AppleScript Working》
6 – AppleScript Studio で log コマンドを使う

AppleScript Studio での Cocoa をベースにした AppleScript のプログラミング環境は、AppleScript のイメージを一新するものと言ってもよいだろう。これまで、スクリプト編集プログラムでのプログラミングとは大きく異なり、ユーザインタフェースの構築ができるのである。しかしながら、スクリプト編集プログラムにあった便利な機能の 1 つである「ログ」の機能は AppleScript Studio では利用できなくなってしまった。ところが、AppleScript Users Group のメーリングリストで、H.Abe 氏が発表したソースを使えば、AppleScript Studio、つまり、Project Builder の上で、log コマンドを使って標準出力への変数の書き出しやイベントの収集ができるのである。H.Abe 氏の一連のソースなどを「HAStudioLogger」と呼ぶことにするが、この利用方法を紹介しよう。

AppleScript でのログ出力

AppleScript には、log というコマンドがシステムレベルでサポートされている。たとえば処理途中での文字列やリストの値を見たいとしたら、log の後に変数名を指定すると、その変数の中身を「ログウインドウ」に表示する。また、やりとりされる AppleEvent の内容についても逐一表示される。従来の AppleScript 開発ソフトにはこうしたログ表示のためのウインドウがあった。いわば、AppleScript プログラミング環境での標準出力のようなものである。デバッガで逐一チェックするのももちろん必要だが、処理を流して結果を見たい場合にはログは非常に便利である。そして、start log と stop log というコマンドも用意されていて、ログの出力の開始や停止もできるようになっている。ところが、AppleScript Studio の開発環境、つまり Project Builder では、AppleScript のログを参照する機能が用意されていない。Mac OS 時代ではサードパーティのツールでもログがあったのだから、もしかすると知られていない API でログをとれるのかもしれないが、H.Abe 氏は独自の方法で、Project Builder でも AppleScript のログを利用できるようにしたのである。

おおまかな動作は次の通りだ。ログを表示する Objective-C のクラスを用意しておくが、Interface Builder でそのクラスのインスタンスを作成するようにしておく。そして、log、start log、stop log のコマンドに対応した AppleEvent が発生すると、自分自身でそのイベントを受け取り、そして中で処理をしてしまうのである。ログの処理は、従来は AppleEvent でシステム任せにできたのであるが、その手法を逆手に取ってのものである。もともと、システムにあったコマンドなので、Mac OS X で作るあらゆる AppleScript のソースにコマンドを書くことができ、規定のイベントが発生する。そこで、作成しているアプリケーションに、そのイベントのハンドラを書いておき、log コマンドを横取りして自分で標準出力に書くとうわけだ。従って、開発している途中でのログの利用はもちろん、生成したアプリケーションでも、Console に log コマンドで出力することができる。

以下のアドレスで、サンプルを含めた Project Builder のプロジェクト一式を入手できる。サンプルのソースで、いきなり試すことができるのであるが、以下、自分で作ったプロジェクトで利用する方法を説明していくことにしよう。

◇HAStudioLogger

http://www.azug.gr.jp/~h-abe/binary/ask/logger.tar.gz

プロジェクトでログ作成をできるようにする

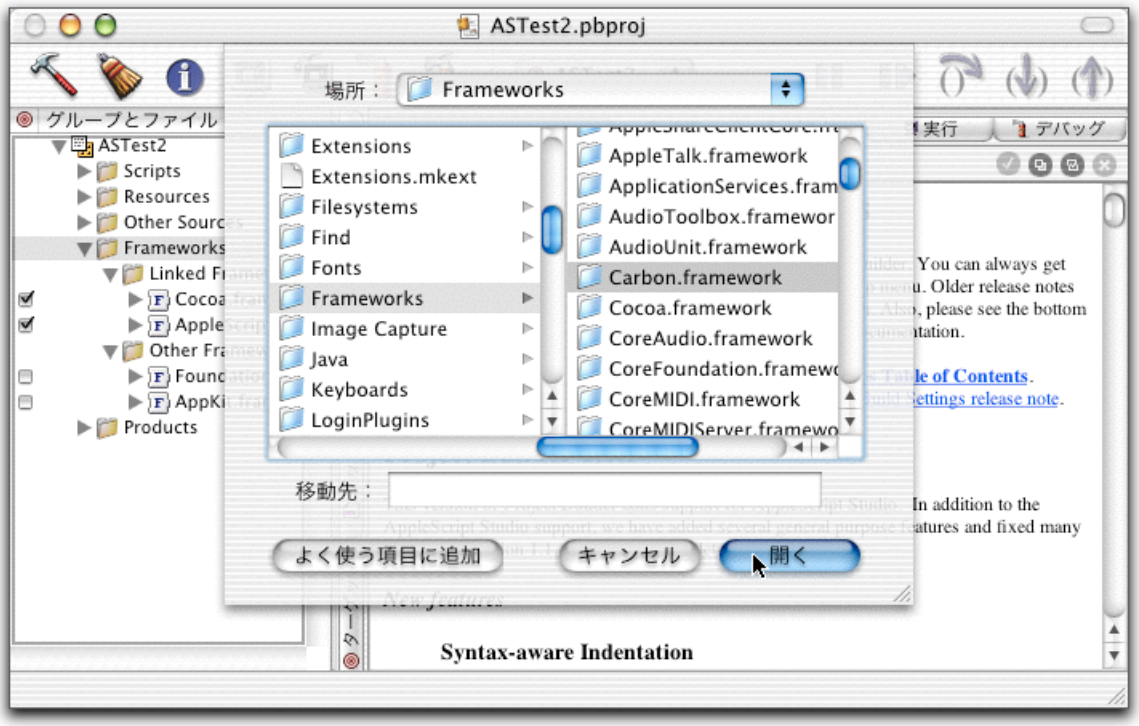
AppleScript Application として作られているプロジェクトがあるとしよう。そのプロジェクトでログを作成できるようにするには以下の 3 つの手続きが必要になる。これらはこの順序で作業をしなければならないわけではなく、必要に応じてやりやすい順序で作業してかまわない。

- Carbon フレームワークを組み込む
- HAStudioLogger クラスのインスタンスを生成する
- HAStudioLogger のソースをプロジェクトに追加する

まず、プロジェクトに Carbon フレームワークを追加する。Project Builder で「プロジェクト」メニューから「フレームワークを追加」（Command+option+F）を選択するが、

その前に、Frameworks のグループを、左側の一覧で選択しておくと良いだろう。メニューを選択すると、システムのフレームワークのフォルダをポイントしているはずなので、そこで、Carbon.framewokrk を選択して、「開く」ボタンをクリックする。

Carbon.framework をプロジェクトに取り込む



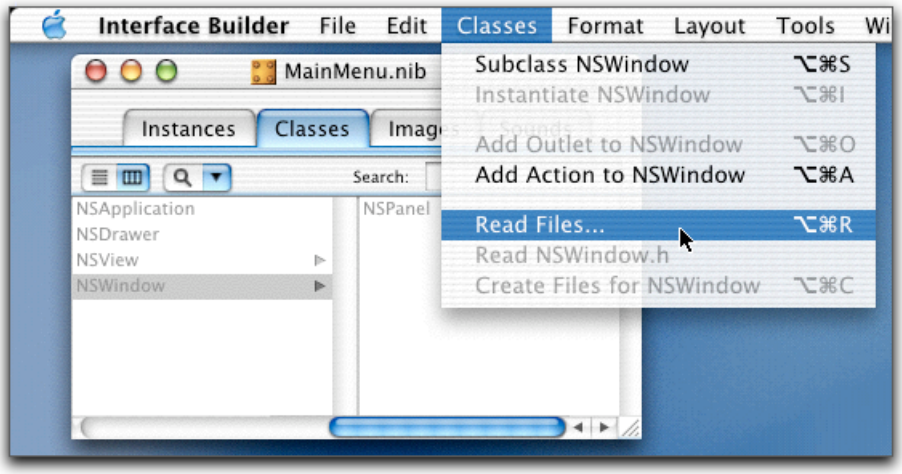
「開く」ボタンをクリックすると、次のように追加するターゲットを指定するシートが表示される。特にターゲットを増やしていないのならその他の設定項目も含めて、このまま「追加」ボタンをクリックすれば良い。

「追加」ボタンをクリックする

これで、Carbon.framework が、Frameworks グループの中に加わったが、もちろん、左側のチェックボックスがオンになっていることを確認しよう。

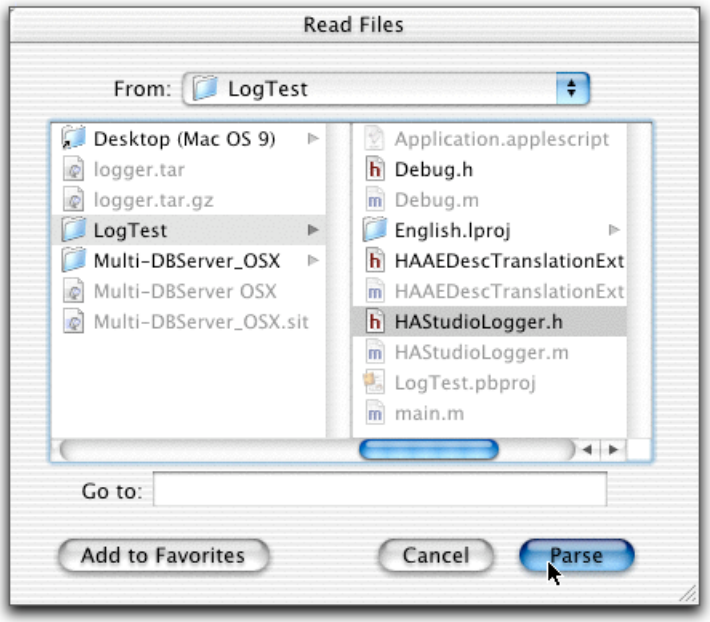
続いて、クラスのインスタンスを作成するが、ここでは、AppleScript Studio のデフォルトの状態であるとすると、最初から組み込まれている nib ファイルの MainMenu.nib ファイルがあるはずだ。これが、起動時にロードされる nib ファイルとして最初から設定されているはずなので、そのファイルを編集することにしよう。Project Builder で MainMenu.nib をダブルクリックして開く。そして、Interface Builder で MainMenu.nib のウインドウをアクティブにして、Classes のタブをクリックして選択する。この状態で、Classes メニューから Read Files（Command+option+R）を選択する。

クラス定義をファイルから読み込む



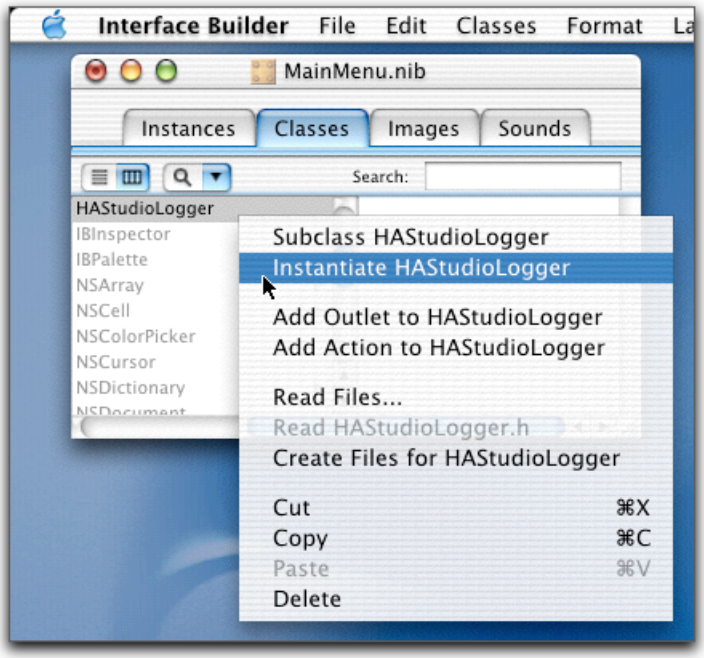
すると、ファイルを選択するダイアログボックスが出てくるので、HAStudioLogger.h というヘッダファイルを指定する。なお、ここでは、ダウンロードしたプロジェクトのフォルダである LogTest にあるものを選択したが、後に示すように結果的にこのソースは自分のプロジェクトにコピーした方がいいので、コピーをしてから、ヘッダファイルの読み込みをした方が、手順的にはスマートかもしれない。

HAStudioLogger.h を指定する



すると、Classes のブラウズリストに、HAStudioLogger というクラスが追加される。このクラスをインスタンス化するため、その HAStduioLogger という名前を、control キーを押しながらクリックし、表示されるメニューで Instantiate HAStudioLogger を選択する。

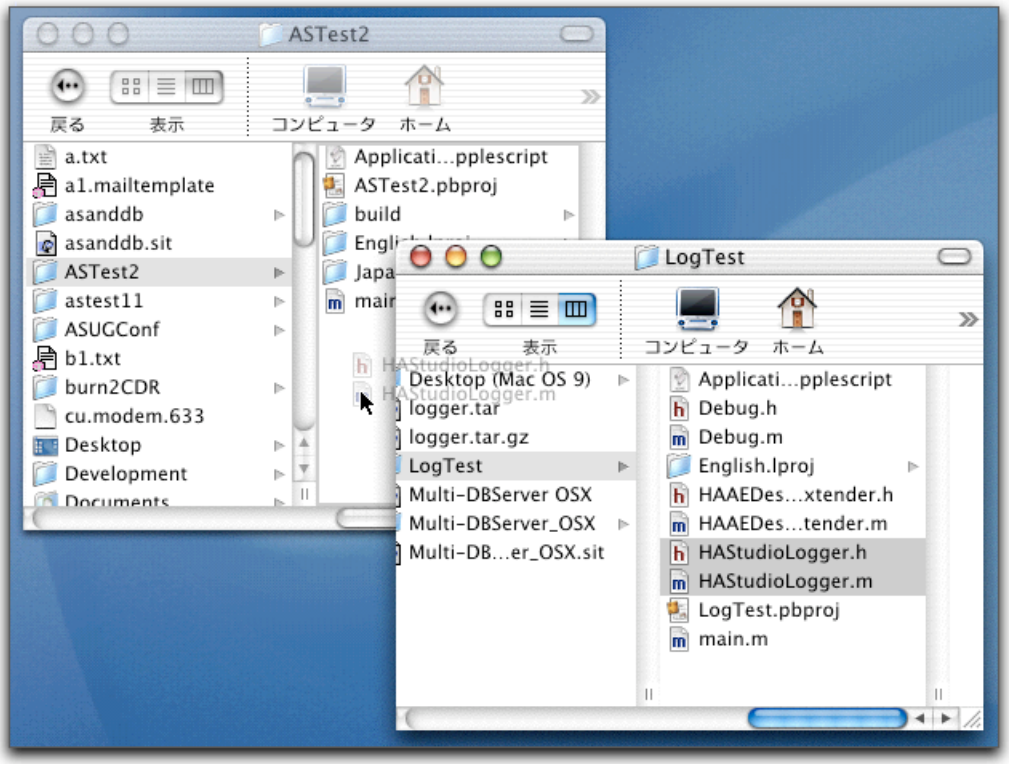
定義されたクラスをインスタンス化する



ここで、MainMenu.nib ファイルのウインドウで、Instances のタブをクリックすると、青い立方体のボックスのアイコンが見えており、HAStduioLogger クラスのインスタンスが nib ファイル内に定義されたことが示されているはずだ。もちろん、ここで nib ファイルは保存しておく。

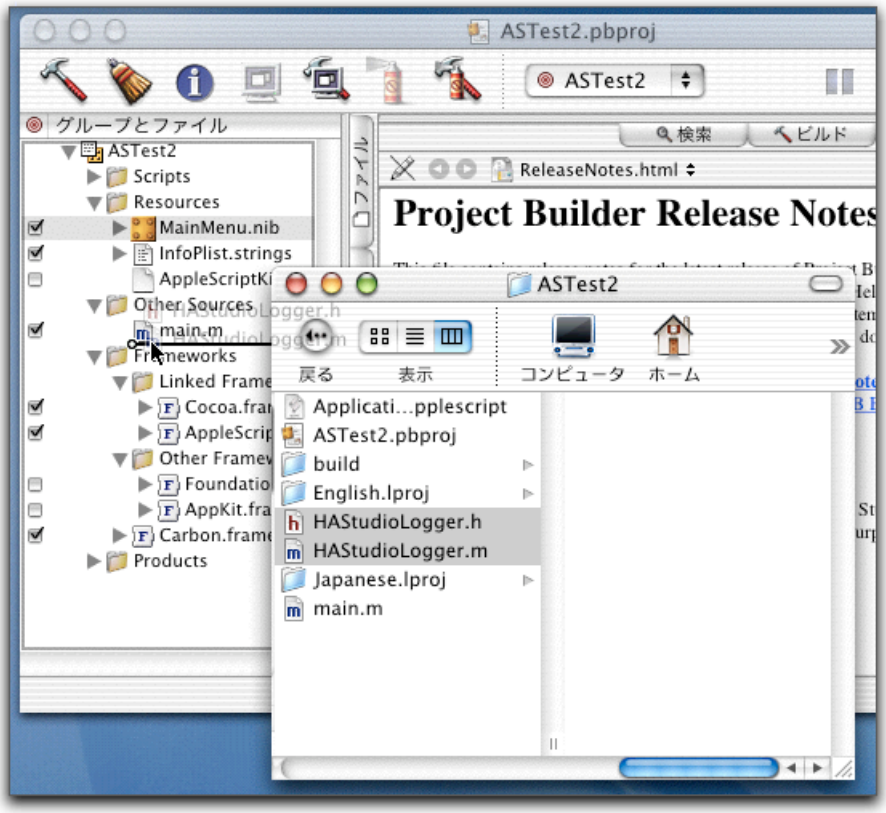
ログを作成するクラスのファイルは、HAStudioLogger.h と HAStudioLogger.m の 2 つであるが、これを、自分のプロジェクトのフォルダにコピーをしておこう。もちろん、Finder で作業を行なうが、適当な方法で作業をすれば良い。

ログ作成を行うクラスのソースを自分のプロジェクトにコピーする



続いて、自分のプロジェクトに、これらのソースファイルを登録する。ここでは、Finder から、プロジェクトの「グループとファイル」のところに直接ドラッグ&ドロップした。場所は別にどこでもかまわないが、たとえば、Other Sources のグループに入れておけばよいだろう。なお、Objective-C のソースとヘッダなので、その他の設定は問題なくなされる。ドラッグ&ドロップした後、Project Builder をアクティブにして追加するターゲットなどをたずねるシートで「追加」ボタンをクリックしなければならない。

ソースとヘッダをプロジェクトに追加する



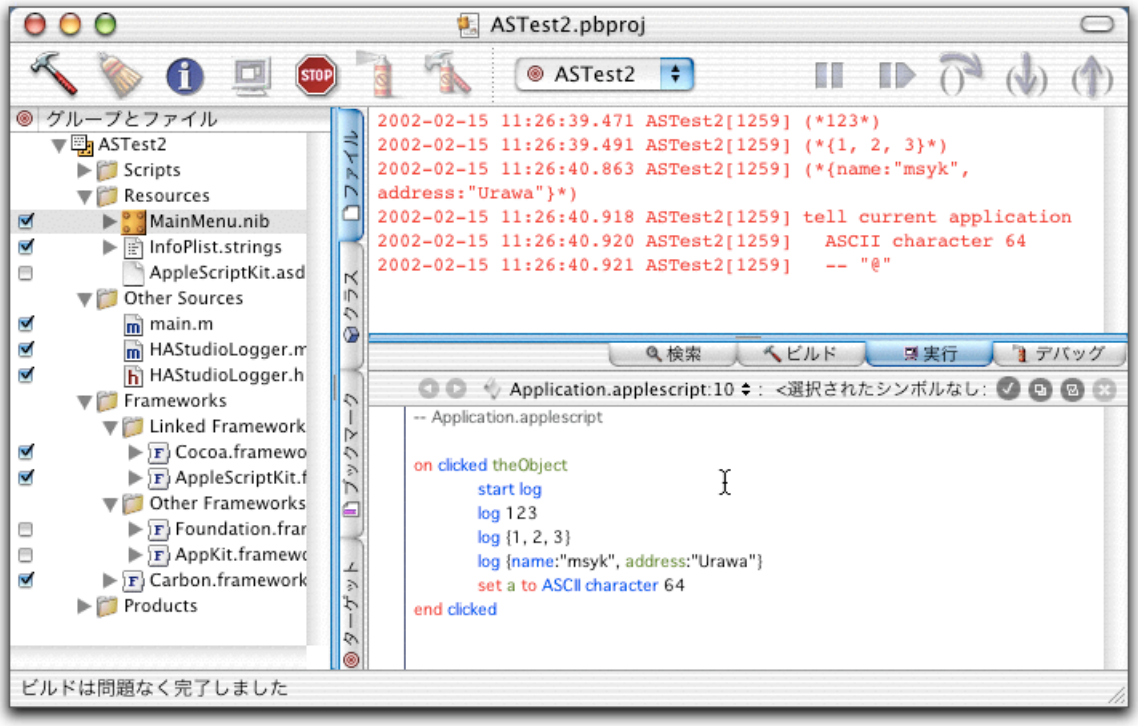
以上で準備は終了だ。

HAStudioLogger の使い方

HAStudioLogger は、これまでのログ関連コマンドと同じように使えばよい。log コマンドの後に、変数名や式などを記述すると、その内容が出力される。数字や文字列はもちろん、リストやレコードでもかまわないし、参照でも基本的にはかまわない。すると、Project Builder から実行したアプリケーションの場合、Project Builder のコンソールに、日付時刻、発生したアプリケーション名とプロセス ID、そして log コマンド以降に記述した式や変数の値が表示される。値はこれまでのログと同様、AppleScript のコメント形式になっている。

一方、AppleScript のプログラムに、start log というコマンドを入れれば、そこからイベントのの収集が始まる。そして、stop log コマンドがあるまで続ける。以下はその実行例だ。プログラムとして示されている部分は、ウインドウにあるボタンをクリックして実行されるプログラムである。

log コマンドの利用例

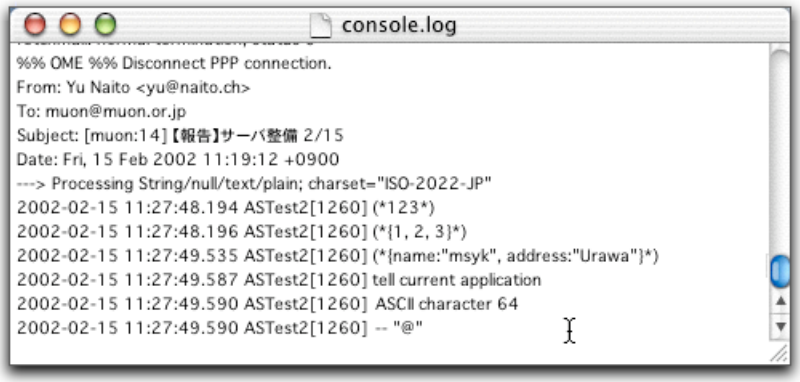


ここにあるように、たとえば、AppleScript で log 123 のようなコマンドがあれば、コンソールに (* 123 *) のように表示される。アプリケーションの処理の途中結果などを、プログラムを流しながら見たいときには、これで log コマンドを使えるというわけである。

一方、start log によってイベントの収集が行われている。ASCII charcter は外部コマンドとして用意されていて、文字コードから文字を生成するものだが、そのイベントが、現在のアプリケーションから発生されたことがコンソールに表示される。コンソールには、AppleScript のプログラム形式で表示されている。

なお、生成したアプリケーションを実行すると、log コマンドの結果や収集したイベントは、Utilities フォルダにある Console で参照できるようになる。つまり、素直に標準出力に出されているというわけである。

ビルドしたアプリケーションでは Console に出力される



HAStduioLogger で行っていることは、ちょっと複雑だが、概要を説明しておこう。まず、このクラスは nib ファイルにインスタンスを作成しているので、nib ファイルをロードしたときに生成される。そのときに AppleEvent のディスパッチを行い、log、start log、stop log コマンドで、HAStudioLogger 内のメソッドが呼び出されるようにしている。実際のログの書き出しは NSLog 関数を使っているが、イベントからの値の取り出しや AppleEvent 関連のさまざまな処理が組み込まれている。

ダウンロードしたプロジェクトには、NSDictionary と AERecord とのコンバートを行うクラスもある、興味のある人には重要な情報源になるかもしれない。

なお、HAStudioLogger は、Interface Builder のツール形式になったものを開発中であるので、近々、バージョンアップがあるかもしれない。その折には、記事でお届けしよう。

（この項、以上）