

PODCASTING HACKS™

*Tips & Tools for
Blogging Out Loud*



O'REILLY®

Jack D. Herrington

HACK
#54

Build a Simple Sound Cart for Macintosh

Use Xcode on Macintosh to build a sound cart application with AppleScript Studio.

Having sample sounds close at hand is important for podcasters. Using AppleScript Studio, you can quickly build a standalone application with whatever interface you want to control your recording application (e.g., [Audio Hijack Pro \[Hack #50\]](#)) and play your samples at the touch of a button.

First I start with Xcode, which is available to Apple Developer Connection members (online membership is free) from <http://developer.apple.com/tools/download/>, and quite possibly on your Mac's hard drive in `/Applications/Installers/Xcode Tools`. With Xcode, I create a new AppleScript Application project using the New Project command. For this example, I'll use the name `SampleCart` for the project.

Now I want to build the interface, so I double-click the `MainMenu.nib` file, which launches Interface Builder (shown in [Figure 8-14](#)). Using the controls from the Cocoa-Controls palette, I drag three buttons onto the window and change their titles to `Start Recording`, `Stop Recording`, and `Sound 1`, respectively. `Start Recording` will start `Audio Hijack Pro`'s recording, `Stop Recording` will stop it, and `Sound 1` will play a sample sound.

Next I select `Show Info` from the Tools menu. This brings up the information palette, which tells me all about the windows and the buttons. I click the first `Start Record` button and change the info palette, now named `NSButton Info`, to display the AppleScript section. Then I select the clicked item under the Action section and select `SimpleCart.applescript` down in the script section.

This links the clicking of the button to the AppleScript file. I do that with all three buttons. I also give each one a unique name. For this example, I called them `startRecording`, `stopRecording`, and `sound1`, respectively.

By clicking the `Edit Script` button, I am taken back to Xcode to write the script for these buttons.

```
on clicked theObject
    if the name of theObject is "startRecording" then
        tell application "Audio Hijack Pro"
            start hijacking session named "bc"
            start recording session named "bc"
        end tell
    end if
    if the name of theObject is "stopRecording" then
        tell application "Audio Hijack Pro"
            stop recording session named "bc"
            stop hijacking session named "bc"
        end tell
    end if
end on clicked theObject
```

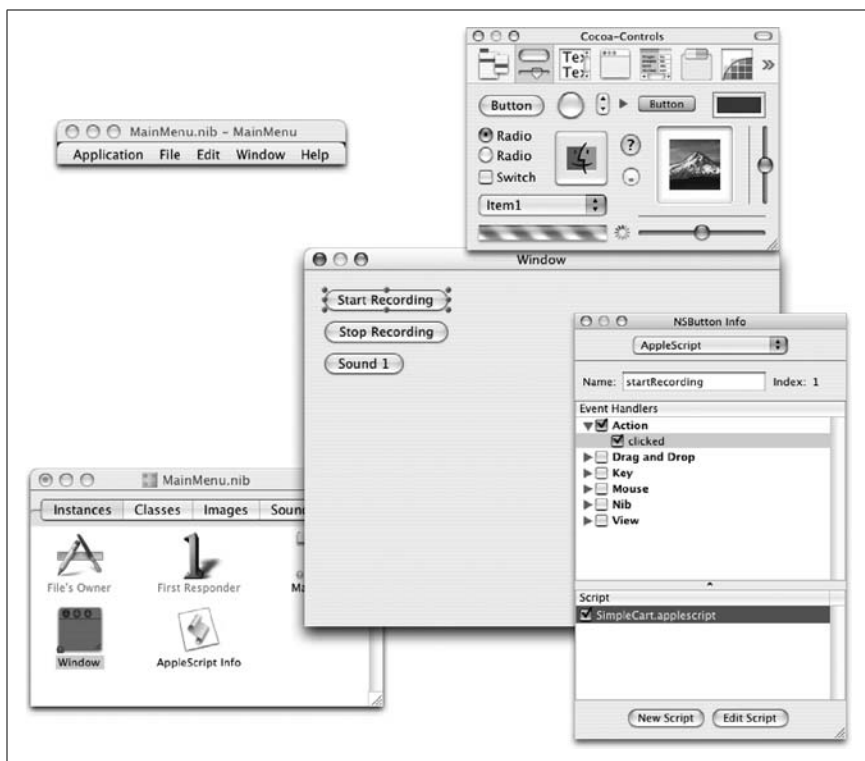


Figure 8-14. Editing the cart application's UI in Interface Builder

```

        end tell
    end if
end clicked

```

The preceding code is the click event handler for the script. It takes as an argument the button that was clicked, called `theObject`. I get the name of the button and then either start or stop recording, depending on what was clicked.

Next I use the Build and Run command under the Build menu to run the application. When it starts, I see Audio Hijack Pro opening up as well. The application knows that I will be using it, so it launches Audio Hijack Pro just in case.

I click the Start and Stop Recording buttons just to make sure they work. If they don't work, I probably have not selected the click event or the AppleScript file in the Info palette. Or the names that I used for the buttons vary from the names I put in the script file.

Build a Simple Sound Cart for Macintosh

If I want to check the name of the button coming into the script file, I put this code at the beginning of the `on clicked` function:

```
display dialog the name of theObject as string
```

That will pop up a dialog with the name of the button before the rest of the script runs.

Now I need to support the Sound 1 button, which means first getting a sound. Really any sound will do. I call mine *testsound.aif*. First I copy the sound file into the project directory. Then I click the SampleCart item in the project and select Add Files... from the Project menu. From here, I select the sound file. When it asks me if I want to add the file to all the projects, I just click OK.

With the sound file in the project, I can add this code to the `on click` handler:

```
if the name of theObject is "sound1" then
    set theSound to load sound "testsound.aif"
    play theSound
end if
```

Wow. That was easy. This loads the sound file from the *resources* folder into a variable called `theSound` and then plays the sound. I run the application again, and lo and behold, clicking the Sound 1 button plays the sound. But there is a slight delay, and that's a problem.

What I would like to do is have the sound file loaded when the program starts up and then just play the copy loaded in memory when I hit the button. So, I need to go back to Interface Builder, click the main window, and select the `awake from nib` event from the Nib section of the AppleScript items in the Info window. Then I assign that to the *SimpleCart.applescript* file and select Edit Script again.

Now I have a new event handler called `awake from nib` where I put this code:

```
on awake from nib theObject
    set theSound to load sound "testsound.aif"
end awake from nib
```

Those with a keen eye will realize that I'm setting a local variable in this function. So, how do I get the variable `theSound` to show up in the click handler? I define it as a property at the top of the file, like this:

```
property theSound : ""
```

Then I change the click handler to play theSound:

```
if the name of theObject is "sound1" then
    play theSound
```

```
end if
```

Now when I run the program again, the sound plays instantly when I hit the button.

The complete script looks like this:

```
property theSound : ""

on clicked theObject
    if the name of theObject is "startRecording" then
        tell application "Audio Hijack Pro"
            start hijacking session named "bc"
            start recording session named "bc"
        end tell
    end if
    if the name of theObject is "stopRecording" then
        tell application "Audio Hijack Pro"
            stop recording session named "bc"
            stop hijacking session named "bc"
        end tell
    end if
    if the name of theObject is "sound1" then
        play theSound
    end if
end clicked

on awake from nib theObject
    set theSound to load sound "testsound.aif"
end awake from nib
```

Going on from Here

This is a simple AppleScript Studio application. AppleScript is a very complete language and the Studio environment allows you to create fully featured and beautiful applications.

One expansion idea is to allow the user to select the sounds at runtime for the various buttons. Another is to allow users to choose which Audio Hijack Pro session to use for recording dynamically.

You can even display HTML show notes in a web browser window that is easy to integrate into your application.

While developing this hack, I took it to the extreme and made myself a nice little AppleScript Cart application. It was too much to document in this hack, but all the code is available on the O'Reilly site in the example code section associated with this book. Then end result looks like [Figure 8-15](#).

The top panel controls Audio Hijack Pro, and you can select which session you want to use from the drop-down list. The bottom section is a list of

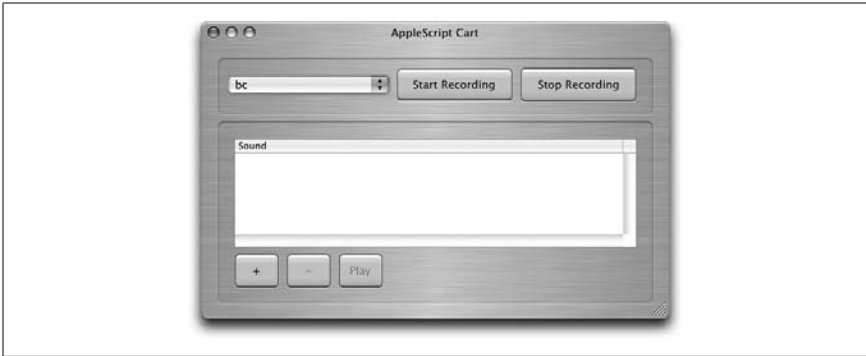


Figure 8-15. The finished AppleScript cart

sounds that you can add to and remove from with the + and - buttons. To play the sound, you either double-click the list entry or press the Play button. The files are selected from anywhere on your machine using the File Open panel.

See Also

- “Build a Simple Sound Cart for Windows” [\[Hack #55\]](#)