p5 was a huge leap forward for the *sharing* of Processing projects. We can now host our projects on the web where others can view, interact with, and learn from them immediately via their browsers. I'd loved Processing when I learned it as a design student but eventually stopped using it because I couldn't share anything easily with others; when I discovered p5 recently I was so distractedly happy I could hardly think straight enough to build my first sketch…in a good way!

However, p5's pixel-based nature still poses limits upon the projects it is used to build, and therefore also upon the audiences of these projects. When a p5 coder assigns the sizes of a canvas and its drawn elements with pixels, they're inherently choosing the screen size that their sketch's experience will be optimized for, and therefore also which screen sizes it will be too large or small for. Using `windowWidth` and/or `windowHeight` for a canvas' size does not necessarily solve the problem easily if one wants to maintain a fixed aspect ratio for their canvas. Even more math and rigor is required to maintain a canvas-size-to-drawn-element-size ratio, both of which are imperative for rendering data visualizations and maps, and often desired for other types of sketches as well. And even so, to keep a canvas' aspect ratio in tact and scale with window size via `windowWidth` or `windowHeight` means sacrificing potential device orientation optimization: a square canvas sized with `createCanvas(windowHeight, windowHeight);` would always look great on horizontal laptop screens and landscape-oriented devices, but by definition would completely overflow the sides of the screen on portrait-oriented devices. This too can be overcome, but only with many more semi-complex calculations and code, thereby limiting its usefulness to those with the mathematical thought processes and patience to do so.

I want to enable the p5 community's creations to be even more shareable and consumable, by building a library that encourages scalable, screensize- and proportion-aware p5 sketches, sans the algebraic overhead currently required.

I am a UX designer and information architect, and spend most of my free time experimenting with code. I have confidence in my ability to envision, plan, and prototype an optimal experience by which p5 coders could employ the responsive, proportional sizing functionality I propose—in a way that aligns with p5's existing syntax patterns.

I've started exploring this idea and baked the functionality I mentioned into a mini-demo as a first pass: https://github.com/liaprins/p5.responsive. Now when I build a sketch, I simply set and change variables as I wish (for parameters like max and min widths and heights, canvas aspect ratio, whether I want drawn elements to be defined as static px (default) or as percentages of the current canvas width, how much margin spacing to put between edge of window and canvas, etc). However, I have never yet built a JavaScript or p5 library, so I'll need to learn more about how to do that and best practices, as well as iterate on the core code I've started (see development document for more detail on my proposed process).

Overall, this would certainly mean a better experience and more options for the p5 coder, but ultimately also a better experience for *their* users. It would mean reading less "best used on desktop" caveats on social media and portfolio sites, replaced instead by immediate interaction with the real project, even for those on their phones. Given that phones are how many people browse social media—and social media in turn is how many people share art projects and progress—responsive sketches could reach broader audiences. It would also mean that those without access to a computer for any reason or at any time (whether due to cost, or due to convenience—such as standing-room-only bus commutes) wouldn't necessarily be inhibited from experiencing the creations that p5's community is capable of bringing into the world.