Customizing Login and Logout

This appendix describes technologies that fill very specific roles. As a rule, if your goal is to have a process running while the user is logged in, you should almost always use either a launch daemon or agent, as described in Creating Launch Daemons and Agents.

Running Agents Before Login

Most software that displays a user interface does not run prior to the user logging in. However, in some rare cases, it may be necessary to create a graphical agent that does.

By default, OS X does not allow any application to draw content prior to login. If you need to do so, your agent must call the `setCanBecomeVisibleWithoutLogin:` method on its windows. For more information, see the documentation for that method and the *PreLoginAgents* sample code.

Authentication Plug-Ins

Authentication plug-ins are the recommended way to perform tasks during the login process. An authentication plug-in executes while the user is logging in, and is guaranteed to complete before the user is allowed to actually interact with their account.

You might write an authentication plug-in if you need to programmatically reset an account to a predetermined state, perform some administrative task such as deleting caches to reduce server utilization, and so on.

To learn more about writing an authentication plug-in, read *Running At Login*.

Login and Logout Scripts

Important: There are numerous reasons to *avoid* using login and logout scripts:

- Login and logout scripts are a deprecated technology. In most cases, you should use `launchd` jobs instead, as described inCreating Launch Daemons and Agents.
- Login and logout scripts are run as root, which presents a security risk.
- Only one of each script can be installed at a time. They are intended for system administrators; application developers should not use them in released software.

One way to run applications at login time is to launch them using a custom shell script. When creating your script file, keep the following in mind:

- The permissions for your script file should include execute privileges for the appropriate users.
- In your script, the variable `$1` returns the short name of the user who is logging in.
- Other login actions wait until your hook finishes executing. Therefore, your script needs to run quickly.

Use the defaults tool to install your login script. Create the script file and put it in a directory that is accessible to all users. In Terminal, use the following command to install the script (where /path/to/script is the full path to your script file):

```
sudo defaults write com.apple.loginwindow LoginHook /path/to/script
```

To remove this hook, delete the property:

```
sudo defaults delete com.apple.loginwindow LoginHook
```

Use the same procedure to add or remove a logout hook, but type LogoutHook instead of LoginHook.

Note: If no plist file exists for com.apple.loginwindow, this method will not work. This file (/var/root/Library/Preferences/com.apple.loginwindow.plist) does not exist on a fresh installation until the user changes a login window setting (such as turning on fast user switching).

If you must install startup scripts programmatically, you should consider providing a copy of this file containing the default configuration options. Then, if the file does not exist, copy that default configuration file into place before running defaults. Again, application developers are strongly discouraged from using login or logout scripts, because only one such script may be installed.

Bootstrap or “mach_init” Daemons

In OS X v10.3, a mechanism similar to launchd was supported to allow the launching of programs either at system startup or on a per-user basis. The process involved placing a specially formatted property list file in either the /etc/mach_init.d or the /etc/mach_init_per_user.d directory. Such daemons also are sometimes referred to as mach_init daemons.

The use of bootstrap daemons is deprecated and should be avoided entirely. Launching of daemons through this process may be removed or eliminated in a future release of OS X.

If you need to launch daemons, use the launchd facility. If you need to launch daemons on versions of OS X that do not support launchd, use a startup item.