NCDraftCard

By: Ramana Rao (Rao.pa@Xerox.COM)

Stored: {qv}<notecards>1.3k>library>NCDraftCard, .dcom, .ted
Last updated: Mar 30, 1987.

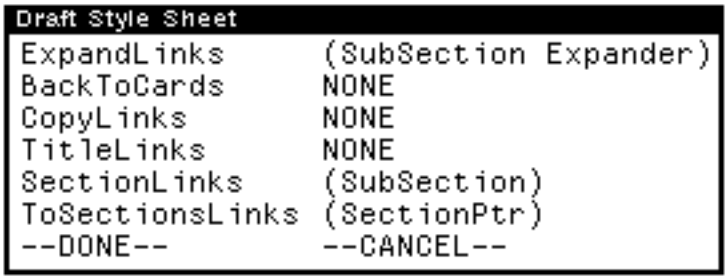
INTRODUCTION

This NoteCards library package defines a new card type called draft cards which supercede document cards. A draft card, primarily, collects the substance from a tree of cards starting from a set of root card, but also does auxiliary processing like sectioning and titling. (The process can be viewed and is implemented as a non-cyclic traversing and interpreting of a notecard network by invoking methods to process card and link types. The draft card defines a set of interpretation methods which implement functionality for document generation).

When a draft card is created, the user is asked to select a set of root card (using the standard NoteCards card selection interface). After this selection, the user is presented with a menu for defining the link and card categorys which are processed by the draft generator. After the draft is generated, a user can use the left button title bar menu to edit the draft style and recompute the draft. A programmer's interface function will be implemented on demand (or rather courteous request) or when I see a break in the clouds. Also true for a long list of desirable features.

CREATING A DRAFT

Draft cards are created by selecting Draft from the NewCards menu (assuming the NCDraftCard library package has been loaded). The user selects a set of root card for generating the document. Then the following draft style editor pops up allowing the user to edit the fields by selecting them. (This is not all there is to a document style as I will say below).



Selecting the items on the bottom line will generate the draft or cancel the creation. Each of the other lines represent a category of links or cards which provides specific functionality during draft generation. The *ExpandLinks* category define the link types which are expanded during draft generation i.e. the substance of the target card (the card pointed to by the link) is pulled into the draft and processed recursively. To prevent runaway loops, a card is only expanded once. The *BackToCards* category define the card types for which back links are put into the draft. Back links allow the user to return to the "underlying" card from the draft. *CopyLinks* are copied directly into the draft. *TitleLinks* produce titles in the draft by inserting the title of the card pointed to by the link. Similarly, *SectionLinks* produce section headings, but in addition they generate section numbers. The section numbers are generated according to "depth in tree using *SectionLinks*" i.e. the section number of a card has as many numbers as *SectionLinks* have been traversed to get to it. *ToSectionLinks* produce references to sections within the draft. Whenever a link of the *ToSectionLinks* category is encountered, the target card is checked for a section number and a reference of the form, "Section xx", is produced. This works for all sections which are part of the draft independent of whether they are before or after the section reference. All categories can be set to ALL, NONE, or a list of link or card types (using the interactive type selector which pops up when the user selects on the field).

The initial settings of the draft style sheet are inherited from a global default style. Any changes made to the style sheet only apply to the associated draft and do not affect the global default settings. To change the global default settings in the current sysout, the daring user can inspect and edit *ncdraft-default-style*. To maintain these defaults across sysouts the user can set *ncdraft-user-props* (a property list using the properties you see when you inspect *ncdraft-default-style*) in her init file. The user can also inspect and edit a draft's style by pulling across on "Edit Draft Style" item in the card menu. Inspecting styles will reveal a host of other parameters which will influence the draft generation process and will also more annoyingly reveal that the field names which correspond to the link and card types have different names then in the style sheet editor. Use the field name you see in the inspector for adding an entry to *ncdraft-user-assoc* in you init file. (This is all hoaky, but at least functional. You are welcomed to bug me about how you can do something).

Alternatively draft cards can be created programatically as follows:

(NCP.MakeDraft <NoteFile> <rootcard> <style-assoc-list> <nodisplayflg> <props> <parent-fileboxes>)

Currently Unimplemented. Bug me when you really need this. Creates and returns a draft card starting from <rootcard>. All parameters not set by <style-assoc-list> will be inherited from the default style. The resulting draft card will have the given props and parents.

DRAFT CARD OPERATIONS

The Draft Card operations are available via the left button (card) menu in the Draft Card card's title bar. **Recompute Draft** recomputes the draft using the associated root card and style. **Edit Draft Style** pops up the style sheet described above and allows you to change the link or card type categories. **Edit with Inspector** is a pull-across item of **Edit Draft Style** which allows you to inspect and edit all of the style parameters. Please READ the last section if you are going to do this.

KNOWN BUGS

Ha! All features, non-features, or future features.

FUTURE FEATURES

User Interface Improvements: Allow the user to edit all the style parameters with a style sheet editor that is like a form (like what FREEMENU would be if it were what it is trying to be). Right now my focus is more on what stylistic things a user might want to dictate, and not on how he will set these parameters.

Automatic generation of Table of Contents. You can do this in a separate draft card with the current functionality by being a little clever at defining which link types to expand, copy, section, or title depending on how you want it to come out.

Sectioning and Title Style parameters. Font choice and other looks options. The hidden style parameters are a small step in this direction, allowing bolding/notbolding of these as well as dictating number of carraige returns before and after a section.

Figure Numbering and Figure references.

Bibliographic References. You can use titling to achieve this with great pain currently, but better ways are coming along. A quick and dirty hack first, and then a full service hack that Miller is working on.

Draft/Network Coupling. Editting the draft would automatically edit the underlying substance. Not an easy feature to provide, but probably a very useful one.

Generate Draft to Tedit Stream.