Alert, **Menu**, **Open** and **Save** are simple wrappers for the *NextSTEP* NXRunAlertPanel routine and OpenPanel and SavePanel objects that allow them to be used from (interactive) Unix shell scripts (to provide consistent GUI panels across *C* and shell executables).

NIDomain, **NIOpen** and **NISave** are wrappers for the *NIKit* NIDomainPanel,NIOpenPanel and NISavePanel objects that allow them to be used from Unix shell scripts. (The included wrapper for NILoginPanel doesn't work.)

Alert takes the message to display and three button labels as arguments:

```
Alert [message] [defaultButton] [alternateButton] [otherButton]
```

All arguments are optional--buttons are not drawn if labels are not supplied. The return value of NXRunAlertPanel is remapped into something useful for shell scripts; for the first (default) button (first button argument but first from the right on the panel), it returns *0* (success), otherwise *1* or *2* for the other buttons and *3* on error.

Thus, you can do things like:

```
Alert 'Are you sure?' 'Yes' 'No' 'Help'
if ( $status == 2 ) ...
```

There is no access to the printf-like features of NXRunAlertPanel as you can provide the same functionality from within the shell script language, eg: Alert "Using the current time (`date`)"

The *title* of the alert panel is taken from the title of the program, so you can have a **Warning** panel by doing 'ln -s Alert Warning' where **Alert** is stored (eg. /usr/local/bin).

Menu takes several optional arguments and at least one required menu item:

```
Menu [-c char] [-t title] [-x xpos -y ypos] [-T seconds] item [item [...]]
```

Where ***char*** is the character from which to start assigning key equivalents ('1' by default) , ***title*** is the title of the menu (the name of the Menu program by default), ***xpos*** and ***ypos*** are the coordinates where the menu should be displayed (upper left corner by default)--both positions must be specified if either is and ***seconds*** is the time to wait before timing out (30 seconds by default)--a value of 0 prevents a timeout.

Menu exits with status = 0 on success (a menu item was selected) and prints the selected item to stdout . If the menu timeout occurs, it exits with status = 1 (failure) and prints nothing. If given bad arguments, it exits with status = 2 and prints a usage message to stderr. An example use of **Menu** from csh:

```
set zones = `find /etc/zoneinfo ! -name '*GMT*' -type f -exec basename {} \;`
set zone = `Menu -t 'Select Time Zone' $zones`
if ($status == 0) setenv TZ $zone
```

Open and **Save** take several optional arguments:

```
Open [-f file] [-d directory] [-t type [-t type]] [-p prompt] [-T title] [-m] [-c]
Save [-f file] [-d directory] [-t type] [-p prompt] [-T title] [-u]
```

Where ***file*** is the file to display in the form field of the panel, ***directory*** is the directory to display in the browser portion of the panel, ***type*** is the type (extension) of file(s) to accept (**Open** can accept multiple type arguments), ***prompt*** is the title of the form field (defaults to *Name:*) and ***title*** is the panel title (defaults to *Open* or *Save*). The ***-m*** option to **Open** allows multiple file selection and the ***-c*** option allows the user to choose directories. The ***-u*** option to **Save** allows the user to travel inside file packages.

Both **Open** and **Save** exit with status = 0 on success and print the (fully expanded) file name(s) to stdout (one per line). If the user selects *Cancel*, they exit with status = 1 (failure) and print nothing. If given bad arguments, they exit with status = 2 and print a usage message to stderr. An example using **Open** from csh:

```
set files = `Open -t txt -m`
if ($status == 0) then
        cat $files
else
        echo 'no files selected'
endif
```

NIDomain doesn't take any arguments and exits with status = 0 on success (a domain was selected) and prints the selected domain name to stdout . If given bad arguments, it exits with status = 2 and prints a usage message to stderr. If the user selects *Cancel*, it exits with status = 3 (cancel) and prints nothing.

NIOpen and **NISave** take several optional arguments:

```
NIOpen [-d directory] [-t title] [-p prompt]
NISave [-d directory] [-t title] [-p prompt] [-f directory]
```

Where ***directory*** is the *NetInfo* directory to display in the domain browser portion of the panel, ***title*** is the title printed at the top of the panel and ***prompt*** is the title of the form field (defaults to *List Title:*). The ***-f*** option to **NISave** sets the initially selected directory.

Both **NIOpen** and **NISave** exit with status = 0 on success and print the (fully expanded) file name(s) to stdout (one per line). If the user selects *Cancel*, they exit with status = 3 (cancel) and print nothing. If given bad arguments, they exit with status = 2 and print a usage message to stderr.

All the panels display the default NeXT logo as their application icon.

Comments and/or suggestions welcome.

Christopher Lane (lane@sumex-aim.stanford.edu).