Q: What's a shlib, and how does it affect me?

A: A shlib is a shared library. It is a library that contains object code that several executable files may use simultaneously. When a program is linked or compiled with a shared library, the library code that defines the external references are not copied into the object file. Instead it binds the address to a branch table which points to the actual code.

Shared libraries help save disk storage because the a.out files don't contain copies of the archive. It also saves memory at runtime because the a.out files point to one copy of the code. It helps to keep all applications up to date, because when a shared library is replaced, all applications use the new version without needing to be recompiled. Unfortunately, it is harder to maintain a shared library because there are more things that need to remain constant in order for new versions to be compatible.

There are two parts in a shared library, the host library and the runtime library. The host library gets linked in at compile time. It's something like **/usr/lib/libNeXT_s.a**. (Note that there should also be a regular archive library that has all the profiling information in itÐ**/usr/lib/libNeXT_p.a**.)   The runtime or target library contains the branch table and the actual code.   It must be installed on the machine in order for the application to execute.   It's something like **/usr/shlib/libNeXT_s.C.shlib**.   In this particular case,   C is the version letter.

When compiled with **-lNeXT_s**, an a.out contains the name of the file that it needs to use at runtime.   The **otool** program, with the **-L** option, shows which runtime libraries it expects. (See the **otool** man page.)

The first application that touches the shared library pulls it into memory.   All others after that point to that copy in memory.

QA402

Valid for 1.0, 2.0, 3.0