Squishy Music Toys

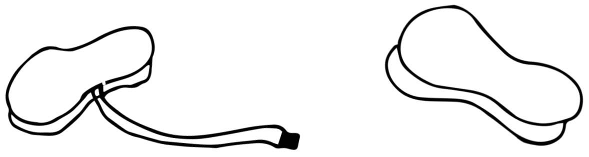
Creating a less stressful, more pliable way to enter the music world

by
Hannah Rhiannon Lienhard
B.S. in Mechanical Engineering, Massachusetts Institute of Technology, 2019

Squishy Music Toys

Creating a less stressful, more pliable way to enter the music world

by
Hannah Rhiannon Lienhard



Submitted to the program in Media Arts and Sciences, School of Architecture and Planning in partial fulfillment of the requirements for the degree of Master of Science in Media Arts and Sciences

Abstract

Electronic music education today does not have a clear path to entry - the existing methods are expensive or overly complicated, making them unavailable to a lot of people. In this thesis I will explore a possible solution to this issue, through the creation of a new kind of music interface designed to let anyone, at any age, have the experience of playing an instrument. This interface - the *Squishy* - is soft and pliable, allowing a comforting yet exciting new way to create and control music. By making it this way, you eliminate much of the detailed technique that is often required when learning an instrument. The *Squishy* guides users through the basics of electronic music with a simple software interface. Each instrument is embedded with sensors that respond to bending and pressure forces on the exterior shell. These sensors are incredibly reactive, so even small changes to the shells can affect the sounds being created. In this way, the *Squishies* can also act as a tool for meditation. By being so sensitive, the interfaces encourage users to concentrate on the sounds they are creating, and can guide them to be more intentional with their movements. The *Squishies* are also able to act as MIDI controllers for more experienced users who want to utilize them with their preferred music software. Throughout this thesis, I will explore the design space and potential use cases for these instruments, as a toy and learning experience, as a high level music controller, as a tool for meditation, and as all three.

Thesis Advisor:
Tod Machover
Muriel R. Cooper Professor of Music and Media

Squishy Music Toys

Creating a less stressful, more pliable way to enter the music world

by

Hannah Rhiannon Lienhard

This thesis has been reviewed and approved by the following committee member:

Advisor

Tod Machover
Muriel R. Cooper Professor of Music and Media
MIT Media Lab

Squishy Music Toys

Creating a less stressful, more pliable way to enter the music world

by

Hannah Rhiannon Lienhard

This thesis has been reviewed and approved by the following committee member:

Reader

Cynthia Breazeal
Professor of Media Arts and Sciences
MIT Media Lab

Squishy Music Toys

Creating a less stressful, more pliable way to enter the music world

by
Hannah Rhiannon Lienhard

This thesis has been reviewed and approved by the following committee member:

Reader

Hiroshi Ishii
Jerome B. Wiesner Professor of Media Arts and Sciences
MIT Media Lab

Acknowledgements

I would like to thank:

Tod Machover for all of his unending support and guidance, and for letting me name my thesis project the *Squishies*. My readers, Hiroshi Ishii and Cynthia Breazeal, for their feedback and wisdom.

Steve Gerasimoff, who did not put up with me doubting myself ever, who fed me pizza when I was busy being a *Squishy* factory, and who never stopped caring for me, even when I was a ball of stress. Alice Hong, who has been by my side throughout this entire thesis, and is always there to provide advice and friendship. Asli Demir and Lauren Schexnayder for being haunted, and Amanda Lin, for being a constant source of good in my life.

My family, for filling me with love and support always.

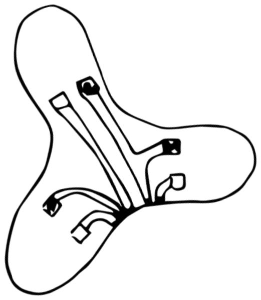
Megadesk, which consists of: Aarón Montoya Moraga, who I want to thank for amazing conversations about music and code and everything in between. Karsten Schuhl, for being a constant sounding board for thoughts and ideas, and Manaswi Mishra for being a positive force of being and inspiration at all times.

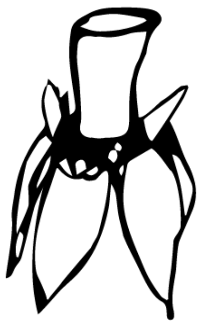
The rest of the Opera of the Future group: Nicole L'Huillier, Rebecca Kleinberger, Alexandra Rieger, Charles Holbrow, Nikhil Singh, Priscilla Capistrano, and Sizi Chen, with a special thank you to Nicole for being one of the best mentors I have ever had, and for being the reason I felt like I could really do something unique with my life.

My amazing UROP, Jenny Zhao, for putting up with me and my weird ideas, and all of my workshop participants, for their willingness to play with a *Squishy* before anyone else.

And a final thank you to Ashley Norwood, who brought me incredible peace through her yoga practice during an otherwise stressful time.

Contents





1

Introduction

Introducing music education and opportunities into the lives of young people is important for several reasons. Music, when presented correctly, can serve as a therapy for those experiencing anxiety in other aspects of their lives.[1] Music therapy is often used to treat anxiety and depression, and has also been used to aid people with developmental disorders. Music education has also been shown to have many developmental benefits[2] for kids, including improvements in language development, memory, mental processing, and problem-solving abilities.[3] There is a huge need for music education - but in a new format that can promote all of these positive qualities, without promoting the ones that can be detrimental to learners, like music-related stress or performance anxiety. This thesis will aim to address this need through the creation of a new kind of educational music experience designed to build confidence in young musicians, while teaching them basic electronic music principles. By doing this, more people will also get the chance to try playing a musical instrument, even if just for fun.

This new music experience will take the form of a soft and squishy music toy - a *Squishy* - that is paired with its own unique software interface, allowing users aged 8-14 to explore and learn on their own. A soft interface was chosen for several reasons - there is a level of pleasantness associated with squishy instruments[4], they are easier for a wider range of people to play (as they do not require fine motor skills), and they provide a tactile experience similar to a stress ball. A soft interface also lends itself to a more fluid kind of music: you do not have to play specific notes or rhythms to make something that sounds interesting.

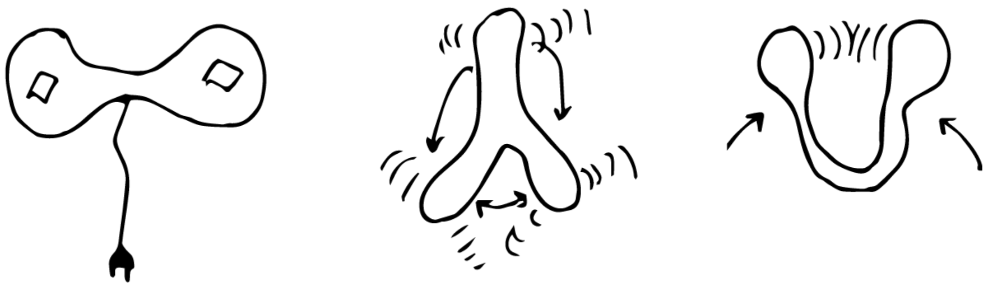


Figure 1: Concept sketches for some of the first *Squishies*, showing potential interactions.

The *Squishies* set themselves apart from past projects by introducing more freedom for 'tinkerability' into the experience through a software interface. Musical tinkering as it is referred to here follows Eric Rosenbaum's definition as "a pathway into music-making through playful creation of musical artifacts."[5] In other words, the *Squishies* will let users explore the different ways they can create music by giving them the tools to manipulate sounds as they wish. This interface will let the user control different aspects of the music - pitch, volume, tempo, etc. - by changing how they are mapped to the different sensors embedded in the *Squishies*. By doing this, kids can have autonomy over how they create music, and they can make their own unique path through the music world.

This concept of more fluid, less structured music is something that I am very interested in. Even though I have spent a large part of my life learning the violin through very structured, traditional music lessons, over the past several years I have begun to gravitate more towards music and instruments that allow me to be fluid with my notes. I received a theremin as a graduation present when I finished high school and was immediately hooked. A theremin[6] is an electronic instrument that has two antennas - one controlling pitch and the other volume. To play it, one would hover each of their hands close to or far from the antennas to change the music they hear. All the pitches that come from a theremin are non-quantized, unlike a violin which is generally played with very specific pitches in mind. The violin as an interface is capable of playing fluid notes as well, but it has such an extensive history and repertoire that much of what users are encouraged to learn, and what becomes associated with it, is very structured and constraining. The theremin also requires no physical contact to play, which makes it even more abstract than most instruments. In other words, it was completely different from anything I had encountered before in my musical career - all I had to do was wave my hand around and I would create a stream of pitches all floating together in space.

Now, the theremin as an instrument can be mastered of course - if you spend a lot of time with it, and learn all the precise places in space to put your hands, you can play incredibly intricate pieces. It is a difficult instrument to master because of how few constraints there are, but the same aspects that make it difficult also make it easy for anyone to approach and start making sounds. All you need to do is be present. This idea especially resonated with me when I first encountered the theremin. As much as I loved learning the violin, it was always a little frustrating knowing you would have to practice so much before you would be able to play anything that sounded remotely like music. So, in designing the *Squishies*, I wanted to include some of the aspects of the theremin, by maintaining both the low floor to entry/music making, and the high ceiling for virtuosity.

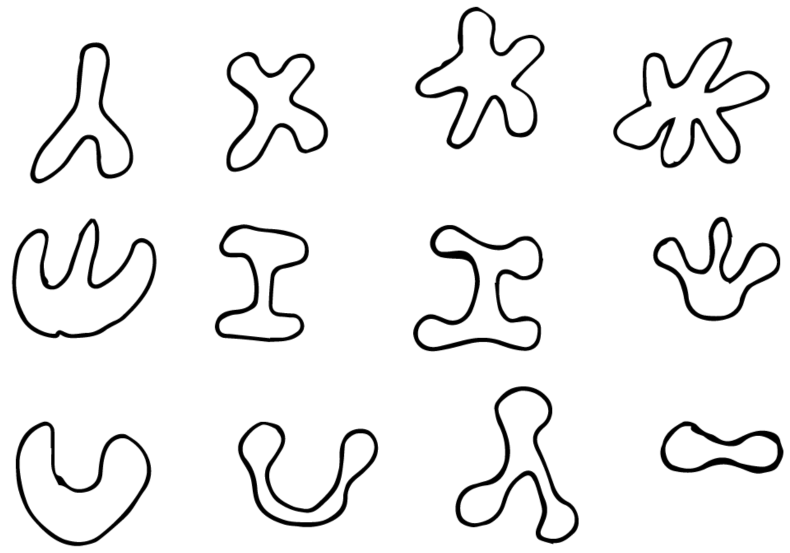


Figure 2: Concept sketches for a variety of shapes that were considered for the *Squishies*.

Western classical music education has traditionally been a very structured process: you choose an instrument, find a private teacher, learn music theory, play in orchestras, give recitals, have auditions, and practice for many hours daily. This experience often produces skilled musicians, but at a cost. Young people who enter (or attempt to enter) the music world this way frequently have negative experiences involving everything from high barriers to entry to performance anxiety. These feelings of stress and inaccessibility associated with music education can and have driven children away from learning an instrument, or have stopped them from ever being able to try.[7]

Another big issue in music education today is the unequal access to resources. Many lower-income school districts do not have in-school music lessons[8], and private music lessons can be very expensive. The costs of renting or purchasing instruments, which often need to be replaced as a child grows, are also expensive (around $300 for a year of violin rental, for example) and not accessible to many. Cheaper instruments, like the infamous recorder many students had to learn in elementary music classes, have a much lower ceiling of mastery, poorer sound quality, and are generally less versatile - and so will be outgrown quickly. By creating a cheaper instrument you can learn from and grow with at home, you allow a greater audience to access music education. Another issue is space. Many kids have living arrangements that do not allow them to make a lot of sound, or do not offer the physical space to store or practice an instrument. For example, if a child is sharing a room with several other people, if a parent works from home all day, or if they live in an apartment building, they may not be able to make a lot of sound without bothering someone or getting in trouble. So again, there is a need for instruments that can be learned at home, with headphones or another device that makes it easier to take up less space (both physically and auditorily) .

Performance anxiety is also present in child musicians from a very young age[9], causing mental and physical reactions which deplete confidence and deter the musician from continuing in their practice. Anxiety levels have been shown to decrease with performance experience[9], but repeated bad or stressful experiences can also make young people think that they are simply not good enough to be a musician. Stressful performance experiences can also change depending on the audience involved - music teachers, for example, can cause higher levels of stress.[10] Many people who experience performance anxiety will feel perfectly comfortable playing or practicing on their own, when those who might judge you cannot hear what you are playing. These associations of stress and performance anxiety can also persist through adolescence into adulthood[10], again deterring musicians from continuing their practice. This creates a culture in the music world that gives young people the mindset that they need to excel at music if they are going to play it at all - which should not have to be true. Everyone should be able to experience playing a musical instrument just for fun - even if they are not perfect at it. Throughout this thesis, I will address these issues through the creation of the *Squishy* - a low-cost, low-floor, low-anxiety way for anyone to learn electronic music within your own home.

2

Related Work

Similar Instruments

Some of the first music toy projects out of the Opera of the Future group were tied to the Toy Symphony[11] project in 2002. These are the Music Shapers, created by Maggie Orth, and the Beatbugs, which were created by Gil Weinberg. The Music Shapers are soft handheld orbs, with shells embroidered with conductive thread that users could touch to create sounds. These instruments are designed to introduce children to some of the more high-level aspects of music, rather than just creating a string of pitches, which is something they may not otherwise be able to experience in traditional music education. The Music Shapers are used in a group, where each toy and it's user can control several different high-level aspects of an existing piece of music - for example one user could be controlling melody with a Shaper, while another mixes in audio filters for the same piece of music.[12] The Music Shapers occurred at a time before microcontrollers like the Arduino were widely accessible, and so were limited by technology in some ways. The Shapers were effective at introducing children to high-level music concepts, but were not necessarily as intriguing as stand-alone instruments, or for long-term use.

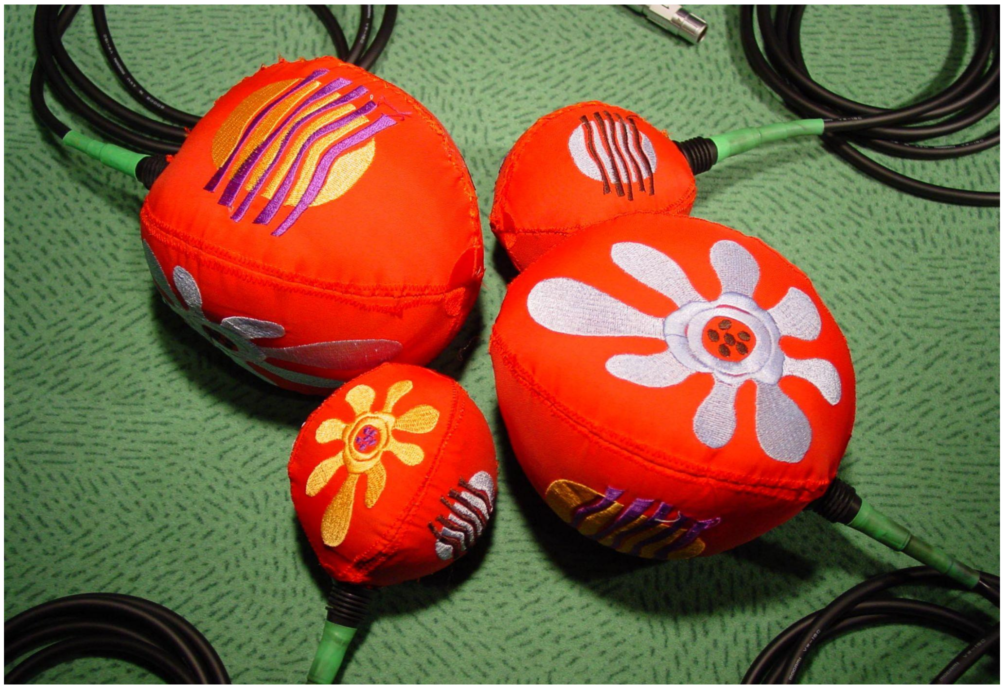


Figure 3: Four of the Music Shapers, displaying the different patterns of conductive embroidery. These were the instruments used in Tod Machover's *Toy Symphony*.[12]

The Beatbugs are percussion instruments designed to let users share rhythms with each other through a connected network of the toys. Children can play these in groups, creating motifs by tapping on the toy, or they can manipulate the pitch and timbre of the rhythms made by others by bending the antennae on the toys.[13] Much like the Music Shapers, these toys were limited somewhat by the technology of the era - they required extensive routing and computing power to run, whereas today, these toys could become standalone objects. The Beatbugs were effective in creating a shared music experience for kids through their networked connection, but did fall short as a standalone or educational instrument someone could stick with for a long time. That being said, as additions to the Toy Symphony and as performance objects, the Beatbugs and the Music Shapers achieved their project goals completely.

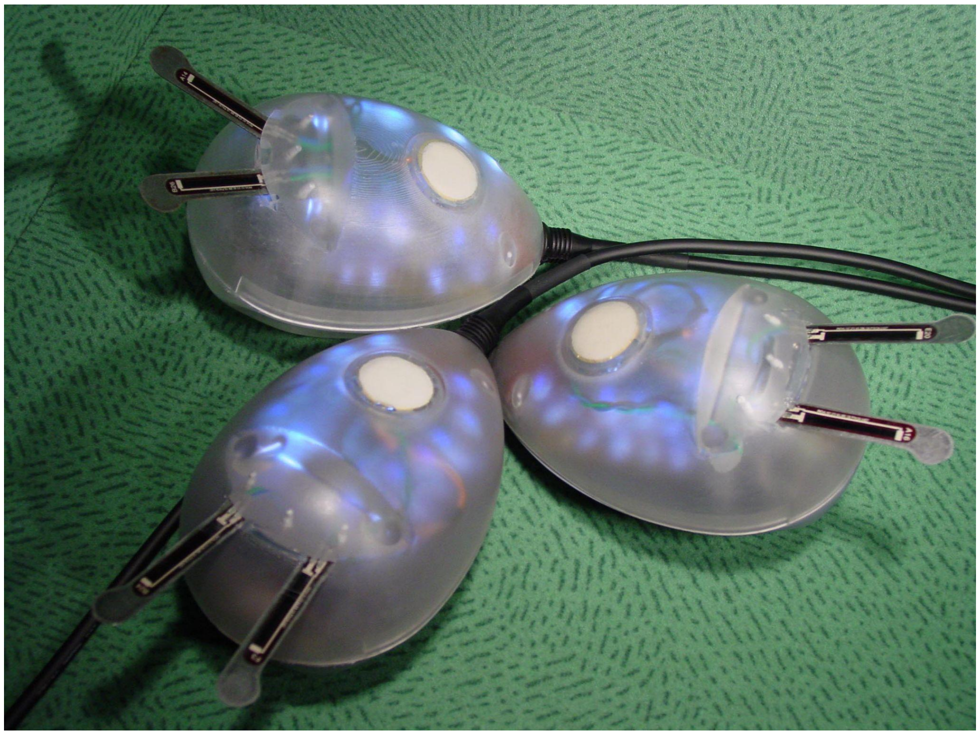


Figure 4: Three Beatbugs with translucent outer shells that show where a user can input percussive beats on the body, and where they can shape the sound they create, with the antennae.[13]

The next music project aimed at a younger audience from the Opera of the Future group was Hyperscore, which also came around the time of Toy Symphony in 2002, and was created by Mary Farbood and Egon Pasztor. Hyperscore is a music software that lets children with no music background compose pieces through freehand drawing. Their line drawings are then color-coded to determine different aspects of the music and notated with chords or other sounds to create a full piece.[14] Hyperscore was incredibly effective at introducing kids to the possibilities of composition, but existed only in the virtual world - there was no physical instrument to go with it.

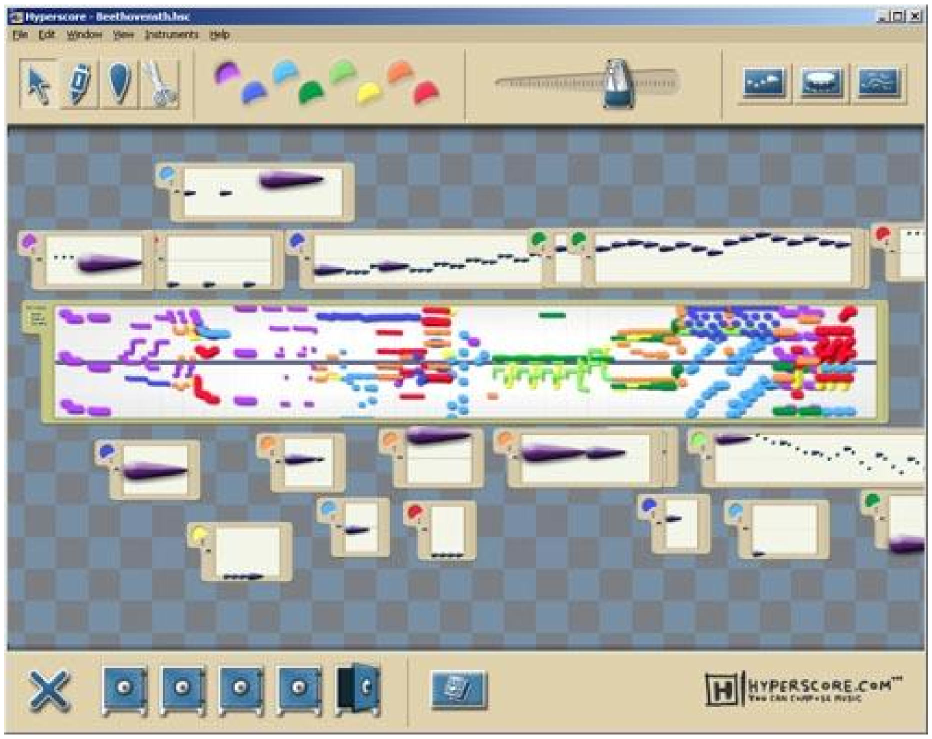


Figure 5: An example of a Hyperscore piece, showing the user's line drawings, color coding, and the different notations they used to create a piece.[14]

The Reactable is a musical experience that has been developed by S. Jordà, G. Geiger, M. Kaltenbrunner and M. Alonso from 2003 onwards. It allows users to manipulate sounds by moving physical blocks which rest on the surface of the table. These blocks can be moved, connected, or twisted, to create different sonic effects. The sounds are also visualized graphically on the surface of the table.[15] The Reactable is either used as an interactive installation in a space, or as a tool for DJs or other electronic musicians to perform live with. In both cases, the Reactable takes the form of a large pedestal with a screen on top where the user can manipulate sound blocks. The Reactable, though interesting as an installation, is not really designed for individual experimentation or musical growth.

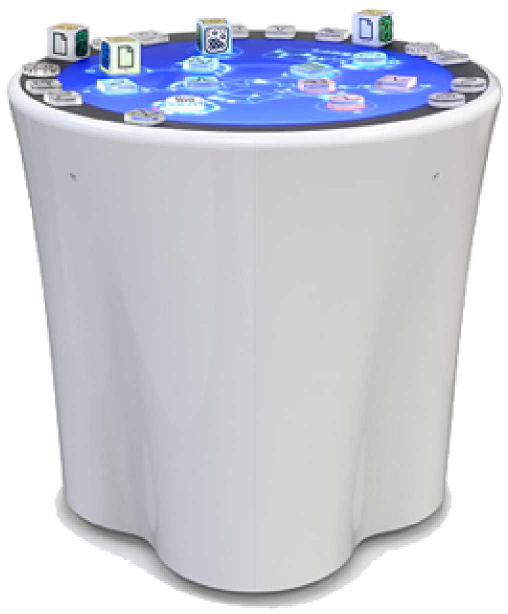


Figure 6: The installation version of the Reactable, showing the display screen on the table's surface, with several sound blocks placed on and off the controller.[15]

The Otamatone is an example of another soft instrument with unusual interaction methods. It takes the form of a large eighth note with eyes and a mouth on the base, the latter of which will open as you squeeze it to produce sound. You can change the pitches you create with the sliding bar on the stem by placing your fingers on it as you might on a recorder. The Otamatone is advertised most accurately as the "world's cutest and weirdest musical instrument."[16] Much like the theremin, the Otamatone is not easily mastered as an instrument, and is not necessarily designed to be played seriously, but is quite fun to play with even if you lack musical knowledge. It was designed by the CUBE toy company and the Maywa Denki design firm in 2009.

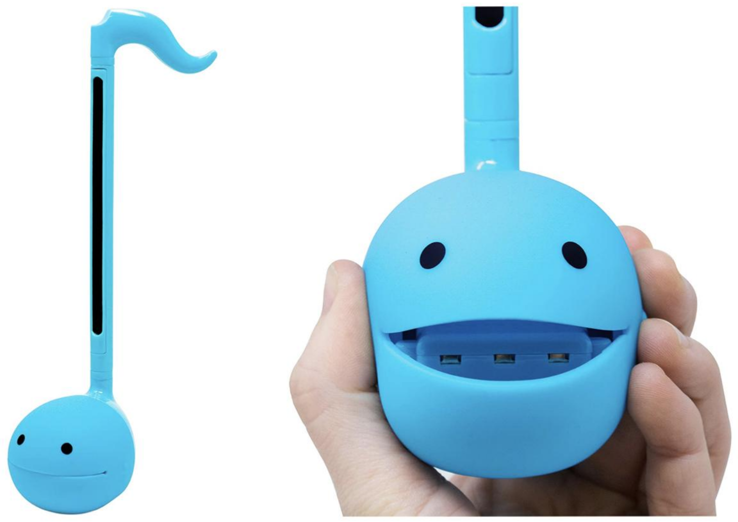


Figure 7: The Otamatone: to play, the black sensor on the staff is touched with one hand, and the silicone mouth is squeezed with the other.[16]

Another example of a soft musical instrument is Skoogmusic, which was created by David Skulina and Ben Schögler in 2009. Skoog is a soft cube with one interactive half-orb on each side. It is a music interface and toy designed to be more accessible than traditional instruments. When users tap on one of the sensitive areas of the cube, they trigger sound bytes that can be changed in an iPad app. Skoog has created an accessible instrument that can be played by most, though it falls short of becoming a lifelong musical companion. Beyond simple apps aimed at kids, it seems it could be outgrown somewhat easily. It also comes with a large price tag (about $200) for what it is, especially considering you also need an iPad to operate it.[17] Skoog and it's smaller counterpart Skwitch have made a good start in creating musical interfaces that are more accessible to people of different abilities, but could have done even more with their interfaces to make them more exciting for both low-level and high-level users. They have also missed the mark a bit in utilizing their soft interface. Each sensitive area seems to act just as a button with sustain, but given that it is a soft interface, they could have used the squeeze-able aspects to their advantage by making each area bending or pressure sensitive as well. This would have allowed for the use of more controllable parameters, and would have made an overall more interesting interface.

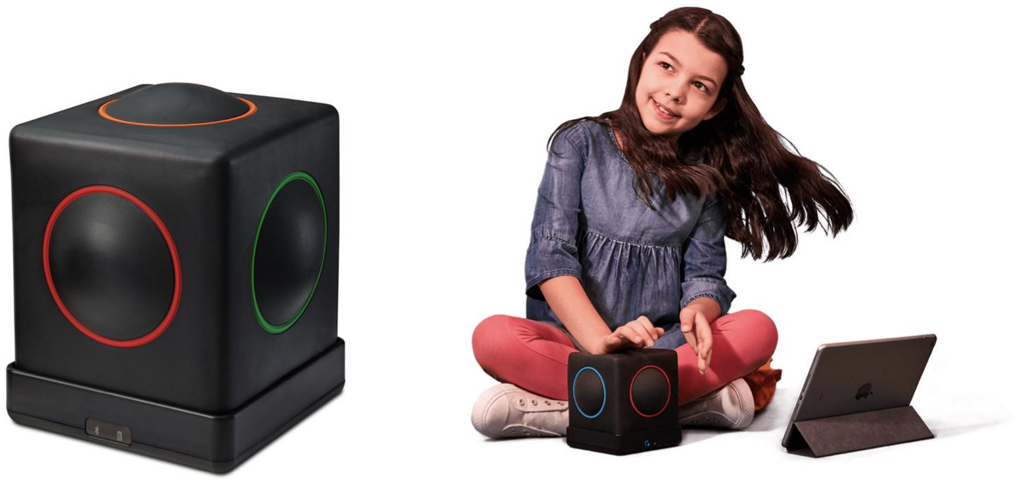


Figure 8: The Skoog instrument, shown with a user interacting with it - the circles on each side represent the sensitive areas.[17]

Blipblox is a more recent instrument aimed at children - it is quite literally a synthesizer with modified controls that make it kid-friendly. Blipblox , which was created by Playtime Engineering in 2018, is unique in that it is designed for kids as young as 3, but still maintains more advanced musical ideas. This makes it an amazing tool for both education, and general musical exploration at a young age.[18] Blipblox does however maintain a lot of the typical affordances of a regular synthesizer - you can push, turn or move the built in controls to manipulate sounds. Apart from making the knobs and buttons larger to keep them kid-friendly, they did not stray far from the norm in terms of synthesizer design. Blipblox is limited somewhat in the sound quality it produces - it has a very low-fi, choppy sound that is not necessarily what an advanced user would want from their synthesizer - but somehow I think a 5 year old would not mind as much.

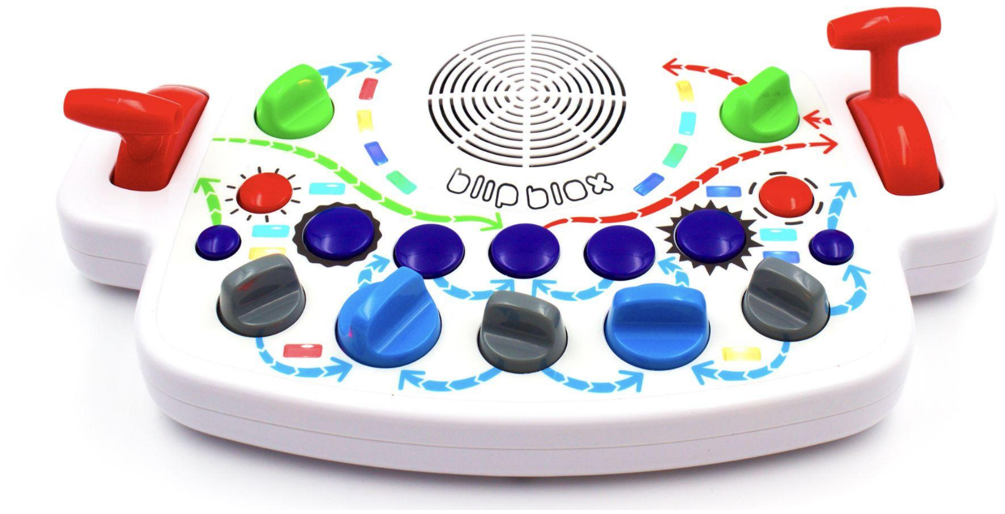


Figure 9: The Blipblox synthesizer for kids, with larger buttons and knobs designed for little hands.[18]

There are a few other unusual synthesizers and controllers I want to mention just for their spectacular unusual-ness. One is the Minibar Liquid Analyzer, which is a guitar pedal that has a spot for you to put any kind of liquid you might have access to. The overdrive and tone created will change depending on what liquid you put in it - the opacity of the liquid affects the treble and bass, and the conductivity of it affects the amount of gain.[19] The Liquid Analyzer is interesting, though it comes off as slightly gimmicky, as users may lose interest in it after a short while. It also strikes me as being somewhat easy to get dirty and hard to clean, which may hinder its effectiveness after using a particularly sticky or otherwise messy liquid.

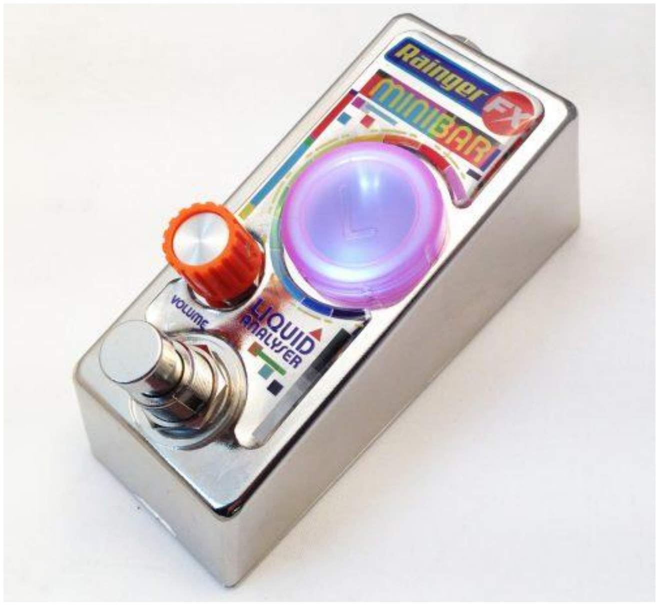


Figure 10: The Minibar Liquid Analyzer: the large pink circle is where one would place liquid.[19]

Another unusual synthesizer is the Thingamagoop from Bleep Labs which is lovingly described as “the ultimate synth / effect / noise friend.”[20] The Thingamagoop is a light-controlled synthesizer with a robot-like form factor that was originally produced in 2006.[16] The simplicity of the interface and it’s sounds make the Thingamagoop an easy instrument to pick up and start using - since it has limited functionality, it has a relatively low-floor to entry. The small number of interactions do also give the Thingamagoop a somewhat limited long-term play value, meaning there is no real mastry to be gained from it.

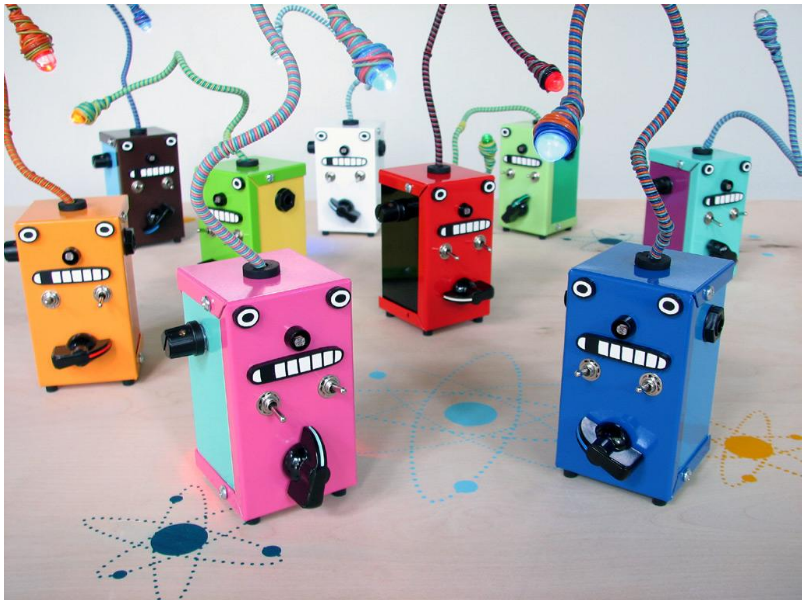


Figure 11: The original Thingamagoop, produced in 2006.[21]

Educational and DIY Experiences

Scratch comes to mind when many people think of educational technology for children. It is a great example of taking something complicated (coding) and presenting it in a clean, easy to understand format that young people can learn from. Scratch does have a music add-on that kids can use to incorporate sounds into their projects. This interface will let you select different sound samples, manually change the tempo and pitch of the samples, and record your own short sounds.[22] These features provide enough music variety that adding a soundtrack to your game or other interface is easy, though as a music education tool it is somewhat lacking since you can only control a few limited parameters. If Scratch were to expand it's music section, it would benefit from more options for composition or maybe the addition of allowing users to connect MIDI instruments to Scratch. That being said, I do not necessarily think intense music education was the goal of the Scratch team in adding music features.

The Makey Makey is a board that lets you connect different conductive objects to it to create controllers out of everyday objects - everything from bananas to graphite pencil drawings. Makey Makey works with Scratch and other similar interfaces - it was also developed by the [Lifelong Kindergarten](#) group at the MIT Media Lab, by Jay Silver and Eric Rosenbaum in 2012 - so users can utilize it as a game controller or a MIDI instrument, depending on the app they want to use.[23] The appeal (and the overall project goal) of the Makey Makey though lies more in the aspects of creating DIY interfaces rather than in creating playable instruments or novel controllers. Because of it's setup with Scratch-like apps, Makey Makey is also somewhat limited in how much you can do with the interfaces you create with it. Unless you are a software developer, you can only use the premade apps, of which there are few. This makes it harder to innovate or fully explore the ideas you might have with the Makey Makey - especially the musical ones, as you end up being limited to pre-made piano and drum kit apps.

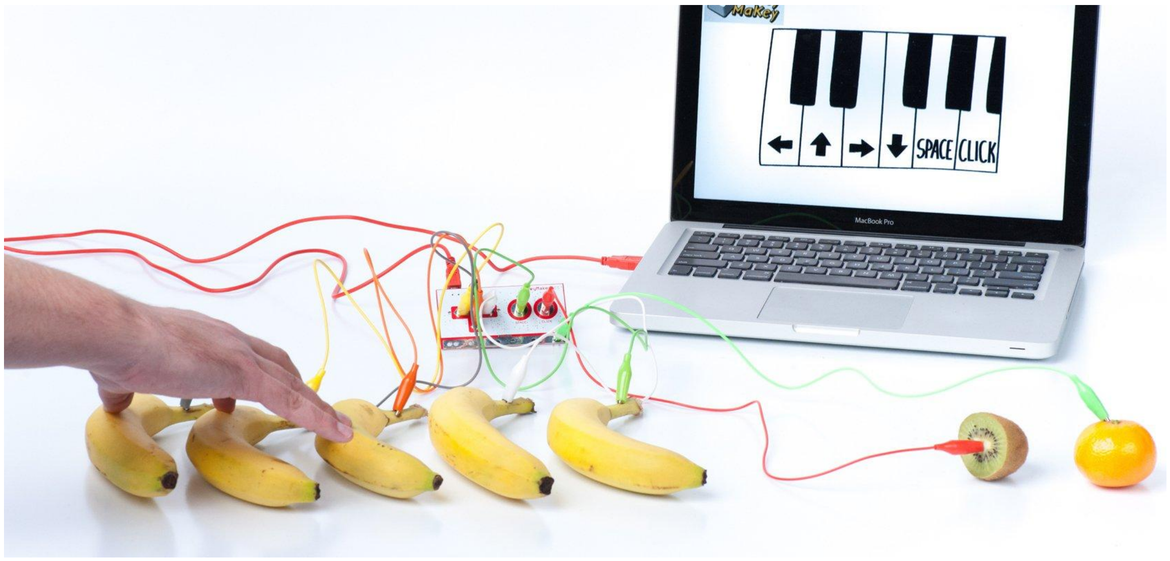


Figure 12: The Makey Makey, shown with a banana piano project, which hooks up fruit to the Makey Makey board using alligator clips, and then uses it to control a piano app.[23]

Drawidio is another project created by Jay Silver, in 2008. This interface consists of a circuit, battery, and small speaker mounted on a pencil, which you can use to "draw audio" by completing the graphite circuit with your fingers. Drawdio succeeds as a fun way to make silly sounds easily, and contains some DIY setup that teaches you how the system works as you do it.[24] The kits for assembly can be purchased on Adafruit for only $17.50,[25] making it an incredibly cheap way to create your own unique sound interface. Drawdio is another great example of a fun and easy interface that is not necessarily designed to be played or pursued seriously as an instrument.

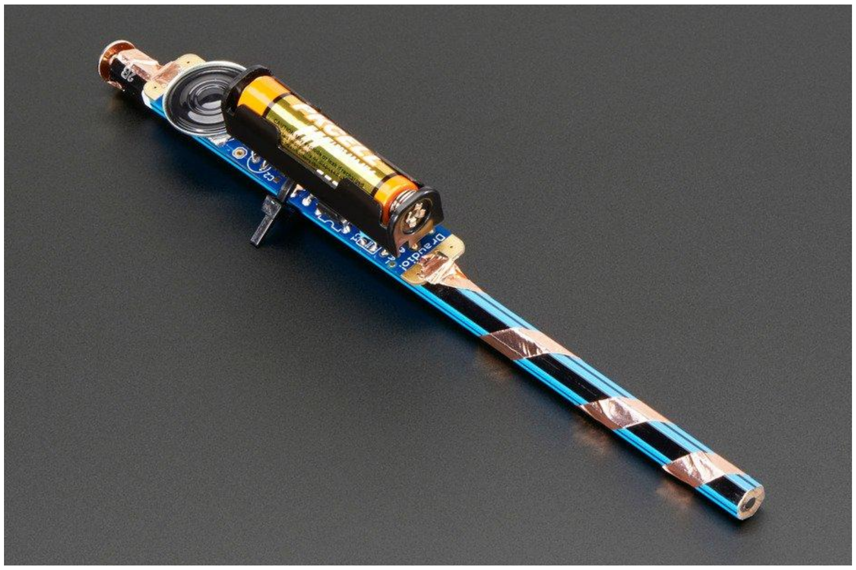


Figure 13: The fully assembled Drawdio, made using the kit from Adafruit.[25]

Another project I would like to talk about is Chibitronics, which is not music related, but is an interesting model for simple electronics education. Chibitronics makes circuit stickers, which can be paired with copper tape to create paper-based circuits. This project, created by Jie Qi, is an excellent example of a low-cost, accessible, educational experience. Chibitronics also has an extensive array of tutorials on their website, which provide instruction and inspiration for users and educators looking for projects to do with their circuit stickers.[26]

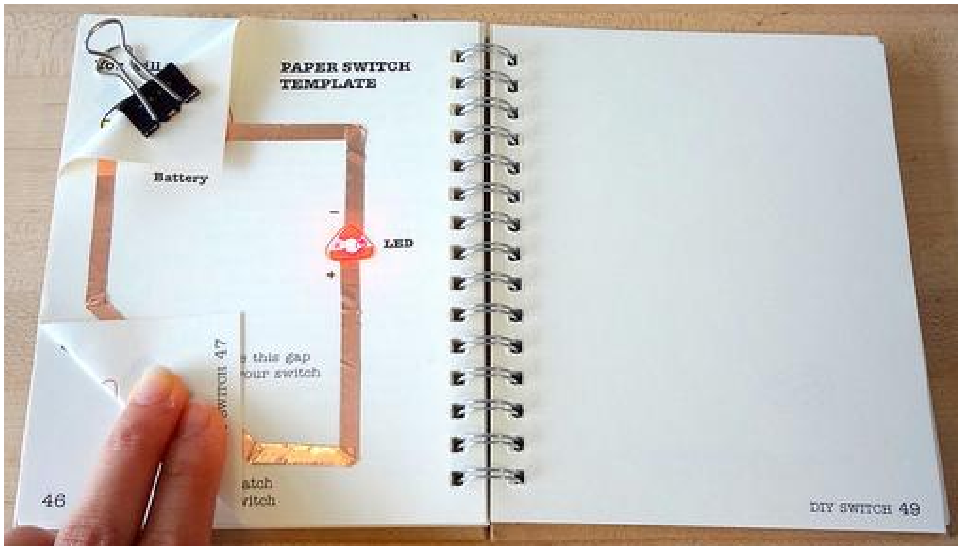


Figure 14: A sample project from the Chibitronics circuit sticker sketchbook, showing how to make an on/off switch using copper tape.[26]

Kobakant is also not directly music related, but is an incredible resource for those looking to learn about soft electronics. It was founded by Mika Satomi and Hannah Perner-Wilson in 2008, and contains a vast collection of tutorials and open-source information about e-textiles, soft sensors, and projects that can be completed with them.[27] I personally used their website heavily when considering how to put together the *Squishies*, and utilized their pressure sensor template for the first prototypes I created.

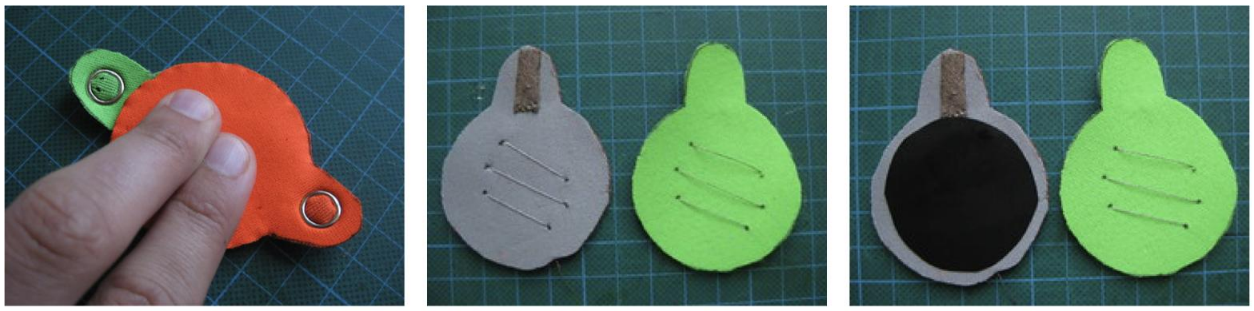


Figure 15: A fabric pressure sensor breakdown, created by Hannah Perner-Wilson for Kobakant.[27]

Interactive & Personal Robots

I would like to touch briefly on some examples of interactive or emotional robots, as I think they are great examples of objects in which people invest a great deal of emotion. There are some parallels between this kind of interaction, and those one might have with an instrument, or a meditation device - even if they have inherently different purposes. The first of these is Huggable - a robot designed for use in pediatric care settings, designed by Sooyeon Jeong in 2012. This animatronic bear looks like a normal stuffed animal, but can interact with and talk to children receiving care at the hospital to bring them comfort in an otherwise stressful situation.[28] Huggable is a great example of a personal care robot that is designed to promote emotional wellbeing in its users. It successfully promoted relaxation in children in a hospital situation - which is a very difficult setting for anyone.

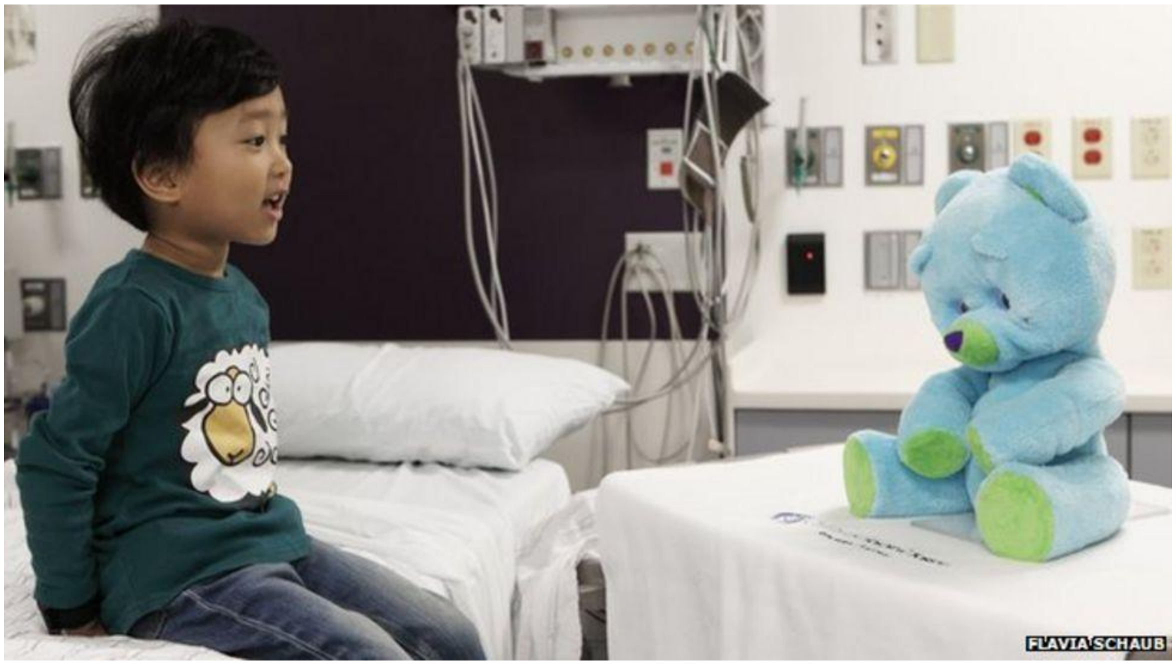


Figure 16: Huggable interacting with a child in a hospital setting.[28]

MyKeepon is the toy version of the Keepon robot, created by BeatBots in 2011. Keepon is a soft robot designed to respond to ambient music through bouncing dance moves that correspond to the beat. It also has a touch mode, where it will respond to a user's interactions like petting, poking, or squeezing. The original Keepon robot was designed as a musical companion for children with autism. MyKeepon lacks some of the more sensitive aspects of the original Keepon - which was primarily created for research - but adds the option of being hack-able with the addition of a microcontroller like an Arduino. The creators of MyKeepon even made the Arduino code available to all on GitHub, so anyone can control their Keepon by programming it with a computer.[29] Keepon is a fantastic example of a physical object that successfully forms a relationship with its users. The adorable sounds Keepon creates upon interaction completely draw you in, and make you feel like Keepon is sensing your behaviors beyond what the toy version is programmed to do. It's appeal though does start and end with the cute factor - Keepon does not have enough options for interaction to keep you drawn in for extended periods of time - in other words, it gets boring after a while. Keepon would benefit from more options for play - maybe something like a game you could play with it, or the option to use Keepon as a musical instrument.

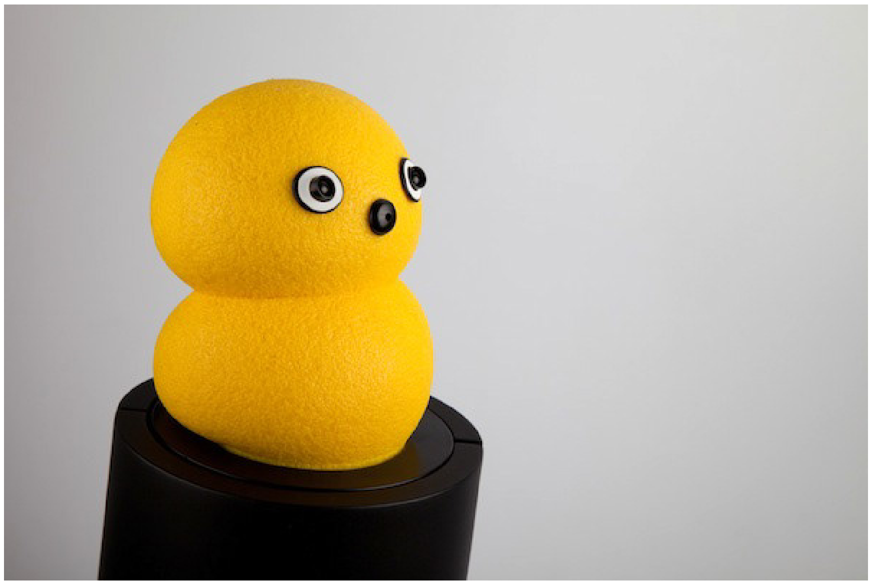


Figure 17: The MyKeepon robot, which will dance to music by bouncing up, down, and side to side.[29]

Ploomi is another robot created by Beatbots, in 2011. Ploomi is an interactive and touch-sensitive jellyfish-like creature. It will light up, make sounds, and vibrate in response to interactions.[30] I bring up Ploomi because I think Beatbots missed a fantastic opportunity to make Ploomi soft and squishy. If it were a soft robot, you could have introduced more opportunities for interactions and subsequent responses from the robot itself. Apart from that, I think Ploomi is an interesting contrast to Keepon, as it creates sound rather than responding to it.

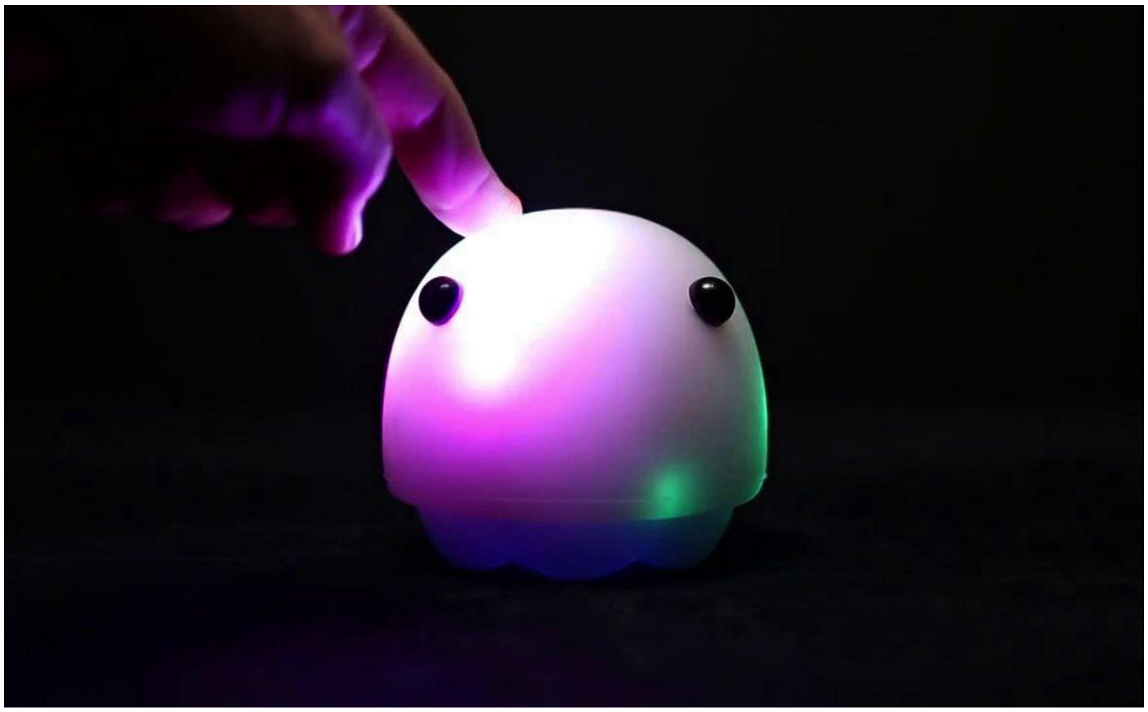


Figure 18: The Ploomi robot lighting up as a user interacts with it.[30]

Pedagogy

It is also important to go over some of the other methods for music education that have been developed in the past. Many of these - like the Suzuki method[31] or YOLA[32] (Youth Orchestra Los Angeles) - focus on western classical music education while some take a more modern approach to youth music education, like the Orff Schulwerk[33] method. I would also like to touch on the laptop orchestra model, which gives a perspective on electronic music education and creation.

YOLA is an opportunity for young people to learn western classical music through free music lessons, instruments, and ensemble experiences. It is aimed specifically at kids who might not otherwise have access to music education, and gives them the chance to join a nurturing community of students from similar backgrounds, and educators who are excited to help them grow.[32] YOLA and similar programs are fantastic examples of musical outreach, and models of education that help get music education into communities where it would not otherwise be. These programs do however start and end with western classical music education, and could potentially be expanded into other areas.

The Suzuki method is a means for children to learn a traditional western instrument, like the violin, viola, or cello. It centers around the idea that given the right circumstances, anyone can learn music. Suzuki utilizes repetition and imitation to teach people music concepts - it is inspired by the way young children learn languages. It takes users through a series of books with increasingly advanced pieces, introducing new techniques as they arise.[34] This is how I learned the violin, starting from the time I was four until I was about fifteen when I decided to switch out. Suzuki is very effective in teaching children music - speaking from experience I feel like I did learn a lot from it throughout my childhood. That being said, it does limit students' autonomy on some level - kids do not really get a choice in what they learn next, and since everything is leveled so carefully, they also do not get the chance to aim high, or choose more advanced pieces earlier on to have something to rise to. Suzuki also stops students from being able to interpret music in their own ways - they are pushed so far into imitation and repetition that kids do not get the chance to create their own musical ideas within pieces. These are all reasons students will often switch out of the Suzuki method as they get older - so they can be more creative and have more autonomy within their musical lives.

The Orff Schulwerk method takes a slightly different approach than Suzuki or YOLA. Instead of teaching kids a specific instrument, Orff Schulwerk encourages active music making, which utilizes movement, speech, singing, and instrument playing to get kids comfortable with music. Their philosophy is that kids will learn more by actively creating and improvising than by only studying, and that the skills they pick up through the Orff Schulwerk method will carry into other aspects of their lives. During lessons, kids will complete a series of exercises that let them explore a musical space - first by imitation and then by improvisation.[35] Orff Schulwerk is in this way a more well-rounded method for music education than some others, as it gives children autonomy over what they learn and do with music through improvisation, and allows them to gain confidence in their musical abilities. However, it does not necessarily allow you to perfect a specific instrument as a method like Suzuki would.

Laptop orchestras are a relatively new construct - the first one was created at Princeton in 2005.[36] They aim to make electronic music more of a community experience. Often when one thinks of a computer, they do not think of something you would use with another person, let alone a lot of other people, but within a laptop orchestra you can make this happen. Laptop orchestras also include unique aspects of live instrument creation through code, which is not something you will find in most other ensemble settings.[36] One flaw with laptop orchestras is that they are not necessarily accessible to a lot of people - you have to know how to code with enough skill that you can do it live, and you need to have enough previous musical knowledge to know how to fit yourself into an ensemble performance. Another shortcoming is the fact that playing a laptop does not necessarily feel like playing an instrument - it does not give the same physical interactive aspects that you would get from playing most instruments, and so is not as satisfying on some level. These ensembles are interesting methods for creating group performances of electronic music, but they fall short as educational experiences.

Prior Work

The *Oniorb* is an explorable musical object designed as a tool for stress relief and music therapy. I developed it for my Mechanical Engineering Bachelor's thesis, which was also advised by Tod Machover. I have always used music as an outlet for stress and anxiety in my life, and I wanted to be able to give that experience of release through music-making to other people who could not play an instrument. I created the *Oniorb* as a way to do this - it is easy for anyone to operate and allows you to have a relaxing musical experience. The *Oniorb* also has a stress-ball like shape, and is operated with headphones, so it provides an intimate and meditative experience with the instrument.[37] You can find a video of the *Oniorb* in action here.

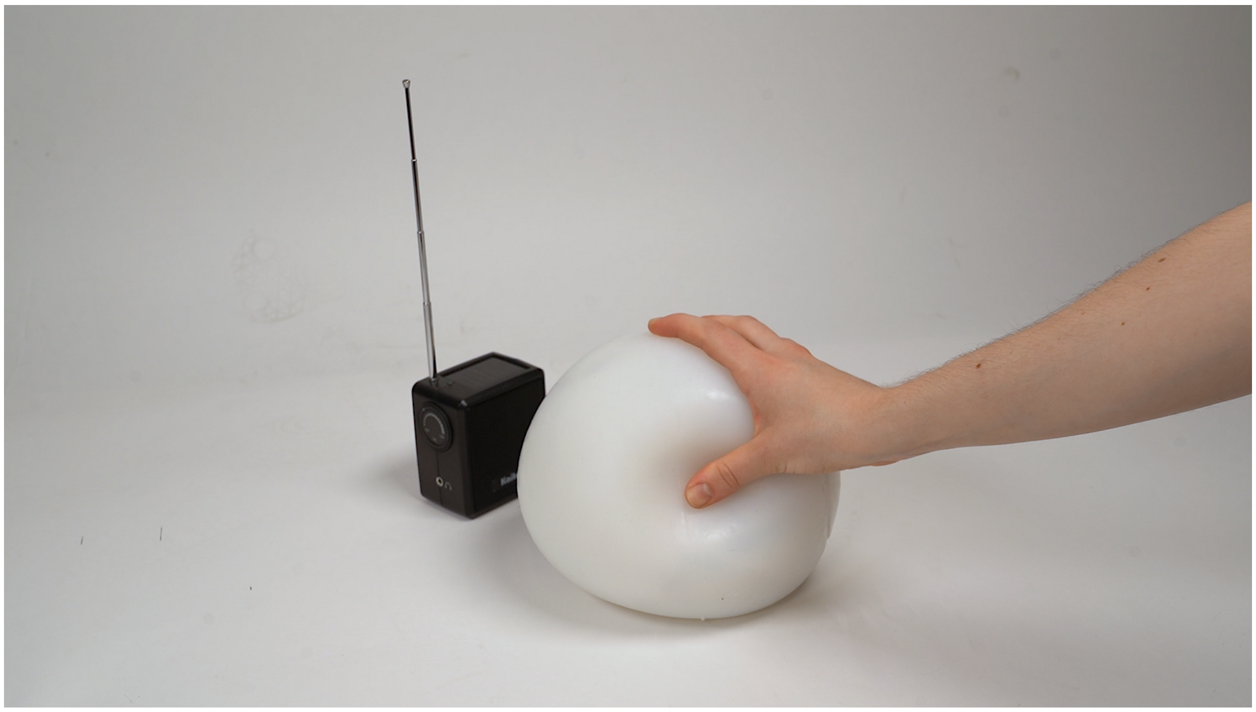


Figure 19: The *Oniorb* as a user interacts with it.[37]

Early in 2020, I collaborated on a project with Kia with several other members of the Opera of the Future group - Nikhil Singh, Karsten Schuhl, and Manaswi Mishra. This project centered around the future of automation - specifically self-driving cars, and what we might spend time doing in the car once we do not have to drive. Kia created an interface that allowed passengers to compose music while in the car - it was designed such that anyone, regardless of musical background, would be able to use it. They asked our group to suggest a physical interface that might add itself to their software nicely. We suggested the ORB, a palm-sized ceramic orb which would vibrate in response to your voice or other audio input, which was created by Rébecca Kleinberger and others.[38] The ORB was installed in the final exhibition setup, as an interaction device for those sitting in the backseat of the car - they would be able to feel the vibrations from the piece those in the front were composing. The Oniorb was also suggested as an interface, which may have also been interesting as a way to input something to the interface through physical touch. This project inspired me to think more about what other objects could become sonic and/or squishy.

Another project I worked on during my Master's also involves squishy music, but in a slightly different way. Like many people, I spent a lot of time during COVID-19 learning how to make bread and other baked goods. For a class project, we were asked to think about a new kind of music - I naturally thought about bread, since it was what filled about 90% of my free time during quarantine. Freshly baked bread makes lovely cracking sounds right when it comes out of the oven. My mom calls it 'singing.' I love it. So I started to collect recordings of my baked goods as they came out of the oven, and then I slowly started recording other parts of the bread-making process. For my midterm project, I compiled all of these recordings, along with some bread doodles I had been making, into a short animation about the bread-making process.

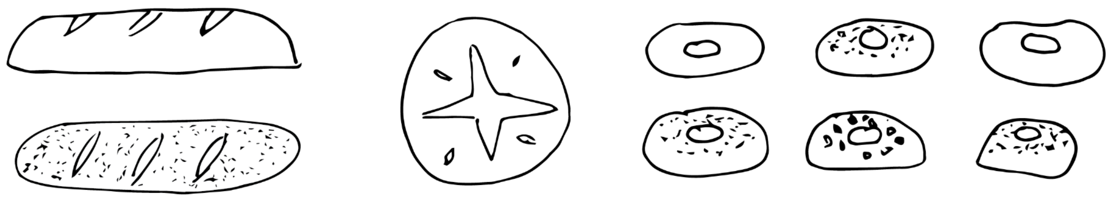


Figure 20: Some of the bread sketches used for the *Bread Sounds* animation.

For my final project for the same class, I wanted to expand on the bread sounds I had collected before. Since I was thinking about soft sensors and interfaces, and because the process of making and working with bread dough is not that different from the process of playing a squishy instrument, I decided to make a soft bread instrument. This interface would play the sounds of bread-making if you interacted with it in the corresponding way. For example, kneading the instrument would result in the slapping, squishing sounds that kneading a loaf of bread would result in. The loaf instrument also played the sounds of bread 'singing' if you stroked the crosses on top of the interface, which was made with conductive thread to make it a capacitive sensor. You can find a video here.

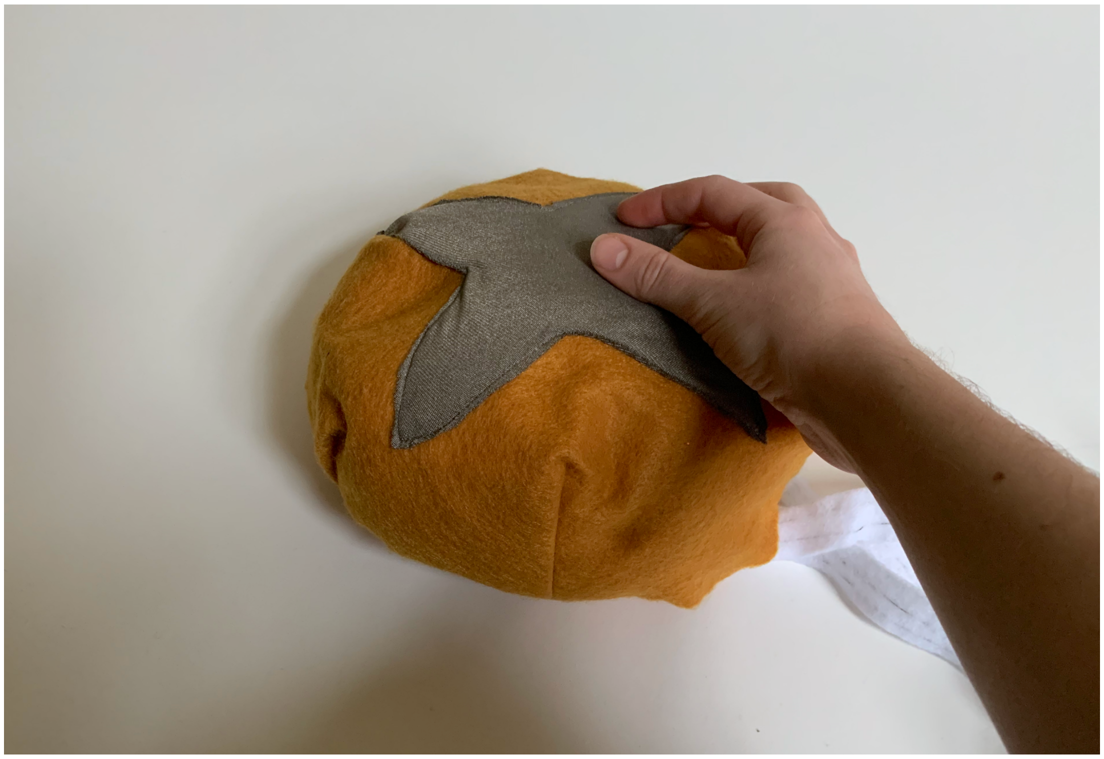


Figure 21: The loaf interface I made for my collected bread sounds, shown with a user stroking the capacitive sensor on the top.

3

The Squishies

Throughout this section, I would like to highlight some of the ways in which the *Squishies* address the issues laid out in the introduction, and how they are set apart from or outperform related prior work.

The *Squishies* allow a wider audience to experience music at a young age. Most music instruments are not accessible to people of different physical abilities. Many require extensive fine motor movement to create any sound at all. The *Squishies* were designed to be played by almost anyone - their soft construction eliminates the need for extensive training or hand-eye coordination, so they are easy for almost anyone to handle. They are not one-sided like many string instruments, meaning there is not a right or left handed version, which you often see in guitars. The *Squishies* can also be played with only one hand, or can be adapted to be played with other body parts.

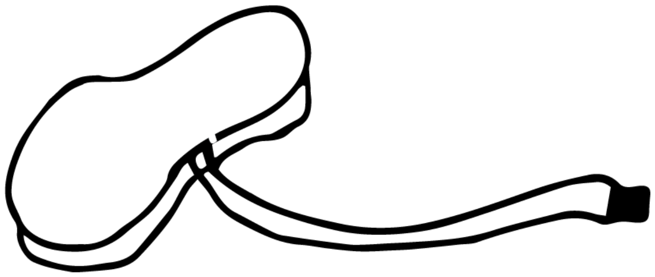


Figure 22: A concept sketch of a *Squishy* toy.

The *Squishies* are also relatively low-cost, especially in comparison to other small MIDI devices that are marketed to beginners. Hardware like this has only been getting more accessible, as computers are cheaper and much more common than they were 5-10 years ago. For example, the most recent Raspberry Pi - the Raspberry Pi 400[39] - is a fully functional computer, and costs only $100. Companies like Arduino and Teensy have also made microcontrollers much more accessible to a wider range of people, while online stores like Adafruit have made all kinds of prototyping electronics widely available. The rise of these low-cost controllers and computers is part of what has made *Squishies* possible, and why something like this was not entirely possible until now.

This project also aims to fill a gap in music education as an introduction to electronic XXth century music-making, which is a space that has not been fully explored. Today, even widely accessible products like GarageBand still have a decently high barrier to entry because of factors like high learning curves and expensive hardware. The *Squishies* fill this gap, by making an interface that walks users through basic electronic music creation by learning about music samples, beat changes, and pitch shifting, while staying accessible and low-cost. In doing this, the *Squishies* create a low floor for entry into the electronic music world.

Another unique aspect of these squishy instruments is that they are able to grow with their users. Beginner users are able to learn basic concepts through the *Squishies*, and acquaint themselves with simple electronic music concepts. After they get acquainted with the *Squishy* and are more comfortable with music, there is an option to pair the Squishy interface with a more advanced music software such as MAX/MSP, as it will register as a MIDI device when plugged into a computer. In this way, the *Squishies* also maintain a high ceiling for users, meaning they are able to continue growing their skills with the toys to become experts without getting bored after a short amount of time. This high-ceiling also applies to already experienced electronic musicians, who could use the *Squishy* as a MIDI controller with any preferred music software or sound library. The *Squishy* could even become a tool for DJs to use for fluidly mixing samples in unique ways that could not be as easily accomplished with existing tools.

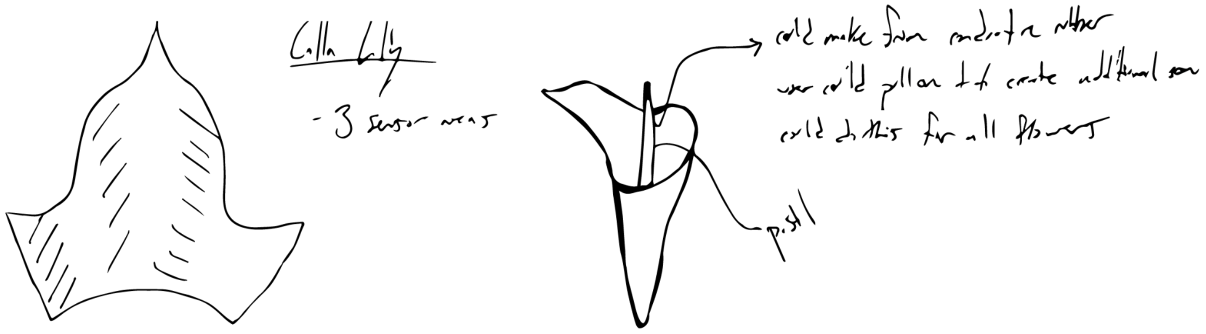

Calla Lily
- 3 sensor areas
could make from conductive rubber
user could pull on it to create additional sound
could do this for all flowers
pistil

Figure 23: Concept sketch of a calla-lily shaped *Squishy* toy.

The *Squishies* can also be utilized as tools for guided meditation when paired with the right soundscapes. Users can complete breathing or concentration exercises with toys by focusing on creating one specific sound or matching their breathing pace to the rate at which they squeeze the toy. The *Squishies* can also be guides for simply being mindful of the subtle movements you can make with your hands, and what effect those small motions are having on the sounds being created. These actions are not possible with other musical instruments, as most do not have a soft enough form factor, or calming enough sounds, to be effective mindfulness tools. A stress ball or similar artifact would also not provide quite the response that a *Squishy* would through meditation exercises - there is no direct response to your physical action.

Through all of these factors, the *Squishies* provide an opportunity for kids who do not have a formal music background to experience the stress-relieving aspects of playing an instrument. They do this while staying more accessible to a larger group of people than most traditional instruments. The *Squishies* also allow for a low barrier to entry into the electronic music world, can be utilized by more advanced users as MIDI controllers, and can even provide a tool for meditation for users of any age or skill level.

Product	Cost	Long Term Play Value	Is an Instrument	Educational	Is Squishy
The *Squishies*	$35, estimated	✓	✓	✓	✓
Music Shapers	N/A	✕	✓	✓	✓
Beatbugs	N/A	✕	✓	✓	✕
Hyperscore	Free	✓	✕	✓	✕
Reactable	N/A	✕	✓	✕	✕
Otamatone	$35[16]	✕	✓	✕	✓
Skoog	$200[17]	✕	✓	✓	✓
Blipblox	$189[18]	✓	✓	✓	✕
Minibar Liquid Analyzer	$149[19]	✓	✕	✕	✕
Thingamagoop	$250[21]	✕	✓	✕	✕
Garageband	Free	✓	✕	✕	✕

Table 1: Comparison of the *Squishies* to similar instruments.

Method	Flexible Program	Accessible	Teaches An Instrument	Low Floor	Teaches Electronic Music
The *Squishies*	✓	✓	✓	✓	✓
YOLA	✕	✓	✓	✓	✕
Suzuki	✕	✕	✓	✓	✕
Orff Schulwerk	✓	✕	✕	✓	✕
Laptop Orchestra	✓	✕	✕	✕	✓

Table 2: Comparison of the *Squishies* to other musical pedagogies.

4

Approach

When I started this project, we had all just been sent home to start quarantine due to COVID-19. I chose to go home to my parents' house, in a suburb of Boston, as it seemed more appealing to go somewhere with a backyard if we were not allowed to go anywhere else. Returning to this place made me inclined to return to hobbies I had when I lived there as a child - mainly, sewing. Slow-paced hobbies like this were always a part of my younger life, but as I left for undergrad and dove headfirst into the chaotic world of MIT, I could not let myself do anything that was not immediately rewarding. It was hard to slow down for any amount of time. Coming back to this slower way of life, and doing things very intentionally and by hand inspired the very beginnings of the prototyping process for this project.

Mechanical Design

Experimenting with sewing and fabric also lent itself to the soft structure I knew I wanted in the toys from the beginning. Soft and squishy interfaces encourage more fluid interactions than instruments that use buttons, strings, knobs, or other more traditional actuators. So, I began prototyping by selecting fabric from my old stash, thinking about what shapes I could make that might be interesting as instruments, and sewing soft shells of them to make them a reality.

Sensor Design

The beginning of my electronics exploration started with simple off-the-shelf bending and pressure sensors, which I was placing into the fabric shells as quick prototypes to test the basic functionality of different shapes. These sensors were connected to a breadboard with alligator clips, making for a generally clunky setup. The sensors themselves were too small to fill the desired sensitive areas of the shells, and were not soft enough to go unnoticed by the user. Hard electronics and wiring are also fragile - soldered connections will break with repeated stress, and sensors will be damaged. If my desired final form was to be fully malleable, I needed to move away from having rigid objects inside the instruments, as they would not only break, they would prevent users from feeling comfortable fully bending and exploring the interface. So from here, I began to research soft, fabric-based electronics, with the goal of finding sensors that could not be felt through the fabric shell by users.

This is when I discovered How to Get What You Want, a website created by Media Lab alumna Hannah Perner-Wilson, that has detailed instructions on how to make sensors with conductive fabric and thread. These all utilize the material Velostat, which has a variable resistance when pressure is applied. My first tests were just recreations of her bending sensor, which has conductive fabric patches on each end that can be connected to a circuit with alligator clips. From there, I started to adapt her designs to better fit into the shells I had made, and I ended up with something square shaped with two adjacent conductive fabric patches at one end to connect alligator clips to. At this point, I was able to have fully formed prototypes, as the sensors could be sewn into my shells, with extra long alligator clips exiting through a small hole on one side of each toy. All the sensors were sensitive to both pressure and bending forces on the outside of the toys, so any kind of squishing or squeezing of the toys would result in data output, as well as sound creation.

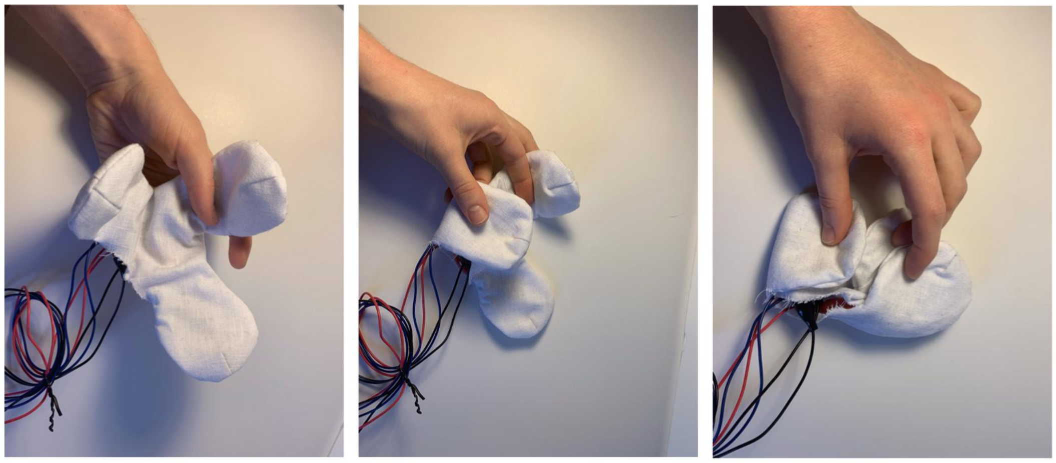


Figure 24: An early prototype of the *Squishy* toy.

Next, I started thinking about the *Squishies* as a potentially modular toy: something a user could take apart and put together in a different way, with sensors in different places or using a different shaped shell. I thought adding an aspect like this would make the toys more enticing as educational or DIY experiences, as users could have a hand in building the *Squishies* as well as playing with them. This idea led to the use of metal snaps as a means to connect the sensor to the shell. Snaps like this are generally low-profile, so they do not take up a lot of physical space, and they are also very common connectors for fabric objects, so they fit aesthetically with the rest of the toy's construction. By adding these, the user could potentially insert and remove sensors, or create different sensitive areas on the toy. The use of these snaps also enabled me to be able to move the alligator clips to the outer edge of the toy. Up to this point, clips were connected directly to the sensors, creating hard spots within the toys that limited movement and diminished the tactile experience. To accomplish this move of the clips, lines of conductive fabric were adhered to the fabric shells using fusible interfacing (a material which will melt under the heat of an iron to fuse two pieces of fabric). These lines ended along one edge of the shell, where clips could be connected without interfering as much with user interactions. One half of each of the metal snaps, which are conductive, was sewn onto the lines of conductive fabric to complete the circuit for each sensor.

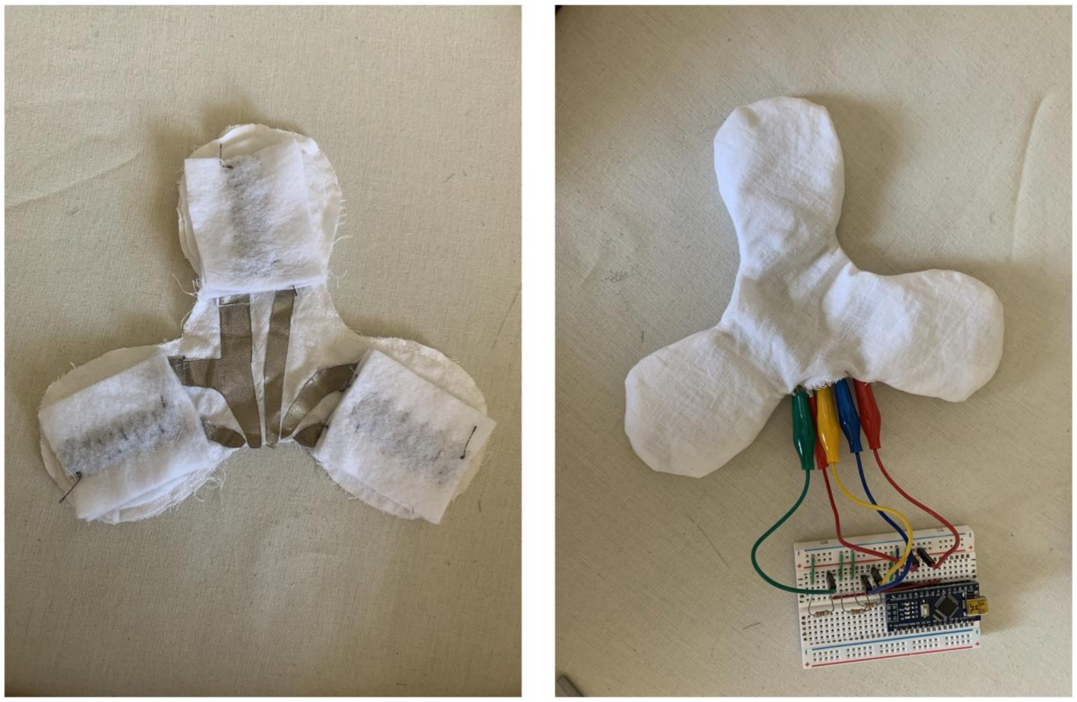


Figure 25: *Squishy* prototype with alligator clips moved to one edge.

The next goal I had with sensor prototyping was to eliminate the alligator clips completely. The first thing I tried in an effort to accomplish this was the Adafruit Flora board. The Flora is a microcontroller designed to be sewn onto clothing or other fabric - all the connection points have copper-lined holes that allow for conductive thread to be used in place of wires. After one prototype with the Flora, I decided it was not correct for my purposes. Though it would allow for an almost fully enclosed/self-contained instrument (as there would be no external circuit board), the board itself is rigid, so it was not tangibly invisible inside the fabric shell of the *Squishy*.

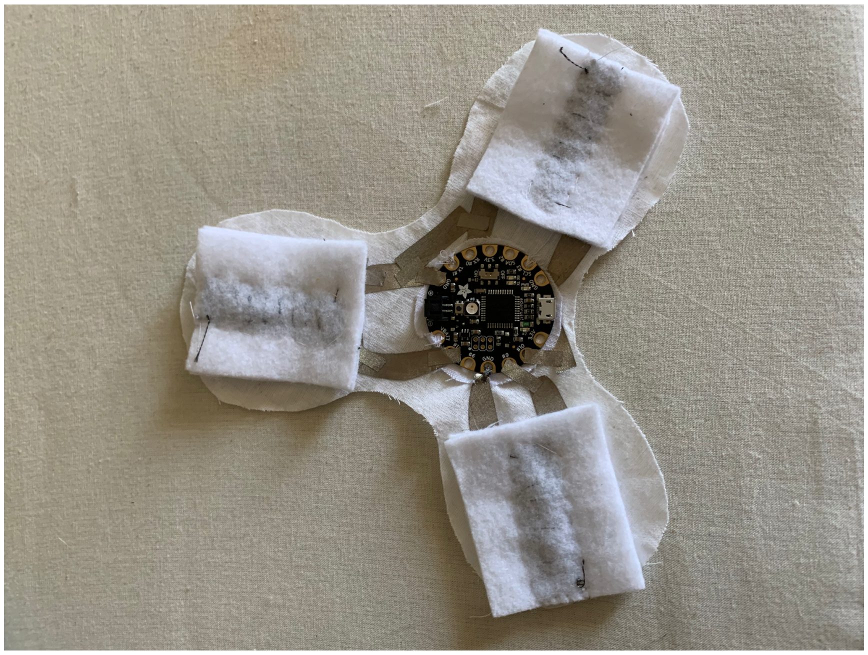


Figure 26: Prototype made with the sew-on Adafruit Flora board.

My next attempt to remove the alligator clips from the *Squishies* was to use more of the metal snaps I had been using to attach sensors to the fabric shells, in combination with ribbon cables made from conductive fabric. These ribbon cables could either be attached to the breadboard with even more snaps, to further the modular design of the *Squishies*, or sewn directly to the holes of the breadboard using conductive fabric. Aesthetically, these fabric ribbon cables were quite interesting, and furthered the organic and soft nature of the *Squishies* - but they also took up a lot of space, drawing away from the toy itself and moving focus to the tangled mess of ribbon connectors. At this point, there was also a ton going on inside of each *Squishy*. Electrical connections relied on the contact of many metal snaps, and there were paths of conductive fabric that could overlap when the toy was squeezed in specific ways, causing interference between sensors. I realized a lot of these issues would be resolved if I removed the 'modular' aspects of the *Squishies* (mainly, the snaps) and focused on making a prototype that could be concise and easily repeatable. So, I began to explore more streamlined concepts for the prototype as a whole.

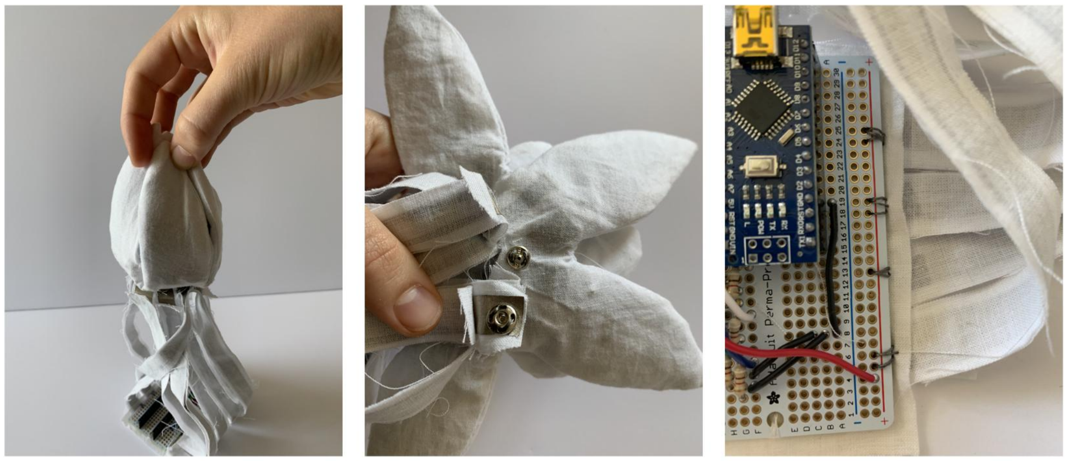


Figure 27: Columbine shaped *Squishy* shown with a sewn-on circuit board and snap connections.

To eliminate snaps, I had to find a new way to attach sensors to the fabric shell, or a different way to keep them in place. If I were to sew them directly onto the shell, and use conductive thread to connect them to the outer edge of the shells, I would need an intermediate piece of fabric to sew onto so all the stitching could be concealed and the outer shell would remain clean. So, to avoid overcomplicating the circuit again, I found a way to include all the sensors into one contained sheet. This sheet was created using the same method as the individual sensors, with two felt pieces containing all the conductive thread traces, and a piece of velostat in between. Each of these sensor sheets was sized to fit exactly into the fabric shell, so there is no mechanical connection of sensor to shell required. To connect these sensors to the circuit board, I had to try a new method for cabling. After some research on soft-hard connections that are frequently used in soft circuit design, I settled on creating ribbon cables with conductive thread sewn into lines down a long piece of fabric. This way, one ‘cable’ could be used for the whole sensor sheet, and it could be sewn directly to a traditional header at the end, where it would connect to a plug on the circuit board.

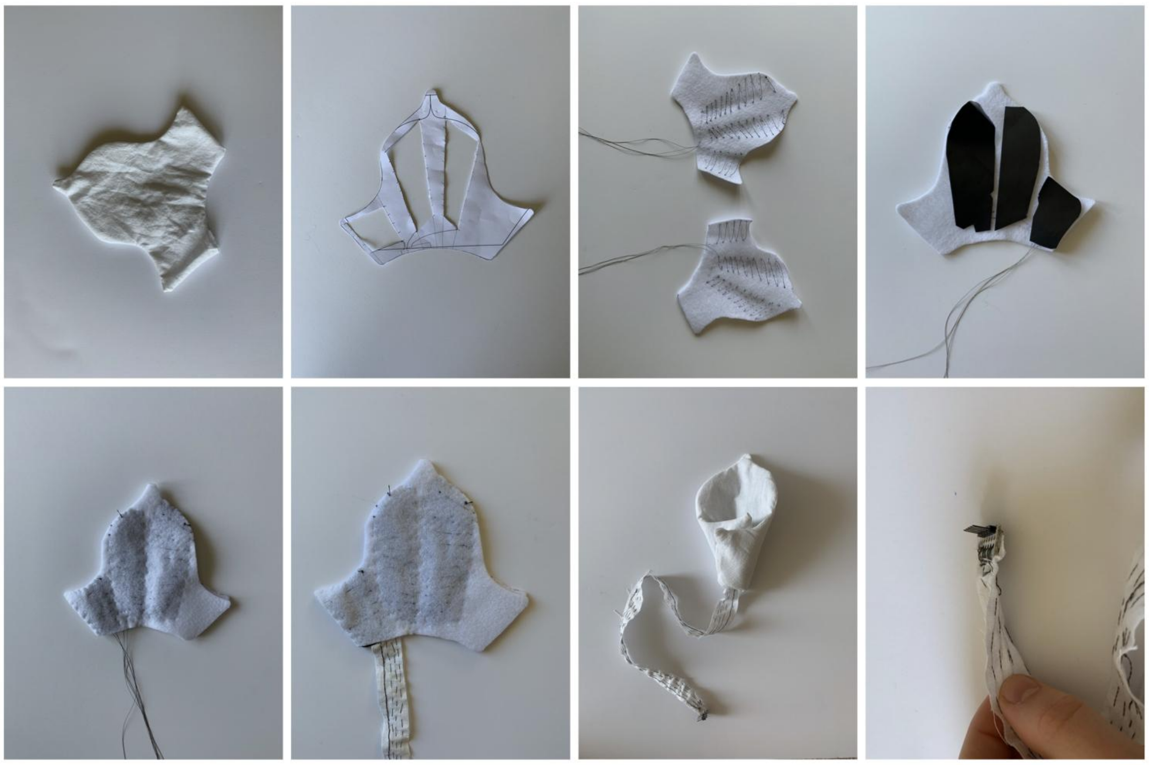


Figure 28: A step-by-step depiction of the process of creating a calla lily shaped *Squishy* toy.

This method of creating *Squishies* was the final prototype through my first round of user testing. At this point, a few issues were discovered with the hand sewn ribbon cables - mainly, they also had the potential for electronic interference between sensors. This was again an issue primarily if they were handled incorrectly, or if there was an error in manufacturing. First, I tried switching the soft-hard header connection at the end of the cable to an off-the-shelf connector. This, however, did not resolve the issues I was running into with interference. So, for the remaining user tests, I needed to shift towards a more durable design, that was also faster to manufacture precisely. At this point I decided to try using traditional ribbon cables, which could be sewn directly onto the sensor sheets using conductive thread. These connection points were then protected using hot glue, which would not only prevent the stitching from coming undone, but would insulate the connections to prevent a short circuit. The use of these traditional ribbon cables resolved all of the interference issues I was running into when using soft cables. Eventually, it is my goal to find a way to improve the fabric connections and move back to using those. Having a fully fabric toy is nicer aesthetically, and furthers the idea that nothing on the toy could snap or otherwise fail from overuse. I believe this would be possible with a careful manufacturing method that insulated all the threads used in the ribbon cable - it was just not possible during manufacturing this time because hand stitching each circuit is a very time intensive process, and I had to get on with user testing.

Squishy Design

The *Squishies*, at the time of their naming, were not actually squishy at all. They were very simple fabric shells with soft sensors embedded in them. “Floppy” might have been a more descriptive name. I knew they needed to be softer and more malleable, but I was not sure at first what would be the best way to make this happen. I started with poly-fil, which is the stuffing I often used in sewing projects as a child. It made the toys feel softer and more full, but they were very lightweight - there was nothing to really push against in the shell if you were trying to apply pressure to make sounds. I then did some experiments with craft foam. This material was interesting because it can be molded into three dimensional shapes with a heat gun. I made a few prototypes using these molded foam structures inside the fabric shells, but they were not strong enough to hold their shape consistently when sensors were also inside the fabric, weighing it down. I even tried embedding wires into the foam to help strengthen it, but you could feel the wires through the fabric shell, so I abandoned the craft foam.

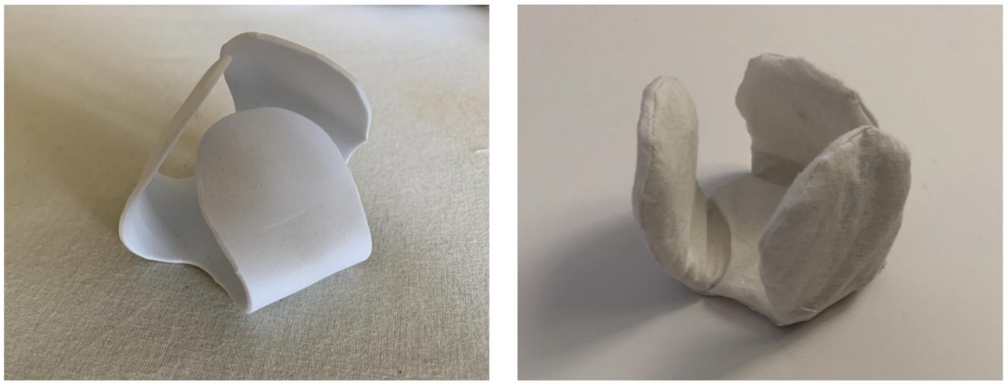


Figure 29: Craft foam shaped with a heat gun, and the same foam inserted into a fabric shell.

From here, I started experimenting with castable materials - mainly silicone and expanding foam. These could both be shaped exactly to fit inside the different fabric shells, and they could be made into different thicknesses, so they were an ideal choice for the *Squishies*. Both silicone and foam have some level of stiffness to them as well, meaning they would provide a surface for the user to push against while playing a *Squishy*. To test these materials, I found several two-part silicone and foam samples, and made test pieces of each that I could experiment with. For each sample, I placed the piece in one arm of a *Squishy*, with a sensor, and tested how it affected the sensor, how it felt underneath the fabric, how heavy it made the toy, and how nice it was to squeeze. I found the expanding foam was difficult to work with, and created samples that were not consistently uniform in texture or shape. It was also not as reactive when squished as the silicone was. So, I chose to go with silicone as my inner material. I picked a silicone that has a relatively low hardness, so it is stretch-able under the fabric - specifically it was the Smooth-On EcoFlex 20. Silicone also creates a very unique feeling when it's placed under fabric, is very hard to rip or otherwise break, and it provides a little bit of weight to the toys. So overall, the silicone gave the most in terms of tangible feeling.

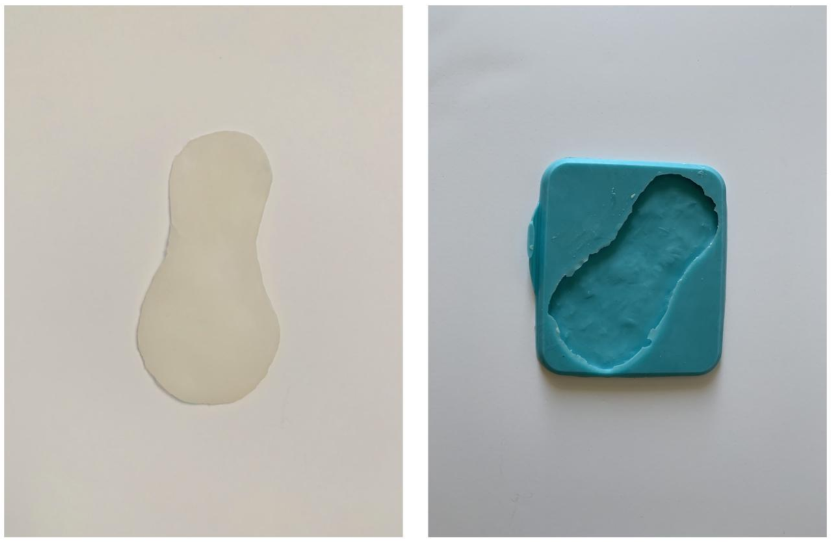


Figure 30: Silicone insert and the mold used to create it.

To cast all of these inserts, I first recreated the shape of each *Squishy* shell in clay, using the desired thickness I wanted my final silicone shapes to have. Then, I cast this clay into molding silicone - I used Smooth-On MoldStar Slow - to create molds for each of the shapes I wanted. From here, I could cast my silicone into the molds (using mold release to prevent sticking), and create my final shapes.

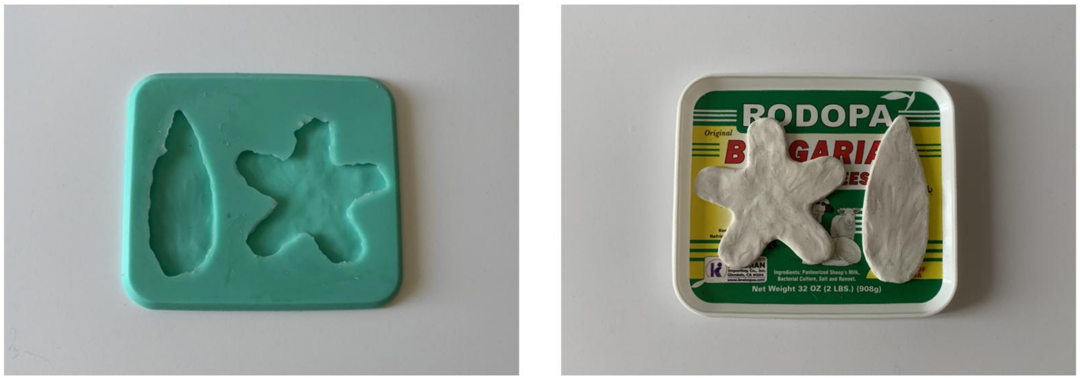


Figure 31: Silicone mold, and the clay used to cast it.

As somewhat of an aside to the *Squishies*, I did a few more experiments with silicone to see if I could use it as the sensor itself. For these tests, I used conductive rubber cords. My initial test was with a small rectangular mold, which I first poured half my silicone into, and let it set for about twenty minutes, which was enough time for it to partially solidify, but not fully set. Then, I placed my conductive rubber cord into the square, twisted it around to create an interesting pattern, and then poured the other half of my silicone over it and left it to set completely. My initial test with this new sensor worked very well, and I was able to bend and pull the silicone and rubber around to create a data output. However, I discovered that after about a day, the silicone had somehow corrupted the rubber, and it was no longer reactive. After some research online, I discovered that this is a known issue with silicone (though it is not very well documented), so I needed to try something else.

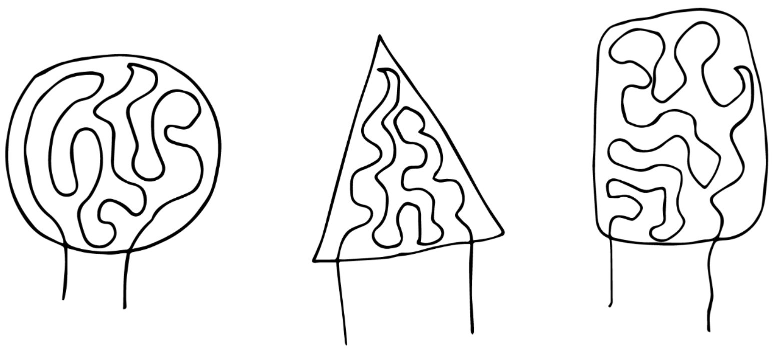


Figure 32: Concept sketches of the conductive rubber tests.

My first thought was to use heat shrink over the entire rubber piece before it got embedded into the silicone - so it could act as a barrier between the silicone and the conductive rubber, preventing the silicone from having an effect on the conductivity of the rubber. In a small scale test, this effectively protected the rubber, but I found that the heat shrink also limits the movement range of the sample, so the sensor is still not super effective. The last thing I tried was to find a very specific kind of heat shrink, which has some stretch to it, and to use that for a test with the rubber. This worked much better than the original heat shrink, and I was able to create a functioning sensor. I did not continue these tests, as the sensors did not work in combination with the non-elastic fabric shells.

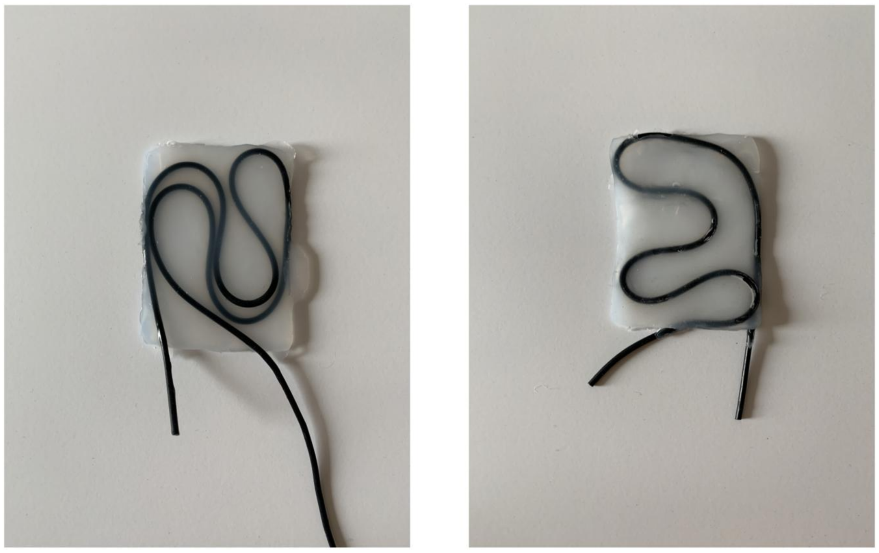


Figure 33: Conductive rubber embedded in silicone.

Shape Inspirations

The initial shells I made were very basic amoeba-type shapes. I made some with three 'arms' and some with two, with the idea that each arm would be a sensitive area on the toy (i.e. it would contain a sensor). These initial shapes were interesting, but I felt that with the arms as they were, it was too easy for users to repeat one motion over and over to create sounds, instead of exploring other ways to interact. They were also more lanky in shape, which encouraged bending as an action more than squeezing. I wanted to move into shapes that had more mass to them, so users could almost get a full handful of the *Squishy* when they went to play it.

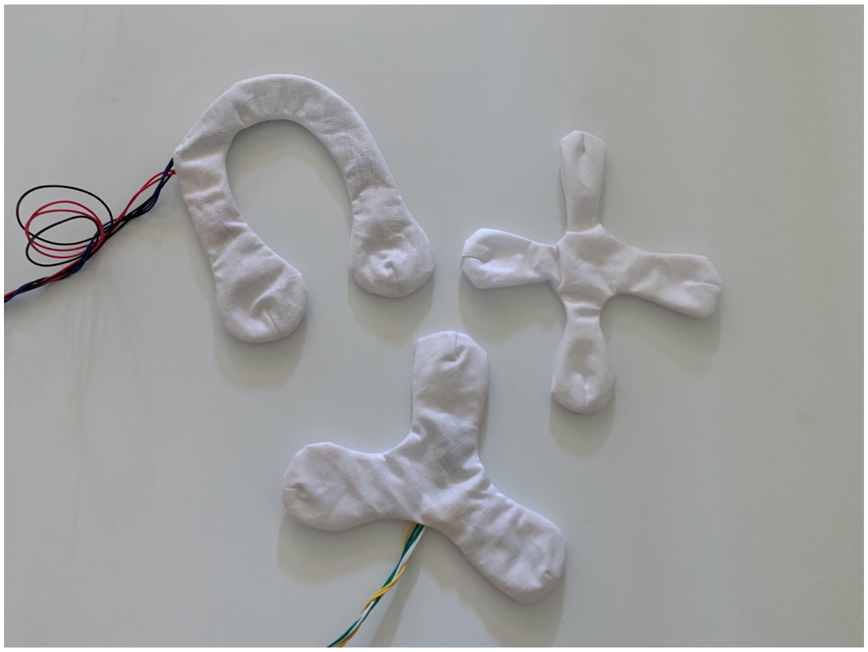


Figure 34: The first three *Squishy* shapes created.

After these initial shapes, I began to visualize how two three-armed toys stacked on top of each other had the rough shape of an iris. This is when I started thinking about the *Squishies* as more organic shapes. Mainly, I was thinking about things in nature that one might want to touch, squeeze or feel, but they cannot necessarily do so without damaging the object itself. I started to research more interesting flower shapes that might translate well into a *Squishy*. After compiling slides full of flower pictures, I created three new shapes for the *Squishies*: a columbine, a lotus, and a calla lily. Each of these contained a different number of sensors (generally, one in each petal) - the lotus had the most at fourteen, and the calla lily the least, with just three.

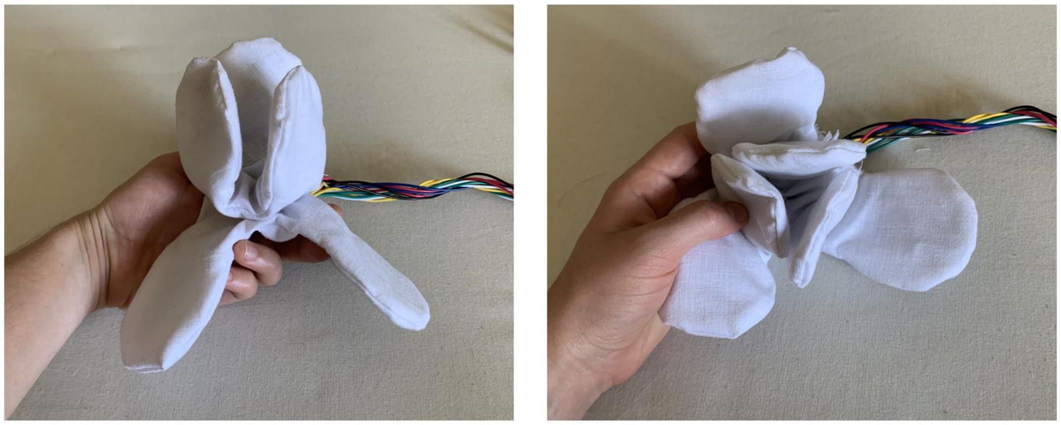


Figure 35: The iris shaped *Squishy*.

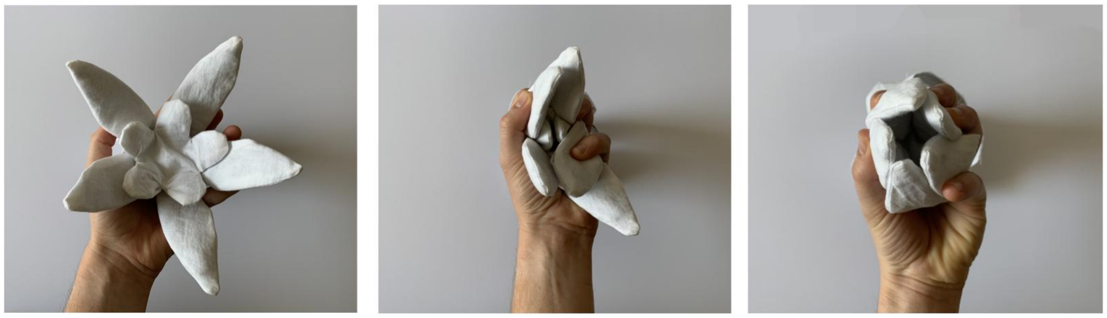


Figure 36: The columbine shaped *Squishy*.

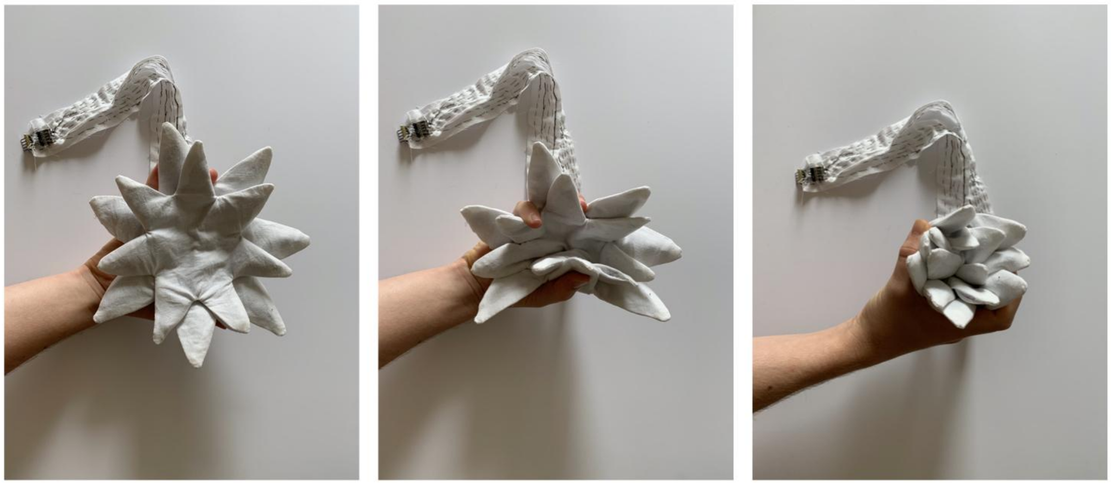


Figure 37: The lotus shaped *Squishy*.

During my research on squishy things found in nature, I also discovered a kind of sea creature called a nudibranch.[40] The nudibranch is a shell-less gastropod species that is generally very colorful. After this discovery, I knew I had to make a *Squishy* inspired by the nudibranch. They were such enticing creatures that looked so much like a colorful lump of silicone that I could not help but want to touch one. I ended up with a very abstracted version of a nudibranch, but it quickly became my favorite *Squishy* shape. It was a unique shape, and I liked that it was not an obviously recognizable shape. The nudibranch also ended up being the shape I did user testing with, as it had a manageable number of sensors (just four), was easier to manufacture en masse than the other shapes, and had a shape that made it easy to differentiate between each of the sensitive areas. I also discovered that the nudibranch shape lent itself well to being split in two, meaning it could become two halves. I made a prototype of this, which had snaps to join the halves, so a user could split it and play it with one in each hand, or, they could share the *Squishy* and play it with a friend.

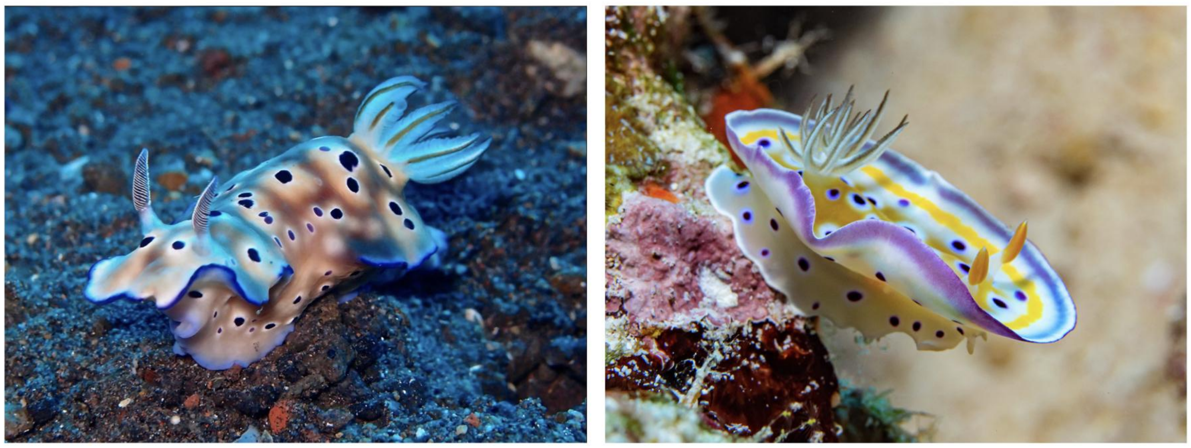


Figure 38: The nudibranch.[41] [42]

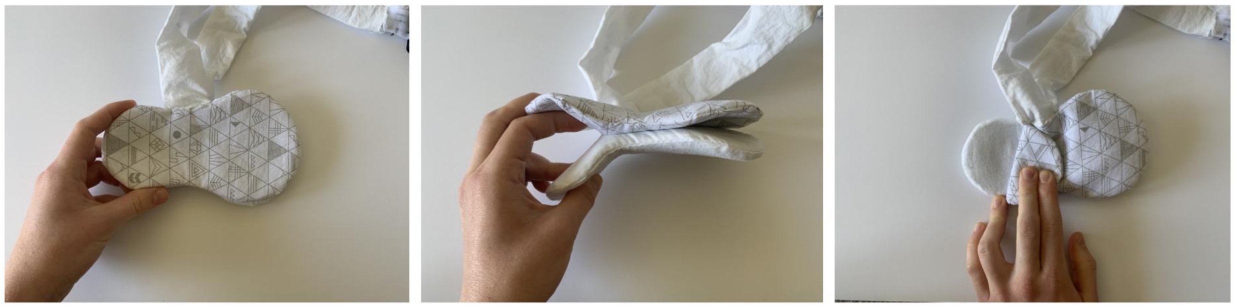


Figure 39: The nudibranch inspired *Squishy*.

Changes for User Testing

There were a handful of things I changed about the *Squishy* design to prepare it for user testing. Most of these had to do with safety: protecting circuits; hiding wires; etc. The first big thing was to make a protective box for the circuit board. This would serve two purposes: preventing children from messing around with the board, and protecting it during shipping. All the toys had to be shipped to users due to the COVID-19 pandemic, so extra precautions had to be taken to make sure everything could withstand shipping and be ready to go with minimal setup for the user on arrival. The protective boxes I used were initially off-the-shelf boxes designed to hold protoboards like mine, with a machined hole for the power cord, and a slot for the *Squishy* cables. After the initial testing, I switched to 3D printed boxes, which allowed me to make enclosures that held the boards more snugly and securely.

For initial testing, I also had to protect the fabric ribbon cables, which had exposed conductive thread. This was again necessary not only to prevent children from messing with them, but also to prevent short-circuiting and interference between the lines of thread on the cables themselves. For this, I made fabric sleeves that enclosed each cable, protecting it, and improving the aesthetics.

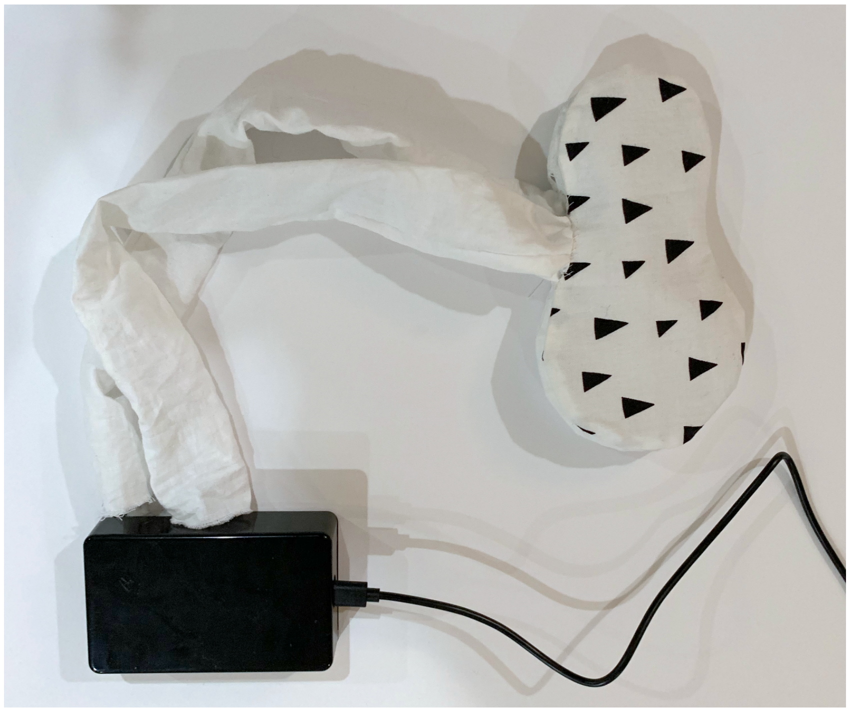


Figure 40: The *Squishy* and associated materials that were sent to each user.

The last big change I made for user testing was to introduce patterned fabric as part of the shell. All of the user testing was done with the nudibranch shape, so I made half the shell out of a patterned fabric, and left the other half white. This way, users would be able to easily differentiate between each arm of the toy. The patterns also helped the toys feel more playful, as well as visually intriguing.

Software Iterations

The very first prototypes I made were using a method I had used for previous projects. I connected my sensors to an Arduino Nano, which collected data and sent it out through a serial port. This serial data was collected through the program Pure Data[43], where it was remapped to a range of pitches and sent to my computer speakers. I started seeking alternate ways to parse my data partially because Pure Data was not fully compatible with my computer, but also I wanted to experiment with methods I had not used previously - it only makes sense to fully explore all the options before making a final decision.

First Prototypes

I continued using the Arduino Nano for the first several prototypes I made, and continued sending my sensor data out via serial port. All the incoming data from the Arduino was from the fabric-based pressure sensors, and I wanted to map it such that as you applied more pressure to the sensor, the note value associated with that sensor would increase. To do this, within the Arduino code I was collecting this data from the Analog ports on the board, and re-mapping it so each sensor had its own range of numbers. This was so when I sent the Serial data to Pure Data or another software, I would be able to differentiate between the sensors using the different data ranges. This also allowed me to map the incoming data to MIDI note ranges, so all the sounds corresponding to each sensor would be audible, and so I had control over what the toy sounded like.

I then faced the problem of needing to convert my serial data to MIDI notes, so it could be read directly by music software. The first thing I tried to accomplish this was the software Hairless, which is a serial-MIDI bridge that directly converts incoming serial data to MIDI notes. This allows the music software to read the Arduino board as it would any other MIDI instrument, which then allows you to use any music software that accepts MIDI instruments as input devices. So from here, I could select a music software that would work best for sound design for the *Squishies*.

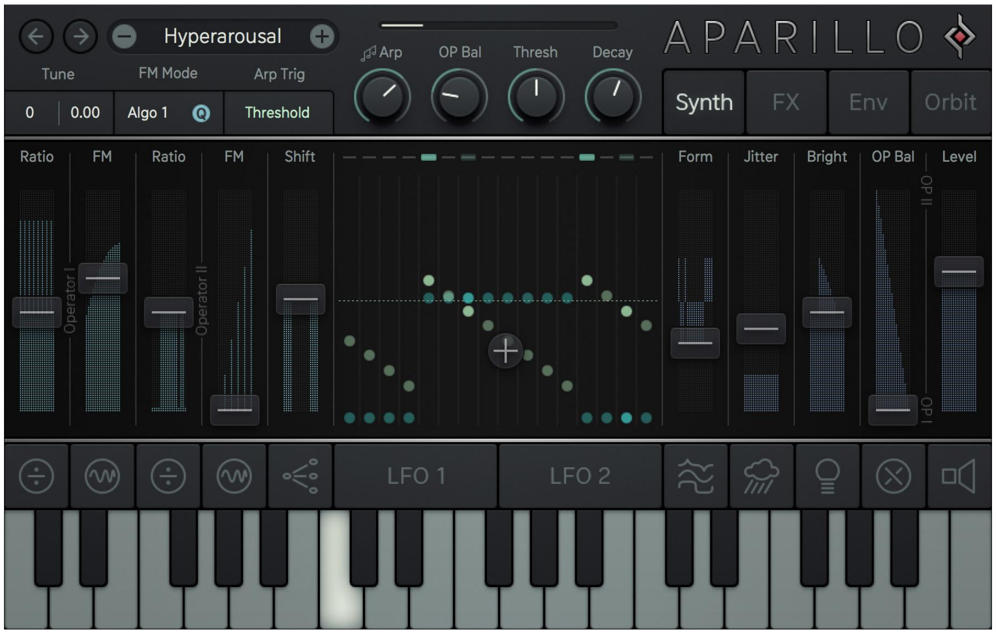


Figure 41: The Aparillo interface.[44]

I started using the software Aparillo on a recommendation from a labmate. Aparillo is an FM synthesizer with 16 voices that is often described as having a cinema-esque sound.[44] It also has a unique user interface that allows you to visualize the sounds and voices you are manipulating through graphs, as well the “Orbiter” which is a freeform control tool (Figure 42). Aparillo seemed perfect for my initial tests with the *Squishies*, as it has a somewhat unstructured interface that allowed for fluid sound control, which is exactly what my goal was for the *Squishies*.

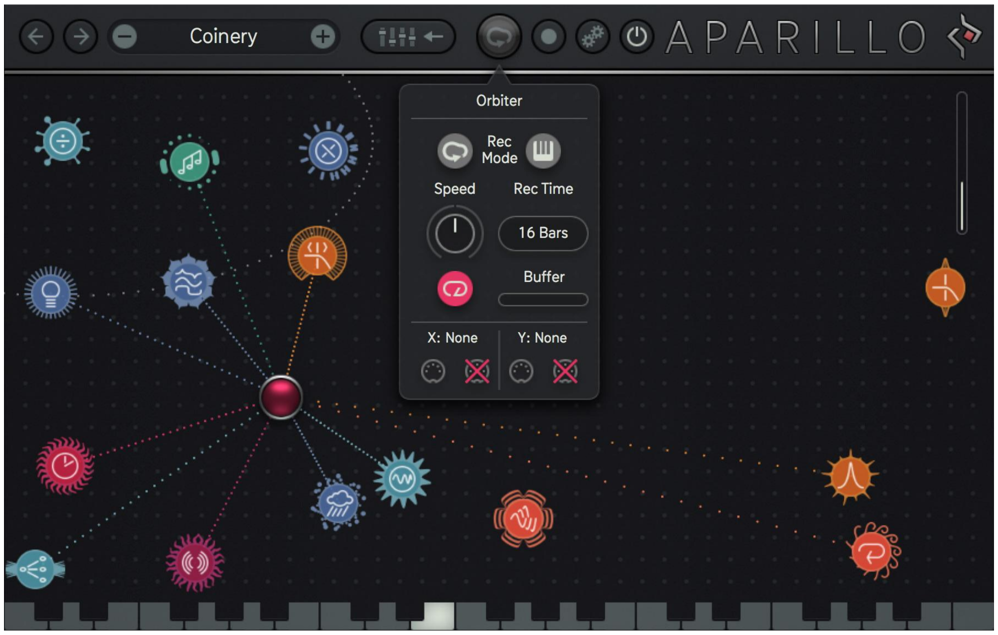


Figure 42: The Aparillo “Orbiter” interface.[44]

After experimenting with Aparillo a bit, and getting inspired by it’s visual interfaces, I wanted to create a way to visualize the sounds I was creating with the *Squishies*. I had used Processing in the past to make visualizations, and I knew that it was able to take in and use data from the serial port, so I decided it would be a good choice for my experiment. My idea was to create something that would show the pressure changes in the sensors as I squeezed them. Taking inspiration from a John Whitney piece I recreated for Prof. Zach Lieberman’s class “Recreating the Past” in Fall 2019, I made a visual with three arrays of dots, each arranged in concentric circles. Each of the circles represented one of the three sensors in the *Squishy* I was using at the time. As I applied pressure to a sensor, the corresponding circles started to move outward in a beating motion, moving faster as more pressure was applied.

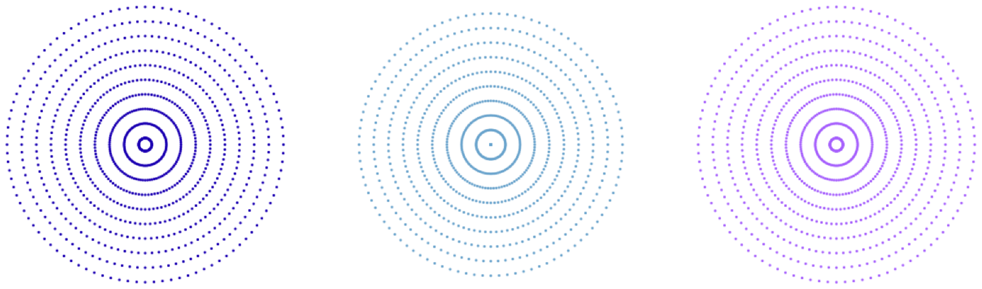


Figure 43: The Processing visualizations I made for the *Squishy* toy. Each circle represented one sensitive area of the toy.

I did not end up continuing with the visualizations, as I realized it was a lot more interesting for the audience members to look at my hands playing the *Squishy*, instead of at a representation of what I was doing with my hands. It was still an interesting aside though, especially since it was the very beginning of virtual school, and we were all looking for more ways to share our work with each other online. Through using Processing, however, I did find a new way to convert Serial data to MIDI, using the MIDIbus function in Processing. This method was more reliable than Hairless, so I continued using Processing to convert my data for a while after I stopped experimenting with visualizations.

One thing I did do with this version of the software was to put on a mini concert as part of a final presentation for Prof. Tod Machover's class - "Sound Past and Future" - that semester (Spring 2020). For this presentation, I wanted to show how the *Squishies* could be used for collaborative music making, so I roped my brother into playing a *Squishy* with me. I connected two instruments to one microcontroller, and mapped one instrument to play only higher tones, while the other played only lower ones. This way, it could be clear who was playing what. The concert was successful in conveying and getting feedback on this initial idea of a squishy instrument, and the collaborative aspect seemed appreciated at a time when most of us were in isolation. Though I did not end up immediately continuing with collaborative music through the *Squishies*, it was still a nice concept to play with, especially through the beginning of the pandemic.

Scratch Interface

I made a Scratch interface for one of the early prototypes of the *Squishies* for a class I took with Prof. Mitch Resnick in Spring 2020. The class, "Learning Creative Learning", was centered around microworlds designed for education. The microworld I was exploring was music, and I was just beginning my *Squishy* prototyping, so, with help from a few Lifelong Kindergarten students, I decided to create a Scratch interface that would let kids tinker with the music their *Squishy* toy was making. To accomplish this, I had to switch to a microcontroller that could connect via Bluetooth, as this is the only way Scratch can interface with physical technology at the moment. On a recommendation from Kreg Hanning (who also helped me get set up with the Scratch server, and his code for the MicroBit), I set my *Squishy* prototype up with the Adafruit Feather 32u4 Bluefruit LE. With this, and Kreg's code, I was able to start putting together my Scratch prototype.

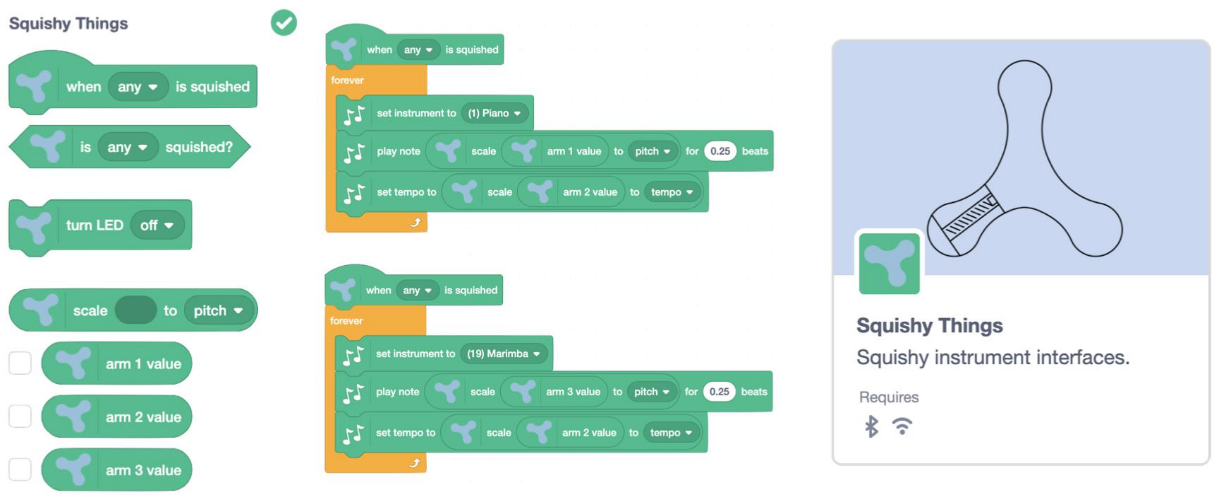

Squishy Things
when any is squished
is any squished?
turn LED off
scale to pitch
arm 1 value
arm 2 value
arm 3 value
when any is squished
forever
set instrument to (1) Piano
play note scale arm 1 value to pitch for 0.25 beats
set tempo to scale arm 2 value to tempo
when any is squished
forever
set instrument to (19) Marimba
play note scale arm 3 value to pitch for 0.25 beats
set tempo to scale arm 2 value to tempo
Squishy Things
Squishy instrument interfaces.
Requires

Figure 44: The Scratch interface I created for the *Squishies.*

The interface I made had several blocks that allowed users to control the sounds each arm of the *Squishy* was making. They were designed to be used with the Scratch music extension, which already exists. One block let you map different arm values to change either the pitch or the tempo of the sounds being made, one allowed you to turn the LED on the microcontroller on or off with an arm of the toy, one allowed you to tell if any arms were actively being pressed on, and one let you test the connectivity of the interface to the toy itself. So with these and the pre-existing music blocks, you could choose an instrument to play with, and set the pitch and tempo of that instrument with different arms of the *Squishy*. This is the furthest I explored with the Scratch interface - though I think there is potential to do more with the *Squishies* and Scratch, it did have a few drawbacks. Mainly, Scratch in its current form is not designed to be used with most physical interfaces, making it tough to do a ton with a *Squishy*. Additionally, Scratch music is somewhat limited in what music concepts you can explore, and what sounds you can use, so I was interested in creating an interface that allowed for more careful sound design. That being said, Scratch did make a wonderful interface that would allow kids to tinker with the sounds they made, and it did inspire a lot of the features that would end up in the final *Squishy* interface.

Switch to Teensy

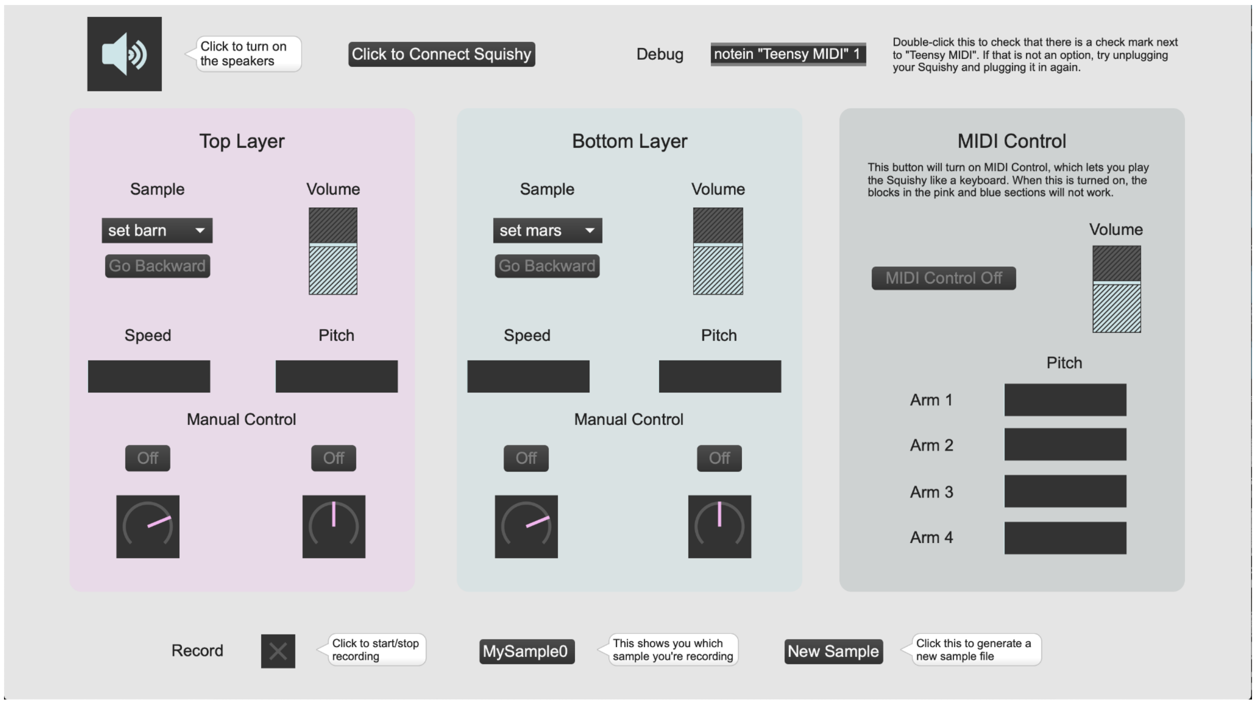
After a few months of testing my prototypes with Aparillo, Arduino, and Processing, a friend suggested I try switching to a Teensy[45] microcontroller. Teensys, as I learned, are able to send out data directly as MIDI notes, eliminating the middleman software that I had been using before. This enabled me to condense my code, so instead of having half the functionality in Processing and the other half in Arduino, it was all in one place in the Teensy code. In doing this, I actually eliminated some of the problems I was having with note mapping, and I was generally one step closer to having a plug and play device. This switch also inspired me to start exploring other options for music software, especially ones that would allow me to make my own interface.

MAX Interfaces

Cycling’ 74 MAX/MSP[46] is a block-based music software similar to Pure Data. I started using MAX when I learned that with it you can make standalone software patches that can be used by anyone (even if they do not have MAX on their computer). It seemed like a perfect way to create the first software prototypes that users would interact with. As I started using it, I was amazed by how well documented it was, and as a result how easy it was for me to make a prototype that did most of what I wanted it to - especially since I am not an expert coder by any means. It even allowed me to make clean, intuitive interfaces that kids could understand. My original intent was to use it only for my first user tests, but after making my first interface I liked it so much that I stayed with MAX for the remainder of my testing.

My first prototype primarily played with the groove block, which lets you change the speed and pitch shift of a music sample easily. I decided to find a set of short samples that could be pre-loaded into the interface for users to mess around with using the *Squishy*. This seemed like a good first step with the interface for user testing, as there would not be too much information for kids to absorb. The minimal interface also seemed like a good idea for my first user test, since it would allow the users (and myself) to focus on the concept more so than it being a polished final product. It also drove the focus more to the physical interface than the software to start. Probably the most fun aspect of the original software interface was the recording functionality. This let kids record their voice or any other sound to create their own samples that they could then mess around with using the other functions in the interface.

In the following prototypes, I added more features to the interface. I included a MIDI note mode that allowed users to play the *Squishy* like a MIDI instrument, with continuously changing pitches, so they had the option to do something apart from manipulating the samples. I also added more recording functionality, making it so users could save the recordings they made on their computer for later use, and so they could save up to five at once on the interface itself. For these later prototypes, I continued to add functionality and improve the overall intuitiveness of the interface. Namely, I made it clearer which sensors on the *Squishy* controlled which parameters on the interface. I also made it a goal to streamline the download and setup process for the toy as much as possible to make it easy for the users to access. Almost all my later changes were driven by user feedback I received throughout testing.

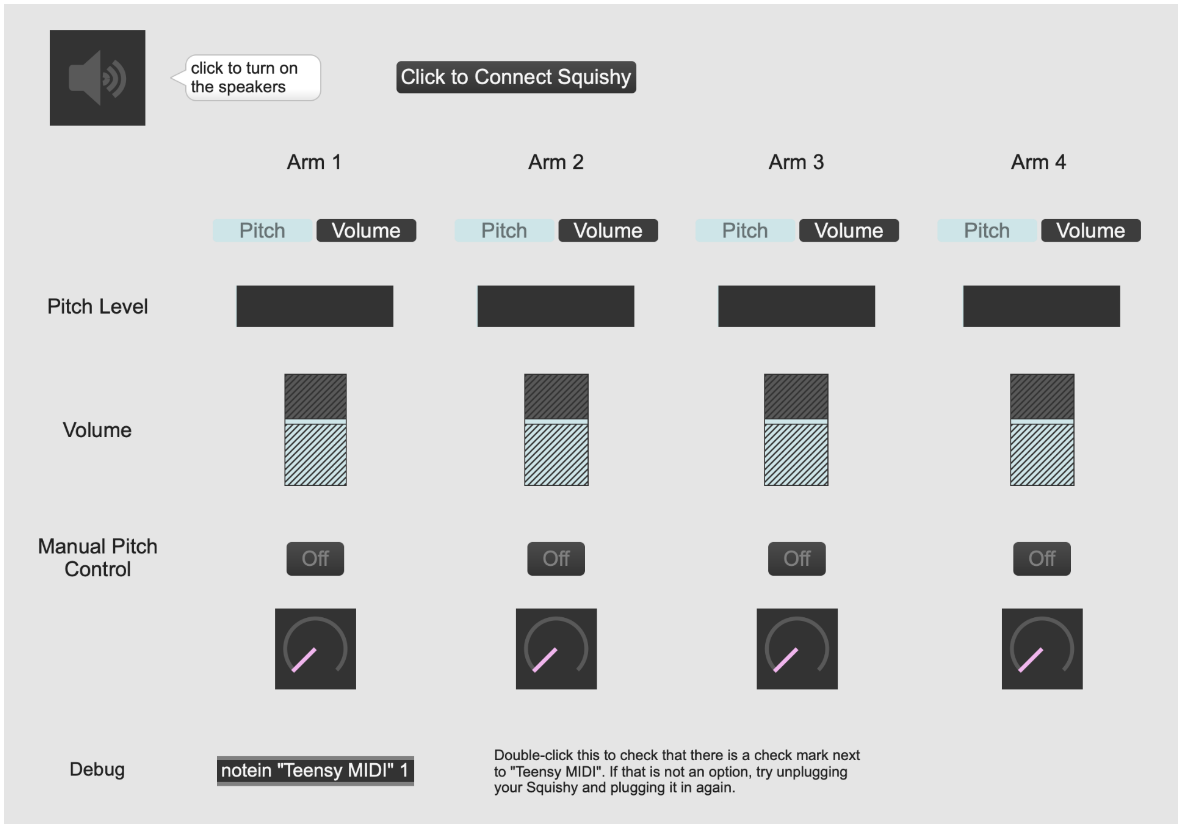


Double-click this to check that there is a check mark next to "Teensy MIDI". If that is not an option, try unplugging your Squishy and plugging it in again.

This button will turn on MIDI Control, which lets you play the Squishy like a keyboard. When this is turned on, the blocks in the pink and blue sections will not work.

Figure 45: The final version of the *Squishy* MAX interface.

I also created a MAX interface designed for meditation by adult users. This interface was intentionally very sparse, as I wanted users to focus more on their breathing and subtle hand movements than on the computer screen. It only worked in MIDI mode, letting users change the pitch of each sensor, and also had an option to change the volume of the tones using the sensors. Much like the interface designed for kids, most of the adjustments made to the interface were improvements to the overall intuitiveness and functionality, and were informed by user feedback. You can download my MAX Patches here.



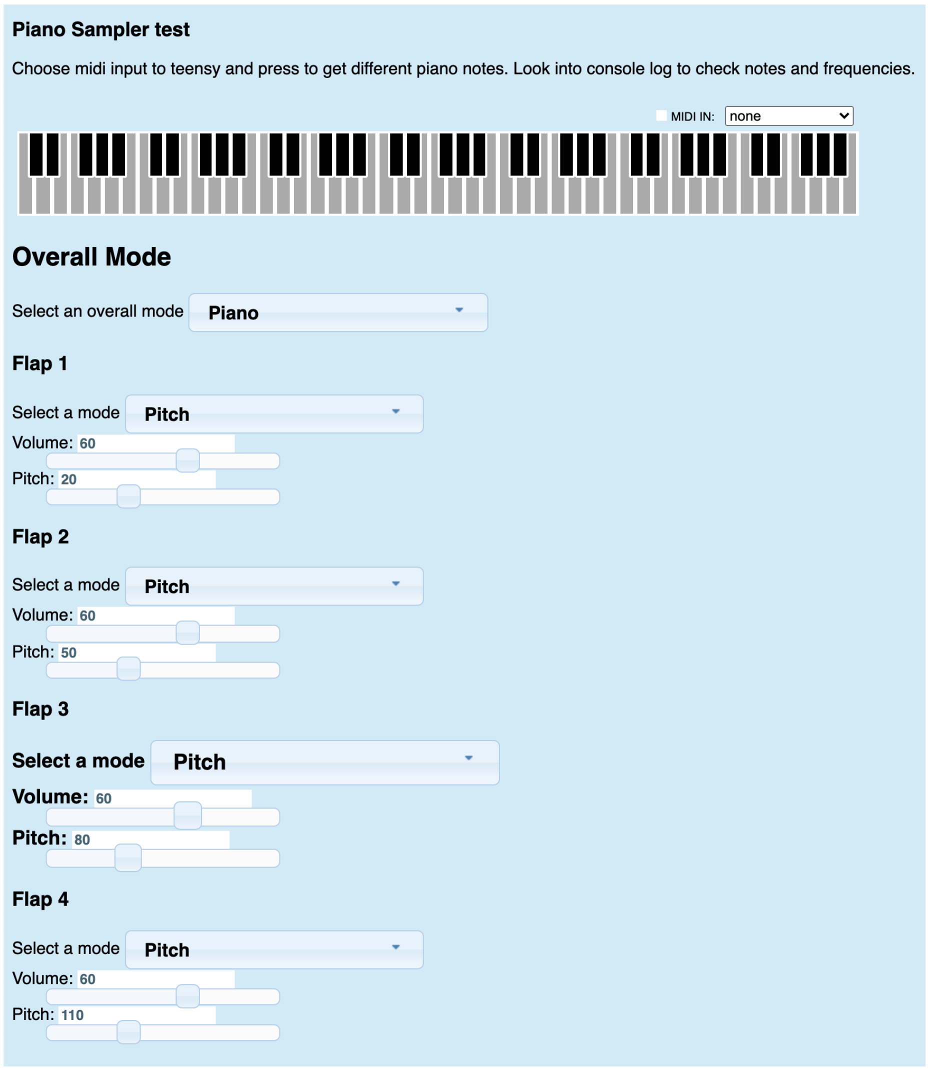
Double-click this to check that there is a check mark next to "Teensy MIDI". If that is not an option, try unplugging your Squishy and plugging it in again.

Figure 46: The *Squishy* meditation interface.

Web Interface

The ideal format for the *Squishy* software interface is as a website. This would make it easy for users (especially young users) to connect the *Squishy* to their computer and start making music, without any hassle. With external software, many operating systems will give security alerts for “unknown” software, making it difficult to walk people through the download process, as there are often extra steps to bypass these security alerts. But on a webpage, it is easy to make the *Squishy* more of a plug and play device. The web interface would also work well for kids, as a lot of them use Chromebooks, which do not allow you to download most external software. The MAX interfaces require a machine running either macOS or Windows to function, meaning that many of my users had to borrow a parent’s machine to use the *Squishy*.

My amazing undergraduate research assistant, Jenny Zhao, helped me create a prototype of a web interface for the *Squishies*. It had more limited functionality than the MAX interface, but was functional enough that it was fun to play with, and was able to be user tested in several later workshops. The web interface could be played in MIDI mode, as a piano keyboard, or it could give the user control over the tempo and volume of one preloaded music sample. It also came in handy when several of my users only had access to a Chromebook during one of the workshops, so they were still able to complete the exercises with the other kids. Overall, a web page would be ideal for hosting the *Squishy* interface, and in the future this prototype will be improved to make that possible.


Piano Sampler test

Choose midi input to teensy and press to get different piano notes. Look into console log to check notes and frequencies.

MIDI IN: none

Overall Mode

Select an overall mode: Piano

Flap 1

Select a mode: Pitch

Volume: 60

Pitch: 20

Flap 2

Select a mode: Pitch

Volume: 60

Pitch: 50

Flap 3

Select a mode: Pitch

Volume: 60

Pitch: 80

Flap 4

Select a mode: Pitch

Volume: 60

Pitch: 110

Figure 47: The *Squishy* web interface, created by Jenny Zhao.

5

Final Squishy Design

To review, I will go over what encompassed the final *Squishy* toy - in this case meaning the toy that was used for the last user tests that I completed. The nudibranch *Squishy* was the shape I chose for all my user tests. This was for several reasons. For one, it had only four sensitive areas, which seemed like a reasonable number of sensors and mappings that kids would be able to keep track of. Having only four sensors also made it easy to create mappings that were not overly complicated. This was a manageable number of parameters that would allow kids (or other users) to easily keep track of what they were doing with which sensor. The nudibranch shape also lent itself well to making each sensitive area distinctive, so it was easier for users to keep track of which did what. I facilitated this by introducing patterned fabric for half of the *Squishy* toy, and instructing users to play the *Squishy* with the white side facing down, and the patterned side facing up. This way, the four distinct areas could be differentiated as: the patterned small side; the patterned large side; the white small side; and the white large side. The user could refer to the top and bottom sides of the toy (patterned and white, respectively). The nudibranch *Squishy* was also easier for me to manufacture, as I was doing everything by hand. It had a smaller number of sensors and a relatively uncomplicated shape that lent itself well to hand-making. This was key since I had to manufacture a lot of them for user testing.

Each of the final shapes had silicone embedded in one half of the *Squishy* toy. Ideally, it would have been both halves, but I realized that having silicone in the top half created too much weight. That triggered the pressure sensors on the bottom half of the toy, so I temporarily removed it. The silicone was cast to fit precisely inside the shell of the *Squishy*, and created a nice hand-feel when the toy was squeezed. All the *Squishies* continued to run on a Teensy as well, and the Teensy code sent out all data as MIDI. This way, all the *Squishies* registered on computers as MIDI instruments, that could be connected to any music software in the same way one would connect any other MIDI controller. Each *Squishy* would connect to the pre-made music software I provided automatically when the software was loaded.

The software used for the final toys was the Cycling '74 MAX/MSP interface I created. This interface was exported as a standalone piece of software, which could be downloaded and run on any Mac or Windows machine, regardless of whether the user had MAX downloaded on their computer or not. This software had a series of pre-loaded music samples - spanning from simple kick drum beats to funky riffs, mostly from the website Freesound - and each half of the *Squishy* (top/patterned or bottom/white) controlled one of the samples. Each side of the half (big/small) controlled either the speed of the sample, or the pitch shift of the sample. The interface also had recording functionality that let kids record their own sounds and then play them back as samples, which could then be manipulated with the sensors embedded in the *Squishy*.

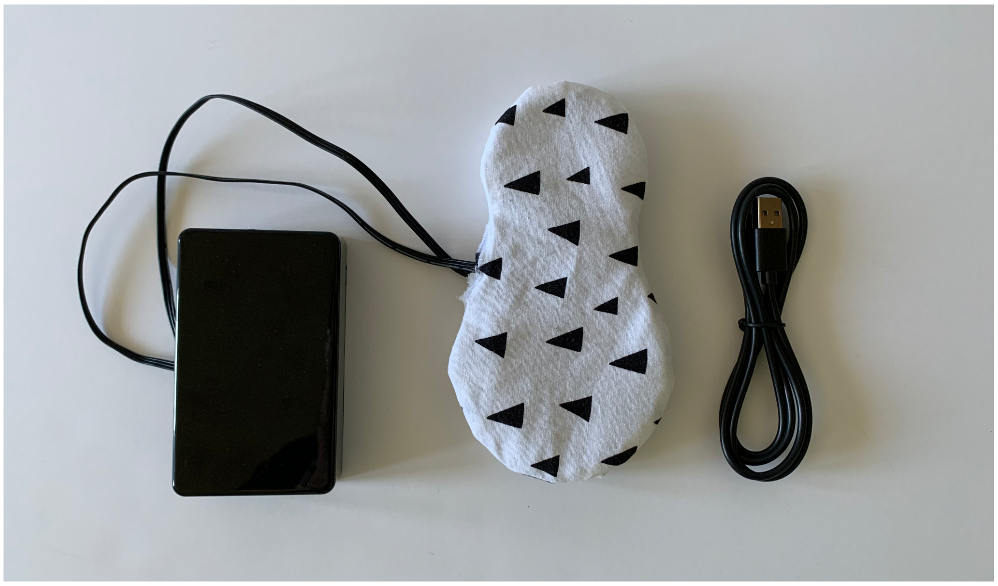


Figure 48: The final *Squishy* design

6

User Testing & Evaluation

To test the success of the *Squishy* toys, I decided to run a series of workshops and solo user tests with people from different backgrounds that I thought might fit with the variety of applications to match the *Squishies*. Due to the COVID-19 pandemic, all of these user studies had to be conducted remotely, so a great deal of planning was required to ensure that all the participants would be able to acquire a *Squishy*, and then use it with the provided software on their personal computers. In the end, all the user studies were conducted via Zoom, after I had mailed a *Squishy* to each participant and sent them the software patches to download over email. The workshops I ran for user tests fell into two categories: adult meditation and beginner electronic music for children aged 8-14.

Kid Workshops

The first workshop I ran was with kids from an MIT-led afterschool program, Afterschool-Tastic. I joined as a guest mentor for the program, and led two workshops with five kids in my age range. I capped my workshops at five kids since I knew there would be some setup work to do over Zoom, and I wanted to make sure I had the time to check that each child had the system running correctly on their computer. To further facilitate this, I structured all of my workshops to have two hour-long sessions on two different days, a week apart. This way, I could spend a lot of time during the first workshop helping kids with the software setup - this usually involved helping to navigate computer firewalls, troubleshooting *Squishy* connection issues, and making sure everyone had the interface speakers turned on. During this first workshop I would also walk everyone through the interface via screen share, to show them the different functionalities and how to use them. I then had them each do a couple of quick exercises to test each function on their own and further confirm that everything was working correctly. These tests were always very simple; for example the recording test involved recording your voice as you counted to ten, and then using the *Squishy* to speed yourself up so you sound like a chipmunk. By getting all of this setup out of the way during the first session, I could avoid spending a lot of time debugging during the second one and instead spend a lot of time with the kids playing music with the *Squishy*. Having the week in between the two also allowed kids some time to explore the interface independently, and let me give them the 'homework' to come back with something they could share - either a sample they made, a fun piece they created, or just an observation.

The second session of these workshops started with me asking for a report of how the kids' explorations of the toys went, and asking if anyone had something to share. I would also give them 5-10 minutes to recreate something they had made in case they had not been able to screen record or otherwise save their recordings. This discussion and sharing part was always particularly hard over Zoom, as kids were nervous to share the sounds they had made - no one wanted to be put on the spot. Throughout the kid workshops, I did get a handful of people who were willing to share something they had made with everyone, but it was not easy to convince others. During the rest of the workshop, I led kids through a series of other exercises that encouraged exploration of the interface. For example, we completed an exercise that taught kids about tempo, by asking them to choose a music sample for one arm of the toy and set a steady speed using manual control. They were then asked to pair one of the arms of the *Squishy* with a drum beat sample, and then match the toy's beat (or tempo) to that of the music sample they had chosen earlier.

We also tried out an exercise in group music-making, which was very difficult over Zoom (because of the audio lag), but as I told the kids in the workshop it was nice to try since we were not able to make music in person at the time. To do this, I asked everyone to set their software interface so we were all using the same samples, and then I had everyone unmute and start playing something. As expected, it was tough to get a coherent sound out of anyone's computer, but it was still a nice thing to try. During a later workshop, I was able to effectively find a way to do a group activity via Zoom, that still utilized the *Squishy* and the software interface. For this, I realized if I made myself the vulnerable party (i.e I was doing all the sound sharing) then it was okay. So, I had everyone compose a piece with me through screen share. This way, I could take suggestions from the kids, and I was the only one who really had to be making weird noises into my microphone on Zoom - which of course I do not mind doing at all. This exercise showed a huge difference in audience reaction - almost everyone in the workshop shouted out suggestions, and we were able to make an interesting, if not slightly silly, piece of music together. You can hear the piece we made here.

In later workshops, I also introduced other exercises for the kids to try out. For one, I started to have everyone try meditating with the toys towards the end of the second workshop, using the meditation exercises I talk about in the next section. I also had everyone try out the prototype of the web interface my UROP, Jenny, had been working on. This was a more bare-bones version of the software interface I had created in MAX, but it was still well-received by the kids. Having the web interface as an option also helped immensely during a round of workshops where one user came with a Chromebook - which can not run external software. This way, the user was able to play along with most of the exercises I had planned, just using the web interface instead of the MAX one.

The last user studies I did with kids were set up as independent explorations instead of workshops. During these, I met with each child individually for an hour to get their software set up and to go over the interface. I also had each person spend a decent amount of time exploring the interface on their own while we were together on Zoom - this way if they had any immediate questions or confusions I could address them right away. After this, I had the kids go off on their own for a week to explore the interface, and then I would follow up with an email asking for sound samples and sending along the user feedback form so I could see how their week went. One question I asked each of these independent study kids was whether they liked exploring the *Squishy* on their own, or if they would have preferred a group workshop session for learning about and using the toys. Interestingly, every person who responded said they preferred the solo explorations that they had experienced. Because of this, if I had continued further workshops after this, I would have done more tests in this manner. I think this response may have been different outside of COVID-19 times, but given the challenges virtual workshops pose, and the level of timidness I received from many kids during the workshops, it does make sense to me that kids felt this way.

That all being said, I believe the kids' response to the *Squishies* was overall very positive. When asked to describe the *Squishies* in a few words, I got responses such as: "fun";"cool"; "possibilities"; "creative"; "interesting"; "entertaining"; and "one of a kind." Most participants also stated that using the *Squishies* made them want to continue learning music in some form. When asked what they learned, kids highlighted "how to control music with my hands", "music can be really fun to play around with", and "how to record my own music". Child users also had fantastic ideas for new features that I could add to the interface, from things as simple as adding animal noises to the interface or making the toys shaped like a stuffed animal, to adding sound filters to further modify the samples. Watching kids play with the toys on Zoom, I also noticed several times that kids called their parents or guardians over to look at what they were doing, and to listen to the sounds they were making. I cannot fully express the joy I saw on their faces while showing off their creations, but it was true happiness. This kind of interaction seems to rarely happen in other kinds of music education - I do not think I ever called my mom over so she could listen to me play my scales.

Meditation Workshops

The first adult workshop I ran was with people from the MIT Media Lab, where we focused on using the toys as tools for meditation. I recruited people through the Festival of Learning, which is a Media Lab event where graduate students can run workshops on anything they know and want to teach about. The five users I sent *Squishies* for this workshop were a mix of graduate and undergraduate students affiliated with the Lab in some way. Unlike the previous workshop, this consisted of only one 1.5 hour session. I correctly assumed that setup would be considerably easier with adults than it was with 8 year olds, so there was no need for the additional session devoted to debugging. During the workshop, I gave everyone a brief overview of what I was doing with the *Squishies* and why. Then we completed two exercises with the *Squishies*. The first focused on slow breathing, and asked participants to squeeze the *Squishy* with increasing force as they inhaled, and slowly release it as they exhaled. The second asked them to try and maintain one pitch with the *Squishy*, by keeping their grip as still and consistent as possible.

I received generally positive feedback from users during this workshop - most people found it helpful to have a physical tool to guide them through meditation exercises, and found the experience relatively relaxing. As one user put it "It was very fun to make sounds by squishing things. I do think it was easier for me to meditate than usual by focusing on the sound". Many people gravitated more towards the arms of the *Squishy* toy that were mapped towards lower pitches than the ones mapped to higher ones. There were also several requests for the *Squishy* toy to be even more squeezable and stress-ball like than it was - something like a single pad that had one sensor you could interact with while meditating. Two people seemed to envision the same thing in this regard when asked what suggestions they had for future *Squishy* shapes: "Small, circular squishy pads could be fun" and "I envisioned having one larger disk with one pressure point and the UI allowing me to choose one." Though I did not explore these new shapes further due to time constraints, I did take some of the other user feedback into account when running the following meditation studies.

The next adult user tests I conducted were solo explorations of the interface with the *Squishy* toy. For these studies, I mailed participants a *Squishy*, and then emailed them the software, as well as instructions on how to complete the two meditation exercises. Each of these tests were staggered slightly, so I was able to utilize feedback from some of the earlier ones to inform the later ones. For example, one of my first participants suggested that it might be interesting if you could control the volume of the tones with the *Squishy* toy in addition to the pitch. I was able to somewhat easily implement this adjustment, so I could send this updated patch to the rest of my participants to use with their *Squishies*. The interface was again generally well received by users, and most people found that having a physical object aided their meditation. One user even suggested that the software aspect might not even be necessary: “I think the software interface is not strictly necessary, especially if it's on a computer/phone. One key benefit people want to derive from meditation is to get away from electronics and external stimuli. Perhaps add a visual option to the toy in a subtle way, i.e., add vibration to signal volume/pitch; the stronger the vibration, the higher the pitch, etc.” This makes a lot of sense, as the goal of meditation is to be un-distracted, and having to mess around with a piece of software is not necessarily ideal for many people. So, removing the computer aspect may be beneficial for this use case of the *Squishy*, and is something that I should explore further in the future.

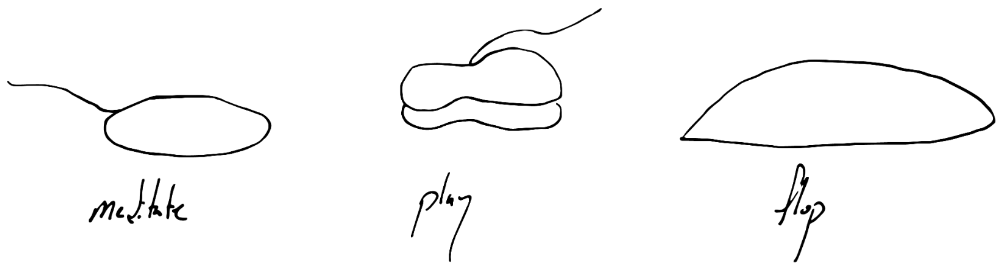


Figure 49: Concept sketches for different *Squishy* use cases.

These solo exploration tests were a good choice for the meditation exercises. Being able to experiment with the interfaces on their own allowed users to really spend a lot of time with the *Squishies*, and also encouraged them to use it more than once while they had it. In the meditation workshops I ran, I got the sense that it was a little bit awkward for people to meditate in front of each other, as it can be a private activity, but these solo explorations solved that. In both tests the majority of participants reported a relaxing experience when using the *Squishies* (I did get one 'maybe' response from each test round). The main reason people were hesitant to say a strong 'yes' for whether it was relaxing or not would probably be the sounds used in the software. During the first workshop, many of the pitches skewed too high, which was irritating to some people. I amended this for the solo explorations, but there was still some desire for more varied sounds, or as one person suggested "Maybe something that sounds more like [an] individual syllable of spoken language. Like 'a,' 'ma,' 'ga,' etc." In the future this would be something to explore and further amend on the interface side. Adding a more ambient, varied soundscape might be more appealing to those trying to meditate or otherwise relax.

Overall, the meditation user studies were a success, and I did receive a lot of helpful feedback that can inform future versions of both the physical *Squishy* and the software interface. Most users found having an object to help them meditate useful, and many did find the experience relaxing. One user who regularly practices meditation thought her experience with the *Squishy* was similar to her meditation style: "I think following the pitches and the physical action (squeeze and release) gives a concrete thing my mind can turn to when other thoughts come and go. This aspect closely resembles my daily meditation practice (Transcendental Meditation), during which I close my eyes, breathe normally, and silently speak a mantra in my head no matter what thoughts come to mind…The effortless aspect is what makes me stick to it. I think the Touch exercise is very similar to an effortless meditation style." This thought is encouraging, and implies that the *Squishy* can really have a place in a meditation practice.

High-Level User Studies

The last group I wanted to do user tests with was a set of people who had extensive background in electronic music. The goal of this was to see if there was a use for the *Squishies* outside of beginner-level music study, and to really see if the *Squishy* could be outgrown. It was a big concern of mine that kids might get bored of the toys after a while, or that they would have no use for them once they learned all the software interface had to offer. By conducting these higher-level user studies, I could see how a more advanced musician might utilize the *Squishy* as a MIDI controller rather than an instrument in the more traditional sense.

For these tests, I recruited other students from the Opera of the Future research group, all of whom had extensive experience using MAX. I sent them the patches I had made for meditation, and the ones I made for kids as a starting point. I then asked them to create something new for the *Squishy* in MAX, something that reflected what they might do with any other MIDI controller if it were soft and malleable. Each of them came back with something completely unique, and representative of their own musical styles.

Aarón Montoya-Moraga used the Squishies in combination with the Korg volca keys, which is an analog loop synthesizer. Their patch utilizes the Squishy as a controller for manipulating the delay effect of the synthesizer. Aarón's patch is a great example of how the *Squishy* could act as a "left hand controller", allowing a musician to change high level aspects of the sound with a flexible controller in their left hand, while using the right hand to control pitch with a keyboard or ribbon. Karsten Schuhl created an additive synthesizer patch, which mapped each arm of the *Squishy* toy to a different sound parameter: distortion, fundamental frequency, and partial structures. He also created a visual representation of his sounds in MAX, which changed color synchronously with the different parameter shifts created by the *Squishy*. Nikhil Singh created a patch that allowed for manipulation of pre-existing sound samples through the *Squishy*. Here, one half of the *Squishy* was mapped to the sample's loudness, and the other was mapped to it's spectral centroid. As the *Squishy* was squeezed, the sample's loudness increased, and it's centroid skewed higher. The sound was then manipulated through corpus-based concatenative synthesis, a method which involves searching the given sound file for pieces with similar spectral centroid and loudness compared to the sound coming from the *Squishy*, and then stitching the two pieces together to create a continuous, smooth, output. You can see and hear what each of them did here.

To explain their experiences with the toys, I will rely on their voices: "*Squishies* are a super versatile interface for music and arts, because of their squeezability and its open source hardware and software, which make them platform-agnostic, and make them usable and tweakable ll sorts of multimedia events, including envelope generation, filter sweeping, sample triggering, and generating streams of data for controlling audio, video, text, and light"; "*Squishies* promote collaboration, and they are highly suited for classroom activities, for childs between 6 and 106, in a way that is on trend with newer interfaces and instruments…"; "*Squishies* suggest squishing, stretching, twisting, scrunching, which feel somehow different to me from most physical metaphors for control (turning knobs, sliding faders, pressing keys or buttons) and gestural metaphors for playing most instruments (strumming, striking, blowing, bowing, etc.). At the same time, these all feel natural and are probably some of our earliest learned behaviors. They also all suggest something about the qualities of sounds. I focused on discovering some of these suggested ways of contorting sounds, which, outside of the constraints of instrument and controller paradigms, both was a lot of fun and yielded some surprising new ideas I do not think I would have tried otherwise"; "Squishy music toys are an artifact engaging a person in non-traditional music-making by inviting them to explore and change musical structures with their hands. As a reasonably nondescript object, a *Squishy* incites curiosity in how to engage with it. Furthermore, their design supports the mainly tactile interaction fluidly, translating it into smooth analogue input data as an extension of any digital music creation practice. This way, it can be used as a controller for higher-level interaction and a low-entry barrier interface for beginners. The principles of interaction, i.e. bending, pulling, and especially squishing, allow for multiple dimensions of freedom of exploration of the object and its effect on inherent software applications. With its physical scale, it is intrinsically linked to the potential movement of a human hand. Its affordance allows anyone to influence music and sound within a spectrum of movements from crass to finely controlled. Squishy music toys grant an experience that is playful and deeply connected to the senses, extending the realm of physical engagement with sound and music."

These high-level tests did answer my question of whether or not the *Squishy* can grow with you as you learn electronic music, or whether you can achieve a mastery of the *Squishy*. The answer is yes, in a sense: you can continue to effectively utilize the toys as a part of your musical life as you gain more experiences, but it becomes more of a tool that enables your sound, rather than an instrument that defines it. This can be attributed to the fact that the *Squishy* will connect to any music software in the same way any other MIDI controller would, allowing it to seamlessly integrate into an electronic musician's collection.

Challenges of Virtual Workshops

I feel that it is important to briefly acknowledge the challenges the COVID-19 pandemic created with respect to this thesis project. Conducting user tests with physical objects was not easy to figure out, and some sacrifices had to be made to safely and effectively evaluate these interfaces. The main difference that came as a result of the pandemic was the shift from in person workshops or user tests to virtual ones conducted via Zoom. Virtual workshops with physical objects are tough for a number of reasons. The first thing I had to figure out was smoothing out the design to make it sturdy enough that the *Squishy* would not break both during transit and during its time with a user. Part of this also included safety measures, which would help ensure that users would not be able to touch or otherwise mess around with the *Squishy* electronics.

During the workshops themselves, there were other obstacles unique to being virtual. For one, I was not able to carefully observe people as they used the *Squishies*, meaning I relied heavily on written and verbal feedback from the users for my evaluation and improvements, but it was also harder to get people - especially kids - to feel comfortable sharing out during workshops. So, I spent a lot of time outside of workshops chasing users and parents for feedback or other media via email, which was not an easy task to say the least. There were also challenges surrounding software, and making sure every user would be able to run the programs I sent them on their personal machines. This was a huge challenge since most kids use Chromebooks at home, which can not run external software. I asked during all of my workshop sign-ups that users come with a machine running Mac OSX or Windows, but despite this some people still came with Chromebooks, which put me in a tough spot. Even kids who came with a Mac or Windows machine ran into problems navigating the security systems most computers have now, which block any software coming from an ‘unknown developer’. As a result of all of this, many users cited the software setup challenges as their least favorite part of the workshops, which was understandable, but also unhelpful for me in wanting to improve and get feedback on the *Squishy* toy and the software itself. To avoid sounding like I am complaining, this is all I will say on the challenges of COVID-19, but I feel it needs to be said in order to put the evaluation of this thesis in the correct context.

7

Future Work

There are many directions I envision the *Squishies* going in the future that range from art installation to open-source hardware. I would like to outline a few of these to further emphasize the true versatility of these squishy instruments, controllers, toys or meditation devices.

A big thing I would like to explore further with the *Squishies* is increased tinkerability of both the hardware and software interfaces. For software, this would mean adding more flexibility to the interface design - making it so you could easily switch between functions such as meditation, electronic music, dj-ing, or general music education. This way, within one interface you could easily toggle between different *Squishy* worlds. Drawing inspiration from Scratch microworlds, which allow you to start with only a small number of coding blocks and then elect to add more once you have mastered the starting few, I would also like to have the option to start out with only a few functions available and then give the user the choice to add more if they want to. This would be a good addition especially for the educational parts of the *Squishy*, as it would allow users to practice and eventually master a few skills or terms before moving on to new ones. Making these changes to the software interface would also further emphasize the idea that the *Squishy* could grow with you as you learn. If the interface had the option to toggle between beginner and advanced modes, they could easily see what comes next and grow towards it by continuing to use the toys.

Tinkerability in the physical interface would look a little different. To make this possible, I would like to give the users the ability to move sensors around, to make their own shells, or to generally take apart the *Squishies* and put them back together in a different way. This could extend to changing functionality as well - if you want to meditate you may only want one sensor to manipulate, and being able to tinker with the *Squishy* itself would allow for that. You could also introduce other kinds of sensors into the toys - adding something like a gyroscope or a temperature sensor could increase the possibilities for interaction methods for the *Squishies*. A natural continuation of this concept would be creating DIY kits for the *Squishies*. These would give you all the materials you need to make your own *Squishy*, and detailed instructions for doing so. This could even be as simple as publicizing a bill of materials from Adafruit (a company which supports similar projects, has an extensive collection of simple electronics education, and is both MIT-born and woman-owned) and making all the instructions and code used open-source and readily available online.

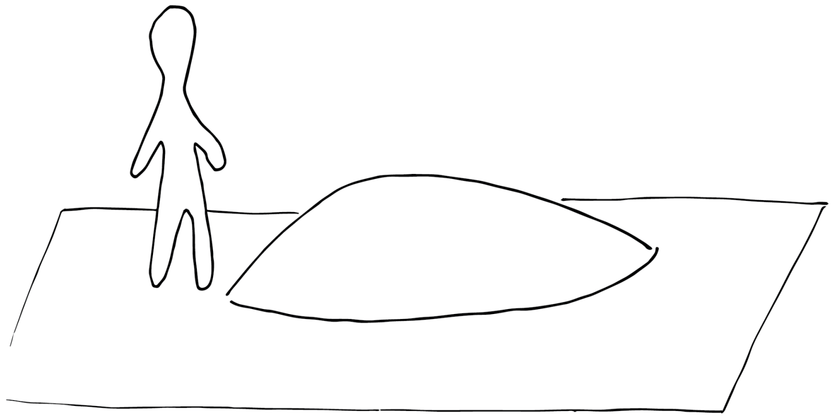


Figure 50: *Florb* concept sketch.

Another dream I have for the *Squishy* world is very large sonic objects one could lie down on top of, or otherwise interact with in a bigger way than just with hands. These giant flopping pads could be part of an art installation-type use case for the *Squishies*, where you have a full room of large *Squishy* objects, or they could be as simple as a sonic bean bag used for relaxing and hanging out. In the simplest form, these giant *Squishies* would simply be large silicone hemispheres that one could flop onto - this version I have been calling *florb* (floor-orb). These could also follow the shapes of the smaller *Squishies* - it would be simply fantastic to have a giant flower you could interact with, for example, or they could take on their own completely different forms. Even just a room with sonic silicone walls and floor could be interesting to play around with.

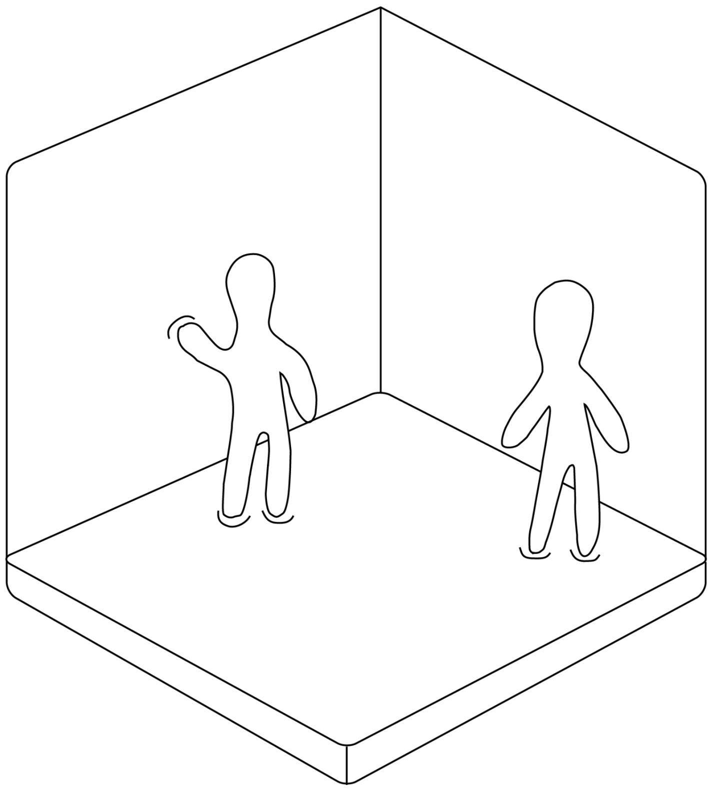


Figure 51: Squishy room concept sketch.

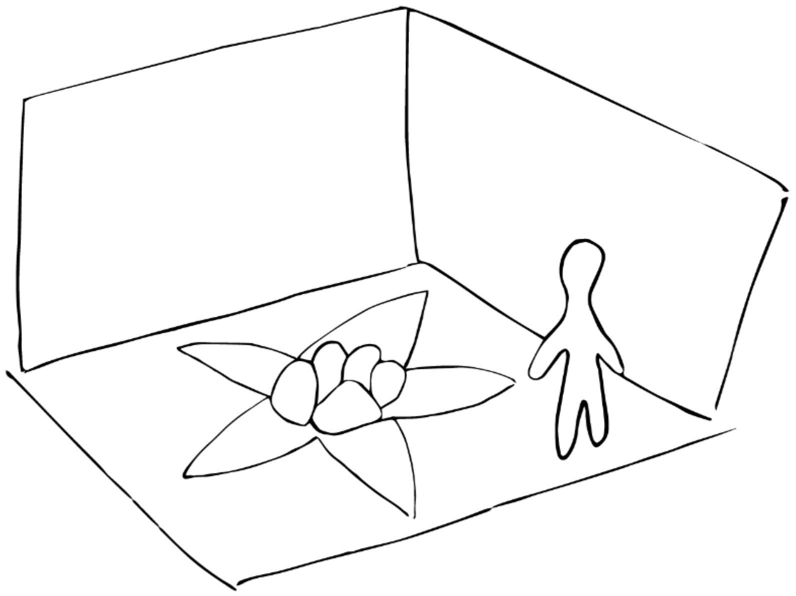


Figure 52: Giant squishy columbine concept sketch.

In terms of evaluation, there is more I would have liked to do with the *Squishies* to fully test their effectiveness. Time constraints and limitations due to COVID-19 did make it tough to do a more rigorous test of the toys, but I will go over some potential ways the evaluation process could be furthered in a pandemic-free future. The main aspect of this would be completing a series of in-person workshops. In person, you would be able to spend a lot more time and focus on music making with users instead of on debugging and interface setup.

For the kid workshops, this would involve inviting participants to sessions with groups of other peers. The main aspect to be added here is group music making, which was not something that was easily achieved via Zoom. It would be great to both test out the ways in which the *Squishies* could work as a group activity, and to create a meaningful bonding experience for participants. Workshops could also be run in school settings with kids. This would involve partnering with local elementary and middle schools to run *Squishy* sessions through general music classes. Running these in schools would be beneficial, as that way the participating children would already know each other, thus eliminating some of the shyness and hesitation I found in Zoom workshop settings.

In-person meditation workshops could also become much more comprehensive experiences. Sessions could be a full hour, encompassing multiple exercises, both as a group and individually. Participants would be evaluated both at the start and the end of the workshop for stress and anxiety levels, so results could be compared. To go even deeper, bio data like heart rate or blood pressure levels could also be recorded throughout the experience in an attempt to measure stress levels.

For all of the demographics I am addressing with the *Squishy* user tests, I would have also liked to run longer-term studies with users. It would be incredibly valuable to have a child use a *Squishy* for a few months - this would let me see if there is a long-term play value for the toy. To execute this, participants would answer questions about musical experience and interest at both the start and the end of the study, so we could properly evaluate what they gain from using the *Squishies*. A similar structure would be used for adult meditation workshops - where participants' experiences and comfort levels with meditation are evaluated at both ends of the study, and they spend several months completing exercises with the *Squishy*. For high-level users, this long-term study might take the form of a more comprehensive piece, or other final result from the *Squishy*. This could even culminate in a concert experience for the participants, where a huge breadth of work would be presented, really showcasing what the *Squishies* can do in a professional music setting.

The final thing I would have liked to draw from my user studies is more concrete musical examples of things created by participants. This is especially relevant to the kid workshops, as the *Squishies* are educational experiences, and it would be useful if I could show a definite increase in skill from the start to the end of the study period. By collecting musical samples from participants, I would be able to compare pieces they created at the start of the study to those they made after spending a lot of time with the *Squishy*. Looking at aspects like musical complexity, how many aspects of the interface they utilized, and generally how nice the sound is would say a lot about what kids picked up during the study, and what they did not.

8

Conclusion

Key Contributions

To wrap up, the *Squishies* are a novel musical experience. They have created an entirely new method for music education that fills the need for a low-level, low-stress way to learn electronic music. Previously, there was no clear method for electronic music education that was easily accessible - both in terms of cost and in path to entry - but the *Squishies* provide this through a simple physical interface that almost anyone can use, and an intuitive software patch that guides users through basic concepts of electronic music. The *Squishies* are also an innovative new kind of music controller for high-level users, which is as easy to use as any other MIDI instrument one might buy, but with a soft form factor that will not be found in other high-level controllers. They can also act as a mindfulness tool to aid in mediation, by guiding users through both sound and touch at the same time - which is also not something that is found in other meditation tools.

This thesis has explored the ways in which all of these applications for the *Squishy* could be possible. It has tested the effectiveness of the *Squishies* through a series of user tests consisting of people from all applicable groups - children aged 8-14 eager to learn about electronic music, experienced musicians willing to explore new interfaces, and stressed-out graduate students looking for a chance to relax. The *Squishies* even have the potential to be released as a product, or as an open-source project encompassing all three of these applications. They could even become a DIY experience, enabling people of all ages to create their own musical objects through a *Squishy* kit, or through a bill of materials (see Appendix II).

To summarize, I'll outline the key contributions of this thesis. The *Squishies* are:

- A novel way to learn electronic music, which is cheaper and more accessible than other existing methods.
- A new kind of tool to aid in meditation and encourage mindfulness.
- A soft MIDI controller that allows experienced electronic musicians to create music in a fluid way they may not have before.
- A path to making music that encourages freeform sound exploration through a soft and squishy interface.

Looking Forward

My hope is that the *Squishies* will set a precedent for other new musical experiences, pedagogies, and instruments, and that we will continue to work towards a more accessible music world. On a slightly different note, I also hope the *Squishies* will inspire more people to make soft interfaces. I imagine a world where everything from car interiors to furniture to kitchen utensils are soft and squeeze-able. It would be incredibly fun if everything you touched had sonic components that could be triggered when desired. I hope that this thesis can inspire others to feel the same way, and to re-imagine their worlds in more exciting, fun, ways that stretch beyond just adding screens to anything and everything.

Appendix A

Squishy User Guide

In your package:

- Squishy Toy
- Electronics box (connected to toy)
- USB to Micro USB Cable
- Return envelope
- Label paper (for printing return label)

Assembly:

Plug your USB to micro-USB cable into the electronics box and your computer.

Note: Please do not open the electronics box at any point.

Computer Setup:

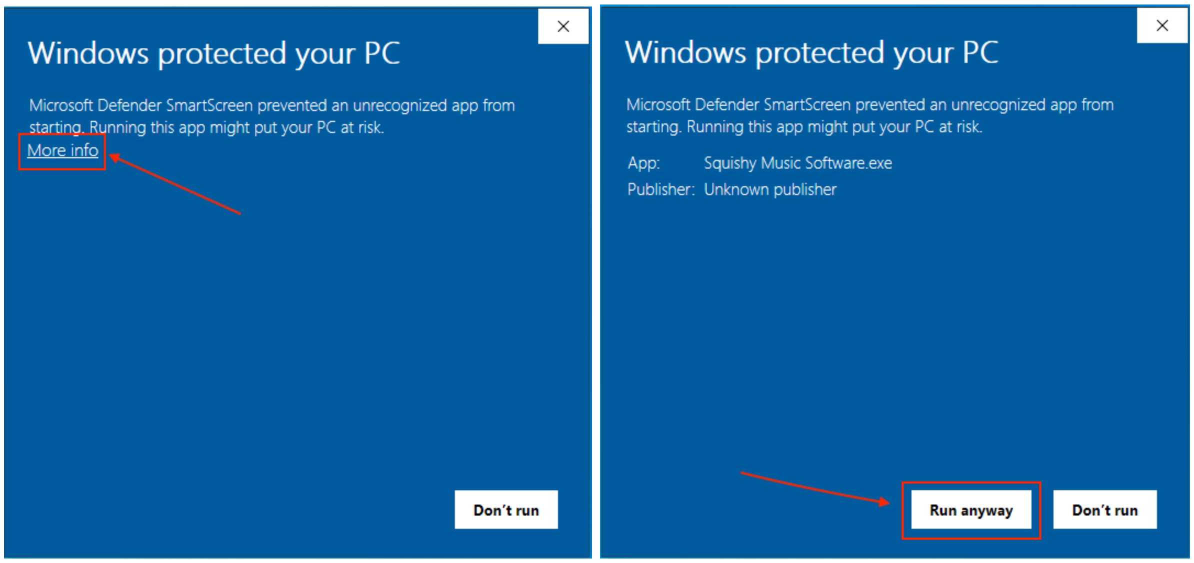
Step 1:

Download the computer app emailed to you. **Remove it from your Downloads** folder and put it in a new folder on your Desktop.

Open the app. You may have to allow your computer to open it in security settings.

To do so on **Windows**:

When you get this error, press “More Info”, then hit “Run Anyway”.


Windows protected your PC

Microsoft Defender SmartScreen prevented an unrecognized app from starting. Running this app might put your PC at risk.

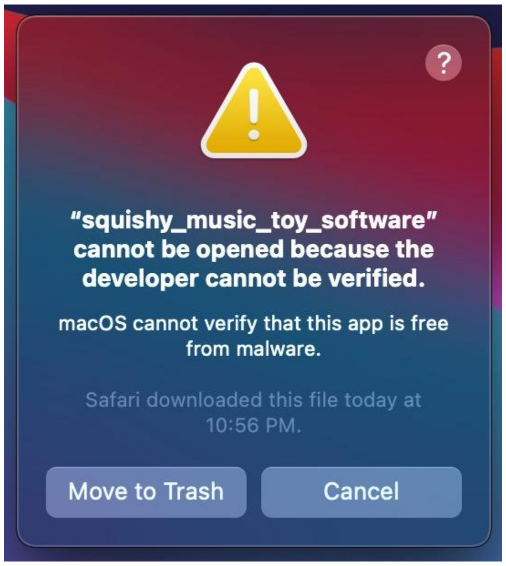
Windows protected your PC

Microsoft Defender SmartScreen prevented an unrecognized app from starting. Running this app might put your PC at risk.

App: Squishy Music Software.exe
Publisher: Unknown publisher

On a **Mac**:

You will see something like this:

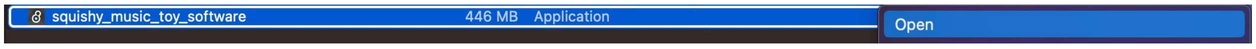

"squishy_music_toy_software" cannot be opened because the developer cannot be verified.

macOS cannot verify that this app is free from malware.

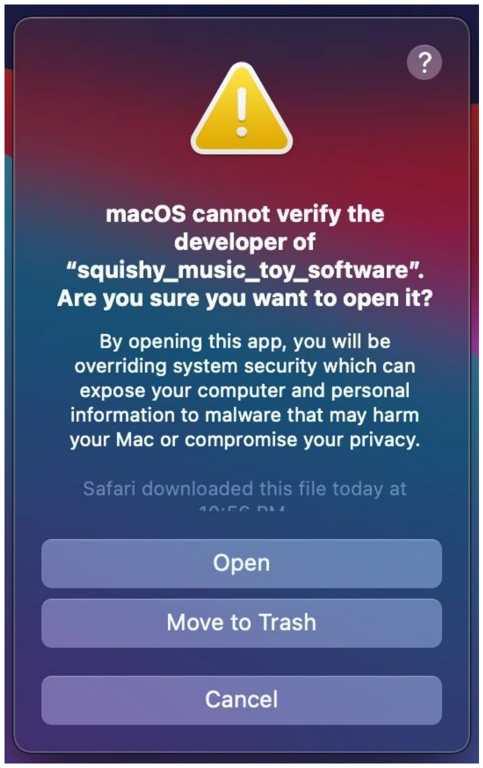
Safari downloaded this file today at 10:56 PM.

To get around this, you need to open Finder → Downloads, and look for the file you just downloaded ("Squishy Music Software").

Right-click on the software, and click "Open"



You will get a screen that looks like this:



macOS cannot verify the developer of "squishy_music_toy_software". Are you sure you want to open it?

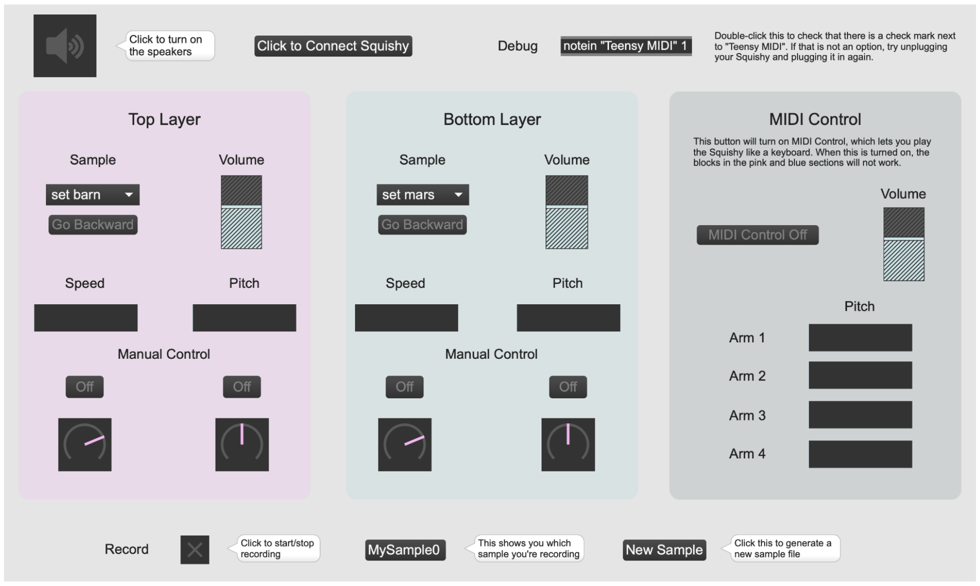
By opening this app, you will be overriding system security which can expose your computer and personal information to malware that may harm your Mac or compromise your privacy.

Safari downloaded this file today at

Click "Open" and you should be all set.

Step 2:

Open the computer app. It will look something like this:

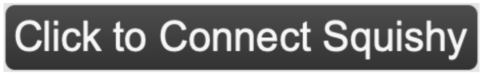


Double-click this to check that there is a check mark next to "Teensy MIDI". If that is not an option, try unplugging your Squishy and plugging it in again.

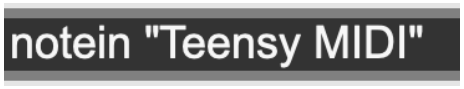
This button will turn on MIDI Control, which lets you play the Squishy like a keyboard. When this is turned on, the blocks in the pink and blue sections will not work.

Step 3:

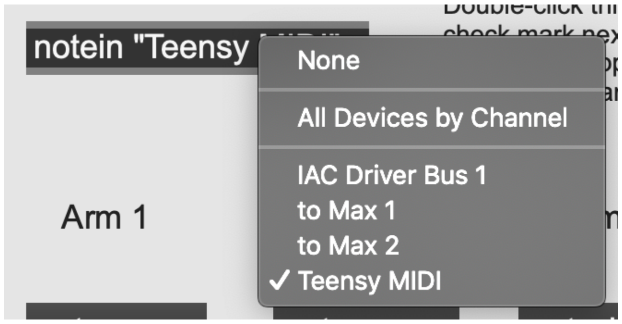
Check to make sure your toy is connected - first by clicking this button:



And then you can double check by double clicking on this button:



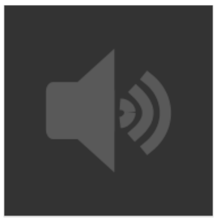
You should see a drop down list like this with a check mark next to “Teensy MIDI”.

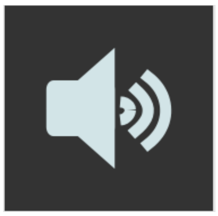


If you don’t see this as an option, check the connection of your USB cable on both ends to make sure they’re both fully connected. Then, unplug and replug your cable and it should appear.

Step 4:

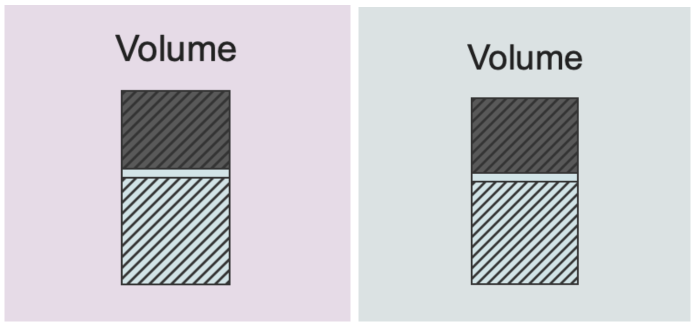
Toggle the speaker button to make sure your speakers are connected to the software. It should turn light blue when the speakers are connected:


Off


On

If your computer asks you for speaker and microphone access, just hit "Allow".

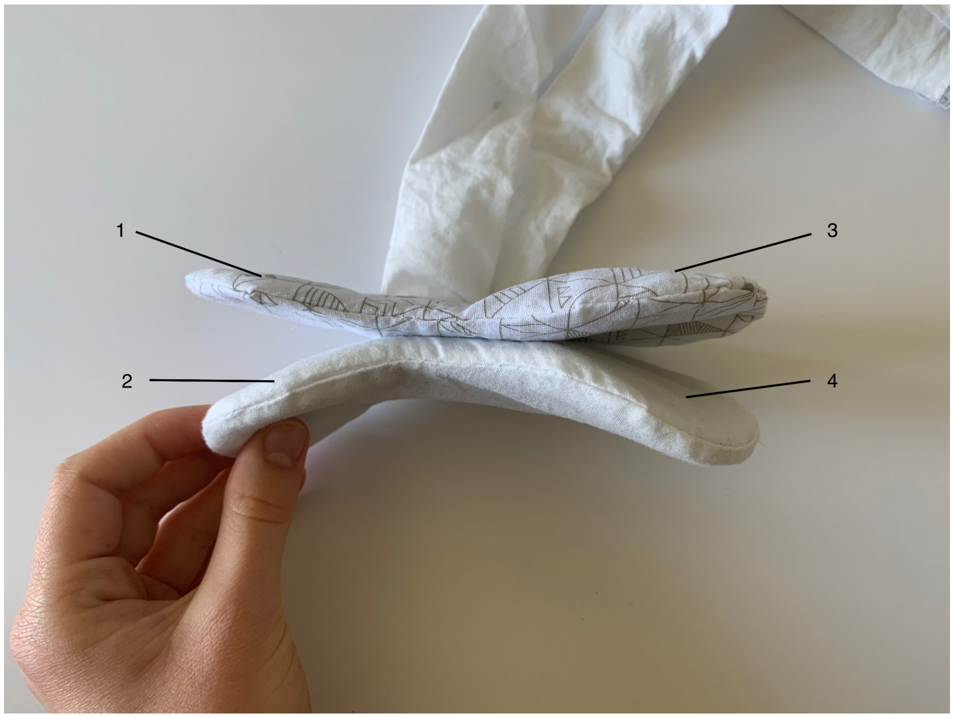
Turn up the volume sliders on your screen...



...and try squeezing it to see if any sound comes out. You may also want to turn up your computer sound. If you hear something, you're all set! You can start playing around with the interface to make some cool sounds!

For best results, use your toy with the **patterned side facing upwards**.

Each Squishy has 4 areas that respond to your touch - one in each of the big ends of your toy (patterned and not patterned), and one in each of the small ends.

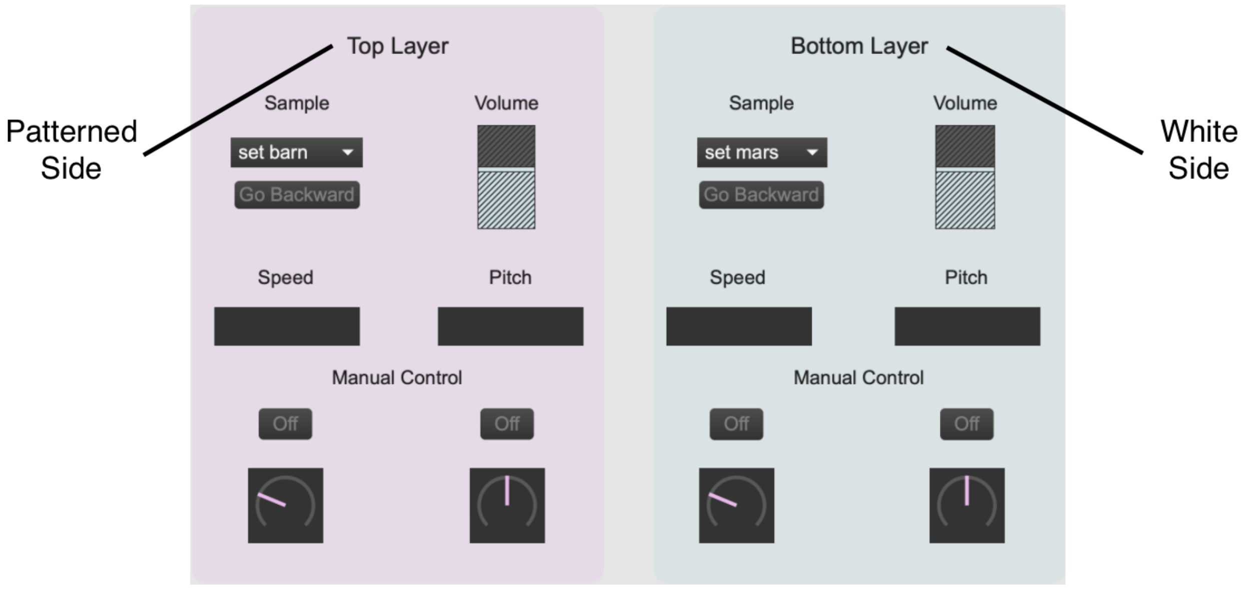


If you run into any problems, please feel free to reach out to me - the toys are prototypes, so they can be a little tricky sometimes :)

Below this, I'll add some extra info about how to use the toys and the interface. We'll go over all of this during the first workshop as well.

Software Interface Guide:

The interface is split into 3 areas - pink, blue, and gray. We'll focus on the first two to begin with (pink and blue), which are labelled as Top Layer and Bottom Layer.

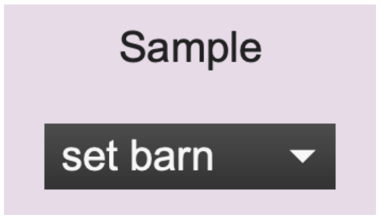


Top Layer here means the top layer of your Squishy, which is the patterned side. Bottom Layer means the bottom layer of your toy, which is the white side.

Everything in the pink box only controls the patterned side of your Squishy, and everything in the blue blox only controls the white side.

Sample Choice:

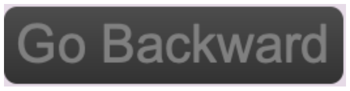
The first button in the box will let you select music clips (samples) to play with. There are a few pre-loaded ones here, and one empty one called 'mysample' that we'll talk about later on.

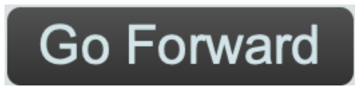


Your squishy toy will change the speed and pitch of the sample you select. As you squeeze harder, the music will go faster, and the pitch will increase.

Forward/Backward Switch:

Next, there is a button labeled "Go Backward". Clicking this will let you play the sample backwards. When you want to go back to playing normally, you can press it again and it will go back to playing forwards.

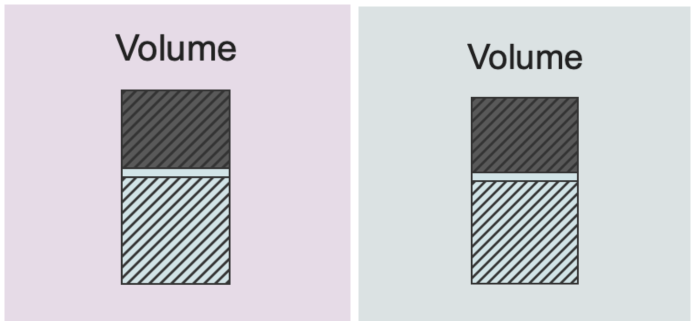




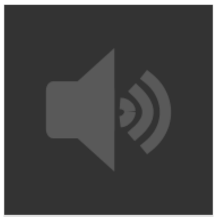
To hear the change after you hit the switch, you have to squeeze the toy arm that connects to the switch.

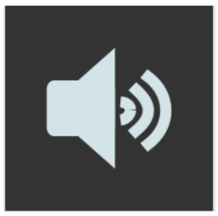
Volume:

These are the volume sliders we mentioned in the setup guide. The blue bar represents how loud each of the samples will play. Pull a slider all the way down to mute that box completely.



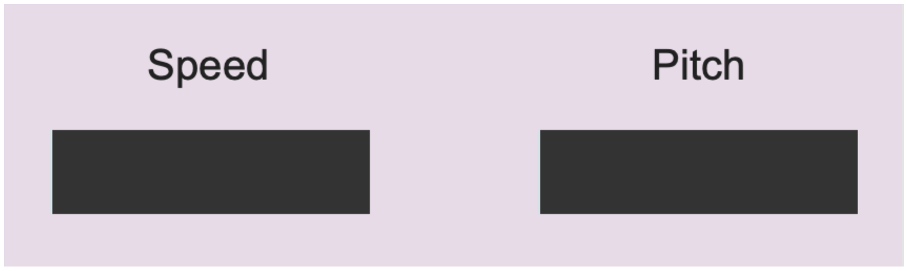
The speaker icon at the bottom turns on and off your speakers completely. Turning off your speakers will turn off all sounds coming from the software.


Off


On

Speed/Pitch Bars:

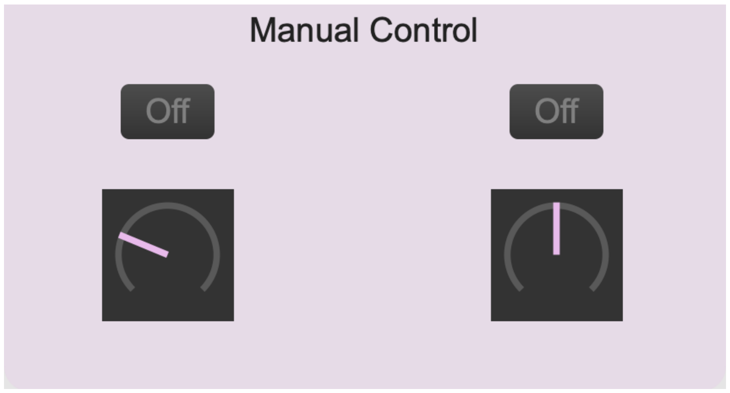
These two bars show you how hard you are pressing on each area of the Squishy. The bar will go up as you squeeze harder. The speed one shows you how much you are increasing the speed of the sample you've chosen, and the pitch one shows you how much you are increasing the pitch of the sample.



One of these will be mapped to the large end of your patterned or white side, and one will be mapped to the small end of your patterned or white side.

Manual Control:

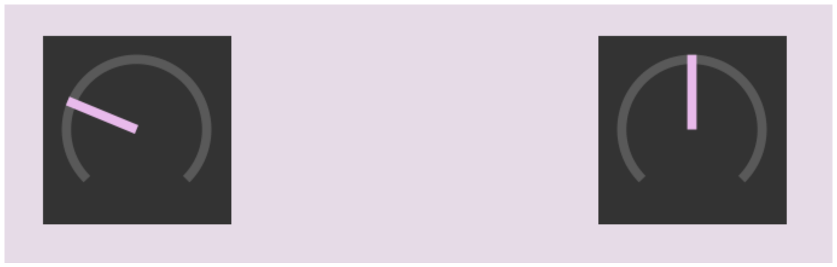
The last area of your pink and blue boxes only apply to manual mode. This is the mode you can trigger if you want to try changing the music without using the Squishy.



You can turn it on for only pitch or speed, or both, by pressing this button:

The left button will turn on manual mode for speed, the right will turn it on for pitch.

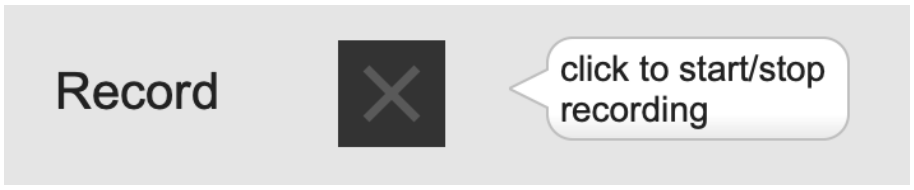
Below that are two dials, which will let you control the speed and pitch manually. Whatever you turn the dial to will be the speed or pitch of the music you're hearing.



Speed Pitch

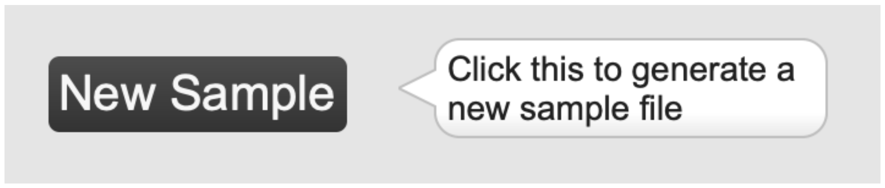
Recording:

The area at the bottom of your interface will let you record your own audio sample to manipulate with the toy! To record a sound, press the button once to start, and once after you're done.

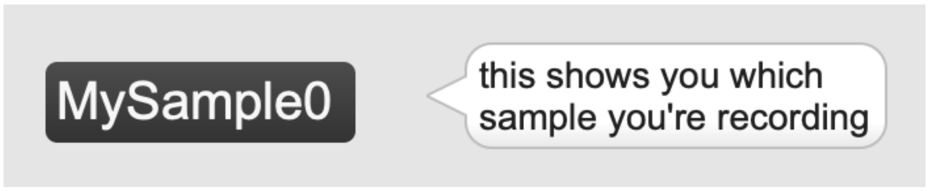


Make sure your other channels are muted (volume bars all set to zero) before you record! Otherwise you will record the sounds coming from your computer too.

There are 6 empty spaces for you to record your samples: MySample0-MySample5. You can toggle between these by hitting the "New Sample" Button in the recording section.



You can see which sample you're recording to on this button:

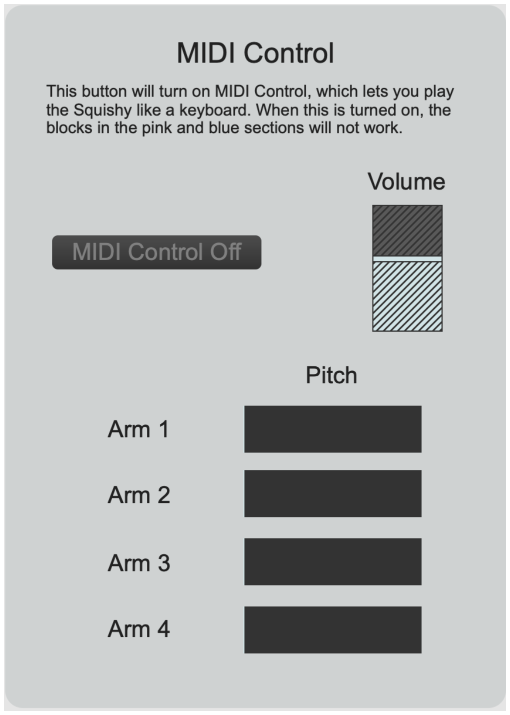


To access your sample, select the option of 'set MySample0-setMySample5' from the Sample menu of your choice.

Your recordings will be saved on your computer as sound files, so they will still be there if you close and open the software.

MIDI Control:

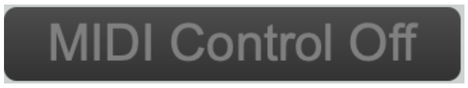
Now we'll talk about the gray box on your screen, labeled MIDI Control.


MIDI Control

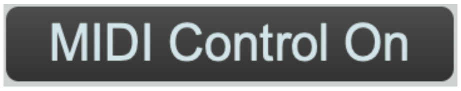
This button will turn on MIDI Control, which lets you play the Squishy like a keyboard. When this is turned on, the blocks in the pink and blue sections will not work.

This area lets you play your Squishy like you would an electronic piano keyboard. A MIDI note is the kind of sound that comes out of these keyboards. When this mode is on, you will hear only pitches coming out of your Squishy - not samples like in the pink and blue boxes.

To turn on MIDI Control, press this button:



It's on when it looks like this:



The bars below it labelled Arms 1-4 will show you how hard you're squeezing each arm. The harder you squeeze, the higher the pitches will sound.

Pitch
Arm 1
Arm 2
Arm 3
Arm 4

Suggested First Tests:

It helps to first play around in manual mode to get a feel for the different samples. I'd recommend setting all the channels to manual to find a few samples you like.

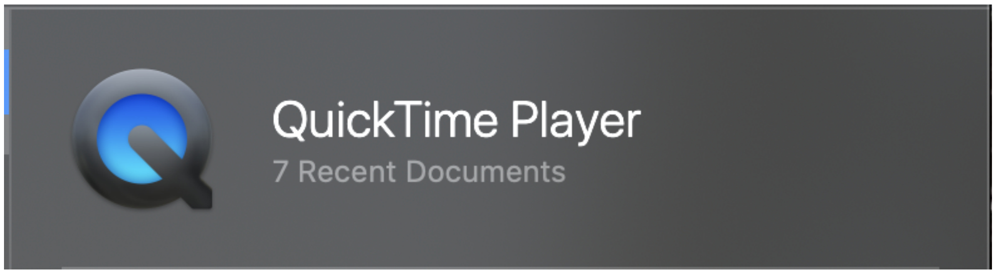
A fun way to test the record button is with the *Ten Second Test*:

> Record yourself counting to ten using the Record button. Then, for one of the columns, pick the 'set mysample' from the Sample menu. Play around with your voice using the manual mode, and then with the Squishy! Try to get your voice to go super slow and then super fast.

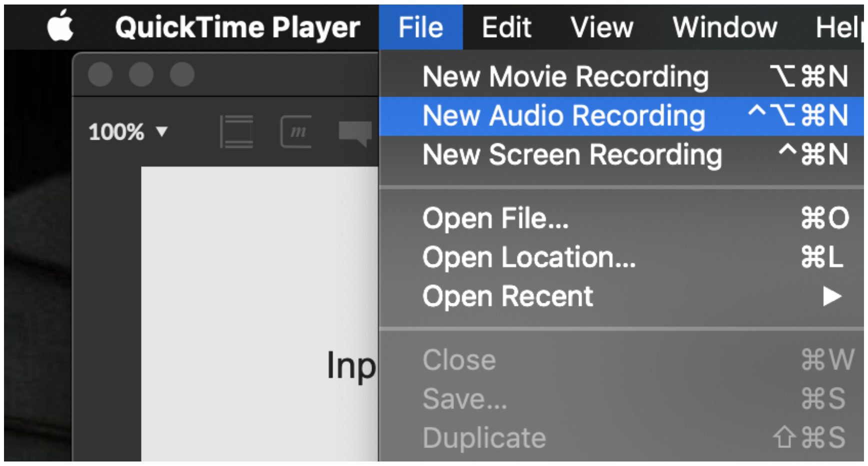
QuickTime Player Screen Record:

Step 1:

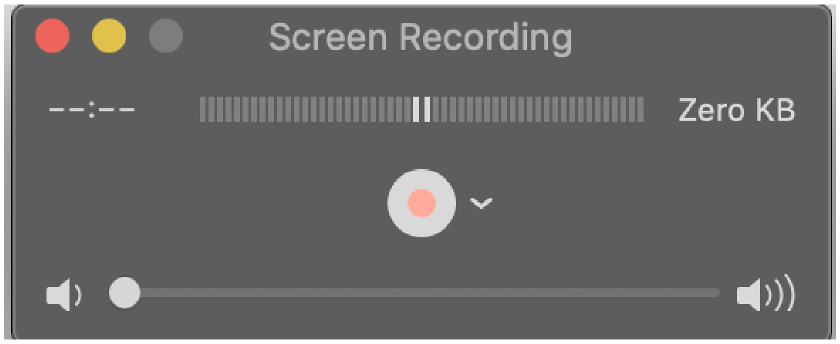
Open QuickTime Player



Step 2:
Click on Edit ⟶ New Screen Recording (or Audio Recording!)

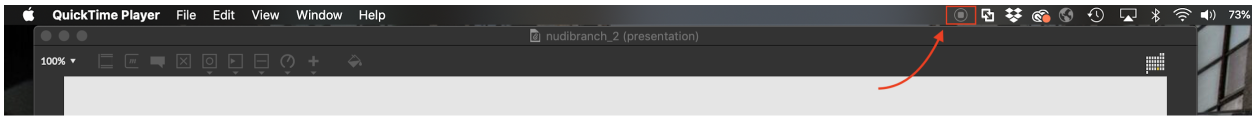


Step 3:
Click on the big red button to start your recording.



Step 4:
Play your squishy sounds!

Step 5:
Stop your recording by hitting the button in your menu bar:



Step 6:
Save your recording, and you should be all set!

Recording on a Chromebook:

https://support.google.com/chromebook/answer/10474268?hl=en

Appendix B

Additional Work

This section encompasses the additional work I completed during the summer of 2021 in collaboration with my Undergraduate Research Assistant, Jenny Zhao. We were exploring the potential for the *Squishies* to be a DIY experience that a user could construct at home, from a kit, or from a list of materials that could be obtained from Adafruit or DigiKey.

Bill of Materials

For the sensors:

- Felt
- Velostat
- Conductive thread
- Regular thread
- Sewing needle

For the breadboard:

- Teensy LC
- Breadboard or Protoboard
- Ribbon cables
- Connector pins
- Jumper cables
- 10K Resistors
- Headers

Useful tools:

- Hot glue gun

- Soldering iron
- Solder wire
- Soldering iron tip cleaner
- Solder sucker
- Wire cutters
- Pliers

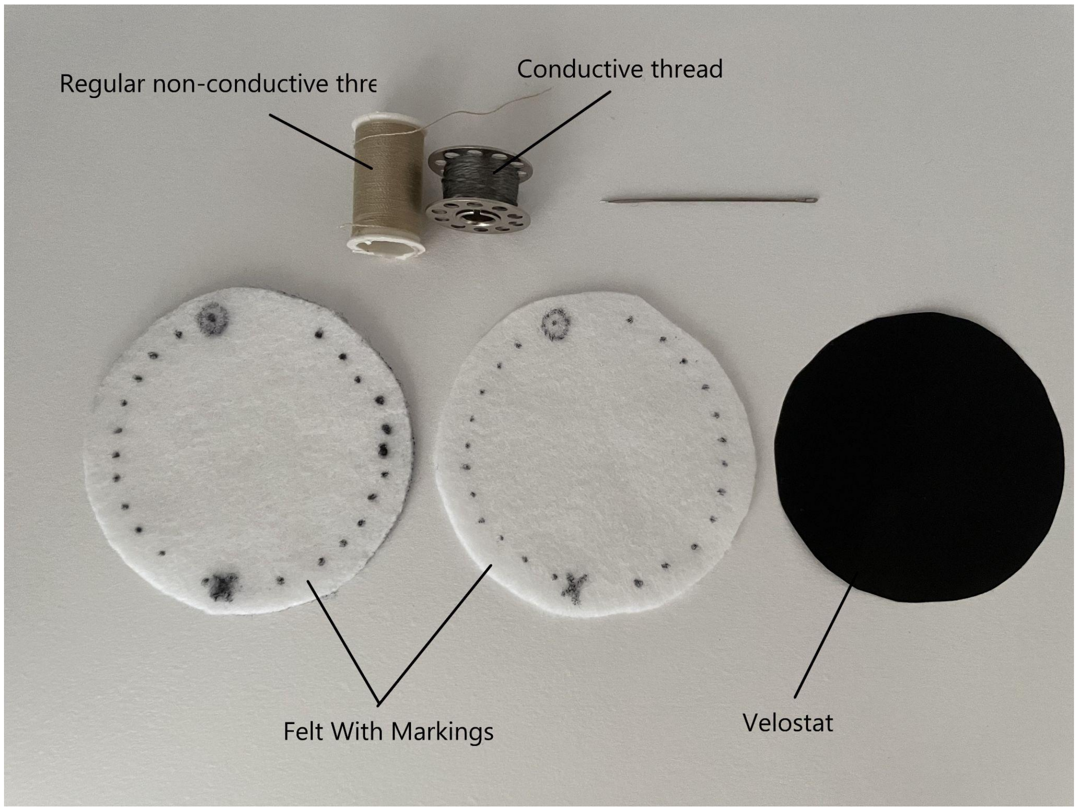
Templates:

- Small circle sensor template
- Large circle sensor template

Squishy Toy Construction Instructions

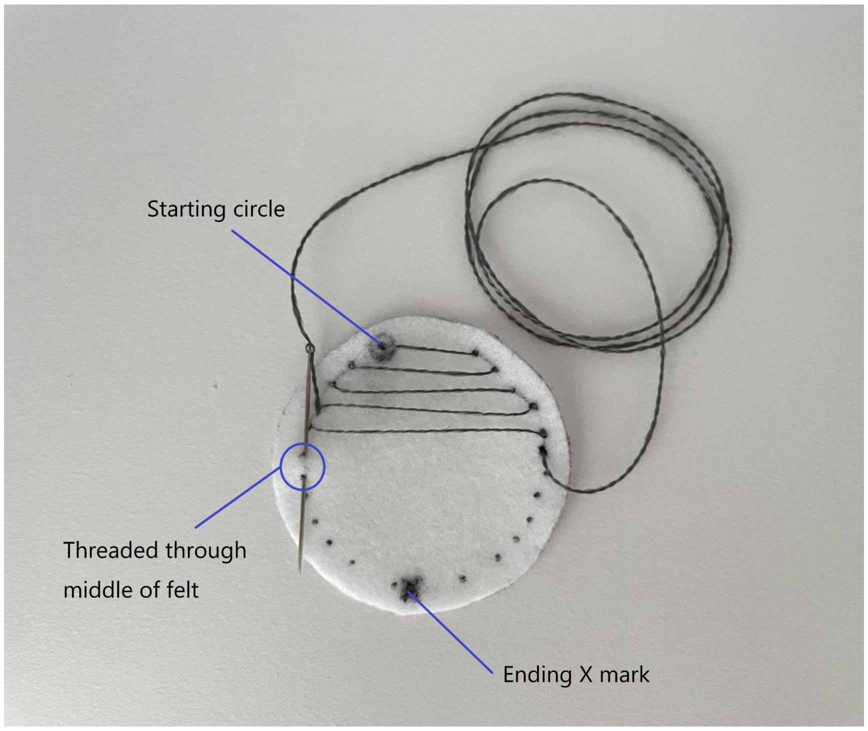
Gather your sensor materials. To start, you need to use the sensor template to cut out two pieces of felt for each sensor you want to make. Use the larger circle in the provided template for this. Next, you should use the edges of the lines on the smaller circle template to mark off where your threads will go with a dark colored pen. You then need to cut out one piece of your Velostat, again using the outer edge of the smaller circle template as a guide.

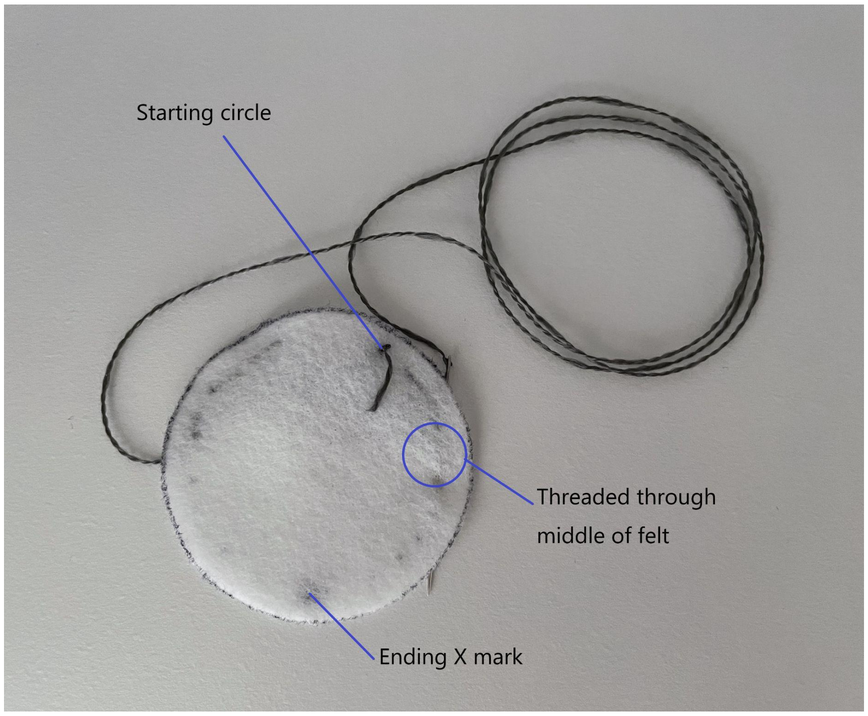
Then, you need to prepare your conductive thread for sewing the sensor. You will need roughly 1.5 feet of conductive thread for each half of the small circular sensor and 2 feet for each half of the large circular sensor.

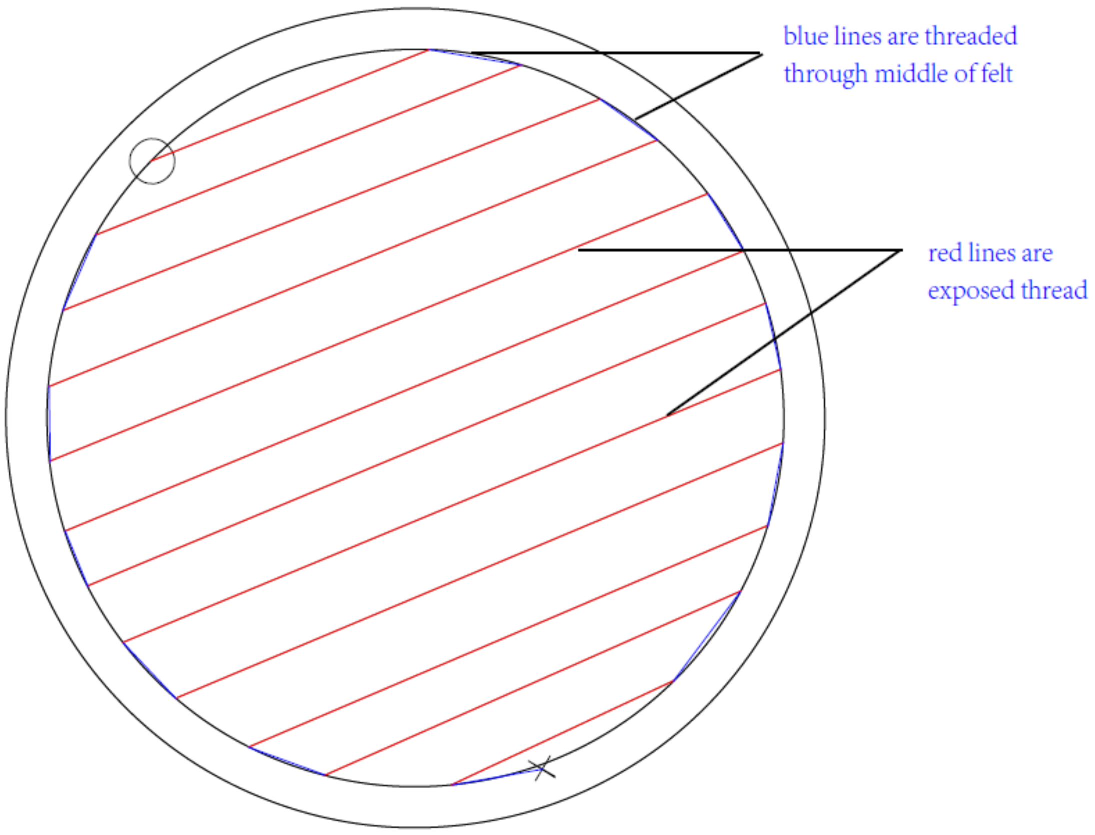


Thread your needle with conductive thread and tie a knot at the end of the thread. Run the needle through the felt starting at the circled marking (see above diagram).

Thread the needle through the middle layer of the felt between the two dot markings on the opposite side of the circled dot. Follow the pattern of exposed conductive thread and middle layer unexposed conductive thread according to the diagram below. Note - don't sew the conductive thread too closely to the edge of the felt. The velostat needs to easily cover all of the conductive thread.

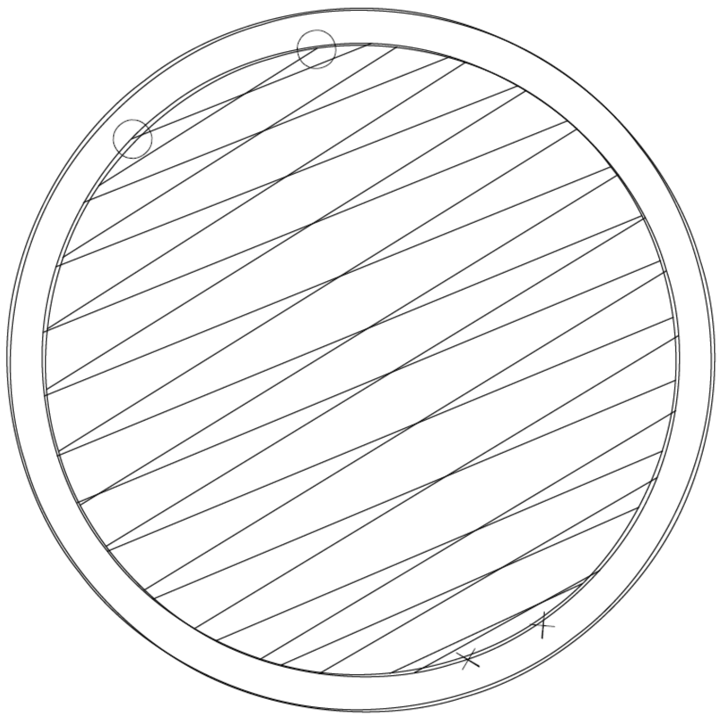

Starting circle
Threaded through middle of felt
Ending X mark


Starting circle
Threaded through middle of felt
Ending X mark


blue lines are threaded through middle of felt
red lines are exposed thread

Once you reach the last normal dot, thread through the middle of the felt and thread out the backside of the X. There should be a long conductive thread hanging out of the side without the exposed conductive thread.

After threading both pieces of felt with conductive thread, arrange the two felt pieces such that the X marks are close but not on top of each other. You should notice that the exposed conductive thread crosses the exposed thread on the other felt and does not completely lie on top of one another. The outside of the two pieces should not have exposed conductive thread except for the long hanging strands. It should form a cross pattern similar to the one below.



Slide a piece of velostat between the two pieces of felt, and make sure the three circular pieces are aligned. The velostat should be a bit smaller than the felt circles, but should **completely cover** all the conductive thread. This is important to prevent short circuits.

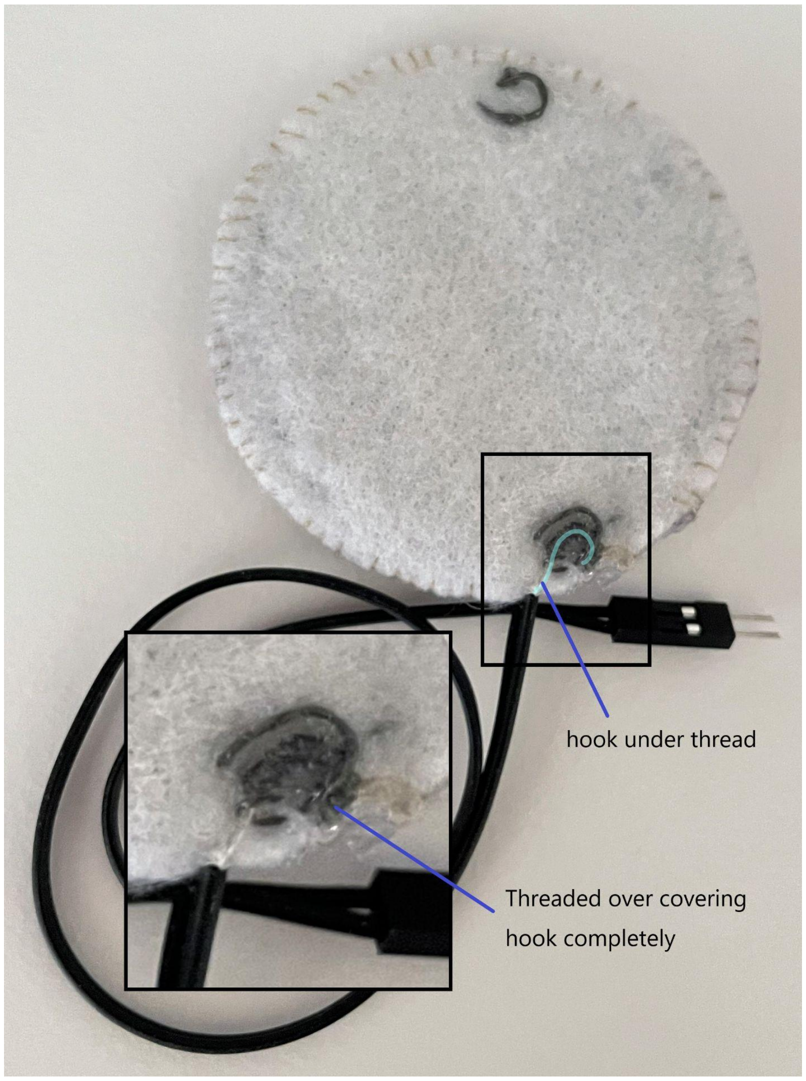
Cut regular non-conductive thread a bit larger than twice the circumference of the sensor. Thread and knot the regular thread onto a needle. Starting slightly outside of the two conductive threads are hanging out, sew together the two felt flaps using the whipstitch.

Continue sewing the sensors together along the outside border. Try not to sew onto the velostat - but it is fine if you have to puncture it a few times. Stop sewing once you hit the part where the conductive thread hangs out - just before where you started - and tie a knot to end the regular thread there.

Next we need to prepare the ribbon cables. Split your grouped cable in two - so you have two sets of two wires. Strip both ends of each of your cables - you should take off about ⅜”. On one end of the cable, affix male pin connectors to each wire, and set them into a housing to make a two-pronged connector. On the other end, twist and bend each open end into a loop or hook shape.

Rethread the hanging conductive thread through a needle and continuously loop around the hook through the middle of the felt to keep the hook in place and ensure a connection between the conductive thread and the ribbon cable.

After thoroughly looping around the hook such that it stays in place on the felt, knot the conductive thread and trim the extra thread. Repeat with the hook on the other side as well. Be careful not to let the two sides touch each other or overlap at all - this will result in a short circuit.



Warm up the hot glue gun. Glue will be applied to all places where there is exposed conductive thread on the outside of the felt. Make sure the hot glue covers all of the exposed thread. Try not to use too much hot glue or try to flatten the glue to make sure there are no huge lumps on the outside of the sensor.

You can find more great information on creating soft sensors like these on Hannah Perner-Wilson's website.

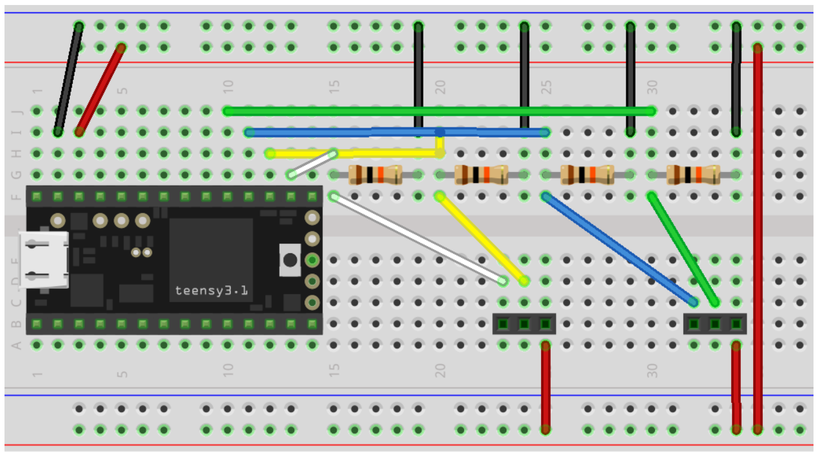
Hardware and Breadboard Instructions

Gather the protoboard, Teensy, headers, jumper wires, resistors, and all of your soldering equipment.

The first thing you need to do is solder male headers to the Teensy board. Cut the headers to the correct length, and set them in a breadboard with the Teensy to solder (doing this in a breadboard ensures that your headers will not come out crooked).

Next, you should attach female headers where shown in the breadboard diagram below - there should be two places for the sensor cables, and two for the Teensy. This is easily done by using masking tape to hold the headers in place while you solder.

Create the rest of the breadboard by following the diagram below. Note: the color of the jumper wires may be different. Make sure that the jumper wires are connected to the correct Teensy labels. Different Teensy's may have different ordering of pins. You can find diagrams of their different boards on their website.



Take the two male headers from ribbon cable connecting to a sensor and place in the rightmost two holes of the open female headers on the board.

Now you are ready to test your teensy!

You can find Teensy code that works with these sensors here.

Appendix C

Additional Media

All of the videos linked throughout this paper can be found here:

https://web.media.mit.edu/~hrl/

Prior Work:

https://web.media.mit.edu/~hrl/prior_work.html

High-Level User Tests:

https://web.media.mit.edu/~hrl/squishy_collab.html

DIY Resources:

https://web.media.mit.edu/~hrl/DIY.html

References

Introduction

1. Jan Walker and June Boyce-Tillman (2002), Music Lessons on prescription? The impact of music lessons for children with chronic anxiety problems, Health Education – The Arts and Health, Vol 102, No 4 2002, pp172-9 ISSN 0965-4283

2. Rauscher, Frances H., and Sean C. Hinton. "Music Instruction and its Diverse Extra-Musical Benefits." Music Perception 29.2 (2011): 215-226.

3. Brown, Laura Lewis. "The benefits of music education." PBS KIDS for Parents (2012).

4. Yuichiro Kinoshita, Masato Nishio, Shota Shiragaand Kentaro Go. “Investigation Of Pleasantness In The Manipulation Of Deformable Interfaces For Musical Expression.” 7th International Conference On Kansei Engineering And Emotion Research (2018). https://ep.liu.se/ecp/146/034/ecp18146034.pdf.

5. Rosenbaum, Eric. Explorations in Musical Tinkering. 2015. Program of Media Arts and Sciences, MIT, PhD dissertation.

6. “Theremin.” Encyclopædia Britannica, Encyclopædia Britannica, Inc., www.britannica.com/art/theremin.

7. Russell, James. “Children Still Face Barriers in Accessing Music Education.” Phys.org, Phys.org, 25 Apr. 2016, phys.org/news/2016-04-children-barriers-accessing-music.html.

8. Albert, Daniel J. “Socioeconomic Status and Instrumental Music: What Does the Research Say about the Relationship and Its Implications?” Update: Applications of Research in Music Education, vol. 25, no. 1, Nov. 2006, pp. 39–45, doi:10.1177/87551233060250010105.

9. Boucher, Hélène, and Charlene A. Ryan. “Performance Stress and the Very Young Musician.” Journal of Research in Music Education, vol. 58, no. 4, 2011, pp. 329–345. JSTOR, www.jstor.org/stable/40961658. Accessed 21 Oct. 2020.

10. Fehm, Lydia, and Katja Schmidt. “Performance anxiety in gifted adolescent musicians.” Journal of anxiety disorders vol. 20,1 (2006): 98-109. doi:10.1016/j.janxdis.2004.11.011

Related Work

11. Machover, Tod. “Toy Symphony.” Toysymphony.net, 2002, toysymphony.net/.

12. Orth, Margaret Ann. "Sculpted computational objects with smart and active computing materials." 2001. Program of Media Arts and Sciences, MIT, PhD dissertation.

13. Weinberg, Gil. “Expressive digital musical instruments for children”. 1999. Program of Media Arts and Sciences, Massachusetts Institute of Technology, PhD dissertation.

14. Farbood, Mary, and Egon Pasztor. “Hyperscore.” Toysymphony.net, 2002, toysymphony.net/.

15. Jordà, S, et al. “Reactable - Music Knowledge Technology.” Reactable, 2003, reactable.com/.

16. Tosa, Masamichi, and Nobumichi Tosa. Otamatone, 22 Jan. 2021, otamatone.com/.

17. Skulina, David, and Ben Schögler. “Accessible Musical Instrument for Everyone.” *SkoogMusic*, 10 Nov. 2020, skoogmusic.com/.

18. “Blipblox Synthesizers.” Blipblox by Playtime Engineering, 2018, blipblox.com/.

19. “Minibar - Liquid Analyser.” Rainger FX, 24 Feb. 2021, www.raingerfx.com/product/minibar-liquid-analyser/.

20. Bleep, Dr. “The Thingamagoop 3000.” Bleep Labs, bleeplabs.com/product/thingamagoop-3000/.

21. Bleep, Dr. “Original Thingamagoop.” Bleep Labs, bleeplabs.com/product/original_thingamagoop/.

22. “Imagine, Program, Share.” Scratch, scratch.mit.edu/.

23. Silver, Jay, and Eric Rosenbaum. “Makey Makey.” Joylabz Official Makey Makey Store, 2012, makeymakey.com/.

24. Silver, Jay. “Draw Music with a Pencil, or a Kitchen Sink, or a Banana, or…” DRAWDIO, 26 June 2011, drawdio.com/.

25. Industries, Adafruit. “Drawdio.” Adafruit Industries, www.adafruit.com/product/124.

26. Qi, Jie. *Chibitronics*, chibitronics.com/.

27. Perner-Wilson, Hannah, and Mika Satomi . “HOW TO GET WHAT YOU WANT.” Kobakant, 2008, www.kobakant.at/DIY/.

28. McKenzie, Steven. “Teddy Tech: Rise of MIT's Robot Care Bear Huggable.” BBC News, BBC, 29 Oct. 2014, www.bbc.com/news/uk-scotland-highlands-islands-29786696.

29. “My Keepon.” BeatBots, 2011, beatbots.net/my-keepon.

30. “Ploomi.” BeatBots, 2011, beatbots.net/ploomi.

31. “About the Suzuki Method.” Suzuki Association of the Americas, suzukiassociation.org/about/suzuki-method/.

32. Dudamel, Gustavo. “Youth Orchestra Los Angeles.” LA Phil, 2007, www.laphil.com/learn/yola/youth-orchestra-los-angeles.

33. American Orff-Schulwerk Association, 25 Mar. 2021, aosa.org/.

34. MasterClass. “How the Suzuki Method Works: Inside the Suzuki Philosophy.” MasterClass, MasterClass, 4 Dec. 2020, www.masterclass.com/articles/suzuki-method-explained#quiz-0.

35. “About AOSA.” American Orff-Schulwerk Association, 20 Jan. 2020, aosa.org/about/about-aosa/.

36. “The Birth and Rise of the Laptop Orchestra.” Artful Design: Technology in Search of the Sublime, a Musicomic Manifesto, by Ge Wang, Stanford University Press, 2018, pp. 248–274.

37. Lienhard, Hannah Rhiannon. A stress-reducing explorable musical object. Massachusetts Institute of Technology, 2019.

38. Holbrow, Charles, Elena Jessop, and Rébecca Kleinberger. "Vocal vibrations." Proceedings of NIME. 2014.

The Squishy

39. Raspberry Pi. “Buy a Raspberry PI 400 Personal Computer Kit.” Raspberry Pi, 2020, www.raspberrypi.org/products/raspberry-pi-400/.

Approach

40. “Nudibranch.” Encyclopædia Britannica, Encyclopædia Britannica, Inc., www.britannica.com/animal/nudibranch.

41. Sartore, Joel. “Nudibranchs.” National Geographic, www.nationalgeographic.com/animals/invertebrates/facts/nudibranchs-1.

42. Abrusci, Giacomo. “Nudibranchs in Indonesia! Jack's October Underwater Photography Feature.” SEVENSEAS Media, 2 Dec. 2019, sevenseasmedia.org/nudibranchs-in-indonesia-jacks-october-underwater-photography-feature/.

43. “Pure Data.” Pure Data - Pd Community Site, puredata.info/.

44. “Aparillo.” Sugar Bytes, sugar-bytes.de/aparillo.

45. “Teensy® USB Development Board.” PJRC, www.pjrc.com/teensy/.

46. “Max 8.” Cycling '74, cycling74.com/products/max-features.

Other

47. Boulanger, Adam. Autism, new music technologies and cognition. Diss. Massachusetts Institute of Technology, 2006.

48. Wistort, Ryan, and Cynthia Breazeal. "TOFU: a socially expressive robot character for child interaction." Proceedings of the 8th international conference on interaction design and children. 2009.

49. Troiano, Giovanni Maria, Esben Warming Pedersen, and Kasper Hornbæk. "Deformable interfaces for performing music." Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems. 2015.

50. Pigeon, Dr. Ir. Stéphane. Irish Coast • Ocean Waves, Wind and Rain Noise Generator, mynoise.net/NoiseMachines/windSeaRainNoiseGenerator.php.

51. Peppler, Kylie. "Equity and access in out-of-school music making." The Oxford handbook of technology and music education. 2017.

52. Chang, Angela, and Hiroshi Ishii. "Zstretch: a stretchy fabric music controller." Proceedings of the 7th international conference on New interfaces for musical expression. 2007.

53. Kidd, Cory David. Sociable robots: The role of presence and task in human-robot interaction. Diss. Massachusetts Institute of Technology, 2003.

54. Weinberg, Gil. Interconnected musical networks: bringing expression and thoughtfulness to collaborative group playing. Diss. Massachusetts Institute of Technology, 2003.

55. Qi, Jie. Paper electronics: circuits on paper for learning and self-expression. Diss. Massachusetts Institute of Technology, 2016.

56. Burland, Karen. "Music for all: Identifying, challenging and overcoming barriers." Music & Science 3 (2020): 2059204320946950.