 Version 1.7.1.

MiscKit Charter

The MiscKit is a kit of resources provided by members of the USENET and InterNet community for use by anyone who develops under NEXTSTEP and/or Objective-C. Changes, bug fixes, and suggestions for improvement in the kit are all welcome and encouraged. Programmers can submit code and tools, while users can suggest ways to improve what is here, to the benefit of all.

In an effort to keep all this in some semblance of order, a few ªrulesº are in place. These rules aren't meant to make things difficult; they are intended to make things run smoothly and without confusion. If this goal isn't being met, then the rules will need to be adjusted so that they are met; send your suggestions to the MiscKit mailing list for discussion there. There is no reason why changes can't be made to improve things.

Altering;¬Altering the License and/or Charter is done by a vote of MiscKit authors and owners. Each author has one vote; if your name is in the Authors.rtf file as an author, contributor, or owner, you have the right to cast a vote. The author and owner of a resource are the same person unless some written

agreement exists which transfers ownership. When approval is requested for a particular change, failure to respond within two weeks of such a request constitutes a ªYESº vote. Voting is initiated by the MiscKit Administrator; the administrator will prepare a new license or charter with the proposed modifications and then present the new version for voting. Note that although changes are allowed in the charter and license, use and distribution of the MiscKit is free, and must remain so. That means that no party may come in, acquire the MiscKit, and then turn around and charge for it, nor may they impose any new restrictions on the distribution of the MiscKit. Changes to the license affect future releases of the MiscKit, but not earlier releases.

Donations;¬Donated code falls under the MiscKit license. There are certain special cases where code is included in the MiscKit but retains its own license. In these special cases, the code does not fall under the MiscKit license. This situation is highly discouraged, and it is up to the MiscKit administrator's discretion to decide whether or not to include such ªindependentº code. By contributing to the MiscKit, the resource owner agrees unconditionally to the terms detailed in the license and charter. (If you don't like the license and/or charter, you shouldn't contribute.) The owner cannot contribute under different terms without special permission from the MiscKit administrator. An owner cannot remove a contribution once it has been contributed. (See License.rtf and License_Notes.rtf for more information on this.) By contributing a resource to the MiscKit, the owner agrees to follow this policy and is giving up the right to remove the contributed

resource from the MiscKit.

;¬Accepted contributions include foundation objects, interface objects, programming tools, Interface Builder palettes, and Project Builder .bproj (subproject) directories. All contributions should be accompanied by .rtf documentation if possible, but this is not required. If no documentation is provided, some willing soul will be assigned the task of keeping the documentation up to date. Typically, this is easiest to do if the code maintainer (see below) is the one who does documentation, and this is encouraged, even though it is not required.

Prefix;¬For consistency, all MiscKit objects, including functions, compiler defines in header files, Objective-C objects, protocols, and defined types will be prefixed with ªMiscº. (Compiler defines should use ªMISCº.) Exceptions to this rule will only be made in cases where a better solution is impossible, or the Misc prefix would cause a problem worse than the potential name conflicts. The MiscKit administrator must give permission for any exceptions. This prefix is intended to assure that the MiscKit's name space will conflict with no other Objective-C name space or kit. Functions, variables, or preprocessor defines seen only in the source files and not in the header files are not required to use the ªMiscº or ªMISCº prefix. Note that Objective-C categories are *not* required to use the ªMiscº prefix.

Resources;¬Some MiscKit objects require the use of external resources (tiff images, sounds, .nib files, string tables, etc.). Currently, there are two ways to

provide for this. First, for an object in the MiscKit library (libmisckit.a), the resources needed should be added to a given project in ProjectBuilder. Second, if the object is available as a ProjectBuilder .bproj file, the subproject should be added to the project; the subproject will contain all necessary resources, which will in turn be incorporated by ProjectBuilder into the project. This is preferred because the resources are encapsulated along with the object that uses them. If enough shareable resources become part of the MiscKit, a third alternative will also be implemented, which at this time is unimplemented, and will be known as the MiscKit runtime library[2];Charter.rtf;DaggerFootnote;¬· It is hoped by most that a pool of shareable resources never becomes necessary.

Administration;¬MiscKit maintenance and integration is overseen by the MiscKit administrator, Don Yacktman. If changes are needed to comply with kit standards and requirements, such as the prefix, the administrator will work with the submitter to bring the resource into compliance. The administrator will accept requests from third party distributors who wish to redistribute the MiscKit and provide them with the latest versions of the MiscKit as well as notification of new releases as they occur. Notification of new releases, performed via an e-mail alias, is available to anyone upon request, and will also be sent to the MiscKit e-

mail alias. The administrator may resign his/her post at any time, in which case a new administrator may be chosen and ratified by a simple majority of the MiscKit authors, and should be chosen based upon ability and resources to do a respectable (good) job keeping things up to date.

Maintenance of an individual MiscKit resource is to be handled through the resource's maintainer, as listed in the Authors.rtf file in the MiscKit distribution. That maintainer will forward tested and completed changes to the administrator for inclusion in the official release of the kit. The administrator will also forward changes back to the maintainer rather than just putting them straight into the kit. (ie., just because the administrator makes the distributions does *not* mean he or she can break this rule and change what an author sends to him!) The maintainer accepts bug reports, code additions and enhancements, and any other feedback pertaining to the resource in question, and is then expected to forward the final revisions, after testing, to the MiscKit administrator for inclusion in the MiscKit. The maintainer for a resource is typically the owner of a resource, which, by default, is the original author. The owner is the person who owns the copyright for

a resource; this is the original author, unless a signed agreement transferring the copyright to another individual exists. The owner has the right to choose a maintainer for the resource. In the event that the owner forfeits the right to choose a maintainer, a maintainer will be chosen from the MiscKit authors, or any other suitable volunteer, by the MiscKit administrator. Although a resource included in the MiscKit cannot be removed by its owner, if the owner decides they do not wish to be a MiscKit maintainer, but does wish to retain ownership for any reason, they may do so and a maintainer will be chosen as delineated above.

New features to the kit and architectural decisions should be discussed in the MiscKit mailing alias so that things are designed right before any code is written. This is not a strict requirement; it is OK to simply submit a resource, but discussion is recommended because it will make for a better design in the long run, and this is the main purpose of the mailing alias anyway. This alias is unmoderated; participants are expected to use their own good judgement. Flame wars are highly unwelcome; only constructive discussion is desired. (Of course, disagreement is expected; without opposing viewpoints it is impossible to consider

all eventualities and thus all viewpoints are to be respected, especially when the topic is design of a particular resource.) Currently, the MiscKit e-mail alias is miscki@byu.edu. Send mail to misckit-request@byu.edu to get on or off the list.

Any pending projects for the MiscKit will be listed in a To_Do.rtf file in the MiscKit distribution. This will include things such as ªobjects we'd like to seeº, ªknown bugsº, ªdevelopment tools we'd like to addº, and so on. Anything that would be nice to have, but isn't a current work in progress would turn up in that file. An In_Progress.rtf file will list any bugs which are currently being chased, any deficiencies which are being dealt with, and any projects in progress. The misckit_proj/Temp directory will hold any projects that are in progress, but are not yet integrated into the MiscKit, if the author wants to make code available for public comment. It is not mandatory to provide the MiscKit administrator with code in progress; this directory is provided solely for those who wish to use it. Nothing in the Temp directory is guaranteed to work. It may work, it may not. Note that items in the Temp directory are not reviewed by the MiscKit administrator, as they are still in the process of becoming an actual part of the MiscKit.

Versioning;¬Versions of the MiscKit follow a form of x.y.z where x is the major version number, y is the minor version number, and z is a sub-version number. The sub-version number is incremented for bug fixes and minor additions to existing resources. The minor version number, y, is incremented when a new resource is added to the MiscKit, or a large enough number of changes to existing resources has been made. The major revision number, x, is incremented for major architectural changes in the MiscKit which affect several resources, or when the group of MiscKit authors feel that it should be incremented. Whenever y is incremented, z is reset to zero, and whenever x is incremented, both y and z are reset to zero. Major and minor releases will be stored on the official MiscKit archive site (currently ftp.byu.edu) and will also be forwarded to the standard NEXTSTEP ftp archives, such as cs.orst.edu. Bug fixes may or may not be forwarded to the major archive sites.

Backward;¬Backward compatibility between versions is not always guaranteed. Minor revisions will be compatible with any previous version of the same major revision level. Between major revision levels, however, nothing is

guaranteed. The programming interface will be kept as uniform as possible. Incompatibilities will occur when it is better to move on and use improved code than it is to keep crufty, old, poorly written code around. In other words, unlike the DOS world, we refuse to keep dumb code around for compatibility's sake alone. The whole point of the MiscKit is to provide a useful kit of state of the art objects. It should be the best that it can be. (To the users of the MiscKit: Note that backward compatibility *will* be attempted, but it will only be achieved in cases where it is not a hindrance. Most of the time, backwards compatibility will occur, so don't worry about us turning your world upside down. We'll try not to do that. Really.)

DaggerFootnote;¬²Although it does not currently exist, the basic shape of a runtime library is already partly fleshed out, in case of the event that it become needed. Basically, it would work like this¼ first, it would be completely contained in a directory that a user would install on their system, installable in either /LocalLibrary (preferred) or in ~/Library. (Another possibility, mimicking

NEXTSTEP's shared resources, would be to install in /usr/local/lib or ~/lib.) The folder would be provided in the MiscKit both expanded and as an Installer .pkg file. The .pkg file could then be distributed with any app the uses the MiscKit; if the user hasn't already installed the package, they would do so before installing the app. When installed, subdirectories would be used to specify a particular version of the MiscKit, so that newer resources won't overwrite old resources, since an older third party application might require an older resource. When a MiscKit object requires a resource, it would then ask a manager object (which would be written and added to the MiscKit) to locate the resource. The manager would search, in the following order, the .app wrapper, the user's local ~/lib and ~/Library directories, then /usr/local/lib and /LocalLibrary. Within each of the listed directories, it would search from the newest back to the earliest version of the resource directory. Note that a version of the searcher compiled for a certain revision level of the miscKit would start at it's revision level and work down, thus avoiding the problem of grabbing a newer, incompatible, resource. Also, to avoid duplication of resources, only resources that change or are new to the MiscKit would appear in the higher directory levels (since the search will eventually find the old resources in the lower levels). This system is a proposal, and it, or a similar scheme, will be implemented if deemed a worthwhile project by a simple majority of the MiscKit authors, but only if someone is willing to actually implement it. Currently, there is no perceived need for a runtime library.