Getting Started with AppleScript

MacLab Session LB
01/05/2009 – 10:00 AM – 12:00 PM

Speaker:
Ben Waldie
President, Automated Workflows, LLC
http://www.automatedworkflows.com
ben@automatedworkflows.com

Session Description:
This session will provide a brief introduction to AppleScript in Mac OS X. We will explore the Script Editor, and discuss recording and saving scripts. We will then discuss the core AppleScript language, and begin writing scripts. Next, we will discuss scriptable applications, and explore how to read and navigate AppleScript dictionaries. Throughout this session, we will write a number of scripts to perform simple tasks in the Mac OS X Finder, and other applications. Finally, we will discuss next steps and resources for a continued learning of AppleScript.

Table of Contents

NOTES

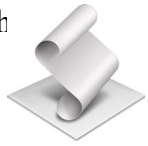
Goals of this Lab

- Provide a brief introduction to AppleScript and its benefits
- Provide a brief introduction to the core fundamentals of the AppleScript language
- Provide a brief introduction to scripting applications in Mac OS X
- Provide a brief introduction to AppleScript Studio
- Learn how to use the Script Editor in Mac OS X
- Learn, through hands-on scripting exercises, how to create simple scripts to automate time consuming and repetitive tasks in the Mac OS X Finder and other applications
- Learn how to save AppleScripts as applications and droplets
- Learn, through hands-on scripting exercises, how to create a simple AppleScript with an interface, using AppleScript Studio
- Next steps and resources for continued learning and usage of AppleScript

NOTES

1. Introduction

What is AppleScript?



AppleScript is a scripting language that is installed with the operating system onto every Macintosh computer. Scripting differs slightly from programming. Programming generally allows you to write a compiled application that directly controls the behavior of a computer. Scripting allows you to write a set of instructions to control existing applications on a computer. On a Macintosh, this includes the operating system.

AppleScripts are written in an editor application, and Apple has included a script editor in the operating system. Simply called "Script Editor," this application can be used by anyone to write, compile, and save AppleScripts.

Not all Macintosh applications are AppleScriptable. However, there are many well-known applications that are scriptable. It is up to software developers to implement scriptability into their products. Some applications provide little or no support for scripting, while others provide a lot of support. In addition, the quality of AppleScript support can vary greatly from application to application, making some applications easy to script, and others more difficult.

Uses for AppleScript

Whether you know it or not, you have probably used AppleScripts in the past. AppleScript is primarily used to automate repetitive tasks that are performed at regular intervals. Many programs such as application installers and email programs use AppleScripts to perform specialized tasks. The following are some tasks that are commonly automated with AppleScript:

- Image manipulation and conversion
- Desktop and database publishing
- Database maintenance
- Server maintenance
- File/Folder maintenance
- CD/DVD duplication

NOTES

Benefits of AppleScript

There are many benefits resulting from the implementation of AppleScripts in daily routines, including:

- Increased Consistency
- Increased Efficiency
- Increased Productivity
- Increased Speed
- Increased Profit
- Reduced Errors

Businesses Using AppleScript

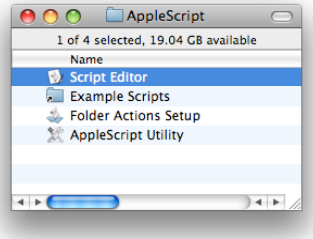
AppleScript is helping companies from across the globe to increase accuracy, consistency, and efficiency, all while reducing mistakes and spending. The bottom line is that companies become more cost effective with the implementation of AppleScript-based workflow automation. Below is a brief list of companies taking advantage of AppleScript-based workflow automation:

- Accuweather
- Adobe Systems, Inc.
- Apple Computer, Inc.
- Associated Press
- Boston Globe
- BMG Direct, Inc.
- Citibank
- Entertainment Weekly
- Epson America
- Fidelity Investments
- Forbes
- Hallmark
- Home Depot
- J.C. Penney
- LA Times
- Lockheed Martin
- Macy's East
- McGraw/Hill Publishing
- Merck & Co., Inc.
- Miami Herald
- Microsoft
- Morgan Stanley
- NASA Langley
- Prentice Hall
- Nikon
- PC World
- Random House
- Readers Digest
- Showtime
- Simon & Schuster
- Smithsonian Institute
- Sony Music
- Staples
- Time Warner
- TV Guide Magazine
- USA Today
- Verizon
- Viacom
- Weather Central

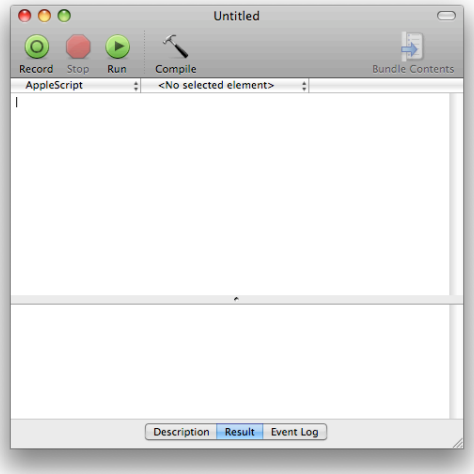
NOTES

2. Script Editor

Script Editor resides in the Applications/AppleScript folder in Mac OS X.



Main Script Editor Window



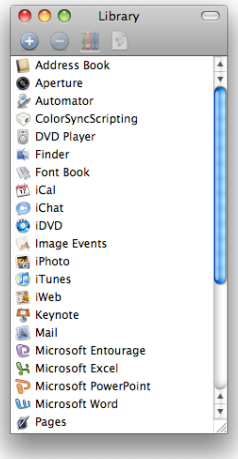
The main script window is comprised of several components:

- Toolbar – Buttons for recording, running, compiling scripts.
- Script editing area – Area for writing the content of your script.
- Description area – Area to include a brief description of your script
- Results area – Area to view the final result when you run your script
- Event log – Area to view all of the events that occur when you run your script

NOTES

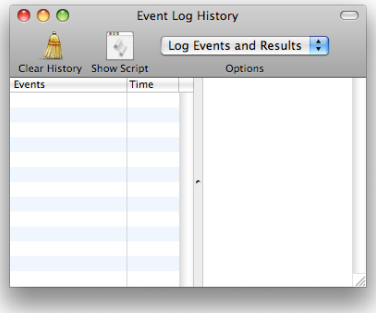
Library Window

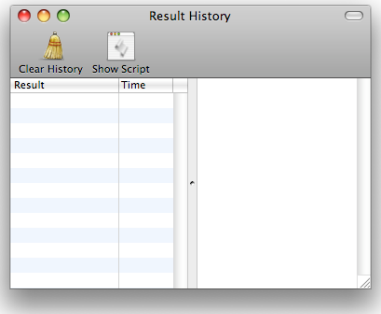
The library window includes a list of frequently targeted applications, and provides quick access to the AppleScript dictionaries of those applications.



Event Log and Result Log Windows

The event log window includes a history of recent event logs, and the result log window provides a history of recent results. Both windows provide a way to revert to previous iterations of a script.





NOTES

3. Recording Scripts

Recording provides a way to capture your manual tasks as AppleScript code. To record manual tasks in an application, the application must be recordable. Unfortunately, not many applications are recordable.

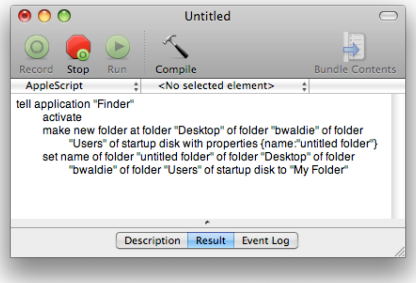
Recordable Applications in Mac OS X

- The Finder
- Fetch
  www.fetchsoftworks.com
- TextWrangler -
  www.barebones.com/products/textwrangler/
- Timbuktu
  www.netopia.com/software/products/tb2/

To determine if an application is recordable, consult the application's documentation, contact the developer, or simply try recording.

Recording Manual Tasks

- Launch Script Editor and create a new script window
- Click the *Record* button in the toolbar
- Navigate to the desired application, and begin performing manual tasks. If the application supports recording, then your manual tasks will appear in Script Editor, as you perform them.



NOTES

4. Saving Scripts

Types of Scripts

- *Compiled Scripts and Compiled Script Bundles* - They can be opened and run in Script Editor. Files and other components can be embedded in compiled script bundles, but not in standard compiled scripts.

- *Script Applications and Script Application Bundles* - They can be run by double clicking on them or dragging and dropping files/folders onto them (if written to allow it – see *Writing Drop Scripts* later in this document). Files and other components can be embedded in script application bundles, but not in standard compiled scripts.

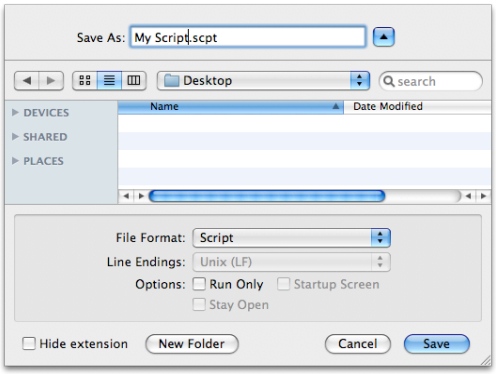
- *Text Scripts* - They can be opened and run in Script Editor, or opened and viewed in any text editor.

Steps to Save a Script

- Choose *Save As* from the *File* menu

- Enter a name for the script

- Choose the desired script type from the *File Format* popup menu in the save panel

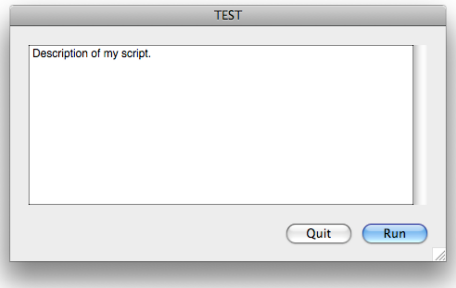


NOTES

Options when Saving Scripts

- Run Only – This option may be specified when saving compiled scripts and script applications. A script saved as run only cannot be re-opened in Script Editor and edited.

- Startup Screen – This option may be specified when saving script applications. If enabled, a startup screen will be displayed when the script is run outside of Script Editor. If a description has been specified for the script, it will appear in the startup screen.



- Stay Open – This option may be specified when saving script applications. If enabled, the script will remain running, once launched. This option is useful for repeating scripts, or scripts that must perform tasks at various intervals.

NOTES

5. Fundamentals of AppleScript

Tell Statements

- Used to target applications or classes within a script
- Formed as follows:

Multi-line Tell Statements

```
tell application "Finder"
    make new folder at desktop with properties
                {name:"XYZ"}
    open folder "XYZ"
end tell

tell application "Finder"
    make new folder at desktop with properties
                {name:"XYZ"}
    tell folder "XYZ"
        set label index to 1
        open
    end tell
end tell
```

Single-line Tell Statements

```
tell application "Finder" to make new folder at desktop
        with properties {name:"XYZ"}
```

Variables

- Used to store a value or result for later use

```
tell application "Finder"
    set theFolder to make new folder at desktop with
            properties {name:"XYZ"}
    open theFolder
    set label index of theFolder to 1
end tell
```

NOTES

If/Then Statements

- Used to analyze a situation, and take a different course of action, based on the desired criteria

Single Criteria If/Then Statement

```
if something = 1 then
    -- Do one thing
end if
```

Dual Criteria If/Then Statement

```
if something = 1 then
    -- Do one thing
else
    -- Do something else
end if
```

Multi Criteria If/Then Statement

```
if something = 1 then
    -- Do one thing
else if something = 2 then
    -- Do something else
else if something = 3 then
    -- Do something else entirely
end if
```

Example:

```
tell application "Finder"
    if (folder "XYZ" exists) = false then
        make new folder at desktop with properties
                    {name:"XYZ"}
    end if
end tell
```

NOTES

Repeat Statements

- Used to perform a series of script steps multiple times, or until a specified criteria has been met

  Infinite Repeat Loop

```
repeat
	-- Do something
end repeat
```

  Finite Repeat Loop

```
repeat 5 times
	-- Do something
end repeat
```

```
tell application "Finder"
	repeat until folder "XYZ" exists
		-- Do something
	end repeat
end tell
```

- Used to loop through a series of items

  Item Looping

```
tell application "Finder"
	set theItems to every folder of desktop
	repeat with anItem in theItems
		-- Do something to anItem
	end repeat
end tell
```

```
tell application "Finder"
	set theItems to every folder of desktop
	repeat with a from 1 to count theItems
		-- Do something to item a of theItems
	end repeat
end tell
```

NOTES

Try Statements

- Used to capture errors when a script runs and, if desired, take a different course of action

```
try
    tell application "Finder"
        make new folder at desktop with properties
            {name:"XYZ"}
    end tell
on error
    -- Do something else here, if desired
end try
```

Handlers

- A portion of code, which exists once, but may be called from multiple locations within a script
- Reduces the amount of code in repetitive scripts
- Allows code to be easily extracted and used in other scripts
- Can accept values as input, and return values upon completion
- Structure of a handler:

```
on handlerName(inputParameter1, inputParameter2)
    -- Do stuff
    return someReturnValue
end handlerName
```

Sample Handler:

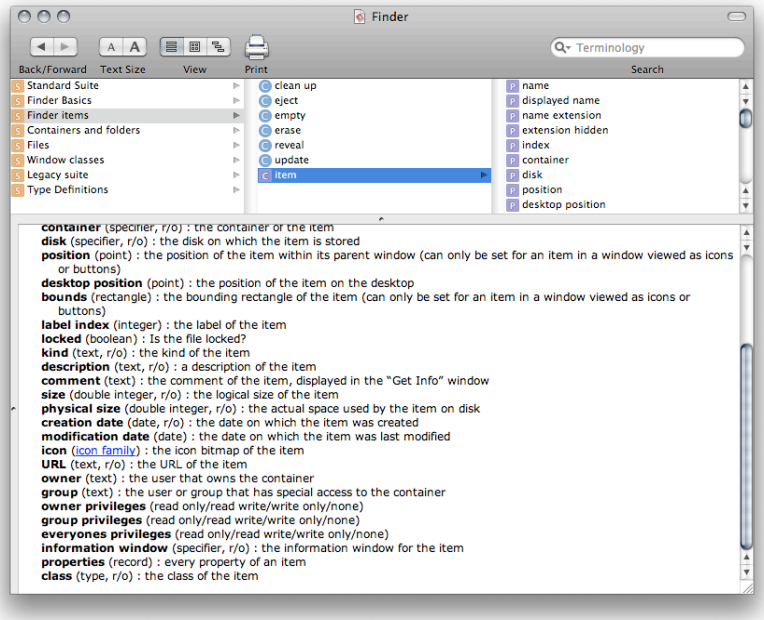
```
set theFolder to makeFolder("XYZ", true)

on makeFolder(theFolderName, openFolder)
    tell application "Finder"
        set theNewFolder to make new folder at
            desktop with properties
            {name:theFolderName}
        if openFolder = true then open theNewFolder
    end tell
    return theNewFolder
end makeFolder
```

NOTES

6. Scriptable Applications

Applications that support AppleScript are considered *scriptable*. Scriptable applications possess a dictionary, which outlines the terminology that the application understands.

Opening AppleScript Dictionaries



An application's AppleScript dictionary may be opened in several ways:

- Drag the application onto the Script Editor application
- Double click on the application in Script Editor's *Library* window
- Select *File* > *Open Dictionary*... from the menu bar in Script Editor

Understanding AppleScript Dictionaries

- *Classes* – Elements within an application, or an application itself, which AppleScript can target. For example, the following elements are all classes in the Finder:
    - The Finder itself
    - Files
    - Folders

NOTES

- *Properties* – Attributes of a class. Some properties are modifiable, and others are read-only. For example, properties of a folder in the Finder would include the following:

    - Name
    - Label index
    - Size
    - Modification Date

- *Commands* – Commands are sent to objects, to instruct them to perform some action. For example, commands that could be sent to a folder in the Finder include:

    - Open
    - Delete
    - Move
    - Close

NOTES

7. Writing Drop Scripts

A script must contain an `open` handler in order to support drag and drop when saved as an application. The open handler supports a parameter, which will pass the dropped items to your script for processing.

Example: Open Multiple Dropped Files in TextEdit

```
on open theDroppedItems
	set theApplicationPath to path to application "TextEdit"
	tell application "Finder"
		repeat with anItem in theDroppedItems
			open anItem using theApplicationPath
		end repeat
	end tell
end open
```

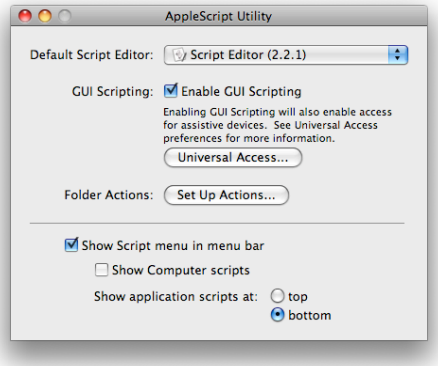
NOTES

8. GUI Scripting

GUI scripting is a method by which an AppleScript can interact with user interface elements in Mac OS X. For example, using GUI scripting, a script can click buttons in a window, select menus and popups, type text, and more. GUI Scripting provides a workaround when non-scriptable situations are encountered.

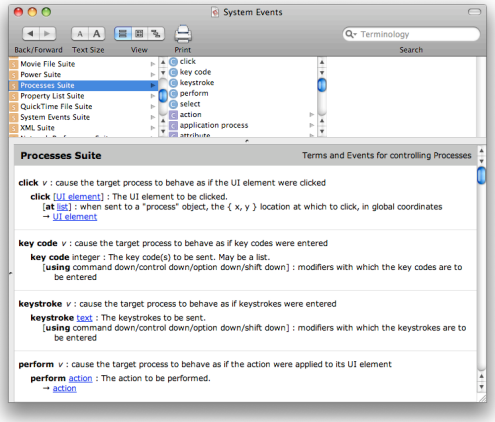
Enabling GUI Scripting

- Launch the *AppleScript Utility* application, located in */Applications/AppleScript*
- Select the *Enable GUI Scripting* checkbox



Accessing GUI Scripting Terminology

GUI scripting terminology is accessed via System Events, a background application in Mac OS X. You can access the AppleScript dictionary for System Events through Script Editor's *Library* window.



NOTES

Example of GUI Scripting

Setting the Appearance System Color to Blue

```
tell application "System Preferences"
	activate
	set current pane to pane "com.apple.preference.general"
end tell

tell application "System Events"
	tell process "System Preferences"
		tell pop up button 2 of window 1
			click
			tell menu 1
				if the name of every menu item
					contains "Blue" then
					click menu item "Blue"
				end if
			end tell
		end tell
	end tell
end tell
```

NOTES

9. Triggering Unix Commands

- Unix commands may be triggered using the `do script` command in AppleScript

  Example 1: Get the current user's name.

```
do shell script "whoami"
```

  Example 2: Retrieve a mini text calendar for a specified date.

```
do shell script "cal 8 2004"
```

  Example 3: Take a screenshot and save it to the Desktop.

```
do shell script "screencapture -xm " & (quoted form of
        (POSIX path of (path to desktop folder as string)
        & "Screenshot.jpg"))
```

  Example 4: Retrieve the source of a URL.

```
do shell script "curl 'http://www.apple.com'"
```

NOTES

10. AppleScript Studio

What is AppleScript Studio?

AppleScript Studio is a feature set of Xcode and Interface Builder, part of the Xcode Developer Tools environment that comes with Mac OS X. Using AppleScript Studio, developers can create feature-rich AppleScript-based applications, complete with user interfaces that have the same look and feel of any other Mac OS X application.

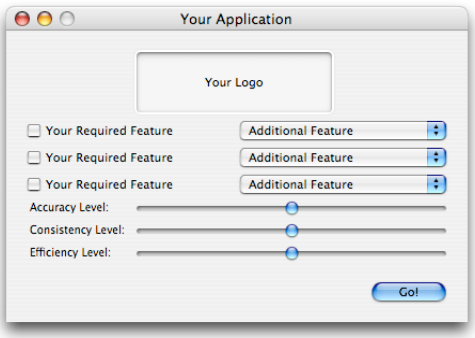

Xcode

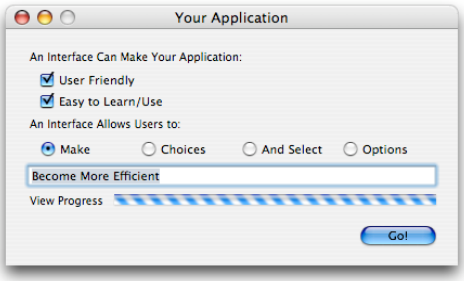

Interface Builder

Benefits of AppleScript Studio

Using AppleScript Studio to construct AppleScript applications gives your applications the following benefits:

- The ability to generate more user friendly applications
- The ability for the users of your script to more easily modify the script's behavior
- The ability to integrate with other languages, such as Java and Objective-C





NOTES

11. Resources for Continued Learning

Mailing Lists and Discussion Forums

- Apple's AppleScript Users Mailing List
  http://lists.apple.com/mailman/listinfo/applescript-users
- Apple's AppleScript Discussion Forum
  http://discussions.apple.com/forum.jspa?forumID=724
- MacScripter's AppleScript BBS
  http://bbs.macscripter.net/

Websites

- Apple's Main AppleScript page
  www.apple.com/applescript
- Apple's Developer AppleScript Documentation
  http://developer.apple.com/referencelibrary/ScriptingAutomation/
- MacScripter
  http://macscripter.net
- Automated Workflows, LLC – Books, Tips, and more
  www.automatedworkflows.com

Books

- AppleScript 1-2-3 (Peachpit Press)
  http://www.peachpit.com/store/product.aspx?isbn=0321149319
- AppleScript: The Definitive Guide
  http://oreilly.com/catalog/9780596102111

Lab Materials

- www.automatedworkflows.com/presentations.html