The MiscKit

Library:	libMiscKit.a
Header File Directory:	/LocalDeveloper/Headers/miscKit
Import:	miscKit/miscKit.h

Introduction

The MiscKit is a collection of useful classes and Interface Builder™ palettes donated by a multitude of authors around the world to speed up development of mission critical applications. A large number of the MiscKit objects are available on InterfaceBuilder™ palettes to further simplify development of applications. Most of the objects found in the MiscKit fall into one of several categories:

- Foundation classes.

- Data encapsulation classes.
- Classes which encapsulate often used UNIX™ and Mach™ protocols.
- User interface extensions.
- Useful examples from test suites used to validate MiscKit objects to complete example applications.

The MiscKit is constantly growing and improving. If you have suggestions for improvement, forward them to the appropriate authors or to the MiscKit developers' mailing list misckit@byu.edu. If you would like to make a contribution to the MiscKitÐand you are heartily invited to do soÐsimply contact the MiscKit administrator Don_Yacktman@byu.edu.

MiscKit Classes and Protocols

Below is a diagram showing the heirarchy of MiscKit classes.

There should be a diagram right here in this very spot.

Figure 1. MiscKit classes

Foundation classes

There are several classes which implement basic data structures. For containing lists of other objects, there are the MiscList, MiscStorage, MiscSortedList, MiscSortedStorage, MiscLinkedList, MiscQueue, MiscPriorityQueue, MiscStack, and MiscTree. Both MiscList and MiscStorage extend standard NeXT™ objects to use a cursor. For keeping sorted lists of objects or data structures, use MiscSortedList or MiscSortedStorage respectively. The MiscQueue implements a FIFO (First In, First Out) data structure. If you need to line up objects in a prioritized order, use the MiscPriority queue, which sorts objects according to their priorities, using a heap data structure internally. If you need a LIFO (Last In First Out) data

structure, use the MiscStack. Use the MiscTree to create tree data structures with arbitrary numbers of children on each branch. Future versions of the MiscKit will probably have other basic data structures included as they become available.

Data classes

Several classes in the MiscKit encapsulate simple datum and provide for easy manipulation. For example, MiscTime handles time to just about any degree of accuracy you need, from days of the year down to the nearest microsecond. There's even a MiscStopwatch which allows tracking of elapsed time with high accuracy. For text string handling, there is the MiscString class. There are also three classes which make color manipulation easier and more powerful than ever before: MiscColor, MiscHalftoneColor, and MiscScreenColor. The MiscClassVariable allows creation of a true class variable, an improvement over using static variables.

OS wrappers

Some of the MiscKit classes wrap around common OS features. With the MiscKit, you can easily make use of the serial ports (MiscSerialPort), do simple file locking (MiscLockFile), and log errors to a file (MiscLogFile).

Appkit™ extensions

The MiscKit also provides new UI widgets and enhances several of the AppKit™ classes. The MiscFindPanel makes it easy to add a Find menu and Find... panel to an application. The MiscSearchText category extends the AppKit's™ Text object so that it can work seamlessly with the MiscFindPanel. The MiscCoolButtons palette contains several prototypes of commonly used buttons, saving developers the time otherwise required to put together all those little buttons every app seems to need. The MiscClockView implements a clock and calendar like the one seen in Preferences.app™. Using the various progress views (MiscProgressView, MiscProgressBar, and MiscProgressPie) makes it easy to give the user feedback about long-running operations. The MiscProgressPie looks just like the one seen in Installer.app. There are also new types of buttons in the MiscKit. The MiscThreeStateButton is a button which can take on a third state and the MiscArrowButton

implements a simple way to choose between two possible options while showing both options simultaneously, but with an appearance which is more elegant than two radio buttons. The MiscValueField combines a TextFieldCell and two ButtonCells to create a new kind of cell, with up/down arrows, which may be used to select among specific values in a list of strings or step through numeric values. The MiscSliderField is similiar, combining a Slider and TextField in a single object. The MiscReadOnlyColorWell allows creation of color wells which are read-only. The MiscDragViews include an abstract superclass and both an Image Well and Icon (File) Well. MiscMatrix is a subclass of Matrix within which the rows and columns may each have their own widths. Finally, there is the ExtendedApp class which provides the programmer with extra information about the runtime environment.

The MiscKit also contains several classes which may be used to painlessly implement a typical Info submenu including an animated Info... panel, registration and order form panels, suggestions to the author, and other features. The MiscInfoController, MiscInfo, MiscOrderForm, and MiscRegistration classes work together to provide this service.

If you need swapping views like in the Preferences™ application or inspectors, there are two kits (MiscSwapKit and MiscInspectorKit) which contain basic classes that may be used as a starting point. At the moment there is no tutorial for theseÐit is in the worksÐso it is recommended that you obtain the source code for the ªBeakerBoyº application to see how these objects can be put to use in a real application.

A complete document control system in currently in the works for the MiscKit; this will make it very easy to create multiple document applications using the MiscKit by extending the framework of the AppKit even farther.

Example applications

There are two main types of example applications in the MiscKit. First are the validation/test suites which were used to test the MiscKit objects. These simple programs, usually run from the command line, put the MiscKit objects through their paces, some more thoroughly than others. There are currently programs to test the MiscString, MiscLogFile, MiscLockFile, and MiscPriorityQueue classes.

There are also applications which make use of MiscKit classes in a setting which is (at least slightly) useful. There is MiscStringService, which uses the MiscString class to provide a few useful Services menu items. The TreeView application displays a scrolling view of a tree, useful for the display of heirarchical data such as inheritance trees. If you'd

like to compare the speed of the MiscKit's searching algorithms to the slower built in routines, try out SearchBench. You'll be amazed. As a simple example of the MiscSerialPort, there is the TinyTerm application which allows you to send and receive data through one of the serial ports. The receiptfilter program allows you to monitor outgoing NeXTMail read-receipts and choose to send (or not send) them.

Also included are several templates which can be used by developers to create documentation which follows the same styles as the NeXT™ on-line documentation. Sample InterfaceBuilder™ files are also provided to demonstrate parts of the MiscKit, such as the Info menu. These files can also be used as templates by developers.

MiscKit Functions

The MiscKit, although it is primarily a library of objects, also includes several useful functions. Most of the currently documented functions provide raw searching and pattern matching algorithms.

Fast searching algorithms

There is a series of functions which implement the Tuned Boyer-Moore algorithm for searching literal string patterns.

- Misc_TBMKpattern_alloc(), Misc_TBMKpattern_free(), Misc_TBMKsearch()

Regular expression searching package

The MiscKit includes the public domain regular expression searching package written by Tatu Ylonen. This package includes the following functions:

- re_set_syntax(), re_compile_pattern(), re_match(), re_match_2(), re_search(), re_search_2(), re_compile_fastmap(), re_comp(), re_exec()

All trademarks used herin are owned by their respective owners. We're just borrowing them to give you a frame of reference by which you can understand what the MiscKit does.