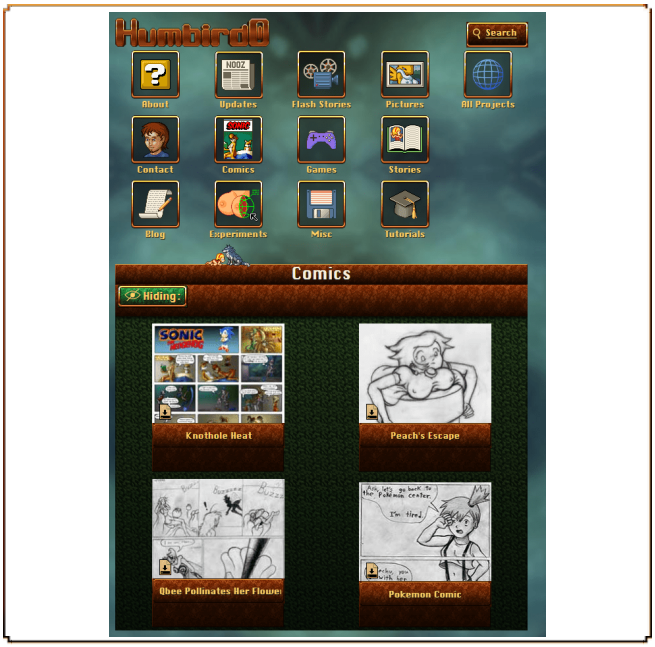
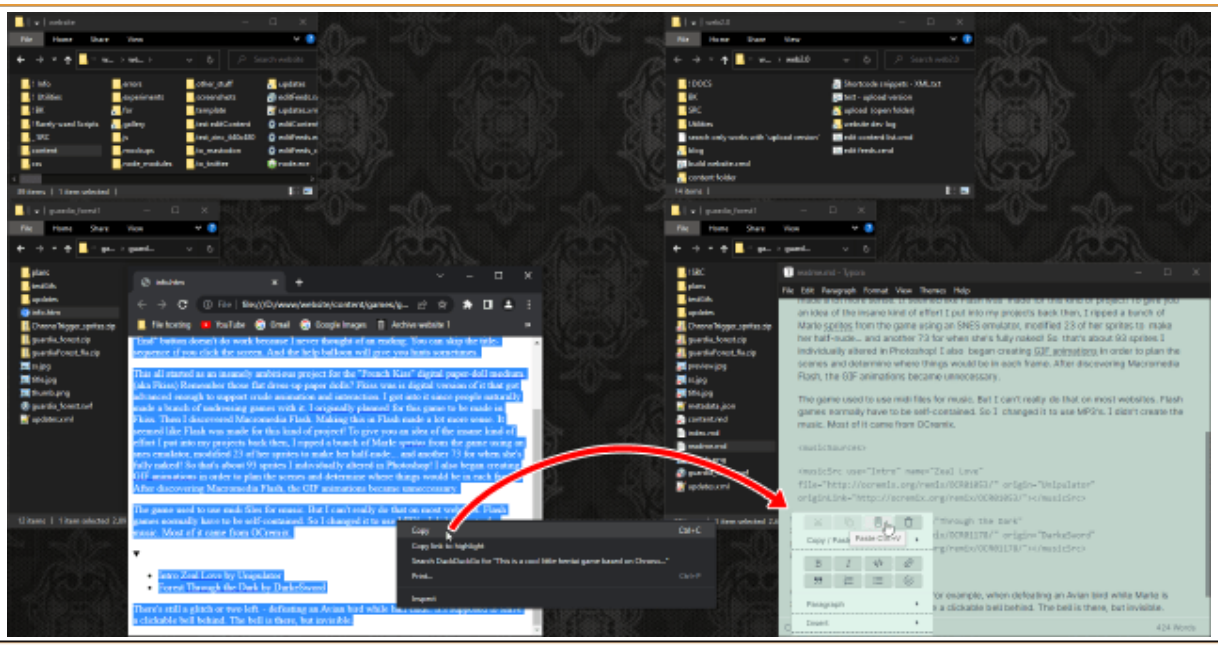


How do you write stuff for a website in a future-proof way? I'm not talking about the HTML and layout. I'm talking about the content. The english language part of it. The lowest common denominator is plain text. HTML itself has pretty good compatibility going back more than 20 years, but not every feature survived. Remember when you could play MIDI files as background music? Or play Flash animations?

Last year in 2023 I redesigned my website. It was a tedious process because I had to double-click the old HTML files, select all the text, copy it, and then paste the text into a shiny new markdown editor.

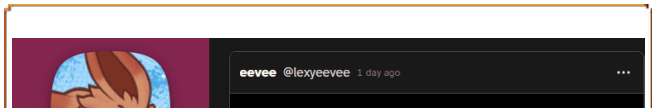


It's not the first time. 14 years ago in 2013, I also redesigned my website. I made a custom editor using Flash to generate JSON for my JavaScript single-page-app website. But that website wasn't all JSON. Each project also had a readme file. A simple HTML file with minimal markup. So I double-clicked each of the older HTML files, selected part of the text,

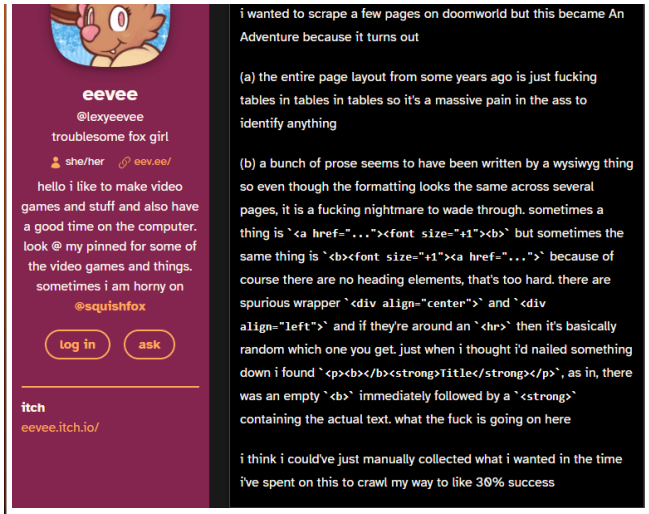
copied... and then pasted it into that project's HTML readme file. This seemed to make sense. HTML is basically just XML right? Surely I could just use an XML program to read the data in the future.

It turns out that HTML is ***not*** proper XML. Maybe XHTML was going to be, but that didn't really take off. To put things in perspective:

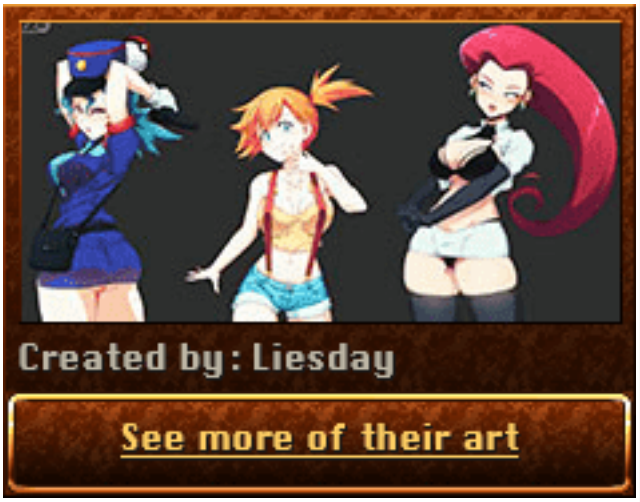
- In 1997 HTML was the format of the future.
- In 2005 XML was the format of the future.
- In 2014 JSON was the format of the future.
- In 2016 Markdown was the format of the future.



Why didn't I just use the HTML I already had? One problem was that my old website was inconsistent. In spite of my efforts to use



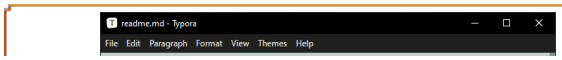
minimal HTML, it was still all hand-written and so the same things would get written in slightly different ways on different pages. A computer program would struggle to recognize them all, and I would struggle to remember all the places I wrote something.

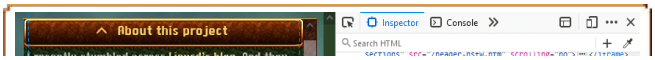


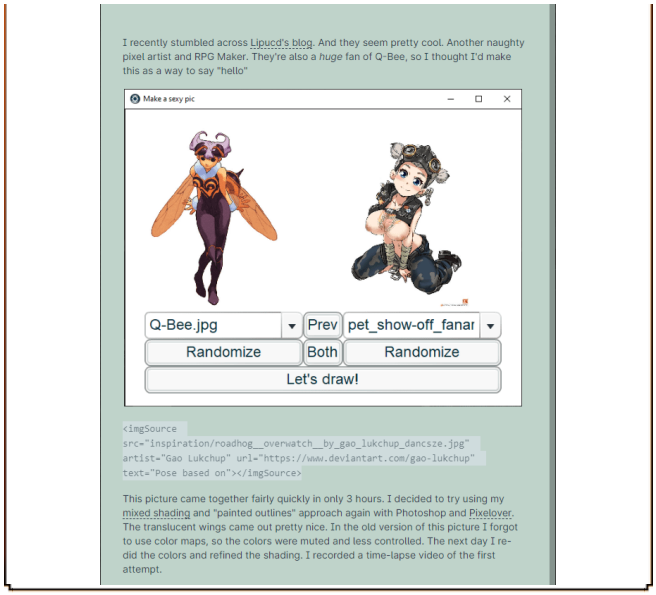
I also had new ideas when I redesigned my website in 2023, about what ***kind*** of information I wanted to store, and how I wanted to present it. I often recommend artists. How should I store a "recommendation?" It's a very niche and specific concept, so there is no single dedicated HTML tag for this. And being an abstract concept, it can also be presented in multiple ways. All equally valid. Should it be a semantic FIGURE tag with a fig-caption? But what if I want to include a link with it?

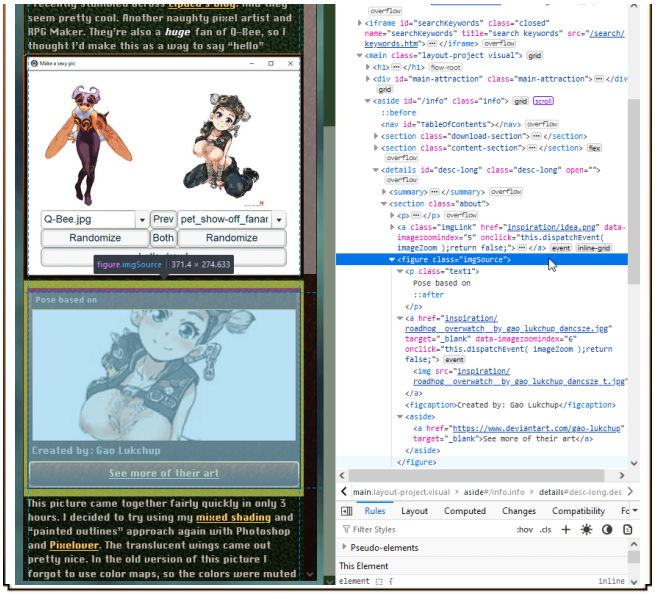
I also learned more about semantic HTML, schema microformats, and new CSS features that make new layout methods possible. What I needed was a more generic way to represent things, and a separate way to handle their presentation. It's always a good idea to separate content from presentation, so that you can change how things look without having to modify ALL the content. That's why CSS was invented in the first place, but it's not enough by itself. And even though HTML lets you add tags with arbitrary names, I still need to include things like normal HTML links inside of them. The real problem is that composing ***multiple tags*** together runs the risk of creating consistency. I can easily write them in any order. But what if I could represent something with a single tag instead?

So here I am in 2023 / 2024 storing my website's source as Markdown, and using HUGO to convert it into HTML. I now represent abstract things like artist recommendations using single "XML" tags because I found a way to make Hugo interpret these as short-codes so that it replaces them with whatever set of tags I want when it generates the HTML files. And if I lose that option in the future, any other markdown editor will harmlessly pass-through these "tags" into the HTML, where something else could potentially handle them.









This has two advantages. HUGO will always write this HTML exactly the same way every time, and my markdown files store the content in a form that is even simpler than HTML, without combining tags together.

Hugo isn't limited to generating HTML either. It can write any kind of text file. So in another 10 years when I (very likely) redesign my website again, I *assume* I can probably just use HUGO to convert the markdown into the next text format "of the future"... assuming computers can still run windows programs written in x86 code, but that's not guaranteed. I suppose there's always emulation.

Part of my inspiration came from reading about how Cory Doctorow handles his blog. He is a very prolific blogger who has used XML for over 20 years and uses various python scripts to convert his articles for his website and social media posts. That probably works,

but I don't envy having to edit XML markup by hand.