Honed theme

This is the theme that powers macwright.com. Improved and optimized for nearly a decade, it's fast, accessible, and – some say – nice looking. It's really, really lightweight: blog post pages can be under 10kb. It's already tweaked for phones, maxed out to 100 on Google Lighthouse's Performance, Accessibility, and SEO metrics. There's built-in support for social share graphics.

Features

- Nearly a decade of painstaking optimization
- Social sharing cards for sharing posts on Twitter, etc
- `rel=me` links for your online identities
- Built-in reading log with star ratings, links to book websites
- Subtly tweaked for search engines: reviews use microschema data, JSONLD included in default template
- Atom & RSS feeds
- Git LFS configuration so that images don't bloat your website
- Minimal, tasteful, custom code highlighting colors

Dependencies

This is a Jekyll theme. The `Gemfile` points to Jekyll 4.2.0, the latest version at this writing. Jekyll rarely has breaking updates, so updating Jekyll in the future is not expected to be much of a problem.

It'll run locally: I highly recommend doing that with bundler. Once you have Bundler and a Ruby installation, installing dependencies is just

```
bundle install
```

I can't help with all of the different kinds of Ruby installations and the issues with them. I use rbenv but there are many options that also work.

The Ruby version that Netlify uses is controlled by the `.ruby-version` file in this folder.

Configuration

Start with `_config.yml`, the standard jekyll configuration in this repository. Most of the settings will be `you` and `Your Name` to start out with: tweak those and you'll get the correct name & domain showing up elsewhere in the site.

Then:

- Update `robots.txt` to point to your domain
- Update the `description` field in `index.html`

Recommended configuration

If you want to monetize with Brave Rewards, sign up for them and drop the confirmation file in the `.well-known` folder.

It's also nice to include a security.txt in `.well-known` if you are potentially receiving security-related emails.

I recommend signing up for the Google Search Console and dropping their HTML file into this site to confirm your ownership of the domain. That tends to help with debugging sitemaps.

The `_config.yml` has spots to drop in Indiebound and Bookshop IDs if you want to link out to those websites.

Icons

Websites require a lot of icons! This includes placeholders for all the ones you need:

- favicon.ico
- apple-touch-icon-144x144-precomposed.png
- apple-touch-icon-57x57-precomposed.png
- apple-touch-icon-72x72-precomposed.png
- apple-touch-icon-precomposed.png
- apple-touch-icon.png
- css/favicon.png
- css/logo-simple.svg

I recommend designing an icon and filling all of these with your generated icon. Highly recommend using optipng (installable with `brew install optipng`) or another PNG optimizer to make sure these are as small as possible.

Deployment

I've been using Netlify to deploy macwright.com happily for many years now, and I highly recommend it. There are optimizations in this repository - all of them optional - that hook into how Netlify works. That said, a motivated developer could translate these to their equivalents in Vercel or another system without much stress.

See `netlify.toml` for the build steps, which are:

```
command = "jekyll build && yarn install && node optimize.js"
```

So: build with Jekyll and optimize with Node.js. There's a custom cache plugin that works with Netlify's build plugin system in `plugins/cache`, which makes this optimization faster by caching optimized versions of image.

Optimization does the following:

- Creates scaled and optimized versions of images with `sharp`
- Removes unused CSS on each page and optimizes CSS with `csso`
- Compresses HTML with `html-minifier`

Images

Basically: you can handle images however you'd like, but there's a fancy path for it. I spent years using Flickr to host images by hot-linking them, and it worked fine. But I wanted to host images on my own domain, so I built something else. Here's how you can use that thing:

There's a script, `_scripts/image-localize.js`, that does the following:

- Finds images referenced in blog posts that are not in the `images` directory but that *are* on your local computer, or that are hosted on Flickr.
- Downloads or loads those images from your computer or Flickr.
- Resizes them so that they're exactly big enough to display properly at 640px wide with a retina display.
- Saves the optimized version into `images/`
- Updates the Markdown to point to that image.

So the workflow can look like:

1. When you include an image in a blog post, drop it in the root folder of your blog, like `/foo.jpg`.
2. Reference that in a blog post.
3. Before you publish, run `make`, which finds all posts and runs the optimize script on them.

In order to follow this workflow, you'll need to install the Node.js dependencies in the `_scripts` directory with `yarn install` or `npm install`.

It is *very recommended* to always use the `alt` attribute of images. In Markdown, that looks like this:

```
![alt test goes here](/image-url-goes-here.png)
```

The alt text will become part of the image's filename, making them a little easier to scan.

Local testing

We're reaching into the area of the Jekyll documentation here, but here's what I run when I'm writing a post locally and want to preview the changes:

```
bundle exec jekyll serve -w --future --livereload
```

Twitter sharing

Posts by default have social sharing features: Twitter card markup. Including thumbnails for posts is recommended.

You can include a thumbnail for a post by adding an `image` field to the YAML header of that post. The code that localizes images in blog posts will also localize and optimize twitter sharing images.

Reading log

This includes a reading log by default. If you don't want to track your reading on your website, you can just remove these files:

- `reading.html`
- `reading/rss.xml`
- `reading/atom.xml`

And delete the bit that links to the reading page in `_layouts/default.html`.

If you do want to track your reading: book reviews fit into the `_posts/books` directory. Book reviews are just like blog posts, but they have additional possible metadata: a rating, author, and book identifiers, which have examples in the example. You can add as many or as few book identifiers as you want.

Notes

This theme is fast in large part because it's minimal. It uses system fonts, doesn't include lots of graphics or any JavaScript by default. There's simply less transferred from server to client than other themes. I trimmed things from my website, year after year, as I figured out what was necessary and what wasn't.

Enjoy yourself with customizations, but remember that the performance of this theme - and all web design - is inextricably tied to our decisions of what is necessary to include. Adding lots of webfonts, images, analytics tracking snippets, or other things could turn this from a 5kb-average page to a 500kb page.

Other writeups

I've been working on this theme for a *long time*, so there's already a bit written about the different decisions behind things:

- How the reading log works
- How a page weighs 15kb
- Math as SVG for fast webpages