

# GRAPHIC EXCHANGE

# WAYS

\$7.95 OCT/NOV 1999



05

07447087183 5

# AppleScript

## Is it worth

BY CARSTEN ARNOLD

Imagine arriving at work one morning and finding half your day's work done for you.

Now, imagine this happens once a week on the most boring, repetitive project you work on.

Making that dream a reality may be easier than you think. In fact, Macs already possess the capability to perform this and other ever more amazing feats. All that's needed to coax the genie out of the computer is a little know-how, and an extraordinary amount of patience.

In an effort to familiarize users with the merits of its core software technologies (AppleScript, ColorSync and Quicktime), Apple has presented a number of free seminars in major cities across the country. While these sessions are invaluable in terms of explaining what the various technologies can do, they are remiss in not giving more specific information about how to use these tools, especially Applescript. Perhaps with good reason. Learning anything about Applescript is an exercise in mental fortitude, capable of consuming massive quantities of time.

Applescript (AS) is Apple's system-level programming language. Its true strength is the ability to automate repetitive tasks and customize the user's workflow.

### BEEN THERE, DONE THAT

I first discovered AS five years ago. It received little more than a passing notice as I was unable to find much in the way of documentation explaining what it could do. I hate to admit it, but if this were a Microsoft product you'd be inundated with books on the subject. Several years later, when I

joined the wonderful world of IT at Canada Wide Magazines, the largest independent magazine publisher in Western Canada, I decided to take a closer look.

One of my goals as Canada Wide's systems administrator was to increase the efficiency of the production process. In addition to examining the production department's workflow, I also took note of the way each operator interacted with his or her software and documents. It soon became apparent that one solution would be to reduce the number of repetitive tasks undertaken by the operators.

As luck would have it, the Apple office in Vancouver was hosting an AS seminar, a half-day teaser that introduced attendees to the basics of AS. The registration package included the AS Language Guide (a worthy, if difficult read), the Introduction to Applescript workbook and a CD containing sample scripts. Suddenly, I was hooked.

How naive I was those first few days after the light came on. No sooner had I returned from the seminar than I boldly announced that I would provide a custom-made AS solution for one of our magazines, a weekly television listings publication. In a speech worthy of Captain Kirk, I professed that a process that normally took one person four hours to complete each week would, henceforth, be automated to finish even before the operator came in to work each morning. (Let's do the math: That equals 26 days of saved time in one year. And that was running only one script!)

### AH, THERE'S THE RUB

Meanwhile, back in the real world...

Nine months later, I finally delivered on my promise.

The reason for the delay is obvious to me now. Simply put, AS takes a long time to learn. Many late nights were spent understanding abstract concepts and perfecting an agonizingly few lines of code at a time. Scripts are, thankfully,

# Script biting into?

written in a plain English manner. Almost anyone can read through a script and decipher most of it, but the difficulty lies in “speaking the language” or “constructing the syntax.”

If every single word, statement or reference to an object isn't precisely constructed, the script won't compile. Syntax



(top) **Syntax errors.** If every single word, statement or reference to an object isn't precisely constructed, the script won't compile.

(bottom) **Execution errors.** The wording itself may be correct, but the script isn't smart enough to know when the text isn't understood by the application called upon.

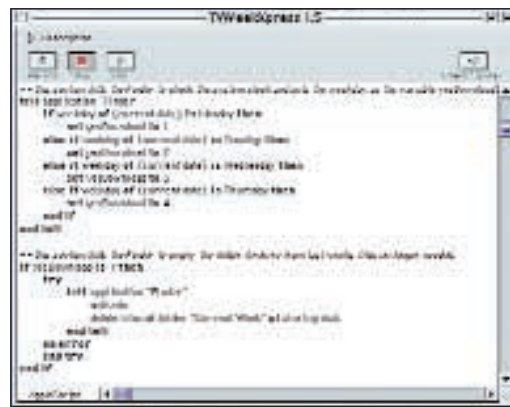
That said, the beauty of AS lies in its ability to let the writer create scripts in stages. For example, my first project required a script that would log onto an FTP site, download a file, then decompress and file it. The concept seemed simple enough, but creating such a script required learning a goodly portion of the entire AS language. That stage took about three months to complete. I'm sure the process would take longer for someone whose job is production, rather than IT oriented.

The snippet of code pictured here (see *TV Week XPress* above right), part of my original script, was written to determine the day of the week and to store the result as a vari-

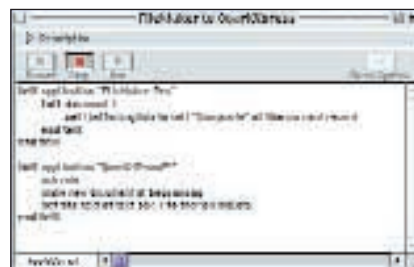
able to be used later in the script. It also empties a destination folder based on that variable prior to storing any new information. The next segment of the script would import the files into *QuarkXPress* templates, save them in various locations, save the text as *Word* documents, and print out the whole thing. This stage was far more time-consuming as it required more advanced programming techniques. Once both pieces were written, it was a simple matter of combining the two and saving the resulting script as an application. Did I say simple?

## EASIER SAID THAN DONE

Becoming proficient in AS is not as easy as Apple and experienced Applescripts would have you believe. Vari-



**TV Week XPress.** This snippet of code was written to determine the day of the week and to store the result as a variable to be used later in the script.



**Filemaker to QuarkXPress.** The AS Extension allows information to be passed between applications. Info from a field in a database program can be passed to a text box in a page layout document.

ables, expressions, if/then routines and repeat loops are but a few of the dozens of concepts you need to understand. As the complexity of the script increases, the logic can become mind boggling. You get really good at talking to yourself.

## HOW DOES IT WORK?

AS is really a collection of components that are called upon when you need them. It consists of the AS Extension, the Script Editor, Scripting Additions, and an applications dictionary of AS commands. Some applications have a rich dictionary that allows almost every function in that program to be controlled; *QuarkXPress* and *InDesign* are two such programs.

The AS Extension is the engine that drives the whole thing and allows information to be passed between applications. For example, it can take the information from a field in a database program and pass it off to a text box in a page layout document (see *Filemaker to QuarkXPress* above).





The Script Editor is where you write this stuff. The simple script below tells *QuarkXPress* to launch, if it isn't already open, and create a new blank document. A command always starts with a 'tell' statement, and has to end with an 'end tell' statement. The Script Editor is also where you can view the scriptable applications dictionary of commands (see below).

The Scripting Additions, which are tucked away in the Extensions folder, contain extra commands that aren't contained in AS or the commands present in the applications themselves.

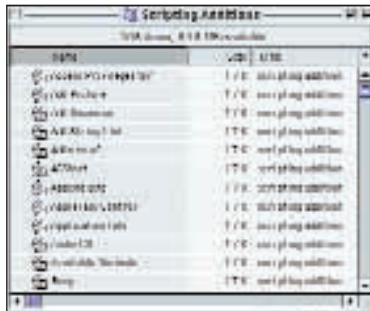
AS, by itself, only knows a small number of commands. Many third-party scripting additions are written because of

**Simple QuarkXPress Script.** This simple script tells QuarkXPress to launch, if it isn't already open, and create a new blank document.



(above) **Finder Dictionary.** The Script Editor lets you view the scriptable applications dictionary of commands.

(right) **Scripting Additions.** Many third-party scripting additions are written because of the shortcomings of the applications dictionary of commands.



**Error Handler.**

This tells the user that they clicked on the Cancel button before selecting a file to open in the Scriptable Text Editor application.



the shortcomings of the applications dictionary of commands. Serious scripters will collect at least a few of these.

One must-have item, *PreFab Player*, enables scripts to do things scripting additions and an application's own scripting dictionary cannot. Its dictionary of commands lets you con-

trol every button or menu item even though the application you are controlling doesn't understand AS. This discovery, late in my AS education, has contributed greatly to my collection of gray hairs.

Understanding these concepts and their interactions is the first hurdle in the AS learning process. This can be quite daunting for the neophyte scripter.

**A FEW BASICS**

Applications are either scriptable, recordable or attachable. Thankfully, the Finder is a highly scriptable application. It is also recordable, a feature not common to all applications. A scriptable program responds directly to AS using its own dictionary of commands.

Recording a script is as easy as clicking the Record button in the Script Editor and performing whatever task you want to automate. Studying the resulting English teaches you how AS prefers to talk to the objects you are controlling. Paragraphs in a word-processing document or cells in a spreadsheet program are examples of objects that are controlled. However, not all applications address the objects contained therein in the same manner. The syntax to address a paragraph of text in *QuarkXPress* for example is different than that in the Scriptable Text Editor, a highly scriptable application included with the Mac OS.

More sobering still is the fact that learning about AS doesn't end with the ability to record a series of events. Other decisions may also need to be taken into account.

It is important to note that recorded script is unable to anticipate what may be needed in the future if circumstances change. For example, it wouldn't take into account the possibility of the user of that script putting the wrong type of information into a dialog box. An effective AS allows for this type of eventuality by giving an alternative. An error handler is the type of "intelligence," which has to be built into scripts and is but one example.

This error handler tells the user that they clicked on the Cancel button before selecting a file to open in the Scriptable Text Editor application (left).

Attachable programs are those that let you write an AS and imbed it within that application. *FileMaker Pro* is a database program that lets you imbed your AS code to supplement its built-in scripting environment.

**WHAT CAN I DO WITH IT?**

With the example above, you could use a *FileMaker* database to track your company's products and have a button that, once activated, launches a built-in AS that exports the records and creates a catalogue in *QuarkXPress*. Watching a script do this is incredible. It will build the entire document for you without you touching your computer. And it does so with amazing speed. This type of script is great for repetitive catalog changes.

When I started scripting, I had only one project in mind: the FTP file transfer, QuarkXPress import text and print project described earlier. It wasn't until I really got into this script that I realized what AS could do in other areas.

The more you script, the more you will discover what you can script. Anything that is repetitive can be automated. Remember, the reason you use AS is to save time and money. The simplest of tasks should not be overlooked. How



**Fetch Download.** When I started scripting, I had only one project in mind: the FTP file transfer, QuarkXPress import text and print project described earlier .



**Make It Illustrator.** For Illustrator eps files that couldn't be opened, I modified a script to force them into Illustrator eps format.

many times do you copy files from a batch of folders to another folder? Write files to PostScript, compress them and send them off to an ftp site? Or receive a file as an attachment to an e-mail, only to discover it can't be opened in the application you think it was created in? The Finder no longer understands what the type and creator codes are for that file. Normally, you would have to open Norton's Disk Editor and manually put them in.

At one time I received so many *Illustrator* EPS files as e-mail attachments that couldn't be opened that I modified a simple script to force them into *Illustrator* EPS format (see *Make It Illustrator* above).

Saving this script as an application allows you to drag and drop an *Illustrator* EPS file with a generic icon and change it to an *Illustrator* icon, making it double clickable once again.

One day you get a Zip disk with hundreds of images on it and you have to convert them all to JPEG format and give them a .jpg extension. If you don't mind opening each one of them manually to do it then we aren't even on the same page here. This is routine stuff for AS.

You can save your scripts as compiled script, an application (which can still be edited), or a run-only application, which cannot be edited. This is great if you want to distribute work without anyone peeking at the scripting code.

By now you get the idea. Once you start using AS to automate your tasks you will find numerous time-saving things

## Where to find out more

Apple has come a long way in the past year by finally giving us the resources that we should have received all along. The OS 8.6 Help menu has an excellent section on AS and is a good place to start. It contains as good an explanation of using the Script Editor, scripting the Finder and Folder Actions as I've seen anywhere. Folder Actions are a great new feature to the Mac OS that allow you to automate interactions with folders.

### AN ESSENTIAL LIST OF RESOURCES INCLUDE:

**The Applescript Language Guide.** Apple's essential guide to the AS language. It is available online at [www.apple.com/applescript](http://www.apple.com/applescript).

**The Applescript User Guide.** This useful guide on how to use AS is also available online at [www.apple.com/applescript](http://www.apple.com/applescript).

Subscribe to the **Applescript Mailing List** at <http://webx.lists.apple.com/webx>. Those having difficulty scripting will find a host of people who offer their assistance.

**PreFab Player.** This extension is de facto standard equipment. A one month full-working trial version is available at [www.prefab.com](http://www.prefab.com).

### OPTIONAL BUT HIGHLY RECOMMENDED:

The scripts you create can be embedded into the application building program, **Facespan**, [www.facespan.com](http://www.facespan.com). This jewel of a program lets you design an elegant Mac-like interface for your scripts.

The script-writing environment of the Script Editor is rather spartan. Other more intuitive script editors do exist, however. One such product is **Scripter**. A trial version is available at [www.mainevent.com](http://www.mainevent.com)

**Danny Goodman's Applescript Handbook.** If you can get a hold of one these great resource books you are way ahead of the game.

**Applescript for the Internet** by Ethan Wilde. A great resource for anyone scripting for the Internet.

Send me an email at [carnold@canadawide.com](mailto:carnold@canadawide.com) with the words "**Applescript Resources List**" in the subject line of your email. An AS I run will auto reply with my complete hyperlink list of resources. The reply also contains a bonus script you can copy and paste into the Script Editor.

you can do with it. Now that I've become proficient with AS, I'm creating new scripts weekly while still perfecting old ones. You get to be a hero real fast writing effective scripts that save your company money. \*

CARSTEN ARNOLD HAS BEEN GIVEN THE ENVIUS TITLE OF 'POSTER BOY FOR APPLESCRIPT' BY APPLE CANADA. HE HOLDS A GRADUATE DEGREE IN TENACITY FROM THE PRESTIGIOUS UNIVERSITY OF INTER-APPLICATION COMMUNICATIONS. IF YOU HAVE ANY QUESTIONS OR COMMENTS ABOUT APPLESCRIPTING, OR IF YOU OR YOUR COMPANY HAS A SPECIAL SITUATION THAT NEEDS A CUSTOM APPLESCRIPT SOLUTION, PLEASE FEEL FREE TO CONTACT CARSTEN AT [CARNOLD@CANADAWIDE.COM](mailto:CARNOLD@CANADAWIDE.COM).