browser capabilities, which let you view nested slot values in very complex soup structures, can save you a lot of time and also help debug your soups. You can also change values and types, and make other soup modifications on the fly. For new adopters of Newton Technology, Ragout is also an excellent tool for understanding how the storage system works.



THE ANATOMY OF RAGOUT

Ragout maximizes the Newton's screen to display the most information possible about a Newton's storage components. The Ragout screen has four main areas: Current Selections, Information Area, Navigation Palette, and Status Bar.

The Newton storage system contains Stores, Soups, Indices, and Soup Entries. A store contains a list of soups, a soup contains indices and entries, and soup entries contain slots which may be accessed using indices and cursors. Ragout lets you navigate and manipulate the information at any of these storage levels using an interface that naturally reflects the organization of Newton storage.

In order to access an entry in a soup you must do the following:

- Select a store,
- Select a soup,
- Select an index, and then
- Scroll to the desired entry.

Once you reach a particular entry, you can browse that entry slot by slot. You can also create new indices, remove indices, and search for a particular set of entries.

VIEWING STORES WITH RAGOUT

When you first start Ragout, the program lists the current stores (the Current Selections display area is blank). In addition, a STORES button appears in the Navigation Palette. Current Newtons support only two stores, so you will likely see two items listed – Internal and Card. If you don't have a PC (PCMCIA) memory card inserted in the Newton, you will only see Internal.

If you select a store from the list, the following things happen:

- The store name appears in the Store area of the Current Selections.
- Other information about the selected store appears in the Information Area: Signature, Size, Free Space, Kind of Storage, Status (locked/unlocked), and the type of protection used.

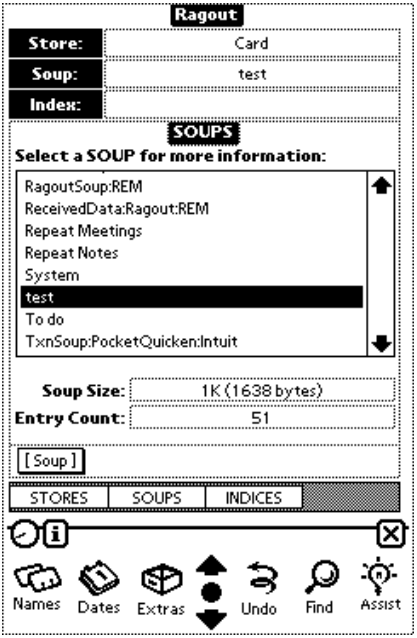
A SOUPS button appears to the right of the STORES button on the Navigation Palette, letting you view the soups in that store.

VIEWING SOUPS WITH RAGOUT

If you tap on the SOUPS button, the store information disappears from the Information Area, and is replaced by a scrolling list of all the soups in the current store. In addition, a [Soup] button appears at the bottom of the Information Area. This pop-up button lets you do such things as create, remove, copy, and move soups.

If you select a soup name from the scrolling list, three things happen:

- The soup name appears in Soup area of the Current Selections.
- More detailed information about the soup, such as it's size and the number of entries in it, appears in the Information Area.
- An INDICES button appears to the right of the SOUPS button in the Navigation Palette.



You can now use Ragout's soup features.

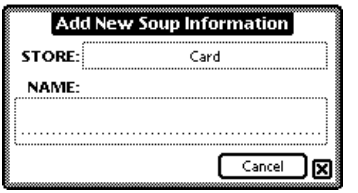
SOUP OPERATIONS

Ragout allows its users to perform operations at the soup level using the [Soup] button. When you tap this button a list of commands appears. Pop-up items with an ellipsis have additional dialogs that prompt you for more information to complete the operation.

CREATING SOUPS

You can use Ragout to create test soups using the **Create** command. A dialog appears asking you to enter the name of the soup. The soup is created when you close the dialog.

The **Create** command does not create Union Soups, it creates a soup in the store where Ragout is installed. Initially, the new soup has only a built-in index based on the `_uniqueID` slot. It is up to you to add more indices.



REMOVING SOUPS

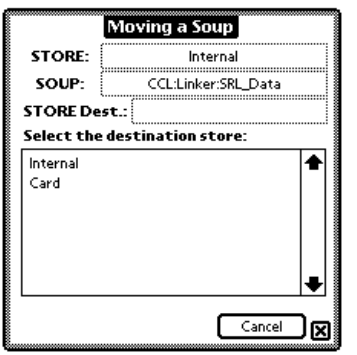
The **Remove** command removes the currently selected soup. You must confirm the deletion before the soup is actually deleted.

COPYING SOUPS

The **Copy** command displays a dialog that lets you select a destination store for the copy. You must select the destination store and possibly change the soup's name if you want to copy it to a store that already contains that soup.

MOVING SOUPS

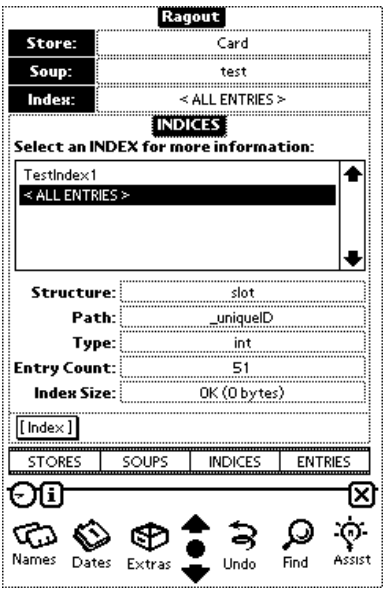
The **Move** command lets you move a soup from one store to another. When you select this command a dialog appears for selecting the destination store. If you don't specify a destination, the soup is not moved.



VIEWING INDICES WITH RAGOUT

If you tap on Navigation Palette's INDICES button, the following things happen:

- A list of all indices for the currently selected soup appears in the Information Area.
- The [Index] button, which provides a pop-up list of index operations, appears.
- The Index field in the Current Selections area of the screen displays < ALL ENTRIES >. This represents the Newton's built-in index which is based on the `_uniqueID` slot. This index is created by the Newton operating system; it's the default index to view all the entries in a soup.
- The ENTRIES button appears to the right of the INDICES button on the Navigation Palette.



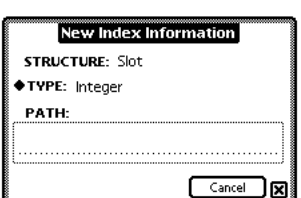
Once you select an index, the structure, path, type, entry count, and index size appear in the Information Area.

INDEX OPERATIONS

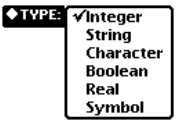
When you tap the [Index] button, a pop-up list of operations that can be performed on indices appears.

ADDING INDICES

When you select the Add command, a dialog appears so that you can enter additional information Ragout needs when adding an index to the currently selected soup. Here's a description of the fields:



- STRUCTURE always defaults to Slot in the current version of Ragout.
- The TYPE pop-up displays a list of possible index types.
- PATH is where you enter the name of the slot.



REMOVING INDICES

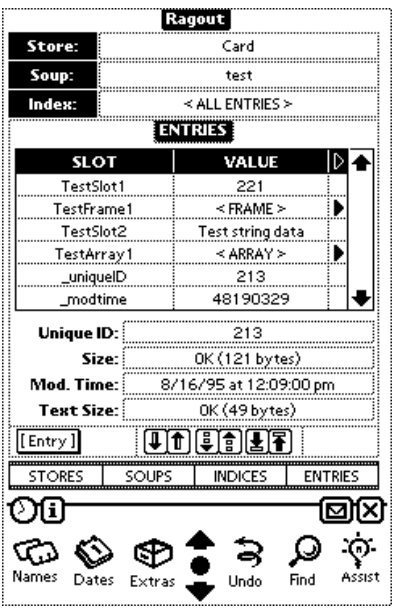
The Remove command lets you remove the currently selected index. Ragout prompts you with a dialog to confirm that you really want to remove the index.

VIEWING SOUP ENTRIES WITH RAGOUT

When you tap on the ENTRIES button on the Navigation Palette, Ragout:

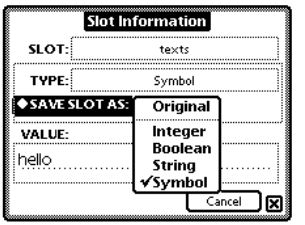
- Displays a three-column scrolling list of entries. The right-hand column contains navigation arrows for stepping into frames and arrays.
- Displays, at the bottom of the Information Area, details about the first indexed entry in the soup – its unique ID, size, last modification date, and text size.
- Displays the [ENTRY] button above the navigation palette. This popup contains commands for entry-specific operations.

In addition, both an arrow palette for browsing soup entries, and a routing button for printing, faxing, and beaming entries appear.



NAVIGATING AND EDITING SOUP ENTRIES

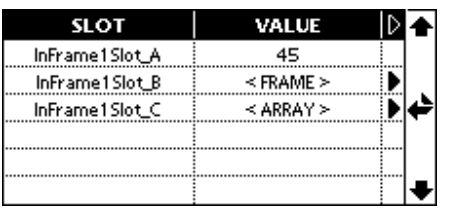
Navigating through a soup entry is simple and intuitive. If you tap on the slot or value cells in a row, an information dialog about that slot appears. Using this dialog you can change the value and type of a slot. The SAVE SLOT AS popup in the slot information dialog shows a list of data types you can assign to the current slot. The value assigned to a slot should be the proper type. If not, Ragout warns you that the type is incorrect.



You need to be careful – if you change the type of an indexed slot, you may get unexpected results when you try to use that index. Ragout has some safeguards built into it to minimize accidental damage. For instance, you cannot change `_uniqueID` *and* `_modtime` *slots that are maintained by the operating system.*

Ragout assumes certain defaults in some specific cases. For example, when you tap an array or frame slot to view its slot information, the value that you see is either <ARRAY> *or* <FRAME> *respectively. These settings can not be changed. If you want to change the slot type of an array or frame, you have to first delete the slot, and then re-add it as a new slot. Deleting a frame slot removes all the information contained in it.*

If the step-into column contains a small triangle, the slot points to a nested structure, either a frame or an array. If you tap on the triangle, Ragout displays the next level of the structure so that you can view and modify the slot values and types as needed. Also, a return arrow appears in the middle of the scroll bar. Tap this arrow to get back to the next higher level of the entry. Once you reach the top level in a frame or array this arrow disappears.



When you are looking at an entry, the Navigation Palette has three pairs of buttons that let you move between entries.

The first pair of buttons lets you move to the next or previous entry within the current soup. It's important to note that the order in which the entries are displayed is determined by the slot the current index is based on. For example, if the indexed slot is a string slot, the entries are displayed in alphabetical order.

The second pair of buttons lets you skip forwards or backwards n-entries at a time. This feature is useful when the soup contains a large number of entries. You can set the number of entries you want to skip at a time in the preferences view available by tapping the info button at the bottom of Ragout's main view. The original setting is to skip five entries at a time.

The third pair of buttons let you move to the first or last entry in the current soup.

ENTRY OPERATIONS

The [ENTRY] button displays a list of entry-related commands.

CREATING A NEW ENTRY

When you tap New Entry, Ragout creates a new entry with only the system slot `_uniqueID`. In order to see this new entry you must first select the <All ENTRIES> index. To view this new entry with other indices you must add the appropriate indexed slot to the entry and then select that index.

REMOVING THE CURRENT ENTRY

The Remove command removes the currently selected entry. Ragout asks you to confirm that you want to do this before actually deleting the entry.

Removing All Entries in a Soup

Remove All removes all the entries in the current soup. Ragout asks you to confirm that you want to do this before actually deleting the entries.

Removing an Index's Entries

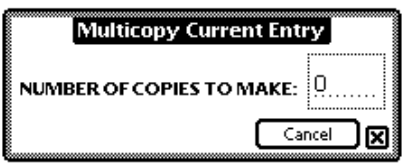
The **Remove All Indexed…** command removes all the entries included by the index that is currently selected. Ragout asks you to confirm that you want to do this before actually deleting the entries.

Canceling Changes

Revert to Original cancels the changes you have made to an entry before you leave that entry.

Copying an Entry

The **Copy** command makes a copy of the current entry in the current soup. It's the easiest way to quickly duplicate an entry.



Making Several Copies of an Entry

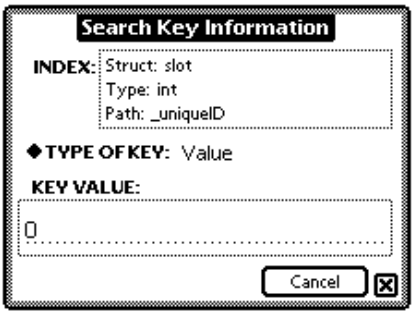
The **Multicopy** command displays a dialog so that you can enter the number of copies you want to make of the current entry. This is an excellent tool for determining the size of soups.

Setting a Tag/Go To Tag

Set Tag marks the current entry so that you can come back to it using the **Go To Tag** command. These two commands provide a quick way to get to a frequently accessed entry.

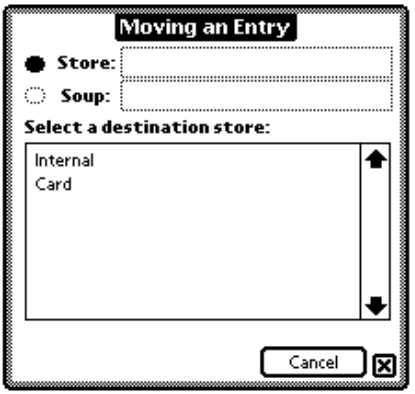
ENTRY SEARCH BY INDEX

Search lets you search for a particular entry based on the value of the current indexed slot. The only type of search key currently supported is **Value**. This may change in the future.



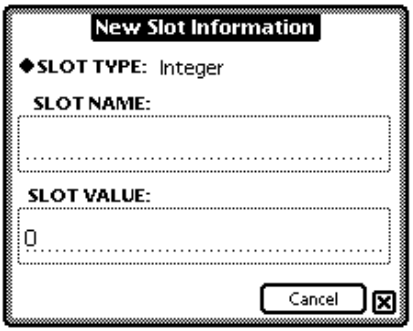
MOVING AN ENTRY

The **Move To** command lets you move the current entry into another soup. When you select this command, a dialog appears that lets you select the entry's destination store and soup.



ADDING A SLOT

Add Slot lets you to add slots to the current entry. A dialog appears that lets you designate the type of the slot, the slot name, and the slot value.



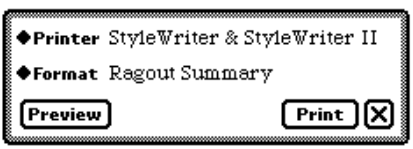
ROUTING

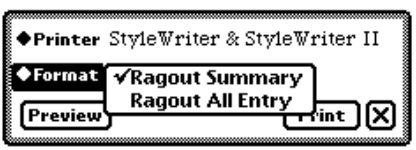
The Ragout routing button lets you print, fax, and beam information about individual entries and their soups. It only appears then you are viewing a soup entry.

When you select **Print Entry**, the standard Newton print dialog includes two print forms that you can select: Ragout Summary and Ragout All Entry.

Ragout Summary, shown on the next page, includes all the information collected about the store, soup, index, and entry and an indented text printout of the entry. Ragout All Entry prints only the text representation of the entry. Faxing utilizes the same formats used for printing.

Select **Beam** if you want to send the current soup entry through the IR port to another Newton. This feature only works between separate copies of Ragout installed on different Newtons. When receiving a beamed entry, the receiving copy of Ragout saves it in a special soup, named `ReceivedData:Ragout:REM`, which is created to receive data. Once the entry is saved you can access this soup and move the entry to another location.





Ragout Summary

STORE INFORMATION

Name:	Internal
Signature:	505023387
Store Size:	485K (497600 bytes)
Free Space:	177K (182144 bytes)
Kind:	Internal
Status:	Unlocked
Protection:	No Protection

SOUP INFORMATION

Name:	Example Soup
Soup Size:	0K (0 bytes)
Total Entry Count:	0

INDEX INFORMATION

Path:	_uniqueID
Structure:	slot
Type:	int
Indexed Entry Cou	1
Index Size:	0K (0 bytes)

ENTRY INFORMATION

Unique ID:	0
Entry Size:	0K (126 bytes)
Last Modification:	9/12/95 at 6:02:00 pm
Entry Text Size:	0K (48 bytes)

Entry:

```
{
  _uniqueID: 0,
  SampleInteger: 37,
  _modtime: 48229562,
  SampleString: "Some text",
  SampleSymbol: 'ASymbol,
  SampleArray: [
      "Text in array",
      251,
      {
          SampleAgain: "Frame inside an array",
          samples: 651,
      },
  ],
  SampleFrame: {
       Another: 2557,
       OneMoreTime: TRUE,
  },
},
```

Date: 9/12/95 6:19 pm

Getting Ahold of Us

Ragout technical support is provided by its author, Ricardo Martinez. He can be reached in a variety of ways:

Ricardo Martinez
8391 Sunfish Lane
Mainville, OH 45039
513.398.2990
remartinez@eworld.com

Inquiries about site licenses, sales, and customer service should be directed to:

Creative Digital, Inc.
293 Corbett Avenue
San Francisco, CA 94114
415.621.4252
415.621.4922 (fax)

cdi@cdigital.com
74774.50@compuserve.com
cdigital@eworld.com

We want to make sure that Ragout remains a useful, innovative product. If there are any features that Ragout doesn't have that you think would make it more useful, please let us know.

APPENDIX A - A BRIEF INTRODUCTION TO SOUPS

Unlike many PDAs in the market today, the Newton was designed from the ground up. One of the new concepts introduced by the Newton is Soups. Soups are used to store data on PC cards (formerly PCMCIA cards) or in the Newton's internal storage. Soups are similar to database files but they are much more extensible and dynamic.

This section of the Ragout manual is a very brief introduction to soups and stores. We strongly recommend that you read Chapter 7 ("Data Storage and Retrieval") of the *Newton Programmer's Guide* for more detailed information about the Newton storage system.

Current Newtons contain two physical storage areas: internal and card. These areas are referred as *stores*. Stores can contain soups or applications (other user-installed packages). The sizes of these stores vary depending on the model of the Newton or its PC card capacity.

Data that needs to be stored permanently must be entered in a *soup*. All information that does not reside in a soup is lost if the application closes or if the Newton is restarted. An application may use one or more soups to store its data.

The Newton provides two types of soups: *union soups* and *plain soups*. The difference between these two is that a union soup creates a soup in every store in the Newton. Plain soups are created only in a single store.

When a soup is created, a list of *indices* must be provided. These indices are specified by their structure, path, and type of slot. Indices are used in the creation of *cursors*, pointers to specific entries in a soup. When using a particular index, the cursor points only to those entries that are qualified by the index. This means that by carefully designing a soup's entries, the cursors may access a subset of entries or all the entries in a soup.

For example, let's say we have a heterogeneous soup (all entries do not have the same slots) with 1000 entries. Furthermore, let's assume there are two types of entries: 20 `type1` entries and 980 `type2` entries. These entries may look like the following (the values listed in <brackets> indicate the slot types):

```
type1 := { Name:  < String >,
           Age: < Integer >, };
type2 := { Name: < String >,
           Phone: < String >, };
```

Let's assume the soup has three indices based, respectively, on the following slots: `Name`, `Age`, and `Phone`. If we create a cursor on the `Name` slot all 1000 entries can be accessed.

If we create a cursor on a non-common slot among the two types of entries, however, we can only use that cursor to access the entries that contain that slot. For instance, if we create a cursor on the `Age` index, we can only access 20 entries in the soup, those with an `Age` slot. This feature prequalifies entries and provides a method for efficient soup access.

It's important to note that the Newton OS adds additional information to frames when these are converted to soup entries and added to a soup. One piece of information added to the frame is the `_uniqueID` slot. This slot is used by the built-in index to access all the soup entries.

MORE NEWTON DEVELOPER TOOLS

PDA DEVELOPERS™
PDA Developers magazine provides in-depth technical information about developing PDA software, focusing on Newton, Psion, GEOS, and Magic Cap devices. Each issue includes news and announcements, programming tips and techniques, product reviews and previews, and programs with in-depth developer descriptions.

PDA DEVELOPERS - SOURCE CODE DISK
Each issue of the PDA Developers source code disk includes the text of each article, the source code for each program in the issue, plus developer-oriented freeware, shareware, and goodies. For subscribers that receive just the disks, we include a print image of the actual printed issue. Available in Mac and Windows versions.

NEWTRTFM™ - ELECTRONIC NTK REFERENCE
NewtRTFM is an electronic summary of the NTK 1.0.1 documentation. A Newton program with superior navigation and searching, NewtRTFM makes the NTK printed documentation obsolete and speeds prototyping and development. For more than 750 topics, the program includes a summary, argument descriptions, and sample code.

GIZMOBEAM™ - MAC/NEWTON IR
GizmoBeam is a device driver for beaming between Mac programs and Newtons. It includes sample code with Think C and CodeWarrior source and detailed documentation. Requires knowledge of the Macintosh Device Manager and Serial Device Drivers, and CE-IR2, 3, or 4 hardware. Distribution licenses are extra.

MICROWAVE™ - DOS/NEWTON IR
MicroWave is a linkable library for beaming between Newtons and DOS-based hardware. It includes a C header file, Microsoft and Borland libraries, three samples with full source, and docs. Requires CE-IR2, 3, or 4 hardware. Distribution licenses extra.

To order any of the above products, please contact:

CREATIVE DIGITAL, INC.
293 Corbett Avenue, San Francisco, CA 94114
415.621.4252 • 415.621.4922 (fax)
cdi@cdigital.com • 74774.50@compuserve.com