The NeXTSTEP development environment includes a rich library of tools to let you get the most out of NeXTSTEP. At each stage of development—design, implementation, compilation, debugging, and tuning—NeXTSTEP tools help you efficiently create robust applications.

In this section, you'll see the full range of NeXTSTEP development tools. Some of them, like Project Builder, Interface Builder, and Header Viewer, have been presented and demonstrated earlier. Other tools described here fill out the NeXTSTEP development environment to help improve programs and make programmers more productive.

PROJECT BUILDER

Project Builder, introduced and demonstrated earlier, is the hub of NeXTSTEP application development. Project Builder provides features to aid the programming process—from managing the components through building and debugging a working application.

Project Builder keeps track of the files used for building an application—source files, image files, interface files—and even specifies dependencies for the code files so that when you rebuild your project, only the parts that have changed are recompiled.

One key feature of Project Builder is its support for multi-architecture application development. Using Project Builder controls, you can compile and debug applications for your CPU's native architecture. When the development is complete, you then select one or more target architectures and Project Builder will compile a multi-architecture version of your application's executable. (For more on multi-architecture executables, see "The Mach Operating System" at the end of this section.)

INTERFACE BUILDER

As discussed in previous sections, you use Interface Builder to design a graphical interface for your application and make connections between objects that define the overall program structure. Instead of writing code, you manipulate graphic representations of objects just as if you were using a graphics editor to create a drawing. Interface Builder also lets you create custom objects and add them to your application.

HEADER VIEWER

Header Viewer is the NeXTSTEP class browser and documentation tool. While editing source code, you can use Header Viewer to quickly look up classes and methods in either header files or NeXTSTEP documentation. Header Viewer also provides class browsing to help you locate inherited instance variables and methods with ease. In addition to providing access to NeXTSTEP header files and documentation, Header Viewer can also index and access your own application source code.

DBMODELER

DBModeler is the NeXTSTEP Database Kit tool for creating data models. DBModeler uses information in the database—such as the data dictionary—to provide its user with an accurate picture of the data structure. You can then use simple graphical techniques to establish relationships between entities in the database. DBModeler is demonstrated in the "Extending the Advantage" section of this guide.

TEXT EDITORS

Edit, NeXTSTEP's simple yet powerful mouse-based editor, is well-suited for programming. Edit manages logical blocks of source code. For example, you can double-click a brace, bracket, or parenthesis to select the text included between it and its matching symbol. Edit can also collapse and expand logical blocks of code so you can see the overall structure.

In addition to the services that all applications enjoy (such as looking up a class specification in the *NeXTSTEP Developer's Library*), Edit is designed to interact with the GNU Debugger (described below) to let you set break points, run your program, and step through source code files.

In addition to Edit, NeXTSTEP provides GNU Emacs, an extensible terminal-based editor, and vi, the native UNIX screen editor.

COMPILER

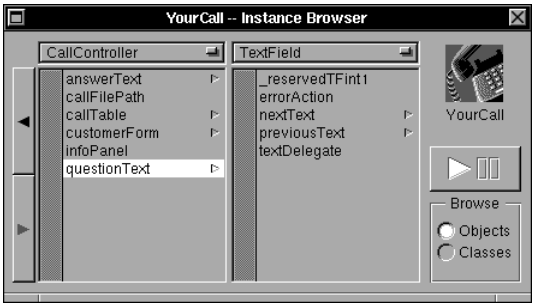
The C compiler used by NeXTSTEP is based on the GNU C compiler from the Free Software Foundation. This compiler accepts a variety of C language dialects, including Objective C, C++, and ANSI C code. The GNU C compiler produces highly optimized object code that is nonetheless compatible with GDB, the debugger described below.

Using this compiler, you have complete access to the Application Kit and other NeXTSTEP kits, even if your program's internal routines are implemented in C++. Unlike most C++ systems, this compiler compiles C++ source directly instead of first translating it into C.

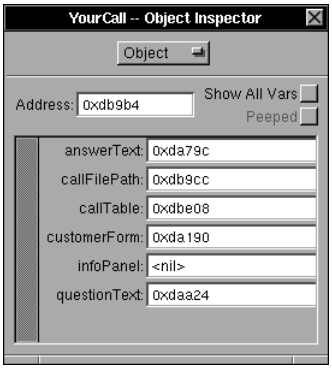
DEBUGGING AND PERFORMANCE TOOLS

NeXTSTEP provides a host of applications and utilities to help you debug your program and improve its performance.

AppInspector



AppInspector is a tool that lets you inspect a running application, examine its objects and variables, and perform run-time tracing of Objective C messages. It also shows information about classes, such as their methods. It can be run standalone or from the debugger. If an object's instance variables are standard types, you can inspect them with AppInspector—and since outlets (references to other objects) are just a kind of instance variable, you can examine the whole network of objects that make up an application.



GDB

GDB, the source- and assembly-level debugger from the Free Software Foundation, has been modified by NeXT to work with the Objective C and C++ languages, as well as with constructs of the Mach operating system. With GDB's object-oriented extensions, you can set breakpoints in methods and send messages to objects in real time. GDB is one of the few debuggers on any platform that supports source-level debugging of optimized object code.

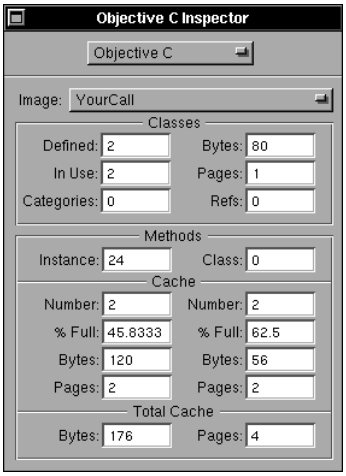
GDB interacts with the Edit application to run and debug your application while accessing and displaying its source files. Edit's GDB panel lets you set break points and simultaneously step through an executable in the debugger and the source code in the editor.

GDB also has features that make it easy to debug programs that weren't started with debugging in mind: GDB can debug core dumps and attach to running processes.

MallocDebug

MallocDebug measures an application's use of dynamic memory. It can measure all allocated memory in an application or just the memory allocated since a given time. MallocDebug also detects incorrectly written memory, so you can tell if you've written data before the beginning or past the end of a block. It also contains a garbage detector that you can use to detect unused memory that can't be reclaimed.

ProcessMonitor



ProcessMonitor displays several types of statistics about a running process or application. You can inspect Mach, Display PostScript, and dynamic memory usage, as well as statistical information about Objective C objects and classes. In addition, processes can be paused or killed with ProcessMonitor.

TOOLS FOR POSTSCRIPT DEVELOPMENT

Since all drawing in NeXTSTEP is done with PostScript operators (supplied by either NeXTSTEP objects or your own code), you need access to PostScript operators from your programs. NeXT provides this access through:

- A library routine for each PostScript operator.
- Functions you create with pswrap to execute your own PostScript routines. The pswrap program generates C functions that can execute any PostScript code. These functions can have arguments that replace variables in the PostScript code.

You also need ways to test your PostScript code. Two programs are specifically designed to give you fast turnaround when testing PostScript drawing:

- Yap, a PostScript previewer that shows PostScript code in one window and the result of its execution in another.
- pft, a shell-based program that you can use to test drawing code. pft establishes a low-level connection to the PostScript interpreter through which you can interactively execute PostScript code and view the results on the screen.

THE MACH OPERATING SYSTEM

The Mach operating system, designed at Carnegie Mellon University, is used as the basic foundation of NeXTSTEP. While NeXT's version of Mach is fully compatible with UNIX 4.3BSD (Berkeley Software Distribution), it was designed with different priorities, including handling multiple-processor and networked machines. For example, in each task (the equivalent of a UNIX process), Mach allows multiple concurrent threads of execution. These threads can be run on one processor or on different processors in the same machine. Mach provides low-overhead interprocess communication with a simple interface that makes it as easy to use between tasks on different computers over a network as for tasks on a single machine.

Mach also supports multiple-architecture executables through its Mach object file structure. Mach object files contain segments, each of which can contain resources required by the executable. This lets a single NeXTSTEP executable file contain the binaries for Motorola, Intel, and any other supported CPU architecture, as well as the text, icon, and other components shared by the binaries. When the user starts a program, NeXTSTEP provides the mechanism for selecting the appropriate binary for the user's CPU.

In addition to UNIX compatibility at the system call level, NeXT supplies the standard UNIX command line tools, such as awk and sed, as well as programming utilities like gprof, the UNIX profiler. You can compile and run UNIX code under NeXTSTEP as easily as you can with any other UNIX 4.3BSD-compliant operating system. NeXTSTEP's operating system also conforms to the IEEE POSIX standard.