8

The Interface to the File System

One of the goals of the NEXTSTEP user interface is to provide a simple, graphical interface to the UNIX operating system. This is largely the responsibility of the Workspace Manager, the application that's brought to the screen after the user logs in. More specifically, the Workspace Manager provides a graphical interface to the file system. It's a substitute for a UNIX command interpreter (or shell) that locates files, displays folder contents, associates files with applications and icons, starts up applications, and keeps track of the user's home folder and working environment. It lets users manage files by manipulating their iconic representations.

For command-line interaction with the operating system, users can put up a window that emulates a VT-100 terminal and runs a standard shell. This window is provided by the Terminal application, which can be started up from the Workspace Manager.

Terminal is documented in the *NEXTSTEP Development Tools and Techniques* manual. The Workspace Manager is documented in the *User's Guide*.

How the File System Is Organized

The NEXTSTEP file system is arranged very much like a traditional UNIX file system. However, by default, when users view the file system in the Workspace Manager, they won't see many of the traditional UNIX folders. Instead, they'll see only a small number of folders that organize the tools and information they most need. Under the root (/) folder, they'll find the following folders:

LocalAdmin
LocalApps
LocalDeveloper
LocalLibrary
me
Net
NextAdmin
NextApps
NextDeveloper
NextLibrary
user

Home Folders

In the list above, the folder labeled *user* stands for the user's home folder. The home folders of other users may also be visible. As is traditional, home folders bear the name of the userÐthe name that the user logs in under. By default, the user's name is **me**, and the home folder is **/me**.

If a machine is connected to a file server over a network, users' home folders are likely to be located somewhere on the file server. The figure above simply illustrates the default location for a home folder and where it would be located if it were on the startup disk. Home folders on remote machines are accessed through the **Net** folder described below.

When typing in a file name, users can specify their home folder as ~. For example, **~/Apps** refers to the **Apps** folder that's in the home folder of the current user.

NeXT Folders

Four of the top-level folders begin with a ªNextº prefixÐ**NextApps**, **NextAdmin**, **NextDeveloper**, and **NextLibrary**. They contain documents, resources, and applications that are bundled with the computer.

The **NextApps** folder contains supported applications for NEXTSTEP users. In it are general-interest applications like Edit, Preferences, Mail, and PrintManager.

The **NextAdmin** folder also contains supported applications, but ones that will be used mainly by system administrators and network managers. They're not for general users like those in **NextApps**.

The main purpose of the **NextDeveloper** folder is to provide applications and files that are necessary for developing NEXTSTEP applications. Some of the folders that can appear in **NextDeveloper** include:

Apps	Holds applications used by developers. This folder might not exist on all systems. Debuggers, profiling tools, language-specific editors, and other window-based programming tools belong here. Like all other applications in the four ªNextº folders, these applications can be run from the Workspace Manager. Command-line debuggers and profiling tools are in standard UNIX folders such as **/bin** and **/usr/bin**.
Demos	Contains programs that demonstrate the capabilities of NEXTSTEP. These aren't full applications and aren't supported by NeXT. However, they include games and utilities that users, not just developers, might find interesting.
Examples	Contains source code for example programs. This folder might not exist on all systems. Its folders contain source code that you can study and compile.

The **NextLibrary** folder contains resource files organized into several folders. Some of the files and folders aren't on every system. Typical folders include **/NextLibrary/Documentation**, which holds NEXTSTEP technical documentation, and **/NextLibrary/Fonts**, which holds the PostScript fonts available to all applications.

Local and Personal Folders

The four folders with a ªNextº prefix can be matched by four identical folders with a ªLocalº prefix. Internally, these four folders should be organized like their ªNextº counterparts. But instead of containing files supplied by NeXT, they should hold information and applications provided for all users at a local site. Any user who logs in to a machine, or boots from it over a network, has access to its ªLocalº folders. If you add a new font for all users of your network, for example, it would reside in **/LocalLibrary/Fonts**.

ªLocalº folders are created as they're needed. They aren't included on the release disk.

Users can add unprefixed **Library** and **Apps** folders in their home folders to hold information and applications that they alone have access to. Users who develop utilities for their own use or purchase private copies of a word processor and spreadsheet, for example, should put them in **~/Apps**.

Net

The **Net** folder gives the user access to file systems that are physically located on remote machines. The immediate folders under **Net** name the machines where the file systems are located. The next level of folders name the root folders of the file systems on those machines. For example, **/Net/willow/misc** is where the **misc** file system located on the **willow** computer would be mounted.

Net contains folders only if the user's computer is set up to be connected to other machines over a network.

All the folders in the root (/) folder, including **Net**, are physically located on the disk that the user's machine was booted from. If a user boots from a local hard disk, for example, **NextLibrary** and all its folders will be stored on the local disk. Remote folders are mounted only under **Net**.

Paths

The Workspace Manager searches for executable files by using a *search path*, an ordered set of folders. The default path used by the Workspace Manager lists these six folders:

~/Apps
/LocalApps
/NextApps
/NextDeveloper/Apps
/NextAdmin
/NextDeveloper/Demos

The Workspace Manager uses this path for three tasks:

- To find the icons it should display for files associated with a particular application.
- To find the application it should start up when the user double-clicks a file.
- To find services offered by applications.

Before using the search path to find which application to start up, the Workspace Manager first looks in the dock. Each application icon in the dock represents a particular application residing in a particular folder on disk. By putting an icon in the dock, the user has indicated a preference for that version of the application over any others.

If an application isn't in the dock, the Workspace Manager looks next in the current working folder, the folder containing the file the user wants to open. Only after failing to find the application there does it turn to the path listed above.

In the path, the Workspace Manager looks first in the **Apps** folder of the current user's home folder. That's where the user's own applications would be. It next looks in the **/LocalApps** folder for sitewide software, then in the **/NextApps**, **/NextDeveloper/Apps**, **/NextAdmin**, and **/NextDeveloper/Demos** folders for NeXT-supplied software. This ordering of folders lets users customize their software to override sitewide software, and lets sitewide software override software supplied by NeXT.

Users can alter the path shown above for the Workspace Manager by setting a value for the ApplicationPaths parameter in their defaults database. You might do this, for example, to add a **/LocalAdmin** or **/LocalDeveloper/Apps** folder to the path.

File Name Extensions

NEXTSTEP applications use file name extensions to identify types of files. The extension consists of the last period in the file name and all characters following it. For example, Mail uses the ª.mboxº file name extension to identify its mailboxes. Typical mailbox names are **Active.mbox** and **Outgoing.mbox**.

The Workspace Manager uses file name extensions not only to identify particular types of files, but also to associate document files with applications. Every application that defines its own data file format appends an identifying extension to the names of its document files.

File Packages

A *file package* is a folder that the Workspace Manager presents to users as if it were a file. Because users normally don't look inside a file package (unless they explicitly open it as a folder), they're not likely to alter or reorganize its contents. ªUsing File Packages,º later in this chapter, describes when you should use file packages.

Using Paths

If your application needs to look up data that could be in several places on the system, it should use an ordered path similar to the one used by the Workspace Manager. For example, if an application requires a particular template file that might be supplied either by the user or by the system administrator, it should search for it first in a folder under **~/Library**, and then in the same folder under **/LocalLibrary**.

Using File Name Extensions

Your application should use its own unique file name extensions to identify (and help the Workspace Manager identify) its documents. Your extension (after its initial period) can include only alphabetic and numeric characters and should be at least three characters long. (NeXT reserves all extensions of under three characters for its own use.) For example, ª.exampleº is an acceptable extension, but ª.egº is not, since it has only two characters after the period.

To request that an extension be registered and reserved for your use, write to:

NeXT Technical Services
Extension Registry
900 Chesapeake Drive
Redwood City, CA 94063

You can also send your request by electronic mail to **ask_next@NeXT.COM** or ...**!next!ask_next**.

The request should include the following information:

- The file name extension you want to register. List a first and a second choice.
- The name of the application the request is for.
- Your name and the name of your company.
- Your postal address and your electronic mail address, if you have one.

· Your telephone number.

You'll be informed when the extension is registered, or if it can't be for any reason. NeXT Technical Services will make the list of registered extensions available to you so that you can choose an extension not already assigned.

Registering a file name extension reserves it for your use. If you fail to register an extension that you intend to use, someone else may register and use it instead.

Using File Packages

An application should create a file package when it has a group of files that it needs to keep together. For example, if your application displays information that's stored in independent text files, or if it makes use of a private utility program, or if it just loads archived objects from Interface Builder ª.nibº files, you may want to keep these auxiliary files in close proximity to the application executable file. A file package (with the ª.appº file name extension) is the way to do it.

Similarly, if your application creates documents that are split into more than one fileÐfor example, if text is in one file and artwork in anotherÐthese files can also be grouped in a file package.

File packages for documents should bear the same extension that's assigned to the application's document files. For example, if a word processor uses a file package to store a document that has artwork, then the file package's extension should be the same as if the document had no artwork and was thus in a single file.

Opening and naming files within a document file package is entirely the application's responsibility.

Creating Unrequested Files and Folders

Applications sometimes need to create files and folders for a user, other than as the result of a Save command.

If the files are associated with a particular document, they should be grouped with that document and placed in a file package.

Otherwise, where the files are located and what they're named depends on whether the user needs to have direct access to them:

· If the user never needs to get at the files or folders independently of using the application that creates them, they should be placed in a folder named **~/.AppInfo**. If an application needs to create many such files, it should put them in its own folder in **~/.AppInfo**. For example, if an application named MyApp needs to create many unrequested files, it should put them in a folder called **~/.AppInfo/MyApp**. If the **~/.AppInfo** folder doesn't already exist, the application should create it.

· If the user might sometimes need to get at the files or folders independently of the application that creates them (for example, template files), they should be placed under **~/Library**. You should name each file or folder so that the user can easily tell which application put it there.

The guiding principle here is that the user's home folder belongs to the user. An application shouldn't leave anything there that belongs to it, not to the user. When the application must create unrequested files and folders, it should put them where the user can find them but where they aren't likely to get in the user's way.

Note: You should generally use the defaults system instead of files to store small amounts of data.

The functions supporting the defaults system are described in the *NEXTSTEP General Reference* manual.

Displaying File Names

File names are often displayed in a browser, where the user can see the path leading to the file. However, sometimes it's necessary to use only text for the file's name and path. In this case, the file's name should usually be displayed followed by two spaces, an em dash, two more spaces, and then the path. For example:

jobRecords Đ /Net/machine/home/records

However, when there isn't much space to display the file nameĐas in a menu commandĐthe path can be shortened to the minimum necessary to differentiate the file name. The part of the path that isn't shown should be replaced with three dots.

For example, if one file called **jobRecords** is listed in limited space, it should be listed as:

jobRecords

If two files called **jobRecords** are listed in limited space, they might be listed as:

jobRecords Đ .../records
jobRecords Đ .../backup