app.tiff ¬ The TickleServices Server Daemon

TickleServices uses a server daemon process to handle services.   A client-server architecture was used for a number of reasons.   Foremost was the consideration that in general usage, services-providers simply do not require a user interface, because the Services menu provides one.   To paraphrase, ``Services providers should be used, not seen.''

Unfortunately, there may come a time when you see side effects from certain properties of the daemon, so I'll try to explain its workings.   In general, the daemon is launched by an entry in the **Workspace LaunchPaths** default.   This default gives a semicolon-separated list of applications which will be launched by the Workspace when you login to the console.   The TickleServices application arranges for the daemon's entry to be added to this list.   If at any time you move or kill the daemon, the simplest means of restarting and/or reinstalling it is to launch the TickleServices application again - it will realize that there's a problem and ask if you want to fix it.

The daemon uses a very specific ordering when loading .ts files.   First the server uses the **ServicesPaths** default to create a list of filenames for .ts files using the following rules:

- If a path starts with ``!'', then any files that exactly match the filename specified after the ``!'' are removed from the list. ``!'' is prepended to a file when the filename is preceded by ``-'' on the Preferences panel. Otherwise:
- If a path ends with ``.ts'', then that path is added to the list of paths. Otherwise:
- If a path specifies a directory, then all .ts files in that directory are added in sorted order to the list of paths.

Once the initial list of paths is generated, the server removes any paths with duplicate basenames (the part after the last ``/''), retaining the first path found for each basename.

The server then loads the services information from each of the files in the list. Each file receives a separate TCL interpreter which is initialized with the file's pathname in the global variable **TSFilename**, after which the file's **!Initialization!** code is executed.

Lastly, the server steps through the files in the list, and generates the list of services entries in sorted order. If there are duplicate entries, the first entry found is the one that is used. The services are checked against the file **~/.NeXT/services/TickleServer**, and if anything has changed, the new version is written in the prescribed format, and NXUpdateDynamicServices() is called to update the services caches.

Once running, the daemon will not reload .ts files unless the user launches TickleServices, saves a file, or selects the **Server/Update** menu entry.   It will then check the services files, and any which have changed since they were loaded are completely reloaded into a new TCL interpreter, with the previous one for that file being discarded.