

BUILDING AN INSTRUMENT ARTIFACT

**By Sam Crawford
Student number - 2010911**

CPO D 0720 MA. Commercial Music Producer

Tutor, Ciaran Robinson

Tileyard Education, University of Wales Trinity Saint David

London

10th February 2021

Designing, creating and developing an instrument artifact.

“Sound design is the process of recording, developing, or producing audio elements. It is used in a variety of areas including in films, in television production, in theatre, and even in video game software development. The process usually involves manipulating previously composed or recorded audio, music and sound effects. In some instances, sound design involves the composition of recording audio to create a certain tone or effect”

Banton. The process of sound design, written by [Banton Joelle April 2020, from Music Gateway], a website that provides commercial briefs for producers. [https://www.musicgateway.com/blog/how-to/what-is-sound-design\](https://www.musicgateway.com/blog/how-to/what-is-sound-design)

Introduction.

The purpose of this essay is to discuss and frame, my ideas, the process and methodology that I followed and, in some cases, didn't follow, along with my understanding informed by research, the creative process in, designing, building and developing the final instrument artifact and self-reflection on the final outcome.

I discuss the various options that were available, and I detail the decisions I made, and the research and sources used to inform those decisions, finished with analysis of the results of my final instrument artifact and reflect upon the outcome.

The idea of; designing, developing and creating any form of instrument artifact, was so far outside of my comfort zone, the very thought of learning to code, to programme and to build an instrument physically, or, inside a platform, that I had not used before, gave me a lot of anxiety. Whilst the lectures covered a range of topics across sound design and creative technology, it is such a broad field with many sub fields, that it captures so many possibilities. From creating a game with sound design, to writing your own software instruments, to music hacking with physical artifacts.

As someone who still has a lot to learn about the concepts, the theory of how sound is created, how sound waves are manipulated and shaped to create sounds that resemble instruments, this was something I was keen to research and develop an understanding in more detail. Sound design covers such a wide spectrum, and this was clear from the weekly lectures, where we started with game design and sound manipulation and worked our way through to software instrument design and music hacking.

Deciding on an initial path to follow was not something that I found naturally, there was so much choice available that it became counterproductive and analysis paralysis set in.

I had identified two possible avenues that were of interest to my own personal development:

- Software instruments
- Music hacking

The first step was to revisit the lecture materials to get more insight into the possible options and to enable my thought process about the type of instrument I wanted to create.

During week six & seven, we had been introduced by “Phelan Kane” to “Native Instruments Reaktor”, “Max / Max MSP” and “Pure Data”, all of which are platforms used to create software instruments. CPO_D_0720 Commercial Music Producer Development - Phelan Kane

- Week 6 - 6.3 Sound Design and Creative Music Technology (Software Instruments Part 1)
- Week 7 - 2.3 Sound Design and Creative Music Technology (Software Instruments Part 2)

The other option of personal interest was music hacking and best described by Helen Leigh during the lectures. Music hacking is the development of physical everyday objects combined with software code to further expand their use in creating music and sound.

Hackaday.com states that, *“Music throughout history has been inspired and changed by hackers and makers, and never more so than in the 20th century.* Helen Leigh is one such hacker. The tape recorder was an invention that kicked off a golden period of exploration in sound, leading to the Musique concrete. Helen is a prolific creative technologist, who experiments with sound and music building hardware, producing a series of interactive sound sculptures. CPO_D_0720 Commercial Music Producer Development - Helen Leigh

Week 8 - 8.3 Sound Design and Creative Music Technology (Music Hacking Part 1)

Week 9 - 9.1 Sound Design and Creative Music Technology (Music Hacking Part 2)

Week 10 - 10.1 Sound Design and Creative Music Technology (Music Hacking Part 3)

Whilst developing physical products and hacking them were of interest, I ultimately decided to create a software instrument. At this stage of the process, I had not defined the type of instrument I wanted to create or the platform I would use to create it.

Platforms to create software instruments.

“There are lots of reasons why you might want to write your own VST plug-ins. The learning curve can be steep, but there are development environments out there that can help” Nick Storr, Sound On Sound <https://www.soundonsound.com/techniques/vst-diy>

What I did do, was to identify my own personal learning outcomes that I wanted to achieve from this module and assignment.

- To collaborate with another peer on learning, developing and holding each other accountable.
- To develop a broader understanding of how sound waves create distinct sounds using oscillators
- To be able to read and create my own schematics and apply the learning to the project.
- To understand more about synthesis and sound manipulation to be able to use these skills in my work as a producer.

Given that I had no experience in building electronics, or being able to read, interpret and understand schematics, I had identified that being able to read a schematic was a skill that would be required and applied to building a software instrument. I knew I had a steep learning curve to climb. I partnered with Steve Phillips, a fellow student on the programme – Between us we researched the various options available

- Reaktor6 - according to Native Instruments; is “the ultimate creative toolkit for deep sound exploration, featuring cutting-edge DSP and high-fidelity sound”. Native Instruments.
- Max / Max MSP - is; “A Playground for Invention” “Max is an infinitely flexible space to create your own interactive software” Cycling 74 <https://cycling74.com/products/max>
- Pure Data: Pure Data - is a visual programming language developed by Miller Puckette in the 1990s for creating interactive computer music and multimedia works. While Puckette is the main author of the program, Pd is an open-source project with a large developer base working on new extensions.

Miller Puckette, is the man behind both programming languages, Max and Pure Data, (the latter is also abbreviated to Pd), he is a computer musician from the University of California, San Diageo,

“These languages, now several decades old, have nourished active communities of avant-garde musicians, who rely on the software’s unique visual interfaces for their performances and compositions”. This article appears in the March 2017 print issue as “Digital Music’s Language Maestro.”

Puckette wrote Max in 1988 while working at the Institute for Research and Coordination in Acoustics/Music, or IRCAM, in Paris. *“What I was trying to do was make software that, with modifications, would be mostly reusable from one piece of music to the next,”* he says. This article appears in the March 2017 print issue as “Digital Music’s Language Maestro.”

To begin, we started with Pd, (Pure Data), as this was an open-source platform. From the moment that we downloaded and tried to use this platform; it became a complex web of terms and features, the user interface was not easy to understand and trying to find information on how to use the tool, was not easy. Whilst there were forums and a large community, the user experience thus far had pushed me away from this as an option and my evaluation was to disqualify it from my process. I recognise that Pd is a very capable platform, but not for me personally.

The next platform I evaluated in conjunction with Steve Phillips was Max MSP, again another very powerful platform, there were a couple of concerns that I had with this platform. The first was that the experience of trying to use Pd had not been successful and the programming languages used are from the same source. The second, was there was a lot of debate online, on forums discussing how Max MSP was more aligned to Ableton and Ableton Live. The thought of taking on another DAW, on top of Pro Tools and Logic, I felt would hinder my development. There was also a monetary factor to consider. Over the Christmas period, I had purchased “Komplete 13” from Native Instruments. Included in that, was Reaktor6, full version.

Native Instruments provides two versions of this tool, the Reaktor6 Player, and the full license, the difference being that a full license is required to create your own instruments or to edit existing instruments Vs the player which is a free download. Given I owned a copy of Reaktor6, I decided that this would be the platform I would use. I was also conscious of time. Evaluating the art of the possible, was going to be a challenge on developing an instrument and I was spending too much time looking at platform options.

“You don't have to see the whole staircase, just take the first step.” — Martin Luther King Jr. This quote from Martin Luther King, summed up where I was in this process and I needed to take the first step. So Reaktor6 was the next decision made.

Reaktor6 & Designing

Reaktor6 is a powerful graphical modular software music studio that enables musicians and sound design specialists with the ability to develop, build and create their own virtual instruments. Their software is supported by both Windows and Mac OS operating systems.

“Reaktor is a daunting environment to work and build in. The sheer number of options you have is limited only by your knowledge of how synths and effects work”. [Tom Jarvis - Top tips for building synths and effects in Reaktor6 <https://audioordeal.co.uk/top-tips-for-building-synths-and-effects-in-reaktor/>]

Upon downloading and installing Reaktor6 you are met with a screen that offers you three options. Play, Patch, Build. (See appendix 2.1 for screenshot)

I was initially drawn to the play selection, this brought up a number of premade synthesisers, effects and utilities from Native Instruments. It was at this point that I started to look at some of the features that these tools provided. After this I then began further expanding my research into other virtual instruments. Serum, Arturia VCollection, and Spitfire Audio. The latter uses the Native Instruments “Kontakt Player” as the framework for the tool to work within. I started to write down specific features that these tools had, from oscillators, to samplers, sequencers, envelopes, filters, scopes, displays. I looked at their GUI (graphical user interfaces) their user experiences, sound banks and pre-sets that comes with them.

This shaped an initial concept, which was to look at creating an instrument that would help others like myself who don't play an instrument or understand music theory, coming from the route of a non-traditional musician. The development of this concept was going to be far beyond my own ability, especially as my own understanding of synthesisers was already limited.

Synthesisers, also known as “synth” or “software synth”, and their vast range of hardware and software products have influenced modern music for the last 50+ years. Synthesise according to the oxford dictionary means:

1) *make (something) by synthesis, especially chemically. "the drug was first synthesized in 1929"*

2). *produce (sound) electronically. "trigger chips that synthesize speech"*

"Nothing in life is to be feared, it is only to be understood. Now is the time to understand more, so that we may fear less." — [Marie Curie]

This is where I moved away from following a step-by-step process and instead moved to a multi-threaded approach.

I needed to understand how Reaktor6 worked and what the art of the possible was, whilst at the same time researching into synthesis, its history and where we are today.

Throughout the rest of this essay, I will be using terminology that may be abbreviated, I will outline what these terms mean, where relevant to my project.

A Synth VST is a software synthesizer designed to run as a "plugin" inside a piece of studio recording software also known as a DAW (Digital Audio Workstation). ... Synth VSTs are also known as VST Instruments or Virtual Instruments. Many of them are emulations of hardware synthesizers but are not limited to being so.

Thankfully there are many sources of information available on how to use the Reaktor6 platform. Starting with ADSRSounds.com a website that provides hours of tutorials by Salamaderanagram, he is a designer and builder of many instruments that sit within Native Instruments Reaktor6 and is a strong voice from within their community. I started with some basic step by step tutorials, understanding the hierarchy of how the Reaktor6 build structure works. Ensembles being a full Reaktor6 project Vs an instrument where you can have multiple macros to make up an ensemble.

ReaktorTutorials.com and Youtube.com were also valuable sites that I used to understand more about how to use Reaktor6, as well as spending hours reading through the manual that comes with Reaktor6.

There was a product that was simple to use from within Reaktor6 which is called "Blocks" and "Blocks Prime", this takes a Eurorack, type of approach, using premade modular instruments and tools that have already been coded and have macros defined, with the theory that sits behind them. Whilst this was straight forward and would have been a much easier route to take. I outlined earlier, that my own development goals were to understand more about how sound is created and manipulated, so I went straight past this and focused on creating from a blank screen. (See Appendix 2.2)

The first of many exercises was to replicate an instrument from the video tutorials. In order to build the instrument, we first spent time creating and coding examples first. Reaktor6 provides a simple graphical user interface that makes it easy to see what you are doing. It splits its screens into three core sections. The panel view, this is where the face of your instrument sits. Underneath this is the instrument, ensemble or macro view. And to the left-hand side is the parameters view, where you can apply changes, from values to text as well as customisation of colours. Reaktor6 simulates an easy-to-use coding and wiring system; Examples of this are the audio In and Out, as well as G = Gate, P = Midi Pitch.

Whilst following along with the process I discovered early on, how important it was to either turn Reaktor6 off prior to connecting wires, and to turn down the volume. This was critical, especially as we would be generating audio signals that can be unpleasant to the ear and could also blow your monitors. The first build created, was an oscillator generator module, that outputs periodic waveforms at specific frequencies. It is then controlled and determined by the keys pressed on the midi controller, the audio signal is then sent to the output, via your interface IO to your monitors.

As someone who was new to this, I wanted to be able to visualise the different types of waveforms, in order to achieve this, I then added a scope, taking signal from the oscillator into the scope. A scope reacts to the midi notes being played and their velocity. The scope has various settings that can be tweaked, including the ability to change how you see the sound wave, changing the pitch to ms, this graphical representation, makes it easier to see that it is a sawtooth wave being used.

This was a very simple instrument at this stage however I wanted the user to be able to select different periodic wave forms.

“A waveform is a graph that shows a wave's change in displacement over time. A waveform's amplitude controls the wave's maximum displacement”. [Comeau – The Pudding, Lets Learn About Waveforms. <https://pudding.cool/2018/02/waveforms/>]

“A waveform is just a schematic that helps us understand sound waves. They measure an air molecule's displacement over time. We can adjust the amplitude of a waveform to make it louder/quieter, and we can adjust the frequency of a waveform to make the pitch higher/lower”. [Comeau – The Pudding, Lets Learn About Waveforms. <https://pudding.cool/2018/02/waveforms/>]

There are different types of soundwaves, the sine waveform comes from trigonometry and according to my research it is known as the fundamental waveform. Sine waves play the frequency selected, for example 440Hz. There is no harmonics, eg no side effects.

Triangle waveform, this waveform is not as curvy as the sine waveform, it starts to add harmonics, a harmonic follows the same set of rules. The fundamental note, root frequency. Second harmonic 2x frequency. Third harmonic 3x frequency. Forth harmonic 4x frequency. Fifth harmonic 5x frequency, this pattern continues to infinity.

Triangle waves use the odd harmonics, 3rd 5th 7th as an example. I should note that the further you get away from the root the harmonics taper off.

Square waveform is binary it is either +1 or -1, it does follow the same harmonics as the triangle waveform but doesn't suffer from the tapping off impact like the triangle. Square waveforms tend to sound brighter.

Sawtooth waveform. A sawtooth waveform is similar to the sound of a string instrument, when you run the bow across it. Sawtooth waveforms have harmonics at multiples of the root frequency, this is unlike the triangle and the square waveforms.

Building in Reaktor6 especially when wiring components together, can become very complex and confusing. To avoid having an over complex structure, I used a macro. In Reaktor6, an instrument can contain items, ensembles and macros also do this.

It was this lesson that informed my own decision to build my instrument using an instrument structure and within that instrument having multiple macros / elements. The reason why I decided to build this way is because:

- They keep things organised.
- They are easy to duplicate
- This duplication enabled me to build at speed, rather than to build each bank or component again and again.

See appendix 2.4 for screenshots of my build.

I continued to watch a number of tutorials to familiarise myself even further.

Simon Hutchinson, has a Youtube channel that supports Reaktor6, - Oscillators and Wavetable Synthesis <https://www.youtube.com/watch?v=WVBPSIk7kMQ&list=PL7w4cOV-VxL6H9iE7Nuvo0vtQHdfXPzTVE&index=2>

It is recommended that you place the oscillators in order of harmonics. Stacking them this way also enables me to see the signal processing as it travels through the Reaktor6 file.

In the appendix, you will find in section 2.3, specific outlines and learning from other builds as well as screenshots from those initial instruments created.

My research led me to recognise that the majority of synthesisers like “Serum” use an oscillator to act as the sound source with the rest of the synthesiser shaping the harmonics of the original sound source, either through filters, modulation and other effects.

As I continued to build multiple instruments, I collaborated with Steve Phillips. I found that collaborating provided a greater learning experience. It also held us both accountable. With that said, the fact that we had both followed the same process and methodology and had similar instruments, meant that I decided to take all of the knowledge I had gathered and start again. This time without following any tutorials or instructions.

My instrument

My instrument artifact takes its inspiration from Serum, but I was not able to create an instrument that has the same capability.

What I wanted to create was an instrument that used multiple oscillators that included subtractive synthesis and additive synthesis, alongside a ferment bank. The purpose of which would allow someone to play with different instruments within one core instrument, enabling them to create any sound.

It will have multiple oscillators, and you'll be able to switch from frequency modulation and amplitude modulation. You can change the wave form of each one, the choices being a sine wave, a square wave, a triangle wave, a saw tooth wave, and pink and white noise. Each will have a coarse tuner that ranges from -24 semitones to 24 semitones, a fine tuner that ranges from -100 cents to 100 cents, and a volume knob. It will also be controlled by a midi keyboard.

The Dulcet, as I have named it, provides many functions. From the middle of the instrument there is a string modulator, that quickly allows someone to build and tune string sounds, either plucks or the sound of a bow striking a string. This is built using a modal bank that takes 64 partials, it is controlled via midi and each note pressed triggers the pitch, the amplitude is generated via a gate, which is connected to a number of orders and values, these are running mathematics behind the scenes. Given that 64 partials would destroy my CPU and

the users, I had to think about how I would achieve this. Using iterations and separators, with multiplication of the signal whilst also including noise, and filters, like the ADSR and a two-pole filter, this enabled me to add knobs and faders so that the user could control the Pcut off, the decay and damp. This leads to creating a more realistic string sound.

All of this I then wrapped into a macro. The macro keeps the various equations and components hidden.

Next to the string modulator is a formant bank, a formant bank enables the user to replicate vowel sounds, like Ah and Ee. Within the instrument there is a preset that demonstrates this. A formant bank, takes the tone of a sawtooth, adds noise and with the help of an LFO and a number of filters, you are able to set specific frequencies to create these sounds.

To the right there is an additive synthesiser. An additive synthesizer creates sounds by combining large numbers of sine waves with different frequencies, amplitudes and phases. In the context of an additive synthesizer these sine waves are called partials or harmonics

Reaktor6 offers a module created specifically for this purpose: Sine Bank.

Rather than creating a large number of individual Sine Oscillator Modules, this one Module allows you to generate many sine waves with only a single Module. It is also optimized for this function and is thus a more CPU-friendly option. At this stage I added multiple Sine wave banks. These are controlled by knobs and switches, routed via a mixer and are using trigonometry as outlined earlier in this essay. To the left, there is a subtractive synthesiser, Subtractive synthesis is an oscillator routed through a filter. A harmonically rich waveform, like a saw wave, has its harmonics subtracted by the filter, hence the term subtractive synthesis. However, even early synthesizer designers knew that a single oscillator and a single filter were not enough on their own and that modulation was required to change the sound over time. Amplifiers, LFOs (Low Frequency Oscillators) and Envelopes were added to give sound designers and musicians more flexibility. Within the mixer macro, I added 5 inputs to align with the four core instrument types and one master output. I then added a signal path and an amp mixer. Wired up the amp mixer to each input and then went back to the main structure and added each oscillator to the macro. At this stage my instrument was able to create unique waveforms, by using the levels from each oscillator and a mix of different waves and pitches, this could be seen and identified via the scope. It was creating a scope where I struggled, I did manage to create one, however it would not refresh at the correct rate, so for this part of the build, I used a premade block from within Reaktor6.

Connecting all of the oscillators was a long process and this required attention to detail, labelling the various fields and math that was added, multiplied or subtracted.

Playing close attention to the fact that different waveforms require additional functionality.

A pulse has a W switch for the width duty cycle, and rather than using the gate, I added a custom control. The custom control enabled me to create a knob on the panel, which appears when the user selects that type from the drop down.

Naming all of the oscillators so that they are clear on the panel and at the macro level adding an aux tool, audio smoother to the gate and the oscillator amplitude, would add a slope and remove pops on the sine wave.

Creating switches to enable the best user experience was also very important. The more control that someone has the better use they will get from this instrument, Once I had created my instrument, I then spent time creating a number of presets, these can be accessed via the tool from within Reactor6 Player. This provides the user with a place to start.

Spending time with each of the oscillators, adjusting the wave signal, the level and the p width for the pulse waves, provided me with a lot of practical insight into how you can tune sounds. Thinking about the user and being aware of monitors, it was important to ensure each pre-set was balanced to avoid clipping on the master volume.

I tried to cram a lot of functionality and additional controls to the oscillator so you could make additional sounds, this included adding a noise generator inside the macro. A noise generator is a random non periodic wave form across the frequencies.

Whilst building the panel I also had to make design choices along the way this included if I should use buttons or menus or lists, how they would sit, spacing and other design GUI decisions.

Reflection

Building a multi-instrument means that it takes up a lot of space on the screen, as I started to think about brand and design, I had to make some decisions, these included, reducing the size of knobs and faders to being small. It is a compacted instrument, and I hid some items, as I was either not able to get them to fully work, or there was not enough space.

I knew that this module would be a challenge from the outset, with that said, I am pleased with the result. I recognise that the instrument is over complicated. There was a set of specific outcomes that I wanted to achieve and approaching this module, this way, enabled me to hit those objectives

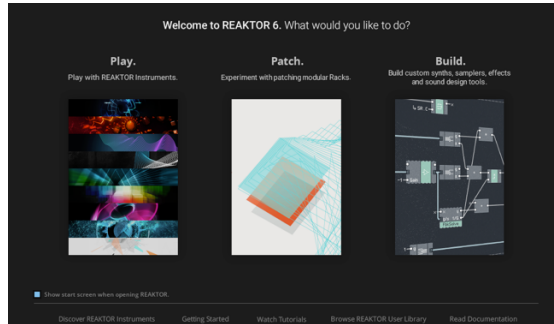
As I continue to modify the instrument, I would look to streamline the ideas into one more defined idea from the start of the process. I went into this project unclear on what I was actually building, and I believe that this can be seen from the end result.

I found Reaktor6 to be easier to use and navigate than I imagined, this was due to its simple key commands. I shared the instrument with Steve Philips and had positive feedback, the design process was difficult. I certainly recognise the skills required to create and develop an

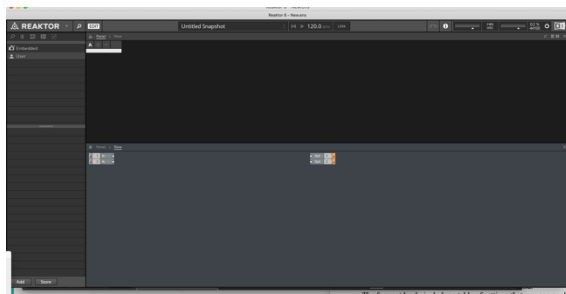
instrument. The market is growing, and more and more people are using software to emulate hardware. I will continue to invest time and this exercise has provided me a solid foundation in areas of sound design that I can transfer into my own music production.

Appendix

2.1 Launch Screen of Reaktor



2.2 Blank Screen within Reaktor6



Appendix 2.3

Note Pitch - to change the pitch via your midi keyboard

Noise Gate - Sends a 0 when you don't press a key and then sends a value when you do, eg note on note off - coding value is 0-1

Sampler

Added Sampler and then a Midi in - Pitch and then Midi in Gate for velocity

Range of midi values is 0-127

Audio Voice combiners via aux

Mention that you embed the samples within the player - so that they contain in the product

L =lowest note on the keyboard to try

H = highest note on the keyboard to trigger sample

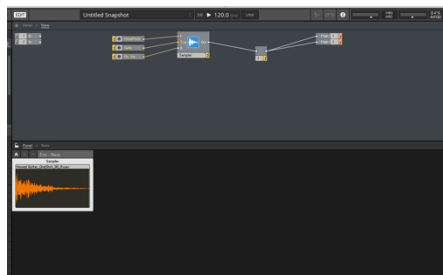
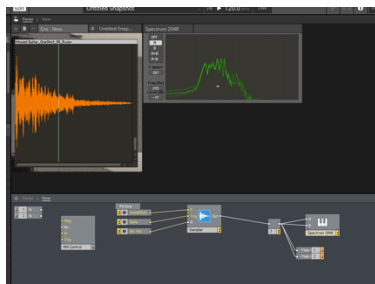
Root should be a value of 60 which is middle c on a keyboard

To change the note lengths, you can either have a gate which will play as long as you hold the key vs velocity which will allow the sample to play from start to finish regardless of if you hold the key down or not

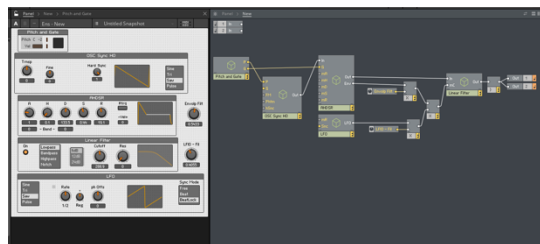
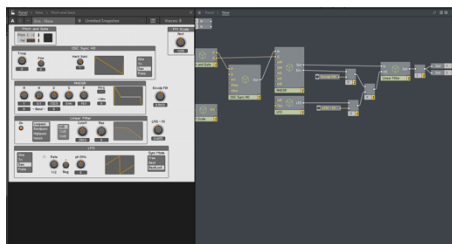
Polyphony - the number of voices available for playback, it would require an audio voice combiner aux as part of the code / programming

Samplers need to be mono and wav - cannot be MP3 \

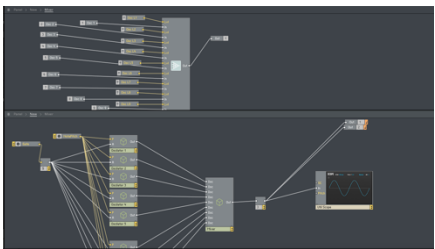
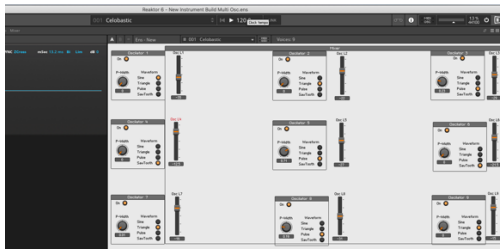
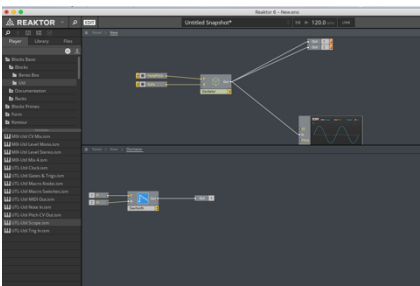
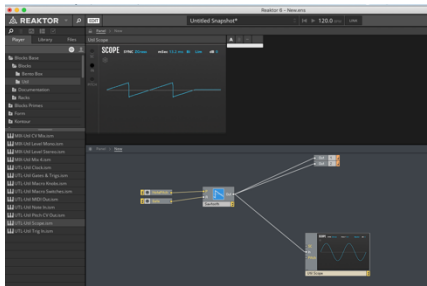
Screenshots of sampler build.



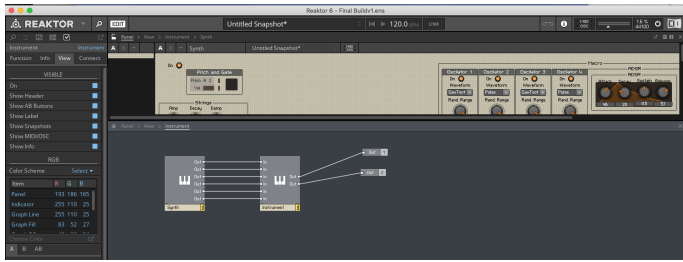
Screenshots of oscillator instruments

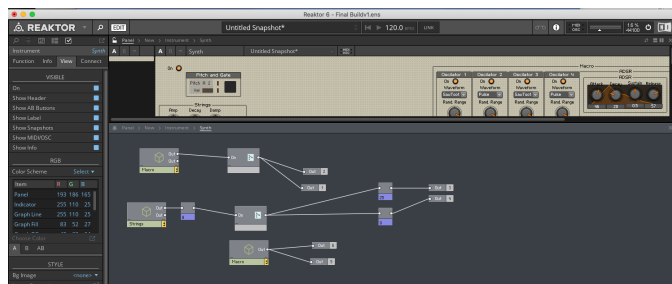
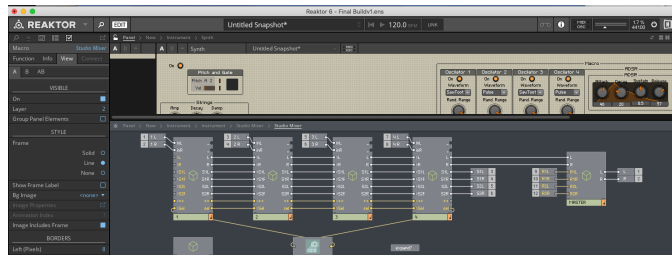
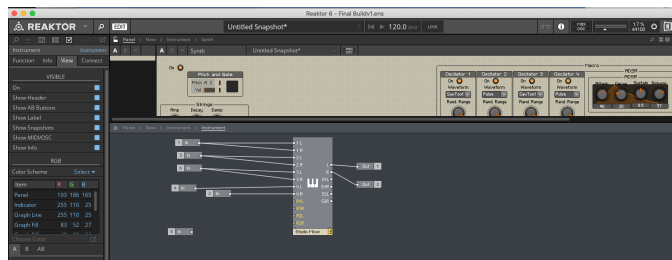


Screenshots of adding a scope and macro.



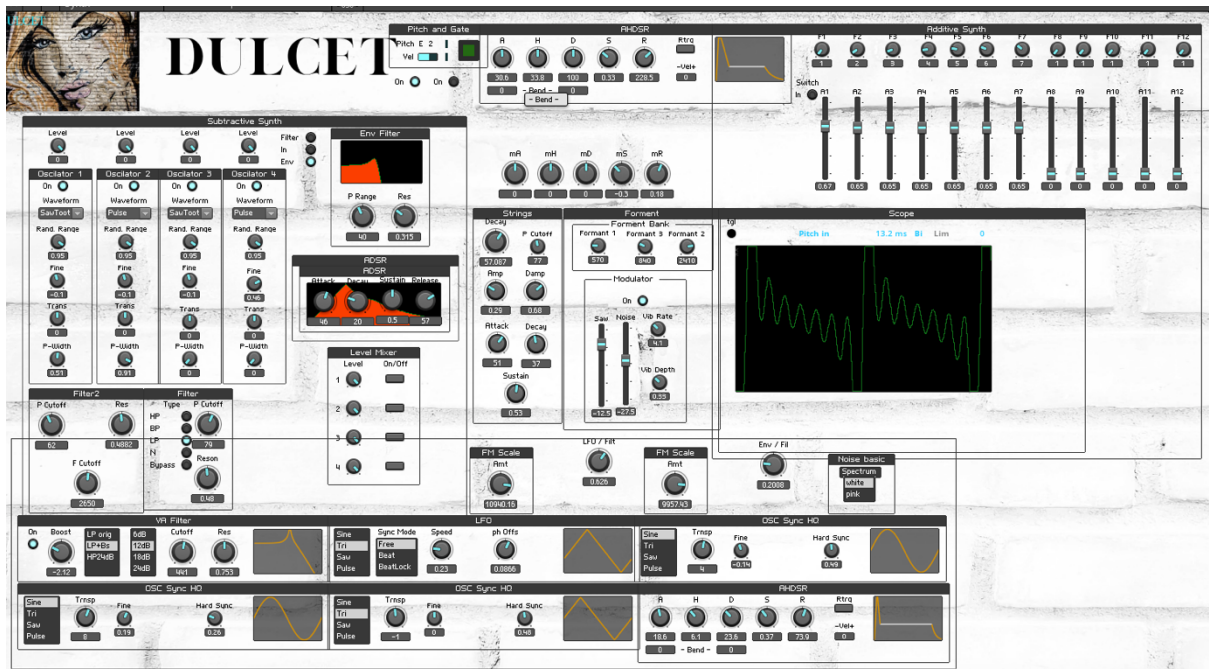
2.4







Final Instrument



Bibliography.

Leigh, H, Sound Design and Creative Music Technology – Available at Tileyard Education
Beepop

CPO_D_0720 Commercial Music Producer Development, Sound Design and Creative Media Technology - Helen Leigh

- Week 8 - 8.3 Sound Design and Creative Music Technology (Music Hacking Part 1)
- Week 9 - 9.1 Sound Design and Creative Music Technology (Music Hacking Part 2)
- Week 10 - 10.1 Sound Design and Creative Music Technology (Music Hacking Part 3)

Kane, P,- Sound Design and Creative Music Technology - Software Instruments, Part 1 & 2
– Available at Tileyard Education

CPO_D_0720 Commercial Music Producer Development Sound Design and Creative Media Technology - Phelan Kane

- Week 6 - 6.3 Sound Design and Creative Music Technology (Software Instruments Part 1)
- Week 7 - 2.3 Sound Design and Creative Music Technology (Software Instruments Part 2)

CPO_D_0720 Commercial Music Producer Development - Sound Design and Creative Media Technology

- Sound Design and Creative Media Technology - Part 1 (Book Chapters 1-6)
- Sound Design and Creative Media Technology - Part 2 (Book Chapters 7-11)
- Sound Design & Creative Media Technology - Part 3 (Book Chapters 12-16b)
- Sound Design and Creative Music Technology - Part 4 (Book Chapters 17-21)
- Sound Design and Creative Music Technology - Part 5 (Book Chapters 22-29)

Tom Jarvis - Top tips for building synths and effects in Reaktor <https://audioordeal.co.uk/top-tips-for-building-synths-and-effects-in-reaktor/> accessed 15th Jan 2021

<https://www.native-instruments.com/en/products/komplete/synths/reaktor-6/> accessed Dec 28th 2020

Additive synthesis <https://www.youtube.com/watch?app=desktop&v=Rudyc39YKUM> accessed Jan 5th 2021

Role of Sound Designer - <https://www.prospects.ac.uk/job-profiles/sound-designer> accessed 1st Feb 2021

Joelle Banton <https://www.musicgateway.com/blog/how-to/what-is-sound-design>
<https://spectrum.ieee.org/geek-life/profiles/miller-puckette-the-man-behind-the-max-and-pd-languages-and-a-lot-of-crazy-music> accessed 19th Jan 2021

Reaktor Primary 3: Subtractive Synthesis | Simon Hutchinson
<https://www.youtube.com/watch?v=WVBPSIk7kMQ&list=PL7w4cOV-VxL6H9iE7Nuvo0vtQHDFXPzTVE&index=2> – accessed 15th Jan 2021

J Acoust Society - http://www2.eng.cam.ac.uk/~jw12/JW%20PDFs/JASA_MSW.pdf accessed Dec 28th 2020

Reaktor Primary 3 EXTRA: Formant Synthesis | Simon Hutchinson
<https://www.youtube.com/watch?v=xV20jWNF-9w&list=PL7w4cOV-VxL6H9iE7Nuvo0vtQHDFXPzTVE&index=4> accessed Jan 20th 2021

Reaktor - MIDI & OSC Learn | Simon Hutchinson
<https://www.youtube.com/watch?v=Lg5GA6uenkM&list=PL7w4cOV-VxL6H9iE7Nuvo0vtQHDFXPzTVE&index=5> accessed Jan 20th 2021

Filtering to create words or sounds
<https://www.youtube.com/watch?v=xV20jWNF-9w&list=PL7w4cOV-VxL6H9iE7Nuvo0vtQHDFXPzTVE&index=4>
8:38 accessed Jan 20th 2021

Reaktor Primary 4: ADSR Envelope | Simon Hutchinson
<https://www.youtube.com/watch?v=QgtcXgGStNE&list=PL7w4cOV-VxL6H9iE7Nuvo0vtQHDFXPzTVE&index=6> accessed Jan 21st 2021