Table of Contents

AsciiDoc is a text document format for writing notes, documentation, articles, books, ebooks, slideshows, web pages, man pages and blogs. AsciiDoc files can be translated to many formats including HTML, PDF, EPUB, man page.

Setting up your editing environment

This doc describes how to set up an environment on a Mac to write for Maker Press. (Or, I should say, it's my running notes for it.) Depending on your system, you might have a bit of work to do to get things up and going. So, take this document with a grain of salt.

We'll need to do a few things to get going:

- Install MacPorts to make it easy to get new packages
- Install git and asciidoc, which is a package for formatting text
- Install TextMate, as well as a few bundles

The following sections explain this in more detail.

Preliminaries

You will need the XCode development tools and X11 (which comes on some Macs). Next, you'll need MacPorts, which makes it much easier to install everything you'll need. You can find it at the MacPorts website, which has all the instructions you need to install it for your system.

```
One annoyance I had on MacPorts is that it lists the OS versions by name, rather than number, which I found annoying.  You can find

* Version 10.0: "Cheetah"
* Version 10.1: "Puma"
* Version 10.2: "Jaguar"
* Version 10.3: "Panther"
* Version 10.4: "Tiger"
* Version 10.5: "Leopard"
* Version 10.6: "Snow Leopard"
* Version 10.7: "Lion"
```

Then make sure your PATH is correct in ~/.profile. So here are the two lines you'll want (updating for your username):

```
PATH=$PATH:/opt/local/bin:/opt/local/sbin:/Applications:/Applications/Utilities:/usr/local/bin:/Users/adam/bin
export PATH
```

Git

If you don't already have git, you can use MacPorts to install it. Just drop into the terminal and type:

```
$ sudo port install git
```

It should fire up and install with no problems.

asciidoc, a2x (8.6.4)

Asciidoc refers to two different things: a wiki-like **markup language** you can write in, and the various **tools** that convert that markup into various other formats. This section describes how to set up the tools. You can probably already see the punchline coming, but here's how you set up the asciidoc tools on your system:

```
$sudo port install asciidoc
```

The downside of this is that this step takes a **very** long time — maybe an hour or so. So, be prepared to run it and then go

out and get some coffee. Or, two coffees.

Textmate

You can edit your documents in whatever editor you like — vi, emacs, or whatever. I've been using TextMate, an editor for the Mac that is popular in the developer community. It costs about $60 U.S., but it's got some addictive features that make it worth the price. Plus, you can try it our for free for 30 days.

On of the coolest features of TextMate is that it offers a *bundles* — collections of macros, commands, snippets, drag commands, templates, preferences, and language grammars — that make development much quicker. There are two main bundles I've been using:

Install the asciidoc bundle

The AsciiDoc bundle makes it much easier to work with AsciiDoc in textmate by offering things like automatic previews, source highlighting, and so forth. Here's what you do:

```
mkdir -p /Library/Application\ Support/TextMate/Bundles
cd ~/"Library/Application Support/TextMate/Bundles/"
git clone git://github.com/zuckschwerdt/asciidoc.tmbundle.git "AsciiDoc.tmbundle"
osascript -e 'tell app "TextMate" to reload bundles'
```

Install the git bundle

The git bundle allows you to save stuff in git right from within TextMate. Not sure what this buys me yet, but here are the commands:

```
mkdir -p /Library/Application\ Support/TextMate/Bundles
cd !$
git clone http://git.gitorious.org/git-tmbundle/mainline.git Git.tmbundle
osascript -e 'tell app "TextMate" to reload bundles'
```

You'll need to add a path variable to tell TextMate where git is installed. First, locate git:

```
$which git
/usr/local/git/bin/git
```

Then go to "Preferences → Advanced" and click the "Shell Variables" tab. Then add a variable named "TM_GIT". You can find the path to git using this command sequence:

Writing in AsciiDoc

As described earlier, AsciiDoc is a text document format for writing notes, documentation, articles, books, ebooks, slideshows, web pages, man pages and blogs. The main advantage of AsciiDoc is that it plays well with DocBook, O'Reilly's native content format, but is something that you can actually write in.

This Asciidoc cheat sheet covers most of what you'll need to know. As you'll see, AsciiDoc is most similar to wiki markup — if you can write a wikipedia article, then you're pretty much 99% of the way there.

Note

Formatting O'Reilly books in AsccDoc

You can get the official guide for writing a full O'Reilly book in AsciiDoc from our Subversion repository:

```
svn checkout --username "odewahn@oreilly.com" "https://prod.oreilly.com/internal/books/sandbox_odewahn_RT79726/current/tool
```

I've also placed relevant files into the "samples" directory in this repository.

This document (still in R&D, BTW!) tells you how to do pretty much anything you'd like. I anticipate that Maer Press will only use a subset but this guide is there if you need it.

Editing files

The best way to get started with AsciiDoc and Maker Press is to pull down the repository for this document (how's that for recursion!) and look out how the files are structured. Here's how you can get the repo:

```
Need to have a link to the git repository for this doc
```

Once you have the files, you can open them in TextMate using the command:

```
mate .
```

This will open the editor and display the *project drawer*, which is a navigation tree that you can use to move around between files. Use the project drawer to open the file called *sec_environments.asc*, as shown in [textmate-fig].

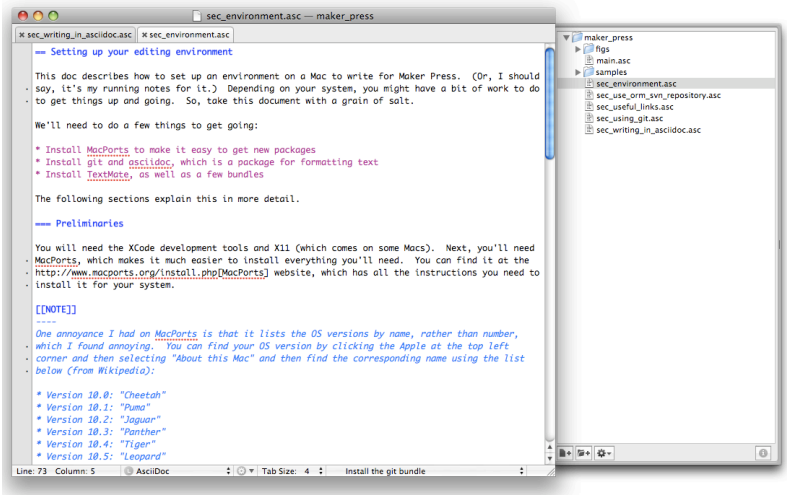

Figure 1. Use TextMate's bundle editor to add in the path

If you've done any wiki markup, this should look pretty familiar. Also, note how the various AsciiDoc elements are all nicely color coded — this is thanks to the AsciiDoc bundle we installed earlier.

Muck around in there for a while and make a few changes (remember — this is all in your local git repository, so you can't hurt anything). When you're ready, you can use the "Bundles → AsciiDoc → Preview" to see how your document will look when it is rendered in HTML, as shown in [asciidoc-preview].

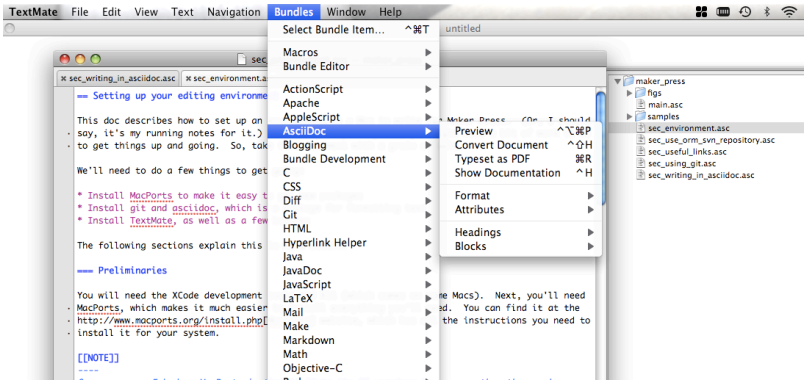

Figure 2. Generating a preview

The preview will convert the AsciiDoc into HTML as best it can. If preview identifies any errors, it will flag them as a red hyperlinks that will bring you to the offending line. For example, [preview-error] shows how preview has found a broken link to an include file.

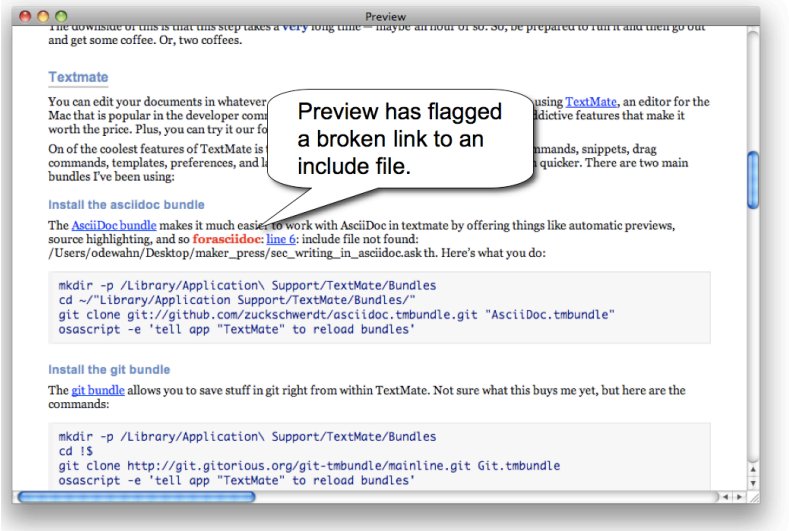

Figure 3. A broken link to an include file shows up as a red hyperlink

Most times, though, you'll have to scan the document and make sure it looks right, just like you would a wiki page. The

advantage of using Preview to clean up your errors is that it will save you time when you try to render the document in the O'Reilly toolchain.

Caution

If you have trouble with Preview, try this first

On my system, there was a weird problem when I tried to run the bundle commands — it couldn't locate the asciidoc command. There's it. First, I located where MacPorts installed asciidoc, which you can do by dropping into a terminal and typing this:

```
$ which asciidoc

/opt/local/bin/asciidoc
```

Then go into the bundle editor (Bundles → Bundle Editor → Show Bundle Editor) and select the asciidoc bundle. You can then go into th asciidoc appears. For example, here's the revised version of the "Preview" command:

```
if [[ -z $TM_FILENAME ]]
  then doc="-"
  else doc="$TM_FILEPATH"
fi
/opt/local/bin/asciidoc --attribute=imagesdir="file://$(pwd)" --out-file - "$doc" 2>&1 | perl -pe 's%(\w+): .*?: line ([
```

It should look something like [bundle-editor-fig] in Textmate.

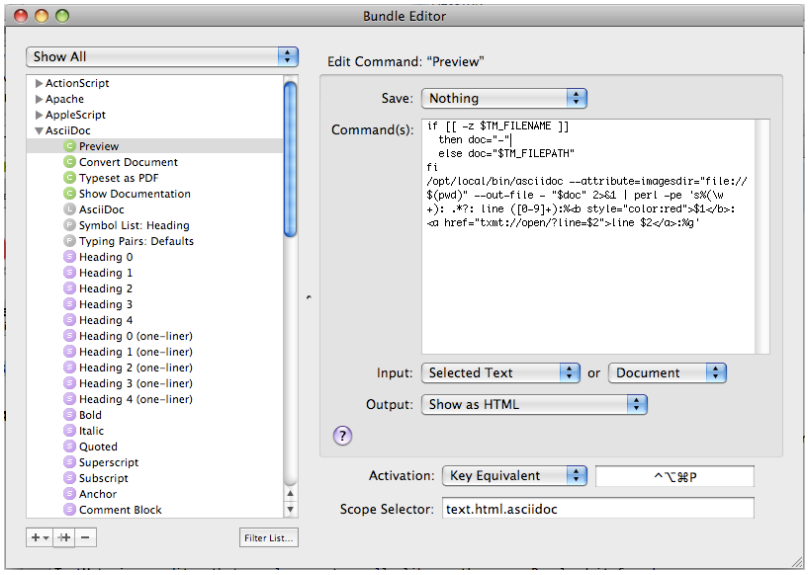

Figure 4. Use TextMate's bundle editor to add in the path

Generating a table of contents (TOC)

You should review your TOC often to make sure that the document is structured correctly. It's really easy to generate this automatically — all you have to do is use the bundle editor to add the option "-a toc" to the asciidoc command.

Open up the bundle editor (Bundles → Bundle Editor), select the AsciiDoc bundle, and then click the "Preview" command. Then, add the put "-a toc" after the AsciiDoc command, as shown in [add-toc]

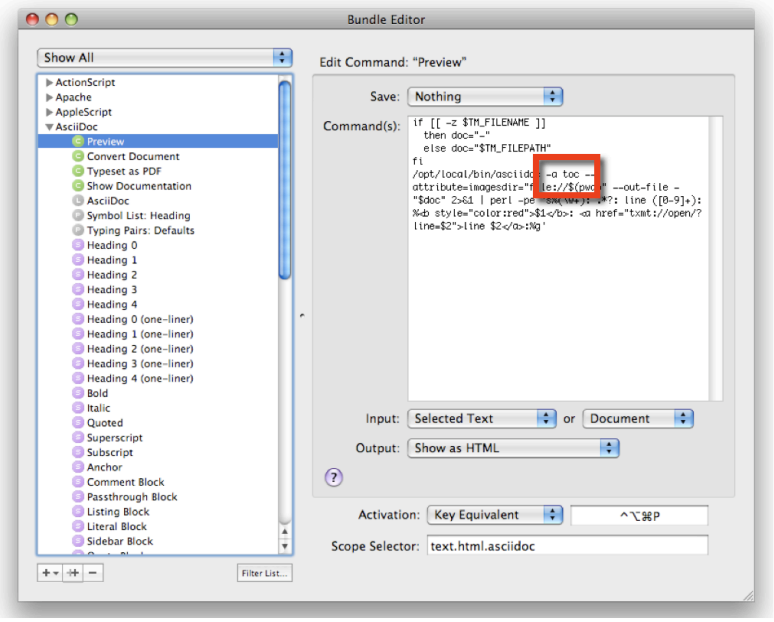

Figure 5. Automatically adding a TOC to Preview

Save your document as an html file

AsciiDoc has options to create stand-alone documents containing embedded images, stylesheets and scripts. The following AsciiDoc command creates a single file containing embedded images, CSS stylesheets, and JavaScript (for table of contents and footnotes):

```
$ asciidoc -a data-uri -a icons -a toc -a max-width=55em article.txt
```

Using the ORM SVN repo

O'Reilly's production department uses Subversion to manage all files.

Integrating with the ORM production workflow

To actually "publish" your document through O'Reilly, you'll need to get your files out of git and into the ORM production workflow, which is based on Subversion. (You'll also need an account for this, so talk to your editor about the process.) You can find all the documentation about this process in the O'Reilly Production Guidelines.

Checking out the files looks something like this:

```
svn checkout --username "odewahn@oreilly.com" "https://prod.oreilly.com/internal/books/sandbox_odewahn_RT79726/current/"
```

Caution

Using "git svn"

In theory, we should be able have git act like an svn client using "git svn" by doing something like this (see Chapter 8 from *Pro Git*):

```
git svn clone --username "odewahn@oreilly.com" "https://prod.oreilly.com/internal/books/sandbox_odewahn_RT79726/current/
```

I don't think this is going to work for us. The main sticking point seems to be that git wants to make a complete history of all the change repository, so it goes through several hundred thousand (literally!) different revisions.

This takes a long time to complete — it has to go through almost 300K revisions.

I wonder if there is a way to speed it up? Maybe we could create a separate repository for just realtime projects, because this is really sl done in the git repo, and the subversion repo is only for publishing content, so it shouldn't be very important.

Generating a PDF

You can generate a fresh PDF of your book every time you commit changes to the SVN repository. To do so, run the following command (substituting your own message before the semicolon):

```
$ svn commit -m'Made some really important changes to Chapter 3; orm:commitpdf'
```

To get the PDF, just run svn up on your working copy about 5–10 minutes after committing your files. The PDF will be downloaded as pdf/book.xml.pdf in your working copy. If there are any problems in generating the PDF, you'll instead get a .buildlog file in the pdf/ directory that lists the errors

Useful links

These are various linke that I've found that help explain AsciiDoc, O'Reilly processes, and other stuff:

- Matt Neuburg article on how to write a book using asciidoc. Nice article that explains how to do lots of good stuff.
- Asciidoc cheat sheet. Pretty much what it says — this shows you the markup in a no-nonsense kind of way.
- O'Reilly Production Guidelines. Describes how to use the main O'Reilly subversion repository.
- Looks like a pretty good intor to the basic features of git

Appendix on git

These are just some random notes about git — I need to add more in this section later as we start to develop some ideas about how to best use it. Most of this is cribbed from Scott Chacon's git in one hour webcast.

Creating a Repository

There are two ways to create a repository:

- clone it from an exitsing repo
- run "git commit -m *initial commit*"
- use "git status" to get a status of working changes
- use "git log" to get a log file of all commits
- "git show" shows what the last commit was

Adding Files

Next, you need to add files using "git add <filename>"

More commit options

Then you use "git commit -m *some message*" to commit the changes You can also used "git add <filename>" to stage your commits, which means that you can commit in different groups. this is helpful if you want to create a clear audit trail of what you were doing.

- To just commit everything, you can use "git commit -a"
- To commit stuff in steps, use "git add <filename>"

Branches

git encourages you to do lots of little stuff inside branches, not just big stuff. For example, you might create a new branch to resolve a ticket, fix it in the branch, and then merge it back in later. This is called a "topic branch".

The command to see a branch is called "git branch".

You can make a new branch using "git branch test" to create a new branch. To work in the branch, you do "git checkout test".

You can make any changes you want in the directory, and all changes happen in just that branch. So, you can add files, make new changes, or whatever, and they'll happen in only that one place. Changes are saved at each commit.

If you want to move back to another branch, all you have to do it check it back out and the files will look just as they did at the last commit point.

You can view all the activity in the branches using "gitk". This will pop up a ltille browser to go through all the changes on the branch you're on. You can use "gitk --all" to see the changes across all branches.

Merging

I need to review this section, but merging basically collapses two branches into a single branch. You can then remove the topic branches using "git branch -d branch name" (double check this syntax)

You do something to add a remote server

You can then use "git fetch" to pull off some remote copy of the repository, but moves it into your repo as a separate branch. This way you can inspect what someone else is doing, but not mess up your own work, which is really pretty cool. You then merge that brach back in as you see fit.

Comparing versions of files

Use "git diff" to see the differences between the current version of the repo and the new verion. You can use "git diff | mate" to see the diff visualized in textmate, which is pretty cool.

Last updated 2011-03-12 19:18:10 EST