Turn your text documents into HTML with Ikiwiki

THE CHANGER

Get your own wiki up and running with Ikiwiki.

BY DAVID A. HARDING

Ikiwiki isn't your typical wiki, but a program that turns documents into HTML (Figure 1). It doesn't need a web server, CGI scripts, revision history, or anything else to reformat documents, so it doesn't depend on any of them. Instead, separate plugins provide each of these features and more.

If you decide to keep track of page revisions, as most wikis do, you won't store those revisions in a database. Instead, Ikiwiki stores revisions in revision control systems such as Subversion, Git, or Mercurial. This lets you edit your wiki with the use of any text editor, command-line program, or application. You can even edit your wiki offline. Of course, you can't publish your changes until you're back online.

Many people use Ikiwiki to blog, and it provides all of the usual tools for blogging: RSS and Atom news feeds, comments, and tags. Ikiwiki also provides a novel feature not seen in most blog software: It lets readers inspect the revision history of a blog post.

Although you can easily create, edit, and browse your wiki locally without a web server, you need one to put your wiki on the web. The following instructions don't require a web server, but for those of you who have one, I'll also describe how to configure Ikiwiki for the web.

Installing and Configuring

To install the ikiwiki package, use your favorite package manager. If you don't have one, try Synaptic in the *System | Administration* menu. Don't quit your package manager just yet – you still need to install a revision control system. On Jaunty (but not other versions of Ubuntu), you also need to fix bug #375159 [1] before continuing by running the following command:

```
sudo sed -i ⮐
  '28 s^/usr/local^/usr^' ⮐
  /usr/share/perl5/IkiWiki.pm
```

Ikiwiki tracks changes with a revision control system. Here, I pair Ikiwiki with the Git revision control system, but you can use other systems instead. Now use your favorite package manager to install the git-core package. If you plan to let users view page history from the web server on this computer, also install the gitweb package. After installing Git, open a terminal emulator and run the following command:

```
    ikiwiki --setup ⮐
  /etc/ikiwiki/auto.setup
```

Ikiwiki asks you to name your wiki, name your account, and enter a password for your account. Additionally, it asks for the name of the revision control system you want to use – answer *git*. As Ikiwiki sets up your wiki, it lists the information you'll need later, which I suggest you save.

In the list, locate the three settings labeled *repository, srcdir*, and *destdir*. These point to your wiki's Git repository, source directory, and destination directory. The repository stores your wiki's revision history. If you want gitweb to display this history for everyone to see, run the following commands to move the repository and tell the source directory about its new location.

```
sudo mv wiki.git /var/cache/git/
cd wiki
git config remote.origin.url ↩
  /var/cache/git/wiki.git
```

The destination directory stores your wiki's HTML files. Ikiwiki creates it in the *public_html* subdirectory of your home directory, which some web servers automatically host; see the list Ikiwiki printed for the location of your wiki on the web.

While you try Ikiwiki, you can leave your destination directory alone. Later, you might want to manage pages outside of your *public_html* directory. If so, move your destination directory to wherever you keep your website files – for example, */var/www* or */srv/www*.

If you move any of these three directories, you need to update your Ikiwiki setup file. Ikiwiki listed the name of this file earlier with the rest of your settings; its name ends in *.setup*. Now open the file and update the *srcdir*, *destdir*, and *git_wrapper* settings. Also, you might need to change the *url*, *cgiurl*, and *cgi_wrapper* settings to reflect the new location of Ikiwiki on your website. Be sure to re-run the Ikiwiki *setup* command

```
ikiwiki --setup wiki.setup
```

whenever you change an Ikiwiki setting.

Editing Your Wiki

Unlike other wikis, you can create, edit, or delete Ikiwiki pages from anywhere, and you don't even need a web server. Activating this feature requires only a little extra setup. To start, make a copy (clone) of your repository:

```
git clone ↩
wiki.git mywiki
```

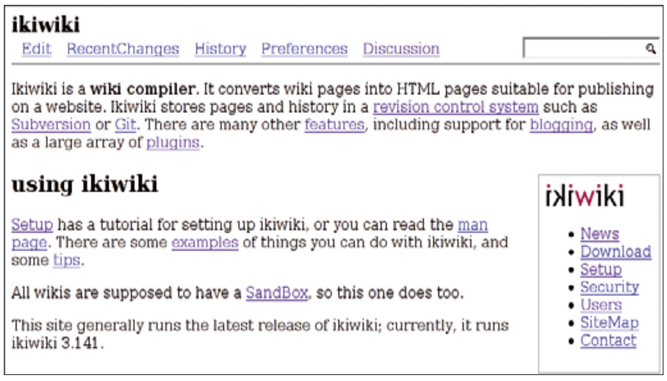
The *git* command creates the directory *mywiki* and fills it with all the files in your wiki. Although you can begin editing files in this directory immediately, you don't have any yet, so I suggest you start by putting something in the file *mywiki/index.mdwn*. Later, Ikiwiki will turn this file into the *index.html* file that everyone sees when they first visit your wiki. Although you can open *index.mdwn* in any text editor – vi, Emacs, Gedit, Kate, or even OpenOffice.org – you must save it as text with a *.mdwn* file extension. For help formatting this file, see the formatting section below. Now go to your terminal emulator, change directory (*cd*) into the *mywiki* directory, and run:

```
git add index.mdwn
git commit index.mdwn
git push
```

The first command adds *index.mdwn* to Git's revision log. The second command asks you to describe the changes you made; you could enter, for example, "created new index file." The third command sends both *index.mdwn* and the log to the Git repository. The Git repository saves *index.mdwn* and the incoming updated revision log. Then it runs a script that Ikiwiki installed: the post-commit hook, which copies *index.mdwn*, converts the copy from Markdown to HTML, and places the HTML in the destination directory. If anything goes wrong, Git prints an error on your screen.

Repeat the second and third commands every time you edit a file; repeat all three commands every time you add a new file.

To view your new wiki page, browse to the location of Ikiwiki on your web server or to the destination directory. For example, substitute your username and wiki name in *http://example.com/~<username>/<wiki_name>/* or *file:///home/<username>/public_html/<wiki_name>/index.html*. To create additional copies of your wiki, use the *git clone* command as many times as you want. Git has special options for the *clone* command that let you create remote offline repositories.


ikiwiki

Edit RecentChanges History Preferences Discussion

Ikiwiki is a **wiki compiler**. It converts wiki pages into HTML pages suitable for publishing on a website. Ikiwiki stores pages and history in a revision control system such as Subversion or Git. There are many other features, including support for blogging, as well as a large array of plugins.

using ikiwiki

Setup has a tutorial for setting up ikiwiki, or you can read the man page. There are some examples of things you can do with ikiwiki, and some tips.

All wikis are supposed to have a SandBox, so this one does too.

This site generally runs the latest release of ikiwiki; currently, it runs ikiwiki 3.141.

Figure 1: Ikiwiki running its own homepage.

If you're interested, read the manual page with the *man git-clone* command. If you make more than one copy or let people edit pages over the web, you should run the *git pull* command twice in the *mywiki* directory: once before you start editing a page and once before you commit it.

Formatting Wiki Pages

Many wikis use their own frustratingly peculiar syntax. For different wikis, it seems, you must learn a whole new markup language. Ikiwiki partially solves this problem by letting you choose the markup language you want to use. Ikiwiki uses the Markdown format by default, but you can also use ReStructured Text (RST), plain HTML, and

Markdown Syntax Summary

```
## Two hashes: an h2 heading

An empty line ends paragraphs.

Greater than symbol: blockquote.

    Four spaces: <code><pre>

`inline code` uses backtics

*italics* and _also italics_

**bold** and __also bold__

[A link!](http://example.com)
```

a Wikipedia-like syntax. New markup languages come as plugins, so you can add more by enabling plugins. Ikiwiki looks at file extensions to figure out which markup language each article uses. For example, the extensions *.mdwn*, *.rst*, and *.html* mean, respectively, the Markdown, restructured text, and HTML formats.

Markdown lets you format pages as typical text-only email messages. For a summary, see the "Markdown Syntax Summary" box (Figure 2), or get a full description of the language from the Markdown homepage [2].

Special Directive

With the use of special commands called directives, you can add special features to specific pages. Each directive comes from a single plugin, but plugins can work together to give a directive new features. For example, the tag plugin lets you add tags to a page with the *tag* directive. All directives share a common format:

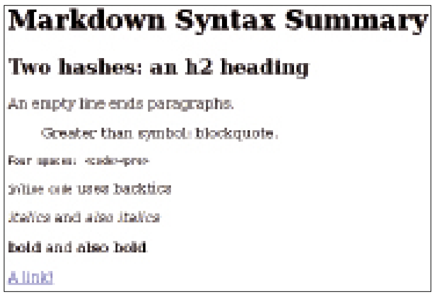
```
[[!tag article linux ubuntu]]
[[!pagecount]]
[[!inline pages="blog/*"]]
```

My favorite directive, *inline*, displays other pages in the wiki on the current page and combines with other plugins to build RSS and Atom news feeds of the included pages.

For example, I use the inline example in the code above to render my own blog. You can easily turn any set of pages into a blog or news page, or you can do something novel. For example, Ikiwiki uses the *inline* directive to track open and closed bug reports.

Plugins

Ikiwiki includes two sets of plugins and numerous ungrouped plugins. The first set enables the features most people expect from even the most basic of wikis by default, such as Markdown formatting and a CGI script that lets users edit pages on the web. Ikiwiki's author named the second set of plugins "goodstuff" and describes them as, "not too intrusive, work well with little configuration, and nice to have on any capable wiki." Ikiwiki disables these plugins by default, but a single change to your Ikiwiki setup file enables them. Ikiwiki also includes dozens of other plugins. For the complete list of over 100 included and third-party plugins, take a look at Ikiwiki's plugin page [3].



Markdown Syntax Summary

Two hashes: an h2 heading

An empty line ends paragraphs.

Greater than symbol: blockquote.

Four spaces: <code><pre>

Inline code uses backticks

Italics and also italics

bold and also bold

A link!

Figure 2: The Markdown syntax summary converted into HTML by Ikiwiki.

To enable plugins that came with Ikiwiki, add their names to the *add_plugins* variable in Ikiwiki's setup file. To disable plugins that Ikiwiki enabled, add their names to the *disable_plugins* variable. For example, to enable the search plugin and the whole goodstuff plugin package but disable password logins, use:

```
# To add plugins, list them here
add_plugins
  => [qw{search goodstuff}],

# To disable plugins, list them
disable_plugins
  => [qw{passwordauth}],
```

Now use the links on the Ikiwiki plugin page to download the third-party plugins you want. Just place the files in the *~/.ikiwiki/* directory and add their names to the *add_plugins* variable as described above. Get the name of a plugin for the *add_plugins* variable by subtracting the *.pm* file extension from its file name.

Some plugins require configuration and some plugins support extra options. In both cases, you will set variables in the Ikiwiki setup file. For details about configuring specific plugins, read the plugins page [3] on Ikiwiki's website – every plugin has a page. Ikiwiki's setup file describes most of the plugin variables that come with Ikiwiki.

Change Wiki's Appearance

Ikiwiki has a very plain default theme – you might even call it barren. Ikiwiki's author might prefer it this way, but he also made the layout easy to modify. Almost all of Ikiwiki's HTML elements have an ID or class, so you can lay out your wiki with CSS. Instead of modifying the main Ikiwiki CSS file, Ikiwiki suggests you put your layout into the *local.css* file. First create this file in the *mywiki* directory, add some CSS, save the file, then run the following three commands:

```
git add local.css
git commit local.css
git push
```

Ikiwiki automatically points to this file on every page of the wiki.

Unfortunately, you can only do so much with CSS. For example, you can't use it to add a new element to a page. However, you can edit Ikiwiki's template files in the */usr/share/ikiwiki/templates* directory.

Ikiwiki combines these templates with your wiki pages and renders the result with the HTML Template Perl Programing Language module. Although you can edit the templates directly, I suggest you copy the directory with *cp -a* and make the *templatedir* variable point to your copy. If you need help editing the templates, start by reading the module's comprehensive manual page with *man 'HTML::Template'*.

Conclusion

In this article, we've set up the bare bones on an Ikiwiki. It works and there is nothing else that you *need* to do to it. With more than 100 plugins, I couldn't describe everything Ikiwiki can do without writing a book, so I leave it to you to explore this uniquely flexible wiki. I recommend beginning with a look at the Ikiwiki homepage [4].

Not only does it run Ikiwiki, but you can see many of the optional plugins at work. Also, you'll find comments and suggestions from people that use Ikiwiki in all sorts of innovative ways. ■

INFO

[1]	Ubuntu bug #375159: *https://bugs.launchpad.net/ubuntu/+source/ikiwiki/+bug/375159*
[2]	Markdown homepage and syntax reference: *http://daringfireball.net/projects/markdown/*
[3]	Ikiwiki plugins page: *http://ikiwiki.info/plugins/*
[4]	Ikiwiki homepage: *http://ikiwiki.info/*