An Experimental Comparison of New Property List Designs

John Panzer and Linda Werner
Dept. of Computer and Information Sciences
University of California Santa Cruz
Santa Cruz CA 95060 USA
UCSC-CRL-95-43 December, 1995
panzer@cse.ucsc.edu and linda@cse.ucsc.edu

ABSTRACT
Property lists are a common way to display and edit object attributes. They are often organized alphabetically, which can lead to usability problems when the number of properties grows large. Two alternative property list designs, categorical lists and split lists, are presented. An experimental comparison found that categorical lists can reduce selection times by 62% for novice users, while split lists show potential for a 43% reduction.

Keywords
Property sheet, property list, split list, categories, grouping

INTRODUCTION

Property lists allow users to edit object properties in a spreadsheet–like interface. Property lists are compact, simple, and extensible. They are particularly useful in environments with many object types and are used in several interface builders [1,2,5] to display and edit the properties of interface elements. A property list usually displays properties for the currently selected object(s) in a separate window, allowing the user to browse object properties simply by changing the selection. A property list can handle multiple objects of different types simply by displaying the intersection of their properties. This would be difficult to do with other kinds of property editors such as specialized dialogs.

Applications often organize property lists alphabetically by property name (Figure 1). This was initially the case with Vibe, an interface builder tool for The Santa Cruz Operation (SCO) Visual Tcl scripting language [7]. Preliminary usability testing of Vibe revealed several problems with its alphabetical property list. For example, users often wanted to edit *Width* and *Height* properties as part of the same task, but the alphabetical list kept these related properties far apart. The alphabetical organization can hinder novice users, who are unfamiliar with property names and so must often scan the entire list when searching for a property.

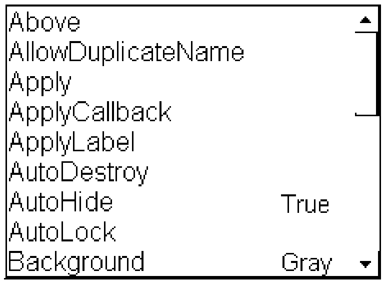

Above
AllowDuplicateName
Apply
ApplyCallback
ApplyLabel
AutoDestroy
AutoHide True
AutoLock
Background Gray

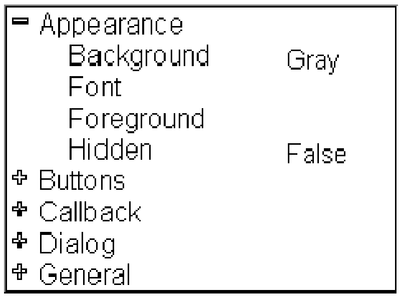
Figure 1 Alphabetical List

In short, the alphabetical organization does not organize the properties in a task oriented way. Advantages of an alphabetical organization are that it is easy to learn and that finding an individual property is easy if the property name is familiar. There are, however, faster ways to find properties if the name is known (by using the keyboard, for example). The alphabetical organization is a compromise between ease of learning and ease of use in which neither goal is completely satisfied.

ALTERNATIVE LIST DESIGNS

Prior to Vibe, SCO used a script based programming environment for creating Visual Tcl applications. Because of this, it was possible to look at existing usage patterns for Visual Tcl object properties. Both of the new property list designs were based on this information [4].

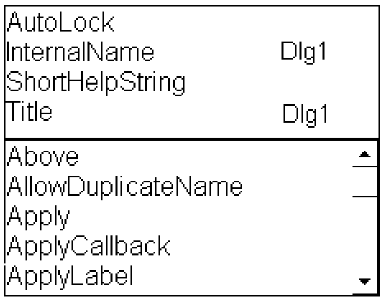
A standard way to organize items on menus is to group conceptually related items together. The same can be done for property lists. A sorting experiment with existing Visual Tcl users revealed a common set of "natural categories" [3] for the properties. These were used to create a *categorical list* (Figure 2) in which properties are grouped according to meaning and function. The list is implemented as an expandable two level hierarchical outline. The first level contains category names, while the second level contains individual properties. This design allows users to display the parts of the category tree they are currently working with, or to rapidly search the properties by expanding and collapsing categories. (Other property lists [1,2] categorize properties by providing separate lists which the user may view one at a time through a set of tabs similar to a notebook.)



− Appearance
Background Gray
Font
Foreground
Hidden False
+ Buttons
+ Callback
+ Dialog
+ General

Figure 2 Categorical List

Sears [6] suggests *split menus* as a way to cache frequently accessed menu items near the top of a long menu. A search through existing script source code found that a small subset of the properties accounted for roughly 50% of property usage. The *split list* (Figure 3) takes advantage of this skewed distribution by placing copies of a few commonly used properties in a nonscrolling cache located above the main list.



AutoLock
InternalName Dlg1
ShortHelpString
Title Dlg1

Above
AllowDuplicateName
Apply
ApplyCallback
ApplyLabel

Figure 3 Split List

It is also possible to combine the two approaches, producing a *split categorical* list. In this organization the cache list is an alphabetical list while the main list is a categorical list. Thus there are four possible interface designs: Alphabetical (the baseline), categorical, split, and split categorical.

EXPERIMENT

An experiment was conducted to determine which interface to implement for Vibe. Six "expert" subjects had experience with Visual Tcl properties, while six "novice" subjects had no experience. The subjects found properties based on short descriptions. They used simulated property lists which contained approximately 40 categories. The hypotheses were that the categorical interface would benefit novices, while the split interface would primarily benefit the experts.

The categorical interface produced the fastest selection times for novices, yielding a 62% average reduction in time over the alphabetical list. For the expert group, there was no significant effect on selection time. The experts did select properties from the cache lists in 72% less time than from the main lists of the split and split categorical interfaces. This indicates that a split list could produce a substantial (43%) reduction in selection time over the alphabetical list, given the 50% cache hit ratio predicted in the split list design. Based on subjective numerical ratings of the interfaces, both groups strongly preferred a categorical organization. These results are significant at the .05 level (t-test).

CONCLUSION

Alphabetical property lists grow unwieldy as the number of properties increases. Grouping properties according to function or frequency, following the example of menus, can provide substantial benefits. The next version of Vibe implements a categorical list with an optional, user configurable cache. This version will be used to investigate whether the cache is useful under field conditions, and whether the categorical organization holds up under sustained use.

ACKNOWLEDGMENTS
The Santa Cruz Operation's Human Factors department was instrumental in usability testing for this project.

REFERENCES

1. Delphi User's Guide. Borland International, 1995.
2. Microsoft Visual Basic Programmer's Reference. Microsoft Press, 1993.
3. Paap, K., and Roske-Hofstrand, R. The optimal number of menu options per panel. *Human Factors 28*, 4 (1986), 377-85.
4. Panzer, J. *Property Lists: An Experimental Comparison of New Property Editor Designs.* A M.S. thesis, University of California at Santa Cruz. Available at <http://www.cse.ucsc.edu/~panzer/th/>.
5. Pausch, R., Young II, N., and DeLine, R. Suit: The pascal of user interface toolkits. In *UIST '91* (1991), 117-125.
6. Sears, A. and Shneiderman, B. Split menus: Effectively using selection frequency to organize menus. *ACM Transactions on Computer-Human Interaction* (March 1994), 27-51.
7. Young, D. SCO Visual Tcl: A new tool to crack the Motif barrier! Draft for SCO World Magazine, November 1994.