About the Subprocess object and the example application

This file describes the Subprocess object and contains an illustrative/example application. It first provides an overview of the Subprocess object, then describes the classes in the example application, and finally lists the files included in the directory with a description of each.

The Subprocess object

The Subprocess object facilitates the management of concurrent, asynchronous UNIX processes within a NeXTstep application. Methods are provided for the creation, termination and communication with the underlying UNIX process. The UNIX subprocess communicates with its parent NeXTstep application through delegation. Three &Cgate methods provide notification of pending output, termination and errors from the UNIX subprocess. It is the responsibility of the Subprocess instantiator to implement the three delegate methods and decide what to do with the resulting data. In addition to providing a controlled NeXTstep interface to standard UNIX utilities (i.e., `ls`, `find`, `man`, `rdist`), the Subprocess object can also provide, on request, the environment necessary for UNIX processes that require pseudo terminal (or *pty*) support. Some UNIX applications that require pty support include `ftp`, `gdb`, `sh`, `csh`, `kermit`, and `tip`. The Subprocess object permits a NeXTstep application developer to utilize the power of underlying UNIX utilities without having to program the low-level UNIX systems calls that would otherwise be necessary. See the specification sheet (Subprocess.rtf) for more information.

Major classes in the example application

The example application is called ªMyShell.º It is a minimal implementation of a Shell-like program which uses a UNIX Bourne shell subprocess. The main user interface of MyShell is a ScrollView. The user and system share the scrollView for input and output. The user's input is displayed in bold font. The system's output is displayed in normal font. The only special key handled (besides the normal printable ASCII keys) is the `Delete` key. It is the responsibility of the application writer to pass on other shell-useful control key sequences (e.g., escape, interrupt, etc.). Below is a description of the major classes in the MyShell example.

Subprocess	Described above and in the specification sheet (Subprocess.rtf).
CommandScroll	This class manages the MyShell ScrollView by setting the fonts as appropriate for user input and system output, tracking user input and handling the `Delete` key. CommandScroll is a minimal subclass of ScrollView in order to implement a simple shell application.
Coordinator	This class manages the example application by starting the accompanying ª/bin/shº subprocess and acts as the liaison between the subprocess object instance and the user's ScrollView. When the user quits the applic&C n, this subclass terminates the subprocess before the application terminates (using the Application delegate ªappWillTerminate:º).

Other files

Subprocess.rtf	The specification sheet for the Subprocess object.
MyShell.nib	The user-interface of the sample application, which, in addition to the

	requisite main menu, consists of a main window with an instance of a CommandScroll customview (a ScrollView subclass).
InfoPanel.nib	A separate nib file, loaded as needed, containing MyShell's Info Panel.
MyShell.tiff	The TIFF file containing the application icon for MyShell.

MyShell_main.m, Created by Interface Builder.
IB.proj,
Makefile,
MyShell.iconheader