MACTECH®

The Journal of Macintosh Technology

Sampler

Welcome to the MacTech Sampler

As MacTech entered it's third decade in print, we took a top to bottom look at the magazine. As you may have already noticed, we changed its look, gave it a new logo, and more ... but what's the most important thing is that MacTech is now for everyone at the power user level on up ... it's for all of us that are "geeks" or "techs" at heart.

On the following pages, we took a sample of articles we've produced in the last year to give you a good look at what the new MacTech has to offer. There's something for everyone ... network admins to IT Pros, hobbyist to Enterprise, scripters to programmers, web development and open source.

We're thrilled about the new MacTech. But what's even more exciting is what the readers and industry are already telling us. In many people's minds, this is now the magazine that the insiders open up the second it hits their mailbox -- and that's something that we're proud of.

MacTech's new production team, new editorial staff and board, and new focus are here to serve you. Let us know what you think. Drop me a note, or anyone on our team for that matter. While we may not respond to everything, we do read it all.

Enjoy the MacTech Sampler. If you've found just one or two articles that were really useful to you in this sample, imagine what a whole year will bring you. If you aren't currently a subscriber, subscribe today. It's RISK FREE. You can cancel at any time.

http://store.mactech.com/sampler

will get you a special offer.

[illegible]

Neil Ticktin
Publisher
MacTech Magazine

T A B L E   O F   C O N T E N T S

TABLE OF CONTENTS

CREATING A COCOA APPCONTROLLER CLASS

This month, we're going to build a Cocoa app with an interface we designed using Interface Builder. The app will use Cocoa's NSSpeechSynthesizer class to speak a line of text. We'll add a pushbutton to start the speech and another to halt it, even in mid-sentence. The example comes from Chapter 4 of Aaron Hillegass' book, *Cocoa Programming for Mac OS X.* We'll start by taking a look at a class diagramming approach Aaron uses throughout the book.

A Diagram Speaks a Thousand Words

Before we actually start the process of building our project, take a look at the object diagram shown in Figure 1. This diagramming convention was developed by Aaron Hillegas and I find it works quite well at describing the interrelationships between the classes, objects, methods, and instance variables that come together to make your program work.

The class at the heart of this example is the AppController class. Note that this class features two methods: *sayIt:* and *stopIt:*. In Hillegas' drawings, each box represents a class and each arrow connecting two boxes represents the control-dragging connection you create in Interface Builder. For example, in Figure 1, note that 4 of the 5 classes are Cocoa classes (they start with *NS*). All of our code will be built into a new class that we create called *AppController*. We'll create two instances of *NSButton*, one labeled *Say it* and one labeled *Stop*. Each of the buttons will target one of the two AppController methods. When we build the project in Interface Builder, we'll create an instance of AppController, then control-drag from each button to the AppController instance and double-click on the method we want called to finish the connection.

We'll also add instance variables to AppController to keep track of the NSTextField (so we can retrieve the text to say it) and the NSSpeechSynthesizer (so we can send it the text to start speaking and send it a stop message to halt the speaking).

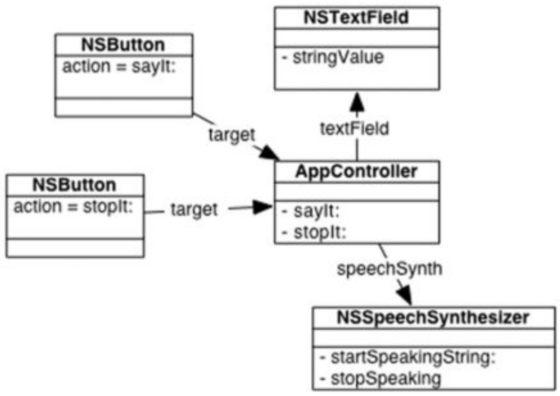


Figure 1. An object diagram for the first incarnation of our SpeakLine program.

In our last Cocoa column, we downloaded the latest and greatest version of Xcode. We created a Foundation Tool, which is an Objective-C program with a console-based interface.

Create the SpeakLine Project

Launch Xcode and create a new project using the *Cocoa Application* template. Name the project *SpeakLine*.

Editing the .nib File

In your SpeakLine project file, in the *Groups & Files* pane, find the file MainMenu.nib and double-click it to launch Interface Builder. You can find the file in the *NIB Files* group as well as under the *SpeakLine* group, in the *Resources* subgroup.

Once Interface Builder launches, click on the third icon from the left in the palette window, then drag an NSTextField from the palette onto the main window. As you can see in Figure 2, the NSTextField is in the upper-left corner of the set of text items.

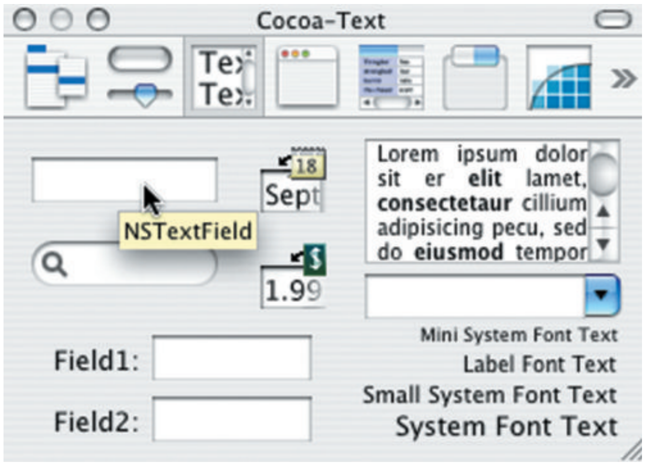


Figure 2. Dragging out an NSTextField.

Drag the NSTextField so it is almost as wide as the window (so the dashed blue line appears when you get about a scrollbar's width from the right side of the window). Double-click on the text field and change its text to read *Peter Piper picked a peck of pickled peppers* or, if you are by yourself, perhaps something a bit more spicy.

Next, click on the second icon from the left at the top of the palette window to show the control palette items. Drag two buttons onto the window, below the NSTextField, with the proper spacing between them and the right side of the window. Label the right button *Say It* and the left button *Stop* (double-click on a button to edit its label).

Finally, resize the window itself, making it as short as possible. Figure 3 shows my Interface Builder session. In this picture, I am dragging the *Stop* button into place. You can see the dashed blue lines showing that the two buttons are aligned with each other and that the *Stop* button is the correct distance from the text field above it and the *Say It* button to its right.

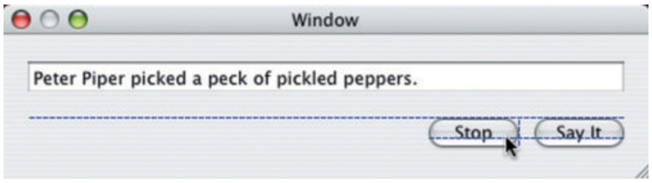


Figure 3. Use the blue dashed lines to line up your buttons and NSTextField.

Create the AppController Class

Now that your interface is laid out, it's time to create the new AppController class.

Click on the *MainMenu.nib* window and click on the *Classes* tab. Scroll all the way to the left and click on the *NSObject* class. With *NSObject* highlighted, select *Subclass NSObject* from the *Classes* menu. Name the new subclass *AppController* (see Figure 4).

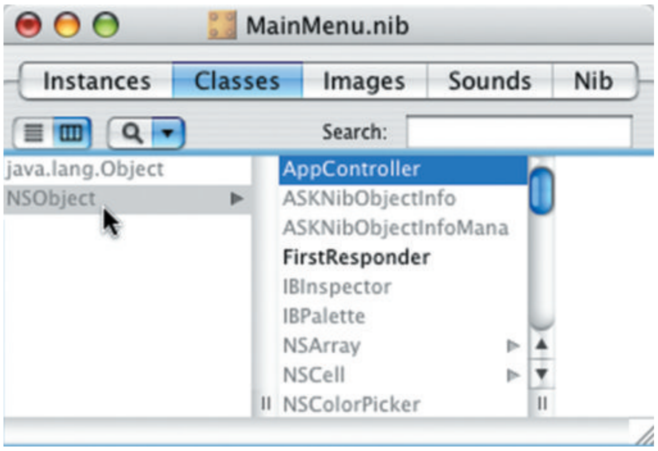
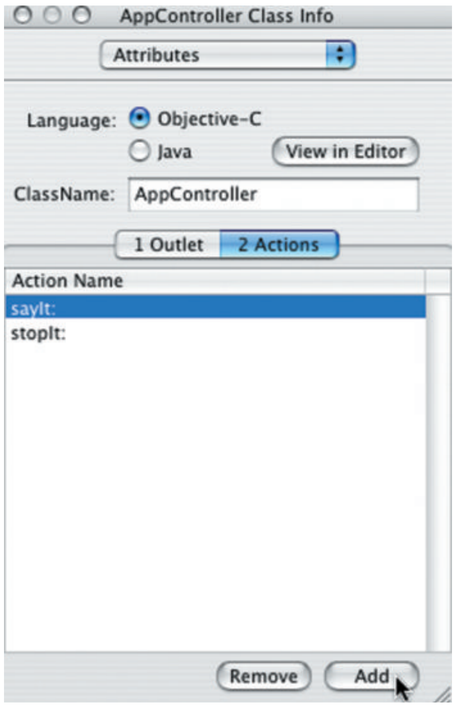


Figure 4. Click on the NSObject class and select Subclass NSObject *from the* Classes *menu.*

Now you'll add two actions (one for each button) and an outlet (an instance variable that points to the text field) to AppController. Open the Info window by selecting *Show Info* from the *Tools* menu. Click on the AppController class in the classes tab in the *MainMenu.nib* window, then click on the Info window and select *Attributes* from the popup near the top of the Info window.

Click on the *Actions* tab, then click on the *Add* button at the bottom right of the Info window. When the new action appears, name it *sayIt:*, then click *Add* and name the second action *stopIt:* (see Figure 5).



Language: (● Objective-C) ○ Java

ClassName: AppController

sayIt:
stopIt:

Figure 5. The Info window, showing the AppController class attributes.

Next, click on the *Outlet* tab and click *Add* to add an outlet named *textField* to AppController. Click in the *Type* column and select *NSTextField* to set the *textField* type to *NSTextField* instead of the generic *id*.

If you look back at Figure 1, you'll see that we've addressed 3 of the 4 arrows in the object diagram. We'll add the missing outlet, *speechSynth*, in code in just a minute.

Be sure that the AppController class is selected in the *Classes* tab and select *Create Files for AppController* from the *Classes* menu. This will generate two source files (*AppController.m* and *AppController.h*) in your Xcode project which we'll edit in a bit.

Next, create an instance of the AppController class by selecting *Instantiate AppController* from the *Classes* menu. Interface Builder will switch the *MainMenu.nib* window to the *Instances* tab and a new, blue cube will appear with the name AppController.

As you can see in Figure 6, the AppController instance is represented by a blue cube. The tiny exclamation point in a circle to the lower right of the blue cube tells you that there is at least one unconnected outlet. Let's take care of that now.

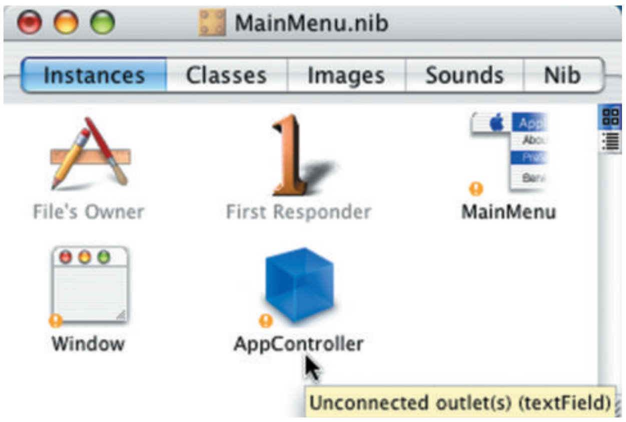


Figure 6. The new instance of AppController with an unconnected outlet.

Making Connections

Before you start making your connections, take a quick peek back at Figure 1. There are four connections that need to be made. Three of them will be made by control-dragging. The fourth (speechSynth) will be made in code.

First, we'll connect AppController's textField outlet so it points to the NSTextField in the main window. Make sure the Info window is open before you start your drag.

Control-drag from the AppController blue cube to the text field in the main window. When you release the mouse button, the Info window should display its *Connections* pane and list the textField outlet. Either double-click on the textField line or make sure it is selected and click the *Connect* button in the lower-right corner of the Info window (Figure 7).

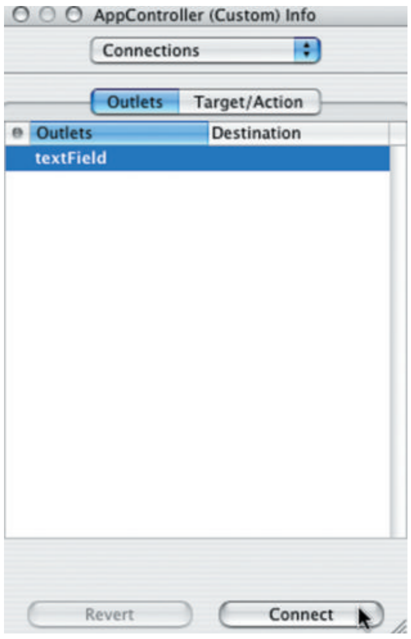


Figure 7. Click the Connect button to connect the AppController to the textField.

Effortlessly edit your PDFs

PDFpen

Now you can fill out and save forms, split, combine, search and even scribble on your PDFs with ease!

"4 mice" – Macworld

$49.95 - download free demo at PDFpen.com
Also available: **PDFpenPro** for creating fillable forms

Creative software for your Mac that does what you want!

Smile on my mac

smileonmymac.com

Next, we'll connect the two buttons to their respective AppController methods. Control-drag from the *Say It* button to the AppController cube then, in the Info window, connect to the *sayIt:* method.

Now control-drag from the *Stop* button to the AppController cube and connect to the *stopIt:* method.

NSWindow's initialFirstResponder

The last bit of Interface Builder work we'll do is to set the NSWindow initialFirstResponder outlet to point to the text field. This tells the window that you want the text field to be active when the window appears so you don't have to click in the text field to start typing. To get a feel for this, try running the program with the initialFirstResponder connected and then with it disconnected to see what happens.

Control-drag from the *Window* icon (to the left of the blue AppController cube) to the text field. In the Info window, click on the initialFirstResponder outlet and click the *Connect* button.

Now let's type in the code!

Enter the AppController Code

Head back over to Xcode and edit the AppController.h file. We'll add the declaration of *speechSynth*:

```
#import <Cocoa/Cocoa.h>

@interface AppController : NSObject
{
    IBOutlet NSTextField *textField;
    NSSpeechSynthesizer *speechSynth;
}
- (IBAction)sayIt:(id)sender;
- (IBAction)stopIt:(id)sender;
@end
```

Next, edit AppController.m to look like this:

```
#import "AppController.h"

@implementation AppController

- (id)init
{
  [super init];

  NSLog( @"init" );

  speechSynth = [[NSSpeechSynthesizer alloc]
initWithVoice:nil];

  return self;
}

- (IBAction)sayIt:(id)sender
{
  NSString *string = [textField stringValue];

  if ( [string length] == 0) {
    return;
  }

  [speechSynth startSpeakingString:string];

  NSLog( @"Have started to say: %@", string );
}

- (IBAction)stopIt:(id)sender
{
  NSLog( @"stopping" );
  [speechSynth stopSpeaking];
}

- (void)dealloc
{
  NSLog( @"dealloc" );
  [speechSynth release];
  [super dealloc];
}

@end
```

Build and run the application. Notice that you can click the *Stop* button to stop the speaking, even in the middle.

Take a look through the code. Most of it should make sense, especially if you've been following along with my previous Cocoa columns.

The *init:* method calls the superclasses' init drops a message to the console, creates an instance of the NSSpeechSynthesizer, then returns a pointer to itself.

sayIt: sends a *stringValue* message to *textField* to retrieve the text, then, if there's at least one character in the field, send it via a *startSpeakingString* message to speechSynth. The string is sent to the console as well, just to help you follow along.

stopIt: sends a message to the console, then sends a *stopSpeaking* message to speechSynth.

dealloc: is called when the AppController object is released. You'll likely never see the console message, since the AppController object was created automatically and is never sent a release message. When it is loaded from the .nib file, the AppController instance has a ref count of one. Not a big deal, but worth noting.

Till Next Month...

One thing that Aaron does in his book is add a color well to the program so the user can choose their own text color. See if you can do this on your own. You'll want to take advantage of the *NSColorWell* class.

Be sure to check out http://spiderworks.com and I'll see you next month…☺

MT

About The Author

Dave Mark is a long-time Mac developer and author and has written a number of books on Macintosh development, including Learn C on the Macintosh, Learn C++ on the Macintosh, and The Macintosh Programming Primer series.

Dave's been busy lately cooking up his next concoction. Want a peek? http://www.spiderworks.com.

Python For AppleScripters

Introduction by Comparison

By Ryan Wilcox

Introduction

Python is easy to use, simple, powerful, and chock-full of great modules (similar to AppleScripts you load via the load script command). The design of the language "just makes sense," the modules are well thought out, and best of all the language has many similarities to AppleScript.

Every day it seems I find more uses for Python than I could have imagined. I use Python along with BBEdit to automate all sorts of common text-based tasks: I have scripts to help me resolve CVS conflicts, to convert decimal to hexadecimal (and back), to encode selected text into URL encoded format and more. Python's readable structure and multitude of included modules lends itself to quick one-off utilities, often with less pain than a similar AppleScript – at least in this author's opinion.

In this article I'll discuss how Python and AppleScript are similar, and how they differ. Then we'll walk through an example script in both languages. Finally, I'll conclude showing how you can use Python and AppleScript simultaneously in your projects.

AppleScript and Python share many similar traits

Looking at a Python script should be a vaguely familiar experience for an AppleScripter – the same indented flow, and understandable syntax. When you are creating a Python script you often try out little chunks of code first, make sure they work, then put them in a larger whole – just like you might when adding code to an AppleScript. Let's tackle these similarities in more detail:

Whitespace matters

In AppleScript whitespace is automatically added by the compiler so that nested commands (such as those in an if, repeat, try, tell, etc) are always indented properly. In Python whitespace is also important – in fact whitespace tells the compiler that a line or block of code is nested. When the indentation stops, the block of code has ended. Contrast this to AppleScript's approach, where blocks of code are ended with end statements. The key difference with Python is that it does not automatically

A Tale of two languages

The advent of OS X brought with it a wealth (some might say invasion) of tools from the Unix world: the command shells, grep, sed, ls, cat, less, vi, emacs – all these utilities and countless others. It also brought with it programming languages – probably the most popular one being Perl. OS X 10.2, however, added to its repertoire a scripting language called Python.

add whitespace as AppleScript does. This isn't as much of an issues as it sounds, as most text editors auto-indent when you type a return. It's worth mentioning here that statements that mark a block of code (such as for loops, if statements, and even functions) require a colon at the end of the "parent" line. The sample script presented later in the article shows several indented blocks of code.

Having whitespace matter is both a good thing and a bad thing. The good news: every Python script you run into will have a similar style, indentation-wise. The bad news: the compiler will complain if you mix spaces and tabs to indent, and it's annoying to have to debug something you can't see. For this reason using a text editor that can Show Invisibles is so very important (maybe even a requirement) while programming in Python. As a sidenote, in cross-platform scripts, using 4 spaces to indent is recommended over using a tab, as spaces are not so easily mangled by unsavvy text editors.

"simple" syntax

Python's syntax is very straightforward, and often compared to pseudo-code. For contrast, look at the sample script later in this article, implemented first in AppleScript, then in Python. The Python version, while it is not as readable as the near-English AppleScript, reads like English plus a bit of 8th grade algebra. AppleScript's approach of English-Like-Syntax-Wherever-Possible often results in extra typing. Compare if you will AppleScript's:

```
  se t end of myList to “the end”
to Python’s:
  myList.append(“the end”).
```

"build as you go"

Both AppleScript and Python make it easy to construct big scripts out of externally created functions. Consider a script that needs to list all the files in a folder. With AppleScript you can create a blank script and start experimenting, and when you have it working incorporate it into a bigger script. Python also has this capability: taking small, building-block pieces and constructing big things with them. Python also has an interactive mode, so you can try something out to see if it works, and when you are satisfied that it works, paste it into your main script.

Python is not AppleScript

For some, the apparent similarity stops there. Python brings its own unique flavor to the language party, differing from AppleScript in some key areas: cross-platformness, case sensitivity, and Python's (significantly) different approach to types are some of what make Python a unique scripting language when compared to AppleScript.

Python is Cross-Platform

Python runs on most major platforms – both flavors of Mac OS, Windows, Linux, Unix, even the PalmOS. Much of Python's functionality knows what platform a script is currently running under, and adjusts platform specific things. For example, the linesep attribute of the os module will return the line separator character(s) for the current platform. There are certain times when you want to use a platform specific API, and that's perfectly acceptable as well. One Python rule of thumb is "We're all consenting adults here," meaning that the language won't try to prevent you from doing something potentially "naughty" if you want to.

Case Matters

In AppleScript, the compiler changes the case of a variable to be the same as the first instance of that variable. So, while case matters, the compiler takes care of it for you. In Python case also matters, except there is no automatic correction – what you type is what you get.

What do you contain? Types Matter

As any experienced scripter knows, AppleScript plays fast-and-loose with type. Sometimes you can't be sure exactly what you will get back. This has its advantages as well as its disadvantages. Take this line of AppleScript for example:

```
set firstNum to "1"
set testVar to firstNum + 1
```

If you know that firstNum can always be converted into number, this works great – it saves everybody some typing. But here's the puzzle: what is testVar? Is it a string? A number? Without a specific declaration, AppleScript will automatically coerce all of the values to the same type, but the question still remains: what type of object will you end up with? (To those of you who answered that the result will be a number, go to the head of the class.) However, as scripts grow in complexity, being explicit regarding what type a variable is becomes essential – you end up almost fighting the implicit coercion you used (and loved) with your smaller script.

With Python, there is no implicit coercion – instead, variables have a very strict sense about what type they are, and what they can do. (For those of you versed in programming terminology, Python is dynamically, but strongly, typed. You can create a variable without caring what type it will be, but Python keeps track of what kind of data that variable currently has in it. Here is that same sample in Python:

```
testVar = int("1") + 1
```

This is how Python does coercion - instead of AppleScript's as xxxxx notation, Python uses xxxx(), as C/C++ does. Trying to run "1" + 1 in python will give a runtime error, as you can not concatenate 'str' and 'int' objects. Python has no idea what to do (it could do two things: cast "1" to an integer, or cast 1 to a string. One answer will result in 2, while the other gives "11"). One of the guidelines (Zens) of Python says: "When faced with ambiguity, resist the temptation to guess." The "Zens of Python" guide both the development of Python as a language and provide a good framework for writing your own scripts and modules. To read more about the culture of Python, and the Zen/Design Principles of Python, visit the following URL: http://www.python.org/dev/culture.html

Batteries Included

Like AppleScript, Python has a small core language, while external modules provide additional functionality. In AppleScript, these external modules come in the form of Scripting Additions and Scriptable Applications (created by Apple and third parties). There are a few Scripting Additions that come preinstalled with every Mac OS installation (Standard Additions, URL Access Scripting, Image Capture Scripting, among others), and several of the apps that come preinstalled are scriptable. All Scriptable Applications and Scripting Additions are written in languages like C/C++ or Objective-C. In Python the focus is not so much on applications as it is on modules – collections of Python routines or objects, usually written in Python, that perform certain tasks. These modules are similar in style to AppleScript's script libraries. While some Python modules include C/C++ code, these seem to be the exception, rather than the rule. Python comes with a huge collection of modules called the Standard Library, so instead of asking the Finder for the size of a file, you would call a function in the Standard Library.

It's in the __doc__s

In AppleScript, there is always some human readable documentation: the dictionary of the application or scripting addition. Sometimes the dictionary is not enough but it is always there, on your machine. When I am writing AppleScript, I always have at least one or two dictionaries open, referring to them as I write my script, like a cheat-sheet right there on my desktop.

Python, on the other hand, takes more of a "reference book" approach to documentation – it is available in a number of different formats, (downloadable from http://python.org), but like any reference book, you hope the documentation is up to date, complete, and that it describes the method you want to use. There have been several utilities written to reduce the risk of these mistakes in the documentation happening, and the Python documentation is usually of high quality. Still, the possibility of out of date documentation exists. The Mac Python IDE includes a module browser, letting you explore different modules like you do an AppleScript dictionary, but it's often not as helpful. As mentioned before, most Python modules are coded in Python itself, and you can usually view the source code for a module, trying to figure out what a function actually does.

The standard Python practice is to add a string literal describing the function and parameters it takes as the first line of the function. This string is called a "docstring." If you view a module in the Mac Python IDE's Module Browser, this string will be described as __doc__ (pronounced "under under doc under under") – however this __doc__ string is what is rendered for the documentation – meaning that if the documentation is poor, the __doc__ will probably be as well.

Here's an example of a function with a docstring. But first it is also important to note Python's string literal functionality. If you have a character in a string literal that you would ordinarily have to escape, for example a quote character, you can instead triple-quote the string literal – the string is considered everything enclosed in triple quotes (""”I'm in triple quotes""", for example, is a perfectly valid string literal.)

```
  def addValues(value1, value2):
    """addValues adds two numbers. Simple. value1 is
the first value to add, value2 is the second. Returns
these two values added together"""
    return (value1 + value2)
```

This standard practice is a great practice to adopt for your own methods. Adopting this documentation convention will help you remember what a function does, why you need it, and what the parameters do when you revisit the function at a later date. AppleScript is without such a standard practice; everybody has their own styles of documenting an AppleScript method, if they do it at all.

An IDE and an example: Kicking the tires

Python makes a great multi-purpose language. Internally we use it from everything from creating shell programs, to making BBEdit Unix filters, creating throw-away one-time scripts, or designing custom CGI scripts for our clients. You can even use Python in conjunction with Apple's Cocoa application framework using PyObjC. With some additional modules, you can use Python just like you would AppleScript – to display simple GUIs, talk to other applications, and do other user administration tasks.

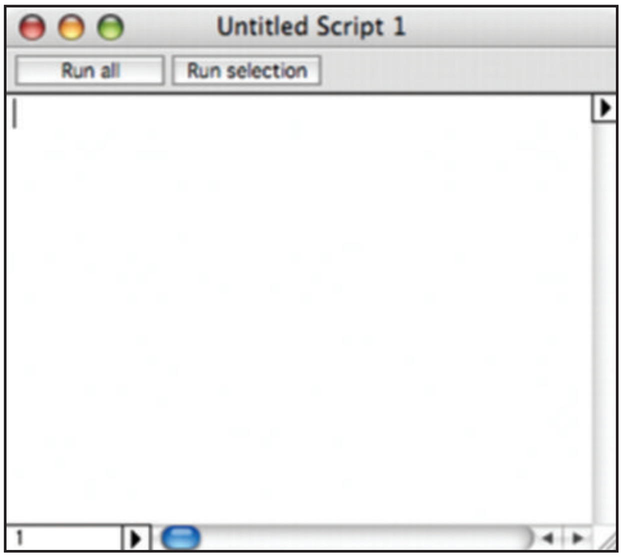
Starting at the beginning: Installing a GUI friendly Python

While you can use Python on the command line, the command line program gives you everything you would expect from a Unix based tool: no GUI capabilities, no IDE and no graphical debugger. In short, it's not the best environment for Mac people who are used to such niceties.

In the pre-OS X days, a Mac OS 9 version of Python, including an IDE, was provided by Jack Jansen. The IDE and all the Mac specific modules from those days still work under OS X, but their appearance has not been updated for OS X. Those looking for prettier IDEs on OS X shouldn't fret – there are several that show promise, but as of this writing most are still in the early stages of development.

You can download the MacPython package at
http://homepages.cwi.nl/~jack/macpython/

This package will install the PythonIDE application (found in your Applications/MacPython-2.3 folder) along with some other things. Double click on the Python IDE and you should get something similar to this:



Got it? Does it look something like this? Good. Let's go to work.

A simple illustration, line by line

Let's start things off with a simple example – a script that accepts user input and appends it to a file. It should be noted here that simple AppleScript display dialog like interfaces aren't Python's strong suit. While the MacPython package helps, it's still not as easy as AppleScript's display dialog. This (and inter-application communication) are two of the things that Python does poorly, however there are two packages currently competing to become the de facto standard for inter-application communication in Python, so the tide (at least on that front) should turn rather quickly.

First, the AppleScript:

```
set filepath to choose file with prompt "select a file
to append to"
set fileRef to open for access filepath with write
permission
```

```
repeat
  set dialogResult to display dialog “enter a line”
default answer “line” buttons ¬
    {“No More”, “Enter”} default button 2

  if button returned of dialogResult is “Enter” then
    set textReturned to text returned of dialogResult
    write textReturned & return to fileRef
  else
    exit repeat
  end if
end repeat
close access fileRef
Now, the Python:
import EasyDialogs, os

filepath = EasyDialogs.AskFileForOpen(“select a file to
append to”)

if filepath:
  fileRef = open(filepath, ‘w’)
  while True:
    textReturned = EasyDialogs.AskString(
      prompt = “enter a line”, default = “line”,
      ok=”Enter”, cancel = “No more”)
    if textReturned:
      fileRef.write(textReturned + os.linesep)
    else:
      break
  fileRef.close()
```

Let’s take the Python sample line by line:

```
import EasyDialogs, os
```

As mentioned before, Python organizes sets of functionality into modules. Import loads these modules into your script. Here we import both the EasyDialogs module (a Mac specific module) and the cross-platform os module.

```
  filepath = EasyDialogs.AskFileForOpen(“file to
append
  to please”)
```

This line calls the AskFileForOpen method in the EasyDialogs module, which will ask the user to select a file. By comparison, AppleScript searches all of the installed scripting additions for you, looking for the command, and sometimes it “helpfully” finds the wrong one. This is what often causes a terminology conflict. If AppleScript required you to specify where to get the terminology from, you might have to write something like set filepath to standard addition’s choose file which may be more typing, but would remove any potential ambiguity. Sadly, AppleScript does not support this style of reference.

```
  if filepath:
```

In AppleScript, if the user presses cancel in a choose file dialog, AppleScript raises an error and terminates the script (unless you handle the error in an on error block). Python’s AskFileForOpen function does no such thing – it just returns None and keeps on executing the script. We must explicitly test the value of filepath for its existence (filepath would be None if the user pressed the “Cancel” button on the dialog). In Python variables that are None are simply considered false. Truth in Python is a tricky thing, but best explained by the following web page:

http://www.users.csbsju.edu/~clusena/python/fundamentals/node10.html.

```
  fileRef = open(filepath, ‘w’)
```

Again, similar looking to the AppleScript – open the file at filepath with write permissions.

```
  while True:
```

Here the aforementioned Zen of Python “when faced with ambiguity, resist the temptation to guess” returns. The above line shows how deeply this statement is ingrained in the Python culture. The equivalent AppleScript statement is just “repeat” – to which Pythonistas would ask “repeat what?”. Here Python explicitly says “do the following as long as this statement is true”. The True must be capitalized – True means true, while true means nothing. Got it? Good.

```
  textReturned = EasyDialogs.AskString(
    prompt = “enter a line”, default = “line”,
    ok=”Enter”, cancel = “No more”)
```

By reading the documentation I found this method, and figured out what parameters to pass to it. These parameters are self-explanatory, but it did take a bit of hunting in the documentation (and maybe even a read of the source) to learn exactly how to construct this line.

```
  if textReturned:
```

Here again we test the value of textReturned – if it contains anything, the if executes. Same as the if filepath line above. It is worth repeating that lines that begin blocks of indented code, such as this line, need a colon at the end.

```
    fileRef.write(textReturned + os.linesep)
```

Here we write the text the user entered, and a line separator (of whatever platform we’re on) to the file. As mentioned before, os.linesep will return the end-of-line character(s) for whatever platform the script is on.

```
  else:
    break
```

Here we come to the end of the if textReturned block. If textReturned is None, as belabored in more detail above, the user pressed the cancel button – we should abort our while loop.

```
  fileRef.close()
```

Always close our file – in this case, by calling fileRef object’s close() method. Note the indentation level of this line – it is on the same level indentation wise, as the while statement. This signals the

Multiple Formats. Multiple Platforms. Complex Installers.

We have the solution:

StuffIt Engine SDK

Solving the compression & multi-platform puzzle!

www.stuffit.com/sdk/

Licenses start as low as $99/Yr.

Put the power of StuffIt to work for you.

- Adds value to your applications by integrating compression and encryption tools.
- The only tool that supports the StuffIt file format.
- Make self extracting archives for Macintosh or Windows
- Available for Macintosh, Windows, Linux or Solaris

Mac

solaris

StuffIt InstallerMaker

An OS X native version ready for developers!

www.stuffit.com/installermaker/

Prices start at $250

Give your software a solid beginning

- Create Macintosh OS X and Macintosh Classic compatible installers
- Get all the tools you need to install, uninstall or update your software in one complete package
- Add muscle to your installers by customizing your electric registration form to include surveys and special offers.

Allume Systems

www.allume.com

email: dev.sales@allume.com

end of the while loop – the indentation level changed. While this was mentioned previously in the article, in the "whitespace matters" section, it deserves repeating here.

Two Worlds Collide: AppleScript, meet Python

Even if you don't want to use Python as your main scripting language, you can slowly move parts of your AppleScripts into Python – for instance having your Python scripts do things that are hard to do (or slow to do) in AppleScript, but easy in Python. Here's an example that will find a string inside a string (or return 0 if it does not). This task is easy to do in AppleScript (using the offset of functionality), but it can be very slow. Instead of using offset of we use a Python script to do it.

Python script: substr.py:

```
#!/usr/bin/env python
#first line tells us where to find python.

#a # character means the rest of the
#line is a comment, just like AppleScript's –

import sys

findWord = sys.argv[1] #get the first command line
argument
thestring = sys.argv[2] #get the string

print thestring.find(findWord) + 1
#AppleScript strings start at 1, python's @ 0. Adjust
the answer for AS.
```

Create the above Python script your favorite text editor, and save it. Make sure the line endings are set to Unix line endings, just to be safe.

Now, create the following AppleScript, and save it in the same folder as the above Python script, in Application format.

```
on run
  display dialog ( "world has been found at character: "
    & pythonSubStr("world", "hello world") )
end run

to pythonSubStr(toFind, theString)
  set myContainer to getContainerofMe()

  set myResult to do shell script "python " &
myContainer
    & "substr.py " & " \"" & toFind & "\"" & " \"" &
    theString & "\""

  –tell python what script to open up, and what params
to pass
  –also note that the quotes we put around both strings
are to      prevent the shell from
  –breaking them into lots of different arguments (the
shell     sees a space
  – as an argument separator)
  – this is usually not what we want to do. These will
be      removed
  –automatically by Python.

  return myResult
end pythonSubStr
```

```
on getContainerofMe()
  tell application "Finder"
    set dest to path to me
    set temp_container to container of dest as alias
    return (quoted form of POSIX path of temp_container)
 –POSIX = unix path
  end tell
end getContainerofMe
```

However, it's worth noting here that do shell script on my test machine (400Mhz Powerbook G4) takes about .5 seconds to execute. This is not because Python is slow, but rather do shell script can take a while to do its initialization and termination routines. This slowness, however, may just beat out a vanilla AppleScript using offset of, depending on the data.

Using Python, you can sometimes build functionality into your scripts that normally would require third party OSAXen in AppleScript. Complex string manipulations, regular expressions, even sending email. Using do shell script to merge AppleScript and Python code might just provide that extra oomph for your script, or may just speed up your development process.

Conclusion

With it's familiar-feeling language, cross-platform abilities, large standard library, and simple, readable syntax, you might find Python an interesting choice for your next project – even if it's only a part of it. Feel free to experiment with the built-in Python interpreter. Fire up Terminal.app and enter the command python to be taken into the command line Python's interactive mode (Control-D to get out). For those of you of the GUI persuasion, see the Python Interactive window in the Python IDE. Learn more about Python by visiting the Python website at http://www.python.org, in particular the Introduction section (http://www.python.org/doc/Intros.html).

References

For additional information on Python, see http://www.python.org. For additional information on using Python on the Mac, see http://www.pythonmac.org/. Thanks go to Matthew Strange and Jared Barden for reviewing this article.

MT

About The Author

Ryan Wilcox is the founder of Wilcox Development Solutions (www.wilcoxd.com) specializing in carbonization, cross-platform application development and e-commerce solutions. He often has a hard time thinking of witty things to say in these blurbs. You can reach him at rwilcox@wilcoxd.com.

Focus Review

By Paul Pharr

PERFORCE

POWERFUL VERSION CONTROL FOR THE MAC (AND ALL THOSE OTHER PLATFORMS)

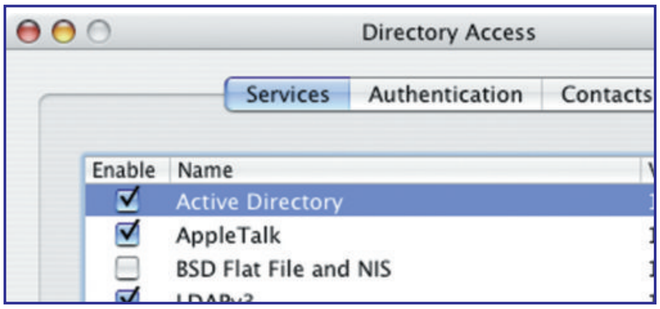


Figure 1 – P4V, the MacOS X client GUI

Introduction

When I was in engineering school, my software engineering professor made a point of saying to the undergraduates, "Use the tools you have." This bit of real-world advice was useful both because the tools we use are rarely the enabling factor for the success of software projects, and also because individual engineers are rarely in a position to dictate the tools to be used on a project. Mac developers are especially aware of limitations imposed on their choices by factors beyond their control. With this in mind, Mac developers are fortunate to have Perforce as an available and well-supported option when shopping for a revision control system.

My Perspective

I work for Nemetschek North America – formerly Diehl Graphsoft – makers of MiniCAD & VectorWorks CAD software. We have a development environment with about 40 engineers interacting with our source code base through Perforce – a tool that we selected three years ago to replace Microsoft's Visual SourceSafe. Before that (a long time before) we used MPW's Projector from Apple for version control. I will use our SourceSafe experience as a point of comparison, and I'll mention CVS comparisons if I happen to know, but I have never used CVS in a production environment. I will try to use Perforce terminology where possible and clarify it where necessary. One exception to this is that I will use the term "check-out" in some cases to indicate the functionality known in Perforce terms as "Open for Edit" because it makes the description more accessible for non-Perforce users.

What is version control

I think it's a safe bet that anyone who works with other developers on a team of any size has a pretty good idea of the basic elements of version control, but to get everyone on the same page, I'll outline some of the basic features. All version control systems allow multiple engineers to work within the same code base by serving as a file librarian and tracking individual users' work on the files that are being modified. They maintain a history of the changes made to files within the system so that earlier versions of the files can be retrieved and differences between versions can be displayed. Version control systems also provide a mechanism to track some additional data about the changes such as who made a change, a description of what was done, and when it happened. Beyond this very basic set of functionality, the capabilities of various systems diverge.

System Capabilities

Perforce, produced by Perforce Software in Alameda, California, is a modern and full-featured version control system intended to be the main repository of all of a software project's files, structure, and history. It has a feature set which can look similar to that of CVS, PVCS, SourceSafe, or ClearCase. With such products, however, a high level outline of the feature set often leaves out a lot about how the product will actually work in a given development environment. Perforce differentiates itself in the following ways:

Robust
Fast & Efficient
Automated merging
Inter-File Branching Model
Atomic change submission
Low administration overhead

I'll cover each in some detail, and then describe some of the Mac-specific tools and functionality. But first, I'll introduce some basic concepts and terminology.

Perforce basics

Understanding Perforce involves a few concepts that I'll cover from a user's point of view so they can be used as a point of reference for examples that follow. Perforce is a client-server system in which the main database runs on a single central server and clients connect using a TCP/IP based protocol to interact with the server. The server can be run on or MacOS X, Windows NT/XP, or various flavors of Linux and Unix. Command line client software is available for almost any conceivable platform, whereas more mature GUI client implementations are available for fewer platforms. Windows has the most mature GUI client software called P4Win - implemented as native Windows code. Mac, Windows, and Linux share a more recently introduced, but very full featured client called P4V (short for Visual) which is about a year old. It is implemented using the QT cross-platform toolkit and is fast and reliable with a native looking GUI on the platforms it supports. The second major release of P4V is in late beta and is what I used on the Mac while working on this review.

Perforce is set up with user accounts for those that will be accessing the system. The "depot" – Perforce's term for the main hierarchy of files under version control – is populated with the files that make up the development projects of the company or department using Perforce. Each user can have one or more client workspaces, which are each associated via preferences maintained on the server with a particular root path on their development machine. Users can have as many workspaces on one or more machines as they find useful. In normal use, the user will keep a copy of some part of the overall depot on their local file system. They will update their workspace files with changes made to the depot by others (described by Perforce as "Sync-ing"), edit files within their local workspace, and submit changes back to the depot.

From this description, Perforce is similar to other version control tools available. Now we'll look at the details that differentiate Perforce.

Perforce is Robust

Keeping source code safe is one of the highest priorities of a software developer, and it's good to know that the tool that is most responsible for the safety of your code places a high priority on maintaining a robust repository. Perforce is architected to facilitate recovery if disaster strikes, but is implemented so well that recovery is rarely if ever necessary.

Perforce is a client-server system in which all client interaction takes place via a TCP/IP connection to a single centralized server. This eliminates a plethora of potential problems compared to systems such as SourceSafe where multiple clients access database files through a shared file system. This architecture gives the server responsibility for recording changes to system data in a way that allows full recovery should disaster ever strike. Perforce uses an industrial strength database for its metadata and provides for checkpointing and journaling, thereby allowing full recovery from most disaster scenarios. Source code files in Perforce are stored using industry-standard formats for reverse-delta storage, compression, and Mac resource file encoding allowing recovery of their content even if Perforce's databases were completely deleted.

The system also has ample tools to assure you that everything is working as expected. Every revision of every file is given an MD5 hash which is stored in the database and it is straightforward to ask the server to verify that every checksum matches for all files and revisions stored in Perforce. It's an easy and common practice for Perforce sites to regularly verify that every revision in the system is corruption-free.

Knowing that Perforce is built with recovery in mind gives you the comfort of sleeping well at night, but productivity is still compromised if you experience frequent problems. Perforce has an outstanding reputation in this regard among its customers, and our site is certainly evidence of their high reliability. In our three years of using Perforce heavily, we have seen only one server crash - related to differencing revisions of a very large file with very long lines. Perforce support worked with us to carefully but quickly identify the problem and a workaround. They had isolated and fixed the cause and issued a public update to their entire product line within a week of the problem report.

Our use of the product has also confirmed for us that it is virtually impossible for any kind of client failure to cause a database or file corruption on the server. We have occasionally seen bugs in the client software which affect specific features, but they have never had material impact on the overall robustness of Perforce, and have, for the most part, been resolved by a subsequent release of the software. Perforce is, as a whole, at least as robust as any other software we use.

A final factor in robust system performance is that Perforce provides outstanding support – especially in case of emergency. I have heard of a handful of cases where a Perforce server was compromised by hardware failure, but have never heard of a significant loss of data. The consensus in the Perforce user community is that they will do anything in their power to maintain the robust reputation of their software.

Every one of these points stands in stark contrast to the experiences we had with SourceSafe, where we would suffer from file corruptions on a weekly basis, and more significant system corruptions every few months. There were no mechanisms to prevent this or aid recovery. The analysis tool always complained of dire corruption, but provided no means of fixing it. Support was non-existent. We felt like we could lose our entire database at any moment. My impression is that CVS is much better in this respect, but support can still be very hard to come by.

Perforce is Fast & Efficient

Speed is not often considered a feature of software, but in the world of revision control where individual operations can involve the inspection or transfer of tens of thousands of files, you will soon come to realize which operations are doing more work than they should – and Perforce prides itself on having built efficiency into the system from the ground up.

The database used by the server that I mentioned as a key element of the system's reliability also has dramatic impact on the speed of most normal operations. I'll use updating or syncing a client's source code as an example. In Perforce, the central database keeps a record of every file revision held by every workspace. If you ask to sync to the latest revision of a set of files you don't have in your local workspace, then the server is forced to send you everything, which can be time consuming. During normal work, however, you will usually already have the current revision of most of the files you're working on. In that case, Perforce will only need to send you the files that have changed since your prior sync operation, and it can determine this with a very fast query on an indexed database without inspecting anything on the client machine's file system. A sync operation on a project with 10,000 files typically takes a few seconds, unless it is very out-of-date. The longest sync operations I routinely see are a couple of minutes. SourceSafe and CVS have no provision for optimizing this common operation and will typically exhibit performance corresponding to the total number of files in the hierarchy being updated. We would often wait 10-15 minutes using SourceSafe, whereas Perforce is almost always done in seconds.

As an example, after a week of vacation, I synced our main development branch in just over 2 minutes and got 3500 new files of 18,600 total files in the branch. I synced a maintenance branch and got 5 new files of 10,000 total in about 3 seconds. Most Perforce operations are similarly efficient, including merging changes between branches. Another example - It took about 15 seconds to list all 243,000 files in our depot to a text file using "p4 files //depot/... > filelist.txt"

An obvious benefit of this efficiency is that off-site work becomes feasible. Three of our users are on another continent connected by a 128kbps internet connection. They certainly need to adjust their work habits to account for the lower bandwidth, but not by much. We have never sent them a code snapshot, and they have never been at our site. Nevertheless, they have the same level of project interaction as our local users. Better still, working over a cable-modem sized pipe for local users connecting from home rarely feels much slower than the 100Mb/s switched Ethernet at the office. Perforce also has a recently introduced remote caching server called Perforce Proxy intended to speed up access for an entire remote site. (Our remote site performance without using Perforce Proxy has been good enough that we have decided not to use this tool yet.)

Perforce Automates Merging

One of the primary benefits of version control is that it enables concurrent development among engineers. The success of this in a production situation varies depending on the extent to which the tools have been refined, and Perforce does this very well. Merging or resolving differences is also an area that has seen marked improvement in the MacOS clients in recent releases – especially P4V, the new MacOS X GUI client.

There are two situations where you can be exposed to the need to resolve differences. The first is during normal development when you are working on the same set of files as someone else. The second is during multi-branch development where one branch has changes which need to be moved or propagated to another branch. For the sake of this discussion, I'll keep it simple and talk about the first case, but bear in mind that all resolve functionality is pretty much the same regardless of whether you are checking in a small file change to the project you are working on or doing a large inter-branch merge.

For example, you and another developer both check-out and begin editing a file at the same time, but your changes are more extensive, and take longer. You attempt to submit your changes and find that the latest revision of the file you've been editing is newer than the one you started with because the other engineer submitted her changes first. Perforce provides a great deal of control over this process using their resolve functionality. Whenever you ask Perforce to update a file using another version of that file, it uses the always-present database to determine what kinds of changes might be coming across and what kind of action may be necessary. It knows if you have opened a file for editing operations, so if you ask to sync to a newer revision of that file, you may need to resolve potential conflicts between the changes made. Similarly, if you try to submit changes to a file that has been changed in the depot by someone else, you may need to resolve differences. This is the case described above, and there are a number of resolve options available to the user.

After any operation that can encounter conflicting differences, the Perforce GUI indicates files that need to be resolved with a special icon. First, you might try an "Automatic resolve" in which changes made by either engineer will be merged to the result file as long as the changes do not overlap. This typically works and the file is merged, but if it fails because of overlapping changes, you will want to interactively merge the changes. When you merge interactively, you are given the opportunity to review all of the changes made by yourself and the other engineer, and choose which to use. The conflicting changes are the most interesting, as they are the ones that require some modification to the code to preserve the original intent of each change in the final merged file. Perforce will typically handle all but the conflicts automatically, leaving only the work that benefits most from direct user interaction.

If necessary, Perforce will apply all of these operations to very large sets of files at once. My company routinely uses feature branches to accomplish large development projects, wherein a complete set of features is developed outside of our main development branch to avoid disrupting other engineers. This development effort may go on for months, and affect hundreds or thousands of files. At the end of all of this, we need to get the changes back into our main branch intact. Without discussing too many details, I'll cover the process you would use with Perforce in this situation.

First, you'd tell Perforce to "integrate" changes from one branch to another. In other words, you are specifying what set of changes to merge without telling it specifically how you want that accomplished. You can limit the scope of the integration based on the path to the files, or specific versions of the files, but you can just as easily tell it to do the whole set of changes at once. Perforce will then "check-out" all of the files in the destination branch that were changed in the source branch. All of these files now have the special icon that indicates Perforce is waiting for you to specify how to resolve differences.

You would then tell Perforce to automatically resolve all the files that had no conflicting changes. This operation usually eliminates about 90%-95% of all changes with no manual work on the part of the engineer doing the merge. It can be done in a single step no matter how many files are involved.

Finally, you are left with a much smaller set of files that still have the special icon that indicates they have differences which have not yet been successfully resolved. Now you'll need to resort to interactively merging the conflicting differences using the visual three way merge tool provided by Perforce.

The above discussion may be too detailed for some, but the overall concept is that even when manipulating large sets of files, Perforce always tries to avoid involving you if it's not necessary, but if there are situations that need your attention, you will be involved, and given the detailed information you need to proceed efficiently. Perforce provides a consistent set of integrate & resolve functionality that is applied the same to all merging operations. It stands far ahead of CVS or SourceSafe in this respect, and ahead of most non-MacOS version control tools as well.

Inter-File Branching

Since we have been talking about merging between branches, I'll discuss the branching model used by Perforce. They call it Inter-File Branching, but in essence it is the use of the depot directory hierarchy to represent different branches of your development projects. The path to each individual file in the depot includes a full human readable representation of the intended purpose of that file. For example, it's easy to differentiate the intent of these two files:

```
//depot/Engineering/VectorWorks/ReleaseBranches/VectorWorks10.0
.0/AppSource/Project Setup.txt
//depot/Engineering/VectorWorks/TaskBranches/VW10/3DDevelopme
nt/AppSource/Project Setup.txt
```

Behind the scenes supporting the seemingly simple Inter-File Branching concept is the Perforce database. which is aware of all branching relationships between any two files in the system. For every pair of files that has a branching relationship, it tracks the specific revisions that have been integrated. For large hierarchies of files that are related to other branched files, it's quite easy to display all changes made to a particular branch chronologically, or even to display all revisions in one branch that have yet to be merged to another.

To a large extent, the Inter-File Branching model works in concert with the automated merging capabilities described above. It allows independent branches within the codebase (and their independent change histories) to exist in a logical environment where every engineer cannot help but know how they are differentiated from each other simply by virtue of the path to the files. Inter-File Branching provides the conceptual foundation that can keep large teams of developers efficiently working on parallel development branches with very little management overhead.

Other tools such as CVS have a file hierarchy for your files, but then each file has a tree of numeric versions with no immediately apparent meaning. Thus every important event in CVS needs to be represented by a label that pulls together an arbitrary set of files and versions into a meaningful package. What an individual engineer needs to do in CVS to accomplish a simple task such as "Merge all of your version 4.1 changes into the main development branch" becomes so complex and error-prone that it prevents projects from even attempting to do large-scale parallel development. (Perforce also has labels, but they are rarely used because other Perforce functionality makes them much less necessary.)

SourceSafe has no meaningful tools to assist merging changes between branches and cannot effectively be used for any significant parallel development efforts.

Atomic Change Submission

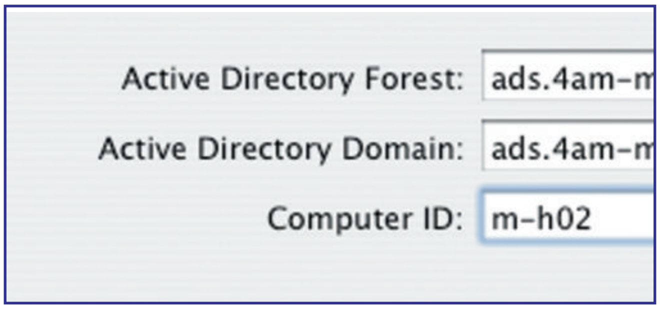
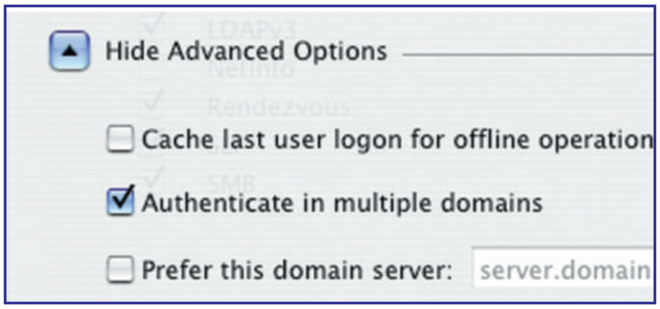

Active Directory Forest: ads.4am-m
Active Directory Domain: ads.4am-m
Computer ID: m-h02

Figure 2 – Revision history of a single file with changelist comments



Cache last user logon for offline operation
☑ Authenticate in multiple domains
Prefer this domain server: server.domain

Figure 3 – Pending changelists before submission

Atomic change submission is another refinement that stems from the rigorous database architecture developed by Perforce at the outset. The concept is simple to grasp, and prevents a host of ugly side-effects which afflict competing products without this feature. Simply stated, if I try to submit changes to a set of files, and for whatever reason I am unable to change one or more files, then the entire submission is rejected. This dramatically improves the chances that the project as submitted to the version control system will be in a consistent, and buildable state, and improves the ability to analyze complex changes that have gone into the project after the fact.

In SourceSafe or CVS, If I submit ten files, and the last one has a conflict with a change that was already submitted by another engineer, I won't know it until the first nine have already been submitted. At that point, I may have a big problem, and will need to scramble to come up with a fix. If other engineers step into that trap, and check-in more files before the problems are solved, then the mess keeps getting bigger. In Perforce, if I submit ten files, and one of them has a conflict, then the entire submit fails before the central database is modified at all. I can then resolve the conflict without the pressure of having just checked-in a partial set of files which do not build.

In normal use, the organization of work allowed by Perforce changelists is also very beneficial. All of an engineer's open files are assigned to one or more pending changelists visible to all users of the system. (See Figure 3) The choice of which files to include as well as the description of the changes can all be prepared in advance to eliminate last-minute errors when submitting changes to the depot.

Unlike some other revision control systems where atomic change submission is tacked on as an afterthought, it is core to the implementation of Perforce. Every change that has been successfully submitted to the system is represented by the set of files that changed, a high level description of the significance of that change, and a list of files that were affected. The description of a change applies to the entire change (and all files that make up the change) rather than each individual file being given a duplicate of the description. The history of a hierarchy of files includes a list of the high level changes and their descriptions, not the less useful list of every file that changed between two dates. One can even implement server-side trigger scripts that can examine a proposed submission and programmatically accept or reject it in its entirety based on a centrally maintained submission policy.

Low Administration Overhead

Perforce requires very, very little administration attention. If you install the server properly, set up your backups, and maintain the server hardware with adequate RAM and drive space, the only administrative attention required is upgrading the software as frequently or infrequently as you like, and a small amount of overhead when adding new user accounts or cleaning up after users who depart. Perforce has never once created an emergency for us, and I can't see a site needing a full time administrator for this system until it has many hundreds of users. Even then, I think there would be a lot of free time on that person's hands.

SourceSafe will quickly burden an administrator with unpleasant tasks such as the investigation and patching of file and database corruptions. Based on our experience, and that of others I've talked to, this is virtually certain over the long term with more than a handful of engineers using the system.

CVS will probably be less troublesome with regular maintenance, but the lack of thorough documentation and support and the need to assemble client and server pieces from various open source projects can certainly add to the initial outlay of effort.

Other info

There are some other features that do not warrant extensive discussion, but are important to mention. Perforce maintains the ability to fully access all of the client functionality through their command line tools. This is the universal Perforce interface. It's available everywhere and can do anything that can be done with Perforce. This means that there is a backup plan if you ever run into something that can't easily be done using the GUI. It also means that the system is highly scriptable and extensible, as you'd expect any mission-critical developer tool to be. It works well with most any scripting environment (Python, Perl, and possibly Ruby being the most commonly used).

Mac Software Support

Having talked at some length about the core system functionality, I'll spend some time talking about Mac-specific support. As far as the server goes, MacOS X is a supported platform as well as Linux & Windows. We use a Windows 2000 server platform, and there are no issues I'm aware of related to mixing Mac & Windows clients and servers.

The client software on the Mac includes MacOS X native command line tools as well as P4V - the visual GUI client, P4Web – a web browser based client, and a CodeWarrior plug-in. There are also legacy clients for MPW, older versions of CodeWarrior, and a MacOS 9 based version of P4Web. Finally, Apple has integrated native Perforce support into the Xcode development environment.

GUI Interface – P4Web & P4V

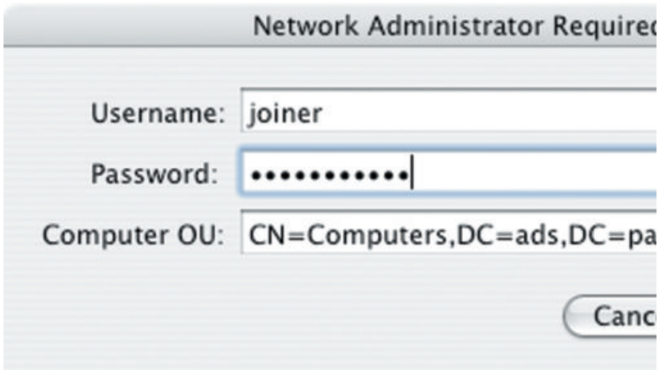

Network Administrator Requirec
Username: joiner
Password: •••••••••••
Computer OU: CN=Computers,DC=ads,DC=pa

Figure 4 – Graphical text differences display

P4Web used to be the only "GUI" interface to Perforce on the Mac, and we used it successfully for a couple of years. It takes a little getting used to, but became quite easy to use, if a little slow. From my point of view, there is no reason for most users to continue using P4Web as the current version of P4V is faster, easier, more capable, and nicer looking. It introduces much better file differencing and merging capabilities, which were sorely lacking on the Mac under P4Web. Figure 4 is an example of the kind of text difference display P4V produces. You can easily display the difference between your local copy and any version in Perforce – or between any two versions of any file in the depot.

P4V is being developed by Perforce as the next generation GUI for Mac and Linux. With only a few exceptions, P4V has the full functionality of P4Win – the native Windows GUI client. In some respects, such as the graphical diff viewer, it's better. I think Perforce would ultimately like for P4V to be the only GUI client – even on Windows. They may have a way to go to achieve that, but they are rapidly improving P4V, and it's already at a state of good usability.

CodeWarrior Integration

The Perforce CodeWarrior plug-in is most useful for environments where CodeWarrior is the development platform, and you want quick access to syncing, checking out, and differencing files from within the IDE. You can submit changes from within the IDE, but the ability to resolve conflicts if they occur is weaker than either P4Web or P4V, and you'll probably gravitate to those more capable tools.

Xcode Integration

Apple has integrated Perforce support directly into Xcode, and it provides the same feature set as their CVS integration – namely sync, check-out, check-in, and diff. As I don't use Xcode for production work, I'm not sure whether this integration works better than the CodeWarrior integration when submitting conflicting files. Either way, we don't find it difficult to use the GUI tools for the more demanding tasks.

References

There is a long list of major software companies, and projects with hundreds or thousands of developers who are happy Perforce users. See the customer spotlight page at http://www.perforce.com/perforce/customers.html for a bunch of interesting reading. Companies like Palm, Symantec, Macromedia, and TiVo, among many other household names, have standardized on Perforce as a best-of-breed solution for Mac development as well as virtually any other platform.

Pricing

Current pricing for Perforce is $750 per user, which includes a year of upgrades and support. Continuing the upgrade & support contract costs $150 per user per year. Perforce has special site licensing available for educational institutions, free licenses available for open source projects, and an unlimited time evaluation version which includes two users and two workspaces. If you'd like to evaluate more in depth in a production environment, Perforce will supply you with a time-limited license enabling a larger number of seats, depending on your environment.

Final Word

Perforce is a full-featured revision control system that differentiates itself from the competition by the uncompromising quality of its implementation. Mac support was acceptable three years ago, but due to recent improvements such as P4V for MacOS X, it is now very good, and still improving rapidly. Perforce is developed by a team that sets priorities early, and sticks to them. Producing a top-notch Mac development product is clearly one of their priorities. Oh - and if you're interested – Perforce is clearly one of the top players in Windows and Unix version control as well.

About The Author

Paul Pharr manages the ongoing software development of the VectorWorks family of CAD applications at Nemetschek North America. You can reach him at pharr@nemetschek.net.

What's under *your* hood?

High-performance inventions are driven by powerful and robust engines. That's why OpenBase SQL delivers real fault-tolerance—and the horse-power today's multi-user applications need.

Find out how **OpenBase SQL** stacks up against other databases at http://www.openbase.com/databases.pdf

OpenBase Forms™ database GUI building application available soon for OpenBase SQL!

Test-drive OpenBase SQL
with a free, single-user developer license

What will you build with OpenBase?
www.openbase.com/testdrive

"OpenBase allowed us to quit worrying about the database so we could focus on our business."

Josh Paul, Overhyped Technologies,
creator of software that stores film clips for Reality TV shows.

NoCode Browser

Using the Apple's Web Kit SDK to make a web browser

By David Linker

Want some rapid development?

One of the advantages that is mentioned about the Cocoa development environment is the well-developed frameworks that are provided, giving impressive functionality without much effort. A recent addition is Web Kit, which is the framework which provides a full suite of components required for web browsing, and are the basis of the Safari web browser from Apple. To illustrate the power of Web Kit, Apple explains in the documentation how to make a browser with only one line of code. Some folks have expanded on this explanation, and in the discussions that ensued, others pointed out that it is possible to write a browser using Web Kit with NO lines of code. This article will explain how, and discuss some of the capabilities of Web Kit as well.

Fast lane

In this age of 24/7, just-in-time and accelerated learning, some of you may want to get right to the point, while others want more details. To satisfy the more experienced readers, or the less patient, here is the executive summary of the main points necessary:

Install Project Builder, December 2002, if not already installed
Install Safari, if not already installed
Install Web Kit
Make a new Cocoa Application project in Project Builder
Add the Web Kit Framework to the project
Open up the MainMenu.nib file in Interface Builder
Drab the Webview header file to Interface Builder
Add a Customview to the main window
Change its type to Webview
Add a text box to the main window
Connect the target output of the text box to takeStringURLFrom in the Webview
Build the project
Type a URL in the text box and hit Enter
Browse away!

The rest of this article fills in the details, and add a few niceties along the way.

Origins and background

Apple built the technology used in its new browser, Safari, on existing open source projects. KDE is an open source "desktop" environment for unix. Part of KDE is a set of tools for rendering HTML, called KHTML, and a set of tools called KJS, which assist with scripting. These two components were used by Apple to develop the core classes that are used in Safari. The framework incorporating these classes which they developed for Safari is called Web Kit, and Apple released the interface to Web Kit during the WWDC in June of this year.

Web Kit provides an amazing amount of functionality with little effort. It supports HTML, DOM, SSL, Javascript, stylesheets, embedding Java applets, and a "history" of recent sites. What this means in practical terms is that it is possible to incorporate all of this functionality in your applications with minimal effort.

Apple provided a very terse description of how to do this on the pages describing the use of Web Kit at:

http://developer.apple.com/documentation/Cocoa/Conceptual/DisplayWebContent/index.html

If you choose "Simple Browsing", there is a short description of how you can create a browser with one line of code, which is as follows:

```
[[webView mainFrame] loadRequest:[NSURLRequest
requestWithURL:[NSURL URLWithString:urlText]]];
```

Of course, there is a fair amount of "wrapper" that has to go around this to incorporate it in a program, including making a header file, class file, and all of the connections in the interface.

Name that tune!

Many years ago, was a TV game show called "Name that tune". In it, the contestants would try to name a tune in the fewest notes, challenging each other with "I can name that tune in N notes", where N was a small number, and the person who named the smallest number got the first chance.

I had a flashback of that show when I was reading through a discussion about using Web Kit. The original page, by Martin Simoneau, was an excellend article on Cocoa Dev Central filling in the details on how to make a browser using the line of code provided by Apple, and the Web Kit framework.

http://cocoadevcentral.com/articles/000077.php

In the follow-up discussions that were posted, there were comments that the one line of code was unnecessary! This sounded to me like "I can name that tune in no notes!". There were some very brief descriptions of how to do this, and then a link to another page which described how to do this in slightly more detail.

http://www.livejournal.com/users/foxmagic/238347.html

The essential ingredient is the fact that there is already a connection called takeURLFrom, which will extract the URL from a text field. This makes it possible to create a functional browser without writing any code.

To do this, there are three essential step that are necessary. The first is that you need to have Safari installed, since that also installs the Web Kit framework. You can find it at: http://www.apple.com/safari/download/ if you don't have it already. Next, you need to have the developer tools, December 2002 version installed, if you haven't already. To get this, you need to become an ADC developer, but fortunately you can join for free. The site to get this from is http://connect.apple.com/. Follow the links Download Software -> Developer Tools to find what you need.

Finally, you need to get the Web Kit SDK. This is also available at the same site, under Download Software -> WWDC 2003. Once you have installed all of the software, you are ready to go.

Start up Project Builder, and choose New Project from the File menu. Pick Cocoa Application from the list, and enter the name NoCodeBrowser in the Project Name field. Click on Finish.

Now, we have to add the Web Kit interface to the project. Choose Add Frameworks from the Project menu (see **Figure 1**).On the list that comes up, choose Webkit.framework and then click Add on the next dialog. If you want to be very neat, you can move the Webkit.framework to the folder Other frameworks under Frameworks. Save the project.

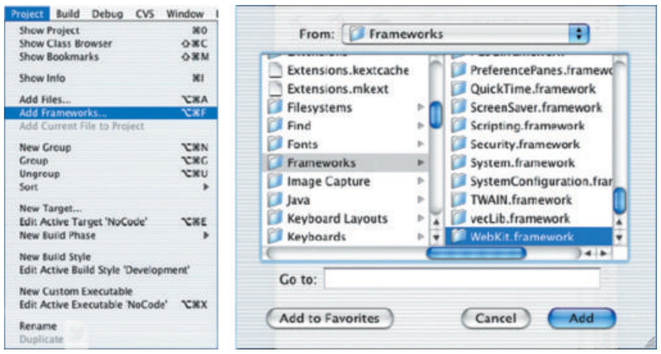


Figure 1- Select Add Framework from the Project menu (left), then choose WebKit.framework from the dialog that comes up, and click Add (right).

We now move to Interface Builder. Click on the little triangle to the left of Resources in the project window, and double click on Mainmenu.nib. This will open Interface Builder, with a Window called "Window". In the window titled Cocoa-Containers, click on the second icon from the right on the top, which shows a tabbed window. Drag a Customview over to "Window", and drop in in the window. Resize it to fill almost the entire window, with space to put a text box at the top. Click on the second icon from the left in the "Cocoa-Containers" window, and drag a text field to the top of the window, and resize it to make it wider.

Now, arrange the windows so that you can see the window "Mainmenu.nib" in Interface Builder, and the main project window from Project Builder. Open Webkit.framework, and the Headers folder inside that (**Figure 2**). At the bottom, there is a file called WebView.h. Drag this over to the Mainmenu.nib window, and drop it there.

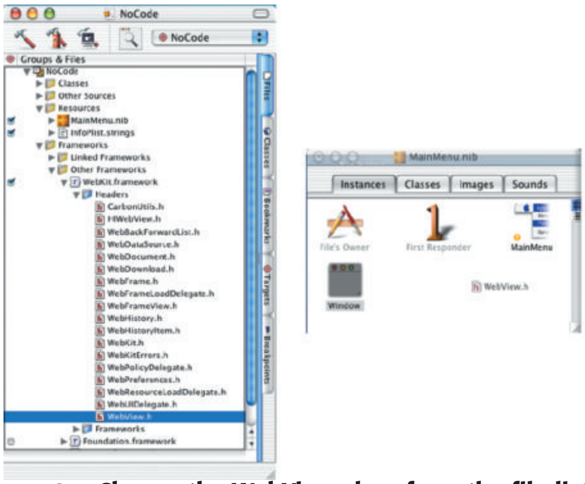


Figure 2 – Choose the WebView class from the file list in ProjectBuilder (left), and drag to the MainMenu.nib window in InterfaceBuilder (right).

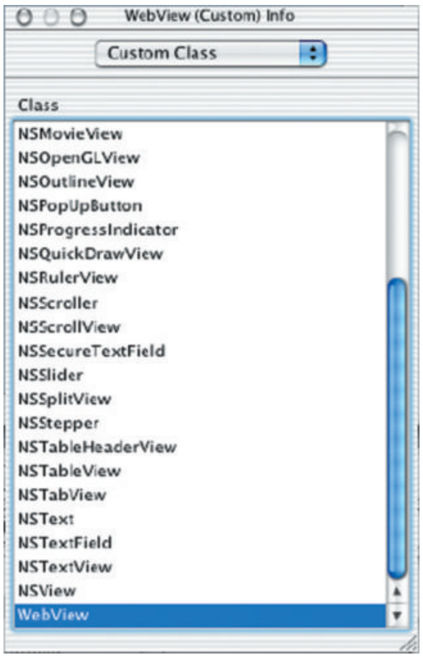


Figure 3 – In the Show Info window, select Custom Class from the pop-up, and then select WebView as the class.

Now, select the CustomView in Interface Builder, and choose Show Info from the Tools menu. On the pop-up menu that says Attributes, choose Custom Class, and then choose WebView from the list (**Figure 3**). Close the info window. Now, use ctrl-click and drag to make a connection from the text field to the WebView (**Figure 4**). In the window that pops up, make the connection from target to takeURLFrom, and click on connect. Save everything, and then build, using Build and Run from the Build menu.

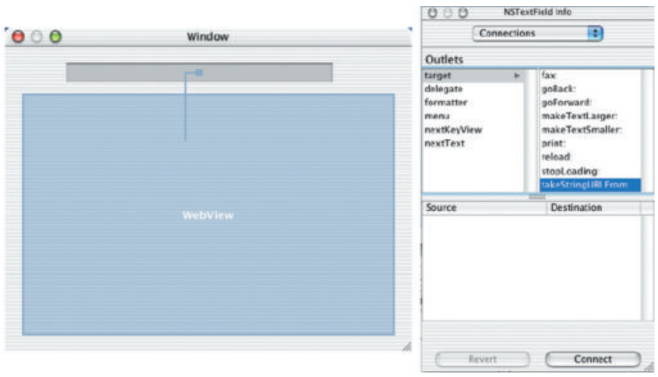


Figure 4 – Connect the text field to WebView and then specify that the connection is to takeStringURLFrom

You can now type a full URL in the text field, and when you hit enter, the page will load! Note that the URL must begin with "http://". You can then click in links in the loaded page, and those links will load as well. You can navigate to secure pages, load java applets, execute javascript, and lots more. Play around with it!

Feature Fill

A number of things are missing but easily added to this first version. First, we can add additional functionality without any more code.

WebView maintains a history of recently visited pages by default. Methods exist in the WebView class to back up a level (goBack), go back down a level (goForward), reload a page (reload), or stop loading a page (stopLoading). All we need to do to implement these is to add a button for each function, and connect them to the WebView class and the appropriate method. We can then add appropriate text or icon to each button.

Another problem is that the WebView does not change it's size when we resize the window. This is easy to fix. Click on the WebView in Interface Builder, and then choose Show Info from the Tools menu. From the pop-up, choose Size (**Figure 5**). The box indicates the WebView object, and the straight lines indicate a fixed relationship. If the lines are straight within the object, the size will not change with a resize. If the lines outside are straight, the relationship to the containing window will not change. If all of the outside lines are straight, the object will be centered with a resize. If you click on a line, it turns into a "spring", which will allow resizing. You can do the same thing to all of the other object, such as the buttons and the text box, to control their behavior during resizing as well.

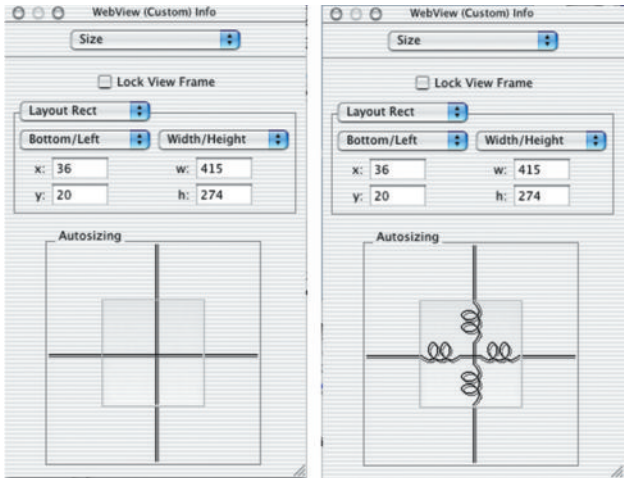


Figure 5 – Click on the interior lines in the box (left) to turn them into "springs" (right), to allow the WebView to resize along with the window.

The final refinement is to change all of the menu items that refer to "NewApplication" to refer to "NoCode Browser". A picture of the main window in my finished version is in **Figure 6**.

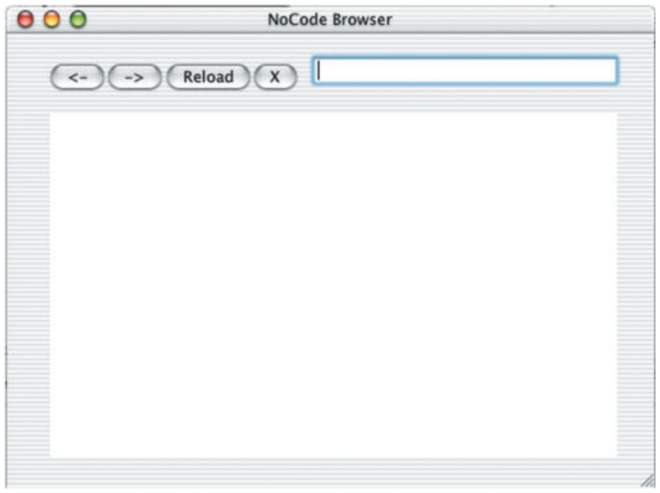


Figure 6 – The final appearance of the main window, after adding back, forward, reload, and stop buttons.

Resources and additions

Although this demonstration is impressive, there are a lot of missing pieces if we were to try to make a complete program.

If you click on a link which should result in opening another window, nothing happens. The same thing goes for email links, download links, and anything other than navigation to another page.

The current version has no error checking, and no error messages. Probably the worst, obvious omission is that if Web Kit framework is not loaded, that is, if Safari has not been installed, the program will not work and will probably crash. Methods for dealing with this are explained in the tutorial pages at:

http://developer.apple.com/documentation/Cocoa/Conceptual/DisplayWebContent/index.html

WebKit has a number of hooks to allow changing or enhancing its behavior. The use of these links is also explained at the tutorial pages.

Finally, there is a Web Kit discussion list at:

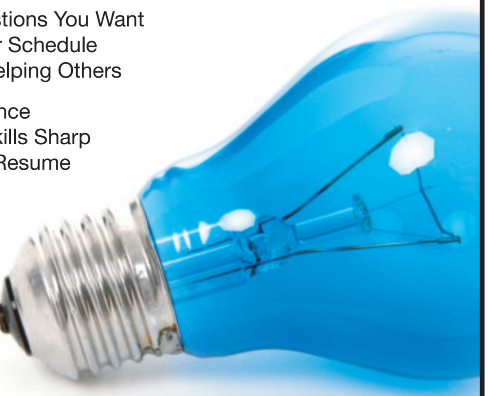
http://lists.apple.com/mailman/listinfo/webkitsdk-dev

In additon to providing impressive functionality that you can use in your programs, WebKit provides a dramatic demonstration of the power of the frameworks available under Mac OS X, allowing you to create a web browser without writing a single line of code. What other platform lets you do that?

MT

About The Author

David is a lover of Mac OS X, because the rich development environment and frameworks allow his inherent lazyness to blossom. You can reach him at dtlinker@mac.com.



THE APPLICATION FORMERLY KNOWN AS. . .

THE APPLICATION FORMERLY KNOWN AS. . .

By Dean Shavit

HOLDING OUT FOR QUARKXPRESS

It was ten years ago this summer when I helped a client, a small monthly print magazine, bring their desktop publishing operation in house. At the time, the operating system (Mac OS 7.1) and the hardware (Quadra 840av) were very different than today, but two key components of the solution still remain in place for my client and other publishers: QuarkXpress for the layout of the publication, and Adobe Type 1 fonts for the typefaces. The initial project had some typographical obstacles including body text that borrowed capital letters from other fonts for use with the main typeface, and some other elements from disparate fonts, including asterisks and dashes within the body text. These peculiarities made the cost of laying out the magazine quite high, since the typographers either had to search and replace the characters every month, or change the selected font as they typed the text.

So, while planning the project, I proposed that they edit their font (Bembo) to accommodate the capital letters from other fonts, the dashes, and the asterisks so that they would appear naturally within the body copy while a typesetter was entering text. At the time, it was also clear which tool was appropriate for the customizing the typeface: Altsys Fontographer (now owned by Macromedia, and still available for purchase but never updated to be a native OS X application). The client purchased Fontographer; I went ahead and modified the font for them, and they adopted the Mac for their desktop publishing (and have been a Mac company ever since), eventually leaving behind dbase for their circulation database and moving to FileMaker. The resulting font, called "SpecialText" proved to be a great solution, as they migrated from Mac OS 7 to 8, and 9, and even OS X in mid-2003.

With the successful transition of the client to OS X and an Xserve and Xserve RAID system, the one application that lagged behind was QuarkXpress 4.1, which they ran in the Classic environment, but grudgingly. My suggestions to look at Adobe InDesign as a possible OS X native replacement for QuarkXpress fell on deaf ears. After all, they publish a relatively straightforward text-based magazine, and just want to work with the same familiar application, just not in Classic mode, and even so, everything worked as they expected, with some additional complexity brought on by the GUI changes when switching from Quark to OS X applications such as FileMaker or Microsoft Word but on the whole without disrupting their work flow.

QUARKXPRESS 6 IS RELEASED - "START THE PRESSES"

When QuarkXpress 6.0 was released in June 2003, after two years in the making, the message on Apple's home page proclaimed, "Start the Presses." QuarkXpress 6 was the last major OS X desktop publishing application to be updated for native OS X operation, and many publications had been waiting for just that moment to make the move to OS X, or upgrade from QuarkXpress 4 or 5. Quite naturally, my client was very interested in upgrading, but based on severe issues with .0 releases of QuarkXpress in the past (versions 3, 4, and 5) I advised them to wait a while to see what early adopters experienced. So they waited a few months, then went ahead and purchased QuarkXpress 6 in January of 2004, with the goal of putting into production by March 2004.

STOP THE PRESSES

After some initial testing the client decided to put QuarkXpress 6.0 into production in early February. Evidently, the initial tests didn't include printing documents, because immediately upon installing and attempting to use QuarkXpress 6 on their design Macs, I received an emergency call, "Quark is claiming that our SpecialText font is corrupt and won't print anything." After talking them through installing fresh copies, changing font locations, disabling their Suitcase XI font management software, we were still unable to get QuarkXpress to cough up a printout.

So there it was, with a deadline looming only a two weeks away, the presses were stopped, the work flow halted and an emergency dumped squarely in my lap: the fonts I'd edited ten years earlier were corrupted or "not done right in the first place" in the opinion of my client. What I'd heard from other consultants was indeed true: Quark had outsourced their technical support to India, and "John," who spoke the King's English beautifully, was obviously reading from a script, and had absolutely no Mac OS X experience, or knowledge about the font issues my client was having that I could discern. Our discussion yielded nothing except a general disclaimer, "it's not our fault, there must be something wrong with your font." That was quite difficult to swallow on my part, considering that the SpecialText font had performed flawlessly without a single error since 1994, with all previous versions of QuarkXpress, all the Adobe design applications up to the current version, and all iterations of FileMaker and Microsoft Office. A quick review of QuarkXpress 6 issues on the Internet yielded many examples of other adopters experiencing the similar difficulties, including problems creating PDFs from within QuarkXpress, and even distilling the Postscript output with another tool such as Acrobat Distiller or OS X's built-in Preview application.

The temperature kept rising at my client day by day; they began looking for a "Quark Expert" who could solve their problem, even though I insisted that the skills necessary for solving the problem went far beyond the realm of Quark expertise. So, with the clock ticking, I began to examine the SpecialText font for possible issues that would offend Quark. The first step was to check the bitmap fonts for possible corruption—I loaded them on an OS 9 Mac and ran ATM Deluxe's "verify" feature, which turned up no alerts, conflicts, or corruption. I then loaded them on an OS X Mac and ran the Font Doctor program that came bundled with Suitcase XI, but found no warnings as well, except for one interesting flag, that mentioned "extra fonts" in the suitcase file that weren't necessary.

THE GREAT OS X FONT DIASPORA

One of the biggest complaints I've heard, and still hear, about OS X is what I like to call the Font Diaspora. It's almost as if Apple took our familiar Fonts folder inside the OS 9 System Folder and scattered the contents to the four winds . . .well, not quite. If you count up the possible font locations, there's more than just four! I've created a small table below showing where OS X can store fonts.

I've had many questions from clients regarding why Apple chose to transition from a single repository of fonts in Mac OS 9 to this multi-tiered structure in OS X. The answer lies in the more complex nature of OS X as a multi-user operating system, where various rights dictate which parts of the OS are accessible to standard users, admin users, and users with root access (the operating system itself). What most Mac admins do in design studios these days is simply strip out undesired fonts, and let their font management software handle things for the user. While Apple's Font Book program has at least given users the ability to deactivate fonts which reside in areas of the OS to which they don't have read/write access, it still is a long way from being the font management solution that designers or layout operators need or expect in a production environment.

Another significant change that further complicates font management in OS X v. OS 9 is the addition of several additional supported font formats to the familiar Postscript Type 1, Type 3 and TrueType. Here's a list of supported font formats in OS X:

- OpenType (should work on Mac OS and Windows)
- PostScript Type 1 (Mac OS only)
- PostScript Type 3 (Mac OS only)
- PostScript Multiple Master (Mac OS only)
- TrueType (both Mac OS and Windows formats)
- dFont (data fork TrueType) (Mac OS only)

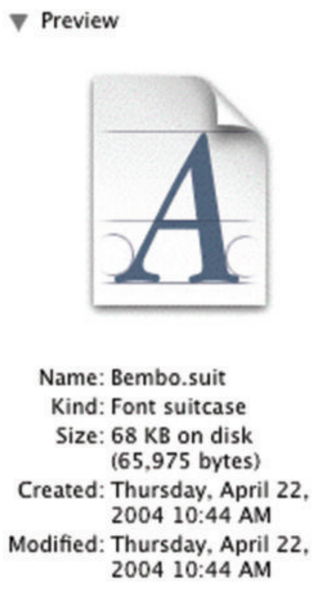
Font Locations

Path (lookup order)	Classic	OS X	Access for Installation without Font Book
/Network/Fonts (6)	No	No	Server Administrator
/System Folder/Fonts (5)	Yes	Yes	Admins only
/System/Library/Fonts (4)	No	Yes	System (root) account only
/Library/Fonts (3)	No	Yes	Admins only
/Users/user/Library/Fonts (2)	No	Yes	Current User only
/Library/Application Support/Adobe/Fonts (1)	No	Adobe Apps only	Admins, only accessible to Adobe applications
FontManager (Suitcase, Font Reserve, FontManager)	Yes	Yes	Depends on location, /Users/shared folder is often used for multiple users

The data fork TrueType fonts that come bundled with OS X are often a source of conflicts, since they are some of the same standard faces such as Times, for instance, that make up the common base set of Adobe Type 1 fonts that are the staple for graphic designers. Data fork TrueType fonts, or “dfonts” as they're often called, are essentially the same as the resource-fork-based TrueType fonts that used to ship with OS 9, but have all of their information in the data fork, so that they can't be damaged by command-line functions like cp or mv. These fonts are often removed as a routine step in preparing an OS X workstation for a designer.

Out of all the formats supported by OS X, it is the OpenType format that's the most intriguing. OpenType is the result of an effort by Adobe and others to produce a font format that works on both Windows and Mac OS 9 and OS X without having support separate versions. Other benefits include a single font file (no separate bitmaps), and a larger, 16-bit address space, instead of an 8-bit address space, allowing for approximately 65,000 “glyphs” rather than the previous limit of 256. The extra capacity allows type foundries to embed swash caps, alternate characters, and more punctuation and dingbats into the typeface, which previously required the production of “Alt” or “Expert” fonts just for that specific purpose. A unique problem with OpenType, however, is that applications must be OpenType-aware to access the extra character space, or they will not work as intended. Accessing the complete set of OpenType characters requires using the Character Palette in OS X, or an application such as one of the Adobe Creative Suite applications that has that ability built-in. To access the Character Palette, open System Preferences, choose the “International” preference pane, click on the “Input Menu” tab and make sure that the “Character Palette” is selected. As an added bonus, you can also select the “Keyboard Viewer” check box, which replaces the “Key Caps” application in OS 9 and versions of OS X prior to Panther.

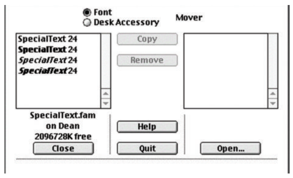
One of the necessary font-related features of OS 9 that has never made it back to OS X, is the ability to move bitmap fonts between font suitcase files by double-clicking, dragging and dropping. In OS X 10.3 double-clicking a font suitcase simply opens the Font Book application, allowing the user to view the fonts contained within it, but doesn't allow deleting of individual fonts or dragging them from one suitcase to another.



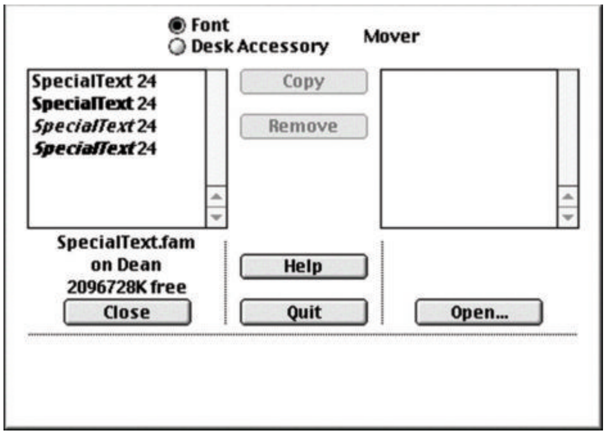
Name: Bembo.suit
Kind: Font suitcase
Size: 68 KB on disk (65,975 bytes)
Created: Thursday, April 22, 2004 10:44 AM
Modified: Thursday, April 22, 2004 10:44 AM

Font Suitcase viewed in the OS X Finder

The ability to double-click on Font suitcase files first appeared in Mac OS 7, supplanting the venerable Font/DA Mover application for Mac OS 6.0.x. So, wanting to see just what was in the SpecialText suitcase file, it was time to fire up the WayBack machine and obtain the last release version of Font/DA Mover 4.1 from the Apple “older” software repository at: HYPERLINK "http://www.info.apple.com/support/oldersoftwarelist.html"http://www.info.apple.com/support/oldersoftwarelist.html.



Font/DA Mover 4.1 Icon



Font/DA Mover Dialog

Using Font/DA mover, I was able to remove the extra font from the suitcase, and this allowed QuarkXpress to finally cough up a printout, but with a bitmap version of the font, and still with the alert that the font “SpecialText might be corrupt.” I felt I was getting closer to a solution with my focus on possible problems with the SpecialText bitmap or the font suitcase which contained it, but moving the fonts to a new suitcase didn't help either. Nevertheless, getting re-acquainted with Font/DA Mover was a real blast from the past, a truly amazing instance of a 14-year-old program still working perfectly in Classic, still useful after all these years.

BACK TO THE DRAWING BOARD

So, my client and I began hunting for the original Fontographer install diskettes, and when we eventually found them, we discovered that they were unreadable by my USB floppy drive (they were 800k disks). So, after digging a PowerMac 9600 out of the closet, I was able to install Fontographer and re-generate the bitmap fonts for SpecialText. However, QuarkXpress had the same complaint about the font. Evidently, the way that particular version of Fontographer produced bitmap fonts (with the file type of NFNT, to be exact), wasn't up to snuff. Other Quark users on the Internet had reported similar printing issues and font corruption messages with older releases of Type 1 fonts that hadn't been edited, such as Frutiger.

So, it was up to me to either purchase, or convince my client to purchase, a newer version of Fontographer (which, though it might be more up-to-date than the Altsys iteration my client owned) that wasn't even OS X native, and there still would be no guarantee that the bitmap fonts it produced would be Quark-digestible. It was time to look elsewhere. Searches on the Internet turned up an OS X native application called Font Lab (http://www.fontlab.com), which cost $549, and which by all available reviews and the features listed was “the bomb” when it came to font-editing and conversion. However, as my client was already unhappy, I felt that asking them to spend the money for Font Lab would have been a bad idea. Since I am not a professional font designer, or even a professional graphic designer, spending $549 to fix one specific font problem seemed to be too costly. It was time to look for other possibilities.

X MARKS THE SPOT

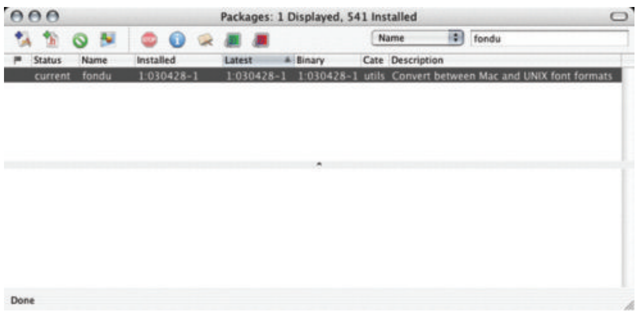
Up to this point, possible solutions to this sticky issue had spanned OS X native applications (Quark) utilities from Mac OS version 6.0.8 (Font/DA Mover), an application from the Mac OS version 7.1 days (Fontographer) and utilities from OS 9 (ATM Deluxe). The only free alternative left for me was to search for an open-source solution.

I am devoted to using open-source software on my Mac. X Windows on OS X is a great solution for many tools that would cost hundreds or thousands of dollars for their commercial counterparts. In May of 2003, I gave a presentation to the Chicago regional chapter of the Apple Consultants Network on X11.app (Apple's release of X Windows for OS X) and the Fink Project (HYPERLINK "http://fink.sourceforge.net/"http://fink.sourceforge.net) which ports open-source Linux and BSD software for use on OS X. I specifically remember one consultant's reaction to the presentation when it was announced. He felt that X Windows on OS X was “too technical” and would not attend the afternoon presentation, just the morning business meeting, because he felt it was a technology he would never use, one that provided no useful solutions to his clients. Although he was the only consultant who spoke up, I was pretty sure others had similar opinions: that running Fink and X11 was too techy for the casual user or even most graphic designers, or maybe even themselves.

However, in this case, my familiarity with Fink, X11, and Xcode (Apple’s free Developer Tools) provided me with the ability to solve my client's font problem. So, I'm going to go out on a limb and say that all Mac consultants should have Fink and X11 in their bag of tricks. Not having that capability means paying for commercial software (understandable if a customer is to use the solution and doesn't have the time/patience to learn how to work in X11), so why shouldn't a consultant be prepared, save money, and be able to service their customers' needs? In this case, the hours of fiddling with X11 and Fink paid off—I found an open-source font editor called Fontforge, which goes by the stage name “the application formerly known as PFAedit.”

GETTING PREPARED FOR FONTFORGE

Getting Fink and X11 going on the Mac today is far easier than it was in May 2003. The latest Fink installer automatically configures itself and edits the bash.profile script which resides in a user's home directory to recognize Fink installations, which are kept in a separate directory tree under /sw/bin, rather than in the standard /usr/local directory where most installers place command-line binaries that aren't part of the standard OS X package, reducing the possibility of conflicts where packages wind up overwriting each other. Fink's separate directory tree also allows easy backup, removal, and even sharing of those programs over a network, and the ability to use Fink Commander, an Aqua GUI for users who are command-line shy. The Fink project has an excellent FAQ area, a helpful document on Fink usage, and a forum for help with particular issues.


Packages: 1 Displayed, 541 Installed

Status	Name	Installed	Latest	Binary	Cate	Description
current	fondu	1:030428-1	1:030428-1	1:030428-1	utils	Convert between Mac and UNIX font formats

Fink Commander Application.

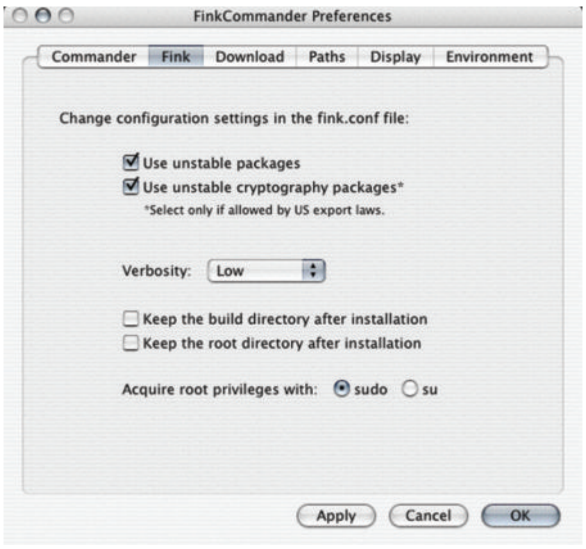
GETTING FINK AND X11

X11 requires Mac OS X 10.3, which ships with an X11 installer on install disc #3. Also needed is the X11 SDK, a separate installer on the Xcode Tools CD, or on one of the OS X 10.3 DVDs that ships with new Macs. First items to install:

- X11.app located on install disc #3 or on an installer DVD (also available as a download from HYPERLINK "http://www.apple.com/support"http://www.apple.com/support
- X11 SDK included with Apple's Xcode tools (also downloadable)
- Full install of Apple's Xcode tools (you'll need this to compile source code to get the latest Fink packages)
- The Fink 0.7.1 Binary Installer from HYPERLINK "http://fink.sourceforge.net/"http://fink.sourceforge.net, this package also includes the wonderful Fink Commander application.

First, install Xcode, X11, and the X11 SDK. Then, download and install the Fink 0.7.1 installer package. Next, copy the Fink Commander software from the Fink installer folder to the Applications folder and launch it. Go to the "Source" menu and choose "Selfupdate-cvs." Now Fink will go though and update its own binaries, as well as get the latest package descriptions.

There are two ways to install Fink software—downloading a pre-compiled binary version of each package, or letting Xcode compile the binary from the source code available through CVS (the concurrent versioning system). Fink also has two "trees" of software distribution, "stable" and "unstable." Many applications that are available in the "unstable" tree aren't available either as binaries or source code in the "stable" tree. To enable the use of the "unstable" tree, go to the "FinkCommander" menu and choose "Preferences. . ." then click on the "Fink" tab, then select the "use unstable packages" check box, then quit Fink Commander, relaunch it, and let it update the package descriptions.



Change configuration settings in the fink.conf file:

☑ Use unstable packages
☑ Use unstable cryptography packages*
*Select only if allowed by US export laws.

Verbosity: Low

☐ Keep the build directory after installation
☐ Keep the root directory after installation

Acquire root privileges with: [✓ sudo] su

Fink Commander Unstable Package Preferences

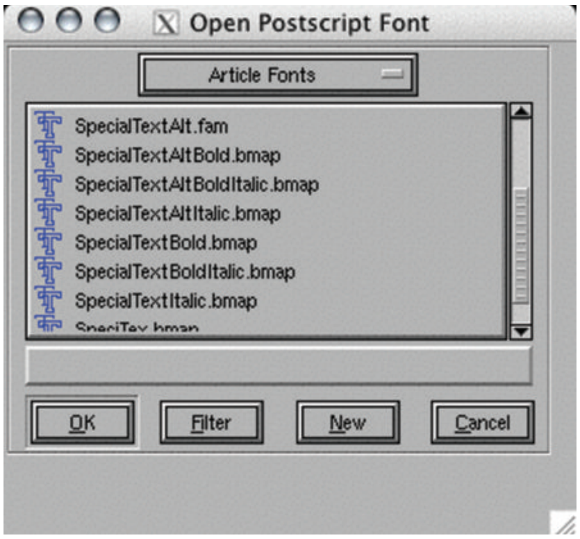
Now it's time to install Fontforge and any dependencies required to run it. If there are dependencies a prompt will come up with choices to make, and generally the first choice will work just fine. Open up FinkCommander, and type "fontforge" into the search box in the upper-right-hand corner. When the fontforge package appears, select it and either choose "install" from the "Source" menu or "install" from the "Binary" menu. Fontforge and all supporting programs will be downloaded and installed. If there's a later version that's not available as a binary, then using the source code is the only way to go, but the binary install will be considerably faster. If errors or warnings come up during the installation, consult the troubleshooting guide within the FAQ section at HYPERLINK "http://fink.sourceforge.net/"http://fink.sourceforge.net. When Fontforge is installed, repeat the process for the package "fondu."

Open up the X11 application which will be in /Applications/Utilities, and when the xterm window appears (this is the X11 equivalent to the Terminal), type fontforge and hit return. This should launch Fontforge. If X11 was already open during the install of Fontforge, it may be necessary to type the rehash command in xterm first. To avoid typing the command in xterm to launch Fontforge, or any other X11 application, customize the X11 "Applications" menu so that Fontforge points to the command /sw/bin/fontforge.

Fontforge Splash Screen

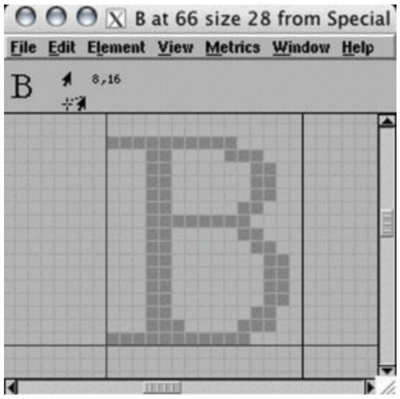
FONT EDITING/CONVERTING WITH FONTFORGE

So, I now had all of the necessary tools at my disposal to edit my client's SpecialText font, but I didn't know much about using Fontforge, so I needed to find out a few things first. Unfortunately, the first thing I found was that I was unable to open the SpecialText outline font file, but could open the SpecialText bitmap file. I had no trouble opening TrueType font outlines, dfont outlines, or OpenType outlines, but getting to the outlines (necessary to regenerate the bitmap font) for any Type 1 fonts proved elusive. Nor was I able to find any "how to" documents that described how to go about it. The outline files didn't even show up in the "Open Font" dialog that came up upon launch Fontforge:



Outline Fonts not Showing

I was, however, able to successfully open and edit a bitmap font, but that didn't get me far enough to generate a fresh copy:


B at 66 size 28 from Special
File Edit Element View Metrics Window Help
B
8,16

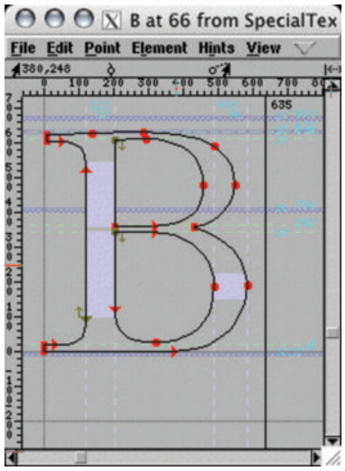
Editing a Bitmap in Fontforge

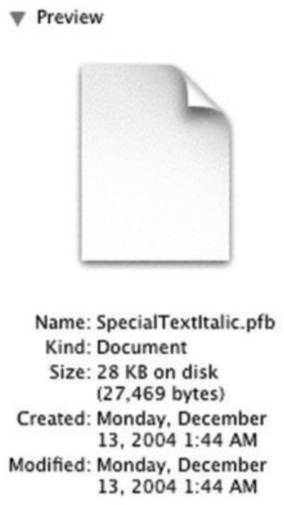
Then, I remembered that OpenOffice required a command-line tool, fondu, to convert Mac OS X fonts for use in its X Windows environment, because it couldn't access fonts with resource forks. Evidently, Fontforge couldn't open them either. So, I navigated to the folder with the Special Text fonts in the Terminal, and issued one simple command:

```
fondu *
```

Suddenly, within my SpecialText Folder, I had a bunch of files with .bdf and .pfb extensions.

Fontforge recognized these as "Postscript Font Binaries." The fondu utility had extracted the Postscript outline information from the resource fork of the outline file and deposited it in a data fork format that Fontforge could work with. The .bdf extensions are simply for identifying bitmap fonts. It is the .pfa (Postscript Font A SCII) extension that Fontforge, the application formerly known as PfaEdit, was originally named for.

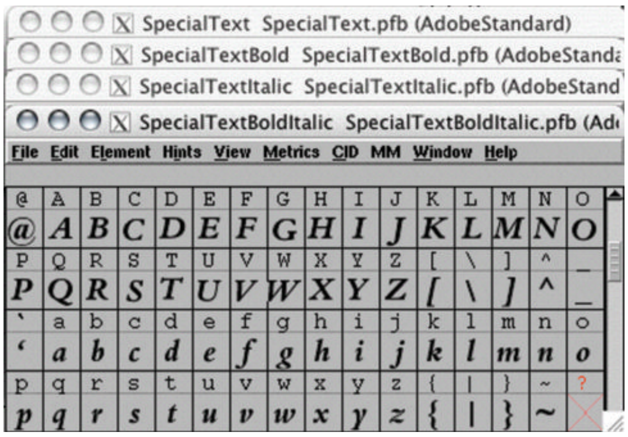




Name: SpecialTextItalic.pfb
Kind: Document
Size: 28 KB on disk (27,469 bytes)
Created: Monday, December 13, 2004 1:44 AM
Modified: Monday, December 13, 2004 1:44 AM

Outline Editor in Fontforge

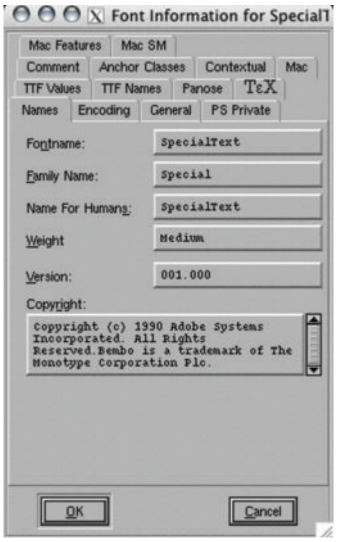
Fontforge supports exporting/converting to all known font formats, including OpenType. So, I thought it might be nice to remove the bitmap font from the equation altogether. After generating OpenType fonts for SpecialText, I was disappointed to find that QuarkXpress 6 was not OpenType-aware; diacritical marks wouldn't appear, and other essential characters, such as double curly quotes, wouldn't show up or print properly. So, I decided to regenerate the bitmap files as a Mac "family" suitcase, with a .fam extension, which required opening all of the SpecialText Type 1 outline fonts (SpecialText, SpecialTextBold, SpecialTextIta, SpecialTextBoldIta) in Fontforge simultaneously as in the picture below:


SpecialText SpecialText.pfb (AdobeStandard)
SpecialTextBold SpecialTextBold.pfb (AdobeStand
SpecialTextItalic SpecialTextItalic.pfb (AdobeStand
SpecialTextBoldItalic SpecialTextBoldItalic.pfb (Ad
File Edit Element Hints View Metrics CID MM Window Help

Preparing a Macintosh Font Family

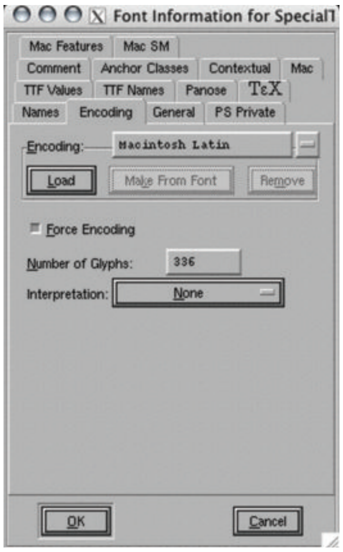
To regenerate the bitmap fonts as a Mac family in a single suitcase, there's several steps:

First, make sure that all of the open fonts have the same family information, click on the "Element" Menu and choose "Font Info. . . "



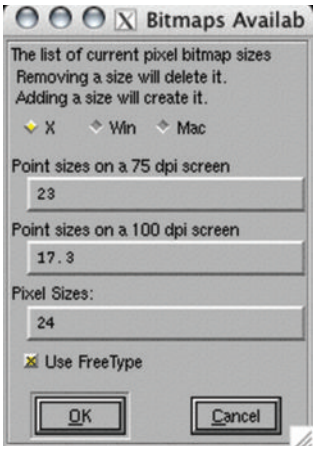
Fontname: SpecialText
Family Name: Special
Name For Humans: SpecialText
Weight: Medium
Version: 001.000
Copyright: Copyright (c) 1990 Adobe Systems Incorporated. All Rights Reserved.Bembo is a trademark of The Monotype Corporation Plc.

Next, select the "Encoding" tab and change the encoding type from "Adobe Standard" to "Macintosh Latin":



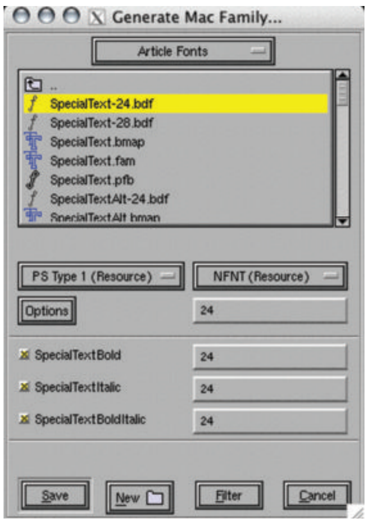
Encoding: Macintosh Latin
Force Encoding
Number of Glyphs: 336
Interpretation: None

After clicking "OK," click on the "Element" menu again and this time select "Bitmaps Available. . ." and enter any number of point sizes:



The list of current pixel bitmap sizes
Removing a size will delete it.
Adding a size will create it.
X Win Mac
Point sizes on a 75 dpi screen: 23
Point sizes on a 100 dpi screen: 17.3
Pixel Sizes: 24
Use FreeType

Now click on the "File" menu in Fontforge and select "Generate Mac Family. . ."



..
SpecialText-24.bdf
SpecialText-28.bdf
SpecialText.bmap
SpecialText.fam
SpecialText.pfb
SpecialTextAlt-24.bdf
SpecialTextAlt.bmap

PS Type 1 (Resource) NFNT (Resource) 24
SpecialTextBold 24
SpecialTextItalic 24
SpecialTextBoldItalic 24

Generate Mac Family

Be sure to select "PS Type 1 (Resource) as the outline format, and NFNT as the bitmap format. Save the suitcase file with a .suit or .fam extension, and load it up with Font Book! There's now a freshly generated bitmap font suitcase for OS X to use.

SATISFIED QUARK, SATISFIED CUSTOMER

The new bitmaps worked perfectly with Quark 6, 6.1, and now 6.5. Quark not only digested the new bitmaps for SpecialText, but seemed to unexpectedly quit a lot less often than with the old bitmaps. Although I'm not specifically against Quirk (whoops, I mean Quark)Xpress 6, I do confess that I'm more than a little annoyed at the fact that somehow it either has bugs that report older bitmap fonts as corrupt, or is missing an essential capability that all other Mac OS X software has to correctly address a bitmap font in a slightly outdated, or different NFNT format. My client is now happily using SpecialText again, and hopefully for the next ten years before another application breaks it, or puts me on the hot seat. Needless to say, I've performed this bitmap regeneration over and over for customers using Quark 6; at one site I fixed over twenty fonts, so the presses could get going again.

Fontforge, unlike some open-source equivalents to commercial software, seems every bit as capable as its counterpart Font Lab, but having never used the latter, I can't specifically say how they match up. But just browsing through the menus in Fontforge is pretty much a mind-blowing ride, revealing things about fonts in general like OpenType extended characters, hinting, kerning, encoding, and other details most of us who aren't font-designers take for granted. But more than anything for me, that weeklong font troubleshooting experience highlighted the wealth and breadth of solutions available on OS X, which serves up an eclectic feast of tools, some new and some not-so-new. So, this article's for all the designers and Mac admins who need to satisfy their font quirks. I hope it plays well in Peoria. And in New Delhi, too.

About The Author

Dean Shavit is an ACSA (Apple Certified System Administrator) who leads training sessions and manages consulting projects for MOST (Mac OS Training & Consulting) in Chicago. If you have questions or feedback you can contact him at dean@macworkshops.com.

The Web Server in OS X

What has Apple Done with Apache?

By Edward Marczak

We've all had occasion to serve up some web pages, right? Well, then you know how great Apache is and all of the flexibility and configuration options available to you, right? What? You've always used the default install? While that may be plenty powerful, we can do plenty more. Apache is so vast, that we're really only going to scratch the surface here. The great thing is that Apple has seen fit to install Apache for us, so we can skip any talk about retrieving the source, compiling and installing from source, preparing TCP/IP, and other topics that come along with most Apache material. As always, get Terminal fired up, and let's go.

History Lesson

I don't like handing out history lessons, especially when people are eager to jump into a topic. However, as a poster-child for the open source movement (even though there's no formal association with GNU or the FSF), Apache really warrants one. *Editors Note: For more on the history of Open Source, look for Dean Shavit's new Column "The Source Hound", beginning next month in MacTech!*

The roots of Apache stretch back to the NCSA's http daemon, written by Rob McCool. In 1995, it was the most popular web server on the Internet, despite being an un-maintained project for about a year. To keep NCSA's httpd going, several webmasters that had been using it contacted each other and shared patches that they had coded themselves. Two of these members, Brian Behlendorf and Cliff Skolnick took this a step further by providing a mailing list and shared file storage space for the core developers. Shortly thereafter, eight people formed the core, and the Apache Group was born.

While the Apache Group still exists with its core developers, the source code continues to be freely available. With OS X, Apple has included and integrated many projects that are open source, and tend to run on many platforms. While this naturally makes certain things easier for Apple, they really have chosen best-of-breed applications: Apache, Postfix, PostgreSQL and others. In turn, Apple has opened up the foundation of OS X, Darwin.

You'll find many reasons why open source developers do what they do, the fact that Apache is open is very important to the Internet itself. As the most widely used http server, it serves as a reference platform. Thanks to the Apache Group, and others like them, the tools of Internet publishing are available to *everyone*. The playing field is leveled, and the protocols of the World Wide Web remain 'unowned' by any one entity. This allows big business, governments and individuals to run a web server, understand the means of delivery and speak to the world.

By the way: for historical purposes, you can find the NCSA httpd page at http://hoohoo.ncsa.uiuc.edu

The Power

To paraphrase the infamous zombo-com, "You can do anything with Apache. Anything at all. The only limit is yourself." Welcome to Apache. Thanks to many factors, such as the ability to write custom modules, Apache is incredibly flexible. And with that flexibility comes the power to massage your web server into doing just about anything you see fit.

If you're into compiling Apache yourself, you're probably way ahead of this article, so feel free skip ahead to, well, the conclusion. However, since Apple has already done this for us, and included some of the more popular modules (including PHP integration), I'm not going to discuss doing so. Also, I'll be concentrating on OS X client, as OS X server has a relatively decent GUI to control Apache.

Up and Running

Before we start, let's make sure everything is in order. Open System Preferences, click on 'Sharing', and make sure 'Personal Web Sharing' is started. If it isn't, check the box. Your panel should look similar to **figure 1**.

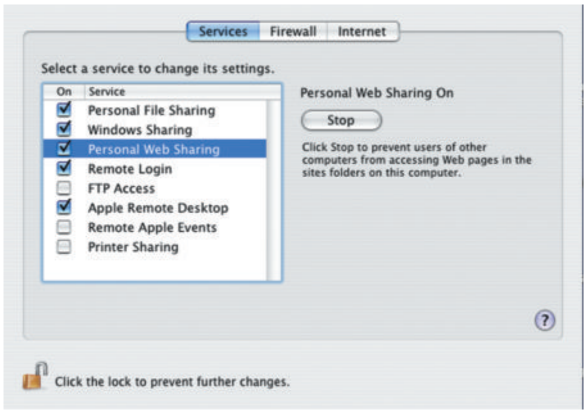
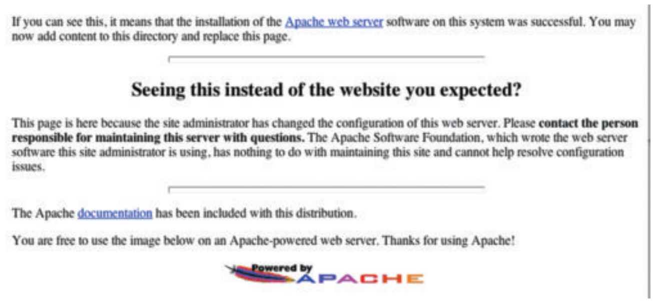

Services
Firewall
Internet
Select a service to change its settings.
On
Service
Personal File Sharing
Windows Sharing
Personal Web Sharing
Remote Login
FTP Access
Apple Remote Desktop
Remote Apple Events
Printer Sharing
Personal Web Sharing On
Stop
Click Stop to prevent users of other computers from accessing Web pages in the sites folders on this computer.
Click the lock to prevent further changes.

Figure 1 – OS X telling us that personal web sharing is enabled.

If, for some reason, you check the box, the machine thinks for a bit, but then un-checks the box for you, see the troubleshooting section next.

Once you've made sure that's running, you can connect to the web server that's now running on your own machine. Launch your web browser of choice (Firefox, Opera, Safari, the old IE-Mac, or other), and type this in the address bar, without the quotes: 'http://127.0.0.1'. If you've never touched your Apache installation, you should be looking at a screen like the one in **figure 2**.



If you can see this, it means that the installation of the Apache web server software on this system was successful. You may now add content to this directory and replace this page.

Seeing this instead of the website you expected?

This page is here because the site administrator has changed the configuration of this web server. Please **contact the person responsible for maintaining this server with questions.** The Apache Software Foundation, which wrote the web server software this site administrator is using, has nothing to do with maintaining this site and cannot help resolve configuration issues.

The Apache documentation has been included with this distribution.

You are free to use the image below on an Apache-powered web server. Thanks for using Apache!

Figure 2 – Welcome to Apache

Again, if something goes horribly awry, we'll try to help in the troubleshooting section below.

If, like me, you're a Terminal person, you can also start and stop Apache from the command line using the 'apachectl' command. 'apachectl start' starts the server, 'apachectl stop' stops the server. Something you can't do from the GUI: 'apachectl restart' will stop and start all in one fell swoop. This is great when you make configuration changes and need Apache to start using them.

Be Careful Out There

I need to preface the rest of this article by stressing the need to be careful. We're going to need to work as root to edit any of the Apache configuration files. This means two things: a) you have a good chance of mucking up your Apache installation and b) you can muck up your entire system. Until you're comfortable working as root, and with the changes that we make to Apache, *do not work on a system currently in production*. Please perform all of these changes on a test system and, well, test them. Sure, the world won't come crashing down if you muck up a web server. But if that server represents someone else's work, or livelihood, *someone's* world will come crashing down.

All the Files

When you're setting up a web server, there are basically two sets of files that you're concerned with: files that tell the web server how to do its job (config files) and files that you're trying to serve to the public (html, mainly). It's important that these files have appropriate permissions (remember those things?), as random people on the system should not be able to alter the server config or the files being served.

Nicely enough, Apple pre-configures our systems with a user named 'www' and a group named 'www'. The server starts as 'root', but then creates child processes that run as 'www'. The root process does not service requests for files at all. It simply manages all of the children. This is good from a security perspective.

Once running, the 'www' user needs to be able to read the files it is to serve to the world. So, once again, permissions must be right. So, I'll mention file permissions at the end of each section that talks about files.

Configuration Files

It's fine to simply make your sever *run*. But how can we really make it do what we want? We need to alter the configuration file that Apache reads at its startup. Apple has chosen to remain with the de facto 'standard' of keeping the config files in a sub-directory of /etc, called httpd. Get into Terminal (10 points for everyone already in Terminal), and become root. Do this either through the command "su –", which will ask for your root password, or the command "sudo bash", which will ask for your password. Once you have the root prompt (should end in a number sign '#'), you're ready. Change directory to /etc/httpd and list the contents ('ls –l'). You'll see something like this:

```
$ ls -l
total 600
-rw-r--r--   1 root  wheel  39884 16 Nov 23:29 httpd.conf
-rw-r--r--   1 root  wheel  37306 18 Nov  2003 httpd.conf.applesaved
-rw-r--r--   1 root  wheel  37047  4 Feb  2004 httpd.conf.bak
-rw-r--r--   1 root  wheel  38008  4 Feb  2004 httpd.conf.default
-rw-r--r--   1 root  wheel  33725 15 Dec  2003 httpd.conf.defaultserver
-rw-r--r--   1 root  wheel  37306 18 Nov  2003 httpd.conf.erm
-rw-r--r--   1 root  wheel  12965 15 Dec  2003 magic
-rw-r--r--   1 root  wheel  12965 15 Dec  2003 magic.default
-rw-r--r--   1 root  wheel  15150 15 Dec  2003 mime.types
-rw-r--r--   1 root  wheel  15150 15 Dec  2003 mime.types.default
drwxr-xr-x  10 root  wheel    340 18 Nov  2003 old
```

There will be some files in this listing that you do not have. Don't worry, you will after this article. The file we're after is 'httpd.conf'. In the early days of Apache, the configuration files were broken into three files: httpd.conf, srm.conf and access.conf – a holdover from its NCSA roots. Apache can still work this way, and I still maintain one or two servers like this (aside from upgrades to Apache itself, they've been running with virtually no changes to the structure since 1997).

Apache reads httpd.conf first, then srm.conf and finally access.conf. After a while, most people would forget which directives were supposed to go in which config file. Use of srm.conf and access.conf are now depreciated, and it is recommended that all directives be put into httpd.conf. So now, not only can you simply ignore the two extra files, that behavior can be completely overridden. Sometimes, though, it may be nice to break up a large configuration file into more manageable chunks (perhaps you really only want to give certain people the ability to change certain parts of the config, but not others…remember permissions?). While it's truly wonderful to have everything that affects your server in one place, it also makes for one big file to trudge through when you're new to it. Apple maintains the current recommendations and simply gives us one httpd.conf file.

Use your favorite editor (vi) and open up httpd.conf. You'll be greeted with a fair amount comments at the top of the file. Hey, look, "Based upon the NCSA server configuration files originally by Rob McCool." History! "This is the main Apache server configuration file." Yup, that's what we're after. "Do NOT simply read the instructions in here without understanding what they do. They're here only as hints or reminders. If you are unsure consult the online docs. You have been warned." Gulp. That doesn't sound too friendly.

Well, in all actuality, the default httpd.conf is *extremely* friendly. In fact, the default values have been chosen very wisely. Between the core team, and input from real, everyday Apache users, the httpd.conf file contains good, real world defaults. Now, the real world according to a web server is very different if you are "Mike's home page" or if you are amazon.com. But for people downloading the source, one can unpack, build and go in a short amount of time. Apple has basically kept all of the defaults, with some Apple-specific changes that I'll point out further on.

Scroll down a bit in the file and you'll come to "Section 1: Global Configuration". This 'section' (it's really only delimited by comments) and it's settings apply to the way the overall server runs. While I can't touch on every single parameter, I will touch on the ones important to our discussion. Anything that doesn't get mentioned should be left untouched. Let's see what these entries do:

ServerType: Can be either 'inetd' or 'standalone'. 'inetd' would apply if you're running Apache through TCP wrappers (to be addressed in a future column). Short story is this: while many, many applications do run through tcp wrappers, I've never personally seen an Apache installation that does so. Leaving this set at 'standalone' lets Apache handle all of its own requests by itself. Leave this set at 'standalone'.

ServerRoot: Here's one where Apple confounds me. 'ServerRoot' is typically where you put all of your stuff: html files, includes, and more. Apple chose '/usr' for this. Odd. In the httpd.conf that accompanies OS X Server, there's actually a note preceding this choice: "For Mac OS X Server: Changing this is OK." Now, we're safe because this gets over-ridden everywhere else by specifying absolute paths (ones that start with '/'). But '/usr' really is an odd choice, as relative paths are relative to the directory specified here.

A little further down, you'll see that the directives that would normally load srm.conf and access.conf (AccessConfig and ResourceConfig) have been commented out. Apple wants everything in one big file.

Next up is 'Server-pool Management'. Apache can be pretty intelligent about using resources. It's important that you feed it good information, though, to base its decisions on. It can dynamically adapt to the incoming request load, and then back off when the load lightens up. There are three directives that are important here: MinSpareServers, MaxSpareServers and StartServers. 'StartServers' tells the master Apache process how many child servers to start up immediately. If you have a heavily hit site, you should crank this up a bit. 'MinSpareServers' tells Apache how many spare httpd processes it should keep hanging around for that big burst of traffic. If you're a major site, you'd load this up. On the other hand, if you're setting up a server for a small intranet, you can leave this alone and let Apache dynamically allocate new servers as needed. 'MaxSpareServers' gives Apache the ceiling on how many child processes will be left hanging around, unused.

Anyone who has set up Apache on their own will notice that Apple has made the defaults a little lower that usual. In the httpd.conf we receive, we start 1 server, have 1 minimum spare, and 5 spare. The config file from the Apache Group sets these values to 5, 5 and 10, respectively. I'll venture a guess as to why Apple does this: each server process that runs sucks up resources. These resources come in the form of RAM and open files (which, have hard limits in the system). So, how many people out there installed Panther, got to the Sharing Pref Pane, saw 'Personal Web Sharing' and said, "Cool!" They then proceeded to start up Apache, only to never use it again. Apple is trying to help this person not have resources spirited away to some unknown place. I'd like to say that they also did this to keep the config that runs on OS X client nice and small, but OS X Server has the same defaults. A little low in my opinion, but at least these values can be altered through the Server Admin GUI.

MaxClients: Says what it does, does what it says. Basically, this is how many clients can access your server at once. Anyone who's been surfing the 'net for a while probably remembers trying to access a moderately popular web page, only to be greeted with, "Service Unavailable." Not having this parameter set high enough is one of the reasons you see this.

The last 4 parameters discussed are really what make a web server individual – from the configuration side, at least. You need to monitor your system, tweak, monitor some more and tweak again. At the very least, if people are getting shut out of your site, you now know one place to look.

The 'Listen' directive: Which IP address and port should the server be listening on? Apple has commented this out, which simply has Apache listen to the default of all IP addresses on the system, and port 80. You can issue several 'Listen' directives, and Apache will add the address or port to its list. A similar directive, also commented out by Apple, is 'BindAddress'. One main difference is that only one BindAddress directive is permitted. Honestly, I never use BindAddress, as you get the same functionality out of 'Listen', 'Port' and '<VirtualHost>'.

Next, you should see a grouping of 'LoadModule' and 'AddModule' directives. It would be way too much to into each one of these individually. Of course, an overview is in order. When Apache is compiled, you have the option of including support for modules that are dynamically loaded at runtime, rather than compiled in (statically). The module responsible for this is mod_so (shared object). 'LoadModule' links an object file into an Apache process at launch. 'AddModule' enables a module's use, as a module may be compiled in , but inactive. Simple rule: you need the module, you load it and you add it. Interestingly, the order that modules are added is important. Modules added later on can override the behavior of ones earlier in the list.

One last comment about the modules: As of Panther, Apple enables PHP by default. Three cheers for Apple! Prior to Panther, you had uncomment the appropriate lines from the httpd.conf file to make PHP load. Thank you Apple. I want my PHP! They even made the appropriate change to allow an index file to be 'index.php'. Nice.

On to section 2! This is often called the 'Main Server Configuration'. If you plan to serve up a single site, this is all you will need. Important stuff here:

'Port'. Which port number your server listens to. Easy.

'user' and 'group'. Important ones, for certain! As mentioned earlier, we give Apache a non-root user to run as. Many Unix systems use the user and group 'nobody', but I like 'www' even better. Many services run as 'nobody', and I want permissions as granular as possible. When we set up permissions, the user and/or group 'www' will need to have access to the files we're going to serve up.

'ServerAdmin': Is this a critical value? Well, sure, the server will run without you setting it. But the default value is 'you@your.address'. The ServerAdmin value sometimes shows up in server-generated error messages. So don't look like you haven't done your homework. This is an easy one. Set it to an appropriate e-mail address that people can send mail to if they're having problems.

'ServerName': Either change this to your machine's Fully Qualified Domain Name (FQDN), or, if this is a development machine, laptop or otherwise, leave Apple's default of '127.0.0.1'. If you're serving pages out to the Internet, you pretty much *must* have working DNS pointing to your box, with the appropriate FQDN in the ServerName directive, otherwise, relative links are going to fail. Heed the warning in the comments: "You cannot just invent host names and hope they work." Yes, you could do everything by IP address, but, do you really want to?

'DocumentRoot' is where your html files live! Apple sets this to "/Library/WebServer/Documents". Now, sure, this can really be anywhere, but as a long time Apache person, this has always felt a little odd. I actually comment this out altogether. More on that in the 'Virtual Hosts' section below.

Now, you should be up to a line that says "<Directory />". What's going on here? This is one location that we get to showcase Apache's flexibility. The brackets around the directive give it the look of HTML. If you look further down, you'll see a matching "</Directory>" tag. The httpd.conf file uses markup in just this way. There will be an opening tag, some directives that apply only to that tag, and then a closing tag. <Directory> can apply to either a URL, or, a path on the file system. In the line were examining, we 're applying some permissions to the entire site, by specifying the URL "/". For the root path "/", the options "FollowSymLinks" is on, and 'AllowOverride' is set to 'None'. These are fairly restrictive permissions (I would even ditch the ability to FollowSymLinks, but more on that later) that save you from yourself if you forget to explicitly set permissions on a directory. Obviously, it would be a chore, especially on a very large site, to have to set the permissions and options on each directory. So the permissions are hierarchical, and get inherited down the tree. So, while we have restrictive permissions at the root, we can override that for a single directory if we like. Let's say we are experimenting, and have a directory called "/Library/WebServer/Documents/experiment". We could open this up a bit with a new Directory block like this:

```
<Directory /experiment>
  Options All
  AllowOverride All
</Directory>
```

This lets us do all sorts of nasty things at http://127.0.0.1/experiment, but would still keep http://127.0.0.1/testlab with the restrictive permissions we set on "/".

Immediately following the short Directory directive, we find a larger one that applies to where our web documents live. This block creates restrictions that are a little looser than the previous block.

What has been covered so far is all you really need to get the server running, modify where it puts its files and serve some custom content like a main site (as opposed to one that lives in ~user/Sites – bah!). However, you'll notice that the httpd.conf file continues on for quite a bit! For now, I'm only going to cover three more directives. Some of which I'm going to save the detailed explanation for a later section.

Move down through the file a page or two until you see 'AccessFileName'. The default, and de-facto standard, is ".htaccess". This is an important directive that needs further explanation, and I'll get to such an explanation in its own section. Following this, you'll see 'HostNameLookups'. This directive has Apache resolve the incoming request's IP address to a name. This defaults to 'off' and should stay that way. Of course, while only you can determine what is right for your site, turning this on can cause a huge strain on both your web and DNS servers. Then again, if you're, say, a bank, and have the infrastructure, you may really have good reason to have this on. Next up are the logging directives. Toward the end of these, you'll see a line that says "CustomLog "/private/var/log/httpd/access_log" combined". Apache actually keeps two logs, a standard log and an error log. The 'CustomLog' directive tells the standard log where to store itself and in what format. For now, just be aware that it exists, and read further on to understand logging.

To edit the httpd.conf file, you've needed to be root, or some root-equivalent. If you harken back to last month's discussion of permissions, you'll see that Apple has marked the Apache configuration files as owned by root and wheel, with rights of 755. All of the files inside httpd are marked 644, also owned by root and wheel. This is exactly where you should leave these permissions set. It should be difficult to edit these files. You should have to be really conscious of what's going on in this directory. No changes should be possible with out being authorized – especially by a rogue program. Plus, you may have no choice: running DiskUtility's 'Repair Permissions' will reset these permissions as just described.

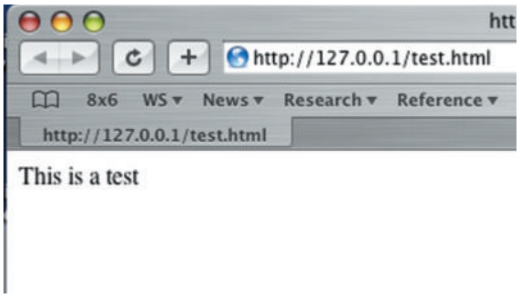
Additionally, make sure you backup your httpd.conf file! Two big reasons for this. Once you have a working version, back it up before you make any major changes. This way, you can roll back to your copy when things don't work the way you expect (or perhaps, at all). Also, Apple likes to step on httpd.conf when they update Apache through Software Update, either because of a security update or other bug fix. While they have started creating an 'httpd.conf.applesaved' file when they muck with it, I personally would trust my own backup much more. So, save early, safe often.

Files to Serve

If you're a visual person, and want to see some content, now is the time! The files that are in your DocumentRoot are meant to be served to the public. What you place here is up to you. Straight html, PHP files that access databases, javascript, text files, have fun. Of course, there are a few things you should know.

If you haven't touched the default web server directory yet, take a look in there ("/Library/WebServer/Documents"). There's a whole load of files. Instead of the familiar ".html" extension, we see files that have extensions such as ".cz", ".fr" and ".po.iso-pl". You've probably guessed that this allows Apache to serve files based on one's language preference. But how does it know? Apache calls this 'content negotiation', and is handled by the 'AddLanguage' statement in httpd.conf. You don't have to understand every facet of how this takes place to make it work nicely on your server. To find more about this feature than I can present here, see the Apache documentation on the subject at http://httpd.apache.org/docs/content-negotiation.html.

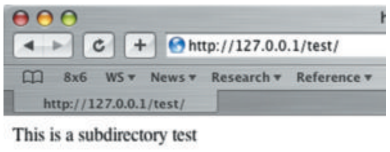
If you'd like, you can backup, move or just delete the contents of this directory (provided that you're working on your own machine and are sure you're not meddling with files someone is relying on). If you have some html docs, great. Otherwise, we'll make a simple one. Launch BBEdit, SubEthaEdit, vi, Pico, emacs – your choice of text editor. Despite the name, do not use Apple's "TextEdit". It does not save files in plain text format. C'mon, Apple! In your editor, simply type, "This is a test." If you're in the GUI, you need to be using an admin-level account to save it in the proper directory. If you're in Terminal, I'll assume you're still running as root. Save your file as "/Library/WebServer/Documents/test.html". In your web browser, type the URL http://127.0.0.1/test.html. You should see the equivalent of **figure 3**.



This is a test

Figure 3 – the working server and web page

Not the prettiest page we've ever seen, but it shows that everything is working as expected. From there, navigate to "/Library/WebServer/Documents" and create a directory called "test". Shift back to your editor, and alter your html file to say, "This is a subdirectory test" and save it as "/Library/WebServer/Documents/test/index.html". Now, in your web browser, go to the URL http://127.0.0.1/test. You'll see the equivalent of **figure 4.**



This is a subdirectory test

Figure 4 – Working with subdirectories

What exactly happened here? You've probably noticed that at most websites you visit, you haven't had to explicitly mention the file you're looking for. When you visit http://www.apple.com, you're not asking for any file in particular, are you? You're just saying, "Hey, web server! Gimme what you've got!" As mentioned earlier, Apple has set both "index.html" and "index.php" to be 'index files': if present in a directory, that file will be served up if the directory itself is asked for. So, by naming our file 'index.html' and putting it in its own directory, it will be served to the browser when that directory is asked for without asking for a specific file.

What kind of permissions do we want to have on these files? We know that the configuration files need to be locked down. However, the files we're putting in this directory are going to get served to the world via the web server. By their very definition, they're public. You'll see that Apple has them owned, again, by root and wheel, and restricted to 654. However, this is a directory that repair permissions does not touch. There is really only one absolute here, and that is that the web server must be able to read these files to serve them! That means that the user, or group, 'www' must have access. Since these files are already owned by root and wheel, you'll see that the web server is accessing them through the read attribute given to 'others'. If you had a large web development team, you could create a new group called 'webdev' and make them the group owner of the files in your DocumentRoot. This way, the people in that group could alter the contents of the web server without having a root account. What I'm saying here is that there is no hard and fast 'right-way' to set up the permissions of this directory. There is a wrong way, though, and that's to mark everything 777. I know you're tempted, but don't do it. Practice safe computing. The right way restricts things down as far as possible, while allowing everyone to do their job, including Apache itself! Do note this, though: you rarely, if ever, want anything actually owned and writable by 'www'! This way, programs that the web server executes, like cgi scripts, can't damage files they wouldn't normally be able to damage or alter otherwise.

Logging

How do you know if anyone is using your web server? How can you tell of there are any problems with content that is being served (or, not served)? How can we tell how often people are visiting, and how much bandwidth they are using? The answers to all of these questions lie in logging. After any transaction performed by Apache, it will write an entry to one of two logs: the access log (success!), or the error log (problems!).

Earlier, I pointed out the 'CustomLog' directive. In Apple's httpd.conf file, it looks like this:

```
CustomLog "/private/var/log/httpd/access_log" combined
```

We tell Apache where we want to store our log file, and in what format. Just above this, you'll see some lines that begin with 'LogFormat'. These directives describe the layout of what gets logged, plus a nickname for that format. I recommend you stick with the 'combined' format, that looks like this:

```
LogFormat "%h %l %u %t \"%r\" %>s %b \"%{Referer}i\"
\"%{User-Agent}i\"" combined
```

Whenever a transaction takes places, this line directs Apache to log the:

- %h Host. The host's IP address, or, if HostNameLookups is on, the resolved DNS name.
- %l – Remote identification. Supplied from the remote identd. You'll rarely get a name with this, and more often just see "-".
- %u – User.
- %t - Time. The time and date that the transfer completed.
- %r – What the browser asked for. Will be a string like "http://www.example.com/webpage.php".
- %>s – Success Code. Otherwise it'd be in the error log, right?
- %b – Size of the transfer in bytes.
- %{Referer}i – How was this user referred? This will either be "-" (for no referrer), or a URL, like "http://www.example.com/somepage.html". You may also have noticed this is mis-spelt by the Apache team…oops. Yes, you have to type it incorrectly to have it recognized.
- %{User-Agent}i - The User Agent – What the browser tells us it is. Safari will show "Mozilla/5.0 (Macintosh; U; PPC Mac OS X; en) AppleWebKit/125.5.5 (KHTML, like Gecko) Safari/125.11"

Why use the combined format? The default on most distributions is "common", which logs everything that "combined" does, minus the referrer and user agent. Frankly, those are nice statistics to have. You are free to come up with whatever log format you'd like. See the Apache documentation on the subject at http://httpd.apache.org/docs/logs.html, where you'll also find some other cool logging tips and tricks. However, "combined" has become a recognized logging format and many off-the-shelf log analyzers recognize it. Basically, I'm telling you to stick with combined and not change a thing.

Once Apache is running, you'll find this log in the place specified by the 'CustomLog' directive. According to Apple, that place is "/private/var/log/httpd with a name of access_log. Looks like a good place to me. You can view this log with Console.App (as found in /Applications/Utilities), or with the command line utility 'tail', or 'less'. I prefer 'tail' with the '-f' switch over Console.App, as it spits out new lines in close to real-time.

The next log to be concerned with is 'ErrorLog'. This name of this log is a bit of a misnomer. While the access log is just that, a clean list of what was accessed, good for log analyzers, the 'error' log consists of errors and general notices. For example, next is an entry from my error_log that shows Apache starting up.

[Fri Dec 3 18:38:09 2004] [notice] Apache/1.3.29 (Darwin) PHP/4.3.2 DAV/1.0.3 configured — resuming normal operations

That's not an error! In fact, it's even marked at a level of 'notice', a.k.a. 'No worries.' The lines that *are* marked 'error', however, need to be paid attention to. Let's sample an error_log from a production site (aspects changed to shield the innocent):

```
[Thu Oct 28 05:03:05 2004] [error] [client 231.50.143.44]
script not found or unable to stat: /www/httpd/cgi-
bin/formmail.pl
[Thu Oct 28 14:27:17 2004] [error] mod_ssl: SSL
handshake interrupted by system [Hint: Stop button
pressed in browser?!] (System error follows)
[Thu Oct 28 14:27:17 2004] [error] System: Connection
reset by peer (errno: 104)
[Fri Oct 29 12:17:38 2004] [error] PHP Warning:
mysql_connect(): Can't connect to local MySQL server
through socket '/www/tmp/mysql-client.sock' (2) in
/www/httpd/includes/functions.php on line 180
```

The first error is a simple file-not-found message. In fact, it's someone searching for the old formmail perl script, which was easy to exploit. The next two errors turned out to be just what the error writer guessed – a browser quit before the SSL handshake completed. The final line is what happens when PHP tries to access a MySQL database that doesn't exist – it had crashed.

Watch your logs closely. Time done so will pay off in buckets.

Virtual Hosts

Apache's virtual hosts, along with the http 1.1 specification, may be the single most important change to web serving, allowing consolidation, easier provisioning and the conservation of IP address space. Unfortunately, I talk to a lot of people who are confused by virtual hosts. If you don't think about it too much, it comes pretty easy.

The hyper-text transfer protocol version 1 specified the request and reply messages that travel between a browser and server to exchange a web page. However, once the client resolves the server name (such as www.example.com) and turns it into an IP address, the browser would simply send a GET request to that IP address. The server would then serve up the web page that was requested on that IP address. If your server had multiple IP addresses, you could run multiple versions of Apache, each bound to one of the IP addresses, that would serve up a completely separate web site (they'd have different 'DocumentRoot's).

The http 1.1 spec came along and added the fact that the browser must now pass the name of the site it's looking for in the request. Now, you could actually run one single copy of Apache with just as single IP address. Apache could serve up the correct site based on the site name in the request passed in by the browser. Let's see how this works.

Back in the httpd.conf file, toward the bottom of the file, you'll find "Section 3: Virtual Hosts". All of it is commented out. Let's change that.

Uncomment the line that says "NameVirtualHost *" (just get rid of the "#"). This turns on virtual serving for all IP addresses that Apache listens to. If you have multiple IP addresses, you could specify only one of them here if you'd like.

Next, we need to come up with a web site, separate from our main site. Create a new directory to hold this site. I like "/www", and that's what my examples will use. After you create "/www", create another subdirectory called "virt1". Inside of that directory, we're going to create two more: "htdocs" and "logs". These last two directories should sit at the same level in the hierarchy. Once that's done, create an html index file to sit in 'htdocs' directory. One line will suffice ("This is a virtual site"). Now, back to the httpd.conf file.

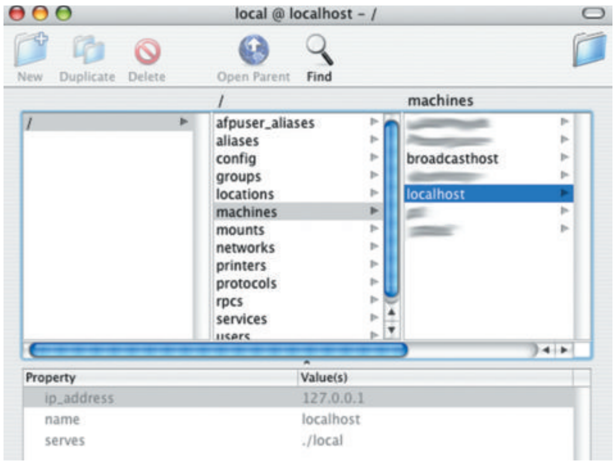
We tell Apache about a virtual site by using the "VirtualHost" opening and closing directives. Here's one for our test site, that I'll comment on after we add it to httpd.conf, right after the "NameVirtualHost" line that we just uncommented:

```
<VirtualHost *>
  ServerAdmin webmaster@virt1
  DocumentRoot /www/virt1/htdocs
  ServerName virt1
  ErrorLog /www/virt1/logs/errorlog
  CustomLog /www/virt1/logs/access_log combined
</VirtualHost>
```

You should recognize everything that went in between the opening and closing VirtualHost tags. The "ServerName" line is what Apache uses to match the request. If you were setting up a real host for use on the Internet, you'd place the server's FQDN here (such as "www.example.com"). The cool thing about a virtual host is that almost anything you can put in the main server config, you can put in a virtual host. The values you supply in the virtual host block override the main site config and apply only to that virtual host. It truly becomes it own, separately functioning site.

There are two more things we need to do to get this to work for us. First, since we're set up for *name* based virtual hosting (and not IP based hosting), we need to access the server by name. This can be achieved through DNS, or more easily on our local workstation, through altering NetInfo (which OS X consults when trying to resolve a name). DNS and name lookups will be covered in a future column. If you're not familiar with editing NetInfo or how name lookups work, just follow along.

Open up NetInfo Manager, which is found in /Applications/Utilities. Click on the lock to authenticate, and be able to make changes. Navigate down to /machines/localhost as shown in **figure 6**.



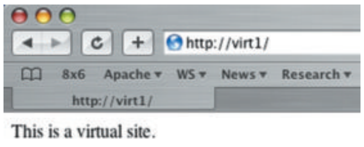
Property	Value(s)
ip_address	127.0.0.1
name	localhost
serves	./local

Figure 6 – NetInfo Manager with /machines/localhost selected, and its properties in the lower pane.

With localhost selected, click the duplicate icon in the toolbar, confirm your choice, and then click on the 'localhost copy' that you just created. In the lower pane, double-click on the 'name' property, and change the value to 'virt1'. This way, when we ask OS X to resolve the name 'virt1', it'll find the name first in NetInfo, and hand us back the IP address of '127.0.0.1', which is our own machine. Our browser, conforming to the http 1.1 specs, passes off the name of the site we're looking for to the web server.

Second, we need to restart Apache to have it read our new additions to httpd.conf. In the terminal, issue an 'apachectl restart' command, and you're ready to go. Of course, you can always restart Apache via the Sharing pref pane also.

OK – ready to test. In your web browser, enter the URL "http://virt1" and go! You'll see something like the display in **figure 7**.



This is a virtual site.

Figure 7 – The Completed Virtual Site.

To 'prove' that this is a virtual site, enter the URL "http://127.0.0.1" in your browser. You should be looking at the index file from /Library/WebServer/Documents again. Also, if you look in /www/virt1/logs, you'll see that two log files were created for this site. Again, they're completely separate from the logs stored for the main site. Bliss.

That's really it. That's not complicated, right? You can continue to add VirtualHost blocks to the httpd.conf file that serves up separate sites. Brilliant, huh? Just be aware that, before you do this, virtual hosts is another area that allows Apache to use up more resources. On a site that has many, many virtual hosts, a webmaster may choose to keep one master log file, and separate out the individual entries later on. This saves on file handles. There are more tweaks like this that can make Apache behave in a certain way. Or, you can tune your OS to raise the limits that are allowed. Either way, it's something you must be aware of.

There's a big benefit to anyone doing web development here: you can set up a virtual host for each site that you develop for that exactly matches the environment of the server you develop for. I use this on a daily basis. I develop for both Internet facing sites on ISP hosted servers, and Intranet sites that are hosted in-house and are only reached via a local LAN. But I do all of the prototyping and testing on my PowerBook. Each site that I work for has a VirtualHost block in my Apache config. This way, I can code and test before I upload the file to the real server. More often than not, the target server is running Apache on Solaris or Linux, but I can have the equivalent server on my Mac.

For more info on virtual hosts, see the Apache documentation at http://httpd.apache.org/docs/vhosts/.

htaccess

The .htaccess mechanism is very interesting. It has both its pros and cons, but lets you do things that couldn't be done without it. The filename ".htaccess" can be customized, as set by the "AccessFileName" directive. In all honesty, most of the time there is very little reason to change this. If you do change it, you need to change some other areas of your httpd.conf file to make sure you don't unwittingly serve the access file up to a client. For this discussion, I'm simply going to refer to this whole mechanism as 'htaccess'.

What is htaccess? The htaccess file is simply a text file, in which you can place Apache directives, just like a mini httpd.conf file. You can place this htaccess file in any directory or subdirectory of your site, and those directives will apply to just that directory. Now, not every directive that works in httpd.conf will be available in an htaccess file. The difference is that, unlike httpd.conf, which is read only at startup time, the file named in the AccessFileName directive is consulted each time that directory is accessed. In actuality, it's a little more complex than that.

Apache, by default, searches all directories above the one being accessed to see if an htaccess file applies. If, for example, someone requests http://virt1/testfiles/file.html that is located on the file system at /www/virt1/htdocs/testfiles, Apache will search:

```
/.htaccess
/www/.htaccess
/www/virt1/.htaccess
/www/virt1/htdocs/.htaccess
/www/virt1/htdocs/testfiles/.htaccess
```

This searching takes Apache some time. Of course, you can turn this change this functionality with an AllowOverride directive. Try this in httpd.conf:

```
<Directory />
  AllowOverride none
</Directory>
```

You can certainly be more selective about applying this.

What then, in reality would anyone use htaccess for? It comes in exceptionally handy when combined with virtual hosts, or anytime you have multiple people responsible for different content served by a single web server (like individual user pages). Speaking from experience, I once needed to set up two sites for a single company. Each site showcased a different side of the company. One of the sites was a more conservative than the other. While both sites were different, they both shared some common elements (like large QuickTime files). The goal became having two separate sites that could share this content. However, we didn't want users of one site to 'discover' the other site. The solution? Virtual hosts with the same DirectoryRoot directives. Each virtual host used a different index file. Additionally, we used .htaccess files to limit access to what each site had access to from the other site.

Also, many people simply associate htaccess with the ability to password protect directories and files. Sure, it has the ability to do that, but you can do that from httpd.conf also. Just be aware that it's not a unique property of htaccess to do this.

If you inherited some content that needed to be integrated with your site, but it perhaps came from a Windows environment where the extension '.htm' is popular, you could copy the site as is and drop it in a subdirectory of your site. We'll further pretend that other files have 'index.htm' hardcoded into them, and it would be too time-consuming to change them all before your deadline hits. Add an .htaccess file with the single line DirectoryIndex index.htm into that directory. This way, for this directory only, your web server will find index.htm as a valid index file, and hand it to a browser when the directory itself is requested.

There are also Apache directives that rewrite or redirect the requested URL. If you move a subdirectory, but still would like a reference to it (perhaps pages out of your control point there), you could drop a line like this in an htaccess file:

```
Redirect permanent /originaldirectory
http://virt1/newdirectory
```

Use of the 'permanent' keyword also returns an http 301 permanently moved code. You can also rewrite incoming URLs to add or subtract all or part of the URL. If you have a subdirectory that should always be accessed over https, you can rewrite the URL. If the directory is on our virt1 site as http://virt1/protected, drop this in the directory as an htaccess file:

```
RewriteEngine on
RewriteCondition %{SERVER_PORT} !^443$
RewriteRule ^(.*)$ https://virt1/protected/$1 [R=301,L]
```

If a browser makes a request on a port other than 443, we're going to catch that and rewrite the URL as an https:// URL.

There are many, many other possibilities. htaccess just adds to the immense flexibility of Apache.

PHP

PHP is not Apache, nor part of the distribution, but I mention it here for two reasons: one, with OS X Panther, it's included and turned on by default and, two, almost everything I touch web-wise has some PHP component so for me, the two have become inextricably linked. Of course, there are other ways to serve up dynamic content, and OS X has all the goodies you want, including perl and mod_perl (but unlike PHP, this one still has to be activated by you).

If you never plan to do anything with PHP, but want to run several virtual hosts and expect a fair amount of traffic, unload it. Just comment out the "LoadModule php4_module libexec/httpd/libphp4.so" and "AddModule mod_php4.c" lines (these are two separate lines in two different places). This will save a fair amount of memory per httpd instance.

The ability to run PHP opens up incredible new possibilities to run many of the free and open source programs that are available. Just be aware of this, though: Using the guidelines in this article and from other official sources like the Apache web site, understand how your security is impacted by these applications. Oh, I know how it starts. You set up a test web server on the network. You're testing a new open source app you've found, and it meet 90% of your needs. However, when you installed it, it asked you to make some changes to your httpd.conf and php.ini files. Perhaps even some changes to the permissions of files on disk. You think, "this is a test machine, I can make these changes without repercussion." Then it happens. You show someone at work the web app. They say it's great, and tell someone else. Before you know it, you're being asked to open up the application for use in a small department. Or the entire company. And you have a deadline. Do you go back and investigate the security of the site, or do you just get it into production?

New to PHP? See Dave Mark's Getting Started column. His current focus just happens to be PHP.

Troubleshooting

What do you do when Apache won't run? Or isn't giving you the results you're expecting? Never panic. There are a few tools we can use to investigate the problem.

First and foremost are the logs. In most cases, they are the best source to figure our what's happening. If you're receiving a message that Apache "can't bind to port…", make sure you're not trying to run two separate copies of Apache that bind to the same port. Failing that, make sure nothing else is running on that port. Use the netstat command in terminal to find out (netstat –an | grep LISTEN).

Is it plugged in? I've dealt with issues where people thought Apache was running, but it wasn't. The complaint was usually, "Apache isn't listening on the right port!" Or, "something is blocking me from getting to the web server." In many cases, people didn't realize that a syntax or other small error stopped Apache from running in the first place. Make sure it's running: use 'ps ax | grep httpd' in Terminal, or fire up Activity Monitor (make sure you select 'All Processes' from the drop down, though).

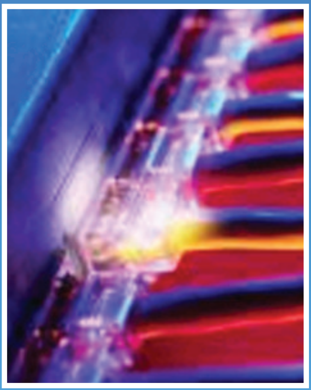
Data Banks Communications Inc

HOME OF THE FREE COMPUTER
THAT'S RIGHT FREE COMPUTER!

GET A MAC MINI OR DELL COMPUTER FREE JUST SIGN UP FOR VOIP DSL BUNDLE

INFOSPEED 768K/128K	$ 69.95
INFOSPEED 1.5M/128K	$ 69.95
INFOSPEED 1.5M/384K	$ 69.95
INFOSPEED 7.1M/768K	$ 199.95
INFOSPEED 384K/384K	$ 69.95
INFOSPEED 768K/768K	$149.95

TeleSoho 1.5M/128	$69.95
TeleSurfer Plus 608K/128K	$69.95
Residential 768K/128K	$69.95
SOHO1.5M/768K	$79.95
SOHO3.0M/384K	$79.95
SOHO3.0M/768K	$79.95
SOHO6.0M/768K	$139.95
TeleSpeed 144K/144K	$125.95
TeleSpeed 192K/192K	$125.95
TeleSpeed 384K/384K	$149.95
TeleSpeed 768K/768K	$159.95
TeleSpeed 1.1M/1.1M	$199.95
TeleSpeed 1.5M/1.5M	$249.95



Ask for business line special!

www.databanksglobal.com

For questions regarding sales or service, please call us 24 hours a day at 1-866-624-6114

Any time you make a change to your httpd.conf file, but before restarting Apache to honor the change, you can syntax check your config file. In terminal, try 'httpd –t'. This will syntax check the configuration files. It's a nice way to catch silly errors.

Virtual hosts not doing what you expect? Try 'httpd –S' (S must be capitalized). This shows the configuration as seen by Apache. You get a listing like this:

```
VirtualHost configuration:
wildcard NameVirtualHosts and _default_ servers:
*:80   is a NameVirtualHost
default server virt1 (/etc/httpd/httpd.conf:1056)
port 80 namevhost virt1 (/etc/httpd/httpd.conf:1056)
port 80 namevhost 127.0.0.1 (/etc/httpd/httpd.conf:1066)
port 80 namevhost radiotope (/etc/httpd/httpd.conf:1098)
port 80 namevhost p2 (/etc/httpd/httpd.conf:1166)
port 80 namevhost mw (/etc/httpd/httpd.conf:1177)
```

We're given the default server plus each of the virtual servers that Apache is parsing. You're also told which file and line number that Apache is finding that information from.

In the rare instance you're experiencing a hard crash, strip your httpd.conf file back down to the basics, and add your modifications in one line (or at least a small chunk) at a time. The only time I've ever seen Apache die a hard death was due to a 'third party' module being compiled or linked in. You might get some indication in the log as to what's happening before Apache dies.

Conclusion

I hope that this article has made you want to dig into Apache a little deeper. No installation is necessary, and you already own the tools to modify it to your liking. There's also a lot more to explore, as space only permits us to cover the basics here.

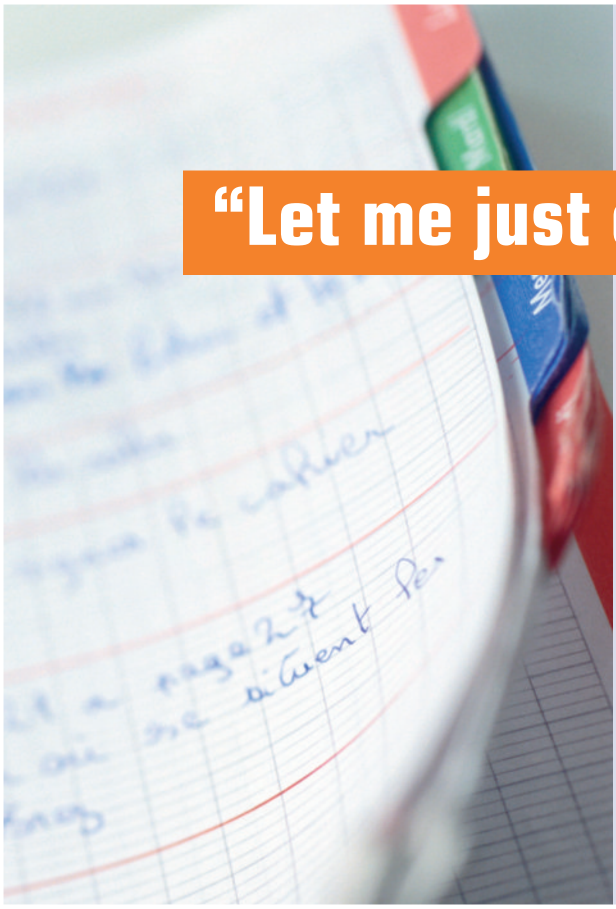
If you're serious about maintaining a web server that talks to the world, this article is a good starting point. Past this, you owe it to yourself to do three things: 1) Read the Apache documentation at http://httpd.apache.org/docs/ (yes, there's a lot there), 2) Buy the O'Reilly 'horse' book (Apache), now up to its 3rd edition and, most importantly, 3) set up a server a fiddle with it. Nothing is more important than hands on experience. Even if you have to use a cast-off G3, just get your feet wet. You'll soon be swimming.

There's a reason Apache is the number one web server on the planet: it's stable, secure, fast and flexible. Nicest of all? It's built into your Mac. Go press it into service.

MT

About The Author

Ed Marczak has been involved with technology since his Atari 2600 broke and decided to make the repairs himself. He finds the 'about the author' box the hardest part of the article to write. His technology time is often spent at http://www.radiotope.com



Connect • Communicate • Collaborate • Securely

"Let me just check my calendar..."

Kerio MailServer

A groupware alternative to Exchange that syncs calendars, contacts and email with Entourage and Outlook. Integrated anti-spam and McAfee virus filtering provide secure, junk-free email for users on any platform.

Mac OS X · Linux · Windows

www.kerio.com

McAfee SECURITY

KERIO

THINKING LOGICALLY

In AppleScript, a script can be written to logically determine a specific course of action, based on criteria that you define. For example, a backup script might be written to perform a specific backup process, based on the user that triggers the script, or based on the day of the week. While this is a fairly simple example, the point is, by adding logic to a script, the script can actually make decisions about which tasks should be performed.

If/Then Statements

Adding logic to a script is done with the addition of an if/then statement. If/then statements can range from the simple to the extremely complex.

When you write a script, you write a series of instructions for AppleScript to perform when the script is run. Each of these instructions is considered to be a statement. Simple statements are written as single lines of code. Compound statements are AppleScript statements that are written as more than one line of code. Compound statements contain other AppleScript statements, and always end with an end clause. Tell statements, repeat statements, and if/then statements would be considered compound statements.

Basic If/Then Statements

The most basic form of the if/then statement appears as follows:

```
if boolean expression then
  do something
end if
```

You can see from the example above, that a boolean expression is used as the basis for the if/then statement. In if/then statements, a boolean expression must evaluate to a value of true in order for the desired code, specified above as do something, to be executed.

In AppleScript, an expression is defined as a series of terms that evaluates to a value. For example, the following code would be considered an expression in AppleScript, and evaluates to a value of 2:

1 + 1

A boolean expression is considered to be any expression that evaluates to a true or false value.

Below is a functional example of a basic if/then statement:

```
set theOutputFolderPath to path to desktop folder
set theNewFolderName to “My Folder”
tell application “Finder”
  if (exists folder (theOutputFolderPath & theNewFolderName
as string)) = false then
    make new folder at desktop with properties
{name:theNewFolderName}
  end if
end tell
```

Last month, we explored adding different types of repeat loops to scripts, which is a very important and useful aspect of AppleScript development. As we discussed, by allowing you to perform a series of repetitive tasks without the need to duplicate code, repeat loops help to make your code less verbose, more efficient, and easier to change in the future. This month, we will focus on adding logic to your scripts, which is another important part of AppleScript development.

In the example above, the first two lines set up variables containing an output folder path and a folder name. Next, the if/then statement determines whether the folder already exists, and it triggers code to create the folder if it does not already exist.

As I said above, a boolean expression in an if/then statement must evaluate to a true value. To fully understand this, let's look at the boolean expression from our example in a little more detail. The complete boolean expression to be evaluated is the following:

```
(exists folder (theOutputFolderPath & theNewFolderName as
string)) = false
```

We can break down this boolean expression into two separate parts. In the first part, the Finder's exists command is used in order to determine whether the folder already exists, and is represented by the following code:

```
(exists folder (theOutputFolderPath & theNewFolderName as
string))
```

The second part of the boolean expression determines whether the result of the first part is equal to a value of false, and is represented by the following code:

```
= false
```

Looking at both parts of the boolean expression together again as a whole, if the result of the exists command is equal to false, then the second part of the boolean expression confirms that the first part is equal to false. Therefore, the boolean expression as a whole has been determined to be true, and the next part of the code will be executed, thus creating the folder.

Initiating a Second Course of Action

It is also possible to initiate a second course of action, should the boolean expression evaluate to a value of false. This is done through the addition of an else clause to the if/then statement. For example:

```
if boolean expression then
  do something 1
else
  do something 2
end if
```

Let's add an else clause to our example from above that creates a folder. In the following example, if the folder does not already exist, then it will be created. However, if the folder does already exist, then the user will be presented with a dialog indicating that a new folder was not created because one already exists.

```
set theOutputFolderPath to path to desktop folder
set theNewFolderName to “My Folder”
tell application “Finder”
  if (exists folder (theOutputFolderPath & theNewFolderName
as string)) = false then
    make new folder at desktop with properties
{name:theNewFolderName}
  else
    display dialog “Did not create a folder because one
already exists.”
  end if
end tell
```

Even with the use of basic if/then statements, you can already begin to see that if/then statements can provide a lot of flexibility with regard to the behavior of a script.

Combining Boolean Expressions

In certain instances, you may need to combine more than one boolean expression together in order to create a new, more complex boolean expression. This can be done with the use of the **and** or the **or** AppleScript operator. For example:

```
boolean expression and boolean expression

boolean expression or boolean expression
```

In either of the above cases, the entire expression will evaluate to a **true** or **false** value. In the first instance, each boolean expression must evaluate to a value of **true** in order for the entire expression to evaluate to a value of **true**. In the second instance, if either of the boolean expressions evaluates to a value of **true**, then the entire expression will evaluate to a value of **true**.

The following example uses a combination of two boolean expressions in order to determine whether to create a folder called *My Tuesday Folder*. One boolean expression determines whether the folder already exists. The other determines if the current date is a Tuesday. In this example, each of these boolean expressions must evaluate to a value of **true**, making the entire expression evaluate to a value of **true**, in order for the folder to be created.

```
set theOutputFolderPath to path to desktop folder
set theNewFolderName to "My Tuesday Folder"
tell application "Finder"
  if (exists folder (theOutputFolderPath & theNewFolderName
as string)) = false and (weekday of (current date)) = Tuesday
then
    make new folder at desktop with properties
{name:theNewFolderName}
  end if
end tell
```

Complex If/Then Statements

As we have seen so far, the most basic form of the if/then statement evaluates a boolean expression in order to determine whether a specific course of action should occur. An if/then statement of this nature also allows you to take an alternate course of action, if desired. However, in some cases, you may need to evaluate a boolean expression against multiple criteria, taking multiple courses of action depending on the results. This can be done by extending the **else** clause in the if/then statement to an **else if** clause. For example:

```
if boolean expression then
  do something 1
else if boolean expression then
  do something 2
end if
```

In the following example, the user is prompted, with a displayed dialog, to click a button indicating whether a folder should be created. In the dialog, the user is presented with 3 buttons – *Yes*, *No*, and *Maybe Later*. With the use of a more complex if/then statement, the script performs a different action, based on the button clicked by the user. If the user clicks the *Yes* button, then the folder is created on the desktop. If the user clicks the *No* button, then the folder is not created. If the user clicks the *Maybe Later* button, then the user is prompted to trigger the script again when ready to create the folder.

```
set theButton to button returned of (display dialog "Would
you like to create a new folder on the desktop?" buttons
{"Yes", "No", "Maybe Later"})
set theOutputFolderPath to path to desktop folder
set theNewFolderName to "My Folder"
if theButton = "Yes" then
  tell application "Finder"
    make new folder at desktop with properties
{name:theNewFolderName}
  end tell
else if theButton = "Maybe Later" then
  display dialog "Trigger the script again when you are
ready to build a folder."
end if
```

Optionally, you may still choose to include an **else** clause in this type of if/then statement, if desired. For example, we could add an **else** clause to our code above in order to display a notice to the user if the *No* button is clicked.

```
set theButton to button returned of (display dialog "Would
you like to create a new folder on the desktop?" buttons
{"Yes", "No", "Maybe Later"})
set theOutputFolderPath to path to desktop folder
set theNewFolderName to "My Folder"
if theButton = "Yes" then
  tell application "Finder"
    make new folder at desktop with properties
{name:theNewFolderName}
  end tell
else if theButton = "Maybe Later" then
  display dialog "Trigger the script again when you are
ready to build a folder."
else
  display dialog "A folder has not been created."
end if
```

Nested If/Then Statements

Another effective way to create complex if/then statements is through the nesting of if/then statements.

If you are not new to scripting, then you may be familiar with nesting already. Nesting refers to the placement of one type of AppleScript statement within another statement of the same type.

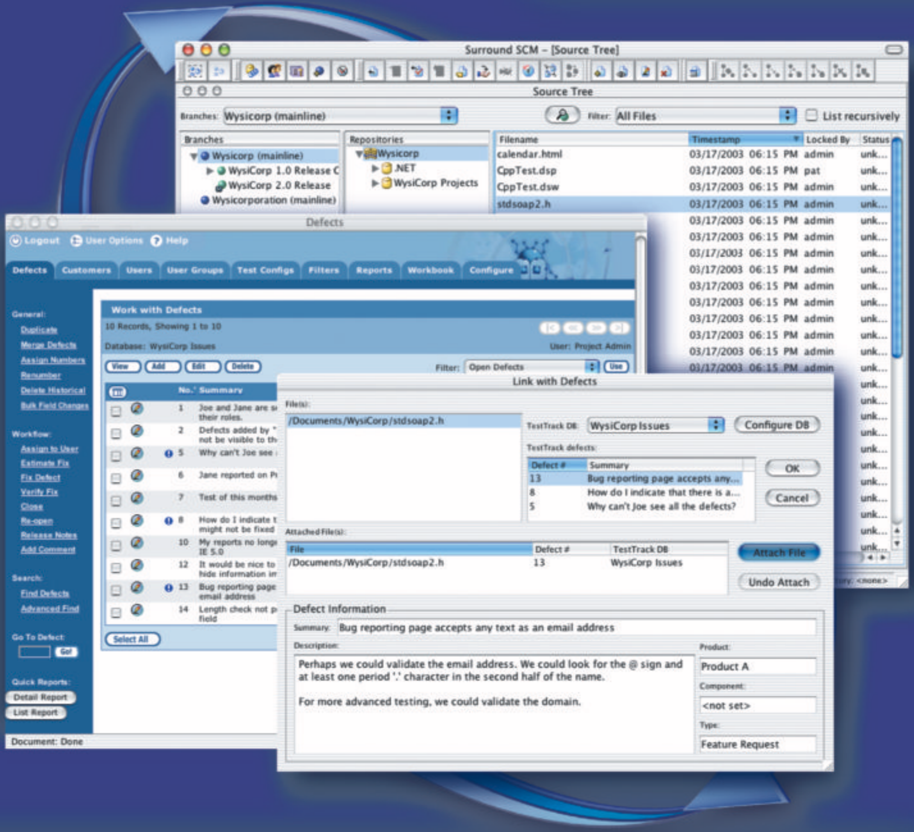
An example of a nested if/then statement's syntax would be the following:

```
if boolean expression then
  if boolean expression then
    do something 1
  end if
else if boolean expression then
    do something 2
end if
```

Looking back again to our folder creation example, the following code has been modified to include a nested if/then statement. Should the user choose to click the *Yes* button to create a new folder, a second if/then statement will be executed.

Complete Source Control and Defect Management for Mac OS X

Seapine Software™
changing the world of software development



Branches: Wysicorp (mainline) Filter: All Files ☐ List recursively

Branches:
- Wysicorp (mainline)
  - WysiCorp 1.0 Release C
  - WysiCorp 2.0 Release
- Wysicorporation (mainline)

Repositories:
- Wysicorp
  - .NET
  - WysiCorp Projects

Filename	Timestamp	Locked By	Status
calendar.html	03/17/2003 06:15 PM	admin	unk...
CppTest.dsp	03/17/2003 06:15 PM	pat	unk...
CppTest.dsw	03/17/2003 06:15 PM	admin	unk...
stdsoap2.h	03/17/2003 06:15 PM	admin	unk...
	03/17/2003 06:15 PM	admin	unk...
	03/17/2003 06:15 PM	admin	unk...
	03/17/2003 06:15 PM	admin	unk...
	03/17/2003 06:15 PM	admin	unk...
	03/17/2003 06:15 PM	admin	unk...
	03/17/2003 06:15 PM	admin	unk...
	03/17/2003 06:15 PM	admin	unk...
	03/17/2003 06:15 PM	admin	unk...
	03/17/2003 06:15 PM	admin	unk...
	03/17/2003 06:15 PM	admin	unk...
			unk...
			unk...
			unk...
			unk...
			unk...
			unk...
			unk...
			unk...
			unk...
			unk...
			unk...

Work with Defects
10 Records, Showing 1 to 10
Database: WysiCorp Issues User: Project Admin
Filter: Open Defects

No.	Summary
1	Joe and Jane are s… their roles.
2	Defects added by "… not be visible to th…
5	Why can't Joe see …
6	Jane reported on P…
7	Test of this months…
8	How do I indicate t… might not be fixed
10	My reports no longe… IE 5.0
12	It would be nice to… hide information in…
13	Bug reporting page… email address
14	Length check not p… field

Document: Done

File(s): /Documents/WysiCorp/stdsoap2.h

TestTrack DB: WysiCorp Issues

TestTrack defects:

Defect #	Summary
13	Bug reporting page accepts any...
8	How do I indicate that there is a...
5	Why can't Joe see all the defects?

Attached File(s):

File	Defect #	TestTrack DB
/Documents/WysiCorp/stdsoap2.h	13	WysiCorp Issues

Defect Information
Summary: Bug reporting page accepts any text as an email address
Description: Perhaps we could validate the email address. We could look for the @ sign and at least one period '.' character in the second half of the name.

For more advanced testing, we could validate the domain.

Product: Product A
Component: <not set>
Type: Feature Request

Effective source code control and defect tracking require powerful, flexible, and easy-to-use tools—Surround SCM and TestTrack Pro

- Complete source code control with private workspaces, automatic merging, role-based security, and more
- Comprehensive defect management — track bug reports and change requests, define workflow, customize fields
- Fast and secure remote access to your source files and defects — work from anywhere
- Advanced branching simplifies managing multiple versions of your products
- Link code changes with defects and change requests — know who changed what, when, and why
- Scalable and reliable cross-platform, client/server solutions support Mac OS X, Windows, Linux, and Solaris
- Exchange data using XML and ODBC, extend and automate with SOAP support
- Licenses priced to fit your budget

Seapine Software Product Lifecycle Management
Award winning, easy-to-use software development tools

Seapine Surround SCM
Seapine TestTrack PRO

software development 13th annual product excellence award JOLT

Download Surround SCM and TestTrack Pro at www.seapine.com or call 1-888-683-6456

This will determine whether the folder exists before creating it, taking a different course of action if it does already exist.

```
set theButton to button returned of (display dialog "Would
you like to create a new folder on the desktop?" buttons
{"Yes", "No", "Maybe Later"})
set theOutputFolderPath to path to desktop folder
set theNewFolderName to "My Folder"
if theButton = "Yes" then
  tell application "Finder"
    if (exists folder (theOutputFolderPath &
theNewFolderName as string)) = false then
      make new folder at desktop with properties
{name:theNewFolderName}
    else
      display dialog "Did not create a folder because one
already exists."
    end if
  end tell
else if theButton = "Maybe Later" then
  display dialog "Trigger the script again when you are
ready to build a folder."
else
  display dialog "A folder has not been created."
end if
```

In Closing

As you can see, if/then statements can be tremendously useful when scripting. With the use of simple and complex if/then statements, your scripts can become infinitely flexible, allowing them to take specific action based on virtually any situation that might occur during processing. In fact, without the use of if/then statements and other powerful scripting techniques, such as repeat loops, certain types of workflows could just not be automated. Imagine trying to create a complex asset management system that uses AppleScript to move files around, without the use of if/then statements or repeat loops. It would be extremely difficult, to say the least, if not impossible. So, I encourage you to begin incorporating if/then statements into your scripts in order to automate more complex workflows, and make your scripts more robust.

Until next time, keep scripting!

MT

About The Author

Benjamin Waldie is president of Automated Workflows, LLC, a firm specializing in AppleScript and workflow automation consulting. In addition to his role as a consultant, Benjamin is an evangelist of AppleScript, and can frequently be seen presenting at Macintosh User Groups, Seybold Seminars, and MacWorld. For additional information about Benjamin, please visit http://www.automatedworkflows.com, or email Benjamin at applescriptguru@mac.com.

PORTLOCK™

portlock.com

Apple®, Linux®, NetWare® and Windows® Platforms

101 North Main Street • Butte, Montana 59701 • Ph: 406-723-5200 • Fax: 406-723-5205

By Mark Underwood

Using Entourage and Mail with an Exchange Server

Making lots of headway in the "mixed" IT world

INTRODUCTION

While most people think that the best thing for the Mac was the return of Mr. Jobs, quite a bit of the people in the Corporate IT departments feel it was Microsoft's introduction of Exchange Server support in Office X and 2004. Not only did it mean conventional corporate users could access their email, shared calendars, Public folders directly from the server…it also meant they would be frowned upon less by their support gurus! In this article, we'll give the detailed instructions for those users and support staffers who want to get or provide direct Exchange services for Mac OS X.

VERSIONS, HISTORY, REQUIREMENTS AND CAVEATS

First off, remember that this integration process is a moving target for two reasons – both of them in Redmond. Microsoft is always improving on the abilities of their Exchange Server (we're now at Exchange Server 2003) and Office (Window's version is 2003 and Mac's version is 2004). While the protocols behind how an Exchange server talks to the clients that can use it hasn't changed much in the last four years, Microsoft only started this integration on the Mac side in the last two versions.

In this article, we'll refer to the three Microsoft-supported versions of Office for Mac as follows:

- 2001 – Office 2001 for Macintosh, which only runs under Mac OS 9 (not classic), and is the equivalent of Office 2000 for Windows
- X – Office X for Macintosh, which is the equivalent of Office 2002/XP for Windows
- 2004 – Office 2004 for Macintosh , which is the equivalent of Office 2003 for Windows

By "equivalent", we mean that not only are the file formats the same between sides, but with some very minor exceptions, the same named programs are functionally equivalent. Word is Word, Excel is Excel, and PowerPoint is PowerPoint. The "minor" exceptions in these named programs mean that Mac-specific nifties were added above and beyond (not instead of or replacing) the Windows-side program. Mostly this is in QuickTime, as you might guess.

Before 2001 was released, Microsoft had created a stab at a Mac-based Office quite that had Word, Excel, and PowerPoint only. They also created a reasonable Exchange client that, oddly enough, was named "Outlook". This program could fetch mail from an Exchange server and see some of the shared folders – but only if extensions to the Exchange server was configured – things not turned on by default when the server is installed. There was another stab at the Outlook equivalent, named (again, oddly enough) Outlook 2001 that was more stable, yet still not quite the Windows-side. If the IMAP or Outlook Web Access service is installed, then just about any email client can access mail – so other than it "looked" more like the windows Outlook, it didn't get any closer to being functionally equivalent.

With 2001, Microsoft dropped Outlook Express support on the Mac and introduced Entourage. Very Mac-like, very useful…but not Outlook. Two years later, just after Mac OS X started hitting the streets, X comes out. Still Mac-like, still useful…and some better Exchange support is integrated – but the IMAP gateway is still required on the server-end. Now we have 2004, and at last, it's useful.

But Apple wasn't asleep, either. The original Mail program in 10.0 was pretty much just sendmail on GUI steroids. No Exchange access except with the Outlook Web Access (OWA) gateway. Cheetah spiffed it slightly – still nothing but OWA. Jaguar added the heavy-duty filters and rules…still nothing but OWA. It took until the Panther entered the IT jungle for apple to add basic Exchange service support to their Mail program. We can expect that support to improve with both vendors' products in their next iteration.

Here's what you need to get this working properly:

Microsoft Exchange Server 2003 or later

Like all things It, the more up to date, the better. Exchange 5.5, even with the POP gateway, can only provide the mail to any Mac-based client. Exchange 2000 with SP 2 or later can provide mail, public folders, and calendaring – the basics of dealing with Exchange from the windows-user perspective. The latest version of server even allows full Active Directory integration for Exchange (as well as Mail), so that a client Mac isn't any different in usage from a Windows box.

Apple Mac OS X 10.3

Panther is best for this, on a lot of little levels and one big one. The Windows-integration inside this version is very clean and works without flinching. We'll give you the details of how to get that properly set up (as you don't usually think to do it, even in a "mixed" network). It can be done under Jaguar…but you haven't upgraded yet? Shame on you!

Microsoft Office 2004 for Macintosh

To get the best of Exchange, we highly recommend Entourage 2004, as Apple didn't get the iCal program to work with Exchange (yet). While Address Book can work with Exchange (through the LDAP gateway) and Mail with Exchange, if you must use, delegate, and share calendaring, Entourage 2004 is it. We'll show you how to use Mail and Address book, but seriously consider Entourage,

The Caveats

As with everything Microsoft-related, your mileage may vary. A full list of what does and doesn't work of the Exchange services and the Mac-based clients can be found on their Mactopia site at:

http://www.microsoft.com/mac/support.aspx?pid=exchange

SERVER CONFIGURATION

This isn't a copy of WinTech magazine, so we won't bore you with the Exchange Server installation and configuration verbosity. However, we can tell you (and should, as Microsoft's support site doesn't have all of this together) what services should be activated to make this approach work for any Mac-based, Exchange compatible client.

Outlook Web Access (or HTTP DAV as it is now know) should be already on, since that's the default in order to get Exchange email through a web browser. But double-check, and if it uses a different port than 80, note that down.

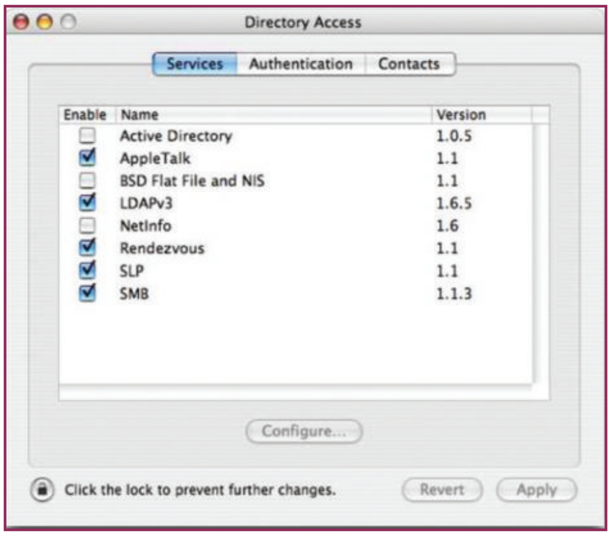
IMAP should be turned on, as this is what Apple's Mail program accesses the Exchange server with. IMAP is a normal method, too…but isn't on by default in some Exchange versions.

LDAP should be turned on. It will be on if you are using 2003, or have your 2000 server integrated with Active Directory, but again, double-check, and note the port if different than 389.

CONFIGURING MAC OS X

With the server ready to go, let's turn our attentions to the Mac. And yes, before we configure the Exchange client, we first configure the Mac to act more like it's a PC, with respect to Windows-based services.

On a Mac, windows-based services are known under the UNIX parlance name of SMB. Go into your Applications folder, then down to the Utilities folder, and start up Directory Access.



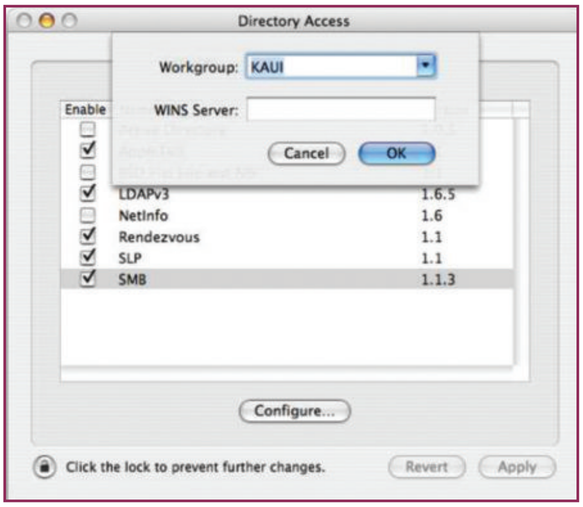
Directory Access

Services | Authentication | Contacts

Enable	Name	Version
☐	Active Directory	1.0.5
✓	AppleTalk	1.1
☐	BSD Flat File and NIS	1.1
✓	LDAPv3	1.6.5
☐	NetInfo	1.6
✓	Rendezvous	1.1
✓	SLP	1.1
✓	SMB	1.1.3

Figure 1 – Directory Access Dialog

Figure 1 shows up. You may need to unlock access to the settings by authenticating – cick the lock in the lower left if needed. Once it's available, check the SMB box, select it once, and click on the Configure button.



Directory Access

Workgroup: KAUI

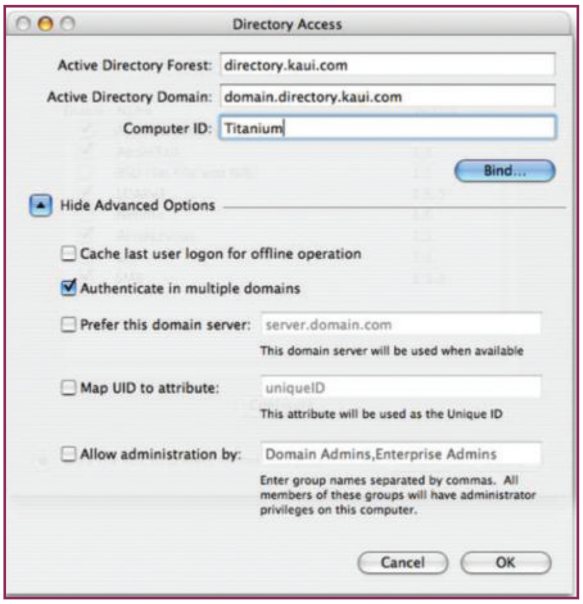
WINS Server:

Enable	Name	Version
✓	LDAPv3	1.6.5
☐	NetInfo	1.6
✓	Rendezvous	1.1
✓	SLP	1.1
✓	SMB	1.1.3

Figure 2 – SMB WINS Settings

Windows environments have two major types of authentication: WINS and Active Directory. What you want to do is to ensure reference from the Exchange server through its background authentication, as well as how the Mac will (in turn) reference to the server. If you're not the IT staffer, consult one to find out which applies…using the simple question of "are we using Active Directory?" Be warned; if you're not the IT staffer, for you to ask such a "geeky" question will astound them – since it's Microsoft-speak. But if answered "no", you use WINS. If answered "yes", you can use WINS and Active Directory…or just Active Directory, if the WINS protocol isn't used at your location.

In either case, you can (and should) specify the Workgroup the Mac is living in, with respect to the other windows machines in your local area. In our example, we have one named "KAUi". But Mac OS X is also savvy enough to help you see which one you're in…pull down the menu at the end of the Workgroup field and you'll see all of the workgroups within your LAN or IP sub-domain. Usually the name of the workgroup is known and called something like your department or company's name. If WINS is being used, it normally means you're using Windows NT Server and Exchange 5.5. Look at **Figure 2**, which is what you'll get when you clicked on "Configure" for SMB. Put the WINS Server's name in the next field. If you're not using WINS, leave it blank. Click "OK" to take these settings, and we'll move on.



Directory Access

Active Directory Forest: directory.kaui.com

Active Directory Domain: domain.directory.kaui.com

Computer ID: Titanium

☐ Cache last user logon for offline operation

✓ Authenticate in multiple domains

☐ Prefer this domain server: server.domain.com

This domain server will be used when available

☐ Map UID to attribute: uniqueID

This attribute will be used as the Unique ID

☐ Allow administration by: Domain Admins,Enterprise Admins

Enter group names separated by commas. All members of these groups will have administrator privileges on this computer.

Figure 3 – Active Directory Configuration

If you got back "yes" from your question on Active Directory, check it in the Directory Access dialog, then select it and click "Configure". You'll see a shorter version of **Figure 3**…extend it by clicking on the triangle next to Show Advanced Options. What to fill in here depends on the way your location's Active Directory was set up, so you can safely take these names to your IT staffers and get what to put in them. Usually only the top three are needed, unless there are more than one AD in the place.

Once you have those answers to fill in, you will need to click on "Bind…" to add the Mac to the Active Directory list of clients. This will throw up a prompt to log in with as an administrator, so you'd better have one handy.

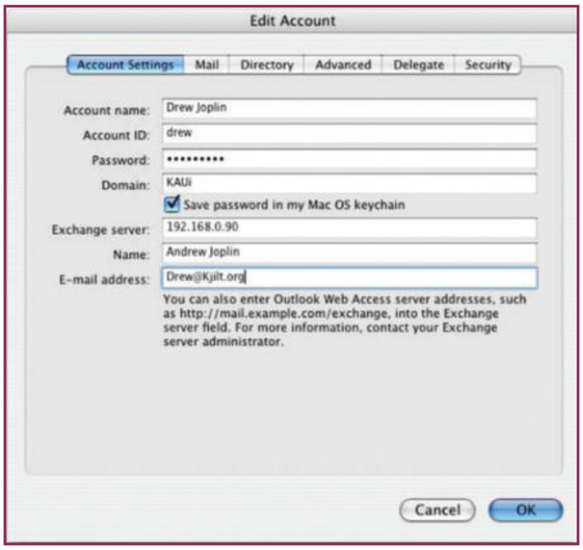
Restart the Mac once both of these are configured to re-register the computer in everything that matters for the rest of the steps.

Now we turn to creating the Exchange accounts in the two mail programs. We'll do Entourage first

CONFIGURING ENTOURAGE 2004

Open Entourage 2004, select Accounts from the Tools menu, and then pull down the "New" menu in the dialog and pick "Exchange…" This starts the Account Setup Assistant, which will try and automatically detect your Exchange server settings if you provide the basics: User ID, Domain, and Password. If they're all correct…and you don't have oddities on the Exchange server with respect to security and such, it will breeze on through and set up your account.

Due to the known and unknown security errors and problems with Exchange, however, most servers will not be so easy. Let's switch instead to the "Configure account manually" button back at the start (if the auto-assistant failed in any way to set the account up) and review the settings.


Edit Account
Account Settings | Mail | Directory | Advanced | Delegate | Security
Cancel
OK

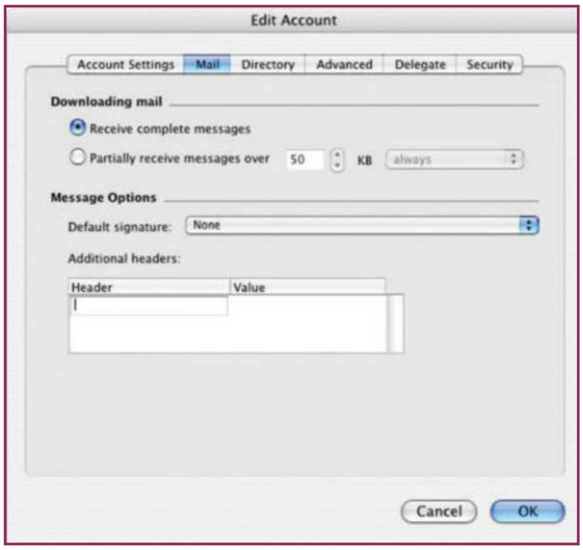
Account name: Drew Joplin
Account ID: drew
Password: •••••••••
Domain: KAUi
[✓] Save password in my Mac OS keychain
Exchange server: 192.168.0.90
Name: Andrew Joplin
E-mail address: Drew@Kjilt.org

You can also enter Outlook Web Access server addresses, such as http://mail.example.com/exchange, into the Exchange server field. For more information, contact your Exchange server administrator.

Figure 4 – Entourage Exchange Account Settings Tab

On this tab, give your account a name, and put in the basics of the server and network. In our example, we used a domain name of "KAUi" to match the Workgroup name for a 5.5/2000 Exchange server without Active Directory. If you use Active Directory, you would put in the fully qualified domain name of the Exchange server. For the name of the server itself, we've used the IP address instead of the short local name for two reasons: speed and speed. Using the short name means an address lookup to get the IP address. Multiply that times the number of end users, and you might get a delay long enough to match the timeout value and get nothing. If the Exchange server speaks to the outside world (which it should), then it has a fixed IP address and you should use that. Name and E-mail address are what you want Entourage to show to the outside world through your sent messages.

Now let's look at the Mail tab.


Edit Account
Account Settings | Mail | Directory | Advanced | Delegate | Security
Header
Value
Cancel
OK

Downloading mail
(●) Receive complete messages
( ) Partially receive messages over 50 KB always

Message Options
Default signature: None
Additional headers:

Figure 5 – Entourage Exchange Account Mail Tab

Whoever said it's cheaper to stay home, didn't get out very often.

The cost of keeping servers in-house can far exceed the cost of server outsourcing.

That's where XservHosting comes in. With Xserve Colocation services, you can afford cost-effective, flexible, and highly reliable network and internet services, freeing up more of your time to take care of business. Don't stay home managing your servers – call us today at **949-480-9701** or visit our website at **xservhosting.com**.

Starting at only $130.00 per month, Xserve Colocation includes:

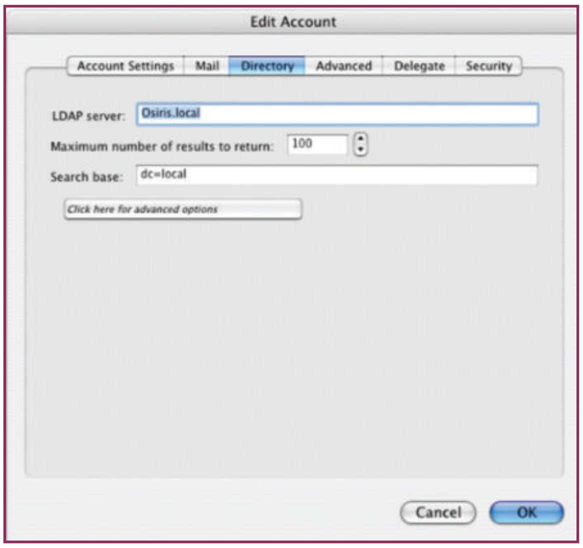
- *Free* Setup for a savings of $50.00*
- Rackspace and Power
- 1Mbit connection burstable to 100 Mbits with unlimited data transfer.

* Please mention code MTM0305.

XSH XSERVHOSTING
powering the xserve revolution
Xserve Colocation | Hosting | QuickTime Streaming

Critical setting here is how to fetch the messages from the server. We chose "Receive complete messages" to make sure we can go offline and still read them. Exchange is similar to a standard IMAP server that leaves messages on itself to retrieve and read. But most folks deal with how POP servers work…messages get transferred down to the client and not left on the server. So if you saw the message headers on the Exchange server and then left work, you wouldn't be able to select and read them without going on-line to complete the process. If you have a slow dial-up situation, you can check the "Partially receive…" option to cut down on the traffic and set a limit as to the size fetched. Message Options for signature and header information go here on this tab. Let's move on to the Directory tab.

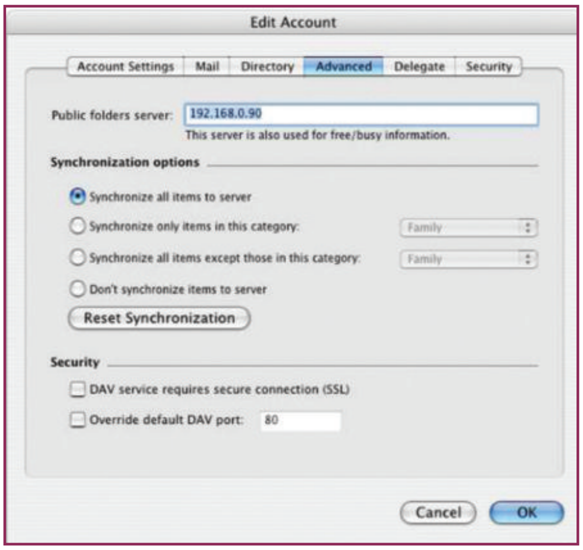
Part of the beauty of the Exchange server is centralized services, such as a single corporate address book, shared folders, and shared calendaring. On this tab, we deal with the Address Book part. If your IT staffers have set up the Exchange server with the LDAP extension, put either the name or IP address of it into this field. Name's okay here, as the lookup's different. With LDAP, there was also a search base established for how broad of a group is examined when accessed…normally "dc=local" will do, but check your settings. Trimming the number of returned names speeds things up, and I'm sure you don't have 100 John Smiths in your company, anyway. **Figure 6** shows our example for you.



LDAP server: Osiris.local
Maximum number of results to return: 100
Search base: dc=local

Figure 6 – Entourage Exchange Directory Services Tab

Next is the Advanced tab, where the spiffy Exchange services get set.



Public folders server: 192.168.0.90
This server is also used for free/busy information.

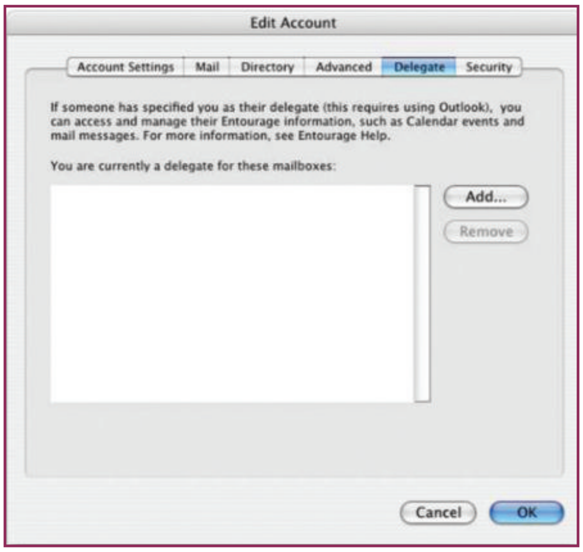
Synchronization options
● Synchronize all items to server
○ Synchronize only items in this category: Family
○ Synchronize all items except those in this category: Family
○ Don't synchronize items to server

Security
☐ DAV service requires secure connection (SSL)
☐ Override default DAV port: 80

Figure 7 – Entourage Exchange Advanced Tab

Normally, the same Exchange server is mail, calendar, public folders, and directory. We again used the fixed IP here to cut down on response time. Synchronization is best done always, hence that as the default setting. You have the option to pick a specific category within Entourage to synch, or to synch all but a specific category (such as a person category), or not to synch at all. This synch applies to everything – calendar, address book, tasks, etc. If you don't want your private life spilling into the corporate Exchange server, you don't use the same account in Entourage to do both types of mail – simply said. On to the next tab.

Delegate is an option within Exchange to allow others of your choosing to act as a 'delegate' with respect to your calendar and tasks information. Administrative assistants, for example, or co-workers on a project. You normally delegate with settings back at the server-end for starters, but within Entourage, you can refine the settings to specific individuals and responsibilities.

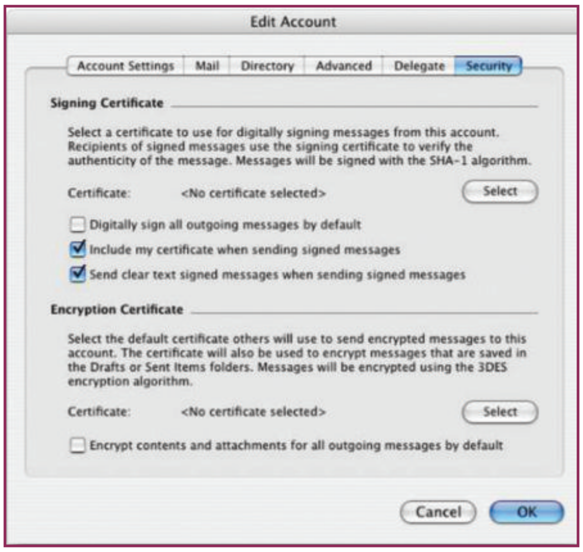


If someone has specified you as their delegate (this requires using Outlook), you can access and manage their Entourage information, such as Calendar events and mail messages. For more information, see Entourage Help.

You are currently a delegate for these mailboxes:

Figure 8 – Entourage Exchange Delegate Tab

Listed here will be the ones your Exchange server has listed for you already, and you can add others within the guidelines and roles set up there. In our example, we didn't do any delegation, as its best to test that later. On to the Security tab.



Signing Certificate

Select a certificate to use for digitally signing messages from this account. Recipients of signed messages use the signing certificate to verify the authenticity of the message. Messages will be signed with the SHA-1 algorithm.

Certificate: <No certificate selected> Select

☐ Digitally sign all outgoing messages by default

☑ Include my certificate when sending signed messages

☑ Send clear text signed messages when sending signed messages

Encryption Certificate

Select the default certificate others will use to send encrypted messages to this account. The certificate will also be used to encrypt messages that are saved in the Drafts or Sent Items folders. Messages will be encrypted using the 3DES encryption algorithm.

Certificate: <No certificate selected> Select

☐ Encrypt contents and attachments for all outgoing messages by default

Figure 9 – Entourage Exchange Security Tab

Digital certificates are often allowed in Exchange servers, as Windows server have the option to generating them for use by clients. Here is where you indicate which one(s) to use for both signing and encryption.

This completes the configuration for Entourage 2004…now you can close the edit by clicking the "OK". Entourage will find the Exchange server, and voila! You should see your left column look something like this:

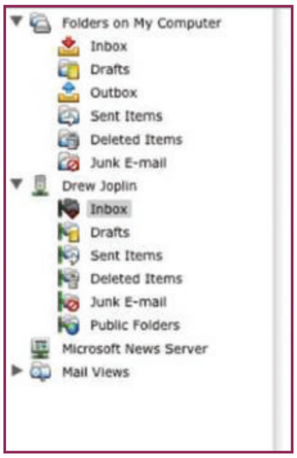
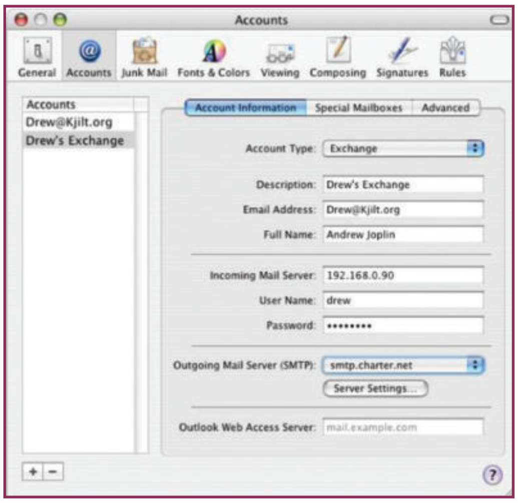


Figure 10 – Exchange Server Folders in Entourage

CONFIGURING APPLE MAIL

If you use – or what to use – Apple's Mail program to access the Exchange server, the steps are a little easier due to the fact that Apple's Mail program relies on the IMAP, LDAP, and WebDAV extensions to the server, instead of the Windows-based default services.

Open Mail, bring up the Preferences, click on Accounts, and create a new mail account to start the process with.



Account Type: Exchange

Description: Drew's Exchange

Email Address: Drew@Kjilt.org

Full Name: Andrew Joplin

Incoming Mail Server: 192.168.0.90

User Name: drew

Password: ••••••••

Outgoing Mail Server (SMTP): smtp.charter.net

Outlook Web Access Server: mail.example.com

Figure 11 – Setting up an Exchange account in Apple Mail

Pick "Exchange" from account type, provide a description for the account, then put the email address and name you want associated with the account from the outside world's perspective. Incoming server is the Exchange server – notice that we again used the fixed IP to save speed – and user name/password are those of the server's.

Our outgoing server relies back on the ISP's, since we're not really using the full Exchange suite here. Recall that many ISPs prevent you from relaying messages, so you normally use the same as any outbound POP account(s) you may have.

The last entry here is the Outlook Web Access Server, should your Exchange server be a pre-2003 version. Mail will switch to using its gateway, rather than the IMAP.

Done! Apple's Mail has been configured for Exchange. Now we turn to the Address Book approach for getting that shared resource to work with Mail.

CONFIGURING APPLE 'S ADDRESS BOOK

Since Address Book is used for many programs under Apple's default suite, it may be nice to add it anyway – even if you use Entourage. Open Address Book, go under its Preferences, click on the LDAP tab, and then the plus sign at the bottom to add a new directory server.

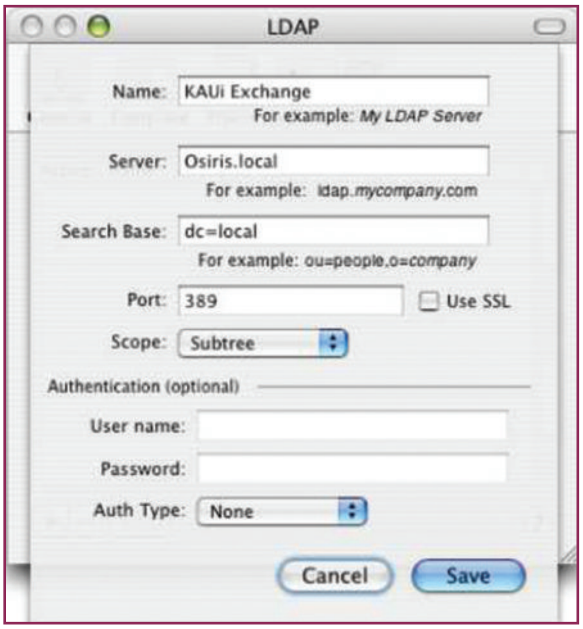

LDAP
For example: My LDAP Server
For example: ldap.mycompany.com
For example: ou=people,o=company
Cancel
Save
Name: KAUi Exchange
Server: Osiris.local
Search Base: dc=local
Port: 389 ☐ Use SSL
Scope: Subtree
Authentication (optional)
User name:
Password:
Auth Type: None

Figure 12 – Exchange through LDAP in Address Book

As we did in Entourage, LDAP settings are the same for Address Book. Server name or fixed IP address, search base, etc. are all the same. Add it, then click on "Save" to save it.

To use the LDAP, exit the Preferences, go to the top of the first list on the lft and click on the icon labeled "Directories". Next to it should now show up your Exchange server via LDAP. Click on it to restrict searches to it, and then type in a name or location in the search field at the top right of the dialog, then press return. Any results will be listed in the bottom pane. Voila!

CONCLUSIONS

<Whew!> Well, whether you went the all-Apple or all-Microsoft approach to adding Exchange services to your Mac – or to the Macs you support if you're an IT staffer – you see that it's much easier than it was just a couple of years ago…and it can only get better as time goes on. We hear rumors of abilities being added to Tiger's versions of Mail and Address Book, as well as more surprises from Redmond in the next wave of Exchange. But this advancement makes it a lot easier for the "mixed" world to let both cats and dogs eat at the same diner indefinitely. ?

While it's not a perfect fit – there are some Exchange services still only for Windows-based clients, but they are nominally not the day-to-day sorts of things – the benefits of migration in either direction by introducing this compatible approach far outweigh the 'single platform' mantra folks have had to hear for decades. Let's all hope the tide has indeed turned, and the folks at both locations keep this alive and flourishing.

MT

About The Author

Mark Underwood was born in Lexington, Kentucky and started writing not too long after that event, cutting his first set of teeth on Robert Heinlein and J.R.R. Tolkien. He currently lives near the Atlanta area with his family, three cats, and close to two dozen computers that are currently combined in solving the children's homework and the amazing mysteries of the IT industry.

Backup! Backup! Backup!

Data Security for the Extremely Paranoid

By Brad Belyeu, Oklahoma City, OK

Welcome

It's similar to earthquake, fire, or (where I'm from) tornado drills. You hope you never have to use it, but you do it in case of emergencies. Such is the case with data backups. A major problem with backing up data is that it changes so frequently. A backup is only as good as it is current. Applying Murphy's Law to a backup situation means that disaster is going to strike when you haven't ran a backup for some time. When is that last time you've backed up your data?

Why should I backup?

Hopefully you've never had to deal with data loss, but if you have I'm sure that you understand the necessity of backing up your data. In my experience, most businesses are so dependent on computer data that they would go out of business if they lost it all overnight. Imagine working on a programming project for work or school over the last several months and then losing it all and starting over. To make modern an old adage, "A Megabyte in time, saves nine."

Here are the most common reasons you should keep a current backup:

- The most overlooked reason why you should frequently backup, is accidental deletion or corruption. What if you're working hard on a file and your Mac shuts off, and you reboot only to find that the file you were working with will no longer open because it is corrupted. You or someone else using your machine could accidentally delete the file or make changes to it that you want to undo. There are numerous scenarios that could effect the security of your data.

- Hard drives crash! This is something we all have to live with. Hard drives are basically read/write heads hovering a fraction of an inch over the top of platters that are spinning 7,200 times/minute all inside an airtight seal. A small speck of dust would be detrimental! Having professionally serviced Macintosh computers for years, I know that hard drives are one of the most common components to fail.

- If you're a notebook user, your Mac is a prime target for theft. Mac iBooks and PowerBooks have a high resell value. If you like to carry yours around, you should make sure you have critical files backed up in a safe place.

- Natural disaster/fire seems rather unlikely, but it does happen. Insurance might pay for you to get a new computer, but they can't recreate your data for you if you didn't backup. In this case, not only do you need to backup, but also you need to backup your data to a different physical location.

- Viruses aren't currently a problem for OS X users but could be in the future. Viruses have been known to delete files or corrupt entire drives. This shouldn't be overlooked as a good reason to have a recent backup.

MEDIA + SOFTWARE

With current technologies, there are a variety of ways to backup your data. Not all backup options are created equal though. I'll rate each option on capacity, dependability, speed, & price.

- Zip disks are slowly fading into antiquity. I say slowly because there are a lot of die-hard zip users out there, and in its day a zip disk was one of the best backup options you had. Today there are options that offer more space for less money. Internal zip drives aren't sold in many retail stores, but a 750mb zip drive can be purchased from Iomega online for $149.99. The disks are around $15 each when purchased individually. Assuming you already have the drive that's around two cents per megabyte. The 250mb zip drives are much more common and can be purchased for $100 online. The disks can be found for less than $15 per disk. The rate at which data can be stored to or retrieved from the disk is average. Zip disks can be overwritten and reused, but they are not permanent. Zip disks will go bad eventually, just ask anyone who's used them much. Zips can be stored off-site, but as with any magnetic disk, you don't want to store your disk next to any kind of a magnetic device. Overall, zip disk backups are better than no backups at all; but there are better options.

- CD burner drives are a better option than zip drives in my opinion. Creating CD backups is definitely the cheapest option available for users with moderate backup needs, and any new computer purchased will come with a recordable CD drive. 700mb CD-R discs can be purchased for a fraction of a dollar each. When a CD-R is full, you can't write over any of the files. If you use an application like Toast, multiple sessions can be written to a CD-R until the disc is full though. Normally when backing up to CDs, you'll just through away your older backups. CD-RW discs are more expensive but allow you to erase and rewrite the disc. The problem with CD-RWs is that the process of erasing and reusing them is time consuming, and there are a limited number of times a disc can be erased and reused. The speed of writing to a CD with current technology is pretty fast and retrieving data from a CD is quicker than most data backup options. CDs are a dependable backup as long as you have a good case to store them in. If you're just throwing them in your desk drawer, they scratch easily making the data on them hard to retrieve. CDs are a good option for people needing a modest backup option for specific files or folders. But someone who wants to backup an entire drive or large folders will face the inconveniences of using multiple CDs to store the data. There are better options for those looking for large-scale backups.

- DVD burner drives offer many of the same benefits as a CD backup, but they will hold over six times the amount of data that a CD will hold. A single layer DVD will store over 4GB while a dual layer will store 8GB! As of this writing, dual layer drives are available for purchase, but I haven't been able to find dual layer disks anywhere. The cost on backing up to DVD is more per disk; but considering that you can fit so much more data on a disc, it evens out to about the same per megabyte. With the costs of DVD discs falling and the speed of DVD drives improving, DVD backups are the way of the future.

- USB Thumb drives (a.k.a. flash drives, jump drives) have become a recent addition to the backup arsenal. While most people use these devices for transporting data, they can also be used for data backup. These drives are very diverse in both size and price. A 512mb drive can be purchased online for around $50. Considering you don't have to purchase disks and data can be rewritten indefinitely, thumb drives are bargain option for small-scale backups. Also because there are no moving parts the drives have a very long lifespan. The only problem with them is that because they are so small they can be easily lost or stolen. With most of the new thumb drives coming with USB 2.0 speeds, they offer a decent speed at which files can be stored and retrieved. I have a thumb drive that I use specifically to backup my QuickBooks data file. This is, of course, in addition to my other methods of backup. Every fourth time you exit QuickBooks it will ask you if you'd like to back up your data. At that point I plug in my thumb drive, click 'Yes', and navigate the menus to the mounted drive. After saving the file, I dismount the drive and put it away until next time. Thumb drives make for stylish, easy, and cheap backups. There are only two drawbacks to using thumb drives that I can see. First, the drives aren't (yet) large enough for me to backup my entire system. Secondly, it would be an easy item for someone to steal which puts my important data at high-risk.

- The best option for backing up large amounts of data is to another hard disk. If you need a large-scale backup, a secondary drive is the most affordable and time effective way to backup your data. There are several ways to do this. The drive could be setup with a similar drive to create a RAID (redundant array of indexed disks). I strongly suggest that people create a mirrored RAID instead of a striped RAID because your data is much safer. In a mirrored RAID, data is mirrored on each drive. The drives end up being an exact copy of one another. Inside your operating system they appear to function as one drive. But if one of your drives ever fails, your data is completely safe on the second drive (assuming that both your drives didn't fail simultaneously, which is highly unlikely). The trick to setting this RAID up is that the drives must be setup before you can save data or install an OS on them. If you create a RAID with a disk that already has data on it, you will lose all the data on the drive because the drive must be repartitioned. To create a RAID disk with Disk Utility you must boot to another volume, and then start up Disk Utility (normally found in the Applications/Utilities folder). With one of the disks (not partitions) selected in the left panel, you can click on the RAID tab in the right panel. From there you drag the disks you want to use to create the RAID into the appropriate fields. After the RAID is setup, the two separate disks will appear as one in both Disk Utility and in the Finder. If one of your hard disks ever fails, you'll get a message with a chance to rebuild the RAID from the remaining disk. This saves your data in the case of a hard drive failure. Some people have a second internal drive that they backup to that is not setup as a RAID. Even though I recommend RAID, there are times when running the second drive independent from the first will work. More often an external hard drive is used to backup data from the primary drive. External hard drives are somewhat delicate pieces of equipment and shouldn't be transported more than necessary. If the drive falls several feet or gets slammed around at all, its likely to fail. But if handled with care, external drives can be a very dependable backup option.

- Another excellent way to backup your data is to a remote computer. Backing up to a remote computer presents its own set of challenges, such as bandwidth and security, but generally allows data to be stored in a secure environment away from your physical location. You can also backup to another server/computer on your LAN. If you're working in an environment with many users on multiple computers, setting up a central server for data backups is a very smart solution saving both time and money. When left up to end users, people usually neglect to backup their data. But with a server, the process can be automated with a variety of software packages and protocols, and a centralized administrator can make sure everyone is backing up on a regular basis.

SOFTWARE SOLUTIONS

Of course you could just use the finder to drag and drop necessary items on a daily basis to your backup volume. But how boring is that. As programmers we believe that anything capable of being automated should be! Plus backups are far more likely to happen if we don't leave it up to people to remember daily.

- One very simple and affordable option for reliable and secure backups is a .Mac account. iBackup is a software package developed by Apple specifically for .Mac users. It can be used to backup to your iDisk which Apple recently expanded to 250mb. Apple's iDisk isn't large for a comprehensive backup, but some major items from your home folder can be backed up. I use some of the default suggestions and then added my mailboxes to the backup list. My contacts, stickies, calendars, passwords, mailboxes, some business documents all fit easily inside my allotted space. iBackup can also be used to backup to a CD, DVD, or another hard drive making it a flexible utility. Because it allows the capability to backup to another hard drive, you can actually backup to any volume mounted on your computer including network mounts. iBackup allows for you to schedule a time to perform regular backups; and as long as your computer is turned on, it will automate the entire process for you. iBackup is an excellent utility with one minor drawback- you must have a .Mac account for the software to work. The software checks in system preferences when you start it up for a valid username and password to .Mac. I believe my .Mac account is well worth the $99/year that I pay for it simply for the backup utility alone!

- There are many free options for backup software; but if you're in a business environment, you'll probably want the power and flexibility of a commercial product. The great news is there are many great off-site backup packages that work with Mac OS X. These companies normally provide you with software and server space to backup. If you have large backups, this can get rather expensive, so in a small business environment, you might consider doing an off-site backup for your most critical data along with entire system backups daily in-house. One great application that can be used for remote or local backup is Retrospect. I've only used Retrospect Express, but it is a very powerful way to backup a lot of data to any type of media as routinely and quickly as possible.

- For those of us who don't want to pay a lot for backup software but still need our entire system backed up, Mike Bombich's Carbon Copy Cloner saves the day. CCC allows you to create a bootable backup of your volume by copying it to another hard disk. You can also choose to create an image file on the target disk instead of making a clone. CCC uses the ditto command to copy the entire drive including important resource forks. CCC also allows you to synchronize the source to the target only backing up items that have changed. This can save a great deal of time and processing power. Another option in the preferences allows to you encrypt the disk image saving it from prying eyes. CCC is a great shareware application that can be downloaded from: http://www.bombich.com/software.

- FTP can be another great way to backup. I've setup a SFTP server on my PowerMac to use as a backup server for my other computers. I recommend only using Secure FTP (SFTP) when backing up over a public network because it will encrypt your data. Most importantly it encrypts your username/password so that someone using a network sniffer can't steal that valuable information and gain access to your FTP server. My FTP client of choice is Transmit because of its great interface and apple scripting ability. I use it to automatically synchronize my iBook's home folder with a home folder on my PowerMac.

Transmit can be automated with some simple applescript. Open Script Editor (found in Applications/AppleScript/Script Editor) and try typing in the following script entering the appropriate values for your configuration to backup your home folder to a remote server running SFTP.

```
— *** CONFIGURATION ***
set myServer to "169.127.0.1" — Put your server
address here
set myUsername to "MacTechReader" — Put your
username here
set myPassword to "password" — Put your password
here
set myServerPath to "/BackupHere" — This is the
path to save
— on your server
set myLocalPath to "/Users/MacTechReader" — this is
the path to backup
— *** END CONFIGURATION ***

try
  tell application "Transmit"

    make new document at before front document
    — Creates new window for use

    tell document 1
      if (connect to myServer as user myUsername
with
 password ¬
        myPassword with initial path myServerPath
with
 connection type SFTP) then
        — Tries to connect to the server with my
username and password & the path
        — specified using Secure FTP.
        if (set your stuff to myLocalPath) then
          synchronize direction upload files method
          mirror with time offset 0
        — Uses the synchronize method to upload
files and deletes files on the
        — server that are not found on the local
computer.

        else
          display dialog "Sorry. Could not set
local
                folder."
        end if
      else
        display dialog "Sorry. Could not connect to
remote
               server."
      end if
    end tell
  end tell

end try

delay 1
tell application "Transmit"
  activate
end tell
tell application "System Events"
  tell process "Transmit"
    tell window myServer
      keystroke return
    end tell
  end tell
end tell

tell application "Transmit"
  quit
end tell
```

Save this script and then you can insert it in your startup items (inside Accounts pane of System Preferences in OS 10.3). Transmit will automatically open and synchronize your files with the remote server on startup. If you don't reboot your Mac often, a crontab can be created to run the backup at a specific time of day at certain intervals using the osascript command. Cronnix is

At Small Dog Electronics, happy customers are our highest priority.

When you shop at Small Dog Electronics, you get more than just great selection and low prices; you also get personalized service from genuine Apple Professionals who take customer service very seriously. And that's a promise... **no if's, and's or but's.**

Small Dog Electronics

www.smalldog.com

Apple Specialist

1-800-511-MACS

A socially responsible business since 1996

a shareware application that gives you a GUI to work with crontabs. CronniX and its documentation can be downloaded at http://www.koch-schmidt.de/cronnix.

This crontab is set to run at 8AM the first day of every week regardless of the day of the month or which month it is. The command osascript launches an applescript file from the command line and should be followed by the path and name of your applescript.

• Another good option for a quick backup is creating your own shell script. Ditto is a powerful command for backing up to a mounted volume, but rsync is an even better option. Rsync can be used to create a backup of a file on the same disk, another volume, or a remote host. According to its man-page there are eight different ways to use rsync, and they are:

THERE ARE EIGHT DIFFERENT WAYS OF USING RSYNC. THEY ARE:

1. For copying local files. This is invoked when neither source nor destination path contains a : separator.

2. For copying from the local machine to a remote machine using a remote shell program as the transport (such as rsh or ssh). This is invoked when the destination path contains a single : separator.

3. For copying from a remote machine to the local machine using a remote shell program. This is invoked when the source contains a : separator.

4. For copying from a remote rsync server to the local machine. This is invoked when the source path contains a :: separator or a rsync:// URL.

5. For copying from the local machine to a remote rsync server. This is invoked when the destination path contains a :: separator or a rsync:// URL.

6. For copying from a remote machine using a remote shell program as the transport, using rsync server on the remote machine. This is invoked when the source path contains a :: separator and the —rsh=COMMAND (aka _-e COMMAND") option is also provided.

7. For copying from the local machine to a remote machine using a remote shell program as the transport, using rsync server on the remote machine. This is invoked when the destination path contains a :: separator and the —rsh=COMMMAND option is also provided.

8. For listing files on a remote machine. This is done the same way as rsync transfers except that you leave off the local destination.

Rsync is installed by default on Mac OS X, but if you need documentation or a download for another machine you can visit http://rsync.samba.org/features.html.

Rsync can be set to work without authentication if you are running a rsync server on the remote host. If you're not running rsync server remotely a password has to be entered to authenticate thus making an automated backup a little more difficult. Here's an example of a command you could put in a crontab specified to run at a particular time (keep in mind someone must type in the password before this will execute):

```
rsync -r /Users/myUsername/Documents
 myUserName@myRemoteHost:/myRemoteDirectory
```

The –r option uses recursion to copy an entire directory. If you'd like more information on setting up a rsync server so that a username and password do not have to be entered, read the man page for rsyncd.conf for details.

SECURITY + ENCRYPTION

If data is important enough to backup, you don't normally want just anyone to be able to read it. Precautions must be taken to make sure the data is safe from prying eyes. In a remote transfer, always be aware if the data is being sent encrypted. You can use ssh as an argument to rysnc and if a computer has remote login enabled you can encrypt your entire session. Secure FTP is encrypted during transmission, but not regular FTP. But even if you're backing up to an external hard disk, you might want to encrypt your data after it is stored in case of theft. If you use Carbon Copy Cloner you can tell it to create an image and encrypt it, otherwise you may want to use Disk Utility after the backup to create an image from your backup and encrypt it. After all the recently made-up Chinese proverb says, "Sometimes it is worse for data to fall into wrong hands than to be lost completely."

MT

About The Author

***Brad Belyeu** is the President of ABConsulting based out of Oklahoma City, OK. He is an Apple Certified Technician and a member of the Apple Consultant Network.*

By Brad Belyeu

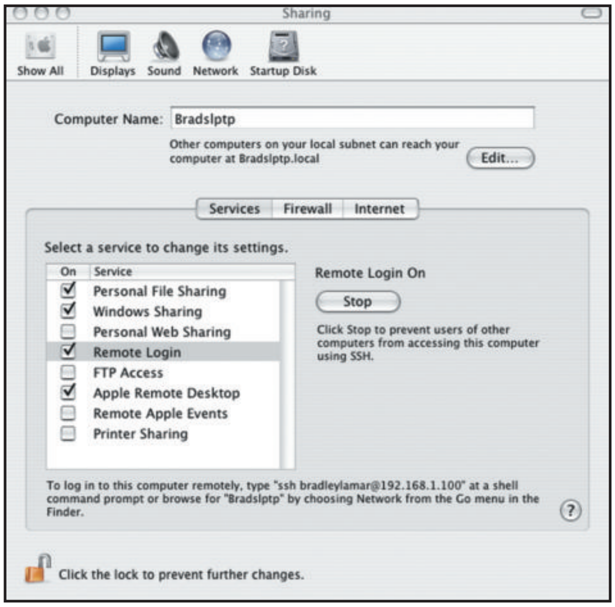
Remote Control It's Not Just For Your TV

Welcome

The purpose of having a remote control for you home entertainment center (other than a good thumb workout) is to save you the effort and time of getting out of your recliner to change channels or adjust the volume. It's a great idea, and everyone uses one. Remote control of your Mac is just as easy and convenient. Anyone who has more than one computer has valid reasons to use remote computing. Most people probably don't make use of this great functionality simply because they don't understand how it can make things so much easier for them. The advantages of a TV remote are pretty obvious, but unless we stop to think about it the advantages of remote controlling another computer are not quite as obvious. We'll take a how-to look at some of the powerful things that can be done with different forms of remote computing.

Take Command Line Control

Thanks to the Unix underpinnings of OS X, remote computing is built right into our operating system. You can enable secure command line remote login via SSH in System Preferences Sharing pane.



Computer Name: Bradslptp

Other computers on your local subnet can reach your computer at Bradslptp.local

Select a service to change its settings.

On	Service
✓	Personal File Sharing
✓	Windows Sharing
	Personal Web Sharing
✓	Remote Login
	FTP Access
✓	Apple Remote Desktop
	Remote Apple Events
	Printer Sharing

Remote Login On

Click Stop to prevent users of other computers from accessing this computer using SSH.

To log in to this computer remotely, type "ssh bradleylamar@192.168.1.100" at a shell command prompt or browse for "Bradslptp" by choosing Network from the Go menu in the Finder.

Click the lock to prevent further changes.

Secure SHell (SSH) came about due to the increasing security needs when transferring data over the Internet. Back in the early days of the Internet, a lot of data was sent as plain text. Back in the "good old days", you didn't need to worry about who was watching. Unix applications like rsh, rcp, & rlogin were perfectly acceptable for logging into a computer remotely. Now any ten year old that knows how to download and use a packet sniffer can intercept plain text transmissions. SSH provides security by creating a "tunnel" between two computers. Every packet of data sent between the computers is encrypted using an authentication key shared between the computers. If SSH is used correctly, it can make remote computing just as secure as sitting in front of the machine itself.

To login remotely to a Mac, you need to know the hostname or IP address of the machine you want to connect to. You also need to have a username and password setup on the Mac you want to login to. Open Terminal and type:

```
ssh username@remotehost
```

You will then be prompted to enter your password. After entering it correctly your current working directory becomes the home folder of the user you logged in as. To make this meaningful, lets run Software Update on another Mac on your network. To list the available updates, try:

```
sudo softwareupdate -l
```

This will list all available updates. It will give you output saying, "Software Update found the following new or updated software" after which it will list each available update. Using the –i argument will install updates. You can specify each one to be installed individually by specifying the updates name.

```
sudo softwareupdate -i name_of_update
```

You can choose to install all "required" updates by using the –r flag in place of the update name. Using the –a command will install all available updates.

Another practical application for remote login would be to fix a "lockup". If your system is stuck to the point where you can't even open Activity Monitor or Terminal, you may be able to kill the process remotely. Try using SSH to login remotely as described above, then use the command:

```
top -u
```

This will list all running applications on the remote machine. The –u argument will sort processes by processor usage. The processes at the top of the list are using the most system resources and are more likely to be locking the system up. Take a note of the PID (process ID) to the left of the application name. You can press the Q key to quit the 'top' command, then you can use the 'kill' command to terminate the offending process.

```
sudo kill process_id
```

This will hopefully kill the offending application so you can use your Mac again. If it doesn't kill the application, try using the argument –9 before the process id. The –9 flag is used to send a non-catchable, non-ignorable 'kill' command.

Let's say it's late and you've been using your PowerBook and Airport network to check your email one last time before you retire for the evening. Suddenly you remember that you left the Mac on in the study. Instead of getting out of your warm, cozy bed, just use SSH again to save the day, or night. After logging into the remote system, use the command:

```
sudo shutdown -h -now
```

The –h option halts the system and shuts it down. You can use the –r option in its place to restart the computer instead of shutting it down. The –now option can be replaced with +anynumber to shutdown the computer a certain number of minutes later.

Another great security feature of SSH is port forwarding. You can forward TCP/IP traffic through an SSH shell to secure your data over the Internet. Port forwarding can be used with FTP, HTTP, POP3, SMTP, etc; this can allow you to connect securely to any of these types of servers. The data will be sent through the SSH tunnel. It works like an encrypted subway system connecting two points. Normally the traffic goes over its respected port (80 for HTTP, for example), but when you use port forwarding it is actually sent over SSH's port (22). The syntax is:

```
ssh -L local_port:hostname:remote_port username@hostname
```

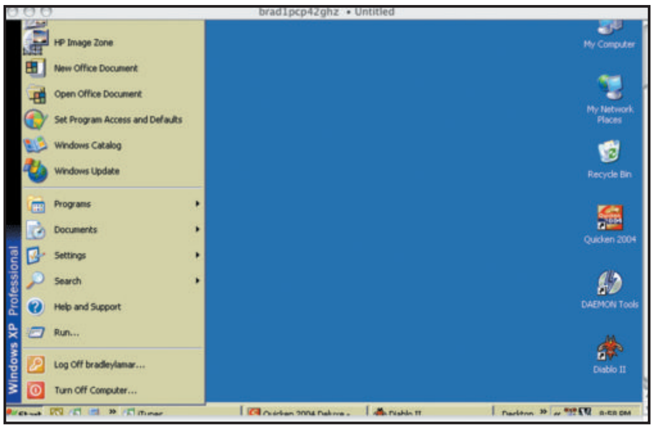
To forward your outgoing mail port, you could use the example below.

```
sudo ssh -L 25:smtp_server_name:25 username host
```

If you are running Mac OS X Server, almost any System Preference option can be set through the command line 'systemsetup' command. If you are running a copy of OS X Server view the man page for 'systemsetup' to view all available options.

VNC- Virtual Network Computing

Virtual Network Computing was developed by AT&T laboratories as an open-source cross platform graphical interface for remote desktop computing. It is currently on version 3.3.7. VNC requires a server application and client application to communicate. The server software must be running first on the computer that you want to connect to before the client software can connect to it. Major advantages of VNC include cost (its free!), small & simple file size, platform-independency, and the fact that one desktop can be shared with several computers. No state is stored at the viewer, which means if you're working on something remotely and your computer crashes or locks up nothing will be lost. It is all stored completely on the server. A good & free VNC server application for Mac OS X is OSXvnc by Redstone Software Inc. My VNC client of choice is VNCDimension, a freeware application from AT&T laboratory developers. My favorite part of VNC is that it is cross-platform. If someone has a VNC server running on a pc, it can be accessed from a Macintosh. VNC loads only the viewer application locally. All other work is processed remotely. I will never use a memory hogging virtual pc application on my Mac again. VNC can be used to run applications off your pc while saving tons of processing power on your Mac. You can initiate any program and it will run remotely. Understanding that most of the computers in the world run Microsoft Windows, it is sometimes necessary to run Windows applications. With the cost of Virtual PC starting at $249 (Win XP Home Ver 7), it is worth the extra couple hundred dollars to me to have the extra processing power and storage of actually owning a Windows pc. I run TightVNC freeware on my pc.



Imagine the extra productivity this allows! Of course, I use VNC more often on external networks than I do on my internal network. VNC allows for connections across the Internet as well. I recommend using an SSH session when connecting over an insecure network such as the Internet via port forwarding techniques explained earlier. For details on setting up VNC over SSH, see Aaron Adams article in MacTech Vol 20, No 7 2004. Other VNC clients worth trying are VNCThing and Chicken of the VNC. ShareMyDesktop is another good VNC server.

VNC is a great way to do simple remote computing but there are professional applications which are much more powerful and feature rich than most VNC clients. Let's take a look at these.

Professional Applications

Whether you professionally service computers or you're just the 'computer guy' for your friends and family, you might want to consider purchasing a professional remote desktop application such as Timbuktu. It'll save you a lot of time and money if you make frequent trips to fix small software related problems. Also if you have an Xserve you want to control without hooking up a monitor to it, remote computing applications are a great solution. If you want to share your desktop with up to fifty other Macs in a training type environment, Apple Remote Desktop would be the perfect application. Timbuktu is great for one to one computer connections to fix problems, but ARD seems to be a better application for a lab or classroom type environment.

Apple Remote Desktop can be used to sleep, wake, restart or shutdown a group of Mac OS X systems. Software packages can be installed onto multiple computers at once, and they can even be scheduled to install at times when you network will have its lightest traffic load. ARD can connect to a VNC server running on any platform. It allows you to copy files remotely, and you can also have a real-time text chat with the user sitting at the remote computer. ARD supports Unix commands being sent to remote Macs. Sharing your screen with up to fifty other Macs allows teachers or trainers to demonstrate on-screen to students. If you're using ARD to troubleshoot a problem, it allows you to get a complete hardware report of remote machines. ARD also has a software difference report, which compares installed packages on the admin computer with other computers in a group. Timbuktu has most of the same options as ARD, but it also allows for voice communications. Unfortunately, due to the cost, both of these products are frequently restricted to commercial use.

About The Author

Brad Belyeu is the President of ABConsulting based out of Oklahoma City, OK. He is an Apple Certified Technician and a certified member of the Apple Consultant Network.

DevDepot has it all!

Get More out of your Mac!

Visit our online store today for special offers and great new products.
www.devdepot.com

KEYSPAN
ONLY $39.99
Digital Media Remote

Control your computer's media programs just like your TV!

For iTunes, Windows Media Player, DVD, CD, and more!

iTripmini
ONLY $39.99
FM Transmitter

Listen in your car!

Designed exclusively for the iPod mini.

NO BATTERIES NEEDED!

iTalk
ONLY $34.99
iPod Voice Recorder

Works as a loud speaker!

Turn your iPod into a world-class voice recorder.

SightLight
FireWire light for iSight

$37.99

AC Adapter
For Powerbook and iBook laptops

$39.99

Keypoint Remote
Multimedia RF remote control

$49.99

Noise Reduction Headphones

$59.99

iMic
ONLY $34.99
USB Audio Interface

A must-have device for people who are serious about high quality audio.

Connect virtually any sound device!

FlashDrive
Cyclops 256 MB USB Flash Drive

$79.99

PowerWave
USB Audio Interface & Amplifier

$89.99

PowerMate
USB Multimedia Controller & Input Device

$36.99

iceKey
Slim USB Keboard

$44.99

TechTool Pro 4.0
for Mac OS X

The ultimate emergency software is now available for OS X!

$89.99

UNIX Utilities
for Mac OS X 3.0

Everything you need to turn your Mac into a UNIX Workstation.

$39.99

Office Applications
for Mac OS X 3.0

Packed with office and productivity apps!

$29.99

Office Applications
for Mac OS X 3.0

Bug reporting and organizing!

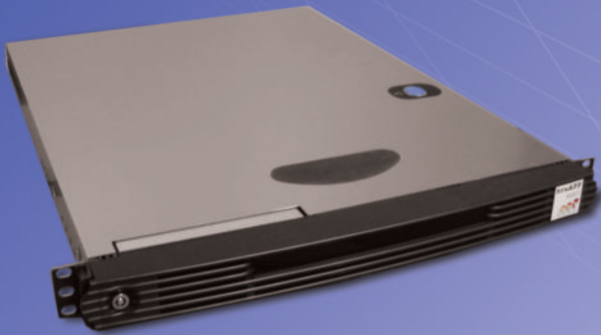
* Includes free bug reporter with each application release!

$79.99

www.devdepot.com

Introducing the bruAPP™

Plug & Play Mac OS X Network Backup System from TOLIS Group



- D2D + D2D2T Backup Appliance
- 250 GB - 4 TB disk stage NEW
- Scalable to support future network growth
- Remote management via graphical and text consoles

- Fully compatible with Xsan environments
- Support for all SCSI and Fibre-Channel tape devices and libraries
- Built in VXA-2 or LTO-2 tape drives in select models
- Client system support for all major operating systems
- BRU™— 20 years of Backup You Can Trust℠

Mac

One System, One Function, One Solid Performer

5 Minutes From Unpack to First Backup—Literally

Whether you're responsible for a few deskside systems or the latest Hollywood blockbuster, the new bruAPP provides ultra-reliable, easy to use disk-to-disk (D2D) or disk-to-disk-to-tape (D2D2T) network-based backup.

bruAPP complements TOLIS Group's BRU Server for Mac OS X and BRU LE for Mac OS X data backup and restore software products. bruAPP pricing starts at $2,999.

To learn more about the bruAPP and BRU products for Mac OS X, please call 480-505-0488 ext. 252 or visit TOLIS Group on the web at www.tolisgroup.com.

TOLIS Group™

BZFlag: A SourceForge Open Source Project

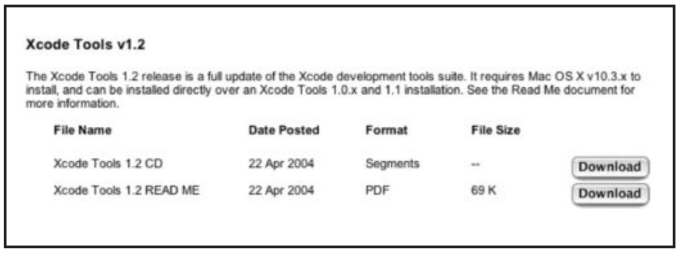
The Latest Version of Xcode

There's a new version of Xcode on the Apple Developer Connection web site. Definitely worth the download. Make your way over to:
http://connect.apple.com

Login, then click on the *Download Software* link. Figure 1 shows the section of the page dedicated to the new version of Xcode, version 1.2.

If you are working off of an unadulterated Panther install, chances are good that you won't have an easy time using Safari to download the pieces that make up the new Xcode installer. Fortunately, there's a nifty piece of freeware that will make your job much easier.

MisFox is from a German programmer named Alexander Clauss and allows you to edit all the file mappings and protocol helpers used by Mac OS X. In our case, we want to set the FTP helper app to whatever FTP client you happen to be dating at the moment. Here's the URL for MisFox:
http://www.claussnet.de/misfox/misfox.html



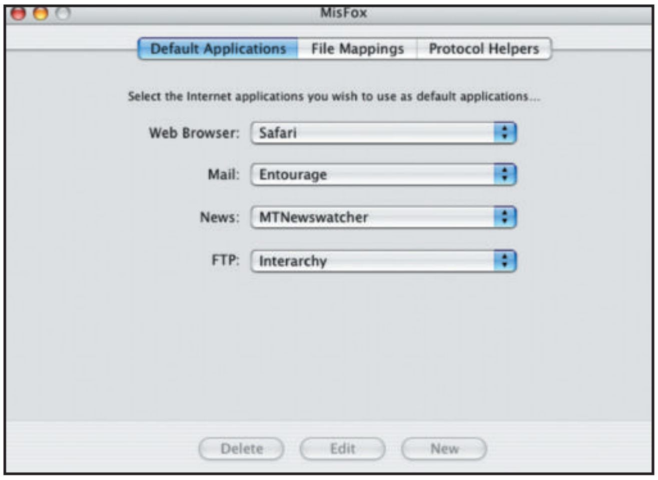
Xcode Tools v1.2

The Xcode Tools 1.2 release is a full update of the Xcode development tools suite. It requires Mac OS X v10.3.x to install, and can be installed directly over an Xcode Tools 1.0.x and 1.1 installation. See the Read Me document for more information.

File Name	Date Posted	Format	File Size	
Xcode Tools 1.2 CD	22 Apr 2004	Segments	--	Download
Xcode Tools 1.2 READ ME	22 Apr 2004	PDF	69 K	Download

Figure 1. The Xcode download listing in Safari.

When I wrote this, version 1.2.1 was the most recent release. Figure 2 shows the main MisFox window. Note that there are 3 tabs to choose from: *Default Applications*, *File Mappings*, and *Protocol Helpers*. I used the *Default Applications* tab to set my default FTP application to Interarchy.



Select the Internet applications you wish to use as default applications...

Web Browser: Safari
Mail: Entourage
News: MTNewswatcher
FTP: Interarchy

Figure 2. The MisFox Default Applications *pane.*

Lots to talk about this month. There's a new version of Xcode up on the net, I got an excellent new toy I'll touch on briefly, and for the main event, we're going to download and build an open source Mac OS X program called BZFlag.

If you go back to Figure 1, you'll see a *Download* button to the right of the *Xcode Tools 1.2 CD*. Once you use MisFox to set your default FTP client, Safari will hand the list of files behind that Download button to your client for easy downloading.

Xcode 1.2 consists of 21 segments. Download them, and if they are not automatically reassembled, just double-click on one of them. The disk image will be mounted and you can run the installer. Obviously, make sure you backup your hard drive before you do the install, just in case something goes horribly, horribly wrong.

When you first launch the new Xcode, you'll have a chance to go through the release notes to see what's new with the new version. If you miss your chance, go to the *Help* menu and select *Show Release Notes*. Lots of good info here. Still some issues to address, but lots of fixes, making this release of Xcode more stable overall.

An Awesome New Keyboard

In the March 29th issue of TidBits, Adam Engst wrote a glowing review of the Tactile Pro keyboard from Matias. Here's a picture of the keyboard:
http://halfkeyboard.com/tactilepro/viewer/tp_mainpic.html

Here's Adam's eloquent review:
http://db.tidbits.com/getbits.acgi?tbart=07607

I *live* on my PowerBook. Have a 17" Apple Studio Display. But have never found a keyboard I was really happy with. Until now. As you'll find when you read Adam's review, the designers made some excellent decisions when it came to laying out the keyboard. But to me, the most important aspect of this keyboard is its incredible responsiveness. The keys on my PowerBook keyboard are mushy. Perhaps a better way of saying this is, I can't feel the micro-switches underneath the keys when I press them. The keys on the Tactile Pro keyboard, however, clack when I press them, the reassuring clack of a micro-switch being depressed. The throw of each key is deeper, meaning that I have to press each key down further as I type. This might seem like a bad thing, but it is not. I can tell when I've pressed a key and the Tactile Pro keyboard is much more forgiving that any other keyboard I've tried. The result is much faster typing speeds with much higher accuracy.

If you've ever typed on the old Apple Extended Keyboard (think back to the days of the original Macintosh II), you'll have a sense of the Tactile Pro. They used the same mechanical keyswitches, and the result is brilliant. You can buy it online at http://tactilepro.com, $99.95.

Mac OS in an Open Source World

A long, long time ago, back before Cocoa and the second coming of Steven P. Jobs, the Mac was pretty much an island. You were either a Mac person or you were not. When it came to Mac sample code, there were some wonderful communities set up, but they were always either Apple-driven, or catered for the most part to Mac people. If you found some cool Unix/X windows code, chances are pretty good that any sample projects were set up for Windows or Unix-based environments, Mac folk need not apply.

Now that Mac OS X has passed through the steep portion of its adoption curve, there is much more of an acceptance of the Mac among the open source crowd. A perfect example of this can be seen at the big dog of development web sites, http://sourceforge.net.

Now, you might think that your favorite web site has lots of sample code and a big community of users, but no. This is big. SourceForge has more than 800,000 registered users (and that number does not include the hundreds of thousands of developers who just like to poke around) and more than 80,000 hosted projects. Those are crazy huge numbers.

In the olden days, there might have been a few scattered Mac projects, but they would have been few and far between. Navigate over to http://sourceforge.net and click on the *software map* button, just below the banner ad towards the top of the window. Here's the URL that will get you there directly:
https://sourceforge.net/softwaremap/trove_list.php

This is a useful page if you are looking for a project and know the topic. But you can also browse by other categories, including OS. Click on the *Operating System* link. Again, here's the direct link:
https://sourceforge.net/softwaremap/trove_list.php?form_cat=199

Figure 3 shows the OS categories, along with the number of projects in each category. More than three thousand Mac OS projects. This is cool!

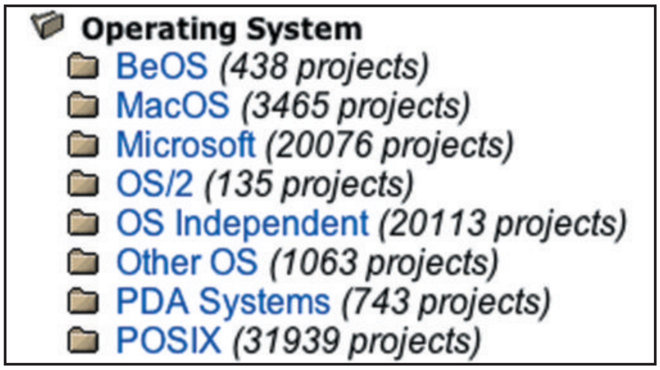

Operating System
- BeOS *(438 projects)*
- MacOS *(3465 projects)*
- Microsoft *(20076 projects)*
- OS/2 *(135 projects)*
- OS Independent *(20113 projects)*
- Other OS *(1063 projects)*
- PDA Systems *(743 projects)*
- POSIX *(31939 projects)*

Figure 3. The list of SourceForge projects, listed by OS.

If you click on Mac OS, and then on Mac OS X, you'll find 2,290 Mac OS X projects, 1,526 of which are Cocoa projects. That is a pretty rich vein of interesting material to explore. And, who knows, you might find a project that interests you so much that you'll dig in and get involved with the development process. SourceForge (and Open Source, in general) is cool that way.

It is worth noting that there are many projects that support the Mac or will just plain run on the Mac that are not necessarily in the MacOS category. For example, there are several thousand Mac-savvy projects in the *OS Independent* and *POSIX* categories.

Read through the rest of this column, then start digging around, see what you can find.

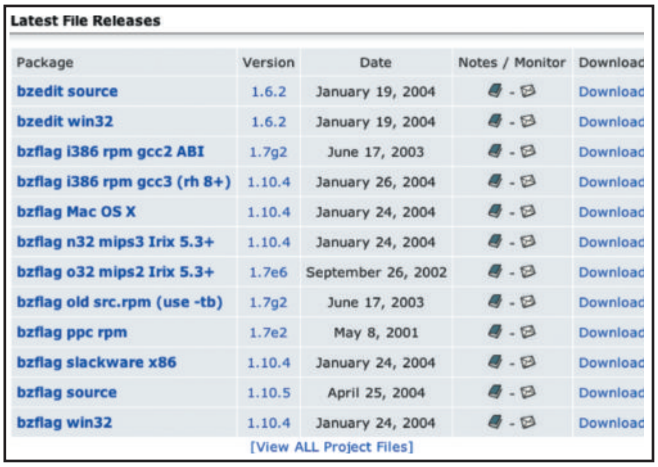
Building BZFlag

Each month, SourceForge nominates a *Project of the Month*. This past April, the project of the month was a cool tank-based shooter game named BZFlag. Here's a link to the SourceForge BZFlag page:
https://sourceforge.net/potm/potm-2004-04.php

Cool! Pictures of Tim Riker, David Trowbridge, and Sean Morrison, BZFlag's key developers. The page contains a lot of interesting info, including an interview with the principals. This is pretty typical of a "potm" (project of the month) page. The potm URLs are well constructed, so you replace the 2004-04 with a 2004-03 to get to the March project of the month:
https://sourceforge.net/potm/potm-2004-03.php

The potm page is more of a marketing page. On it, you'll find a link to the SourceForge BZFlag *Project page*:
https://sourceforge.net/projects/bzflag/

This is the actual SourceForge home for BZFlag, where you'll go to download the various sources and binaries for the different platforms BZFlag supports. These pages are *very* well organized. Figure 4 shows the listing of recent BZFlag file releases. The link at the bottom of the figure takes you to a page with a much more exhaustive listing.


Latest File Releases

Package	Version	Date	Notes / Monitor	Downloac
bzedit source	1.6.2	January 19, 2004	-	Downloac
bzedit win32	1.6.2	January 19, 2004	-	Downloac
bzflag i386 rpm gcc2 ABI	1.7g2	June 17, 2003	-	Downloac
bzflag i386 rpm gcc3 (rh 8+)	1.10.4	January 26, 2004	-	Downloac
bzflag Mac OS X	1.10.4	January 24, 2004	-	Downloac
bzflag n32 mips3 Irix 5.3+	1.10.4	January 24, 2004	-	Downloac
bzflag o32 mips2 Irix 5.3+	1.7e6	September 26, 2002	-	Downloac
bzflag old src.rpm (use -tb)	1.7g2	June 17, 2003	-	Downloac
bzflag ppc rpm	1.7e2	May 8, 2001	-	Downloac
bzflag slackware x86	1.10.4	January 24, 2004	-	Downloac
bzflag source	1.10.5	April 25, 2004	-	Downloac
bzflag win32	1.10.4	January 24, 2004	-	Downloac

[View ALL Project Files]

Figure 4. The list of recent BZFlag file releases on the project page.

Need to find something fast?

INDEX
INDEX
INDEX

With c-tree Speed.

TRY IT NOW!

FairCom's c-tree Plus®

embedded database engine offers Superior Indexing Technology – the key to performance, data integrity, and concurrency. c-tree Plus offers direct record-oriented C and C++ APIs with an industry-standard SQL interface that allows use of any combination of APIs within the same application. Furthermore, we offer source code access for intimate programming control, unmatched portability, and developer-to-developer technical support.

Heterogeneous Environments

c-tree Plus and c-treeSQL™ Servers are the perfect solution for your mixed platform environments. Mac servers to Windows clients? No problem. Linux Servers to Mac clients? No problem. c-tree has a long history of cross-platform development solutions. Byte incompatibility between platforms is handled seamlessly with our Uniformat data handling technology.

Low TCO

c-tree is priced affordably, requires minimal hardware resources, and needs no IT staff for maintenance. If Total Cost of Ownership (TCO) is important to you, c-tree is the perfect database.

Easy Deployment

c-tree Servers are designed for ease of use and deployment as well. Out of the box, our servers can be installed and running in minutes.

Start indexing your data today!

FairCom®
USA • Europe • Japan • Brazil
www.faircom.com

Go to www.faircom.com/go/mteval for a FREE evaluation of c-tree Plus!

If all we were interested in was the app itself, we could download the binary for our platform. Now BZFlag *is* a lot of fun to play, but there's something gratifying about successfully building an app on your own machine, especially an app as complex as this one.

Click on the *bzflag source* link towards the bottom of the list. Be sure to click on the one with the most recent date. As you can see, I'll be working with the one labeled *April 25, 2004*.

This link will take you to a page listing various source releases, including the one you selected. Source code is archived in a number of different ways, depending on the OS. In general, you do *not* want to download a .zip file. This will likely be a build intended for a Windows machine. The permissions will not be setup for Mac OS X, and there will also likely be carriage return/line feed confusion as well. For Mac OS X, look for a tar.bz2 or a tar.gz archive.

Click on the link labeled:

bzflag-1.10.5.20040426.tar.bz2

This will take you to a download page. Select your nearest location and click the link to start the download. StuffIt Expander will expand the archive just fine.

In the Finder, open the newly expanded directory. In my case, it was called *bzflag-1.10.5.20040426*. Now open the subdirectory *src/platform/MacOSX*. As the name implies, these are the build files for Mac OS X.

Open the Xcode project file in the *MacOSX* directory.

When the Xcode project window appears, click on the targets popup menu (it's in the upper-left corner) and select *BZFlag* (Figure 5).

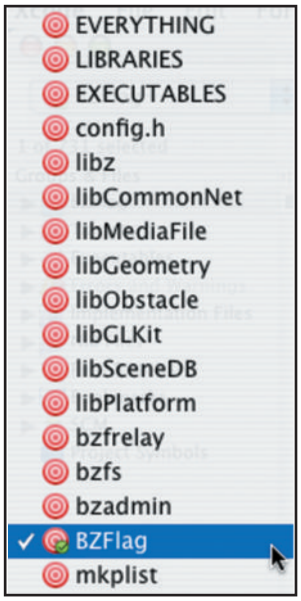
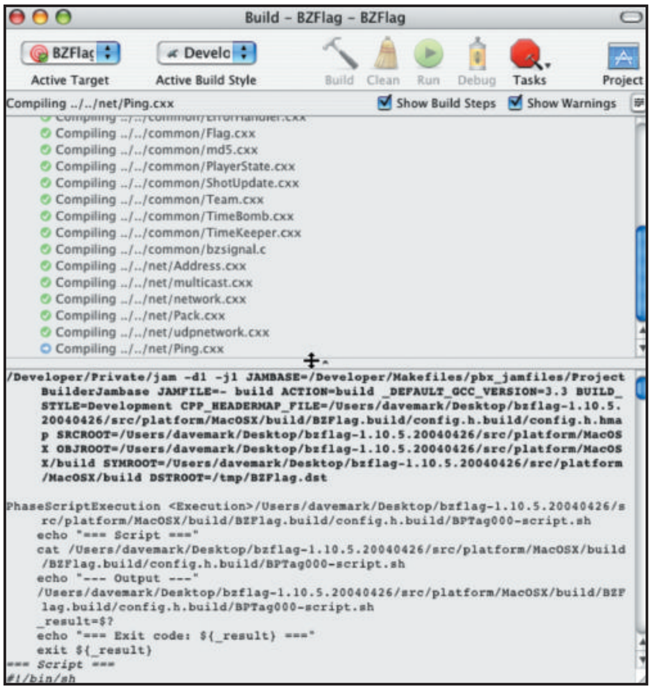


Figure 5. The BZFlag targets popup from the BZFlag project window.

Now select *Build and Run* from the *Build* menu. This is gonna take a while, so this might be a good time to alphabetize those Zamfir pan flute CDs you've been collecting.

When you start the build, if nothing appears to be happening, it may be that you have the split closed in your build window. If your build window is not divided into two distinct panes, click on the dot on the bottom edge of the window and drag it up, towards the middle of the window (Figure 6). The build is driven by a set of Makefiles (Unix build scripts) and the script steps are shown in the bottom pane, while Xcode's compile messages will be shown in the top pane.



```
Compiling ../../net/Ping.cxx
  Compiling ../../common/Flag.cxx
  Compiling ../../common/md5.cxx
  Compiling ../../common/PlayerState.cxx
  Compiling ../../common/ShotUpdate.cxx
  Compiling ../../common/Team.cxx
  Compiling ../../common/TimeBomb.cxx
  Compiling ../../common/TimeKeeper.cxx
  Compiling ../../common/bzsignal.c
  Compiling ../../net/Address.cxx
  Compiling ../../net/multicast.cxx
  Compiling ../../net/network.cxx
  Compiling ../../net/Pack.cxx
  Compiling ../../net/udpnetwork.cxx
  Compiling ../../net/Ping.cxx
```

```
/Developer/Private/jam -d1 -j1 JAMBASE=/Developer/Makefiles/pbx_jamfiles/Project
    BuilderJambase JAMFILE=- build ACTION=build _DEFAULT_GCC_VERSION=3.3 BUILD_
    STYLE=Development CPP_HEADERMAP_FILE=/Users/davemark/Desktop/bzflag-1.10.5.
    20040426/src/platform/MacOSX/build/BZFlag.build/config.h.build/config.h.hma
    p SRCROOT=/Users/davemark/Desktop/bzflag-1.10.5.20040426/src/platform/MacOS
    X OBJROOT=/Users/davemark/Desktop/bzflag-1.10.5.20040426/src/platform/MacOS
    X/build SYMROOT=/Users/davemark/Desktop/bzflag-1.10.5.20040426/src/platform
    /MacOSX/build DSTROOT=/tmp/BZFlag.dst

PhaseScriptExecution <Execution>/Users/davemark/Desktop/bzflag-1.10.5.20040426/s
    rc/platform/MacOSX/build/BZFlag.build/config.h.build/BPTag000-script.sh
   echo "=== Script ==="
   cat /Users/davemark/Desktop/bzflag-1.10.5.20040426/src/platform/MacOSX/build
    /BZFlag.build/config.h.build/BPTag000-script.sh
   echo "--- Output ---"
   /Users/davemark/Desktop/bzflag-1.10.5.20040426/src/platform/MacOSX/build/BZF
    lag.build/config.h.build/BPTag000-script.sh
   _result=$?
   echo "=== Exit code: ${_result} ==="
   exit ${_result}
=== Script ===
#!/bin/sh
```

Figure 6. Be sure you split your build window by dragging the dot from the bottom of the window towards the middle.

If you get an error message during the build process, try changing the target to BZAdmin, doing a *Build* of that target, then selecting BZFlag as a target and doing a *Build and Run* again. Worst case, do a *Build* of the *Everything* target, then select *BZFlag* and do a *Run*. You need to run the *BZFlag* target to run the game.

Have fun playing BZFlag. It will take over the entire screen and put up a simple menu. Navigate the menu with the arrow keys and the return key. Escape takes you back up a level. Your choices are *Join Game*, *Options*, *Help*, and *Quit*. Review the help and options if you like, then select *Join Game*. This will take you to a new menu screen. Select *Find Server*, scroll through the list of servers to find a game you like, select the game, then select *Connect*.

That's it. You're in the game. And there is *always* a game going. It is amazing how big this game has gotten. When you are done playing, hit escape, then arrow down to *Quit*.

Doing a Unix Build

Not every Mac OS X project will ship with an Xcode project. That's OK, though. As long as the developers included the appropriate make files, we can use those to do our build. So if you had some trouble with your Xcode build or if you just want to give the Unix build process a try, here's the process.

Start by launching the Terminal app.

Use the cd command to navigate into the top level of the BZFlag folder you downloaded. For example, I unstuffed my archive onto my desktop. From my home directory, I did:

```
cd Desktop
```

```
and then I did:
```

```
cd bzflag-1.10.5.20040426
```

```
I then did:
```

```
./autogen.sh
```

The leading "./" tells the Unix shell to execute the shell script contained in the file *autogen.sh* found in the current directory, that is, in the directory *bzflag-1.10.5.20040426*.

The script hangs a bit, then spits out this message:

```
BZFlag sources are now prepared. To build here,
run:
 ./configure
 make
```

And that's exactly what you want to do next. First, type:

```
./configure
```

This will create a bunch of Makefiles and do a few other things. When it is done, type this command:

```
make
```

```
Got a dual processor G5? If so, do a:
```

```
make -j2
```

This tells make to split the task into two concurrent jobs. Gets the job done in half the time!

This will take a while. You'll be compiling a fairly large batch of code, so sit back and dream about *Gilligan's Island*. Who would you be? Hmmm…

By the way, in case you were wondering, gcc is the C compiler, and g++ is the GNU C++ compiler.

The build should take about 10 minutes (obviously, depends on your processor). To run the binary, be sure you are still in that top directory and type the command:

```
src/bzflag/bzflag
```

If you get an error message, do an ls and verify that there is a directory named src in the current directory. If not, you may have changed directories

Till Next Month...

Hopefully, between the Xcode method and the Terminal method, you've gotten BZFlag to run. Here are a couple of BZFlag related sites that can help you with your gameplay:
http://www.dervishdukot.org/home/bzflag/bzflag.html
http://bzflag.org/wiki/

Also, if you are IRC-savvy, you can make your way over to #bzflag on irc.freenode.net with any questions or comments.

Your homework assignment is to search through some of the other Open Source projects on SourceForge.net listed in the Mac OS X area, find one you like, and get it to build.

A *huge* shout out to Sean "brlcad" Morrison for his most excellent help in getting BZFlag to build.

Oh, and be sure to head over to http://www.spiderworks.com and sign up for the email list. See you next month…☺

MT

About The Author

Dave Mark is a long-time Mac developer and author and has written a number of books on Macintosh development. Dave has been writing for MacTech since its birth! Be sure to check out Dave's latest and greatest at http://www.spiderworks.com.

GETTING STARTED WITH PHP

What Makes PHP Different

In most programming languages, the output from your program goes to the user, in some form or another. For example, in your Cocoa program, you might put up a window, then display some data in the window, perhaps with a scrollbar that lets the user scroll through the data and buttons that allows the user to accept the data or cancel whatever process brought up the data in the first place. The point is, some portion of your program is directly dedicated to interacting with the user.

PHP is different. Though PHP does sport a set of user interface functions, one of its strongest uses is as a means to dynamically generate web content. For example, suppose you were building a web site and you wanted to serve up one set of pages for folks running Safari and a different set of pages for other folks. There are lots of ways to do this, and PHP offers its own mechanism for doing this (and we'll take a look at this a bit later in the column).

More importantly, suppose you wanted to build a web site that was data-driven. For example, you might build a table that lists the most up-to-date statistics from your weekly *Halo 2* league. You *could* embed the statistics directly in your HTML, or you could use PHP to build an interface to an open source database like mySQL or PostgreSQL. You'd write one PHP-laden web tool for entering the data, then another for displaying the data.

PHP is idea for managing HTML forms. Especially if you'll want to back your forms up to a database. Once you get a taste of PHP, you'll definitely want to play with it yourself. And the cool thing is, if you know a bit of C, you'll find PHP quite easy to pick up.

The point is, some portion of your program is directly dedicated to interacting with the user.

Installing PHP

In the golden, olden days of Mac OS X, installing PHP was a bit of an adventure. For starters, you downloaded the latest version of Apple's developer tools, then used Google or the like to find the most recent source code distribution of PHP that some kind soul had ported to Project Builder (later Xcode) format. Each source code distribution came with a set of scripts that would drive the process of compiling the source and installing the binaries in the right place. These scripts are known in the Unix world as makefiles. Once you downloaded the distribution that was right for your version of Mac OS X, you ran the makefile, got yourself a sandwich and, hopefully, by your last bite you had yourself an installed version of PHP. More often than not, however, you ran into a funky error that required you to tweak some permission or other, or perhaps rename a directory. Bottom line, this was not rocket science, but it was far from trivial.

One of the things I love most about writing this column is the opportunity to play with different technologies. I especially love it when this means playing with a technology that takes a sharp left turn from the traditional. A perfect example of this is PHP, a programming language that looks like C, smells like C, but takes C in a very different direction.

Nowadays, PHP has really hit the mainstream. Soon after each rev of PHP or Mac OS X release, web pages pop up with instructions on the best way to install that version of PHP on your particular configuration of Mac OS X. Some folks even go to the trouble of creating traditional Mac OS X installers that do all the hard work for you.

I used Google to search for:

```
"php 5" "mac os x"
```

Why php 5? When I wrote this article, PHP 5.0.2 was the latest release. By the time you read this, I suspect PHP 5.0.3 will be out, if not an even later version. But the major release will still likely be 5, so a search for PHP 5 will work.

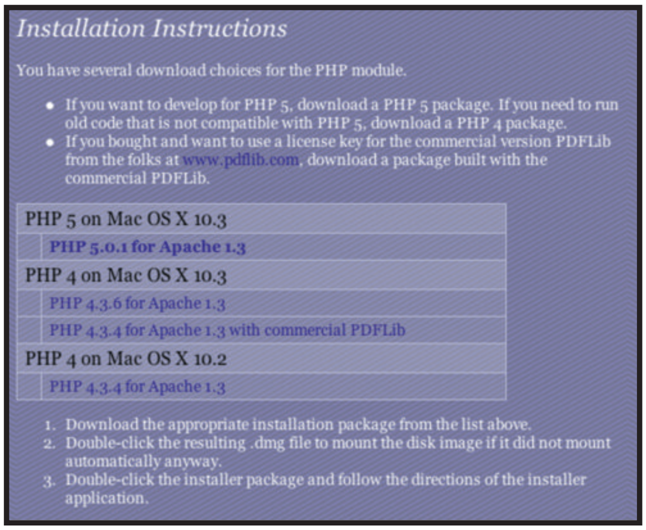
The site I chose for this month's column has a packaged installer for PHP 5.0.1, as well as a forum section where you can message with other folks with similar experiences. Hopefully, by the time you read this, the installer will have been updated to the latest version, but for the purposes of this discussion, 5.0.1 will do just fine.

Here's the link to the main page:
http://www.entropy.ch/software/macosx/php/

Note that this install is for the version of PHP designed to work with the Apache web server that ships with Mac OS X. You are going to install PHP on your own machine, then put our php test pages in the *Sites* folder in your home directory. In effect, the Apache web server on your local machine will be serving up pages to you, without going over the net. This is a *great* way to learn PHP. With this setup, you can make changes to your code without using an FTP client. Just edit the file in place, save, then test. As long as you save your change, you don't even need to close the file before switching back to your browser to hit the reload button to retest the page. This is an extremely efficient way to work.

If this is still a bit confusing to you, not to worry. After we do the install, we'll run a few examples so you get the hang of working with PHP.

Figure 1 shows the install instructions from our download page. You'll want to click on the link labeled *PHP 5.0.1 for Apache 1.3* (remember, the page might have been updated to a more recent version of PHP). Click the link and a disk image will be downloaded. It's about 20 megs, so it might take a while. Once the *.dmg* file downloads, the disk image will mount. Navigate into it and double click on the file named *php-5.0.1.pkg* (or whatever the *.pkg* file is called when you download it).



Installation Instructions

You have several download choices for the PHP module.

- If you want to develop for PHP 5, download a PHP 5 package. If you need to run old code that is not compatible with PHP 5, download a PHP 4 package.
- If you bought and want to use a license key for the commercial version PDFLib from the folks at www.pdflib.com, download a package built with the commercial PDFLib.

PHP 5 on Mac OS X 10.3
- PHP 5.0.1 for Apache 1.3

PHP 4 on Mac OS X 10.3
- PHP 4.3.6 for Apache 1.3
- PHP 4.3.4 for Apache 1.3 with commercial PDFLib

PHP 4 on Mac OS X 10.2
- PHP 4.3.4 for Apache 1.3

1. Download the appropriate installation package from the list above.
2. Double-click the resulting .dmg file to mount the disk image if it did not mount automatically anyway.
3. Double-click the installer package and follow the directions of the installer application.

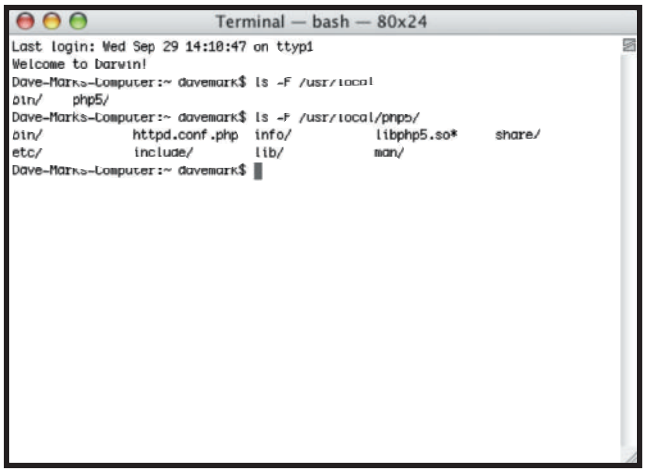
Figure 1 – Installation instructions for installing PHP

Follow the installation instructions and PHP should install without a hitch. To verify that the install went OK, fire up Terminal and enter these two commands:

```
ls -F /usr/local
ls -F /usr/uocal/php5/
```

For you non-Unix folks, the ls command lists the contents of the specified directory or file. The -F option asks ls to put a * at the end of executable files and a / at the end of directories. This makes an ls listing a bit easier to understand.

Figure 2 shows my results after I did my PHP install. Note that PHP is installed in the directory /usr/local/php5/.



```
Last login: Wed Sep 29 14:10:47 on ttyp1
Welcome to Darwin!
Dave-Marks-Computer:~ davemark$ ls -F /usr/local
bin/    php5/
Dave-Marks-Computer:~ davemark$ ls -F /usr/local/php5/
bin/            httpd.conf.php  info/           libphp5.so*     share/
etc/            include/        lib/            man/
Dave-Marks-Computer:~ davemark$ 
```

Figure 2 – A Terminal listing showing the location of the newly installed PHP files

To test the installation, create a plain text file using a tool like TextEdit, BBEdit, or Xcode. Be sure that your text files are plain text files. Personally, I use BBEdit for all my PHP and web editing. Here's a link to a page that will let you download a demo of BBEdit:

http://www.barebones.com/products/bbedit/demo.shtml

Regardless of how you create the plain text file, here's what goes in it:

```
<?php phpinfo() ?>
```

Save this line in a file named *test.php* and place the file inside the *Sites* directory in your home directory. It is critical that the file have the .php file extension so the Apache web server knows to pass the file through the PHP pre-processor before serving up the page. The PHP pre-processor will scan the file looking for PHP code to interpret. PHP code always starts with "<?php" and ends with "?>". You can have more than one block of PHP code in a single file. We'll show examples of this a bit further along in the article.

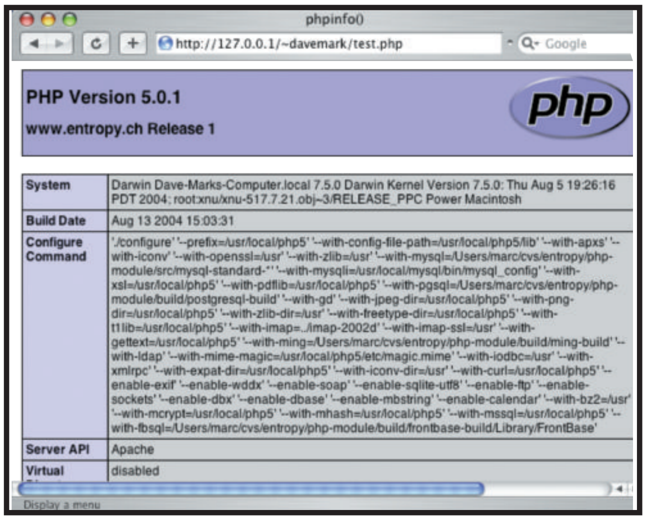
The line above contains a single PHP statement, a call of the function phpinfo(). This function returns a bunch of information about your PHP installation, all formatted in a two column HTML table. Why does the function return HTML? That's one of the most important aspects of PHP. Your PHP code will generate HTML code, which will appear in line with the HTML code in which it is embedded. Once the PHP code is done running and its output is incorporated into the surrounding HTML, the full HTML is returned by the server and rendered by your browser. Again, we'll get into this more later on in the article.

For the moment, save your one line php file into the *Sites* directory in your home folder. To test the file, use this link:

http://127.0.0.1/~davemark/test.php

Obviously, you'll replace "davemark" with your own user name. The 127.0.0.1 is an IP address that represents your local machine. The ~davemark represents the *Sites* directory of the user davemark. And, of course, the file name *test.php* is the file we are passing along to the Apache web server.

Figure 3 shows the output when I ran *test.php* on my machine. Obviously, this is just the first few lines of a very long series of tables.



PHP Version 5.0.1

www.entropy.ch Release 1

System	Darwin Dave-Marks-Computer.local 7.5.0 Darwin Kernel Version 7.5.0: Thu Aug 5 19:26:16 PDT 2004; root:xnu/xnu-517.7.21.obj~3/RELEASE_PPC Power Macintosh
Build Date	Aug 13 2004 15:03:31
Configure Command	'./configure' '--prefix=/usr/local/php5' '--with-config-file-path=/usr/local/php5/lib' '--with-apxs' '--with-iconv' '--with-openssl=/usr' '--with-zlib=/usr' '--with-mysql=/Users/marc/cvs/entropy/php-module/src/mysql-standard-*' '--with-mysqli=/usr/local/mysql/bin/mysql_config' '--with-xsl=/usr/local/php5' '--with-pdflib=/usr/local/php5' '--with-pgsql=/Users/marc/cvs/entropy/php-module/build/postgresql-build' '--with-gd' '--with-jpeg-dir=/usr/local/php5' '--with-png-dir=/usr/local/php5' '--with-zlib-dir=/usr' '--with-freetype-dir=/usr/local/php5' '--with-t1lib=/usr/local/php5' '--with-imap=../imap-2002d' '--with-imap-ssl=/usr' '--with-gettext=/usr/local/php5' '--with-ming=/Users/marc/cvs/entropy/php-module/build/ming-build' '--with-ldap' '--with-mime-magic=/usr/local/php5/etc/magic.mime' '--with-iodbc=/usr' '--with-xmlrpc' '--with-expat-dir=/usr/local/php5' '--with-iconv-dir=/usr' '--with-curl=/usr/local/php5' '--enable-exif' '--enable-wddx' '--enable-soap' '--enable-sqlite-utf8' '--enable-ftp' '--enable-sockets' '--enable-dbx' '--enable-dbase' '--enable-mbstring' '--enable-calendar' '--with-bz2=/usr' '--with-mcrypt=/usr/local/php5' '--with-mhash=/usr/local/php5' '--with-mssql=/usr/local/php5' '--with-fbsql=/Users/marc/cvs/entropy/php-module/build/frontbase-build/Library/FrontBase'
Server API	Apache
Virtual	disabled

Figure 3 – The output from phpinfo().

Hello, World!

Our next example shows what happens when we mix PHP and HTML. Create a new plain text file and type in this code:

```
<html>
  <head>
    <title>PHP Test</title>
  </head>
  <body>
    <p>This is some pure HTML loveliness.</p>
    <?php
      echo "<p>Hello, World!</p>";
    ?>
    <p>Did we echo properly?</p>
    <?php
      echo date("r");
    ?>
    <p>It works!!!</p>
  </body>
</html>
```

Save the file as *hello.php* and save it in your *Sites* directory. When you send this file to Apache, Apache will note the *.php* file extension and send the file to the PHP pre-processor. The pre-processor will scan the file, copying the HTML to its output as is, until it encounters the open PHP tag:

```
<?php
```

As soon as it hits the end of those characters, the pre-processor starts interpreting the rest of the file as PHP code, until it hits the close PHP tag:

```
?>
```

Once it hits that close tag, the processor runs the PHP code it just scanned and places the output from the PHP code following the HTML code it just copied. In the *hello.php* example, this means executing this statement:

```
    echo "<p>Hello, World!</p>";
```

and copying the output from that statement into the HTML stream. The echo command simply copies its parameters to output, where it joins the HTML stream.

Once the close PHP tag is encountered, the pre-processor continues copying the HTML to its output until it hits the end of the code or encounters another open PHP tag.

Note that our example has two chunks of PHP code. The second chunk executes this line of code:

```
echo date("r");
```

The first version of echo you saw copied the text string to output. This version of echo has a function as a parameter. In that case, the PHP pre-processor calls the function and returns the output of the function call as input to echo. echo simply echoes that output to the HTML stream.

Confused? Type in this link to execute your copy of *hello.php*. Be sure to replace my user name with your user name:

http://127.0.0.1/~davemark/hello.php

BMS

THE LAW OFFICE OF BRADLEY M. SNIDERMAN

Need help safeguarding your software?

If you're developing software, you need your valuable work protected with trademark and copyright registration, as well as Non Disclosure Agreements.

Then, when you are ready to sell it, you can protect yourself further with a licensing agreement.

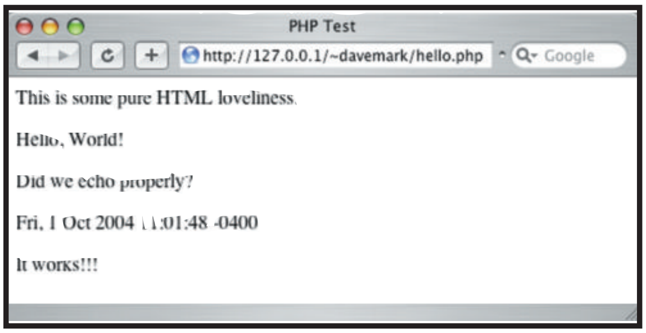
I am an attorney practicing in Intellectual Property, Business Formations, Corporate, Commercial and Contract law.

Please give me a call or an e-mail. Reasonable fees.

23679 Calabasas Rd. #558 • Calabasas, CA 91302

PHONE 818-222-0365 **FAX** 818-591-1038 **EMAIL** brad@sniderman.com

The output of this example is shown in **Figure 4**.



This is some pure HTML loveliness.

Hello, World!

Did we echo properly?

Fri, 1 Oct 2004 11:01:48 -0400

It works!!!

Figure 4 – hello.php in action.

To get a true sense of this process, choose *View Source* from Safari's *View* menu. This will show you the merged PHP output and HTML code. Here's my merged source:

```
<html>
  <head>
    <title>PHP Test</title>
  </head>
  <body>
    <p>This is some pure HTML loveliness.</p>
    <p>Hello, World!</p>   <p>Did we echo
properly?</p>
    Fri,  1 Oct 2004 11:01:48 -0400       <p>It
works!!!</p>
  </body>
</html>
```

Notice that the output from the PHP commands is right there in the mix. One bit of funkiness, though. Notice that the PHP generated HTML did not include a carriage return, so the follow-on HTML starts on the same line as the end of the PHP output. In this line of output:

```
    <p>Hello, World!</p>   <p>Did we echo
properly?</p>
```

you can see that the two paragraphs are on the same line of source. This will not affect the final output, but it does make the source a bit harder to read. An easy solution to this is to embed a carriage return character "\n" at the end of each line of PHP output.

Here's a new version of *hello.php*:

```
<html>
  <head>
    <title>PHP Test</title>
  </head>
  <body>
    <p>This is some pure HTML loveliness.</p>
    <?php
      echo "<p>Hello, World!</p>\n";
    ?>
    <p>Did we echo properly?</p>
    <?php
      echo date("r");
      echo "\n";
    ?>
    <p>It works!!!</p>
  </body>
</html>
```

Note that we added the "\n" directly at the end of the first echo's parameter string. Since the second echo did not use a string, we added a second line of code, just to echo the "\n".

When you run this chunk of code, the output will be the same. But let's take a look at the source code that is generated when you do a *View Source*:

```
<html>
  <head>
    <title>PHP Test</title>
  </head>
  <body>
    <p>This is some pure HTML loveliness.</p>
    <p>Hello, World!</p>
    <p>Did we echo properly?</p>
    Fri,  1 Oct 2004 11:39:16 -0400
    <p>It works!!!</p>
  </body>
</html>
```

Notice that the carriage returns we added made the intermediate source a bit easier to read.

Include Other PHP Files

Here's another example, for you folks who like the organizational power of include files. This one is a slight mod of one from the php.net site. Create a new file named *vars.php* and type in the following code:

```
<?php
  $color = 'green';
  $fruit = 'apple';
?>
```

Save the file in the *Sites* folder and create a second new file named *inc_test.php*. Here's the code:

```
<html>
  <head>
    <title>PHP Include Test</title>
  </head>
  <body>
    <?php
      echo "<p>A $color $fruit</p>"; // A
      include 'vars.php';

       echo "<p>A $color $fruit</p>"; // A green apple
    ?>
  </body>
</html>
```

Save this file in the *Sites* folder as well. Run this example by typing in:

http://127.0.0.1/~davemark/inc_test.php

$32 < 1 Year

2 Years > $64

Interviews

Tapping into the world of celebrities and their Macs, only MacDirectory offers exclusive interviews. Get a close and personal view from Sarah Jessica Parker, Steve Jobs, Madonna, Harry Connick Jr., George Lucas, Jennifer Jason Leigh, Steve Woz and other leaders in the Mac community.

Features

Designers, writers, musicians, business leaders & our technical expert team offer their own personal interpretation of things that only the Mac system can deliver. With more than 200 pages of news, insights, trends and the largest Macintosh buyer's guide including over 5,000 Mac products and services.

MacDirectory

BEYOND ANY MACINTOSH MAGAZINE.

Culture

MacDirectory takes you to the wildest corners of the world and uncovers how Macintosh computers are being used by other cultures. Travel to Japan, Australia, Germany, Brazil & Russia and learn more about Apple's cultural impact around the globe.

Reviews

Find out all you need to know about the latest Mac products including the hottest Mac OS software and hardware.

Subscribe >

macdirectory.com

Send check or money order to:
MacDirectory Subscription Dept.
326 A Street, 2C
Boston, MA 02210

Remember to replace davemark with your username. Your output should look like this:

```
A

A green apple
```

In a nutshell, *inc_test.php* is made up of two identical echo statements, with an include statement in between. The echo statements print the value of two variables, each of which is set in the include file. Notice that the first echo does not have values for $color and $fruit and does not print anything for those values. The second echo occurs after the include file. Since the include file sets the values for the variables, the second echo prints those values.

This example was included as a bit of food for thought. Many web sites achieve their unified look and feel through included header, nav bar, and footer files, as well as through a judicious use of variables. No doubt you'll want to take advantage of include files and variables as you build your own PHP projects.

Restarting Apache

If you happened to reboot your machine since you did the PHP install, you may have noticed that Apache is no longer running. Not to worry. There are two easy ways to restart the server. The simplest way is to select System Preferences… from the apple menu, then select the Sharing icon. On the Sharing page, click on the Services tab and make sure the Personal Web Sharing checkbox is checked. As soon as you check the checkbox, Apache will be started and the Start button will change to Stop so you can stop the server. Unless Apache runs into a problem at restart, it should be restarted automatically when you restart your machine.

Another way to start and stop the server is by using the *apachectl* command in Terminal. Start up Terminal and type in this command:

```
sudo apachectl restart
```

You'll be prompted for your admin password, since the sudo command executes a shell with root privileges. Type in the password and the Apache server will be stopped (if it is running) and restarted. Either way works just fine.

Till Next Month...

We'll be doing a bit more with PHP next month. In the meantime, check out the web site http://www.php.net, the official web site for PHP developers. There are a ton of resources on this site, including complete on-line and downloadable PHP documentation. To figure out why the date() function did what it did, find the search field at the top right of the main page and type in the word "date", make sure the popup menu says "function list", then hit return or click on the right arrow icon. This will take you to the "date" documentation.

Check out all the format character options in Table 1 and play with a few of them. Enjoy, and I'll see you next month.

Oh, if you haven't done so already, be sure to head over to http://spiderworks.com and sign up. The new version of *Learn C*

MT

About The Author

Dave Mark is a long-time Mac developer and author and has written a number of books on Macintosh development. Dave has been writing for MacTech since its birth! Be sure to check out Dave's latest and greatest at http://www.spiderworks.com.

3D CAD for Woodwork

Macworld ★★★★½ September 2004

Design Intuition 1.3

"...easy and fun, as well as a great value." – Greg Miller

"A joy to use," say our users

- materials lists
- cutting diagrams
- accurate to Ångstroms
- Metric or Imperial (American)
- library of customizable designs

www.gizmolab.com/save/mt/1

More Finder Scripting

Manipulating Finder objects

In this article, when I refer to a *Finder* "item" or "object", please note that I am simply referring generically to either a file or a folder.

Opening

We have already seen how the following code can be used to open a folder using the *Finder*:

```
tell application "Finder"
  set theFolder to make new folder
at desktop with properties
{name:"My Folder"}
  open theFolder
end tell
```

You may also have the need to open a file using the *Finder*. To simply open a file, the AppleScript syntax is the same:

```
set theFile to choose file
tell application "Finder"
  open theFile
end tell
```

In some cases, you may want to open a file using a specific application. For example, let's say that you have a *Photoshop* or *ImageReady* droplet, or an AppleScript droplet, and you want to process one or more dropped items. You can do this with AppleScript in the *Finder* by making use of the `using` parameter along with the `open` command.

```
set theApplication to choose application as alias
set theFile to choose file
tell application "Finder"
  open theFile using theApplication
end tell
```

Revealing

If you simply want to locate and navigate to an item in the *Finder*, you can use the `reveal` command. This will locate the object in the *Finder*, open a new window if necessary, display and select the specified object.

```
set theFile to choose file
tell application "Finder"
  reveal theFile
end tell
```

Moving

Moving files and folders around is another common task involving the *Finder*. To move a file or folder, use the `move` command.

```
set theFile to choose file
set theDestinationFolder to choose folder with
prompt "Please select a destination folder:"
tell application "Finder"
  move theFile to theDestinationFolder
end tell
```

By default, the `move` command will not automatically replace existing items in the destination location with the

We've taken a look at some basic Finder scripting, including creating, naming, and updating folders. This month, let's expand a bit further, and begin looking at some other scriptable Finder functionality.

same name. If you want to replace existing items in any situation, then you can simply use the `replacing` parameter to indicate that any existing items should be replaced, if necessary. For example:

```
tell application "Finder"
  move theFile to theDestinationFolder replacing true
end tell
```

If you do not want to replace existing items, but you still want to move the item to the destination folder, then you will need to create custom code to handle the situation as you see fit. For example, if you wanted to add a unique numeric suffix to the item name, and then move it, you could use the following code:

```
set theFile to choose file
set theDestinationFolder to choose folder with prompt "Please
select a destination folder:"
tell application "Finder"
  set theFileName to name of theFile
  set thePathToCheck to theDestinationFolder & theFileName as
string
  if item thePathToCheck exists then
    set theSuffix to 1
    repeat
      if (item (thePathToCheck & theSuffix) exists) = false
then exit repeat
      set theSuffix to theSuffix + 1
    end repeat
    set name of theFile to theFileName & theSuffix
  end if
  move theFile to theDestinationFolder
end tell
```

Duplicating

To copy an item, you need to use the `duplicate` command, rather than the `copy` command. For example:

```
set theFile to choose file
set theDestinationFolder to choose folder with prompt
"Please select a destination folder:"
tell application "Finder"
  duplicate theFile to theDestinationFolder
end tell
```

As of the writing of this article, the *Finder* dictionary did indicate the presence of a `copy` command. However, this command was not yet implemented in the *Mac OS X Panther (10.3.3) Finder*. In addition, once functional, the `copy` command will be used to copy items to the clipboard, rather than to copy them from one location to another.

Duplicating is similar to moving, in that it will not automatically replace existing items with the same names. In order to replace existing items, you need to use the `replacing` parameter.

```
tell application "Finder"
  duplicate theFile to theDestinationFolder replacing true
end tell
```

If you do not want to replace existing items in a destination folder, but you still want to copy an item to the destination folder, then you will need to create code to handle this situation. For example:

```
set theFile to choose file
set theDestinationFolder to choose folder with prompt
"Please select a destination folder:"
tell application "Finder"
  set theFileName to name of theFile
  set thePathToCheck to theDestinationFolder & theFileName
as string
  if item thePathToCheck exists then
    set theSuffix to 1
    repeat
      if (item (thePathToCheck & theSuffix) exists) = false
then exit repeat
```

```
      set theSuffix to theSuffix + 1
    end repeat
    set name of theFile to theFileName & theSuffix
  end if
  duplicate theFile to theDestinationFolder
end tell
```

Deleting

To delete an item, use the `delete` command:

```
set theFile to choose file
tell application "Finder"
  delete theFile
end tell
```

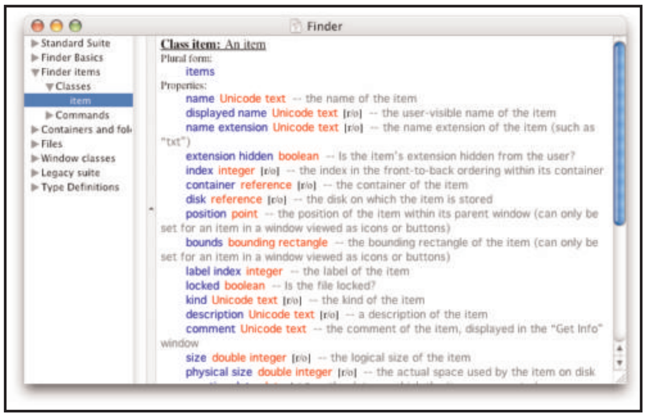
Please note that the `delete` command will not actually delete an item. Rather, it will move the item to the trash, where it may be retrieved until the trash has been emptied. If you want to fully delete a file, you will need to tell the *Finder* to empty the trash after performing the deletion. For example:

```
set theFile to choose file
tell application "Finder"
  delete theFile
  empty the trash
end tell
```

Keep in mind that by emptying the trash, you will be removing any other items residing in the trash as well.

Getting Object Info

When working with an item in the *Finder*, you will probably want to retrieve information about the item. You can do this by accessing the properties of the desired object. Common properties shared by both files, folders, and disks can be found under the *Finder Items* suite in the *Finder* dictionary.



Class item: An item
Plural form:
    items
Properties:
    name Unicode text -- the name of the item
    displayed name Unicode text [r/o] -- the user-visible name of the item
    name extension Unicode text [r/o] -- the name extension of the item (such as "txt")
    extension hidden boolean -- Is the item's extension hidden from the user?
    index integer [r/o] -- the index in the front-to-back ordering within its container
    container reference [r/o] -- the container of the item
    disk reference [r/o] -- the disk on which the item is stored
    position point -- the position of the item within its parent window (can only be set for an item in a window viewed as icons or buttons)
    bounds bounding rectangle -- the bounding rectangle of the item (can only be set for an item in a window viewed as icons or buttons)
    label index integer -- the label of the item
    locked boolean -- Is the file locked?
    kind Unicode text [r/o] -- the kind of the item
    description Unicode text [r/o] -- a description of the item
    comment Unicode text -- the comment of the item, displayed in the "Get Info" window
    size double integer [r/o] -- the logical size of the item
    physical size double integer [r/o] -- the actual space used by the item on disk

Figure 1. Finder Dictionary > Finder Item Detail

Some commonly accessed properties of *Finder* items include the `modification date`, `name`, and `size` of the item. For example:

```
set theFile to choose file
tell application "Finder"
  set theModDate to modification date of theFile
  set theName to name of theFile
  set theSize to size of theFile
end tell
```

In addition to these common properties, files, folders, and disks also have additional properties specific to their particular class. For example, files have a `file type` and `creator type` property, whereas folders and disks have other properties not possessed by files. For example:

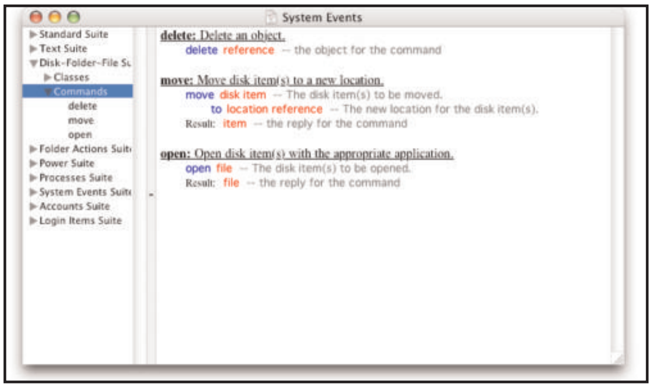
```
set theFile to choose file
tell application "Finder"
  set theFileType to file type of theFile
  set theCreatorType to creator type of theFile
end tell
```

Some developers prefer to avoid scripting the *Finder* when possible, and resort instead to using a scripting addition to access certain properties of files and folders. The *Standard Additions* scripting addition, which is installed with *Mac OS X*, contains a command for just this task – info for. This command may be used to retrieve a variety of properties for files and folders, such as name, modification date, size, and more. For example:

```
set theFile to choose file
set theFileInfo to info for theFile
set theName to name of theFileInfo
set theModDate to modification date of theFileInfo
set theSize to size of theFileInfo
```

What About System Events?

If you have done some scripting in *Mac OS X* before, then you may be somewhat familiar with the *System Events* background application. This application allows you to automate various system related activities.



delete: Delete an object.
    delete reference -- the object for the command

move: Move disk item(s) to a new location.
    move disk item -- The disk item(s) to be moved.
        to location reference -- The new location for the disk item(s).
    Result: item -- the reply for the command

open: Open disk item(s) with the appropriate application.
    open file -- The disk item(s) to be opened.
    Result: file -- the reply for the command

Figure 2. The System Events Dictionary Window

Some of the commands in the *System Events* dictionary are very similar to commands found in the *Finder* dictionary, including the `delete`, `move`, and `open` commands. For these specific commands, *System Events* may be used instead of the *Finder*. However, please note that certain parameters, which are present when scripting the *Finder*, are not present when scripting *System Events*. For example, when moving an item using

System Events, there is not currently a way to specify whether existing items should be overwritten. When using this command, items will never be overwritten.

```
set theFile to (choose file) as string
set theDestinationFolder to choose folder
tell application “System Events”
  move disk item theFile to theDestinationFolder
end tell
```

The *System Events* dictionary has expanded significantly with the last few major *Mac OS X* releases, and I expect it to continue to expand in the future. My guess is that more *Finder*-like functionality will continue to be built into *System Events* with every major OS release. *System Events* contains much more than the few commands I mentioned above, and I encourage you to explore it in greater detail in order to find out more about what *System Events* has to offer.

In Closing

This month’s article should take you a little further down the road of *Finder* scripting. For some editable examples of *Finder* scripting, you may want to check out the example *Finder* scripts included with *Mac OS X*. These can be found in the *Library > Scripts > Finder Scripts* folder on your machine. In addition, Apple’s AppleScript web site contains some *Finder* scripts, which can be triggered from the *Finder’s* toolbar. For additional information about all of these scripts, as well as links to download the toolbar scripts, please visit http://www.apple.com/applescript/finder/.

Until next time, keep scripting!

MT

About The Author

Benjamin Waldie is president of Automated Workflows, LLC, a firm specializing in AppleScript and workflow automation consulting. In addition to his role as a consultant, Benjamin is an evangelist of AppleScript, and can frequently be seen presenting at Macintosh User Groups, Seybold Seminars, and MacWorld. For additional information about Benjamin, please visit http://www.automatedworkflows.com, or email Benjamin at applescriptguru@mac.com.

Website Not Found By Clients
HTTP 404 - Website not found

The website you are looking for can't be found by your clients. It may have been improperly marketed, had poor design, or didn’t work. Regardless, it's not helping your business!

Your Options:

- Rent a chicken suit and stand on the corner handing out flyers
- Paint the company URL on your chest and face during a major sporting event
- Contact SharpNET Solutions internet marketing and web design specialists, watch your traffic and rankings increase, get great feedback from all your new customers.

HTTP 404 - Website needs SharpNET

Not Marketing Your Site?
If you are not marketing your website online, you may as well have a 404 error for a website.

Web Design
Internet Marketing
Consulting
Lead Generation
Multi-Media

1-877-583-8396
www.sharpnetsolutions.com

Mac OS® X compatible

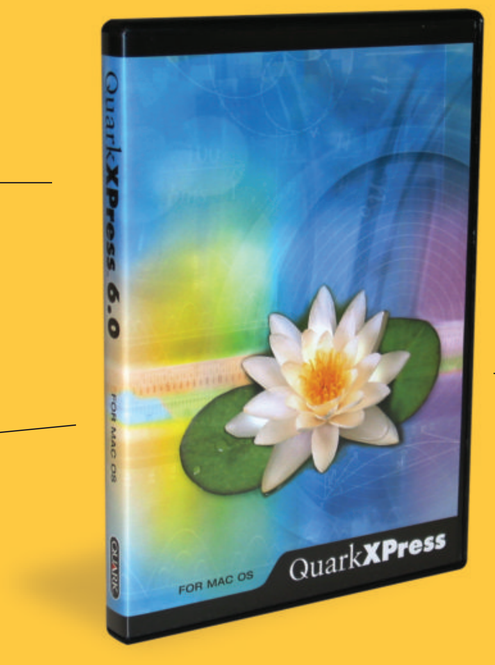
QuarkVista™
image editing

OpenType
fonts from
Linotype*

Direct PDF export

QuarkXClusive™
variable data printing

Multiple levels
of undo



The joy of six.

QuarkXPress 6

QuarkXPress® 6 has arrived. Take it out for a spin at www.quark.com/demo6.
Show it what you can do. It'll return the favor.

*Offer ends December 31, 2005. Available for download by registered QuarkXPress 6.5 users. ©2004 Quark Technology Partnership. All rights reserved. Quark, the Quark logo, and QuarkXPress are trademarks of Quark, Inc. and all applicable affiliated companies, Reg. U.S. Pat. & Tm. Off. and in many other countries. QuarkVista and QuarkXClusive are trademarks of Quark, Inc. and all applicable affiliated companies. All other marks belong to their respective owners. 60198AD

FILL ONLINE PDF FORMS USING HTML FORMS

BY SID STEWARD

Adobe's Portable Document Format (PDF) is really only as portable as the viewer used to read or print it. This has become an issue in recent years as the Adobe Reader (née Acrobat Reader) has evolved to support some platforms better than others. Web publishers who desire maximum portability must now take stock: would this work on OS X or Linux as well as Windows? This issue is complicated by the rise of alternative PDF viewers such as Apple's Preview and alternative web browsers such as Konqueror.

Basic PDF viewing and printing is generally okay. Interactive PDF forms, however, are a different story. Adobe Reader on Windows integrates closely with popular web browsers, allowing a web developer to drive an interactive PDF form filling session using the web server (e.g., http://pdfhacks.com/form_session/form_session-1.1/). OS X users, however, won't have the same experience, nor will many Linux users.

One solution is to use HTML form features instead of PDF form features when collecting data. The web server can manage this data collection session, providing data validation and any necessary database access. When the form is complete, the web server can load the PDF form with the user's data, flatten the form, and then serve it to the user. "Flattening" makes the dynamic form data a permanent part of the page, so the resulting PDF will display properly using any PDF viewer.

Collecting data online using HTML forms is old hat. We'll discuss the part where you pack this data into the PDF form for delivery to your user. We'll also talk about how you can automatically convert a PDF form into an HTML form. My free, command-line tool, pdftk, makes both of these possible. We'll need to discuss how to get pdftk working on OS X (it also works on FreeBSD, Linux, Solaris and Windows). We should also touch on PDF forms.

PDF Forms

Using Adobe Acrobat 4, 5, or Acrobat 6.0 Pro (but not 6.0 Standard), you can add interactive form fields to PDF documents. PDF form fields closely resemble the form fields available to HTML form programmers. You have text boxes, check boxes, radio buttons, combo boxes, list boxes, and buttons. These can be further configured to suit your needs. For example, a text box can be configured to be multi-line or to mask password input, and buttons can be configured to submit the form data to a web server.

You can even program PDF forms using JavaScript, although the PDF document object model is quite different than the DOM familiar to web developers. To learn more about programming PDF using JavaScript, see the Acrobat JavaScript Object Specification (http://partners.adobe.com/asn/developer/pdfs/tn/5186AcroJS.pdf) and the Acrobat JavaScript Scripting Guide (http://partners.adobe.com/asn/acrobat/sdk/public/docs/AcroJSGuide.pdf). We won't discuss using JavaScript with PDF forms, here. Though, I will mention the site http://www.math.uakron.edu/~dpstory/acrotex.html, where you will find JavaScript powered PDF games, such as Tic-Tac-Toe and Naval Battle.

For our purposes, the important thing about PDF forms is that you can permanently merge them with form data. You can do this using Acrobat, or you can use the free, command-line PDF Toolkit, pdftk.

Under pdftk's hood, the iText PDF library does all the heavy lifting. iText is written in Java, but I prefer programming in C++. So I used GCJ, the Java compiler maintained as part of GNU GCC. GCJ allows me to compile iText and then link it with my C++ program. The result is a stand-alone binary that does not need Java. Very cool.

The problem is that your OS X system probably doesn't have GCJ. You must build GCJ (along with GCC) before you can build pdftk on OS X. Happily, John M. Gabriele provides instructions at: http://users.bestweb.net/~john3g/gcj_osx/gcj_on_osx.html. Brian D. Foy documents his experience building GCJ and pdftk at: http://www.oreillynet.com/pub/wlg/5500.

COLLECT DATA USING AN HTML FORM, DELIVER A FILLED-OUT PDF FORM THAT WORKS IN PREVIEW

Pdftk, the PDF Toolkit

Pdftk is a command-line program for manipulating PDF documents; it is free software. I created it one year ago to fulfill my own requirements. Since then, I have added features that I believed this free, general-purpose PDF tool should provide. It can:

- Merge PDF Documents
- Split PDF Pages into a New Document
- Decrypt Input as Necessary (Password Required)
- Encrypt Output as Desired
- Fill PDF Forms with FDF Data and/or Flatten Forms
- Apply a Background Watermark
- Report on PDF Metrics such as Metadata, Bookmarks, and Page Labels
- Update PDF Metadata
- Attach Files to PDF Pages or the PDF Document
- Unpack PDF Attachments
- Burst a PDF Document into Single Pages
- Uncompress and Re-Compress Page Streams
- Repair Corrupted PDF (Where Possible)

The pdftk web site (http://www.pdftk.com) describes these features and explains how to get pdftk working on your system. Pdftk does not require Acrobat or Java. An OS X 10.3 installer is available for pdftk 1.11 from the site. Alternatively, you can build pdftk yourself, a non-trivial task described below. You must have version 1.11 if you want to automatically create an HTML form from a PDF form.

After building and installing GCC/GCJ, download and unpack the latest version of pdftk (currently 1.11) from http://www.pdftk.com. If you configured GCC/GCJ with —prefix=/usr/local/gcj as John describes, then you won't need to edit the OS X Makefile. Otherwise you will need to edit Makefile.MacOSX so that TOOLPATH matches your location of GCC/GCJ.

After unpacking pdftk 1.11, change into the pdftk-1.11/pdftk directory and run make -f Makefile.MacOSX. It will take awhile to finish compiling. When it is done, move the resulting pdftk program to a convenient location in your $PATH, such as /usr/bin. Test pdftk by displaying its help page:

```
pdftk —help
and merging a couple PDFs together:
pdftk 1.pdf 2.pdf cat output 12.pdf
```

Note that you cannot name pdftk's output PDF so it overwrites an input PDF. Also, upon success, pdftk will overwrite files with its output without warning. Change this latter behavior by appending do_ask to the end of the command line, or change the ASK_ABOUT_WARNINGS setting in Makefile.MacOSX and recompile pdftk.

Before we begin using pdftk to fill PDF forms with data, let's talk about FDF.

Store Form Data Using FDF

FDF is Adobe's Forms Data Format, a file format for storing and managing PDF form data. FDF is usually plain text, so you can create it pretty easily using a text editor or your favorite scripting language. FDF is fully documented in section 8.6.6 of the PDF Reference, fourth edition. You can download the latest version of the PDF Reference from: http://partners.adobe.com/asn/tech/pdf/specifications.jsp. Here is an example of an FDF file that assigns the value “San Francisco” to the PDF field named city:

```
%FDF-1.2
%âãÏÓ
1 0 obj
<< /FDF << /Fields [<< /T (city) /V (San Francisco) >>
                    << /T (state) /V (California) >> ]
>>
>>
endobj
trailer << /Root 1 0 R >>
%%EOF
```

To simplify FDF creation, I created a PHP program called forge_fdf. It takes form data as name/value pairs and then spins out the matching FDF. The program logic should be easy to reproduce in any language. Visit http://www.pdfhacks.com/forge_fdf/ to download the latest version. In PHP, you would use forge_fdf like so:

```
<?
require_once( 'forge_fdf.php' );

// use this array for text fields, combo box, and list box form field values
$fdf_data_strings= array(    'city' => 'San Francisco',
                            'state' => 'California' );

// use this array for check box and radio button values
$fdf_data_names= array();

// these aren't used in this example
$fields_hidden= array();
$fields_readonly= array();

$fdf= forge_fdf( '',
                $fdf_data_strings,
                $fdf_data_names,
                $fields_hidden,
                $fields_readonly );

$fdf_fn= tempnam( '.', 'fdf' );
$fp= fopen( $fdf_fn, 'w' );
if( $fp ) {
  fwrite( $fp, $fdf );
  fclose( $fp );

  // serve PDF, but prompt the user to save it to disk
  header( 'Content-type: application/pdf' );
  header( 'Content-disposition: attachment; '.
          'filename=filled_form.pdf' );

  // our pdftk magic; "flatten" merges data with the page
  passthru( 'pdftk form.pdf fill_form '. $fdf_fn.
            ' output - flatten' );

  unlink( $fdf_fn ); // delete temp file
}
else { // error
  echo 'Error: unable to write temp fdf file: '.
$fdf_fn;
}
?>
```

One FDF peculiarity is that text field, combo box and list box form field values are represented as PDF “strings,” where check box and radio button values are represented as PDF “names.” For our purposes, names and strings are the same; they are just encoded a little differently in the FDF. That is why forge_fdf takes two arrays of data: fdf_data_strings and fdf_data_names; pack them appropriately. By default, check boxes and radio buttons use the values “Yes” and “Off” to represent their true and false states, respectively. The form designer can choose an alternative to “Yes,” but “Off” always means false.

The arrays fields_hidden and fields_readonly have no role in this discussion, so you can ignore them.

Now things are beginning to come together. We have a PDF form, we have an FDF data file, and we can also see, above, that pdftk can merge these two files into a single, non-interactive PDF. Let’s talk about that.

PDF Form Filling and Flattening with Pdftk

The pdftk command for filling a PDF form looks like this:

```
pdftk <input PDF form> fill_form <input FDF data> output
<output PDF file> [flatten]
```

The PDF input, the FDF input, and the PDF output can be a filename, a hyphen (-), or “PROMPT.” Passing a hyphen into pdftk instead of an input filename causes pdftk to look for data on stdin. Similarly, passing a hyphen into pdftk instead of an output filename causes pdftk to return data on stdout. You can see we used this latter technique in the snippet, above. Finally, you can pass “PROMPT” into pdftk if you would like pdftk to ask you for the necessary filename at run time.

If you include the flatten output option, then all form field data is converted into static page elements. All of the interactive form features are removed, so the result is a plain old PDF that any viewer can handle. If you omit the flatten option, then form fields are filled to match your input data, but they also remain interactive. You can flatten a PDF form at any time by running:

```
pdftk filled_form.pdf output flattened_form.pdf flatten
```

So, these are the back-end pieces to our workaround for online PDF forms. We can take form data, cast it into FDF, merge it with the PDF form, and then serve it to the user. Now let’s look into creating the front-end HTML form. To help us along the way, we’ll use pdftk to discover PDF form field information.

PDF Form Field Discovery with Pdftk

A PDF form can have dozens of interactive fields. Manually mirroring these fields in HTML would be cumbersome and error-prone. Instead, let’s use one of pdftk’s reporting features. You can learn everything you need to know about your PDF’s interactive form fields by running:

```
pdftk form.pdf dump_data_fields > form.pdf.fields
```

This will create an easily parsible plain text report on your form's fields. The output might look like this:

```
---
FieldType: Text
FieldName: name_last
FieldNameAlt: Last Name
FieldFlags: 8392706
FieldValue:
FieldJustification: Left
FieldMaxLength: 200
---
FieldType: Button
FieldName: previous1
FieldFlags: 0
FieldJustification: Left
FieldStateOption: Off
FieldStateOption: Yes
---
FieldType: Choice
FieldName: select_one
FieldFlags: 4587520
FieldValue: a
FieldValueDefault: c
FieldStateOption: a
FieldStateOption: b
FieldStateOption: c
---
```

You can see that the field named title has a maximum length of 200 characters, that a button named previous1 has two possible states: Off and Yes, and a combo box named select_one has three possible states: a, b, and c. Note that push buttons, check boxes and radio buttons all have a FieldType of Button. To tell them apart, you must consult the FieldFlags. Similarly, list boxes and combo boxes both have a FieldType of Choice. See section 8.6 of the PDF Reference for details on field flags and their meanings. We won't be bothering with them, here.

This plain text report should provide you with all the information you need to create an HTML interface to your form. For fun, let's use PHP to do this automatically. Here's a script that reads this text report and generates an HTML form to suit. If you added a "Short Description" to each field in Acrobat, then that text will appear as the FieldNameAlt entry in our report. Our script will use this information, if present, to label the HTML field.

```
<?php

// this function loads a data file created using pdftk dump_data_fields
function
load_field_data( $field_report_fn )
{
  $ret_val= array();

  $fp= fopen( $field_report_fn, "r" );
  if( $fp ) {
    $line= '';
    $rec= array();
    while( ($line= fgets($fp, 2048))!== FALSE ) {
      $line= rtrim( $line ); // remove trailing whitespace
      if( $line== '---' ) {
        if( 0< count($rec) ) { // end of record
          $ret_val[]= $rec;
          $rec= array();
        }
        continue; // skip to next line
      }

      // split line into name and value
      $data_pos= strpos( $line, ':' );
      $name= substr( $line, 0, $data_pos+ 1 );
      $value= substr( $line, $data_pos+ 2 );

      if( $name== 'FieldStateOption:' ) {
        // pack state options into their own sub-array
        if( !array_key_exists('FieldStateOption:',$rec) ) {
          $rec['FieldStateOption:']= array();
        }
        $rec['FieldStateOption:'][]= $value;
      }
      else {
        $rec[ $name ]= $value;
      }
    }
    if( 0< count($rec)) { // pack final record
      $ret_val[]= $rec;
    }

    fclose( $fp );
  }

  return $ret_val;
}
// open our web page; the form action is a script we provide, below
echo '<html>
<head>
</head>
<body>
<form method="POST" action="pdf_form_fill.php">
<table>';
// create the file form.pdf.fields using pdftk's dump_data_fields
$field_arr= load_field_data( 'form.pdf.fields' );
foreach( $field_arr as $field ) { // iterate form fields
  echo '<tr><td>'; // one row per field
  if(array_key_exists('FieldNameAlt:', $field)) {
    // use human readable name, if available; you can add these in Acrobat
    echo $field['FieldNameAlt:'];
  }
  else {
    echo $field['FieldName:'];
  }
  echo '</td><td>';

  if( $field['FieldType:']== 'Text' ) {
    // construct an HTML text form field to match our PDF text form field;
    // cannot use periods in field names with PHP, so translate them to tildes
    echo '<input type="text" name="'.
        strtr($field['FieldName:'],'.','~'). '" ';
    // text field default value
    if(array_key_exists('FieldValueDefault:', $field)) {
      echo 'value="'. $field['FieldValueDefault:']. '" ';
    }
    // text field size and maxlength
    if(array_key_exists('FieldMaxLength:', $field)) {
      echo 'maxlength="'. $field['FieldMaxLength:']. '" ';
      if( $field['FieldMaxLength:']< 80 ) {
        echo 'size="'. $field['FieldMaxLength:']. '" ';
      }
      else {
        echo 'size="80" ';
      }
    }
    echo '>';
  }
  else if(array_key_exists('FieldStateOption:', $field)) {
    // use an HTML selection field for all other PDF form fields
    // (check boxes, radio buttons, list boxes, combo boxes);
    // cannot use periods in field names with PHP, so translate them to tildes
    echo '<select name="'.
        strtr($field['FieldName:'], '.', '~'). '">';
    foreach( $field['FieldStateOption:'] as $option ) {
      echo '<option>'.$option.'</option>';
    }
    echo '</select>';
  }
  echo "</td></tr>\n";
}
// close our table and our HTML page; don't forget the submit button
echo '</table>
<input type="submit" value="Create PDF">
                        </form>
</body>
</html>';
?>
```

Now, we need a companion script that takes this submitted data, packs it into the PDF form and serves it to the user. This script is the pdf_form_fill.php action in our above HTML form. It looks much like our earlier form filling example:

```
<?php
require_once( 'forge_fdf.php' );

$fdf_data_strings= array();
$fdf_data_names= array();

// funny thing; for our purpose, we can get away with packing everything
// everything into fdf_data_strings; that's handy
foreach( $_POST as $key => $value ) {
  // translate tildes back to periods
  $fdf_data_strings[ strtr($key, '~', '.') ]= $value;
}
// ignore these in this example
$fields_hidden= array();
$fields_readonly= array();

$fdf= forge_fdf( '',
                 $fdf_data_strings,
                 $fdf_data_names,
                 $fields_hidden,
                 $fields_readonly );

$fdf_fn= tempnam( '.', 'fdf' );
$fp= fopen( $fdf_fn, 'w' );
if( $fp ) {
  fwrite( $fp, $fdf );
  fclose( $fp );

  header( 'Content-type: application/pdf' );
  header( 'Content-disposition: attachment; '.
          'filename=filled_form.pdf' );

  passthru( 'pdftk form.pdf fill_form '. $fdf_fn.
            ' output - flatten' );
  unlink( $fdf_fn ); // delete temp file
}
else { // error
  echo 'Error: unable to write temp fdf file: '. $fdf_fn;
}
?>
```

When filling forms this way, it turns out you can pass everything into forge_fdf using the fdf_data_strings array; there's no need to use fdf_data_names. That's handy.

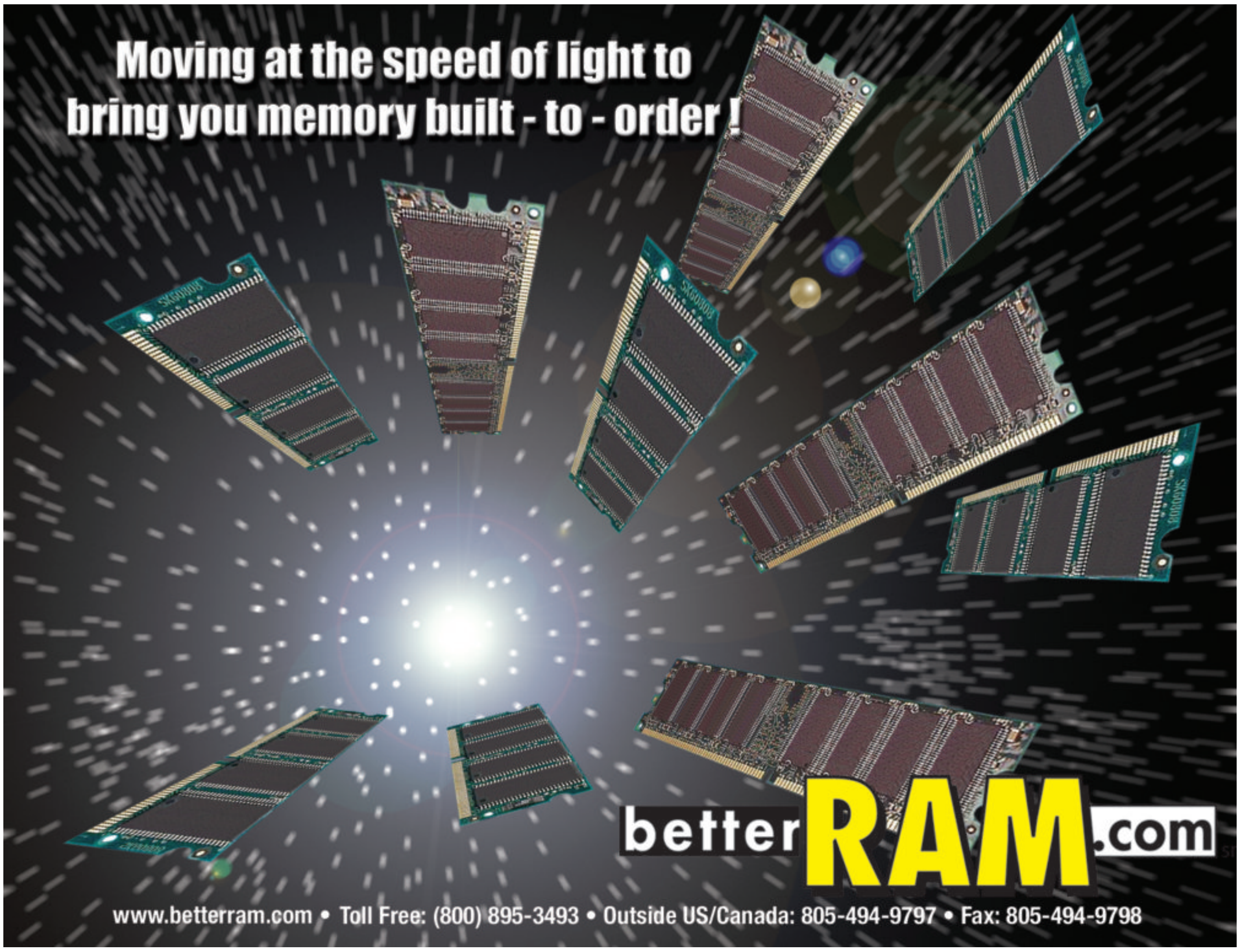
Now the Fun Begins

We have done it! We have created an HTML front-end to filling PDF forms. This is where the fun begins. You can now take everything you know about web programming, such as data validation and database access, and use it to fill PDF forms. Your users will be glad, too, because your resulting PDFs will work in alternative viewers such as Preview, and because you give them a filled-out PDF form for their records (which Adobe Reader does not provide).

To see an online example of these scripts, visit http://www.accesspdf.com/html_pdf_form/. You will also find the code, quoted in this article, available for download.

About The Author

Sid Steward is a longtime PDF service provider and software developer. He developed the free PDF Toolkit (http://www.AccessPDF.com/pdftk/) and wrote the book PDF Hacks (O'Reilly Media). You can reach him at sid@accesspdf.com.


Moving at the speed of light to bring you memory built - to - order !
betterRAM.com
www.betterram.com • Toll Free: (800) 895-3493 • Outside US/Canada: 805-494-9797 • Fax: 805-494-9798

Becoming a Blogger with iBlog

by Maria Langer

Blogs, the latest Internet craze, have become a wildly popular way for people of all backgrounds to share ideas, opinions, and information. A blog, which stands for Web log, is basically an online journal of frequently added, chronologically organized entries. Although most blog entries are short and to the point, they can be any length. They can also include hyperlinks, pictures, movies, and sounds.

Blogs, the latest Internet craze, have become a wildly popular way for people of all backgrounds to share ideas, opinions, and information. A blog, which stands for Web log, is basically an online journal of frequently added, chronologically organized entries. Although most blog entries are short and to the point, they can be any length. They can also include hyperlinks, pictures, movies, and sounds.

I got my first look at blogging about a year ago. I'd just renewed my .Mac membership and was surfing the .Mac site for software perks. That's where I downloaded a special version of iBlog for .Mac users. iBlog offered a very Mac-like interface for creating blogs, blog categories, and entries. It then enabled users to publish their blogs on their .Mac Web space on Apple's server. I downloaded iBlog, set it up, and started publishing my own blog. Within a remarkably short time, I was hooked.

iBlog makes it easy for me to share my thoughts with Web site visitors without HTML coding or fiddling around with Dreamweaver (my current authoring tool of choice). iBlog entries are automatically put in chronological order, with the most recent on top. Old entries are automatically archived by date. All I have to do is type in my entry and publish it. iBlog does the rest.

If you want to start your own blog and you don't feel like putting a lot of effort into programming or paying a blog hosting site to host your blog, iBlog is the answer. Here's how you can get started.

Download and Install iBlog

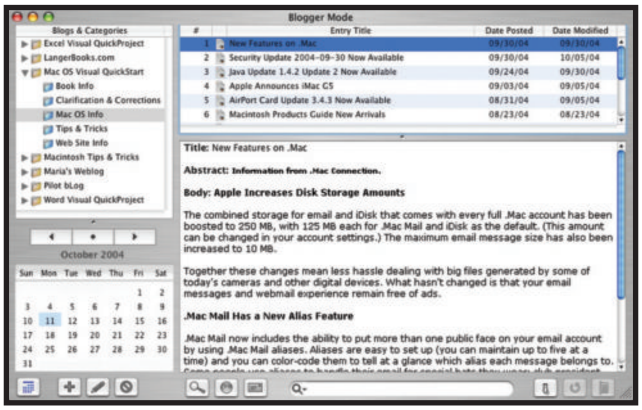
That special version of iBlog that I downloaded from .Mac is no longer available. That's the bad news. The good news is that newer, more feature-packed versions have been released since then. And although the software is distributed as shareware with a 2-week trial period, it only costs about $20 to buy. (Payment is in Indian rupees, so the exact amount varies.) You can download a copy from the Lifli Software Web site at www.lifli.com.

iBlog comes as a disk image. Open it up and drag the iBlog icon from the disk image's folder to your hard disk to copy it there. That's all there is to it.

Using iBlog creates an iBlog folder in ~username/Library/Application Support/. Depending on how you configure iBlog, it may also create an iBlog folder in ~username/Sites/. So if you decide later that you don't want to use iBlog after all, just delete the application and the two iBlog folders. That'll remove it and its data files from your computer.

Getting Started

Double-click the iBlog application icon. iBlog opens and displays its Blogger Mode window. **Figure 1** shows what my well-established iBlog window looks like. When you first launch iBlog, this window will be empty.


Blogger Mode

Blogs & Categories
- Excel Visual QuickProject
- LangerBooks.com
- Mac OS Visual QuickStart
  - Book Info
  - Clarification & Corrections
  - Mac OS Info
  - Tips & Tricks
  - Web Site Info
- Macintosh Tips & Tricks
- Maria's Weblog
- Pilot bLog
- Word Visual QuickProject

#	Entry Title	Date Posted	Date Modified
1	New Features on .Mac	09/30/04	09/30/04
2	Security Update 2004-09-30 Now Available	09/30/04	10/05/04
3	Java Update 1.4.2 Update 2 Now Available	09/24/04	09/30/04
4	Apple Announces iMac G5	09/03/04	09/05/04
5	AirPort Card Update 3.4.3 Now Available	08/31/04	09/05/04
6	Macintosh Products Guide New Arrivals	08/23/04	08/23/04

Title: New Features on .Mac

Abstract: **Information from .Mac Connection.**

Body: Apple Increases Disk Storage Amounts

The combined storage for email and iDisk that comes with every full .Mac account has been boosted to 250 MB, with 125 MB each for .Mac Mail and iDisk as the default. (This amount can be changed in your account settings.) The maximum email message size has also been increased to 10 MB.

Together these changes mean less hassle dealing with big files generated by some of today's cameras and other digital devices. What hasn't changed is that your email messages and webmail experience remain free of ads.

.Mac Mail Has a New Alias Feature

.Mac Mail now includes the ability to put more than one public face on your email account by using .Mac Mail aliases. Aliases are easy to set up (you can maintain up to five at a time) and you can color-code them to tell at a glance which alias each message belongs to.

October 2004

Sun	Mon	Tue	Wed	Thu	Fri	Sat
					1	2
3	4	5	6	7	8	9
10	11	12	13	14	15	16
17	18	19	20	21	22	23
24	25	26	27	28	29	30
31						

Fig 1: iBlog Blogger Mode with lots of blogs, categories, and entries.

(iBlog has two modes: Blogger Mode and Reader Mode. You use Blogger Mode to create and publish blogs. You use Reader Mode to read other people's syndicated blogs. This article covers Blogger Mode only.)

iBlog's interface should look familiar. It's a lot like iPhoto, iTunes, and other Apple software. A pane on the left side of the window lists blogs (gold folder icons) and their categories (blue folder icons). When you click a blog or category, a list of its entries appears to the right. When you click an entry name, the entry appears in a window beneath it. A calendar beneath the Blogs & Categories list offers a way to view entries by date. Click a date to see its blogs.

Create a Blog

The first step to using iBlog is to create a blog. I like to think of a blog as an online book. You need to create the book before you can fill it. Although I have lots of blogs, you really only need one to be a blogger.

Click the Add New Blog/Category/Entry button at the bottom-left of the window **(Figure 2)**. Choose New Blog from the menu that pops up. A dialog sheet with options for the new blog appears.

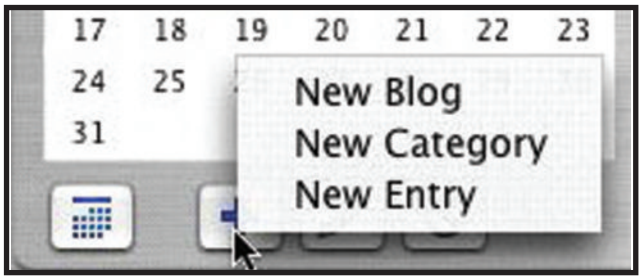
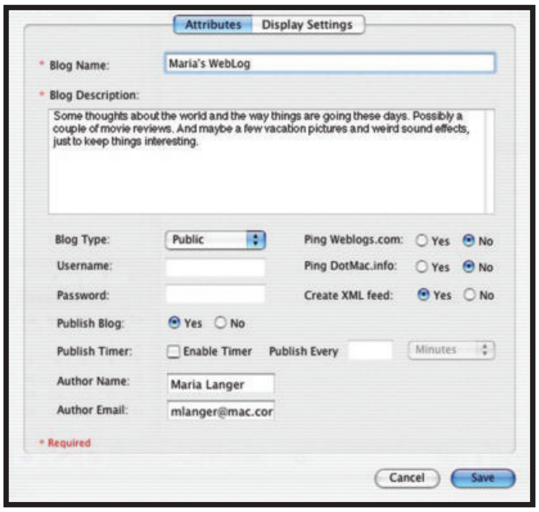


Fig 2: The New Blog/Category/Entry button displays a menu.

If necessary, click the Attributes button **(Figure 3)**. You must enter information in the Blog Name and Blog Description fields. This information is used for the Web pages iBlog creates automatically for you.


Attributes | Display Settings

* Blog Name: Maria's WebLog

* Blog Description: Some thoughts about the world and the way things are going these days. Possibly a couple of movie reviews. And maybe a few vacation pictures and weird sound effects, just to keep things interesting.

Blog Type: Public

Ping Weblogs.com: Yes [✓ No]

Username:

Ping DotMac.info: Yes [✓ No]

Password:

Create XML feed: [✓ Yes] No

Publish Blog: [✓ Yes] No

Publish Timer: Enable Timer Publish Every Minutes

Author Name: Maria Langer

Author Email: mlanger@mac.cor

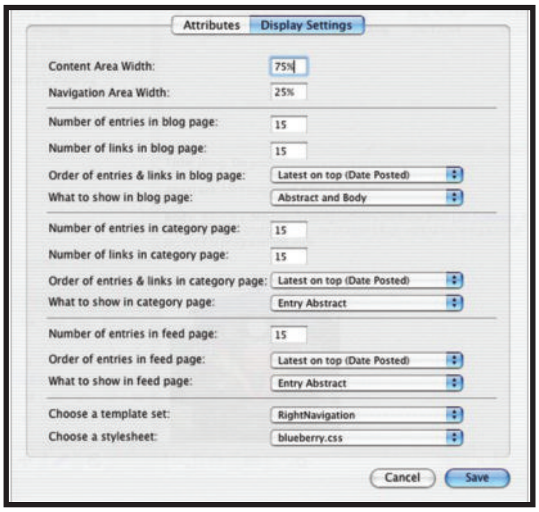
* Required

Cancel Save

Fig 3: Blog Attributes include the Blog name, description, and other settings.

There are a few other important settings in this dialog. Make sure Blog Type is set to Public and that the Publish Blog option is set to Yes. The Author Name appears at the bottom of the page in a copyright notice and the Author Email is used for a feedback link.

Click the Display Settings button **(Figure 4)**. This dialog sheet determines how the pages of your blog will appear and offers some customization options. iBlog has many other powerful customization options, as I'll discuss briefly later. For now, just set the What to show in blog page option to Abstract and Body.



Content Area Width:	75%
Navigation Area Width:	25%
Number of entries in blog page:	15
Number of links in blog page:	15
Order of entries & links in blog page:	Latest on top (Date Posted)
What to show in blog page:	Abstract and Body
Number of entries in category page:	15
Number of links in category page:	15
Order of entries & links in category page:	Latest on top (Date Posted)
What to show in category page:	Entry Abstract
Number of entries in feed page:	15
Order of entries in feed page:	Latest on top (Date Posted)
What to show in feed page:	Entry Abstract
Choose a template set:	RightNavigation
Choose a stylesheet:	blueberry.css

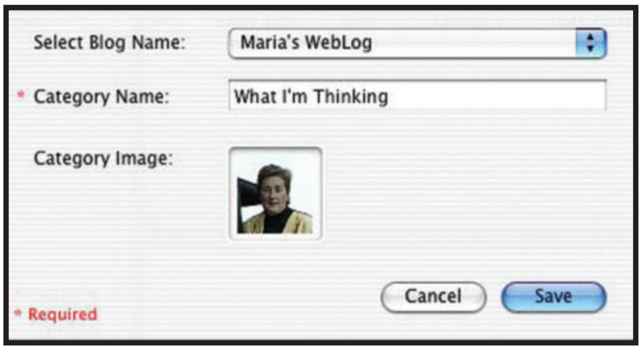
Fig 4: Display settings offer some page customization options.

When you click Save, your new blog appears as a gold folder in the Blogs and Categories list. You can repeat these steps to create multiple blogs if you like.

Create a Category

If a blog is like a book, then a category is like a chapter in the book. Although you don't have to create categories, I recommend it. Categories offer a way to organize entries in a way other than by date. iBlog creates category pages that list entries for just that category.

Click the Add New Blog/Category/Entry button (Figure 2) and choose New Category from the menu that pops up. A dialog sheet for the new category appears **(Figure 5)**. If you have more than one blog, choose the name of the blog the category belongs to from the Select Blog Name pop-up menu. Enter a name for the category in the Category Name box. And, if you want to get fancy and assign an image to the category, drag the icon for an image file from the Finder to the Category Image well. The image appears in the well.

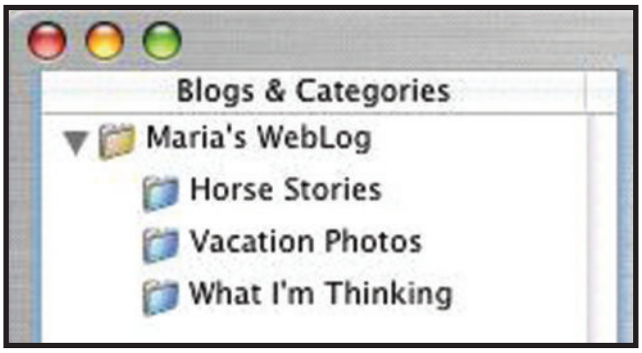


Select Blog Name:	Maria's WebLog
* Category Name:	What I'm Thinking
Category Image:	

Fig 5: A the category settings sheet with some information entered for a new category. (Yes, that's my mug.)

When you click Save to save your settings, the category appears in the Blogs & Categories list as a blue folder beneath its blog. If you can't see it, click the triangle beside the blog folder to display it.

Of course, you can repeat this process for as many categories as you'd like to create. **Figure 6** shows the Blogs & Categories list with three categories created.



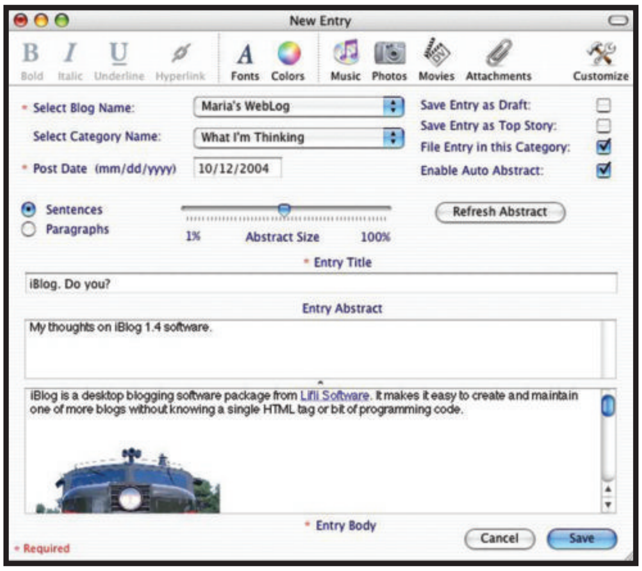
Blogs & Categories
- Maria's WebLog
  - Horse Stories
  - Vacation Photos
  - What I'm Thinking

Fig 6: A single blog with several categories.

Create an Entry

If a blog is a like book and a category is like a chapter, then an entry is like a page. Well, something like a page. Entries are what make up the contents of your blog.

Click the Add New Blog/Category/Entry button **(Figure 2)** and choose New Entry. Use the New Entry form window that appears to create your entry **(Figure 7)**.



* Select Blog Name:	Maria's WebLog
Select Category Name:	What I'm Thinking
* Post Date (mm/dd/yyyy)	10/12/2004
Save Entry as Draft:	☐
Save Entry as Top Story:	☐
File Entry in this Category:	✓
Enable Auto Abstract:	✓

[✓ Sentences] Paragraphs

* Entry Title
iBlog. Do you?

Entry Abstract
My thoughts on iBlog 1.4 software.

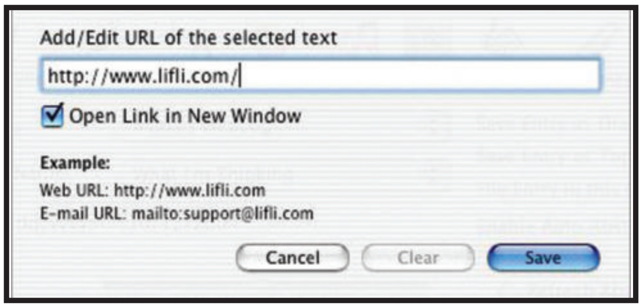
iBlog is a desktop blogging software package from Lifli Software. It makes it easy to create and maintain one of more blogs without knowing a single HTML tag or bit of programming code.

Fig 7: A blog entry.

This window is pretty self-explanatory. You set options at the top of the window to determine which blog and category the entry should be associated with. The Post Date is entered automatically, but you can set it to a different date to change the order in which the entry appears or set it to publish at a later date. Then you enter the entry's title in the Entry Title box and the entry's body in the big box at the bottom of the window.

The Entry Abstract is a shorter version of the entry. The Auto Abstract option, which is relatively new, will create an abstract based on the entry body. I don't like this feature, so I turn it off. Instead, I manually enter a one-sentence description of the entry in the Entry Abstract box. That's just the way I use iBlog, though. You may prefer real abstracts and think the Auto Abstract feature is great. Give it a try and decide for yourself. It's easy enough to turn off if you decide you don't like it.

The toolbar at the top of the window has buttons for the usual formatting options, as well as options to add features to your blog entry. For example, you can select text in the blog window and click the Hyperlink button to display a dialog like the one in **Figure 8** for turning the selected text into a hyperlink. Enter the URL, turn on the check box if you want the page to open in a new Web browser window, and click Save.


Add/Edit URL of the selected text

http://www.lifli.com/

☑ Open Link in New Window

Example:
Web URL: http://www.lifli.com
E-mail URL: mailto:support@lifli.com

Fig 8: Use this dialog to create a hyperlink.

If you use iPhoto to organize your photos and other graphics, you can click the Photos button to insert a photo into the entry. Choose an iPhoto album in the dialog that appears **(Figure 9)** and click the photo you want to insert. The Width and Height boxes work a little weird; the size of the photo is determined by the height and the width is adjusted proportionally. Click Import to insert the image into iBlog. Once the image is inserted, you can double-click it to display a dialog sheet **(Figure 10)** for setting other image options. Although you won't see those options in the iBlog entry window **(Figure 7)**, you will see them applied on the Web page iBlog creates **(Figure 13)**.

Long Distance

2.9¢ Per Minute!

Straight 6 second billing increments

Excellent rates on intrastate, intralata/toll calls and international calling with no term contract.

Toll Free (800/888/877/866) service, same low per minute rate for incoming calls.

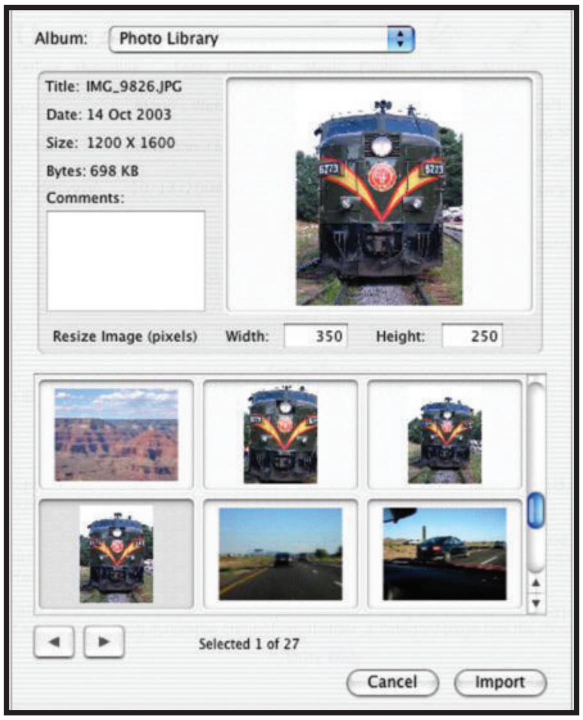
10 cents per minute calling card.

Detailed billing directly from OPEX.

Quality electronic and telephone customer support. No monthly billing fee if your bill is over $20.00 each month.

(NOTE: $2.00 billing fee is charged when your bill is under $20.00.)

www.lowcostdialing.com


Album: Photo Library

Title: IMG_9826.JPG

Date: 14 Oct 2003

Size: 1200 X 1600

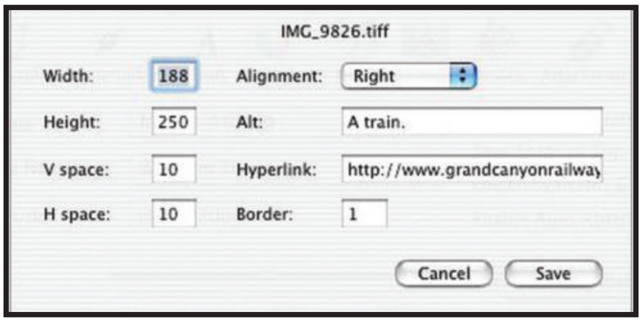
Bytes: 698 KB

Comments:

Resize Image (pixels) Width: 350 Height: 250

Selected 1 of 27

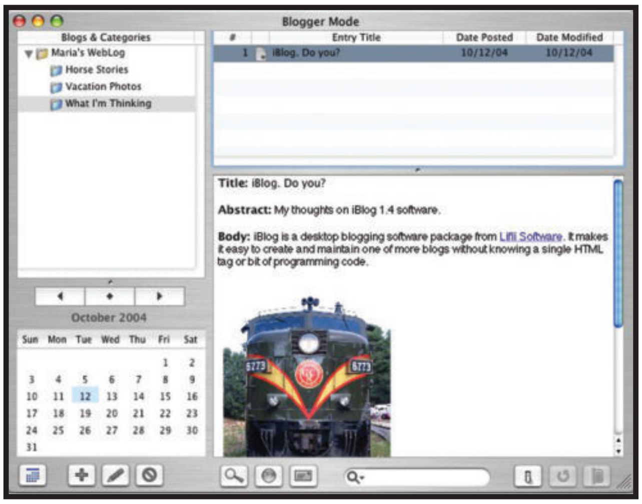
Fig 9: Use this dialog to insert an image from your iPhoto photo library.


IMG_9826.tiff

Width: 188
Alignment: Right
Height: 250
Alt: A train.
V space: 10
Hyperlink: http://www.grandcanyonrailway
H space: 10
Border: 1

Fig 10: Then use this dialog to set image options.

When you click the Save button to save your entry, it appears in the Blogger Mode window **(Figure 11)**.


Blogger Mode

#	Entry Title	Date Posted	Date Modified
1	iBlog. Do you?	10/12/04	10/12/04

Blogs & Categories
- Maria's WebLog
  - Horse Stories
  - Vacation Photos
  - What I'm Thinking

October 2004

Sun	Mon	Tue	Wed	Thu	Fri	Sat
					1	2
3	4	5	6	7	8	9
10	11	12	13	14	15	16
17	18	19	20	21	22	23
24	25	26	27	28	29	30
31						

Title: iBlog. Do you?

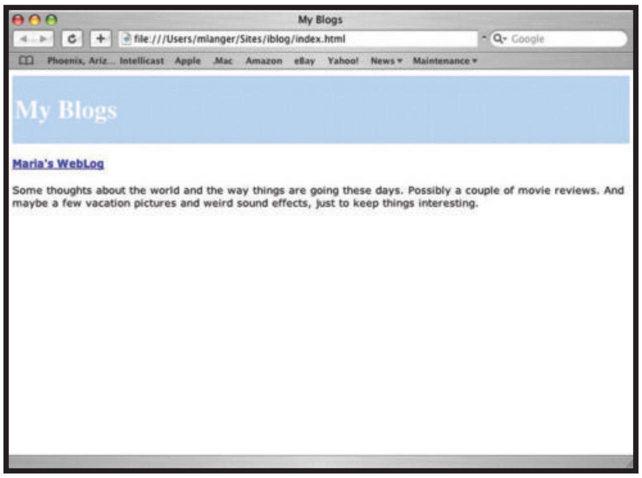
Abstract: My thoughts on iBlog 1.4 software.

Body: iBlog is a desktop blogging software package from Lifli Software. It makes it easy to create and maintain one of more blogs without knowing a single HTML tag or bit of programming code.

Fig 11: A completed blog entry in the Blogger Mode window.

Preview

You can use iBlog's preview feature to get a look at your entry before it's published. Just click the Preview button (which looks like a magnifying glass) in the bottom of the Blogger Mode window. iBlog launches your default Web browser and displays the Home page for your blogs **(Figure 12)**.


My Blogs

file:///Users/mlanger/Sites/iblog/index.html

Google
Phoenix, Ariz... Intellicast Apple .Mac Amazon eBay Yahoo! News Maintenance

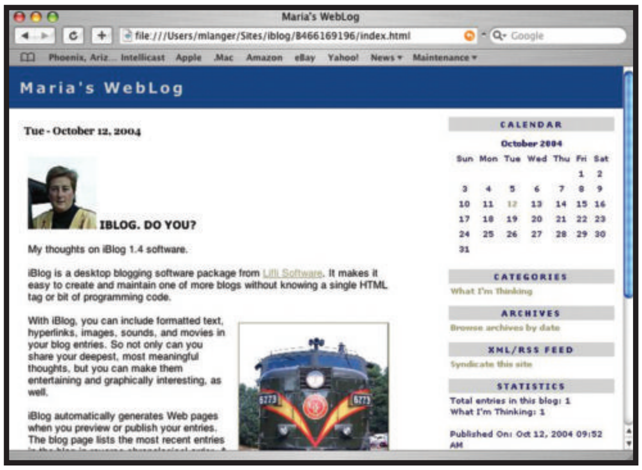
My Blogs

Maria's WebLog

Some thoughts about the world and the way things are going these days. Possibly a couple of movie reviews. And maybe a few vacation pictures and weird sound effects, just to keep things interesting.

Fig 12: iBlog creates a home page for all of your blogs.

Click the link for a blog name. The blog page for that blog appears, with the entry you created at the top **(Figure 13)**. As you can see, iBlog also creates a navigation bar complete with calendar and links. Try out the links to see what other pages iBlog created. You should be impressed. I was. iBlog is a heck of a lot quicker than creating all those pages manually.


Maria's WebLog
file:///Users/mlanger/Sites/iblog/B466169196/index.html
Google
Phoenix, Ariz... Intellicast Apple .Mac Amazon eBay Yahoo! News Maintenance

Maria's WebLog

Tue - October 12, 2004

IBLOG. DO YOU?

My thoughts on iBlog 1.4 software.

iBlog is a desktop blogging software package from Lilli Software. It makes it easy to create and maintain one of more blogs without knowing a single HTML tag or bit of programming code.

With iBlog, you can include formatted text, hyperlinks, images, sounds, and movies in your blog entries. So not only can you share your deepest, most meaningful thoughts, but you can make them entertaining and graphically interesting, as well.

iBlog automatically generates Web pages when you preview or publish your entries. The blog page lists the most recent entries

CALENDAR

October 2004

Sun	Mon	Tue	Wed	Thu	Fri	Sat
					1	2
3	4	5	6	7	8	9
10	11	12	13	14	15	16
17	18	19	20	21	22	23
24	25	26	27	28	29	30
31						

CATEGORIES

What I'm Thinking

ARCHIVES

Browse archives by date

XML/RSS FEED

Syndicate this site

STATISTICS

Total entries in this blog: 1
What I'm Thinking: 1

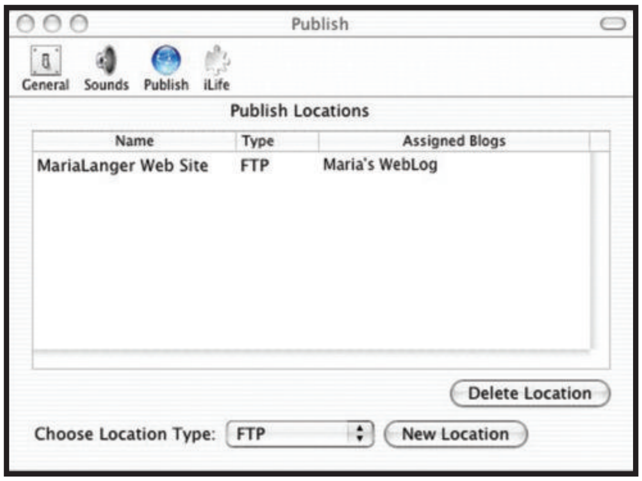
Published On: Oct 12, 2004 09:52 AM

Fig 13: Previewing an entry.

Publish

Publishing with iBlog is just as easy as previewing. But first you have to set up a publish location and match the blog to the location. You do this just once and iBlog remembers your settings for each time you want to publish.

Choose Preferences from the iBlog menu. In the dialog that appears, click the Publish button, you should see a dialog like the one in **Figure 14**, but it'll be empty.



Publish Locations

Name	Type	Assigned Blogs
MariaLanger Web Site	FTP	Maria's WebLog

Choose Location Type: FTP

Fig 14: Publish preferences with a publish location already set up.

Choose a location type from the pop-up menu at the bottom of the screen **(Figure 15)**. For this example, I'll choose FTP to publish on an existing Web server. Then click New Location.

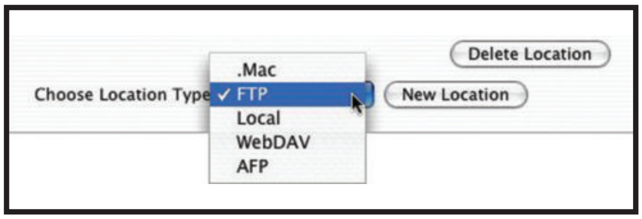
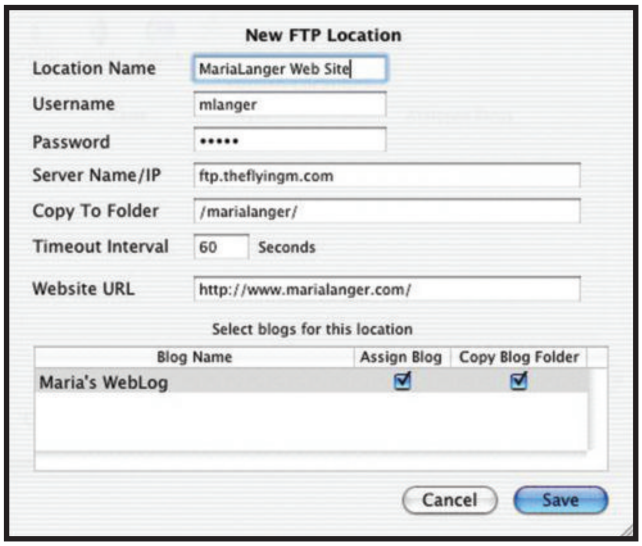


Fig 15: Use this menu to choose a location type.

Use the New FTP Location dialog **(Figure 16)** to enter the usual information for accessing the appropriate FTP server. Then turn on the check boxes beside the name of the blog you want to publish to that site. As you might imagine, if you have multiple blogs, you can publish them to multiple locations. Click Save and your settings are saved as a Publish Location **(Figure 14)**. You can close the Publish preferences window.


New FTP Location

Location Name	MariaLanger Web Site
Username	mlanger
Password	•••••
Server Name/IP	ftp.theflyingm.com
Copy To Folder	/marialanger/
Timeout Interval	60 Seconds
Website URL	http://www.marialanger.com/

Select blogs for this location

Blog Name	Assign Blog	Copy Blog Folder
Maria's WebLog	✓	✓

Fig16: Enter FTP site information in this dialog.

Back in the Blogger Mode window, click the Publish button, which looks like a little planet Earth. If you have multiple blogs, a dialog appears so you can specify which blog(s) you want to publish. But if you have just one blog, iBlog goes online and uploads all the pages and related files it has created for your blog. While it's doing this, it displays a Publish Status dialog. When the dialog disappears, your Web browser launches, displaying the blog page that has been uploaded to your site.

Simple, no?

Going the Next Step: Customization

While I admit that iBlog isn't the perfect solution, it's pretty darn close for a non-programmer like me. Although the pages it creates are perfectly acceptable "right out of the box" as discussed here, iBlog can be customized to change the appearance and layout of pages and the items that appear in the navigation bar. You do this by creating or customizing templates, style sheets, and navigation bar contents. A complete discussion of this is far beyond the scope of this article, but if you're familiar with HTML and CSS, you already have the knowledge you need to make the changes. If you want a good idea of what can be done with iBlog to create a truly customized site, check out the support Web site I've created for my books, www.langerbooks.com. That entire site is iBlog-generated and it's a heck of a lot easier to maintain than the old sites I managed with Dreamweaver.

iBlog. Do you?

As you can see, iBlog offers a simple solution for someone interested in blogging. As it continues to be developed and refined, it may well become a leading product for Mac bloggers everywhere.

MT

About The Author

Maria Langer is a freelance writer who has been writing how-to books and articles for Macintosh users since 1992. You can read her blogs at homepage.mac.com/mlanger/iblog and visit her on the Web at www.marialanger.com.

Which color picker would you rather use?
This...
Or this?
The HVC Color Composer, a plug-in for Adobe® Photoshop®
● A powerful new artist-friendly color space, HVC
● Create thousands of brilliant palettes, with one click
● Save palettes as Photoshop® palette files
Instantly formulates thousands of harmonically resonant palettes designed specifically for your artwork. Guaranteed!
Master Colors™
www.master-colors.com
Winner of the 2004 Seybold Hot Picks award.
HOT PICKS

Active Directory & Mac OS X

Fitting in, not standing out.

By Michael Bartosh

Ancient History

Many in the Mac IT community will recall the Windows NT revolution and its impact on Apple's place in the market. Microsoft began heavily promoting NT, selling its centralized management capabilities (which appealed to the rapidly professionalizing field of desktop IT management) as well as Apple's seemingly delayed reaction. You could make a case that Apple's strategy in this area was unsuccessful and may have had some degree of negative impact on its overall market share. In 1993 when Windows NT 3 was introduced Apple enjoyed a nearly 10% market share. By 1997, 1 year after NT4's introduction, Apple's market share had declined to around 4.4 %, and Microsoft had made great inroads, particularly in the server space, into some of Apple's core markets. Rather than embracing and fitting into the infrastructures that its customers had chosen, Apple continued to employee strategies that have not been proven successful in the enterprise marketplace.

History Reloaded

In 2003, Active Directory (Microsoft's Directory Services product) was in much the same position that Windows NT was in the late 1990's. In its second revision (as a part of the Windows 2003 server product; the first was included with Windows 2000) Active Directory had achieved a deep level of penetration into several of Apple's core markets. Once again, Apple had a decision to make—embrace the idea of heterogeneous multi-vendor networks driven by ad-hoc market standards, or retreat to its less successful roots. Luckily in 10.3 Apple seems to have at least tentatively chosen the former path, engineering into the Mac OS X specific capabilities for integration with Active Directory. This is a new and bold step forward for Apple, and something that goes a long way towards bring legitimacy to Apple in enterprise and institutional markets.

The Active Directory Plug-in: Features

Active Directory Integration is nothing new to Mac OS X—previous to Panther a certain level of interoperability was feasible. Feasible, however implies neither secure nor straightforward, to say nothing of the simplicity Mac users are accustomed to. The problem with pre-10.3 configurations is that they were tedious, complex, and relatively insecure. There was no standardized or consistent integration procedure, and the result was typically a mess of hacks, work-arounds and duct tape. Additionally, really complete solutions tended towards intrusiveness, often requiring extensive changes to the Active Directory.

Panther's Active Directory Plug-in provides a simpler and more full-featured platform for directory services configuration. It supports a number of line-item features (which I'll cover) but its single most important feature is not a line item at all, but a basic architectural principal. It seeks to emulate a Windows client as much as possible. Its communication with the Windows domain is nearly indistinguishable from its Microsoft-bred counterparts. Its goal is to integrate as seamlessly as feasible into the infrastructure that many of Apple's customers have chosen, requiring no infrastructure changes and little massaging or special treatment. Beyond that strategic goal, its specific features are many.

- Single Sign On: The Active Directory plug-in leverages the extensive client-side work Apple has pursed in order to effectively make use of the Kerberos authentication protocol. Kerberos is Active directory's native authentication platform, and effective Active Directory integration demands a mature Kerberos environment. When a User logs into Mac OS X using an Active Directory account they are granted a TGT (ticket granting ticket) which ultimately allows them to access Kerberized services— including Exchange's Outlook Web Access— without having to authenticate again.

- Windows Home Directories: In its default configuration, the Active Directory Plug-in obtains the location of the User's home directory (in the homeDirectory attribute of the User's Active Directory account) and puts it into the User's Dock so that it is easily accessible. It can also be configured, however (as covered later in this article) to use the SMB-mounted Windows home share as a Mac OS X network home directory.

- Password Policy Enforcement: In Jaguar password policies set in Active Directory were not effectively enforced. Although the situation isn't perfect in Panther, users are allowed to change expired passwords on log-in, and password changing is effectively integrated into the user experience, allowing users to change their Active Directory password with either the passwd command line utility or the Accounts pane of the System Preferences application.

- Multiple domain authentication: Active Directory is capable of scaling to very large deployments. These large deployments often form forests made up of multiple domains. Apple's Active Directory plug-in optionally allows for users from any domain in a forrest (even domains in a different namespace— for instance a domain called pantherserver.org in a forest called apple.com) to log into the Mac in question.

- Active Directory Group Support: Active Directory keeps track of groups in a way that is very different from Mac OS X (and most other Unix-based Operating Systems). This made it very difficult in Jaguar to leverage the extensive group management capabilities of Active Directory. Panther's Active Directory Plug-in is designed specifically to interpret Active Directory group data, manipulating it in a way that makes it useful to Mac OS X.

- Delegated Administration: Active Directory largely operates on the principal of delegated administration— granting certain administrative privileges in a granular fashion to users who are not domain administrators (users, for instance, are often delegated the authority to add their workstations to the domain). The Active Directory Plug-in is friendly to this concept, optionally allowing one or more Active Directory groups to administer the Mac OS X workstation. Administrative capabilities can additionally be granted to specific users (rather than groups) the same way they would be on the Windows platform, by setting the Managed By attribute in the Windows Active Directory Users and Computers application.

- Disconnect Behavior: Panther displays far better disconnect behavior when domain resources are not available. In Jaguar the user experience as Open Directory (Apple's Directory Services infrastructure) struggled to re-connect to missing directory domains was painful to say the least, with difficult timeouts that left the client useful for extended periods of time. This has improved in Panther to the extent that Active Directory user accounts can even be cached locally, so that the user is able to log in even if the domain is not available. When the domain is available password policies (such as expiry) are enforced.

We mean it this time.

DEV DEPOT

DevDepot has it all!

Get More out of your Mac!

DevDepot sells the tools, toys and technology to put more muscle into your Mac. Visit our online store today for special offers and great new products. **www.devdepot.com**

Television is about to get a whole lot more interesting.

NOW ONLY $324.99

EyeTV 200

The future of Television

Watch TV on your Mac!

Pause live tv! Record, edit and archive!

go **digital**

Convert your analog video tapes to digital in realtime, then burn them to DVD! *

Watch TV on your Mac

EyeTV features a 124-channel cable-ready analog tuner and DVD quality MPEG-2 video encoder.

Record TV on your Mac and Archive to Disk

Collect and organize your favorite shows to **watch whenever you want**, then update video content to DVD.* (Toast 6 Titanium required).

Don't miss a thing

Use EyeTV's Electronic Program Guide to find exactly what you are looking for, and program EyeTV to record it. **Program EyeTV from anywhere via the Internet.**

NEW!

New Export Functionality

EyeTV now lets you export to iMovie®, iDVD® and DVD Studio Pro®, making it easier to create professional quality recordings.

from elgato

Features and Benefits

- Record TV otn Your Mac
- Edit Out Unwanted Content
- Archive Recorded TV To DVD*
- MPEG-2 Video Encoding
- Digitize Analog Video
- The Speed And Power Of **Firewire**

More Great Products!

Buy Today and Save!

SYNCBOX Portable USB Data Copier

ONLY $43.99

Transfer data from your USB devices with the touch of a button.

✳ Never run out of memory space again!

- No computer required, 100% portable
- Sleek and compact design
- Easy to use, one button operation
- Complies with USB 1.1 specifications
- Supports both USB 1.1 and USB 2.0
- Uses 3 AAA alkaline batteries (not included)

DIGITAL CAMERAS

FLASH DRIVES

EASILY GET DATA FROM:

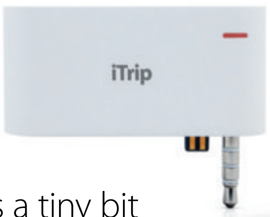
- Storage Devices
- Digital Cameras
- Flash Drives
- MP3 Players
- Card Readers
- AND MORE!

FITS IN YOUR HAND!

from macally™

iTripmini FM Transmitter

Listen in your car!



The iTrip mini was **designed exclusively for the iPod mini.**

NO BATTERIES NEEDED!
The iTrip mini only needs a tiny bit of power that it gets directly from your iPod mini.

ONLY $39.99

Its form **matches all the curves of the iPod mini,** and sounds even sweeter!

a Griffin Technology

iMic USB Audio Interface

The iMic is a must-have device for people who are serious about high quality audio.

Connect virtually any sound device to your iBook, PowerBook, PowerMac or other Mac or PC with a USB port.

iMic SUPPORTS:

- Line Level Microphones
- Mic Level Microphones
- Multimedia Devices
- Headsets
- Communications Devices

ONLY $34.99

from Griffin Technology

DEV DEPOT

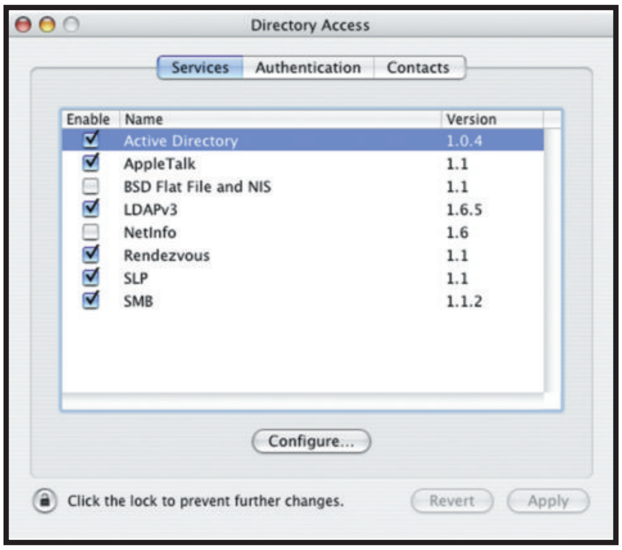
www.devdepot.com

- UniqueID Generation: One of the biggest challenges of integrating any Unix operating system with Active Directory is the UniqueID. This integer (sometimes called uid) is used to uniquely identify Unix accounts. Active Directory supports several unique identifiers (among them guids and sids) but they are 128 bit hex identifiers. This results in quite a bit of difficulty during integration, since a user account without a UniqueID isn't really a user account as far as Mac OS X is concerned. The Active Directory Plug-in works around this issue, generating an integer UniqueID from the 128 bit hex guid. This conversion (done according to Microsoft's specification at blah) produces an integer UniqueID that is both consistent throughout the domain and Unique among all user accounts. The only real downside to this process is that the converted UniqueID's are very large, and not all Applications deal with them correctly.

- Domain controller preference: When the Active Directory Plug-in joins a domain it attempts to locate the nearest domain controller using site policy published to DNS. Unfortunately site policy is not always configured correctly, so the Active Directory Plug-in allows you to specify a preferred domain controller.

Configuring the AD Plug-in

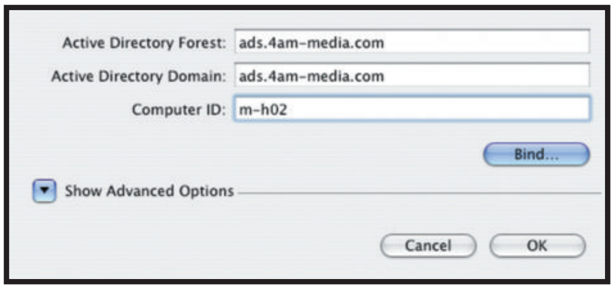
The Active Directory Plug-in, like most other end-user configurable Open Directory Plug-in's, is configured using the Directory Access application, which is located in an out of box Mac OS X install in /Applications/Utilities. Configuring Active Directory integration is as simple as starting Directory Access, choosing its Active Directory Plug-in, and clicking on the configure button, as seen in **Figure 1**.



Enable	Name	Version
✓	Active Directory	1.0.4
✓	AppleTalk	1.1
	BSD Flat File and NIS	1.1
✓	LDAPv3	1.6.5
	NetInfo	1.6
✓	Rendezvous	1.1
✓	SLP	1.1
✓	SMB	1.1.2

Figure 1. The Directory Access application.

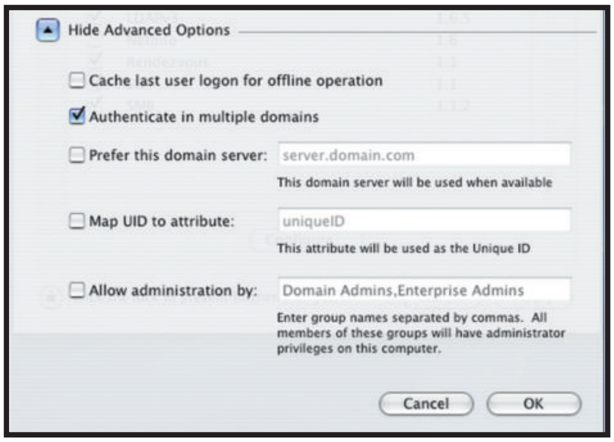
Configuring the Active Directory Plug-in results in the dialog seen in **Figure 2**. It is, in its basic form, extremely straightforward, prompting for a domain and forest to join, along with an ID for the computer. In the case of Mac OS X clients this computer ID should reflect local naming conventions and policies (machines are often named sequentially, or based on their user or physical location). Servers joining Active Directory should use the unqualified portion of their hostname. Homes.pantherserver.org, for instance, would have a computer ID of homes (this allows for easier single-sign on interoperability between the server an the Active Directory Kerberos environment).



Active Directory Forest: ads.4am-media.com
Active Directory Domain: ads.4am-media.com
Computer ID: m-h02

Figure 2. The Active Directory Plug-in's basic configuration dialog.

Optionally, clicking on the Show Advanced Options triangle reveals the interface pictured in **Figure 3**. This is where most (but not all) of the features listed earlier in this article are implemented.

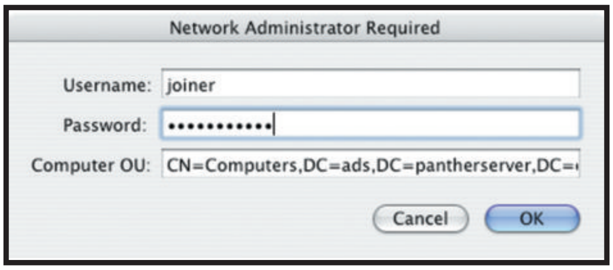


☐ Cache last user logon for offline operation
☑ Authenticate in multiple domains
☐ Prefer this domain server: server.domain.com
This domain server will be used when available
☐ Map UID to attribute: uniqueID
This attribute will be used as the Unique ID
☐ Allow administration by: Domain Admins,Enterprise Admins
Enter group names separated by commas. All members of these groups will have administrator privileges on this computer.

Figure 3. The Active Directory Plug-in's advanced configuration dialog.

Options from this interface (along with other interesting bits of data) are stored in the Active Directory Plug-in's configuration file (/Library/Preferences/DirectoryService/ActiveDirectory.plist) which is discussed in more depth later in this article.

When the desired options have been specified, you may select the Bind button. This results in an authentication dialog, pictured in **Figure 4**. This dialog accepts a container or organizational unit in Active Directory along with credentials required to add a computer account to it. This is an important concept— the graphical interface erroneously implies that Domain Administrator credentials are required to add the Mac to the Active Directory Domain. In reality, all you need to supply are the credentials of a user able to add computer accounts to the specified container or ou. As mentioned earlier this is a commonly delegated task, often left to users or low-level IT staff.



Username: joiner
Password: •••••••••••
Computer OU: CN=Computers,DC=ads,DC=pantherserver,DC=

Figure 4. Joining Active Directory requires that the specified credentials be able to add computer accounts to the indicated organizational unit or container.

Troubleshooting

Other than a thorough understanding of Active Directory, Open Directory, and directory services in general the two most important tools for troubleshooting Active Directory integration issues are network sniffers (like tcpdump and ethereal) and the Directory Service daemon's debug mode.

tcpdump is installed on Mac OS X (and most other Unix operating systems) so I most commonly use it to initially gather data, later examining that data using a graphical tool like ethereal. Kerberos data in particular looks largely like a bunch of hex over the wire, and ethereal can be a great help translating this data into something that's human readable.

```
big15:~ mab$ sudo tcpdump -w join.dump -i en1 port domain
or port 3268 or port kerberos or port kpasswd or port ldap
```

Comment

The work of the Active Directory plug-in is actually executed by the DirectoryService daemon, which produces very good logging data, particularly in the case of Active Directory interoperability. To turn debug logging in, you need to send the USR1 signal to the DirectoryService process. This begins logging to /Library/Logs/DirectoryService.debug.log. Active directory messages are prepended with the string ADPlugin:, so the log itself (which is very verbose) is easy to filter.

```
xsg5:~ tadmin$ sudo killall -USR1 DirectoryService
xsg5:~ tadmin$ tail -f
/Library/Logs/DirectoryService/DirectoryService.debug.log |
grep ADPlugin
2004-10-14 01:38:17 PDT - ADPlugin: Calling CustomCall
2004-10-14 01:38:17 PDT - ADPlugin:    Doing
CheckServerRecords......
2004-10-14 01:38:17 PDT - ADPlugin:           Good
credentials for joiner@ADS.4AM-MEDIA.COM
2004-10-14 01:38:17 PDT - ADPlugin:           No connection
in connection mgr for joiner@ ADS.4AM-MEDIA.COM@ads.4am-
media.com:389
2004-10-14 01:38:18 PDT - ADPlugin:           Secure BIND
Session with server w2k.ads.4am-media.com:389
2004-10-14 01:38:18 PDT - ADPlugin:       Processing Site
Search with found IP
2004-10-14 01:38:19 PDT - ADPlugin:           Added
connection to connection mgr joiner@ADS.4AM-
MEDIA.COM@ads.4am-media.com:389
2004-10-14 01:38:19 PDT - ADPlugin:       Found Default
Domain ads.4am-media.com
```

Turning on debug logging in the DirectoryService daemon. The debug log is easy to filter with grep.

DirectoryService debug logging remains enabled until the daemon is re-started or until it receives another USR1 signal. Sending a USR2 signal enables API logging, which logs every Open Directory API call to the system log (/var/log/system.log). USR2 logging is heavy-weight, and will automatically turn off after 5 minutes.

The User Experience

In its default configuration, users from Active Directory are allowed to log in to Mac OS X using several forms of their user name (in order to be as compatible as feasible with the Windows user experience.) John Doe for, for instance might be able to log in as jdoe, John Doe, jdoe@ads.pantherserver.org or ADS\jdoe. The user is given a local home location in the /Users directory and a Kerberos TGT (ticket granting ticket) is obtained on log-in. This means that users can access most domain resources— from kerberized file servers to Outlook Web Access (using Safari) without re-authenticating. In the client flavor of Mac OS X (Mac OS X Server's behavior differs) the user's SMB home directory (if it is listed in their user record) is placed in their dock and automatically mounted using NTLMv1 authentication (the TGT is not obtained early enough to mount it using Kerberos, which is far more secure).

Advanced Configuration

Some of the Active Directory Plug-in's most significant features are not available in its graphical interface. Most of these are available through the dsconfigad command, the AD Plug-in's command-line configuration interface. The Active Directory homeDirectory UNC, for instance, can be used as the Mac OS X home directory (rather than being mounted on the desktop) using the dsconfigad command's –localhome flag.

```
djou:~ djou$ dsconfigad -localhome disable
djou's Password:
Settings changed successfully
djou:~ djou$ dsconfigad -show

You are bound to Active Directory:
  Active Directory Forest     = ads.4am-media.com
  Active Directory Domain     = ads.4am-media.com
  Computer Account             = m-h02

Advanced Options
  Mount Style                  = smb:
```

Using dsconfigad, first to turn off the the default local home behavior, then to examine the Plug-In's configuration. When –localhome is disabled, user home directories are mounted late enough to support Kerberos authentication.

This disabled localhome behavior has two variants, controlled by the mountstyle flag. A mountstyle of SMB (the default configuration) interprets the UNC as an SMB URL plist, allowing Mac OS X to use it as an SMB home directory.

```
djou:~ djou$ dscl /Active\ Directory/ads.4am-media.com -
```

```
read /Users/winnie homeDirectory HomeDirectory
homeDirectory: \\w2k\homes\winnie
HomeDirectory: <home_dir><url>smb://w2k.ads.4am-
media.com/homes</url><path>winnie/</path></home_dir>
```

Coupled with the AD Plug-in's –localhome disable option, the SMB mount style interprites the Active Directory homeDirectory UNC as a Mac OS X HomeDirectory (a URL plist). Notice the case sensitivity here— the Active Directory attribute is called homeDirectory. It is used to produce the Mac OS X HomeDirectory.

Conversely a mountstyle of AFP interprets the UNC as an AFP URL. In general, AFP offers a better home directory experience than SMB, so this option has potential to improve the overall effectiveness of your infrastructure. In the vast majority of cases this is less than useful, though, since Microsoft's AFP Server is specifically not up to the task of supporting Mac OS X home directories. This setting becomes advantageous in two cases: when the home directory server is running the newest version of ExtremeZ IP (which features an AFP implementation that is far more capable than Microsoft's) or when the home directories are housed on Mac OS X Server. The latter case is a new and powerful option, implying that Windows clients will mount the same (Mac OS X-hosted) home directory using SMB that Mac OS X clients mount using AFP. The homeDirectory UNC in the AD User record in this case actually specifies a share on Mac OS X Server. This allows you to leverage Apple's compelling and relatively affordable server solutions, even in a Windows-centric infrastructure. That this is feasible is a testament to the deep level of integration that Mac OS X Server is capable of. The only real down side is that in Panther this does limit administrators Unix user-group-other permissions, rather than the deep set of access controls provided by the Windows platform.

```
djou:~ djou$ dsconfigad -mountstyle afp
djou's Password:
Settings changed successfully
djou:~ djou$ dscl /Active\ Directory/ads.4am-media.com -read
/Users/winnie HomeDirectory
HomeDirectory: <home_dir><url>afp:// w2k.ads.4am-
media.com/homes</url><path>winnie/</path></home_dir>
```

Changing the –mountstyle to afp indicates that the Active Directory homeDirectory attribute should be interpreted as an AFP (rather than SMB) url.

Most other, graphically available, options can also be set with dsconfigad; these options are well documented on dsconfigad's man page.

The Active Directory Plug-in: Architecture

Important to the deployment of any application is a good architectural knowledge of the files, executables, data stores and logs that support its functionality.

- /Library/Preferences/DirectoryService/ActiveDirectory.plist: The configuration file for the Active Directory Plug-in. Options (set both graphically and using dsconfigad) are stored here, in addition to mappings between Active Directory and Open Directory record types and attributes.

- /Library/Preferences/DirectoryService/SearchNodeConfig.plist: The file that stores the Open Directory search policy, specifying which directory domains should be searched for user accounts and other directory data.

- /Library/Preferences/DirectoryService/ADGroupCache.plist: As of 10.3.4, exists only in Mac OS X Server. Active Directory stores group data differently from Mac OS X and most other Unix operating systems. The transformation of Active Directory groups into something that Mac OS X understands is relatively heavy weight. Because of this and because Mac OS X frequently likes to look up group membership Apple initially cached a local copy of every group in the Active Directory. This solution did not scale, taking up to three days and sometimes producing a cache file that was a hundred megabytes or more. In 10.3.4 Apple abandoned this strategy in Mac OS X, reasoning that dynamic lookups of group membership data probably could be achieved efficiently enough to meet user performance expectations, meaning that (in the client OS) the ADGroupCache is no longer used. The legacy behavior is preserved in Mac OS X Server since it needs access to a full listing of group membership no matter who is logged in. The frequency of the Plug-in's interrogation, though, is configurable by editing the Group Search Interval Hours key in ActiveDirectory.plist (it has a default value of 12 hours).

Data transformation

The Active Directory Plug-in is interesting in that it doesn't just query Active Directory for data. That wouldn't be very useful, since Active Directory doe not contain all the data that Mac OS X needs for valid user or group records. In addition to querying Active Directory, the Plug-in performs a number of data transformations, sometimes even appending data to the user records it finds. The best example of this is probably Managed Client data (MCXSettings). When the Plug-in's localhome flag is set to enable, a great deal of Managed client data is added to each user record, specifically place the user's Active Directory Home Directory into their dock (and to have it mounted at log-in) Other examples include:

- Authentication Authority: A user's AuthenticationAuthority is the attribute that Apple uses determine how the user should be authenticated. Users without AuthenticationAuthorities can not support login-time password policies (such as expiry and forced changes) or password changes in the Accounts pane. The AD Plug-in generates an AuthenticationAuthority for every user based on the domain's configuration, allowing for the seamless support of Active Directory password policies.

```
djou:~ djou$ dscl /Active\ Directory/ads.4am-media.com -read
/Users/winnie
AuthenticationAuthorityAuthenticationAuthority:
1.0;Kerberosv5;83981D08-027D-3843-BE3B-
AB80FA3DA07F;winnie@ADS.4AM-MEDIA.COM; ADS.4AM-MEDIA;
comment
```

- HomeDirectory and NFSHomeDirectory: HomeDirectory and NFSHomeDirectory describe the location of a user's network home directory. The former is an XML plist describing how to create the latter, which is a file system path. Neither is a standard part of an Active Directory user record (although the latter can be supplied by Active directory's msSFU30HomeDirectory if services for Unix are installed). As seen earlier, both are automatically generated based on the account's home directory as described in their Active Directory user record (using the UNC path described earlier in this chapter).

- Mount Record: The mount record works in conjunction with the User's HomeDirectory and NFSHomeDirectory attributes to help support network home directories. Like the user home directory attributes it is generated on the fly based on the User's home directory UNC.

```
djou:~ djou$ dscl /Active\ Directory/ads.4am-media.com -
read /Mounts/w2k:\\/homes
ADDomain: ads.4am-media.com
AppleMetaNodeLocation: /Active Directory/ ads.4am-
media.com Comment: Dynamically generated - DO NOT
ATTEMPT TO MODIFY
RecordName: w2k:/homes
VFSLinkDir: /Network/Servers/w2k/homes
VFSOpts: net url==smb://w2k.ads.4am-media.com/homes
VFSType: url
```

The AD Plug-in generates an automount record designed to help mount network home directories. Guest access doe not have to be enabled on this share point. For now, Mac OS X is incapable of using virtual home shares (\\server\username) as user network home directories, and must be able to locate home directories at a path below a share point.

• Kerberos Auto Configuration record: One of Mac OS X's more intriguing Kerberos integration features is auto-configuration. Mac OS X, when it determines that it needs to be configured for Kerberos, will execute the kerberosautoconfig command. Kerberosautoconfig, in turn, will search the directory domains that the client is aware of, looking for a Kerberos configuration record. This record is very specific to Apple's infrastructure, and not typically found in non-Mac directories. The Active Directory Plug-in, however, is smart enough to auto-generate this configuration record, allowing for easy Kerberos interoperability.

Caveats

Panther's Active Directory Plug-in is by no means perfect, and although Apple has done a relatively good job I'd be remiss if I did not mention some of the pitfalls I've encountered. The most common issues tend to be unrelated to the Plug-in itself, and are more related to other capabilities in the OS. –localhome enabled's use of NTLMv1 authentication (which is disabled in security-sensitive environments), for instance, is due to the fact that the OS does not obtain a TGT early enough during log-in. There's not much that the AD Plug-in can do about that. Similarly, Mac OS X may not access user home directories on either DFS or a clustered CIFS file system. Incidentally, Thursby's ADmitMac product, which is a commercial Open Directory plug-in for both NT domains and Active Directory, is not subject to these particular limitations, since it uses Thursb'y Dave CIFS / SMB client. AdmitMac also overcomes a less common issue, where Computer accounts are not allowed to read certain user attributes. This measure is sometimes implemented to protect user privacy, but since Panther's AD Plug-in connects to the domain as the computer (rather than as the user) this could have the effect of keeping users from being recognized. AdmitMac pulls some tricks to actually connect to the domain as the user (rather than the computer) ensuring full access to at least some user data. Finally, note that AdmitMac supports Packet Signing, a cryptographic security feature turned on by default in Windows 2003 server. The AD Plug-in does not. Neither AdmitMac northe AD Plug-in support nested groups, a common management strategy in Active Directory.

Another common issue that is encountered at the basic integration level is the use of DNS. Mac OS X, like Windows clients, uses DNS to locate domain resources during the join process. This means that Mac OS X clients must have the Active Directory DNS server listed in the Network pane of the System Preferences application. Another DNS-related issue revolves around the common use of the .local TLD. This conflicts with Apple's Rendezvous multicast DNS implementation and must be worked around. Apple documents one procedure for this in kbase 107800. There are several other, less intrusive solutions but they are beyond the scope of this article.

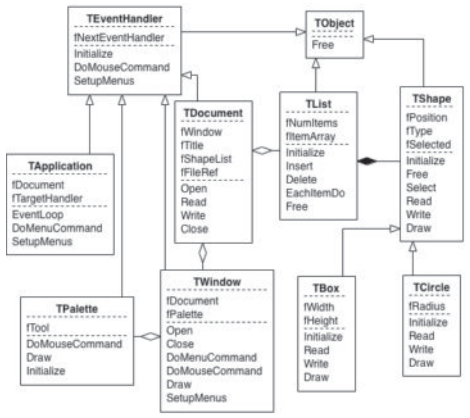
Conclusion

Someone other than me said that "A willow tree bends in the wind and so the branches, being supple do not break." It takes little imagination to understand that Microsoft is a force of nature right now and that competing all out against them is ill advised. What matters to Apple's survival is sales, and Panther's Active Directory Plug-in, in making Mac OS X more willow-like, makes sales a lot easier. Good solutions support— rather than fight— existing IT infrastructures.

MT

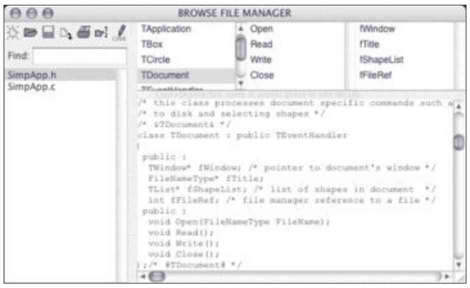
About The Author

Michael Bartosh is a consultant specializing in large scale server deployments, directory services integration and scalable systems management.



UML

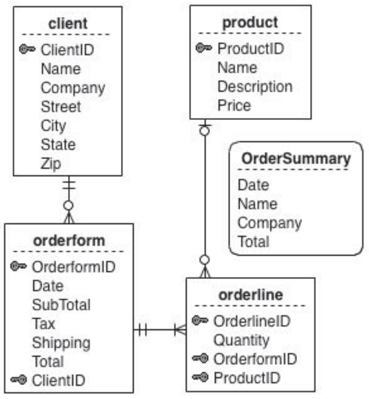
Code Generation



```
/* this class processes document specific commands such a
/* to disk and selecting shapes */
/* &TDocument& */
class TDocument : public TEventHandler
{
 public :
  TWindow* fWindow; /* pointer to document's window */
  FileNameType* fTitle;
  TList* fShapeList; /* list of shapes in document  */
  int fFileRef; /* file manager reference to a file */
 public :
  void Open(FileNameType FileName);
  void Read();
  void Write();
  void Close();
};/* #TDocument# */
```

Reengineering

Data Modeling

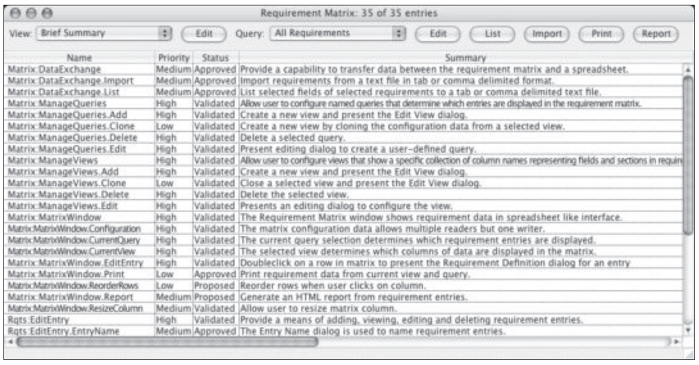


Flexible Reporting

MacA&D OSX

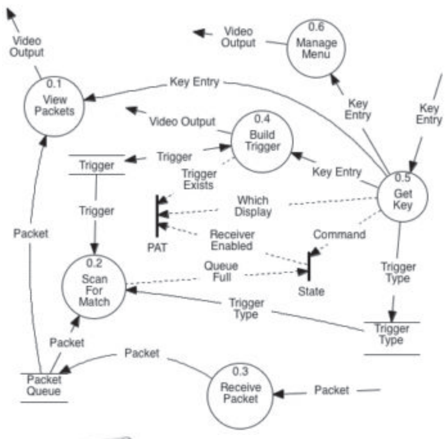
Software Development Solutions

Requirements Management



Name	Priority	Status	Summary
Matrix:DataExchange	Medium	Approved	Provide a capability to transfer data between the requirement matrix and a spreadsheet.
Matrix:DataExchange.Import	Medium	Approved	Import requirements from a text file in tab or comma delimited format.
Matrix:DataExchange.List	Medium	Approved	List selected fields of selected requirements to a tab or comma delimited text file.
Matrix:ManageQueries	High	Validated	Allow user to configure named queries that determine which entries are displayed in the requirement matrix.
Matrix:ManageQueries.Add	High	Validated	Create a new view and present the Edit View dialog.
Matrix:ManageQueries.Clone	Low	Validated	Create a new view by cloning the configuration data from a selected view.
Matrix:ManageQueries.Delete	High	Validated	Delete a selected query.
Matrix:ManageQueries.Edit	High	Validated	Present editing dialog to create a user-defined query.
Matrix:ManageViews	High	Validated	Allow user to configure views that show a specific collection of column names representing fields and sections in requir
Matrix:ManageViews.Add	High	Validated	Create a new view and present the Edit View dialog.
Matrix:ManageViews.Clone	Low	Validated	Close a selected view and present the Edit View dialog.
Matrix:ManageViews.Delete	High	Validated	Delete the selected view.
Matrix:ManageViews.Edit	High	Validated	Presents an editing dialog to configure the view.
Matrix:MatrixWindow	High	Validated	The Requirement Matrix window shows requirement data in spreadsheet like interface.
Matrix:MatrixWindow.Configuration	High	Validated	The matrix configuration data allows multiple readers but one writer.
Matrix:MatrixWindow.CurrentQuery	High	Validated	The current query selection determines which requirement entries are displayed.
Matrix:MatrixWindow.CurrentView	High	Validated	The selected view determines which columns of data are displayed in the matrix.
Matrix:MatrixWindow.EditEntry	High	Validated	Doubleclick on a row in matrix to present the Requirement Definition dialog for an entry
Matrix:MatrixWindow.Print	Low	Approved	Print requirement data from current view and query.
Matrix:MatrixWindow.ReorderRows	Low	Proposed	Reorder rows when user clicks on column.
Matrix:MatrixWindow.Report	Medium	Proposed	Generate an HTML report from requirement entries.
Matrix:MatrixWindow.ResizeColumn	Medium	Validated	Allow user to resize matrix column.
Rqts:EditEntry	High	Validated	Provide a means of adding, viewing, editing and deleting requirement entries.
Rqts:EditEntry.EntryName	Medium	Approved	The Entry Name dialog is used to name requirement entries.

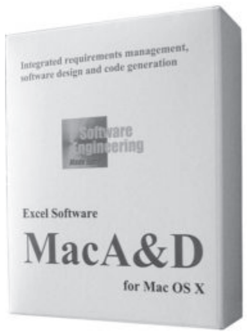
Structured Analysis & Design



When critical software is needed for the next space mission, automotive control, defense system, telecommunications or desktop application, we can help. MacA&D and WinA&D have been there, done that with 18 years of field proven innovations.

Excel Software provides thousands of developers with tools to design and manage object-oriented software, database systems and embedded, real-time firmware.

www.excelsoftware.com


Integrated requirements management, software design and code generation
Software Engineering
Excel Software
MacA&D
for Mac OS X

ES

Excel Software
19 Misty Mesa Ct
Placitas, NM 87043

Ph. 505-771-3719
Fax 505-771-3718

SCREEN SAVERS IN COCOA

One of the classic mantra-like goals for computer science over the past 20 years or so has been to "Make simple things simple, and complex things possible". Programming with Cocoa has a sometimes-complex learning curve, but once you've swerved through that curve, there are definitely a bunch of things that are much easier to accomplish than they were in the old pre-OS X world we knew and occasionally loved. Writing a screen saver is a perfect example: it should be simple. Most typical OS 9 application programmers never dipped their toes into the slightly wacky world of screen savers, but with Cocoa in OS X, implementing a screen saver is well within everybody's grasp. In this month's column, we'll take a look at how to get your very own screen saver up and, er, saving.

Here's What We're Gonna Do

Let's start by taking a look at the process for creating a screen saver in OS X. Here are the broad steps:

1. Create a new screen saver project in Xcode.
2. Edit our .h file.
3. Override methods and write other code in our .m file.
4. Build the project to create a .saver package.
5. Install the .saver package by putting it into the Library/Screen Savers/ folder.
6. Open System Preferences and see a preview of our screen saver.
7. Enjoy the savings!

We'll go through each of these steps in greater depth now.

Little Help

The basic magic that makes screen savers so easy is the Screen Saver framework in Cocoa. This framework defines the ScreenSaverView class, which is a subclass of NSView. By creating your own subclass of ScreenSaverView and adding some code, you define your screen saver. The Screen Saver framework also defines the class ScreenSaverDefaults, which you can use for handling preferences for your saver. Along with these classes, the framework provides some handy utility functions you can use in your code.

We'll go over some of the most interesting methods and functions in the ScreenSaverView class. You will rarely call methods defined by ScreenSaverView – most of the work is creating your own subclass and override some methods.

```
initWithFrame:isPreview:
- (id)initWithFrame:(NSRect)frame isPreview:(BOOL)isPreview
```

You override initWithFrame in your ScreenSaverView subclass. The system calls initWithFrame when the screen saver is about take over the screen or is selected in System Preferences. The frame parameter is the frame rectangle for the view. The isPreview parameter tells whether the screen saver is actually being invoked or is merely being asked to preview itself in System Preferences (as shown in Figure 1).

One of the classic mantra-like goals for computer science over the past 20 years or so has been to "Make simple things simple, and complex things possible". Programming with Cocoa has a sometimes-complex learning curve, but once you've swerved through that curve, there are definitely a bunch of things that are much easier to accomplish than they were in the old pre-OS X world we knew and occasionally loved.

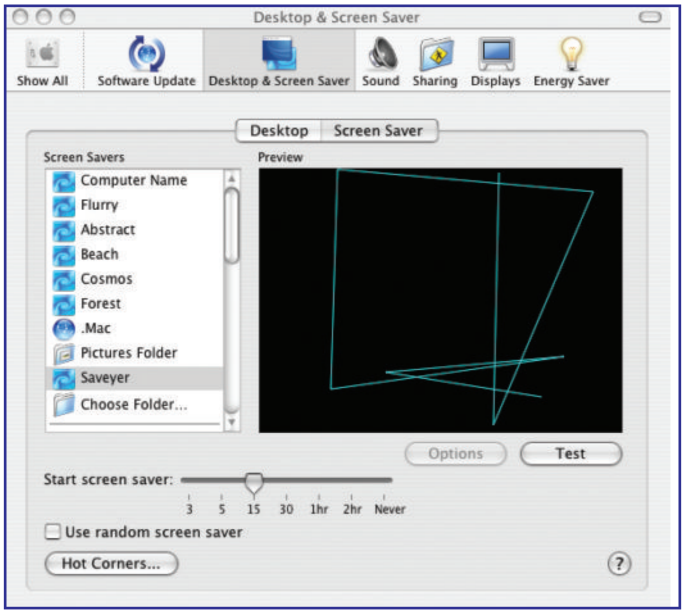


Figure 1. You can preview the screen saver in a little box in System Preferences. In your code, you can tell whether the screen saver is drawing full-screen or in the preview box.

```
startAnimation
```

```
- (void) startAnimation
```

The system calls startAnimation right before the screen saver is about to start drawing. You should override startAnimation to set up your screen saver's initial state, such as setting line widths or loading images. You should call the inherited implementation, or bad things might happen, such as incorrect drawing, or all water on earth instantaneously evaporating.

```
animateOneFrame
```

```
- (void) animateOneFrame
```

This is where your screen saver gets to show off its amazing graphical skills. Mac OS X asks your screen saver to do its drawing by calling animateOneFrame repeatedly. You can actually do your drawing in animateOneFrame or in drawRect, or even a little in both places. If you make any changes here that require further redrawing, your implementation should call setNeedsDisplay:YES, which will cause drawRect to be called.

```
drawRect
```

```
- (void) drawRect:(NSRect)rect
```

Override drawRect to draw the screen saver view. You can do your drawing in animateOneFrame, or you can do some or all of it here. rect is the rectangle you're drawing into, which is handy to have when you want to erase the view and start drawing afresh.

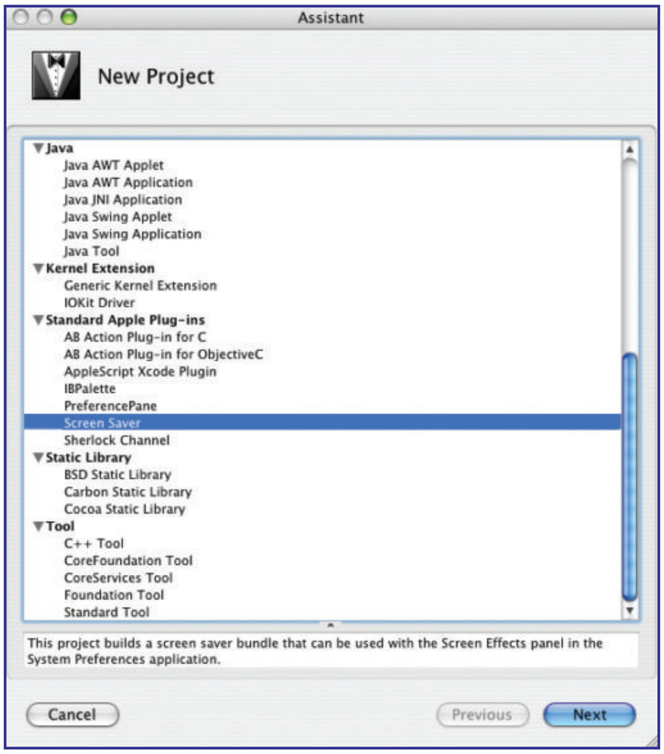
```
stopAnimation
```

```
- (void) stopAnimation
```

When Mac OS X wants your screen saver to stop doing its thing, it calls stopAnimation. You can override stopAnimation to release resources or do any other cleanup you want before your screen saver goes away.

Saving Time

Now that you're familiar with the cast of characters in ScreenSaverView, let's go ahead and code up our screen saver. To start, we'll open Xcode and create a new project of type Screen Saver (see Figure 2).



This project builds a screen saver bundle that can be used with the Screen Effects panel in the System Preferences application.

Figure 2. Creating a new Screen Saver project gets you started with the Screen Saver framework, including your own subclass of ScreenSaverView.

This proves that Xcode already knows about the screen saver framework, which saves us plenty of work. Our new project already contains a subclass of ScreenSaverView, and we already have the usual .h and .m files. We'll edit the .h file that Xcode gives us until it looks like this:

```
#import <ScreenSaver/ScreenSaver.h>


@interface SaveyerView : ScreenSaverView
{
  NSBezierPath *path;
}


@end
```

The header file is pretty darn basic. All we do here is create a subclass of ScreenSaverView and add an NSBezierPath object to keep track of what we're drawing.

Now let's get into the implementation files and see what we can find. When we told Xcode to create a new ScreenSaver project, it start us off with some code, including the implementation for initWithFrame:isPreview:, the designated initializer. In this case, we're able to use the supplied code for initWithFrame without any changes:

```
- (id)initWithFrame:(NSRect)frame
isPreview:(BOOL)isPreview
{
    self = [super initWithFrame:frame
isPreview:isPreview];
    if (self) {
        [self setAnimationTimeInterval:1/30.0];
        // Draw 30 frames per second
    }
    return self;
```

The code here starts by calling the inherited implementation. After that, we use setAnimationTimeInterval to tell Mac OS X that we want our screen saver to draw 30 frames per second. As I mentioned, this is the default code that Xcode writes for this method. You can modify it if you want to perform some other task when the screen saver starts up. For example, if your screen saver has user-settable options, you can handle them here.

Next, we'll take a look at our startAnimation method, which the system calls right before asking our screen saver to start drawing. Our implementation of startAnimation begins by calling the inherited implementation. Then, we create our Bezier path and choose a nifty line join style:

```
- (void)startAnimation
{
  NSPoint x;

  [super startAnimation];

  path = [[NSBezierPath alloc] init];
    // We'll use a Bezier path for drawing

  [path setLineJoinStyle: NSRoundLineJoinStyle];
    // Just for fun, connect the lines with
    // a round joint
```

When the system asks our screen saver to get ready to draw, we can call the view's isPreview method to see if we're being asked to draw on the full screen or in the little preview box in System Preferences (as shown back in Figure 1).

We can use the result of isPreview to make decisions about just what to draw. In our screen saver, we'll make the lines skinny for the preview, and fatter for the real, full-screen version:

```
  if ([self isPreview])
    // When drawing a preview, make the lines
    // much thinner than when saving screens.
  {
    [path setLineWidth: 0.0];
    // This is the thinnest possible line width
  }
  else
  {
    [path setLineWidth: 10.18];
    // This line width was chosen at random.
    // OK, actually, it's my son's birthdate.
  }
```

Our last task here is to get the Bezier path started. We'll do that by picking a random starting point and moving the path there:

```
  x = SSRandomPointForSizeWithinRect
      (NSMakeSize (0,0), [self bounds]);
    // Call utility function to get a random point

  [path moveToPoint:x];
    // Start the path at the random point
}
```

We get a random point by calling SSRandomPointForSizeWithinRect, a handy function provided by the screen saver framework for just this purpose. Hooray for handy functions! Then, we simply move the path pen to that random point to start it out.

Everything that starts must end, and the next method we define is stopAnimation, which is called when the system doesn't need the screen saver to draw any more. Here's our implementation of stopAnimation:

Shine.

Big nerd ranch

Apache Bootcamp

Cocoa® Bootcamp

Core Bootcamp

PHP 5 Bootcamp

PostgreSQL Bootcamp

Python® Bootcamp

Core Mac OS® X Bootcamp

Get your dark sunglasses ready. Tell your boss to roll out the red carpet. Are you ready for your breakthrough role? Director Mark Dalrymple, author of *Core Mac OS® X and Unix Programming*, will take you from indies to a blockbuster in five days.

Sample Scenes:	Multithreading	Distributed Objects
	Rendezvous™	Network Programming
	System Daemons	Authentication & Authorization

Big Nerd Ranch offers intensive training classes for developers and administrators. Your expert instructor guides you through a rigorous week of learning. You leave ready to start developing (but with instructor support if you get stuck).

Big Nerd Ranch: Baby, we'll make you a star!

www.bignerdranch.com • 678.595.6773 • roundup@bignerdranch.com

```
- (void)stopAnimation
{
  [super stopAnimation];

  [path release];
      // Release the path

  path = nil;
      // Tell our screen saver view that there's no path
}
```

The standard stopAnimation provided by Xcode simply calls the inherited implementation. In our version, we keep that super call, and add code to release the Bezier path object and set the path instance variable to nil.

Every time the system wants our screen saver to draw another piece, it calls our animateOneFrame method. Let's take a look at that. First, we'll call that convenient SSRandomPointForSizeWithinRect utility function to get another random point:

```
- (void)animateOneFrame
{
    NSPoint x;

  x = SSRandomPointForSizeWithinRect
      (NSMakeSize (0,0), [self bounds]);
        // Get a random point to extend the Bezier path
```

We want our screen saver to draw a bunch of lines on the screen, and every so often, we want it to erase the lines and start over. Let's say we want 50 lines at a time, in honor of our 50 states. If we haven't reached 50 yet, we add the new random point to the path:

```
  if ([path elementCount] < 50)
    // Draw 50 lines before erasing
    {
      [path lineToPoint: x];
        // If we don't have 50 yet, add the
        // new point to the line
    }
```

Once we have 50 points in the path, we want to reset the path by callously discarding all points and then start building it up again:

```
    else
    {
      [path removeAllPoints];
      [path moveToPoint:x];
        // If we do have 50, clean out the path
        // and get ready to start over
    }
```

We finish by telling the system that we've messed with the path and it needs to be redrawn by calling the screen saver view's drawRect method. Alternatively, we could do the actual drawing right here in animateOneFrame:

```
  [self setNeedsDisplay:YES];
    // Tell the system that something has changed
    // and drawRect should be called
}
```

The actual drawing happens in drawRect, which we'll look at NeXT. We start by calling the inherited drawRect, which by default erases the background to black.

```
- (void)drawRect:(NSRect)rect
{
  NSColor *color;

  [super drawRect:rect];
```

We then choose a pretty color, and call set to make sure that the drawing happens in that color. Then we call stroke on the Bezier path object to actually draw the thing:

```
  color = [NSColor colorWithCalibratedRed:(0.0)
            green:(1.0) blue:(1.0) alpha:(1.0)];

  [color set];
    // Set the color to teal. Go Sharks!

  [path stroke];
    // Draw the Bezier path
}
```

The last method we implement is our version of dealloc. The view's Bezier path is the only allocated object we have to worry about, so our method looks like this:

```
- (void) dealloc
{
  [path release];
    // Release the Bezier path

  [super dealloc];
}
```

Put Me In, Coach

When we have all the source code done, we build our project. If everything builds OK, a file with the suffix .saver ends up in the project's build folder. To install the screen saver, start by quitting System Preferences if it's running. Then move or copy the .saver file into the /Library/Screen Savers directory. You can put it in ~/Library/Screen Savers if you want to keep it all to yourself and prevent other users from seeing it.

Once our screen saver is in the folder, you can start System Preferences, click Desktop & Screen Saver, click the Screen Saver tab, and select our screen saver in the list. You should see the skinny lines in the preview mode. Then click Test, and observe the big teal lines with their round elbows. There you go! You can get this month's code at http://www.papercar.com/mt/Jun04.zip

If you're interested in making your own screen savers, there are lots of directions you can go from here. Add user-settable options by overriding the hasConfigureSheet and configureSheet methods. Use random colors. Do some much fancier drawing in your animateOneFrame method – for example, draw shapes, use curveToPoint instead of lineToPoint, or load images from disk. Whatever you do, have fun, and remember: the screen you save may be your own.

MT

About The Author

Scott Knaster writes books, including the recently published Mac Toys and the brand-new Hacking iPod and iTunes, both from Wiley Publishing. Scott can't read and listen to vocal music at the same time. Scott writes these little bios in the third person. Write to Scott at scottk@mactech.com.

By Paul Day

Securing Mac OS X

A guide to security hardening for Mac OS 10.3

INTRODUCTION

This article covers numerous methods to harden Apple's Mac OS X, from both a local user and network perspective. It is primarily aimed at the single-user Macintosh client machine owned and used by a security conscious user. Its methods can be equally applied to a multi-user machine; however there are numerous additional security risks presented the moment a Mac OS X machine is made multi-user.

BACKGROUND

Apple's MacOS has taken a dramatic change from its predecessors ("MacOS Classic"), introducing numerous parts of FreeBSD, NeXT and the Mach (Darwin) kernel into the MacOS environment.

"Keep others out - With Mac OS X, you may never need to worry about HYPERLINK "http://www.apple.com.au/macosx/features/security/" security again."

A default install of Mac OS X is one of the more secure Unix operating systems from a network-security point of view, with no network services open by default. However, there are still numerous drawbacks to its local and network security which can be addressed by the administrator of the machine.

ROOT USAGE

By default, the root user account within Mac OS X has its password disabled. Throughout this paper, you are required to run a command "as root". The method of doing this is left up to the reader, but possibilities (in order of considered strength) include:

- sudo <command> as a normal admin user.
- sudo /bin/bash as a normal admin user and then running the commands.
- Enabling the root account password, using su to start a shell as root and then running the commands.

LOCAL SECURITY

The following section covers numerous methods to harden security within Mac OS X from a local user perspective:

- With local physical access to the machine via its console. With interactive local access to the machine via methods such as Secure Shell (SSH) or Apple Remote Desktop (ARD).

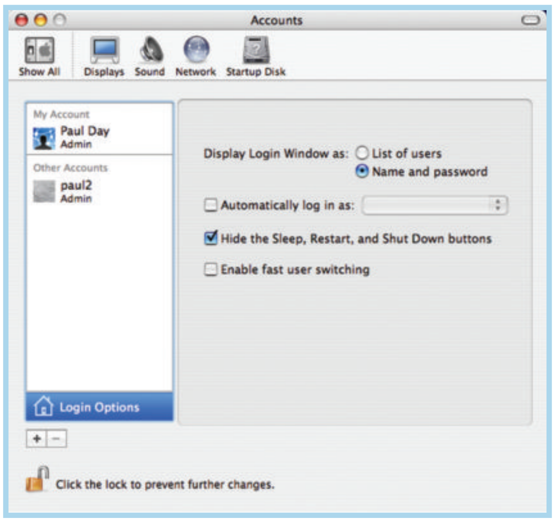
THE LOGIN WINDOW

The following includes instructions to enable and lock down the GUI login window. By default, Mac OS X automatically logs in rather than forcing the user to authenticate at a login window.

Enabling and locking down the Login Window

To enable the GUI login window, disable password hints, access to shutdown/restart controls and automatic login you can edit the file /Library/Preferences/com.apple.loginwindow.plist as root or use the System Preferences Accounts pane as follows:

- Apple menu -> System Preferences -> Accounts -> Login options -> Display Login Windows as -> Name and Password
- Uncheck Automatically log in as:
- Check Hide the Sleep, Restart and Shut Down buttons
- Uncheck Enable fast users switching if not used



Display Login Window as: ○ List of users [✓ ● Name and password]

☐ Automatically log in as:

☑ Hide the Sleep, Restart, and Shut Down buttons

☐ Enable fast user switching

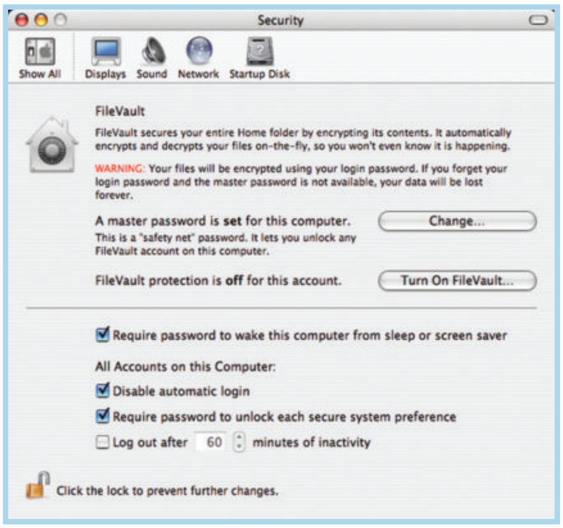
Click the lock to prevent further changes.

Securing Login Windows options

Fast user switching is handy on a multi-user machine, however on a single-user machine where it is never used, it is an unnecessary risk (eg, An Apple Remote Desktop root compromise used Fast User Switching).

To disable automatic login on a global basis:

- Apple menu -> System Preferences -> Security
- Check Disable automatic login



FileVault

FileVault secures your entire Home folder by encrypting its contents. It automatically encrypts and decrypts your files on-the-fly, so you won't even know it is happening.

WARNING: Your files will be encrypted using your login password. If you forget your login password and the master password is not available, your data will be lost forever.

A master password is set for this computer.
This is a "safety net" password. It lets you unlock any FileVault account on this computer.

FileVault protection is off for this account.

☑ Require password to wake this computer from sleep or screen saver

All Accounts on this Computer:

☑ Disable automatic login

☑ Require password to unlock each secure system preference

☐ Log out after 60 minutes of inactivity

Click the lock to prevent further changes.

Disabling automatic login

To enable a text message to be displayed as part of the login window, you will need to edit the file /Library/Preferences/com.apple.loginwindow.plist as root. The file may look like:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE plist PUBLIC "-//Apple Computer//DTD PLIST
1.0//EN" "http://www.apple.com/DTDs/PropertyList-1.0.dtd">
<plist version="1.0">
<dict>
   <key>DisableConsoleAccess</key>
   <true/>
   <key>LoginwindowText</key>
   <string>Authorized users only.</string>
```

Note the <string> line below the key LoginwindowText. Insert the text you would like to appear in the Login Window here and finish it with the </string>.

Changing passwords

It is good security practice to regularly change your password, especially as the login window does not presently make of mlock() or encrypted swap and a user with physical/root access to the machine could potentially get your login password from the swap files.

- Apple Menu -> System Preferences -> Accounts
- Select your username -> Select the Password field
- If asked, type in your current password -> Type in a new password -> verify the new password

SCREENSAVER

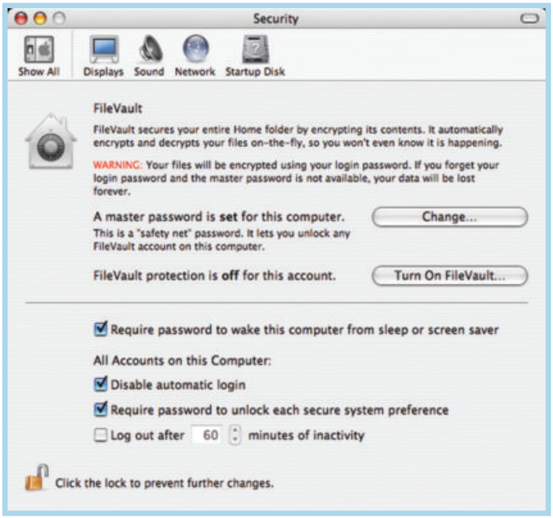
Mac OS X comes with a built-in screen-saver that includes password locking. This should be enabled to stop someone from using your computer when you step away from it.

To enable the screen-saver:

- Apple menu -> System Preferences -> Desktop & Screensaver -> Screen Saver -> (Select a screen-saver)
- Change Start screen saver to 3 minutes

To require a password to exit the screen saver:

- Apple -> System Preferences -> Security
- Check Require password to wake this computer from sleep or screen saver



Security

FileVault

FileVault secures your entire Home folder by encrypting its contents. It automatically encrypts and decrypts your files on-the-fly, so you won't even know it is happening.

WARNING: Your files will be encrypted using your login password. If you forget your login password and the master password is not available, your data will be lost forever.

A master password is **set** for this computer. This is a "safety net" password. It lets you unlock any FileVault account on this computer.

FileVault protection is **off** for this account.

- [✓] Require password to wake this computer from sleep or screen saver

All Accounts on this Computer:

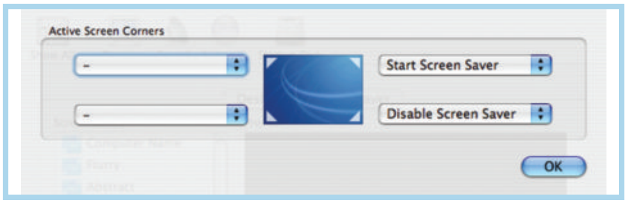
- [✓] Disable automatic login
- [✓] Require password to unlock each secure system preference
- [ ] Log out after 60 minutes of inactivity

Click the lock to prevent further changes.

Enabling password locking within screen saver

You may also wish to enable an active-corner to disable the screensaver for times you don't want it to come on after inactivity (e.g. while watching a movie) and, more importantly, to instantly load the screensaver:

- Apple menu -> System Preferences -> Desktop & Screensaver -> Screen Saver -> Hot Corners
- Choose a corner, e.g. bottom right -> Disable Screen Saver
- Choose a corner, e.g. top right -> Start Screen Saver



Active Screen Corners

Top left: -
Top right: Start Screen Saver
Bottom left: -
Bottom right: Disable Screen Saver

Enabling a screen saver corner

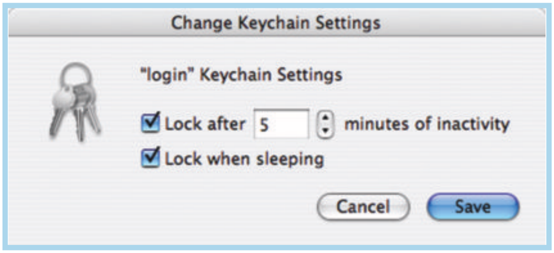
KEYCHAIN

Mac OS X includes a utility for caching commonly used passwords. It should be noted that there is always a risk with caching a password on disk in any form, regardless of the software used.

Keychain stores its passwords on disk in an encrypted form and it is difficult for a non-root user to sniff a password between applications. However, similar to the Login Window, it is possible to get hold of a user's Keychain password with root or physical access to a machine. The best practice is to remember your passwords without storing them.

There are a number of steps you can take to minimise your risk when using Keychain Access. To enable Keychain automatic locking:

- Applications -> Utilities -> Keychain Access -> Edit -> Change settings for Keychain "login"
- Check Lock after
- Change minutes of inactivity to 5 minutes
- Check Lock when sleeping
- Save



Change Keychain Settings

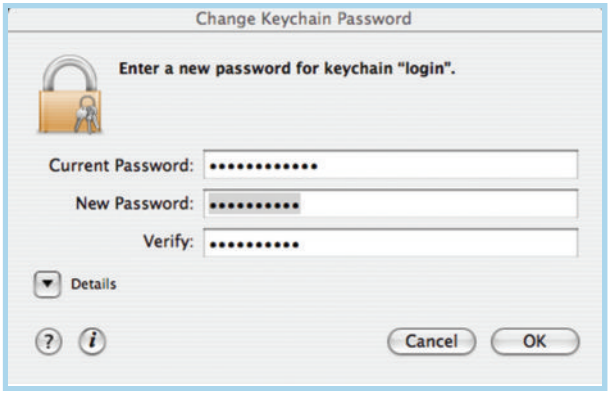
"login" Keychain Settings

- [✓] Lock after 5 minutes of inactivity
- [✓] Lock when sleeping

Configure Keychain Access security settings

By default, Mac OS X makes your Keychain password the same as your login password. It is good practice to keep each password different:

- Edit -> Change Password for Keychain "login"
- Type in your current user's login password
- Type in a new different password twice
- OK



Change Keychain Password

Enter a new password for keychain "login".

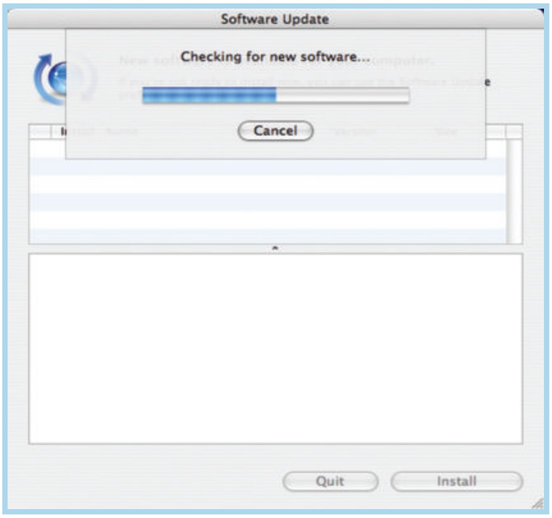
Current Password: ••••••••••••
New Password: ••••••••••
Verify: ••••••••••

Changing your Keychain password

As is generally the case, you should keep your Mac OS X machine regularly patched with the latest software updates, which often include security fixes.

Apple Software Update

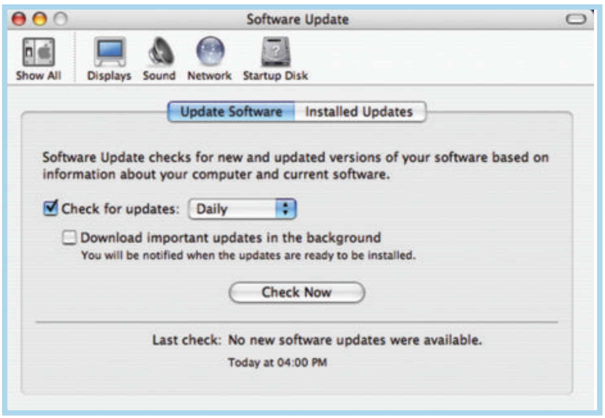
Mac OS X includes an automatic software update tool to patch the majority of Apple applications. Software Update often includes important security updates which should be applied to your machine. The tool automatically checks what updates are available and, with major upgrades, can download patches rather than full installations, to minimize the amount downloaded.



Software Update

It is best to configure Software Update to automatically check for updates on a frequent basis:

- Apple Menu -> System Preferences -> Software Updates
- Check Check for updates
- Choose Daily from drop-down menu.



Software Update checks for new and updated versions of your software based on information about your computer and current software.

Check for updates: Daily

Download important updates in the background
You will be notified when the updates are ready to be installed.

Last check: No new software updates were available.
Today at 04:00 PM

Software Update, automatically check for updates

Your machine will now check with Apple for software updates once a day and notify you when there are new ones ready for download.

Software update can also be run from the command line as root with:

```
/usr/sbin/softwareupdate -ia
and scheduled to run with:
/usr/sbin/softwareupdate -schedule on
```

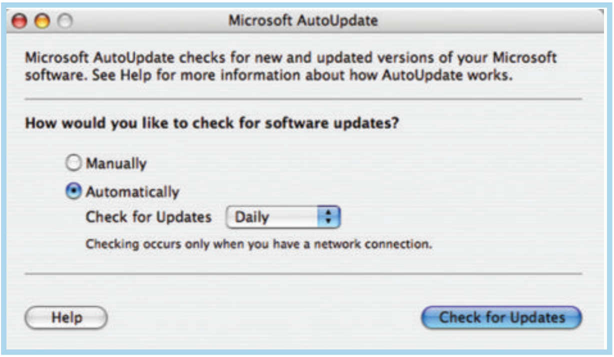
Software update for Fink

If you are using the Fink packaging systems, you may also wish to have the following in root's daily crontab or in /etc/daily:

```
/sw/bin/fink -y selfupdate
/sw/bin/fink -y selfupdate-cvs
/sw/bin/fink -y update-all
/sw/bin/fink -y scanpackages
/sw/bin/fink -y index
/sw/bin/fink -y cleanup
/sw/bin/apt-get -y update
/sw/bin/apt-get -y install fink
/sw/bin/apt-get -y upgrade
/sw/bin/apt-get -y dist-upgrade
/sw/bin/apt-get -y clean
/sw/bin/apt-get -y autoclean
/sw/bin/apt-get -y check
```

Other updates

Many other major software packages include their own automatic software update utilities. These may be separate utilities such as Microsoft's AutoUpdate:



Microsoft AutoUpdate checks for new and updated versions of your Microsoft software. See Help for more information about how AutoUpdate works.

How would you like to check for software updates?

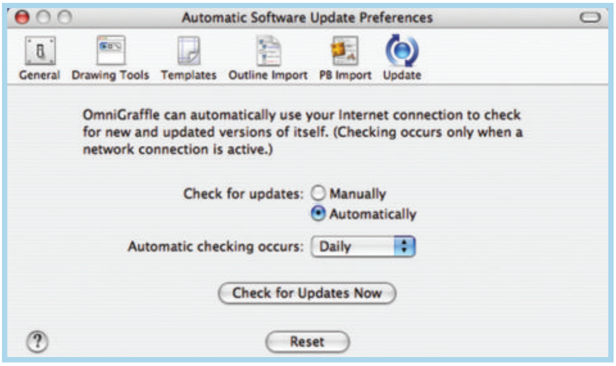
Manually [✓ Automatically]

Check for Updates: Daily

Checking occurs only when you have a network connection.

Microsoft AutoUpdate

Other packages, such as Omni's OmniGraffle, include automatic updating from within the software package itself:



OmniGraffle can automatically use your Internet connection to check for new and updated versions of itself. (Checking occurs only when a network connection is active.)

Check for updates: Manually [✓ Automatically]

Automatic checking occurs: Daily

OmniGraffle automatic software update

You are encouraged to use these tools whereever possible, however specifics are beyond the scope of this paper.

FILE ENCRYPTION

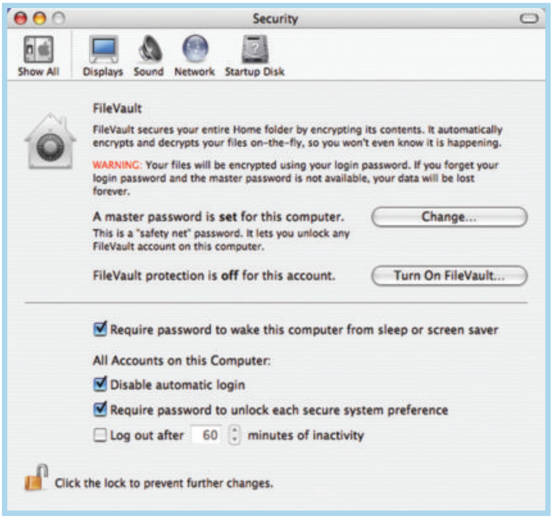
There are number of major ways of encrypting files within Mac OS X. By far, the most secure method is to use GnuPG; however Apple's FileVault and disk images are much more convenient.

FileVault and encrypted volumes

Apple's FileVault is an implementation of its AES-encrypted volume images that automatically mount as your home directory as you login and decrypt/encrypt data on the fly. Encrypting data on your hard-drive is nothing new but MacOS 10.3 is the first Unix to integrate decryption and mounting seamlessly into the system. From the point-of-view of the user and applications, there is no encryption taking place, beyond a slight performance hit.

To enable FileVault:

- Apple menu -> System Preferences -> Security
- Turn on FileVault



FileVault

FileVault secures your entire Home folder by encrypting its contents. It automatically encrypts and decrypts your files on-the-fly, so you won't even know it is happening.

WARNING: Your files will be encrypted using your login password. If you forget your login password and the master password is not available, your data will be lost forever.

A master password is set for this computer.
This is a "safety net" password. It lets you unlock any FileVault account on this computer.

FileVault protection is off for this account.

☑ Require password to wake this computer from sleep or screen saver

All Accounts on this Computer:

☑ Disable automatic login

☑ Require password to unlock each secure system preference

☐ Log out after 60 minutes of inactivity

Click the lock to prevent further changes.

Enabling FileVault

Depending on the amount of data in your home directory, it may take a while to convert it into a FileVault. It should be noted here that after encrypting your home directory, it is not securely deleted. It is simply unlinked and hence could be recovered.

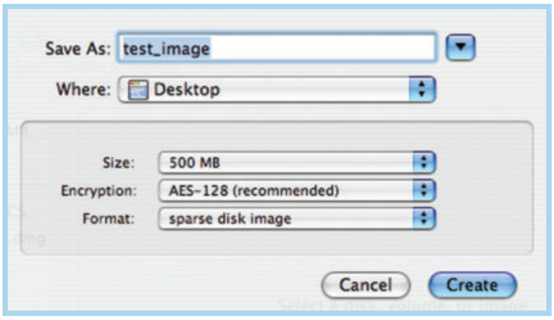
You may also wish to set a master password for the computer. The master password should be different to your login (and hence FileVault) password and can be used to decrypt your FileVault in the case of password loss.

From a security point of view, keep in mind that due to a lack of mlock() in FileVault, an attacker with physical or root access can gain your FileVault password and access to your encrypted files.

Encrypted AES disk image

Apple's encrypted disk images don't offer the seamless mounting of FileVault, but do still encrypt on the fly as you write to them. To create an encrypted disk image:

- Applications -> Utilities -> Disk Utility
- New Image
- Save as -> Choose a name for the file system and image file name
- Where -> Choose a location to save the image file
- Size -> Choose a maximum size to allow the image to grow to
- Encryption -> Choose AES-128
- Format -> Sparse Disk Image
- Create -> Enter and Verify password
- Check or uncheck Remember password (add to Keychain)



Save As: test_image

Where: Desktop

Size: 500 MB

Encryption: AES-128 (recommended)

Format: sparse disk image

Creating an encrypted sparse image

It is no less secure to save a disk image's password in the Keychain as Apple's SecurityAgent (the program that takes the password from the user) suffers from the same vulnerability as Keychain itself.

Once you have created the disk image, you can mount it by double-clicking on it in Finder. It will then mount as /Volumes/<image file system name> and an icon will appear on your desktop.

Openssl encrypted files

Another alternative is using openssl and a password to encrypt a file. Openssl does not employ asymmetric keys (i.e. a private and public key) and allows you to just assign a single password to the encrypted file. However, openssl under Mac OS X may suffer a similar vulnerability to FileVault.

To encrypt a file using openssl and the (128bit) blowfish encryption algorithm:

```
openssl bf -salt -in <plain file> -out <encrypted file>
```

Then securely remove the original file:

```
srm -fm <input file>
```

Finally, decrypt the file back:

```
openssl bf -d -in <encrypted file> -out <plain file>
A script to encrypt an entire directory could be:
#!/bin/sh
#
# Script to encrypt a dir and securely remove it.

if [ $# -lt 1 ] ; then
  echo "Usage: $0 dir_to_encrypt"
  exit 1
fi

file=`echo $1 | sed s/"\/"//g | sed s/"\."//g`
dir=$1

echo -n "Checking if $dir actually exists... "
if [ -d $dir ] ; then
  echo "Yes."
else
  echo "No. Exiting."
  exit 1
fi

echo -n "Checking to make sure $file.tar.gz.bf doesn't
already exist... "
if [ -e $file.tar.gz.bf ] ; then
  # exists
  echo "Yes. Exiting."
  exit 1
else
  # doesn't exist
  echo "No."
fi

echo -n "Checking to make sure tempfile doesn't already
exist... "
if [ -e temp.tar.gz ] ; then
  echo "Yes. Exiting. You need to remove temp.tar.gz."
  exit 1
else
  echo "No."
fi
echo "Tarring up directory..."
tar -zcvf temp.tar.gz $dir
echo "Done."
echo "Encrypting directory..."
openssl bf -salt -in temp.tar.gz -out $file.tar.gz.bf
echo "Done."
echo
echo "Here is what the encrypted archive looks like:"
ls -l $file.tar.gz.bf
echo
echo "Is it safe to securely remove $dir? (y)/n"
read remove
if [ x$remove = xn ] || [ x$remove = xN ]; then
  echo "Ok, exiting without removing it."
  srm -fm temp.tar.gz
  exit 0
else
  echo "Ok, removing $dir securely and exiting..."
  srm -rfm $dir
  srm -fm temp.tar.gz
  echo "Done"
fi
exit
```

Finally, a matching script to decrypt the archive back to a directory in the current working directory:

```
#!/bin/sh
#
# Script to decrypt a tar.gz.bf archive

if [ $# -lt 1 ] ; then
  echo "Usage: $0 archive_to_decrypt"
  exit 1
fi

file=$1
dir=`echo $1 | cut -d "." -f 1`
echo -n "Checking if $file actually exists... "
if [ -f $file ] ; then
  echo "Yes."
else
  echo "No. Exiting."
  exit 1
fi
echo -n "Checking to make sure $dir doesn't already exist...
"
if [ -f $dir ] ; then
  # exists
  echo "Yes. Exiting."
  exit 1
else
  # doesn't exist
  echo "No."
fi
echo -n "Checking to make sure tempfile doesn't already
exist... "
if [ -e temp.tar.gz ] ; then
  echo "Yes. Exiting. You need to remove temp.tar.gz."
  exit 1
else
  echo "No."
fi
echo "Decrypting..."
openssl bf -salt -d -in $file -out temp.tar.gz
echo "Untarring..."
tar -zxvf temp.tar.gz
echo "Cleaning up..."
rm temp.tar.gz
echo "All done."
echo
exit
```

GnuPG encrypted files

Gnu Privacy Guard (an open source version of PGP) allows you to encrypt a file using a public key. You would then be able to decrypt the file at a later date using the private key and the key's passphrase.

Unlike FileVault, GnuPG makes use of mlock() and hence doesn't suffer from the same vulnerability. However, it has had a number of its own security concerns.

This section assumes you have already managed to install GnuPG and have created yourself a public/private key-pair. Numerous resources to help you can be found on the web. To then encrypt a file, you would use:

```
gpg -r <your key's name> --encrypt-files <filename>
```

This will create the file filename.gpg. You should securely remove the original plain-text with:

```
srm -fm <filename>
```

Apple's srm is included with OS 10.3 (some users may prefer using the GNU fileutils rm). Similar to the GUN utility shred, srm over-writes the file 7 times with random data before unlinking it from the file-system.

To then decrypt the encrypted file:

```
gpg -r <your key's name> --decrypt-files <filename.gpg >
filename
```

gpg can also be used with just a symmetric cipher and a single password by using the -c switch.

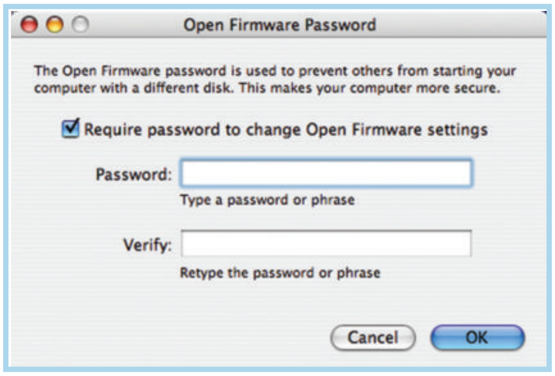
The two scripts in the section REF _Ref87526463 \p \h * MERGEFORMAT above cover en/decrypting entire directories and could be easily modified to use gpg instead of openssl.

CONFIGURING OPEN FIRMWARE PASSWORD

Configuring an Open Firmware (OF) password on your Mac will disable any boot keys when your machine is booting. This means a user with physical access to the machine is unable to boot the machine into target-disk mode, from CD-ROM or into single-user mode.

The simplest way to set an OF password is to use Apple's utility, which can be found at < HYPERLINK "http://www.apple.com/downloads/macosx/apple/openfirmwarepassword.html" http://www.apple.com/downloads/Mac OS X/apple/openfirmwarepassword.html>.

The utility asks for your user password so that it can run sudo nvram to set the OF password and then asks for a password to set as the OpenFirmware password:



The Open Firmware password is used to prevent others from starting your computer with a different disk. This makes your computer more secure.

☑ Require password to change Open Firmware settings

Password:
Type a password or phrase

Verify:
Retype the password or phrase

Setting an OpenFirmware password

To set the password yourself directly from OpenFirmware:

```
<power-button>
option-apple-o-f
password
<enter your password>
setenv security-mode command
reset-all
```

You may wish to remove the OpenFirmware password when you are unable to boot the machine properly and need to re-install, back data up using target mode or boot using single-user.

To do this, remove it directly from OpenFirmware:

```
<power-button>
option-apple-o-f
<enter password>
setenv security-mode=none
nvramrc
reset-all
```

In an emergency, the OpenFirmware password can also be removed by changing the amount of RAM and then resetting the PRAM three times (press and hold option-apple-p-r while powering up until you hear the machine reboot three times). This is obviously also a potential security risk and for this reason, your machine should be physically secured.

You should also be aware that anyone with root/sudo access to the machine can easily get the OpenFirmware password. Like Sun's OpenBoot, OpenFirmware is unagble to hash the password before placing into non-volatile memory. The hex code of the ASCII password can be revealed with, as root:

```
nvram security-password
```

You can then convert the output back to ASCII to get the current OpenFirmware password.

DISABLING FIREWIRE DIRECT MEMORY ACCESS

By default, the FireWire protocol gives the FireWire device access to the host's physical memory. This could potentially be used to suck the entire memory contents out of the machine (including your passwords and current working data). Alternatively, an attacker could determine where in memory the screensaver is and insert some random bytes to crash the screensaver, gaining access to the machine.

An undocumented side-affect of enabling an Open Firmware password (see section above) is that it indirectly disables physical memory access for FireWire devices through the IOFireWireFamily kernel driver.

Disabling FireWire DMA appears to have little affect on the performance of FireWire.

DISABLING SINGLE-USER LOGINS

A default installation, without an OpenBoot password (or with a subverted OpenBoot password), can be booted into a single-user shell by holding down the "S" key during power-up (or boot disk –s from within OpenBoot). This could be used by an attacker with physical access to read your data, add extra accounts or change your passwords.

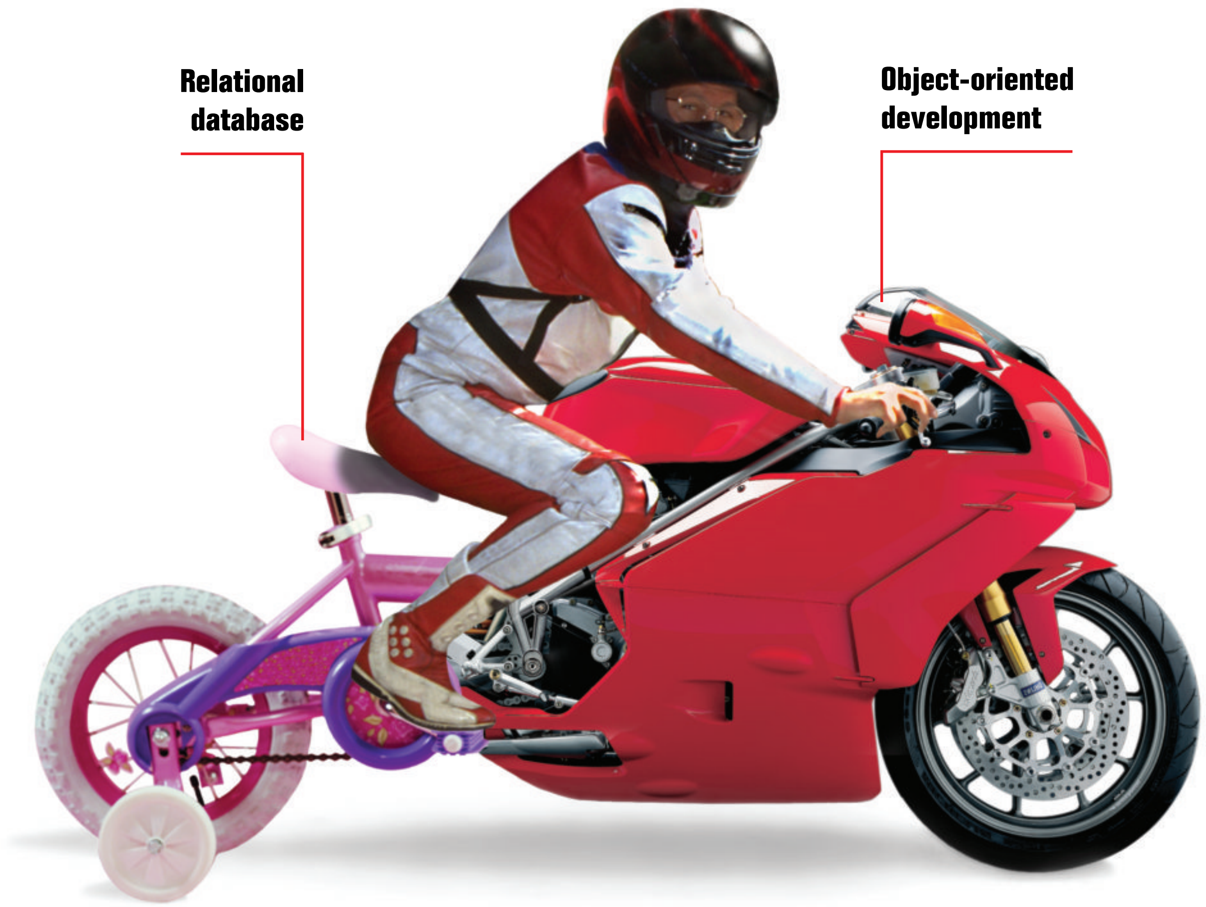
```
The following section introduces a method of ensuring a user
must enter a password before being presented with a root-
user shell as part of a single-user login.
As root:
vi /etc/ttys
:1,$s/secure/insecure/g
:wq
```

To generate a password for root to use when logging into a single-user booted system we use openssl:

```
openssl passwd -salt <xy> <password>
```

Replace <xy> with two random letters to act as salt for the hashing and <password> with the password you want to use for the single-user login. This is completely separate from the local root password, which, if it exists, is stored in the NetInfo database by default.

Now copy the hash that was returned by openssl into your paste buffer, open the file /etc/master.passwd in vi (or your



A BETTER DATABASE CAN SPEED UP YOUR DEVELOPMENT CYCLE

If your relational database isn't a good match for your object-oriented development, you need a new database.

Caché, the *post-relational* database from InterSystems, combines high-performance SQL for faster queries and an advanced object database for rapidly storing and accessing objects. With Caché, no mapping is required between object and relational views of data. That means huge savings in both development and processing time.

Applications built on Caché are massively scalable and lightning fast. They require little or no database administration.

More than just a database system, Caché incorporates a powerful Web application development environment that dramatically reduces the time to build and modify applications.

Caché is so reliable, it's the world's leading database in healthcare – and it powers enterprise applications in financial services, government and many other sectors. With its high reliability, high performance and low maintenance, Caché delivers your vision of a better database.

We are InterSystems, a specialist in data management technology for over twenty-six years. We provide 24x7 support to four million users in 88 countries. Caché is available for Windows, OpenVMS, MAC OS X, Linux, and major UNIX platforms – and it is deployed on systems ranging from two to over 50,000 simultaneous users.

InterSystems CACHÉ™
Make Applications Faster

Try a better database. For free.

Download a free, fully functional, non-expiring copy of Caché or request it on CD at www.InterSystems.com/match3 favourite editor) and replace the asterisk (*) next to "root:" with the hash so the file looks something like:

```
##
nobody:*:-2:-2::0:0:Unprivileged
User:/var/empty:/usr/bin/false
root:8d4Gfm/Dhzw6Q:0:0::0:0:System
Administrator:/var/root:/bin/sh
```

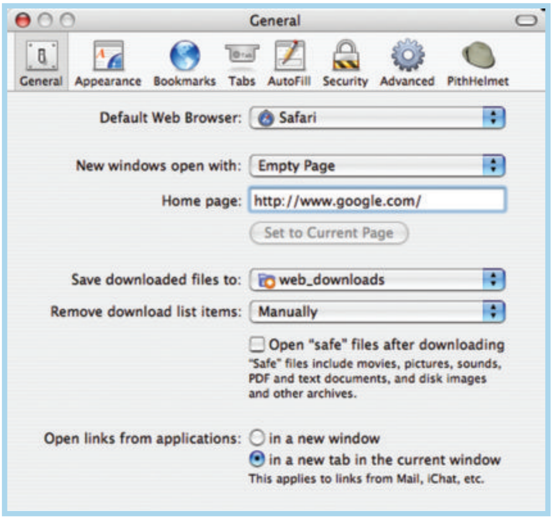
Write the file to disk (with :wq) and exit vi. You will now be asked for the password when booting into single-user.

DISABLE SAFARI AUTO-OPEN

Safari, Apple's web-browser, includes a feature where it will automatically launch a number of different file types with their associated application. This could potentially pose a risk with the user unwittingly opening a file without realising it.

To disable the feature:

- Safari -> Preferences... -> General
- Uncheck Open 'safe' files after downloading



Default Web Browser: Safari

New windows open with: Empty Page

Home page: http://www.google.com/

Save downloaded files to: web_downloads

Remove download list items: Manually

☐ Open "safe" files after downloading
"Safe" files include movies, pictures, sounds, PDF and text documents, and disk images and other archives.

Open links from applications: ○ in a new window
● in a new tab in the current window
This applies to links from Mail, iChat, etc.

Disabling Safari auto-open

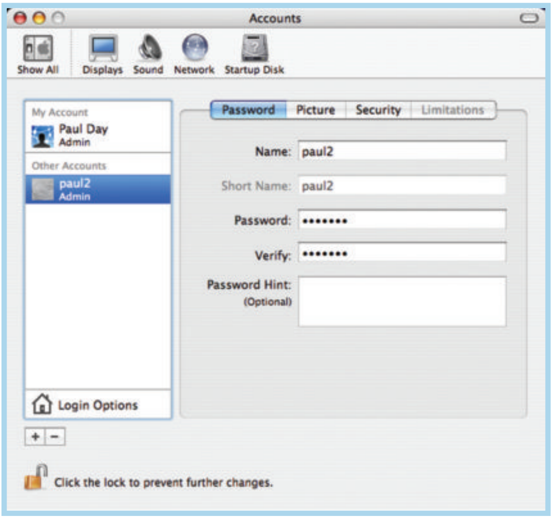
REMOVING OTHER LOCAL USERS

There are other vulnerabilities within Apple's Mac OS X 10.3.6 that have not yet been publicly disclosed and hence won't be discussed in this article. However, it should be noted (although probably obvious) that to ensure the security of your Mac OS X machine, you should avoiding allowing any other local users access to your machine, whether by Fast User Switching or SSH.

Removing normal local users

The cleanest and easiest way to remove extra users is by using the Accounts System Preferences pane:

- Apple menu -> System Preferences -> Accounts
- Select the other account
- Click the minus (-) button -> Delete Immediately



My Account
Paul Day
Admin

Other Accounts
paul2
Admin

Name: paul2

Short Name: paul2

Password: •••••••

Verify: •••••••

Password Hint: (Optional)

Click the lock to prevent further changes.

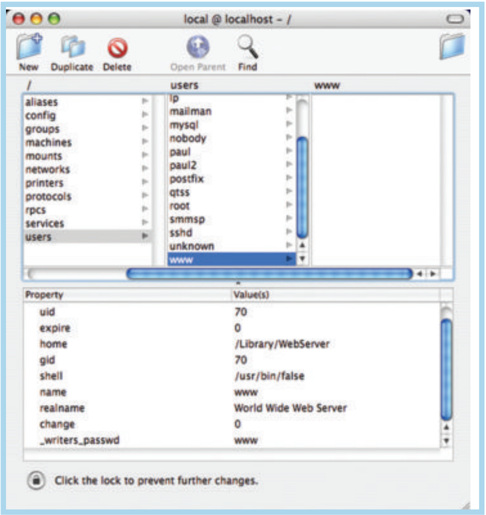
Using the Accounts preferences pane to remove extra users

Checking system user accounts

You may also wish to ensure that no other accounts (not shown in the Accounts preferences pane) have been added by an application installation and left with insecure/default passwords. These could be exploited by an attacker allowing them login to your machine.

To do this you need to make changes within the NetInfo database, either via the GUI or the command line. To remove passwords on extra system accounts using the GUI:

- Applications -> Utilities -> NetInfo Manager -> Domain -> Open -> / -> OK -> / -> users
- Choose a system user -> Ensure it has no passwd entry
- If it does have a password entry, click the lock in the bottom left -> authenticate -> select the passwd line -> Delete
- Close the window -> Save -> Update this copy



/	users	www
aliases	lp	
config	mailman	
groups	mysql	
machines	nobody	
mounts	paul	
networks	paul2	
printers	postfix	
protocols	qtss	
rpcs	root	
services	smmsp	
users	sshd	
	unknown	
	www	

Property	Value(s)
uid	70
expire	0
home	/Library/WebServer
gid	70
shell	/usr/bin/false
name	www
realname	World Wide Web Server
change	0
_writers_passwd	www

Click the lock to prevent further changes.

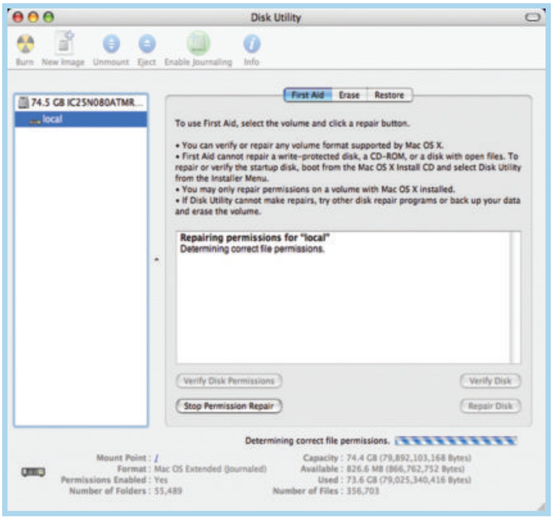
Checking for active users in NetInfo Manager

FIX FILE PERMISSIONS

Over time, permissions and ownership of numerous files may become insecure. This is generally caused by installation of packages put together by non-security-savvy software developers.

To try to correct this situation, it is a good idea to regularly use Apple's Disk Utility to fix file permissions. This can be done by:

- Applications -> Utilities -> Disk Utility
- Select your / disk-partition
- First Aid -> Repair Disk Permissions



To use First Aid, select the volume and click a repair button.

• You can verify or repair any volume format supported by Mac OS X.
• First Aid cannot repair a write-protected disk, a CD-ROM, or a disk with open files. To repair or verify the startup disk, boot from the Mac OS X Install CD and select Disk Utility from the Installer Menu.
• You may only repair permissions on a volume with Mac OS X installed.
• If Disk Utility cannot make repairs, try other disk repair programs or back up your data and erase the volume.

```
Repairing permissions for "local"
Determining correct file permissions.
```

Determining correct file permissions.

Mount Point : /
Format : Mac OS Extended (Journaled)
Permissions Enabled : Yes
Number of Folders : 55,489
Capacity : 74.4 GB (79,892,103,168 Bytes)
Available : 826.6 MB (866,762,752 Bytes)
Used : 73.6 GB (79,025,340,416 Bytes)
Number of Files : 356,703

Repairing file permissions

It can also be done from the command line as root:

```
/usr/sbin/diskutil repairPermissions /
```

The output may look like:

```
Started verify/repair permissions on disk disk0s3 local
Determining correct file permissions.
We are using special permissions for the file or directory
./System/Library/Filesystems/cd9660.fs/cd9660.util.  New
permissions are 33261
Permissions differ on ./private/var/log/install.log, should
be -rw-r--r-- , they are -rw-r-----
Owner and group corrected on ./private/var/log/install.log
Permissions corrected on ./private/var/log/install.log
Permissions differ on ./private/var/log/wtmp, should be -rw-
r--r-- , they are -rw-r-----
Owner and group corrected on ./private/var/log/wtmp
Permissions corrected on ./private/var/log/wtmp
The privileges have been verified or repaired on the selected
volume
Verify/repair finished permissions on disk disk0s3 local
You may choose to add this to root's or the system cron
files, e.g. /etc/weekly.local.
```

diskutil is unable to automatically correct all insecure/incorrect permissions for you. To list all files with potentially insecure or strange permissions, run the following commands as root and examine (or redirect) the output:

To list all setuid/gid (binaries that run with a user or group ID of someone other than the user running then, commonly root) files:

```
find / -type f \( -perm -4000 -o -perm -2000 \) \-exec ls -al
{} \; 2>/dev/null
```

To list all world writable files:

```
find / -type f \( -perm -2 \) \-exec ls -al {} \; 2>/dev/null
```

To list all world writable directories:

```
find / -type d \( -perm -2 \) \-exec ls -ald {} \; 2>/dev/null
```

To list all un-owned files:

```
find / -nouser -o -nogroup \-exec ls -al {} \; 2>/dev/null
```

Based on the output of these commands, you may choose to change or remove permissions to some files manually. Make sure you are fully aware of the purpose of a file before fiddling with its permissions. Random permission changes may result in an unusable system!

REMOVING CLASSIC

Some users may have chosen to install Mac OS Classic support. Classic provides Mac OS 9 emulation support within Mac OS X, which allows a user to seamlessly run an old Mac OS application on their new Mac OS X machine.

If you're not actually using any Classic applications, it is best to disable and remove Classic support entirely. Run the following commands as root:

```
rm -rf /System/Library/PreferencePanes/Classic.prefPane/
rm -rf '/System/Library/Classic/'
rm -rf '/System/Library/CoreServices/Classic Startup.app/'
rm -rf '/System/Library/UserTemplate/English.lproj/Desktop/
Desktop (Mac OS 9)/'
rm -rf '/System Folder/'
rm -rf '/Mac OS 9 Files/'
rm -rf '/Applications (Mac OS 9)'
```

SECURING BLUETOOTH

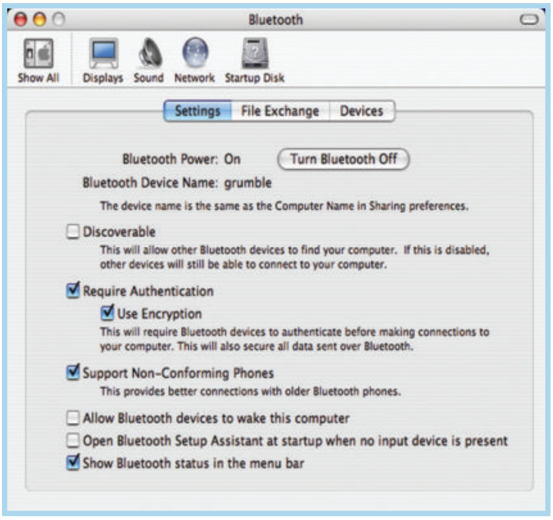
Bluetooth is a radio (2.4GHz) data technology that allows a user to wirelessly connect numerous personal devices to allow communication between them. Bluetooth achieves what is sometimes referred to as a Personal Area Network (PAN), allowing you to, for example, have your mobile phone, hands-free kit, PDA and computer all communicating wirelessly.

Unfortunately, Bluetooth has numerous security drawbacks. This section discusses a number of methods to help lock down Bluetooth on your Mac OS X machine. The methods can also be applied to your other, non-Mac OS X, Bluetooth devices (eg, PDA, mobile phone).

Turn it off

If you're not actively using the Bluetooth connection, you should disable it:

- Apple menu -> System Preferences -> Bluetooth -> Settings
- Turn Bluetooth Off



Bluetooth Power: On

Bluetooth Device Name: grumble

The device name is the same as the Computer Name in Sharing preferences.

☐ Discoverable
This will allow other Bluetooth devices to find your computer. If this is disabled, other devices will still be able to connect to your computer.

☑ Require Authentication

☑ Use Encryption
This will require Bluetooth devices to authenticate before making connections to your computer. This will also secure all data sent over Bluetooth.

☑ Support Non-Conforming Phones
This provides better connections with older Bluetooth phones.

☐ Allow Bluetooth devices to wake this computer

☐ Open Bluetooth Setup Assistant at startup when no input device is present

☑ Show Bluetooth status in the menu bar

Disabling Bluetooth

Put the device in hidden/invisible mode

Your devices only need to be in "visible" or "discoverable" mode when pairing them with your other Bluetooth devices. Once you have paired devices, you should disable visibility. Paired devices are still able to communicate even when not in discoverable mode.

To make your Mac invisible:

- Apple menu -> System Preferences -> Bluetooth -> Settings
- Uncheck Discoverable

Note that invisible/non-discoverable mode does not make your device entirely invisible. It simply makes it harder to find.

Turn on authentication

Once Bluetooth authentication is on, devices generally need to then use a common password to pair with another device, although there are vulnerabilities in some vendor's implementation. To turn on password authentication:

- Apple menu -> System Preferences -> Bluetooth -> Settings
- Check Require Authentication

Turn on encryption

Turning on Bluetooth encryption means that the majority of data transmitted between Bluetooth devices is encrypted with a common key. This makes it difficult for a third party to sniff the data or use recorded data in "replay attacks". To turn on Bluetooth encryption:

- Apple menu -> System Preferences -> Bluetooth -> Settings
- Check Require Authentication -> check Use Encryption

Do not allow auto-acceptance of files

It is best to always be asked for confirmation when accepting a file, stopping a dangerous file or Trojan to be automatically uploaded. To do this:

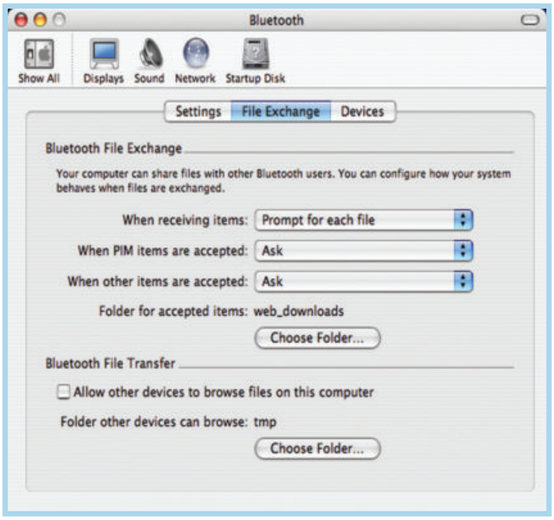
- Apple menu -> System Preferences -> Bluetooth -> File Exchange
- When receiving items: -> Choose Prompt for each file
- When PIM items are accepted and When other items are accepted: -> Choose Ask

If you never use Bluetooth to push files from another device to your Mac, set it to automatically Refuse all.

Disable file shares

If you do not actively share files from the Mac to your other Bluetooth devices, disable all sharing (read-only and read/write) of files:

- Apple menu -> System Preferences -> Bluetooth -> File Exchange
- Uncheck Allow other devices to browse files on this computer



Bluetooth File Exchange

Your computer can share files with other Bluetooth users. You can configure how your system behaves when files are exchanged.

When receiving items: Prompt for each file

When PIM items are accepted: Ask

When other items are accepted: Ask

Folder for accepted items: web_downloads

Bluetooth File Transfer

☐ Allow other devices to browse files on this computer

Folder other devices can browse: tmp

Disabling Bluetooth file sharing

Do not pair with unknown devices

To alleviate the chances of an attacker pairing with your machine, do not pair with an unknown device or allow physical access to your machine to any un-trusted party.

NETWORK SECURITY

The following section describes methods of securing Mac OS X from an external, or network, perspective.

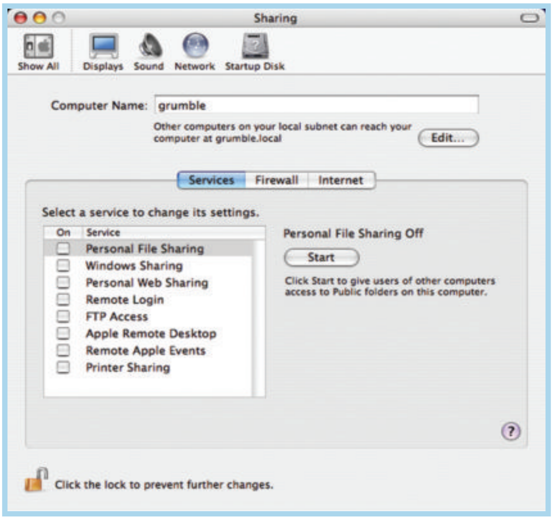
DISABLING SERVICES

By default, Mac OS X does not come with any network services enabled. However, some services may have been enabled unwittingly or by installing extra software. This section describes methods of ensuring unknown services are disabled.

Sharing

Apple's Sharing preference pane is a front-end to xinetd and SystemStarter. It is used to enable and disable a number of common Internet services such as SSH (Remote Login) and the Apache web-server (Personal Web Sharing).

By default, Mac OS X 10.3 comes with all the Sharing network services turned off. However, some users may have enabled services unnecessarily.



Computer Name: grumble

Other computers on your local subnet can reach your computer at grumble.local

Select a service to change its settings.

On	Service
☐	Personal File Sharing
☐	Windows Sharing
☐	Personal Web Sharing
☐	Remote Login
☐	FTP Access
☐	Apple Remote Desktop
☐	Remote Apple Events
☐	Printer Sharing

Personal File Sharing Off

Click Start to give users of other computers access to Public folders on this computer.

Click the lock to prevent further changes.

The Sharing preferences pane

To disable all services:

- Apple menu -> System Preferences -> Sharing
- Uncheck any checked service

A very basic description of each service can be read by selecting the service and reading the description provided below the Start/Stop button.

The following table shows the Apple service name, normal Internet service name, and software associated with providing the service:

Apple Service	Internet Service	Software
Personal File Sharing	AFP(overTCP)	AppleFileServer
Windows Sharing	SMB/CIFS	Samba
Personal Web Sharing	HTTP	Apache
Remote Login	SSH	OpenSSH
FTP access	FTP	tnftpd
Apple Remote Desktop	ARD	ARD Helper
Remote Apple Events	EPPC	AEServer
Printer Sharing	LPR/printer	CUPS

Table showing hostconfig entries and descriptions

If you must have remote access to your Mac, SSH ("Remote Login") is considered to be one of the more secure methods. SSH can also be used for file transfer by using SCP (Secure Copy) and SFTP (Secure FTP). You can also use it for securely tunnelling other services, for example ARD or VNC. See below for instructions on restricting to particular IPs (either through xinetd or ipfw) and securing the default sshd settings.

inetd

Mac OS X uses the xinetd Internet Super Server for providing a number of IP-based services. Some are enabled/disabled through the Sharing preferences pane while many others (including what are commonly referred to as "useless Unix services") aren't. A list of all services it can provide (from a default installation) can be found in /etc/xinetd/.

A listing of any services that have been enabled (either through the Sharing preferences pane or otherwise) can be found by:

```
grep disable /etc/xinetd.d/* | grep no
```

Any services that are not required should be disabled. This can be done by editting the file revealed by the command above and changing the line disable = no to disable = yes. For example, your ssh file may look like:

```
service ssh
{
  disable = yes
  socket_type = stream
  wait = no
  user = root
  server = /usr/libexec/sshd-keygen-wrapper
  server_args = -i
  groups = yes
  flags = REUSE IPv6
  session_create = yes
}
```

Once all unnecessary services have been disabled, you can restart xinetd with:

```
kill -HUP `cat /var/run/xinetd.pid`
```

If you have disabled every service and want to kill off xinetd entirely:

```
kill `cat /var/run/xinetd.pid`
```

If you're choosing to leave a service enabled, you can either restrict what IPs can connect to it within xinetd, or within the ipfw firewall software (see section below). If you decide to restrict it within xinetd, you have the choice of either "allow some, deny rest" or "deny some, allow rest".

As the final line (i.e. above the closing }) within the xinetd configuration file for the service you're restricting, add in your specifications. To "allow some, deny the rest":

```
  only_from = <ip or subnet>, <ip or subnet>, <ip or subnet>
Or to "deny some, allow the rest":
  no_access = <ip or subnet>, <ip or subnet>
```

Insecure services can also be tunnelled with encryption using SSH. Doing so, you leave the service firewalled to the outside world and tunnel a connection into the machine using SSH. OpenSSH itself also has specific user access controls on top of xinetd's and a firewall. See the section below for specifics on securely using SSH.

OSX hostconfig Services

Mac OS X uses a service start-up system called SystemStarter, which replaces the init scripts most people would be familiar with from Unix System V variants. It does include a number of features not available in init, such as including dependencies in the service, rather than relying on manual ordering within a certain run-level.

A number of SystemStarter scripts source the /etc/hostconfig file to see if they should start or not. This file contains variables we can set to quickly enable/disable services at boot time.

The following table lists items you may find in /etc/hostconfig and a short description of what they're used for:

Service	Description
AFPSERVER	Apple File Serving, over TCP for "Personal File Sharing"
AUTHSERVER	Apple NetInfo Authentication service
AUTOMOUNT	Automatic mounting of NFS mount-points (not to be confused with amd)
CUPS	Local printing services
IPFORWARDING	IP routing for other clients
IPV6	IP version 6 protocol support
MAILSERVER	The postfix SMTP mail server
NETINFOSERVER	Bind to a NetInfo server for directory and authentication access
NFSLOCKS	Network File System file locking support
NISDOMAIN	Bind to a NIS domain server for authentication
RPCSERVER	Remote Procedure Call support for numerous Unix services, such as NFS
TIMESYNC	Run NTPd to maintain constant time synchronisation
QTSSERVER	Apple QuickTime Streaming Server modules
WEBSERVER	The Apache web-server for "Personal Web Sharing"
SMBSERVER	Windows file sharing using Samba
DNSSERVER	BIND DNS server
COREDUMPS	Writes a core dump to disk in the case of a kernel panic
VPNSERVER	Apple's VPN service daemon (LT2P and PPTP)
CRASHREPORTER	Apple's crash logging service
XGRIDSERVER	Act as a server for Apple's grid computing software, xgrid
XGRIDAGENT	Act as a client for Apple's grid computing software, xgrid
ARDAGENT	Apple Remote Desktop server

Table showing hostconfig entries and descriptions

Suggested services to enable include CUPS (with -YES-) to allow printing and NETINFOSERVER (with =-AUTOMATIC-), which will load netinfod on a stand-alone machine for authentication.

You can enable ntpd for consistent time synchronisation for meaningful logs if you wish. If you choose to disable it, you may wish to add the ntpdate command to /etc/daily or root's crontab:

```
/usr/sbin/ntpdate -p 8 -u time.asia.apple.com
```

Change "time.asia.apple.com" to a local NTP server closer to your location.

Other OSX Services

Finally, some SystemStarter and mach_init.d scripts don't actually refer to an entry in /etc/hostconfig to see if they should be run or not. These scripts require manual examination.

SystermStarter and mach_init store their scripts in three locations: /Library/StartupItems/, /System/Library/StartupItems and /etc/mach_init.d.

An example service that starts from StartupItems without examining a /etc/hostconfig entry is the NFS server, nfsiod, starting from /System/Library/StartupItems/NFS/NFS. To de-activate it, as root you would edit the script and comment out the line that starts nfsiod:

```
  # nfsiod is the NFS asynchronous block I/O daemon, which
implements
  # NFS read-ahead and write-behind caching on NFS clients.
  #nfsiod -n 4
```

Apple's auto-mount daemon (ADM – not to be confused with the NFS automount service) is used for automatically mounting CDs and image files. It can be disabled in /System/Libraries/StartupItems/AMD/AMD. It also checks /etc/hostconfig for a AMDSERVER:=-NO-, which can be inserted manually (it isn't included in /etc/hostconfig by default).

A default system is unlikely to have any further items that aren't controlled by /etc/hostconfig. However, third-party applications you have installed may. You may wish to examine the contents of each /System/Library/StartupItems/*/* and /etc/mach_init.d/* file to determine what services start automatically.

Finally, you can check for any services left running by using, as root:

```
/usr/sbin/lsof | grep LISTEN
```

DISABLING DIRECTORY ACCESS METHODS

By default, Mac OS X comes with a number of directory access methods enabled, which could be open to exploitation (e.g. the LDAPv3 service accepts an LDAP server from DHCP by default, which could be faked by a rogue DHCP server on the LAN).

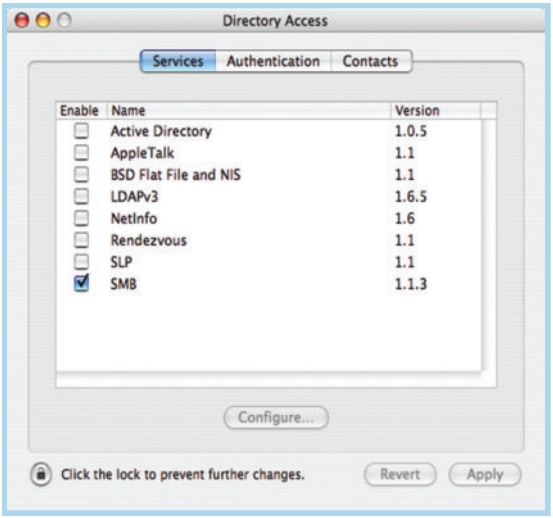
For a stand-alone Mac OS X client, the majority of (or potentially all) services are not required. The following is a table of each of the Directory Access methods and a description of its use:

Directory Access method	**Use**
Active Directory	Windows 2000 domain file sharing and authentication
AppleTalk	Apples legacy protocol for discovering file and print services
BSD Flat File and NIS	/etc flat files and Unix Network Information Service (NIS) or Yellow Pages (yp) directory and authentication
LDAPv3	LDAP directory access and authentication
NetInfo	Apple's directory access and authentication
Rendezvous	Apple multicast protocol for file, print, chat, music and other network services
SLP	Service Location Protocol – open standard file and print server discovery
SMB	Windows workgroup file and print sharing/serving

Table showing Directory Access methods and their use.

To disable services you don't require:

- Applications -> Utilities -> Directory Access
- Uncheck unrequired services

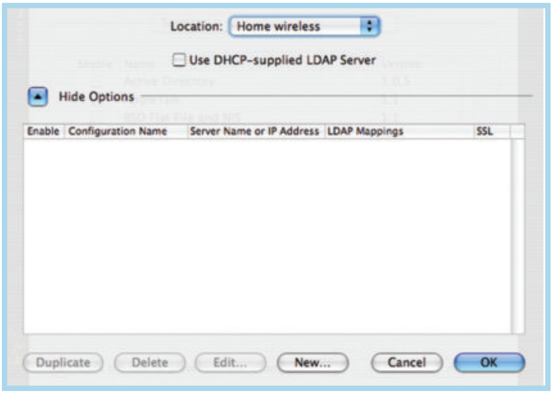


Enable	Name	Version
☐	Active Directory	1.0.5
☐	AppleTalk	1.1
☐	BSD Flat File and NIS	1.1
☐	LDAPv3	1.6.5
☐	NetInfo	1.6
☐	Rendezvous	1.1
☐	SLP	1.1
✓	SMB	1.1.3

Configuring Directory Access

If you need to use LDAP for directory services (such as an enterprise LDAP email address book), ensure you have disabled the DHCP-supplied LDAP Server option:

- Applications -> Utilities -> Directory Access -> LDAPv3 -> Configure
- Uncheck Use DHCP-supplied LDAP Server



Location: Home wireless

☐ Use DHCP-supplied LDAP Server

Enable	Configuration Name	Server Name or IP Address	LDAP Mappings	SSL

Disabling DHCP-supplied LDAP Server

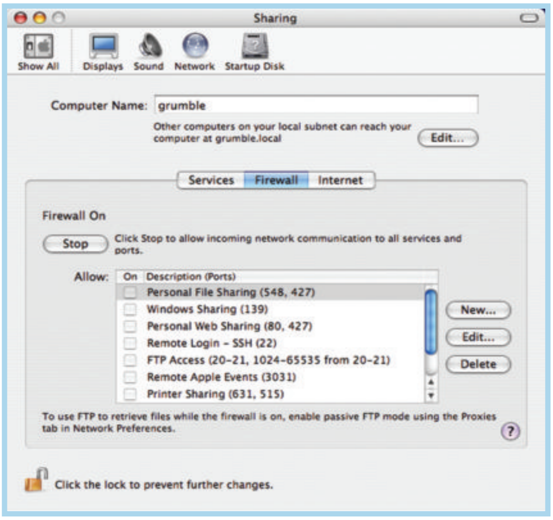
CONFIGURING A FIREWALL

By default, Mac OS X does not come with its built-in firewalling software, ipfw, enabled. The following section shows how best to enable a firewall on your machine.

Mac OS X's built-in firewall configuration

Mac OS X includes a method for enabling a default set of firewall rules within the Sharing preferences pane:

- Apple menu -> System Preferences -> Sharing -> Firewall -> Start



Computer Name: grumble

Other computers on your local subnet can reach your computer at grumble.local

Firewall On

Click Stop to allow incoming network communication to all services and ports.

Allow:

On	Description (Ports)
☐	Personal File Sharing (548, 427)
☐	Windows Sharing (139)
☐	Personal Web Sharing (80, 427)
☐	Remote Login – SSH (22)
☐	FTP Access (20-21, 1024-65535 from 20-21)
☐	Remote Apple Events (3031)
☐	Printer Sharing (631, 515)

To use FTP to retrieve files while the firewall is on, enable passive FTP mode using the Proxies tab in Network Preferences.

Click the lock to prevent further changes.

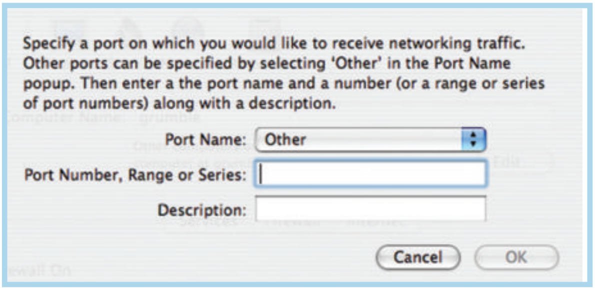
Enabling ipfw through System Preferences

By default, the firewall Sharing install isn't is relatively mediocre from a security point of view, but much better than no firewall at all. The following is a list of the rules it adds:

```
02000 allow ip from any to any via lo*
02010 deny ip from 127.0.0.0/8 to any in
02020 deny ip from any to 127.0.0.0/8 in
02030 deny ip from 224.0.0.0/3 to any in
02040 deny tcp from any to 224.0.0.0/3 in
02050 allow tcp from any to any out
02060 allow tcp from any to any established
12190 deny tcp from any to any
65535 allow ip from any to any
```

If you have enabled services, they will automatically be allowed through the firewall from 0/0 (everyone). If you have installed a third-party service, you may need to manually add it firewall:

- New -> Port Name -> Other
- Port Number, Range or Series: -> Type in the port number/s or range of ports the application needs inbound access for
- Description: -> Type in the name of the service


Specify a port on which you would like to receive networking traffic. Other ports can be specified by selecting 'Other' in the Port Name popup. Then enter a the port name and a number (or a range or series of port numbers) along with a description.

Port Name: Other

Port Number, Range or Series:

Description:

Adding an extra service to the firewall

A number of manually-installed services are already listed in the New window under the Port Name menu.

Manual firewall configuration

The following section discusses designing and implementing a manual firewall script using ipfw.

As root, create the a SystemStarter directory and open its parameter's list in your favorite editor:

```
mkdir /Library/StartupItems/firewall
vi /Library/StartupItems/firewall/StartupParameters.plist
Insert the following into StartupParameters.plist:
{
  Description = "firewall";
  OrderPreference = "None";
  Provides = ("firewall");
  Requires = ("Network");
  Messages =
  {
    start = "Starting firewall";
    stop = "Stopping firewall";
  };
}
```

Next, edit /System/Library/StartupItems/IPServices/StartupParameters.plist and insert the following between Provides and Uses:

```
Requires = ("firewall");
So that it reads:
{
  Description = "Internet services";
  Provides = ("Super Server", "Config Server");
  Requires =("firewall");
  Uses = ("mDNSResponder", "Portmap", "NetworkExtensions");
  OrderPreference = "None";
}
```

This creates a dependency and ensures the firewall has been configured before any network services you've left enabled are loaded. This ensures none of the services are loaded with no protection between them and the outside world.

Finally, open up /Library/StartupItems/firewall/firewall in your editor and, at a minimum, insert the following rule-set. You may wish to add extra rules in the appropriate section from the example rules below this section.

```
#!/bin/sh

## Declare variables
# Path to firewalling software
FW="/sbin/ipfw"

## Flush any existing rules from the firewall
$FW -q flush

## Outgoing

# Drop MS VPC7 license checking going out
$FW add deny udp from any to any 21790
# Drop MS Office license checking going out
$FW add deny udp from any to any 2222
# Allow pretty much anything else out
$FW add allow all from any to any out

## Incoming

# Allow all from/to local loopback interface
$FW add allow all from any to any via lo0
# Then deny anything pretending to come from 127 on other ifs
$FW add deny log all from 127.0.0.0/8 to any in

# Allow relevant outgoing connections back in
# Allow half open TCP back in (although not active ftp)
$FW add allow tcp from any to any established in
# Allow related UDP back in
# DNS - UDP/53
$FW add allow udp from any 53 to any 1024-65535 in
# NTP - UDP/123
$FW add allow udp from any 123 to any 123 in
$FW add allow udp from any 123 to any 1024-65535 in
# DHCP - UDP/67
# DHCP request to server back in to client
$FW add allow udp from any 67 to any 1024-65535 in
# DHCP offer from server in to client
$FW add allow udp from any to any 68 in
# Allow the neccesary ICMP in
# (echo reply, dest unreachable, ttl exceeded, IP header bad)
$FW add allow icmp from any to any icmptypes 0,3,11,12

###
### Insert your custom rules here
###

# Reject IDENT/AUTH with an ICMP reply
$FW add reject tcp from any to any 113 in

# Deny (drop without ICMP) the rest and log to
/var/log/system.log
$FW add deny log all from any to any

exit
```

Some rules you may wish to insert could include the following:

```
# Windows/SMB/Samba client access
$FW add allow udp from any 137-139 to any in
$FW add allow udp from any 445 to any in
$FW add allow tcp from any 137-139 to any in
$FW add allow tcp from any 445 to any in

# PPTP VPN client access
# (replace <ip> with your VPN server's IP)
$FW add allow 47 from <ip> to any in

# H.323 client access (NetMeeting and similar)
$FW add allow udp from 0/0 to 0/0 1720 in
$FW add allow tcp from 0/0 to 0/0 1720 in
$FW add allow tcp from 0/0 to 0/0 30000-30010 in
$FW add allow udp from 0/0 to 0/0 5000-5099 in
```

```
# XWindows client in an XNest running in display :1
# (replace <ip> with the Unix box's IP)
$FW add allow tcp from <ip> to any 6001 in

# XDMCP client in an XNest running in display :2
# (replace <ip> with the Unix box's IP)
$FW add allow tcp from <ip> to any 6002 in
$FW add allow udp from <ip> 177 to any in

# SSH server
$FW add allow tcp from 0/0 to any 22 in
```

To determine what TCP or UDP port a service uses (so that you can let incoming requests through your firewall), you can check the /etc/services file:

```
grep -i <service name> /etc/services
```

Monitoring ipfw

The final rule in the above script tells ipfw to log any packets hitting the final deny rule before silently dropping them. As root, you can see which packets are being dropped with a command like:

```
/usr/bin/tail -f /var/log/system.log | grep ipfw
```

KERNEL TWEAKING

The following section describes a number of kernel variables that should be set to ensure the most secure network settings. Insert the following into /etc/sysctl.conf to ensure they're at their most secure:

```
# Verbose firewall logging
net.inet.ip.fw.verbose=1
net.inet.ip.fw.verbose_limit=65535
# ICMP limit
net.inet.icmp.icmplim=1024
# Stop redirects
net.inet.icmp.drop_redirect=1
net.inet.icmp.log_redirect=1
net.inet.ip.redirect=0
# Stop source routing
net.inet.ip.sourceroute=0
net.inet.ip.accept_sourceroute=0
# Stop broadcast ECHO response
net.inet.icmp.bmcastecho=0
# Stop other broadcast probes
net.inet.icmp.maskrepl=0
# TCP delayed ack off
net.inet.tcp.delayed_ack=0
# Turn off forwarding/routing
net.inet.ip.forwarding=0
# Turn on strong/randomized TCP sequencing
net.inet.tcp.strict_rfc1948=1
```

They can also be manually entered at the command line (or in another script) at any time with the following syntax as root:

```
/usr/sbin/sysctl -w <variable>=<setting>
```

SECURING SSH

SSH (Secure Shell), is provided under Mac OS X using the open-source package OpenSSH. It can be used for a secure remote interactive shell (SSH), secure file transfer (SFTP), secure copy (scp), secure X-windows forwarding (X11Forwarding) and encrypted tunnelling of other IP services.

General SSHd changes

SSHd is highly configurable and can be further locked down from it default settings. Its server configuration file can be found under Mac OS X as /etc/sshd_config and the following changes from the default configuration are recommended:

```
#Protocol 2,1
(to)
Protocol 2

#PermitRootLogin yes
(to)
PermitRootLogin no

Subsystem sftp /usr/libexec/sftp-server"
(to)
#Subsystem sftp /usr/libexec/sftp-server
```

Using SSH keys for authentication

It is considered more secure to login with an SSH key pair than a password. A machine that has already been hacked may have a trojanned sshd binary or authentication services which may be able to give a copy of your password to the attacker. If you have the same password on multiple machines (which is obviously not recommended) they may then login to those other machines using your credentials.

On the other hand, logging in with an SSH key does not allow an attacker to gain your password, even if you are using the same SSH key (with same passphrase) to login to other machines. To disable password authentication:

```
#PasswordAuthentication yes -> PasswordAuthentication no
```

To generate an SSH key pair on your external machine (assuming it runs OpenSSH):

```
user@host:~$ ssh-keygen -b 4096 -t dsa -C "Key for user@host
Nov 2004"

Generating public/private rsa key pair.
Enter file in which to save the key (/Users/user/.ssh/id_rsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /Users/user/.ssh/id_dsa.
Your public key has been saved in /Users/user/.ssh/id_dsa.pub.
The key fingerprint is:
f3:99:d7:05:be:7f:41:42:64:97:b1:e7:d1:41:c9:08 Key for
user@host Nov 2004
```

DSA is considerably faster than RSA for key generation and signing, however some argue that DSS has some potential security flaws in its signing process on machines with low random number entropy.

Ensure you add a pass-phrase to your key to protect it if the remote machine is compromised.

Now put ~/.ssh/id_dsa.pub from the remote machine into ~/.ssh/authorized_keys on your Mac. Your key will now be automatically used instead of a password for SSH, SCP and SFTP remote access to your machine.

Forwarding X11 through SSH

Finally, if you have X11 programs that you want to export back to a remote machine, it is recommended that you use SSH's in-built

X11 Forwarding in /etc/sshd_config:

```
#X11Forwarding no
(to)
X11Forwarding yes
```

From the client machine, you setup the SSH tunnel by typing:

```
ssh -X -l username <remote Mac>
```

Tunnelling other IP services through SSH

SSH can also be used to tunnel an otherwise insecure protocol through it.

For example, you may wish to use a VNC server running on the Mac OS X machine. VNC by itself is not encrypted and it's password is sent plain-text over the network. A somewhat more secure solution to this problem is to leave the SSH port firewalled, tunnel a VNC connection through to the machine and connect to the VNC port on it's loop-back interface.

For example, to make a tunnel through to the remote Mac's TCP port 5900 (commonly VNC), you would do:

ssh –N –L 5900:127.0.0.1:5900 <remote Mac>

This command binds SSH to port 5900 on the localhost and tunnels it, via SSH, to port 5900 on the remote Mac. You would now point your VNC client to 127.0.0.1 (ie, the localhost's loopback interface) on port 5900 and it will securely connect to the VNC server on your remote Mac.

Restarting sshd after config changes

Because Mac OS X spawns sshd from xinetd rather than as a stand-alone server, there is no need to restart anything. Changes you make to sshd_config are read in on the next connection to that service.

CONCLUSIONS

With the move from Mac OS Classic's roots to a Unix-based operating system, Apple's Mac OS has undergone massive changes.

While it is one of the more secure Unix operating environments by default, there are a number of methods the administrator of the machine can make use of to harden the environment further.

This article has outlined a number of these methods to secure Mac OS X from a local and network perspective.

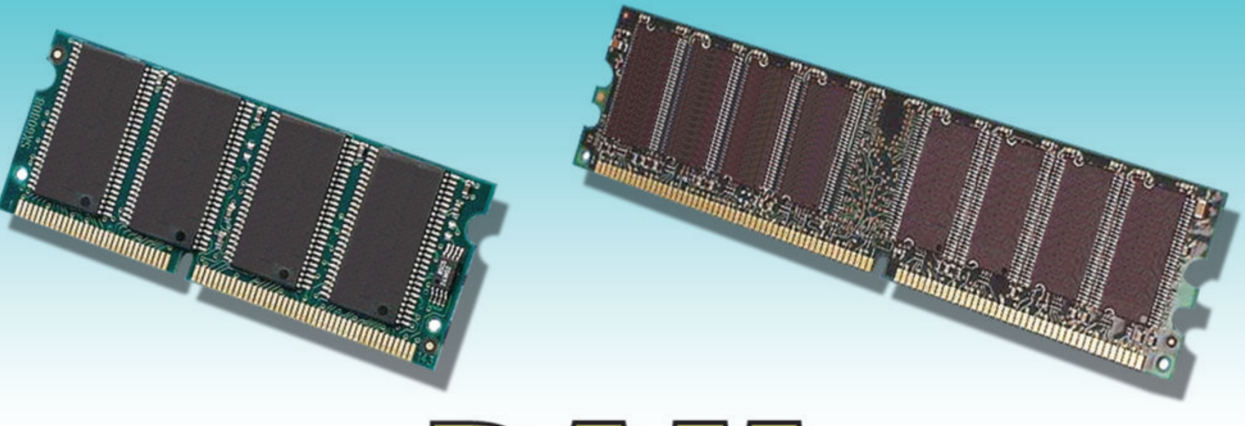
REFERENCES

- de Vries, Stephen. "A Corsaire White Paper: Securing Mac OS X", Corsaire, 22 June 2004
- Systems and network Attack Centre (SNAC), National Security Agency (NSA). "Apple Mac OS X v10.3.x "Panther" – Security Configuration Guide", 2004
- Beale, Jay, JJB Security Consulting, LLC and GWU Cyber Security Policy & Research Institute. "Locking Down Mac OS X" Delivered at Black Hat USA, 2003.

MT

About The Author

Paul Day is a Security Administrator who currently works for the University of Western Australia. He primarily deals with security policy and the administration of security in Cisco, Solaris, Linux and OS X based environments. His passions revolve around Unix & network security and optimization



betterRAM.com sm

Memory built to order - so you always get the right memory with up-to-the-minute prices!

Toll Free: (800) 895-3493 • Outside US/Canada: 805-494-9797 • Fax: 805-494-9798

The Motorola RAZR is cool, but I want to use Bluetooth!

By the time you read this article, Tiger will likely have shipped. Reportedly, Apple will be enhancing the connections via iSync, Bluetooth, and more to a variety of phones.

But, what if you have that slick new Motorola phone (like the RAZR), and you are using Panther? You can sync the phones via a cable in the mean time, but what about connecting to the net?

I took the challenge. I wanted to get my PowerBook running 10.3.8 to go over Bluetooth to my Motorola RAZR phone and dial up to the net. Not fast, but it worked. I believe this will work on any newer Motorola phone (e.g., 551 as an example) as they have similar interfaces.

As you may know, Cingular had an early exclusive on the RAZR, so if you have this phone, you probably have Cingular.

If you call Cingular, you will find them to be of little (or more likely no) help. In searching around the net, I found several bits of information, and the below is the best instructions I've been able to cobble together ... and have actually gotten to work myself.

You will need some sort of data package with Cingular, but you don't need to spend the $80/month that they tell you at the first level support. It's not competitive with other services, and not reasonable to ask. I've got a MediaNet package for $20 that's unlimited that I'm using. (I've been told that they no longer offer the unlimited version, but that might be regional restrictions.)

a) First, you have to download the latest modem scripts. There are several places, but you can download updated modem scripts http://www.taniwha.org.uk>http://www.taniwha.org.uk (You want "Motorola GPRS Scripts")

b) Put the new scripts in the following folder:
[your hard disk]/Library/Modem Scripts

Now you have to configure your Network settings.

c) Go to your System Preferences/Network

d) Go to your "Network Port Configurations" and create a new Bluetooth port

e) First Tab TCP/IP:
Using PPP
DNS Servers: 66.209.10.201, 66.209.10.202
(these are the Cingular DNS machines)

e) Next tab PPP:
Service Provider: CINGULAR
Account Name: WAP@CINGULARGPRS.COM
Password: CINGULAR1
Telephone Number: WAP.CINGULAR

Note: If you do this in mixed or lower case, it won't work. Make sure it's all caps ... strange as that may seem.

Then click on the PPP Options button and uncheck both "Send PPP echopackets" and "Use TCP header compression", click OK

f) Bluetooth Modem tab: Modem: Choose Motorola GPRS CID2 (If you dropped in the new modem scripts into the correct folder, you should have this in your drop down menu.

g) Open up your Internet Connect application (Found in your Applications Folder) and select Bluetooth and click connect.

That's it. Happy connecting!

MT

by Neil Ticktin, Publisher

NEW TECHNOLOGY

Podcasting 101

How to create your own podcast

By August Trometer

LIVING THE DREAM

OK, I admit it: I've always wanted my own radio show. You know, a show where I could talk about the things that interest me, play the songs that I like, and interview fascinating people. The unfortunate truth is that radio is a tough business to get into, and I never got to have that show. Until now.

Podcasting is the hottest new term on the internet. In the last few weeks, a Google search for podcasting has gone from only a few hundred results to well over half-a-million. Some media analysts predict that podcasting is going to change broadcasting forever. Why? Because with podcasting, everyone can have their own show. Instead of a huge studio, all you need is a computer and a bit of software. In fact, in the method I'm going to describe, you can begin podcasting for under $50. This article will show you how.

WHAT IS PODCASTING?

The idea behind podcasting is simple. It's like Tivo for internet audio. The term podcasting is a play on the word broadcasting, but it turns traditional broadcasting on its head. Programs aren't streamed, like radio, but instead delivered, like magazine subscriptions, right to your desktop. With the right client software, all this happens transparently, and you wake up with fresh content to listen to all day long. The beauty of this method is that you're no longer tied to the show's schedule – you can listen to the programs when you have time, rather than when they're "on."

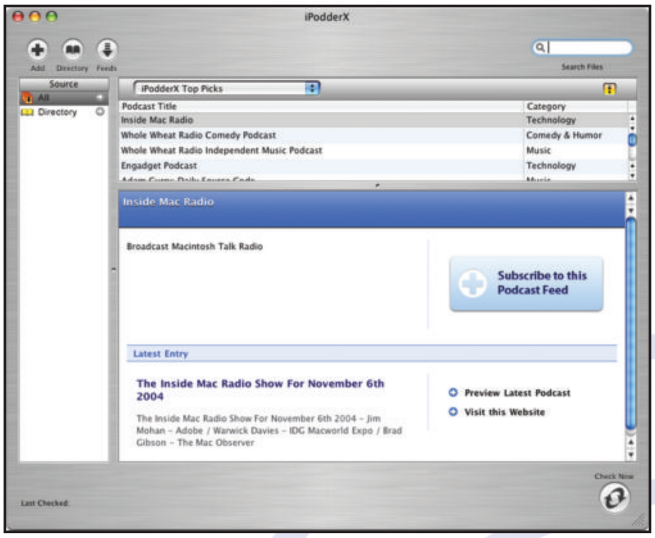
Podcasting works using syndication feeds, such as those using RSS, to deliver these shows to you. If you use a newsreader, you already know how to subscribe to a podcast, since it is exactly the same as a newsfeed with just an enclosed file. Podcast clients let you subscribe to syndication feeds. If there are any files available, the client automatically downloads them for you.

There are hundreds of people producing podcasts, and more shows are popping up every day. Even some of the big media names are starting to take interest as more and more people turn to podcasts rather than regular radio for their information and entertainment.

Even more interesting is that anyone can create and distribute his own podcast. The power of today's computers and the vast reach of the internet mean that having your own radio show is nearly as easy as hitting record on your Mac.

RECEIVING PODCASTS

Before you start creating your own show, it's probably a good idea to listen to some of the other podcasts out there. To do that, you'll need a podcast client. For the Mac there are currently two clients available, iPodderX (http://iPodderX.com), which is shareware, and iPodder Lemon (http://ipodder.sourceforge.com), which is open source. Both have their own set of features, but the basic functionality of each is similar.



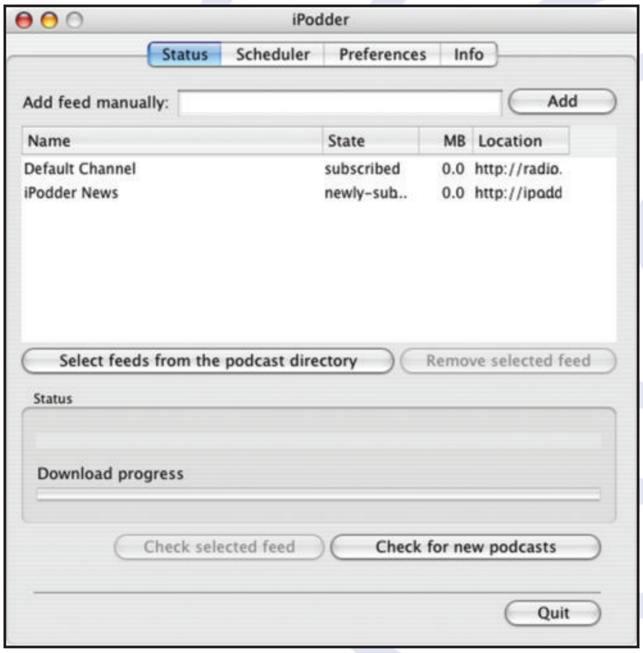
Podcast Title	Category
Inside Mac Radio	Technology
Whole Wheat Radio Comedy Podcast	Comedy & Humor
Whole Wheat Radio Independent Music Podcast	Music
Engadget Podcast	Technology

Broadcast Macintosh Talk Radio

The Inside Mac Radio Show For November 6th 2004

The Inside Mac Radio Show For November 6th 2004 – Jim Mohan – Adobe / Warwick Davies – IDG Macworld Expo / Brad Gibson – The Mac Observer

Figure 1. iPodderX



Name	State	MB	Location
Default Channel	subscribed	0.0	http://radio.
iPodder News	newly-sub..	0.0	http://ipodd

Figure 2. iPodder Lemon

Download and install the client of your choice. Launch it, and you can begin adding feeds. Both clients feature an integrated podcast directory listing many of the available podcasts, so feel free to browse and choose the podcasts that interest you. Both clients can also be set to automatically download podcasts, so you don't need to manually activate the downloads. I highly recommend that you find this setting and turn it on.

Once you're all set up, the fun begins. Whenever your client checks for podcasts, it looks at the list of feeds you've subscribed to. If there are any available files, it automatically downloads them to your computer. Now the magic: if the file is an audio file, it will be automatically moved to iTunes for you, where you can listen at your leisure. If you have an iPod, the next time you sync, all those podcasts will be copied to your iPod. You'll have fresh content to go!

YOUR FIRST PODCAST

Now that you've familiarized yourself with the client-side of podcasting, it's time to start thinking about your own show. What kind of show do you want to do? The sky is the limit. Some shows are over half hour in length and feature music, commentary, or interviews. Others are short. KOMO, out of Seattle, has brief two to three minute podcasts containing their news stories. One podcast I know is simply a guy reading a bit of poetry each day. Use your imagination, and who knows what you might come up with.

After you've decided on a format, the next step is a bit of paperwork. I've noticed that the most popular shows are the ones that are the most professional. By professional, I don't mean slickly produced. Instead, I'm simply talking about a little preparation so your show goes smoothly, so taking a few minutes to organize yourself will help increase your eventual listenership.

I suggest sitting down and drafting a brief outline of your show. Perhaps you'd like a music intro. Then maybe a few minutes of commentary. Then another song. Whatever you decide, put it on paper. That way, as the show goes along, you can refer to the outline to make sure you don't have any embarrassing gaps of silence in the program.

Tape the outline to your computer monitor so that it's in easy view. If you've ever been to a real recording studio, they do the same thing. You don't want the sounds of rustling paper to be recorded, so with it taped to the monitor you can refer to it without handling it.

THE GEAR

There are probably as many ways to set up a podcasting studio as there are podcasts. The method I'm going to show you I use for two reasons. One, it's easy to set up. Two, it's cheap. For under $50, you've got a podcasting studio that's ready to roll. I suspect that sooner rather than later someone will develop a Podcasting Studio application. Until then, we need to use several bits of software.

Here's the list of things you'll need:

- A microphone. Most Macs have a built-in microphone. If you's doesn't, you can use an iSight (you've been looking for an excuse, right?) or a USB mic, such as Griffin's iMic. This article will assume you're simply using the built-in mic.
- Headphones or earphones. You'll need to wear earphones during the entire podcast, otherwise, you'll end up with feedback which will ruin your recording.
- iTunes or QuickTime. You'll use these to play audio files.
- iChat or Skype. If you want to interview remote guests, you can do so using an Audio iChat or Skype.
- WireTap from Ambrosia Software (http://www.ambrosiasw.com/utilities/freebies). WireTap is

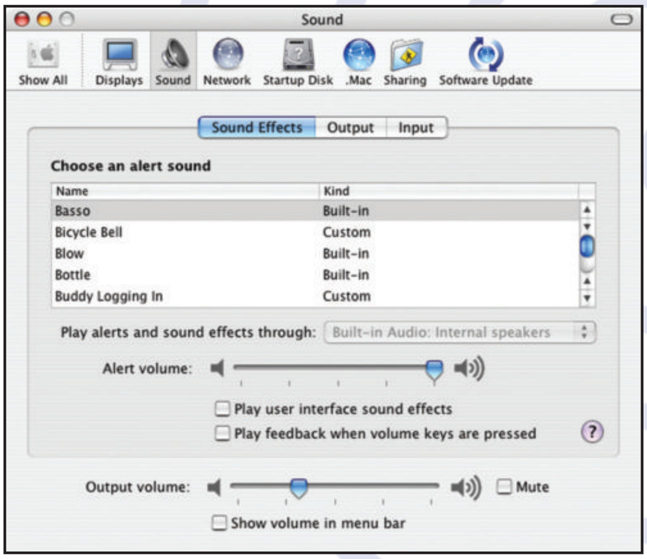
freeware, and it's what we'll use to capture your computer's output and record the audio.

- Audio Hijack Pro from Rogue Amoeba (http://rogueamoeba.com/audiohijackpro/). Audio Hijack Pro is $32. There is a Demo version, so you can try it before paying for it. If you have GarageBand, you can use it instead of Audio Hijack Pro, but it tends to be a little more resource hungry.

SETTING IT ALL UP

The job of creating the podcast breaks down into two basic tasks: recording what you say into the microphone, and recoding the audio output from your Mac. While recording, you want to make sure that nothing is recorded other than the show. So you need to be in a quiet room, with the TV and stereo turned off.

You'll also need to make sure that any error sounds from your Mac are not recorded. The easiest way to fix this is to turn them off. In your System Preferences, choose the Sound panel, then choose the Sound Effects tab. Make sure the checkboxes for playing interface sounds and volume sounds are unchecked.



Choose an alert sound

Name	Kind
Basso	Built-in
Bicycle Bell	Custom
Blow	Built-in
Bottle	Built-in
Buddy Logging In	Custom

Play alerts and sound effects through: Built-in Audio: Internal speakers

Alert volume:

☐ Play user interface sound effects

☐ Play feedback when volume keys are pressed

Output volume: ☐ Mute

☐ Show volume in menu bar

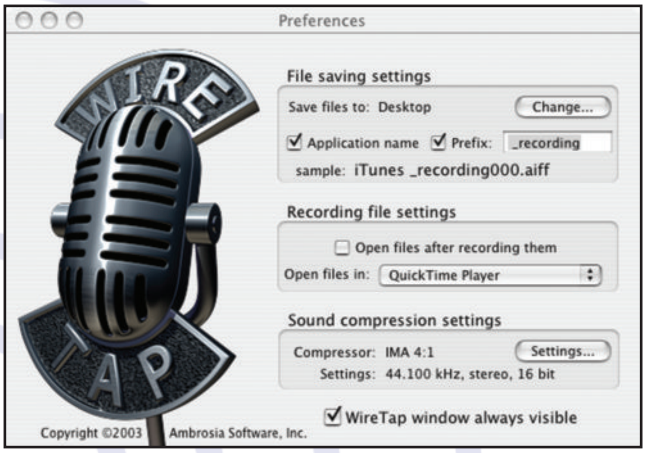
Figure 3. Sound Effects Off

While you're here, click the Input tab and check your microphone settings. These will need adjustment as you become accustomed to how your particular setup records sound. The idea here is to make sure your recording levels don't clip, but the sensitivity is still strong enough for you to be heard.

You also need to make sure to quit any non-essential applications. Recording audio can be a processor intensive task, and any stray processes running on your system will slow things down considerably. You'll also want to make sure you turn off the automatic checking on your podcast client. Invariably, as soon as you set down to record your client app will start downloading files, causing you all kinds of headaches.

Plug in your headphones, put them on, and launch all the apps you are going to use for your podcast, including WireTap and Audio Hijack Pro. You'll also want to get any audio clips you want to play ready as well. Put them within easy reach or in a playlist in iTunes. Finally, if you're going to have a remote guest via Audio iChat, it's time to get them ready as well.

We're going to use WireTap to record all audio output by your Mac. It's very easy, just configure it the way you like, and press the record button. WireTap always saves your file as an AIFF audio file. Once it's recorded, we'll convert it to an MP3 or AAC in iTunes.



File saving settings

Save files to: Desktop

✓ Application name ✓ Prefix: _recording

sample: iTunes _recording000.aiff

Recording file settings

☐ Open files after recording them

Open files in: QuickTime Player

Sound compression settings

Compressor: IMA 4:1

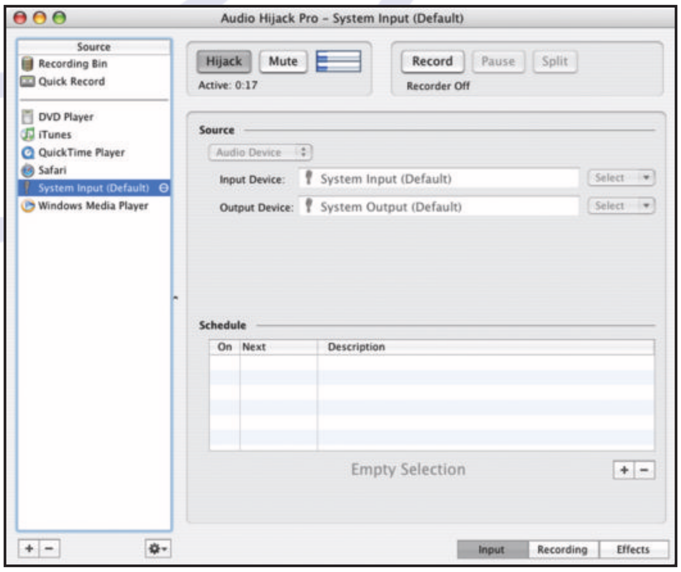
Settings: 44.100 kHz, stereo, 16 bit

✓ WireTap window always visible

Copyright ©2003 Ambrosia Software, Inc.

Figure 4. WireTap Settings

The main problem with WireTap is that it won't record the input from your microphone. We need to use another application to monitor your mic that WireTap can record from. That's what Audio Hijack is for. In Audio Hijack, you'll see a list of potential sources in the left hand pane. Choose System Input (Default). Then, click the Hijack button in the main window. You do not need to turn on recording – that's what WireTap is for.



Active: 0:17

Recorder Off

Source

Audio Device

Input Device: System Input (Default)

Output Device: System Output (Default)

Schedule

On	Next	Description

Empty Selection

Figure 5. Audio Hijack Settings

In Audio Hijack, you can also add some effects to your voice. In the lower right corner, click the Effects tab. Click any space to insert an Effect, and a list of possible effects will pop up. Choose what you'd like here, but unless you have a specific purpose in mind, don't go too crazy. A robot voice is fine for a few seconds, but several minutes of it can be tiresome. I use the Bass and Treble effect to boost the bass in my voice. It helps give me that "radio sound" that we all know so well.

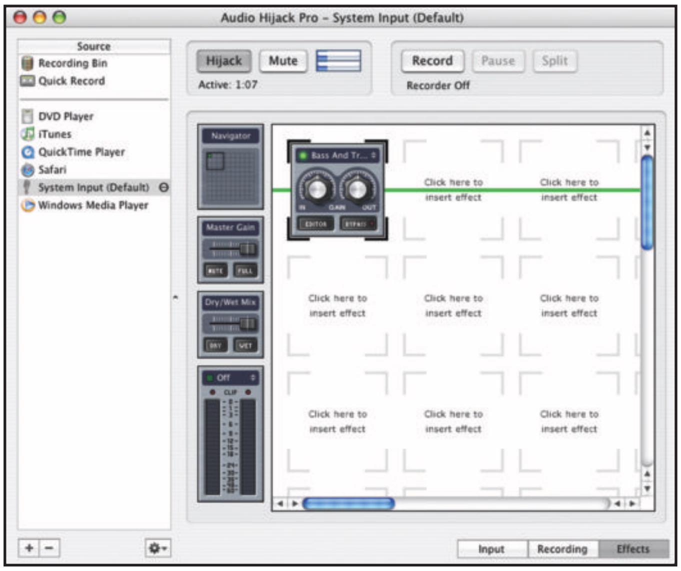


Figure 6. Adding a Voice Effect

THE CURTAIN RISES

You're all set to go! In WireTap, press the Record Button to begin the show. Follow your outline. If you need to play something in iTunes, go ahead, and WireTap will record it automatically. Call up your pal on iChat, and they can be a guest on your show as well.

When you're done, just click the Stop button in WireTap. The entire file will be saved to your hard drive in the location you specified.

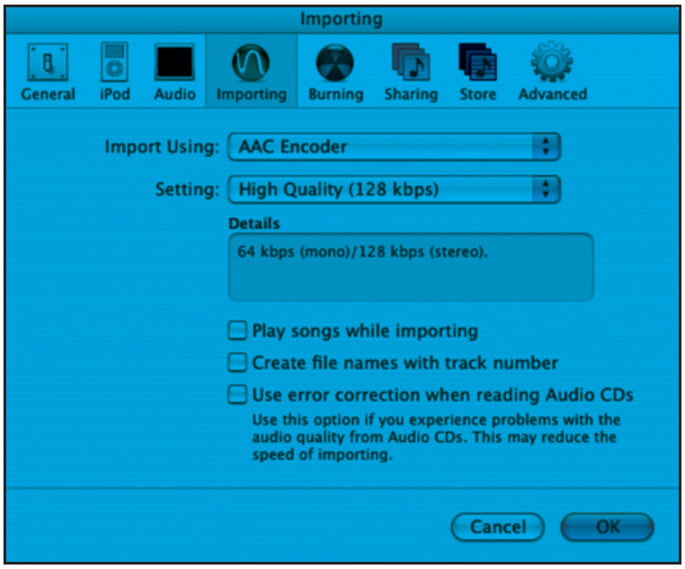
That's it! You just recorded your first podcast!

At this point, you're finished with the recording. The method described above will give you a single file, nearly ready to publish, but you could just as easily record small clips and edit them together with an audio editor to form a longer piece. If you'd like, you can also edit to remove any long pauses, erms, and ahs from your recording.

COVERSION AND PUBLISHING

As I mentioned, WireTap records the file in AIFF format. We need to convert that to either MP3 or AAC.

Drag the file into the iTunes window. The podcast will be copied into iTunes. In the iTunes Preferences, choose the Importing tab. Set these preferences however you'd like them. You'll want to use either MP3 or AAC. Set the bitrate to whatever you'd prefer, but keep in mind that the higher the bitrate, the bigger the file. This means longer download times for the listeners and higher bandwidth usage for you.



Import Using: AAC Encoder

Setting: High Quality (128 kbps)

Details

64 kbps (mono)/128 kbps (stereo).

☐ Play songs while importing

☐ Create file names with track number

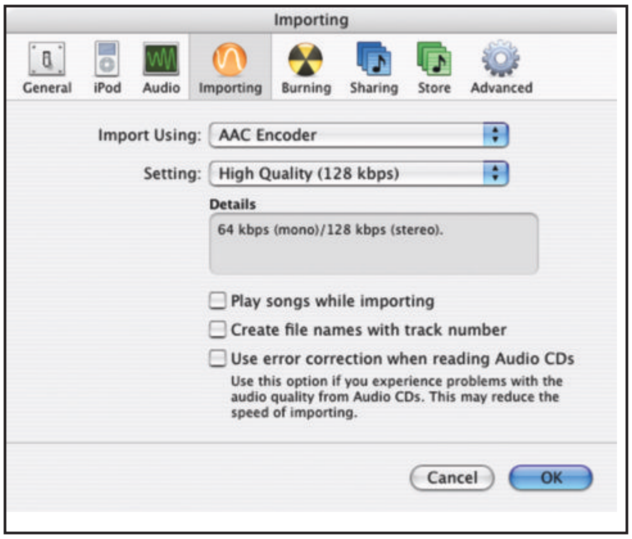
☐ Use error correction when reading Audio CDs

Use this option if you experience problems with the audio quality from Audio CDs. This may reduce the speed of importing.

Figure 7. iTunes Import Settings

Control-click on the podcast file in your iTunes library. A contextual menu will pop up. Choose "Convert Selection to AAC" (or MP3), and iTunes will convert your AIFF file to the proper format.

Finally, we need to change some ID3 tags so that the listeners will have it properly added to their iTunes library. Select the converted file in iTunes, then type Command-I to bring up the file's information. Change this as you see fit. Generally, you should put the name of the podcast ("Mac News") in the Album slot, while the name of the individual episode ("MacWorld 2005") goes in the Name blank. It's also nice if you set the Genre to 'Podcast', so those who subscribe to your show can sort by Genre in iTunes to get all of the podcasts. You can also, if you'd like, give the file some "Cover Art." Simply drag an image, say the show's logo, into the image well in iTunes. It will be converted and stored in the ID3 tag.



Import Using: AAC Encoder

Setting: High Quality (128 kbps)

Details

64 kbps (mono)/128 kbps (stereo).

☐ Play songs while importing

☐ Create file names with track number

☐ Use error correction when reading Audio CDs

Use this option if you experience problems with the audio quality from Audio CDs. This may reduce the speed of importing.

Figure 8. ID3 Tag Settings

GETTING IT ONLINE

OK. You've got the show recorded, and you've got it converted to MP3 or AAC. All that's left is getting it on the web. Upload the file to a webserver you have access to. This could be your .Mac iDisk, an ISP, or even your own home computer.

You've probably already got a weblog, and if you've got a weblog, it's most likely already got a newsfeed. More and more weblog publishing tools are adding support for podcasting. WordPress and MoveableType both have plugins available that will format your RSS feed for podcasts. If yours doesn't, you'll need to hand code it. While syndication feeds are far beyond the scope of this article, I'll show you the one small change you need to make to your RSS 2.0 feed.

In the RSS 2.0 feed (2.0, only, not .91,.92, or 1), between the <item> tag of the appropriate item, you need to add an <enclosure> tag with the following format:

```
<enclosure url=http://your.podcast.com/file.mp3 length="5404566"
type="audio/mpeg" />
```

The url is the URL of the file, the length is the size of the file in bytes, and the type is the MIME type of the file. All three are required.

Once that tag is added to your feed, any podcast client that has subscribed to your feed will automatically get your new podcast.

A COUPLE OF CONCERNS

Since podcasting is such a new phenomenon, there are few kinks that still need ironed out. One of the big ones is bandwidth. If you have a successful podcast, for example, be prepared to use a lot of extra bandwith. If your file is, say, 10MB in size and 500 people listen to it, you're looking at a pretty big bandwidth bill if you pay by the GB.

To alleviate the bandwidth concerns, both client developers and content producers are experimenting with alternatives such as BitTorrent, but there's no real magic bullet yet. Your bandwidth costs will likely go up, so just be aware of that.

Also, be careful of the music you play during your show. If it's copyrighted, and you don't have the rights to play it, you could be opening yourself up for legal trouble. There is a lot of Creative Commons music available online that is royalty free, or you could create your own with GarageBand. Above all, be careful with what you publish.

NOT JUST AUDIO

We've spent this entire article talking about audio, but syndication enclosures can be any type of file. With a good podcast client, images will be moved into iPhoto where they can be synced with the iPod Photo, while video podcasts (which can't go on your iPod yet) will be saved to your hard drive where you can watch them with QuickTime. Even sharing Applications is possible in a podcast feed! Use your imagination – there's no end to what can be done with this technology!

THE SHOW MUST GO ON

Now you know how to create and publish your own podcast. But that's just the beginning! You've got to keep going, creating more content. Like anything else, podcasting takes practice. In fact, I recommend producing several podcast shows just to get the feel of it before you actually publish anything for the world to hear. Once you get the hand of it, add the shows to your feed, and you're on your way to being an internet icon!

Good luck – I can't wait to hear your show!

MT

About The Author

August Trometer is the creator of iPodderX, a podcast receiver for Mac OS X, as well as the .Mac-centric website dotmac.info. He can be reached at BlueGus@mac.com.

BLOB XML PHP C++
REALbasic Director COM .NET
SQL Object-Relational Client Server Unicode
Joining Worlds of Information

PARADIGMA software

Valentina 2: Deploy True, Royalty Free Client-Server Solutions
www.paradigmasoft.com

THE TERMINAL: WHY?

LOVE IT OR LEAVE IT!

If you're a bit more of a CLI veteran, but are coming from a different platform, you may simply want to jump down to 'Apple-fying the CLI".

The Past

I work with a broad spectrum of people that reign over technology in some way: from low-level hardware and software hackers, to networking experts, high-level FileMaker developers and GUI-rangers. These people have all learned, or come close to mastering, the command line, and are better for it. I've grown up in a world of teletypes, Commodores, IBMs, Unix boxes, Apples, Netware servers, DOS and Windows environments. All of these machines started with a command line (and some ended there). In the timeline of computing, the GUI was an afterthought. Not for the Mac, of course. But go back to an Apple II, (if you've got one laying around!) and you'll find that when you boot up, you're presented with a ']' prompt and blinking cursor. Since this environment has been around so long in the Unix world, it is very well thought out and very mature. But it's certainly one of the reasons people not-in-the-know would panic when they'd turn on a computer. What do I do? It's just sitting there blinking at me. Will I break it if I type the wrong thing?

The Mac OS tried to end all of that command line fear and present a graphical interface at boot time that made people feel comfortable. They did a great job. But fast-forward to now, and Apple is singing a slightly different tune. However, I find many people who are dyed-in-the-wool Mac users simply pretend that Terminal.app doesn't exist.

The Present

Here we are, and there's a command line in a Macintosh operating system. They just couldn't keep it out of there. In all honesty, if it weren't in there, I'd be writing for a Linux magazine right now. As a techie, and someone who likes (and many times needs) to troubleshoot, there was no bigger breath of fresh air when I fired up Terminal.app under OS X (10.0 beta, on my Powerbook G3). I immediately typed 'ping 192.168.30.1' for the network I was on and saw the replies come back. Wow, Apple did it. Keep Word. Keep Safari. Heck, even keep Quake and Tron 2.0, I don't want a computer without access to the command line. But why?

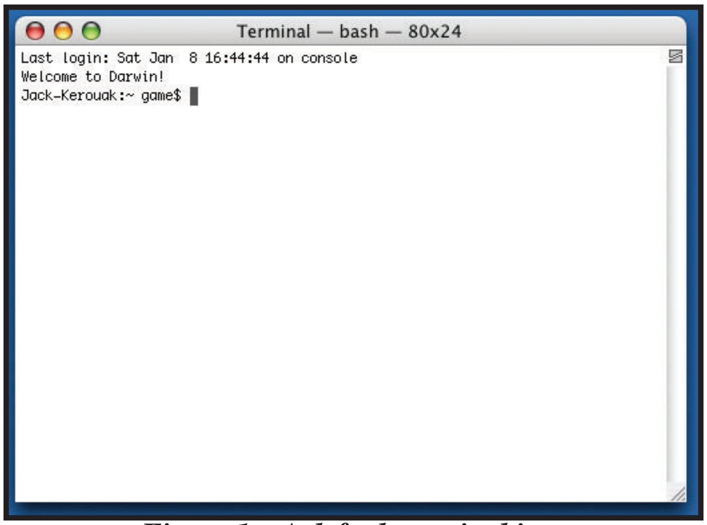
Terminally Acquainted

I mentioned the power that lies in the terminal, what is that all about? Why is it so much more powerful? Firstly, you can typically type more quickly than you can mouse around. From time to time, I see people launch Calculator and click on each number rather than use the number pad on their keyboard. Doing that just doubles the time required.

You've got a fancy Aqua GUI in front of you (errr…if you have the print magazine in front of you, look at your OS X box now), why would anyone use the command line? The command line!?! We're here at MacTech because we use a Mac – the computer that popularized the GUI!!! The computer that said, "CLI? Gag me with a spoon!" (well, it was California in the 80's). However, despite Apple initially eschewing a command line altogether, the CLI has survived. There's a lot of power there, and OS X lets you tap into it. Furthermore, anyone can fire up a GUI utility and make some changes. But if you want to impress your date, you have to learn some command line tricks.

Secondly, as people like to customize their GUI (I'll admit that when I work on other people's machines and I find the dock on the left it drives me a bit batty…) and GUIs change over time (look at the differences going from 10.0 to 10.3), the CLI is pretty much the CLI. Of course, it can be customized, but it's usually done in such a way that it doesn't change the way standard utilities run. Third, it gives you a consistent way to administrate a machine. Fourth, it gets you a little closer to the operations of the machine. Have you ever had the GUI lock up on you? I have. But everything else was still running and I was able to console in and reset the machine gracefully. Fourth, and most importantly, Apple lied to us! When OS X shipped, we were told that we'd never have to see a command prompt if we didn't want to. OK, perhaps not. But that stopped us from doing certain things with our machines. While the entire situation is getting better, there are things you can do in the terminal that there is simply no GUI equivalent for. With those notes, let's get familiar with Apple's Terminal.app, starting with the configuration that ships with OS X 10.3.

Launch Terminal.app from /Applications/Utilities/Terminal. Perhaps the fact that you find the app in 'Utilities' rather than 'Applications' is something that scares people right away, as if it's not something one should normally run. Figure 1 shows approximately what the default terminal looks like.

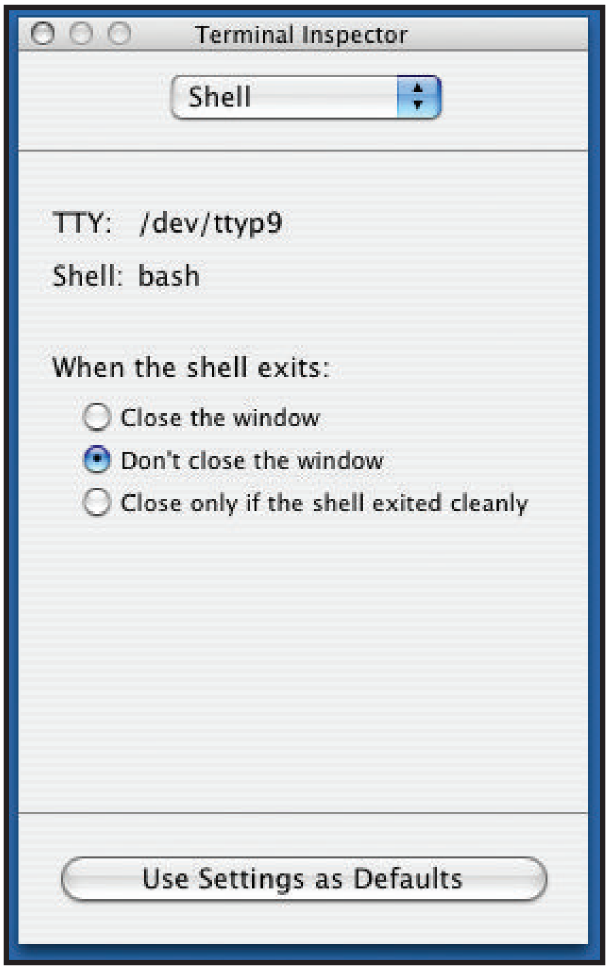


```
Last login: Sat Jan  8 16:44:44 on console
Welcome to Darwin!
Jack-Kerouak:~ game$
```

Figure 1 – A default terminal in OS X (Panther 10.3.7)

I say 'approximately' because you will have some differences. Of course, the time of your last login will be different. Unless you've already changed it, your "message of the day" will still read "Welcome to Darwin!" The next line is your prompt, and it is generated at run-time. 'Jack-Kerouak' is the name of my machine (because, if you must know, it's a laptop and I'm always "On the Road"), and you'll have the host name of your machine. The "~" shows my current path, and by default, we start out in our home directory (which is represented by the tilde). "game" is the short name of the account I'm logged in as (remember: Quake and Tron 2.0!), and you'll have your user name. Then, there it is, the cursor. Patiently waiting for you to type.

Black text on opaque white. Boring. Lets go check our window settings. Choosing the "Terminal->Window Settings…" menu gives us some ways to modify the look and behavior of the terminal. Figure 2 shows the first of several preferences that can be changed in the 'Terminal Inspector'.



TTY: /dev/ttyp9

Shell: bash

When the shell exits:

( ) Close the window
(●) Don't close the window
( ) Close only if the shell exited cleanly

Figure 2 – Terminal Window Settings

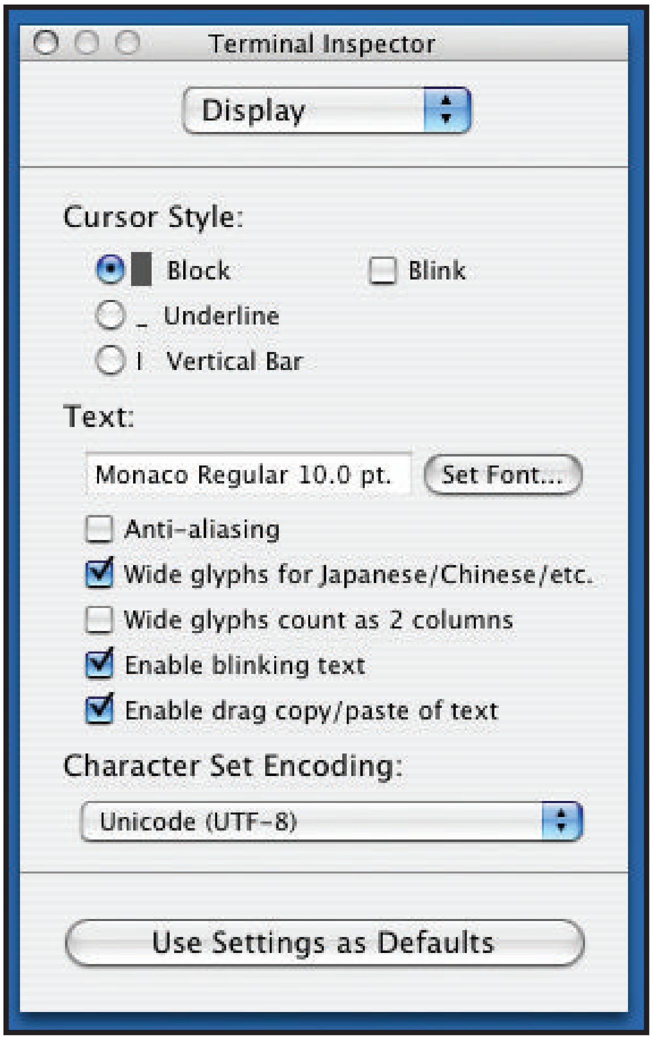
Of course, these are all preferences, and are unique to each individual. I'm going to share how I like my terminal to behave, but by all means, choose what makes you most comfortable.

The first set of preferences, "shell", gives us one option: choose what to do when the shell is done. I think I've only had one occasion to keep it at the default, so I immediately change this to "Close only if the shell exited cleanly."

The "process" preferences work perfectly at "prompt before closing window if there are processes other than:". I like being prompted as little as possible for anything. The "emulation" settings have good defaults, but may need to be tweaked for a particular case. The only thing I do here is check the "option click to position cursor" checkbox, despite actually using that function very little myself.

The "buffer" preferences only deserve one change: set the scrollback to 'unlimited'. If you ever start compiling things from the command line, like a custom Apache install, 10,000 lines can disappear pretty quickly.

The "display" preferences are a little more fun, as their effects can be seen instantly. See figure 3 to get a look at this one.



Cursor Style:

(●) Block [ ] Blink
( ) Underline
( ) Vertical Bar

Text:

Monaco Regular 10.0 pt.

[ ] Anti-aliasing
[✓] Wide glyphs for Japanese/Chinese/etc.
[ ] Wide glyphs count as 2 columns
[✓] Enable blinking text
[✓] Enable drag copy/paste of text

Character Set Encoding:

Unicode (UTF-8)

Figure 3 – Display settings

Call me old-school, but I want a block cursor that blinks. Depending on the display I'm using, I'll sometimes drop the point size down to 9. A quick tip for you: never turn on anti-aliasing. Not only does it look terrible, it slows Terminal.app down – yes, even more so than it starts out. This is one valid gripe that users of Terminal.app have. Its speed is nowhere near a real terminal, a terminal emulator on other systems, or even a Windows DOS box (or, for that matter 'DOSbox' under OS X. Man is that thing quick!). While things did get better in Panther, Terminal is still the laggard, comparatively. But, hey, it looks great.

Next up are the color prefs, which go hand in hand with the display prefs. Have fun with this one: there are no wrong answers here. Come up with a style that is easy on your eyes and makes you feel at home. Again, I go for the old-school combo of green on black, with a pinch of ultra-modern transparency. I have one terminal combo of light-blue on dark-blue with a rather large font. Yes, it looks like a Commodore 64….

Stepping down brings us to the 'Window' preferences. I tend to check off just about everything in the lower-half of the inspector window. Additionally, I like to make the terminal fairly large. Why have text wrap if it doesn't have to?

On the last page of options, the 'keyboard' prefs allow one to alter the escape codes that are sent to Terminal.app for each key. Unless you have a great need to change these (and you may), just leave these at their default settings.

Now, I know you've been eyeing that large "Use Settings as Defaults" button at the bottom of the Inspector window. Well, if you have everything set the way you like, click it! As soon as you click it….nothing happens! Well, OK, it does save your preferences, but there is absolutely no feedback that it done anything. For proof, quit Terminal.app and relaunch it. You should now have a terminal that defaults to your settings. Nice, eh?

Now What?

So, now we've made the terminal pretty. Great. Besides staring at a blinking cursor, now what? Let's start with the basics. Again, you'll see where you are in the filesystem based on your prompt, which at first should read '~'. We can start from the top to best illustrate how this works. The very top level of the filesystem is represented by '/', or, 'the root'. Type 'cd /' and press enter. This will 'c'hange 'd'irectory to "/". You're now at the top level of your disk tree, basically represented by "Computer" in the Finder. You should also notice that the terminal prompt changed from "~" to "/". Now, type "cd Users" (capitalization is important). You've moved into the familiar Users folder. Let's see what's in here. Type "ls –l". This produces a file 'l'i's'ting of the current directory. The '-l' following the command is a switch that modifies the behavior of the command. In this case, we want a 'l'ong list. Try an 'ls' without the '-l' switch and you'll immediately see the difference (and, hopefully, why I prefer the long list).

So far so good, right? Nothing broke. Just remember, although the terminal brings you down to a lower level, there's still a thin veneer between you and the OS. Not quite the movie screen the GUI covers everything up with, but still, a level of abstraction exists. For example, when you ask for a file listing by typing 'ls', sure, you had to do something manually. Directory information didn't just come flying onto your screen. But neither did you have to tell the disk drive which blocks to access. So, as always, unless you pour liquid onto your CPU, you're not going to break anything.

If you feel comfortable with these two basic exercises, the command line just may be for you! Naturally, this doesn't scratch the surface of what can be done via the CLI. Not even the surface of the smallest surface that exists on the surface of the CLI.

Want More?

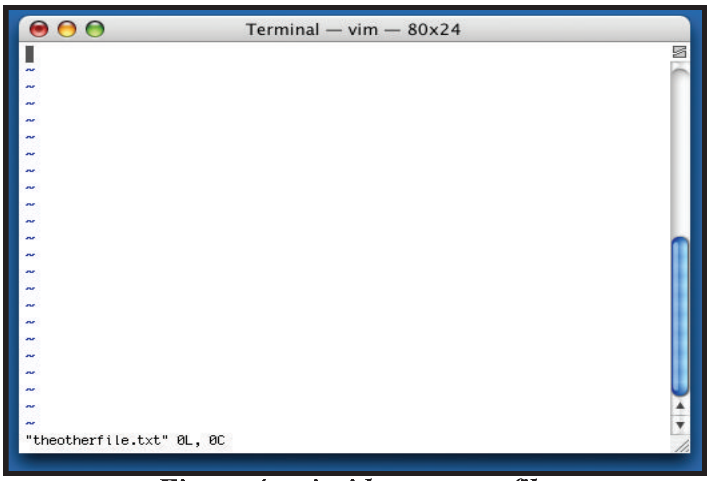
Listing files? I can do that in the Finder! Where's the power? If you're comfortable moving from directory to directory, we can look at some more powerful commands.

Continuing with file related commands is important, as Unix treats just about everything as a file. Your disk drive? A file. Even the terminal display can be treated as a file. We'll get into this deeper in future columns, but safe to say, file manipulation is important.

Back in the terminal, type 'cd'. Simply typed by itself, the change directory command will bring you back to your home directory. Now, type 'touch thefile.txt'. In short, the touch command will either create a zero-length file, or, if the name you specify already exists, will update the date stamp of that file to the current date and time. Get a directory listing and see if your file is there ('ls –l', remember?).

To copy that file, you use the 'cp' command. Type 'cp thefile.txt theotherfile.txt'. 5 points if you typed 'cp thef' and hit tab to complete. This will copy 'thefile.txt' to a file called 'theotherfile.txt'. We can alter these files as well as having copied them. There are some holy wars in Unix-land as to the best text editor in the world. I use vi. No apologies, that's just what I use. It will absolutely be the subject of a future column. If you know another text editor, feel free to use it here.

Invoke vi (the 'v'isual 'e'ditor) by typing 'vi theotherfile.txt' (did you use tab completion?). You'll be presented with a blank-ish looking screen with tildes running down the left-hand side. Figure 4 should mirror what you're seeing.



```
"theotherfile.txt" 0L, 0C
```

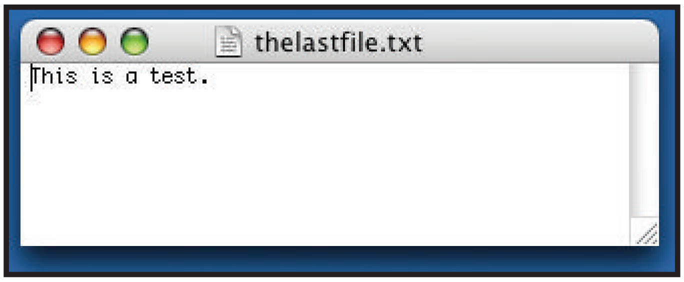
Figure 4 – vi with an empty file

The tildes represent a non-existent line, which, admittedly, can sometimes get confusing if you're editing a file with tildes. I'll just give the key presses with a short description, since I'll cover vi in a future column – case is important, by the way. Press 'i' for 'i'nsert – you're now free to roam about the cabin. You should see a bold 'INSERT' notification at the bottom of the edit window. Type whatever text you'd like. I just typed 'This is a test.' When you're done, press the escape key on your keyboard. Type ":" (colon), and you should see a ":" appear at the bottom of the window you're editing in. Follow that with 'wq' and press Enter. That tells vi to 'w'rite the file to disk and then 'q'uit. You'll be dropped back to your prompt.

I'd like to delete the original file that we created. This is done with the 'rm' command. Now, just like files that you throw in the Trash via the Finder, be careful what you follow the 'rm' command with. Unlike the trash, though, the files that you list will be deleted immediately. No trash, no undo. Gone. Tab completion can be great, or you can use it without thinking after an rm command and nuke the wrong file. Be careful out there! That said, type 'rm thefile.txt' and, after checking yourself, press Enter. You've just deleted 'thefile.txt'.

The 'mv' (move) command moves and renames files. Renaming, after all, is just moving a file within the same directory. Type 'mv theotherfile.txt thelastfile.txt' and press Enter. 'theotherfile.txt' just became 'thelastfile.txt'.

To bring this all home, we can open the file we created in the Finder. Switch to the finder, and open your home directory. 10 points if you've left a Finder window of your home directory open this whole time and watched all of these machinations take place. You should see a text file named 'thelastfile.txt' sitting there. If you double click it, it should simply launch TextEdit. Check out our handiwork in Figure 5.



```
This is a test.
```

Figure 5 – TextEdit displaying our file

While this was all a bit contrived and trivial, I'm sure you can imagine some automated routines that compile information, save it to a file, and then display it via TextEdit or any other program. In fact, let's try something a little more serious.

Hop back over to terminal. Fire up vi or your favorite editor. I'll give instructions for vi. Type 'cd' so you're sure you are in your home directory. Type 'vi showdi.sh'. This will be a bash script that will show us a report of disk information for our main disk and display it in TextEdit. Press 'i', and you'll again see the bold 'INSERT' along the bottom of your editor window. Type the following exactly:

```
#!/bin/bash
diskutil info /dev/disk0 > /tmp/disklist.txt
open /tmp/disklist.txt
```

Save this file by pressing escape, typing ':wq' and pressing Enter. Type 'chmod 700 showdi.sh' and press enter. This gives this script the ability to be executed (run) as a program (ok, this is a bit simplified, but without this command, this script is just a text file).

Before we run this, let me note that you'll need to be an admin for this to work. When you're ready, type "./showdi.sh" to run our script. That's dot, forward-slash, showdi.sh. Don't forget the tab-completion for this one! Press enter. In about two seconds, TextEdit will pop up with a short report about our disk 'disk0'. See figure 6 for what this looks like.

3,248 hours typing code

184 hours finding that one bug

142 hours of meetings

108 pizzas

14 cancelled weekend trips

11 all-nighters

1 call protects it all

HASP®

SOFTWARE DRM

The new **HASP** family of products is the next generation in protection ensuring the highest level of security for your software. It provides an easy to use set of tools to protect once and deliver through a variety of licensing options.

1 Develop
- For Windows, Mac OS, Linux

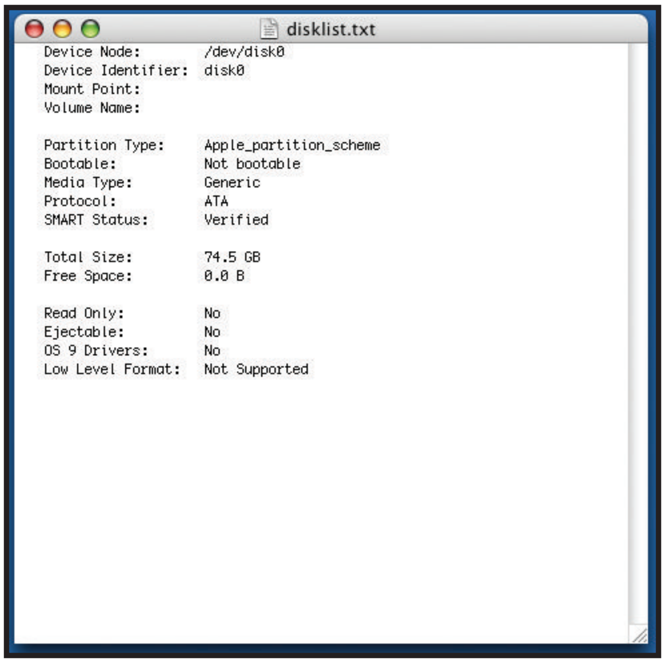
2 Protect Once
- Data encryption with industry standard hardware-based AES 128-bit
- Strong wrapper with code obfuscation prevents debugging and reverse engineering
- API code generator for easy implementation and customization

3 Distribute Many
- Single and multi-user license management
- Secure license updates for keys in the field
- Innovative licensing models defined separately from protection

Get the kit: Developer's Software, Guide and Demo Key at www.aladdin.com/hasp

Aladdin
SECURING THE GLOBAL VILLAGE

North America: 1-800-562-2543 **International:** +972-3-636-2222 **UK:** +44-1753-622266 **Germany:** +49-89-89-42-21-0 **Benelux:** +31-30-688-0800 **France:** +33-1-41-37-70-30 **Spain:** +34-91-375-99-00 **Israel:** +972-3-636-2222 **Asia Pacific:** +852-21-66-8605 **Japan:** +81-426-60-7191



```
Device Node:         /dev/disk0
Device Identifier:   disk0
Mount Point:
Volume Name:

Partition Type:      Apple_partition_scheme
Bootable:            Not bootable
Media Type:          Generic
Protocol:            ATA
SMART Status:        Verified

Total Size:          74.5 GB
Free Space:          0.0 B

Read Only:           No
Ejectable:           No
OS 9 Drivers:        No
Low Level Format:    Not Supported
```

Figure 6 – Our showdi.sh script in action.

Again, the details of this script and all the commands we typed will be covered in future articles.

Apple-Fying The CLI

If you're a more seasoned user, you may have skipped some of the earlier bits of this article. You already know your way around. You know what a hard link is and you know how to use it. You like to fire up Terminal.app, dive in and never look back. Some things that may escape you if you're coming from another environment:

If you're a big xterm person, there are some notable differences here. Mainly:

- Terminal.app doesn't honor switches (like "-bg") that allow you to customize the Terminal at app launch. You have to use Terminal Inspector as described earlier.
- $TERM defaults to "xterm-color", which is great on your own system, but can throw remote systems not ready for it.
- You can't launch a new terminal from the command line! Goofy, eh? You just have to slap Apple-N.

However, despair not. There are some really nice Terminal attributes. Such as:

- You can set your window title on the fly (though escape sequences and an 'echo').
- Split-screen bar (see figure 4)
- Tab completion. I couldn't survive without tab completion.
- Integration with the Services menu.

We'll step through these bits here.

If you're really into customization and want to set your window title from the command line, or have a script that uses this functionality, you can! Try this:
echo -n -e "\033]0;Title\007"

The "\033" is the 'escape' key, needed to start an ANSI escape sequence. Follow this code with the title you want, and close it out with a "\007".

You can split your terminal horizontally by clicking the 'broken square' icon in the upper right-hand corner of the terminal window. This will display a horizontal bar that can be adjusted to size the windows as needed. Figure 4 shows a split screen with a file listing in the upper split, and 'top' running in the lower pane. While this functionality is useful, I use it very little. The reason for that will be part of a future article.

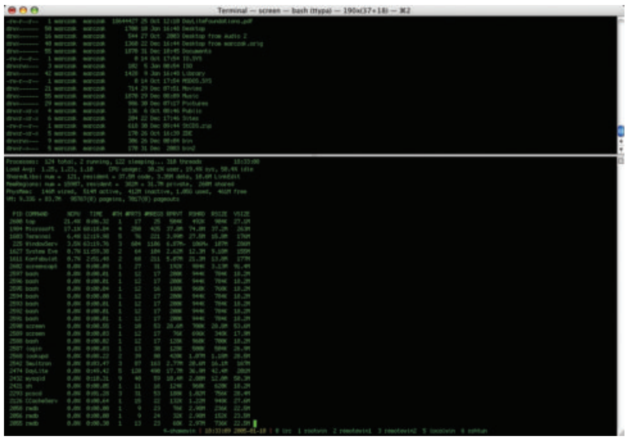


Figure 7 – Terminal with split-screen activated

Tab completion!!! All Unix veterans know some kind of completion. And when you start using it, you'll never give it up. For you hard-core Unix people: OS X has standard tab-completion, 'nuff said. If you don't know what this is, here's an illustration: once again, type 'cd /' to get to the root. Now, type 'cd Li' and then press the 'tab' key. Suddenly, the line you're working on fills itself out (to become 'cd Library/'). Now, type 'W' and a 'tab'…boom! You now have 'cd Library/WebServer'. This cuts down on the keystrokes you need to type by a huge factor. Sometimes, your hit 'tab' and you simply hear a beep. That's either because nothing matches, or more than one thing matches. As an example, if you still have Classic loaded on your machine, and you type 'cd /Sy' and press tab, you get a partial completion (to 'System') and a beep. If you press 'tab' again, the shell will show you the matches. In this particular case, you can either accept the match of 'System' (because it's valid), or type '\ " (backslash-space) and press 'tab' again to have the shell complete the next match. The more you use it, the more you'll get the hang of it. Just don't practice on Windows XP, which now supports tab completion, but has a really poor substitute of it.

OS X has a great feature in the Services Menu – which someone else can cover much better than I can. Terminal.app has nice integration with this menu. Highlight some text, and then go check Terminal->Services. There's some nice functionality there, such as: Send to mail, create new sticky note, create new window in TextEdit, and more.

One last note for those so inclined: You should also take a look at the actual Terminal preferences, accessed through the 'Terminal->Preferences' window. This will allow you to define several aspects about your terminal that can help out in situations where you're trying to emulate a different terminal. Be aware that changing the shell in the Terminal preferences screen will only change it for shells launched through Terminal.app. Alternate terminals (see below) and ttys from remote sessions, such as telnet or ssh, use the shell defined in your user profile. The good old 'chsh' works for this purpose, or, if you want to get all OS X about it, change the shell key in your NetInfo record.

The Future

Well, naturally, I can't predict the future. But I can tell you that a text, command line interface will be with us for some time to come. There are new applications showing up all the time that are CLI only. You can find MP3 players, Gnutella clients, games, web browsers, e-mail programs and more, that are all CLI driven. Although the Mac certainly needs it less than other platforms, which may still be text-driven by nature, learning the CLI is of great benefit. It helps you troubleshoot a Mac with a boot time problem, and it can help you automate your machine in better circumstances.

I'd be remiss if I didn't mention that Apple's Terminal.app isn't the only terminal for OS X! There are two more that I'm aware of. GLTerm takes the speed issue head on. All display is done through OpenGL. It also supports X .bdf fonts. There may be cases where Terminal.app doesn't handle some graphics issue correctly. Chances are, GLTerm will handle those cases just fine. Find it at http://www.pollet.net/GLterm

The second terminal alternative is iTerm. iTerm shoots for features. If you spend time in KDE or Gnome, check out iTerm. It has support for bookmarks (saving session settings), tabs, an anti-idle function and more. You can find it at http://iterm.sourceforge.net.

The End

This is the end…of this column only (whew!) Obviously, I'm a huge proponent of the CLI. Now, I'm not so stubborn that I use the Terminal for everything! After all, I am a Mac user!

P S

Anyone who read my 'Unix in OS X' article that appeared in the December MacTech should note that I did find the solution to making an OS X 10.3.4 through 10.3.7 machine accept remote syslog connections. In your /etc/rc file, you need to alter the syslog invocation to read:

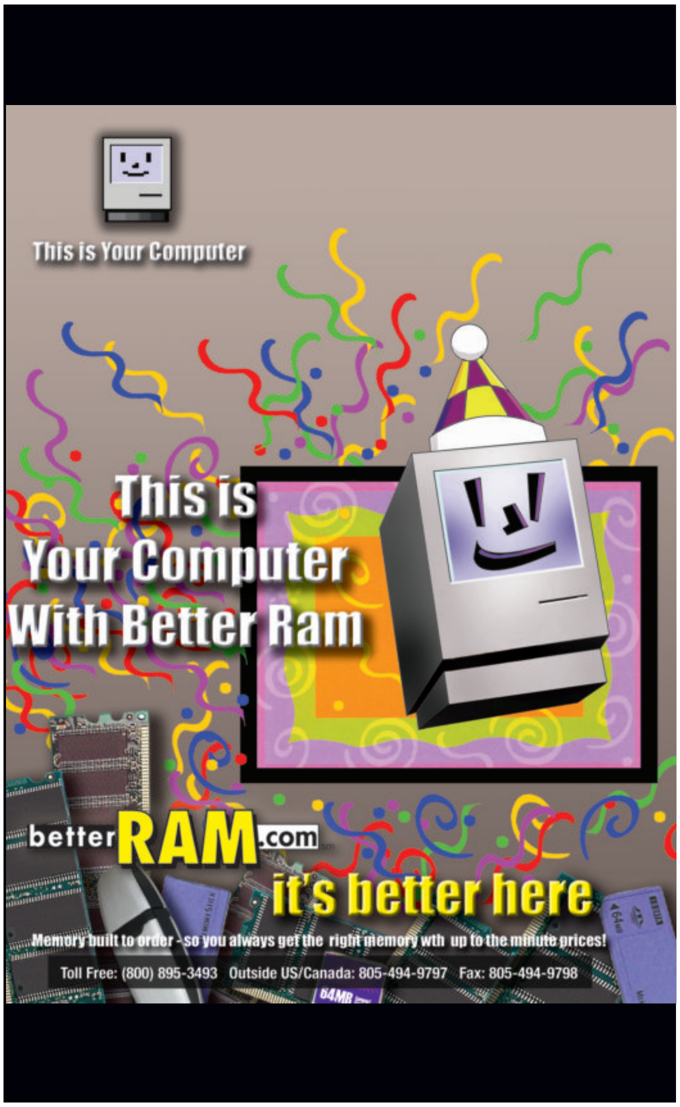
```
/usr/sbin/syslogd -a 192.168.1.100/24:\* -m 0
```

where the IP address and mask (in CIDR notation) represent the interface to listen on.

Unfortunately, this incantation has changed a few times and the syslog binary is out of sync with the man page. Stay tuned for any changes to remote logging!"

About The Author

When he's not helping the clients of Radiotope, you'll find Ed Marczak on the grid, fighting for the users.


This is Your Computer
This is Your Computer With Better Ram
betterRAM.com
it's better here
Memory built to order - so you always get the right memory with up to the minute prices!
Toll Free: (800) 895-3493 Outside US/Canada: 805-494-9797 Fax: 805-494-9798

Advertiser/Product Index