To start a project is to kill a portion of yourself. To kill a portion of yourself is to acknowledge the mortality of the whole. To be confronted with the mortality of the whole is to know terror. To conquer the terror of mortality requires creation, and thus we arrive at the beginning once more, as to begin to create is to start a project. To do so alone is folly, but what **ally** could possibly help us? Who would possibly follow us **from start to finish** through, in some strange way, a death? Better, why? Why follow, yes, but why ask? You must speak for yourself, **Madison Scott-Clary**, for it was by your own hands that so much of you is now dead. And now, you have acknowledged at last your inability to stop, and however slowly, your graphomania consumes you. Do you continue? Do you carry on until you are nothing? Are dust?

ally

from start to finish

ally

from start to finish

Madison Scott-Clary

Also by Madison Scott-Clary

Arcana — A Tarot Anthology, ed.

Rum and Coke — Three Short Stories from a Furry Convention

Restless Town

Eigengrau — Poems 2015–2020

ally

This publication uses the fonts Gentium Book Basic and Merriweather Sans and was typeset with XeLaTeX.

ally from start to finish

How many layers of remove is enough?

We ask how, because the question of why must ask itself.

How does one start a project?

With a bang, or with a whimper?

Very funny.

In all seriousness, though. How? Does one come up with an idea and just...what, go? Just start going and when you get to the end, stop? Can everything be, as NaNoWriMo would have it, pantsed? Run by the seat of your pants such that everything is done without planning, and thus nothing is unsurprising to the author?

Or does one plan meticulously? Does one craft an outline of such startling beauty that to finish the project itself feels almost a betrayal?

Which are you guilty of?

Both, of course. I have my fair share of projects I planned so thoroughly that they fell through, just as I have my fair share of projects that I tried worked and worked and worked on and kept adding and adding and adding, and by the end they were so wandery as to be incomprehensible. They didn't hold together, and the story had gone so far off the rails that it was unfixable without a total rewrite.

And which type was I?

Don't preempt me. All of the projects that I've actually succeeded at have been somewhere in the middle. It's important to plan, as I've learned from all those countless unfinished projects, but there is also a fine balance of planning required, lest you plan your work out of existence.

Qoheleth, the book that follows *ally*, has, as a major theme, the difference between honing and forging. To hone is to take an idea and work it to an ever sharper point, whereas to forge is to take an idea, work until its good enough, and forge onwards.

- ☒ *untitled furry thing*
- ☒ The Manifesto Project
- ☒ Tarot Experiment
- ☒ On Music
- ☒ Consequences of Dissonance
- ☒ Inner Demons
- ☑ Rum and Coke
- ▣ On Furry
- ▣ Sawtooth Universe
- ▫ No Thoughts Our Own
- ▣ Jaroudi
- ▣ ally
- ▣ Post-Self
- ▫ *untitled surgery novel*
- ▫ It's Not About The Dishes
- ▣ Poetry

Neither is bad, of course. There is no value judgement in this distinction. Neither, also, is there any sense of permanence to the label. *ally* was a project borne of forging: I was always trying to do something new with the typography, the wordchoice, the colors and textures of each of the sidequests, and so on. *Restless Town* was a project borne of honing, though. My goal with those stories was to try and somehow come to the finest possible point of the lives involved and the tropes and identities that drive them. I wanted to take aspects of myself — my gender, my mental health, my sexuality, my polyamory — and hone each into a story worth reading.

But, as with outlining versus pantsing, one can go too far in either direction.

> *And still, you will never not giggle when you write 'pantsing'.*

Correct.

All that to say that, as Herbert would hvae it, beginnings are such delicate times. To start a project is to kill a portion of yourself, because, whether or not you succeed in finishing the project, whether or not you are trying to hone something to a cruel point or to forge into new territory, you will never start that project again. You will never again be the you who started that project.

> *And so why am I here?*

May I throw your words back at you?

> *By all means.*

"Can an ally disinhabit a mind so easily?"

> *A question I remember you being decidedly uncomfortable with.*

Yes.

> *But why am I here **now**? Why when we are talking about how this project was made?*

Do you not deserve to be here for such a conversation? I trust that you will have little to say of much of the mechanics, but much to say about the process of research.

> *I do not doubt you. And yet you began this as a list of neat LaTeX things you learned along the way. How often does one write a LaTeX cookbook with one's imaginary friend?*

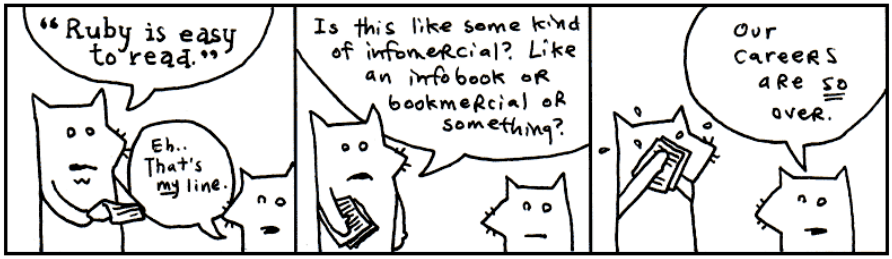
I don't know. Probably not often. There is precedent, though, for overused literary devices in technical writing. `Coy`[1], anyone?

> Error messages
> strewn across my terminal.
> A vein starts to throb.
>
> Their reproof adds the
> injury of insult to
> the shame of failure.

[1] Coy module on CPAN

When a program dies
what you need is a moment
of serenity.

Or perhaps you enjoy foxes[2] as I do:



There are all sorts of instances of folks writing technical things in a decidedly non-technical fashion.

> *If you say so. What, then, are you going to talk about in this technical guide?*

Thanks for writing my segue for me.

ally.id How the interactive side of ally is built, including some fun examples.

Page 5

The *ally* book How the book itself was built.

Page 15

Gotchas Some problems I ran into along the way.

Page 25

[2]From *Why's Poignant Guide to Ruby* by Why The Lucky Stiff, licensed under a CreativeCommons Attribution-ShareAlike license.

ally.id

We promise ourselves that we live our memories in a linear fashion. And who knows, perhaps we do.

What we emphatically do not do is remember our lives in linear fashion. *ally* began as an interactive project specifically to explore this aspect. The goal was to use the concept of interlinked pages to represent the way that one memory can be interlinked to another, and another, and so on.

> *And dreadfully distinct within the dark, a tall white fountain played?*

Something like that.

And here, we lean on a very specific deinition of hypertext. Hypertext is used to imply that some portion of a document can link to another portion of a document. This linking goes beyond simply the links that one clicks, as it can mean inclusions, such as when an image is included on a page full of text. It can, indeed, mean a link that leads from one page to the next, but what means 'next' here? Does it mean moving on to the next page in a series of pages, or does it mean moving from one section of the site to another? Perhaps it means moving from this site to the next.

With *ally.id*, this became a core component of exploring memory. It means something different to wind one's way down the singular path a

memory treads than it does to jump the track onto something wholly different. The central axis of the story is the death of Matthew as told through conversations with–

Me!

–with an imaginary alter-ego who, by virtue of that 'ego', knows all the same things I do.

Or more.

Perhaps, yes.

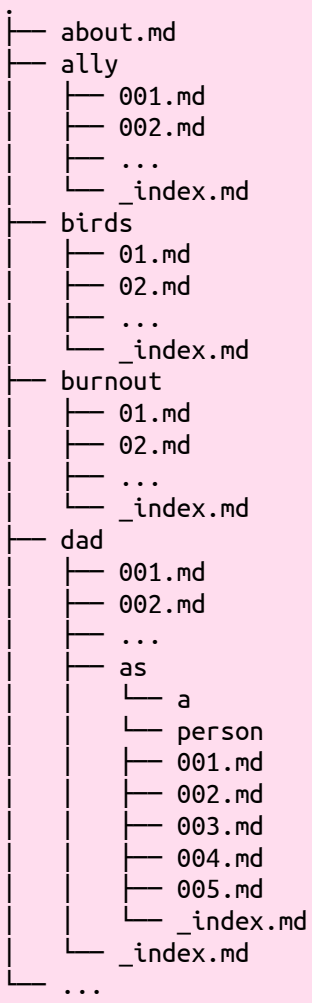
As a memory would be touched upon, it would spark a new branch of exploration that would proceed in much the same way. This is why the project is described as "arborescent": there is a central trunk with a defined beginning at the root, and from there, it blossoms up and out.

Or, it turns out, down and out.

Hypertext types

Axial	A set of linked documents that travels down a single axis, from start to finish.
Arborescent	A set of linked documents with a central axis, of of which may sprout other documents (which may in turn be axial or arborescent hypertexts).
Networked	A set of linked documents with no discernable axis. No start or finish, no direction to travel in.

While working on *ally.id*, Each page was kept in a single Markdown file, and each exploring branch was kept in a folder. Thus, we wind up with a file structure akin to what we see on the right.



> *Never content to condense your thoughts into something simple and easy to read, were you?*

To your scattered files go, I suppose.

Do keep in mind that this is a website, however. This directory, these filenames, this structure all play a role in how the whole project works. These whole tree of files are in the `content` directory of a Hugo project. Hugo is a program which knows how to take these Markdown files and turn them into HTML files while following a simple set of rules.

In each of those `_index.md` files, one will usually find nothing. That is, in terms of content, for at the top of each file comes a header which describes some of those rules. For instance, in some directories, we want the background to be a certain color, or perhaps we want there to be a link down at the bottom saying "back to where we left off".

> *There is no going, and there is no back.*

We can always pretend.

Another rule that is stated in these files is that they are intended to list the pages in that directory. Usually, this is done on a blog page where you might see the titles and first paragraphs of ten blog entries in a list, each with a "read more" link, followed at the bottom by a list of 'pages of results'. In my case, though, I set Hugo up so that, when it listed all of the 'posts', it would only do one per page, and instead of showing only the first paragraph, it would show the whole page. This essentially made it work as a book would: you simply turn the page when you're done reading. My

`list.html` file for this 'serial' layout loops over the pages in the directory as follows:

```
{{ $paginator := .Paginate .Pages.ByWeight 1 }}
{{ $content := .Content }}
{{ range $paginator.Pages.ByWeight }}
...
```

With this theme in place and with the files all in the right orders (each with a `weight` key in their own headers), Hugo will build the site as it stands.

> *Oh, but there's more to it than that.*

Of course.

Early on in *ally.id*'s life, I decided that there needed to be a map of the site. A literal map, too. No carefully broken down list of links for you to click on, but something more clearly representing the paths one takes through memory.

Your "Catastrophically Maddy" counter is ticking up.

Did you expect anything less?

I suppose not. Carry on.

Score one for Maddy.

So, even thought it takes a bit of work by hand everytime I add a page, I decided that it would be worth it to construct a graph that showed the arborescent nature of the project. I was tempted at one point to do the whole thing in Javascript using SVG so that it would track your movement through the pages, but even that was too much for me.

Wonder of wonders.

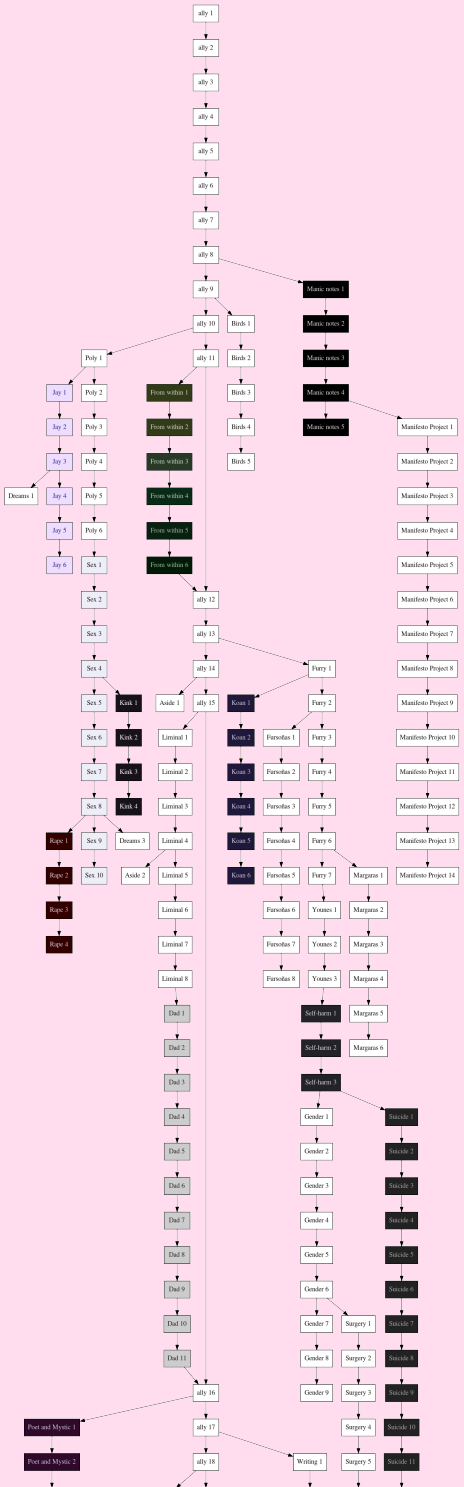
So instead, I leveraged existing tools–

Gag.

Right, sorry. So instead, I decided to use what I already had installed, which meant dusting off my knowledge of Graphviz.

In my `assets` folder lives a `map.dot` file which contains a node for every page and the edges that connect them. It's easy to add to; every time I add a new branch, I list every page inside of it along with its URL, and then draw all of the links that connect those pages together. At the bottom, there is the links from the trunk (or parent branch) to the branches.

```
digraph Map {
```



```
    node[group="dad",
         style="filled",
         fillcolor="#cccccc",
         fontcolor="#222222"]
    "Dad 1" [href="/dad"]
    "Dad 2" [href="/dad/2"]
    "Dad 3" [href="/dad/3"]
    "Dad 4" [href="/dad/4"]
    "Dad 5" [href="/dad/5"]
    "Dad 6" [href="/dad/6"]
    "Dad 7" [href="/dad/7"]
    "Dad 8" [href="/dad/8"]
    "Dad 9" [href="/dad/9"]
    "Dad 10" [href="/dad/10"]
    "Dad 11" [href="/dad/11"]
    "Dad 1" -> "Dad 2" -> "Dad 3" -> "Dad 4" -> "Dad 5" ->
    "Dad 6" -> "Dad 7" -> "Dad 8" -> "Dad 9" -> "Dad 10" ->
    "Dad 11"
```

And so on, until:

```
    "ally 29" -> "Burnout 1"
    "As a person 5" -> "ally 16"
    "From within 6" -> "ally 12"
    "Younes 3" -> "Self-harm 1"
    "Furry 1" -> "Koan 1"
    "Jay 3" -> "Dreams 1"
    "Liminal 8" -> "Dad 1"
}
```

While the whole file is much, much larger than just this, it is really no more complicated[3]. I rarely go back and change orders, and almost never tack a branch onto the trunk in a different place, so I just trace the lines once and let Graphviz sort the rest out.

Once I've updated the Dot file with the new pages, I type `make`, which, on seeing that the file has changed, runs `dot -Tsvg map.dot -omap.svg`, which generages the image itself.

The cool part about SVG is that it renders in the browser just as well as HTML, including with clickable links, so I get a *navigable* sitemap for free. All I need to do is copy and paste the contents of that `map.svg` file into the proper place in the site[4].

[3] Except... — page 26

[4] ...sorta — page 26

We have our content, we have our map, now we just need a way to get it up on the web so that others can see it, right?

> *A question like that never has an easy answer.*

Correct.

Hugo generates a set of folders filled with HTML files and static assets. It's no good on my machine, since that'd mean that I'm the only one that can see it. I could FTP the project up to a server and drop it where everyone could see that but...you know, now that I think about it, I can't remember the last time I used FTP.

Better, instead to just have another tool do it for me. Two other tools, actually.

> *You know, if you are trying to sell this as an easy way to approach a project, I don't know that you are succeeding.*

This is fair. One of the reasons it feels easy to me is that I was already running several other projects using the same technology. The reasons those work so well for me are closely tied with how I run my writing setup, and how I use my writing setup is closely tied with how I program.

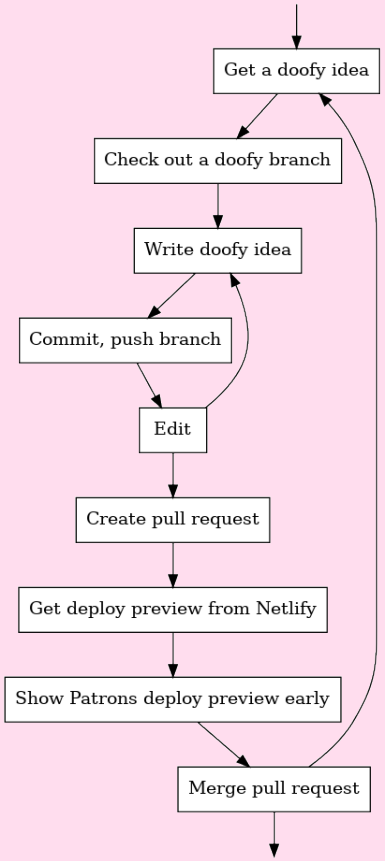
It makes sense for me to have a system that relies on convention over configuration, given how many software projects have worked that way. It makes sense for me to have a system that relies on publishing stories in files rather than database entries as I might with Wordpress or Ghost, given how much of my life is spent tooling around with other files. I'm hardly recommending this as a path forward. There are doubtless easier ways, regardless of how well this worked for me.

> *Right.*

Right.

So, given this Hugo site, the best way for me to work with deploying it uses a pair of tools: Github, which allows me to keep the entire site in a version-controlled repository synced remotely, and Netlify which will automatically build and serve static sites such as this one.

When I get an idea for a "sidequest" to work on with *ally*, I create a branch away from the `master` branch — 'branch' both in terms of git as well as in terms of hypertext, here — and work there. When I push that branch up to Github and create a pull request, Netlify sees this and automatically creates a deploy preview which I can share with those Patrons who get early access. When I click the big grean "merge" button on github, Netlify then builds the main site which everyone can see



The *ally* book

Okay, tell a lie.

> *Oh no.*

It's not that bad.

Another benefit that I get out of working with a set of files on the file system is that it is startlingly easy to get the very same data that is published on the web into the eventual book. I don't have to pull data out of a database to do anything, I just have to run a few commands.

As mentioned, I write in Markdown. It's a plain-text format — meaning I can edit it anywhere without additional software — with the ability to generate HTML. For instance, the first few lines of this chapter in Markdown look like this:

```
Okay, tell a lie.

> Oh no.

It's not that bad.
```

Using the `pandoc` command, it's super easy to translate Markdown to LaTeX, which is what is used to typeset the book[5]. For instance, `pandoc -o out.tex --wrap=none in.md` will do all the translation for me. Had I written this file in Markdown originally, that would have netted me:

```
Okay, tell a lie.

\begin{quote}
  Oh no.
\end{quote

It's not that bad.
```

I don't particularly like the look of the quote environment for you.

> *Neither do I.*

Right. It just looks like this:

> Yar har I'm the *ally* I'm not your friend.

Which is...not great.

> *Do I really sound like that?*

No. But I, of all people, am allowed to poke fun at you.

> *True enough.*

So instead I created an `ally` environment in LaTeX that closely matches the styling of the blockquotes on *ally.id*. It's easy enough to search for all instances of `\\(begin|end)\{quote\}` and replace it with `\$1{ally}`

```
\newenvironment{ally}{
\noindent\ignorespaces
\begin{quotation}
    \allyFont\itshape
    \noindent\ignorespaces}{
\end{quotation}\ignorespacesafterend }
```

[5] Well, a subset of LaTeX called XeLaTeX which allows custom fonts and colors and whatnot.

To help simplify this process, I made a `make` target which pandocs all Markdown files in the Hugo site over to LaTeX for the book.

Easy, as they say, peasy.

Lemon squeezy.

Deciding what to put into the book was almost as hard as writing the content itself. Deciding what to put in, where to put it, and how to display it when suddenly left without the benefits of clickable links was overwhelming.

And yet.

And yet.

I toyed with a few ideas. Originally, I considered printing the book wider than it was tall and giving each page five columns that would be filled in as the story branched.

This proved to be largely illegible. The mind can deal with perhaps two columns on a page, and even then, tying them together across pages rather than within a page requires additional visual clues such as different widths or different background colors.

So I took that and ran with it. I gave each page two columns of unequal widths and ensured that I could mess with the background colors separately — much like these pages here.

Actually *finding* a package that could accomplish this was surprisingly difficult. I was used to the `multicol` package, but that proved less than flexible. I poked a bit at `parcolumns`, but that also left much to be desired.

My solution wound up being the `paracol` package[6], which had everything that I needed, and much more besides. *Actually* learning it, however,

[6] When it didn't bitch about being too full — page 27

proved to be a bear. Much of the package documentation seemed to be written for those who already had some familiarity with its usage, and that lead to a steep initial learning curve.

In the end, though, I wound up with a setup that worked well for me.

> *And me.*

Well, yes.

In order of appearance:

Uneven column widths Before beginning the `paracol` environment (with the second argument being `2` for two columns), one needs:

```
\columnratio{0.65}
\twosided
```

Now, the main body of the text can take place in the `leftcolumn` environment, and the bullshit notes can appear in the `rightcolumn*` environment[7].

Column colors `paracol` allows you to set the background color of the column as well as the surrounding area, selecting the former with the `c` option and the latter with the `C`[8] option like so:

```
\backgroundcolor{c[1]}[HTML]{eeddff}
\backgroundcolor{C[1]}[HTML]{eeddff}
```

[7] the starred version begins the switch at the current position in the page, rather than the beginning of the `paracol` environment

[8] There are some margin issues, granted — page 27

Font colors The `fontspec` package proved to be most useful here. One can specify the color of text to use for the font family as arguments in the font family creation[9]. This often meant I had to renew font families:

```
\fontspec{Gentium BookBasic}[Color=222288FF]
\renewfontfamily\allyFont{%
  Merriweather Sans}[Color=4444AAFF]
```

One of the things that I've had a lot of fun with throughout this project is the style of the text, as well as having played around with the colors, as is perhaps obvious with the way I treat you.

> *You know you love it.*

I do.

Another thing I started poking at when working with the site was unique presentation of text beyond just blockquotes.

Part of that was to do with poetry. Early on when working with poetry and web publishing, I defined a new block in Markdown that would let me write how I wanted and have it show up (pretty close) to that on the screen. I wrap a block in triple single-quotes and it winds up in a `<div>` that has the `white-space: pre-wrap` style set. This lets me insert ridiculous spaces just how I want.

> *Ridiculous spaces for the ridiculous Maddy.*

It's not all that bad.

And hey, it's easy enough to accomplish in LaTeX, too. Just stuff it in a `verse` environment. Indentation is as easy as using the

[9] Along with some other garbage — page 29

`vin` command, which indents the line by...well, however large a space you set the verse indentation length to be.

At one point, however, I decided to fuck with some more complicated spacing. Given how focused LaTeX is on layout, it's maybe no surprised at how much easier this was to accomplish than in HTML.

And yet you were.

I suppose, yes. Part of the problem was going from how difficult things were in HTML and just kinda...expected they would be here, too.

Anyway, there were two that I wanted to accomplish. First, and easiest, was the spacing for portions of the surgery poem. While most of that poem was fairly simple in structure: sixteen stanzas of the two of us talking, consisting of rhyming couplets with a trailing line. That was all well and good, but, on describing the sensation of anaesthesia, I decided that the text should go all wibbly as well.

In HTML, this required quite a few repetitions of ` ` to space out the words. It was probably not the best way to do it–

Oh, but the heady intoxication of repetition...

–as I could have accomplished much the same as with the LaTeX `phantom` and the verse environment's `vinphantom` commands, which create an amount of horizontal space equal to the length of the text in the argument. That is, just use a set of `<span>` tags with transparent text to space out the visible words. Bit harsh on screen-readers, as it would mean plenty of the lines would be repeated–

How appropriate.

–upon reading, but it *is* being used as a visual medium here.

As mentioned, it's much easier in the print version:

Was this death?
Nothing. Nothing, death? nothing.
Nothing,
Was this death?
Death? Nothing.

There was nothing.

This is accomplished with the aforementioned commands:

```
\vinphantom{Nothing.} Was this death?\\*
Nothing. \phantom{Was this death?} Nothing, death? \phantom{Nothing,} nothing.\\*
\vinphantom{Nothing. Was this death? Nothing, death?} Nothing,\\
\vinphantom{Death?} Was this death?\\*
Death? \phantom{Was this death?} Nothing.\\*
\vspace{1em}
\vinphantom{Death? Was this death? Nothing.} There was nothing.\\
```

Secondly, I wanted to vastly reduce the line-height for a section of verse. In CSS, I need only set the `line-height` property to something less than 1. Thankfully, in LaTeX the `spacing` package exists for this.

```
\begin{Spacing}{0}
speak to me\\\vspace{-7pt}
speak to me\\\vspace{-7pt}
speak to me...
\end{Spacing}
```

And was it?

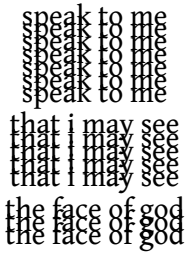
Was it what?

Death.

Of a sort, yes. You will have to read the book to see.

Note that this must be centered or otherwise not indented, or the first line of the paragraph will space itself separately.

leads to:



A few locations in the book require full-page illustrations. This is surprisingly difficult using the usual `includegraphics` command from the `graphicx` package, as there's a lot of page layout stuff that happens before and after the image itself is included.

Rather than trying to include the image, however, it is startlingly easy to just include a PDF within LaTeX. Using the `pdfpages` package, all I need to do is save the image as an appropriately sized (that is, 8.625"x8.625") PDF and then use the `includepdf` command instead. LaTeX will manage all of the page numbers accordingly, and I don't have to worry about splicing it after the fact.

Speaking of, for the digital version (as well as for this PDF), attaching the cover(s) to the final PDF *does* require some splicing. This is made very easy with the `pdftk` command.

```
pdftk cover-front.pdf cover-back.pdf internal.pdf \
  output digital-edition.pdf
```

`pdftk` commands are just read from start to finish, so the above says, "Take these three PDFs and output them all to the single file, digital-edition.pdf." As a longer example, providing a sample from the book works like so:

```
pdftk book.pdf cat 1-63 output intro.pdf
```

Can you imagine so blinding a sight?

Yes. I could see it with my eyes closed.

which says, “Take book.pdf, concatenate pages 1-63 and then output them into intro.pdf.” It’s a pretty neat command.

The last step of the process was generating the index.[10] Back when I was writing *everything* in LaTeX, I remember this being far harder than it actually turned out to be.

Wherever you want an index entry, all you nened to do is put `index{...}` in the text with whatever the entry is. This supports nexting entries (separated with an exclamation point) and page ranges (referenced with a `|(` and a `|)`. So, for instance, I indexed poems like so:

```
\index{Writing!samples!poetry|(}
\begin{verse}
Blah blah blah\\
My name's Maddy and I can't stop writing\\
...
\end{verse}
\index{Writing!samples!poetry|)}
```

To generate the index, I have to first run `makeindex book` (given that the filename is book.tex), then I can run the usual `xelatex book.tex; xelatex book.tex`. I have to run that twice in order for the page references to be correct, of course.

At the beginning of the book in the preamble, I include the `makeidx` package and immediately run
`makeindex`, and at end of book.tex, where I want the index to appear, I call `printindex`.

[10] Aw heck, really? — page 28

Gotchas

Hugo, snaps, and the treachery of automatic updates

Look, I *understand* the draw to automatic updates. It means that no security bugs go ignored, that everything can be done unattended, all that wonderful nonsense to avoid users causing problems.

But...

Yeah, but.

But the problem comes when a bit of software's API changes, and suddenly your entire setup breaks for reasons unfathomable. Hugo, the software that runs *ally.id*, is installed via a snap, which is a bit of auto-updating software. It's all well and good, except when Hugo — still a young project — decides to change some aspect of how they build the site. The amount of times I push a change to Github only to find that, when it's built on Netlify, the entire homepage is blank, minus the header and footer, is nontrivial.

Another problem was when they switched away from one Markdown processing library to another, resulting in a change that disallowed raw HTML, meaning that all of those fancy pages with Javascript and whatnot suddenly didn't work.

The things we do for art.

Alright, are we ready for a gentle dunk-on-myself session?

Can I talk you out of it?

I'd prefer not. There's a lot that has come up over the past few months, and I need to get at least a little bit of it off my chest. I need to come up with some way to square a few things that have happened since I started *ally* with much of who I was before.

Have at it, then, but fair warning, do not expect me to be anything other than what I am.

An ally, not a friend. Right.

So, I titled this section "Gotchas" with the full intent of going into it full steam ahead, bitching about all these

I guess.

The solutions varied, but it mostly meant that I had to stay on top of making changes via PRs on Github rather than pushing directly to the main branch. If things broke, then I could have my little sulk and then go through the Hugo changelog and figure out what fucked up where. Half the time, it simply meant clearing the build cache on Netlify.

Branches to left or right in dot file

One of the problems I ran into with the map of nodes was that graphviz generally sorts branches from left to right. If I added a new branch of nodes after the main `ally` branch, they'd be positioned further to the right of all existing branches, leading to an unbalanced tree.

Does that make me the trunk?

But you're in all the branches.

Oh, nevermind.

Right.

So.

The solution is to list those nodes before the central axis. You can make all the links later on, but the rank direction is based on when the nodes are defined.

Graphviz sets fonts to stupid things

When building the SVG version of the map, Graphviz helpfully sets the font of all the nodes to `Times,serif`, which is...well, maybe it's a really set of fonts? Either way, it's a gross set, and it's not cohesive with the rest of the site.

tiny problems I ran into along the way that kept me from focusing on the task at hand.

And I did, too. I think so, at least. There were all of these tiny things that kept me from sitting down, just writing a whole slew of pages, and throwing them up for Patrons or whatever. I kept running into tiny problems that would eat hours of my time with minuscule amounts of payoff, only for them to suddenly resolve, sometimes inexplicably.

But...?

But the biggest problem I came across when working with the project is not really anything technical, but more one of self-worth. More than any one technical problem, the thing that I wasted most of my time on instead of creating was dealing with your...your what? Your inverse? Your mirror?

Can a mirror have a mirror?

Are you a mirror?

Touché.

There is something within me that is the negative of you. Something which, when

It's all about the brand, baby.

You're not wrong. And the brand, here, is to use Gentium Basic.

There is no elegant solution to removing the `font-family` other than a very dumb `:%s/ font-family="Times,serif"//g` in Vim. Alas.

`paracol` environments can only handle so much

A strangly baffling problem that I ran into when switching to `paracol` was this delightful error:

```
! Package paracol Error: Too many unprocessed columns/floats
```

It's one of those problems that, like...makes enough sense on its own that I thought it was something that I had done. Had I added an extra column in there? Had I forgotten to add a column somehow?

Of course, then I read the docs and it turnsout that this is a known problem. The problem crops up when one of the columns gets too long without you either switching to another column environment or just starting a new paracol environment. In the end, it worked out such that, so long as I didn't have more than one `paracol` environment per file, it'd be fine, whereas initially I'd had a single environment wrapping several files.

Setting color margins in `paracol`

It is all well and good to set colors for the column background, but this does not, by default, extend to the limits of the page. Having just a bar of color with text in it to the side surrounded by white is not very pleasing.

To get around this, the `paracol` packageallows you to set an extent for the colors surrounding the background area. For the background of the text block, one uses the `c` selector as above:

light is shone through it upon me, makes you, and yet for that bears some level of horror to it. Where you are light, it is dark and vice versa, and you do not realize until you see it the terror that is involved in the inverse of a shadow.

And this thing, this not-you, does the opposite of what I know you will provide in the future. Where an ally may kick the shit out of me for all those things that I do to self-sabotage, this thing kicks the shit out of me for all of my successes, for everything that I do right. It is *The Tower*, to your *Star*, or perhaps *Moon*.

The biggest gotcha I ran into is doubt. Doubt as to my worth. Doubt as to my skill. Doubt as to my wisdom in partaking in a project so counter to that which life demands.

Not your enemy, but your adversary.

Yes, that. I like that. I like the way the word hints at devils and demons. I like that it implies that this adversary is not striving against me that it may succeed, but simply that I fail.

```
\backgroundcolor{c[1]}[HTML]{aaaaaa}
```

and then for the area surrounding it, one uses the `C` selector. However, there is an optional set of arguments to that selector for how *far* that area should extend beyond the text block. A good default was to have it extend 60% of the way into the gutter between the two columns (50% risked a thin white stripe between them, and the overlap was still handled gracefully with a higher number), and 10,0000 points on all other sides.

```
\backgroundcolor{C[1](0.6\columnsep,10000pt)(10000pt,10000pt)}
[HTML]{aaaaaa}
```

I tried to figure out more exact numbers, but it turns out that that is *a)* not trivial and *b)* not worth it because LaTeX does't really care.

Munging the index for page breaks

The index is generated blithely. That is, the `makeindex` command generates a .ind file which contains the basic layout of the index: nested lists of references and page numbers set into columns, each new letter separated by a vertical space as specified by the
`indexspace` command.

Unfortunately, without that being any smarter, it's easy to wind up with widows and orphans here, which, when indentation holds semantic content, can cause problems. In the case of *ally*, the entry for *Mental health*, the first entry under *M*, was the last entry on the page, making the first entry on the next page *anxiety*, which was a sub-entry under *Mental health*. Without that indentation being visible, of course, that was impossible to see.

The adversary would like that I understand, deep down to my core, that I am in all ways unworthy of this project. I am unworthy of the right to talk about myself. I'm unworthy of the words I write and the folks who read it and the reviews I get.

It is the one who stood before me when I was looking at getting reviews, at asking my friends and partners for feedback on the book, and said, "Who are *you* to ask such a thing?"

The number of times I set aside working on this project with the thought *it's right, after all; to ask for someone to engage with me on such a level is to ask for them to consider me as a person, and there is no greater sin* is nontrivial.

> *It feels a little unfair to say because of how trite it sounds, but the biggest `gotcha' with the project was yourself.*

Oh, one hundred percent.

It's not totally this project, either. I ran into the same thing with *Restless Town*. Anything I make that is at all meaningful to me — that is, anything

The solution was, when generating the index, to warn myself that this was the case, and then go into the book.ind file and manually add a `newpage` command right before the *Mental health* entry. Sigh. Such a pain.

Needing `Ligatures=TeX` in fontspec when renewing text

That LaTeX provides such fine control over fonts and typesetting is fantastic. It's also a royal pain in the ass.

As mentioned, *ally* uses XeLaTeX specifically for its delightful handling of fonts. When you specify a font in the preamble, it does everything right. However, when you `renewfontfamily` within the document — say to change the color — it stops using the proper ligatures. That is` stops displaying as ".

In order to fix this, one must pass the `Ligatures=TeX` option to the command:

```
\renewfontfamily\allyFont{Merriweather Sans}[Scale=0.9,
Color=777777FF,Ligatures=TeX]
```

Various printing problems

Oh paper, why must you be based in such imperfect reality?

The treachery of inanimate objects.

Yeah.

There were a few problems with the printing process. The first batch of books came with bleed problems, color saturation problems, and color mismatch problems:

that *I* feel is worth sharing — is too much to ask others to engage with. "How dare you," it says. "How dare you ask that others consider your work meaningful."

Master sigh.

That's an Andrew Bird song.

Does it not encompass the mood of ``I know that this thing is wrong and am able to understand that on an intellectual level, and yet I must still deal with it on a constant basis''?

The sigh to end all sighs, yes.

So how did you conquer it?

Conquer?

We are here, after all, yes?

I don't know that I'd say conquer. Won a battle, perhaps, but not the war. I suspect the war will end with true-death.

And as for that victory, I suppose it was through the aid of allies.

Me!

You can see that the ink coverage on some of the pages is so high that it caused the pages to buckle when running through the printer. Additionally, while the intensity of the colors remained much the same as in the PDF, the saturation has been knocked down slightly. The improper bleed is

To an extent, yeah, insofar as you are a manifestation of graphomania, at least in part. But also exocosmic allies. Allies outside of myself. Allies like Robin and JC and Justin. I don't know who else read the site when it was getting regular updates, but I suppose I thank them. I thank those days on Matomo when I would see someone wander through almost the entirety of *ally.id*, page after page, without stopping.

And what feedback I did receive (for not enabling comments was an intentional decision) helped push me over the edge. Linnea's review, that anonymous *Kirkus* reviewer, all those little words of, "This is cool. This is interesting. This is impactful." All of those helped push me over the edge and into publication.

This reads like a dedication.

So?

No harm in it, but it bears mentioning. Either way, I'm happy that this became a project that you could believe in enough to turn into something big.

shown by the thin white strips along the colored borders where the printing stopped before where the page was cut.

The last one was on me, but the first two were just due to the mechanics of the printing process. The solution, if I wanted to keep my colors as they were, was to use a thicker paper. This drove up the cost of the book, but I think, in the end, led to a much lovelier product. At $50 retail, it's steep, but for a book that is more experience than anything, I'm alright with that trade-off.

"Happy"? Is that a thing you can be? Has your bailiwick expanded to include additional departments?

I'm cross-training.

Well, thank you.

And, of course, thank *you*.

So. There it is. A project from start to finish. A story. A file. A book. To start a project is to kill a portion of yourself, and that is what I've done. I've destroyed that bit of me that was there before I began this whole process. It's not there anymore. It's gone.

I feel its loss.

I feel wrung out.

I feel empty.

And for what?

Will this project — *ally* and this making-of — go anywhere? Will I somehow gain notoriety of any amount by publishing this? Will they provide others with meaning? With understanding?

I don't know.

Do you want to?

I don't know. Maybe. Maybe, like everyone else, I just want to be seen.