BACKGROUNDMENU

By: Mike Dixon
New Owner: Burwell (Burwell.pa@Xerox.com)

INTRODUCTION

If you love to load all those fun LispUsers packages but can't deal with background menus that look like this:

don't despair! With just a few quick calls you can have a background menu that looks like this:

(don't worry, they didn't disappear, they're just hiding under "Exec").

DESCRIPTION

BackgroundMenu defines several functions for rearranging your background menu to suit your taste.

(BkgMenu.rename.item *item newname*) [Function]

changes the name of a background menu entry

(BkgMenu.move.item *item superitem atend*) [Function]

makes *item* a subitem of *superitem*. If *atend* it is placed after any subitems of *superitem*; otherwise it is placed before them. If *superitem* is NIL *item* is placed at the top level of the menu.

(BkgMenu.reorder *items superitem atend*) [Function]

just like BkgMenu.move.item but moves a list of items. Useful for changing the order of the items in a menu.

(BkgMenu.remove.item *item*) [Function]

throws *item* out of your background menu.

(BkgMenu.fixup) [Function]

BackgroundMenuTopLevelItems [Variable]

BackgroundMenuFixupMode [Variable]

each top level item which isn't on the global BackgroundMenuTopLevelItems is made a subitem of BackgroundMenuSuperItem. If BackgroundMenuFixupMode is 'top they're added before any subitems of BackgroundMenuSuperItem, if it's 'bottom they're added after, and if it's NIL items moved from the top are added at the top and items moved from the bottom are added to the bottom.

(BkgMenu.subitems *item*) [Function]

returns a list of the subitems of *item* (or the top level items, if *item* is NIL).

(BkgMenu.add.item *item superitem atend*) [Function]

adds a new menu item *item* as a subitem of *superitem*. If *atend* it is placed after any subitems of *superitem*; otherwise it is placed before them. If *superitem* is NIL *item* is placed at the top level of the menu

EXAMPLES

As an example of using BackgroundMenu, this is what I've got in my init file (which produces the changes shown above) (note that i've already loaded LISTEN):

```
(BkgMenu.rename.item "Lisp Listener" " Exec ")
        (* "Lisp Listener" is just too long.  the blanks before and after Exec are just there to
        improve the spacing)

(SETQ BackgroundMenuTopLevelItems '(Idle Snap " Exec " Chat PSW TEdit))

(SETQ BackgroundMenuSuperItem " Exec ")

(BkgMenu.fixup)
        (* Push everything i don't use regularly under the now-renamed Lisp Listener)

(BkgMenu.reorder.items BackgroundMenuTopLevelItems)
        (* and put the top level items in the order i prefer)
```

If I later add more packages which add junk to the top level of my background menu, just calling (BkgMenu.fixup) again will hide anything new under " Exec " with the rest of the junk.

When any of the above functions (except BkgMenu.add.item) require you to specify an item, you can usually just give a string with the menu entry (or an atom, which is coerced to a string). The case has to be correct, and blanks have to be in the right place. The function will do a breadth first search of the background menu and all its submenus to find such an entry. If for some reason you have the same entry in more than one menu, you'll have to disambiguate it. To do this, you pass a list for the item, where the first thing in the list is the menu entry, and the rest of the list is a path through the tree to find it. For instance, the item (one two three) means find an entry whose text is "three", then find an entry in the tree underneath it whose text is "two", and the find an entry under that whose text is "one".

The item argument to BkgMenu.add.item is a standard menu item, i.e. a list of (label form help.string).

All of the functions return T if they were able to do as asked and NIL otherwise (you tried to do something with a menu entry which isn't there, or you tried to make a circular menu structure). The only exception to this rule is BkgMenu.subitems, which as previously mentioned returns a list of the subitems, or the atom NotAnItem if it's given a nonitem.