Tampere University of Applied Sciences

Audio production and implementation for indie games

Using Wwise with indie games

Mikael Vanninen

BACHELOR’S THESIS
May 2022

Bachelor’s degree of media and arts
Music production study path

ABSTRACT

Tampereen ammattikorkeakoulu
Tampere University of Applied Sciences
Bachelor’s Degree of Media and Arts
Music Production

VANNINEN, MIKAEL:
Audio production and implementation for indie video games

Bachelor's thesis 45 pages
May 2022

The purpose of this thesis was to gather information about the different techniques related to audio production and implementation in a small-scale indie game project using the audio middleware Wwise. The analysis was based on a practical project as well as literary sources.

This study was carried out as a project called Fast Food Simulator. In the thesis, several audio production stages were analyzed such as pre-production, implementation, mixing and mastering. The workflow between the audio designer and the game programmer was examined. The main research question was is using audio middleware beneficial in small scale video game projects. The analysis was made from the standpoint of development efficiency and the workflow between the programmer and the audio designer.

The results indicate that working with audio middleware can be beneficial even in a video game with a small budget as audio middleware moves the responsibilities in the audio side of video game development to the audio designer. With the audio designer working with audio middleware, the programmer can focus on developing the game further instead of having to worry about audio mixing or other audio related aspects.

Key words: audio implementation, reaper, wwise, games, indie

CONTENTS

GLOSSARY

DAW	Digital audio workstation
Hz	Audio frequency
Audio channel	A source of the audio signal
Game build	A compiled version of a game
MIDI	Musical Instrument Digital Interface
Middleware	Software used to implement sounds into a game build
SFX	A sound effect

1 INTRODUCTION

The availability of various audio production and implementation tools for video games has made video game sound design easier to access for a wide variety of people. With the low licensing fees and plenty of learning materials, using tools such as audio middleware and DAWs has become viable even in small scale video game projects.

In small game projects there could be just one programmer in charge of all the code that is made for the game. Audio implementation tools such as audio middleware can help relieve some of the workload from the programmer and help the sound designer get the game to sound as intended. This can help the workload to be more balanced and can help the programmer and sound designer to focus better on their respective fields.

The goal for this thesis is to examine the tools and production techniques that a video game sound designer can use to achieve the best possible results. The main audio production tools this thesis focuses on is Audiokinetic Wwise and Cockos Reaper. The production side includes planning, pre-production as well as the audio production and implementation itself.

The practical part of this thesis includes an early playable build of the game with a working title Fast Food Simulator. Fast Food Simulator is a game about a summer job in a fast food restaurant. It has been developed using the Unity game engine. The game experience is a balance between chaotic work hours and relaxing after hours. The objective for the sound design of the game was to reflect the game experience and enhance the chaotic feeling the player gets when trying to complete several food orders at once.

2 GAME AUDIO

In video games, audio is an essential tool to create immersion for the player. Audio can affect the players experience of playing the game in various ways. In the videogames we play today, audio can control the mood and react to what the player is doing in real time. The main goals for audio in video games are total immersion for the player and the player hearing everything that is happening around him. In most cases, game audio is also non-linear. In a movie, for example, the soundtrack and other audio progresses in a linear manner and the viewer doesn’t affect the movie at all. In video games, the player should always be the centerpiece whose actions affect everything, including sound. (Aaron Marks 2017, 4-5.)

2.1 A brief history of game audio

One of the very first video games ever developed was “Tennis for two” by William Higinbotham, a nuclear physicist who was involved in the development of the first nuclear bombs (Brookhaven National Laboratory, 2008). However, the game did not have any sound whatsoever, nor did any other video game that was developed after “Tennis for two” for over a decade. Finally, in 1972, the first video game with a sound component “Pong” was released. (Glenn Mcdonald 2005.)

PICTURE 1. Tennis for Two on an oscilloscope (Brookhaven National Laboratory 2013).

Allan Alcorn, the creator of “Pong”, created the game as a training exercise that was assigned to him by Nolan Bushwell, the co-founder of Atari. Bushwell had a vision of having a sound of a huge crowd roaring, cheering and applauding during the game. This grand vision was hard to achieve with the programming tools of the time so Alcorn ended up with the tones that were already available to him in the sync generator. (Al Alcorn interview 2008.)

The first home video game console, the Magnavox Odyssey, was released in the same year as the arcade game “Pong”, in 1972. The console did not have any audio capabilities however, and the home video game console market did not have a machine that could produce sounds until the release of Atari VCS in 1977. (McDonald, 2005.) It had a dedicated 2-channel monaural 4-bit sound chip with volume control which could produce simple sound effects and even music (Paul Scolum 2003). In the years following the release of the Atari VCS the sound design started to evolve with games like Space Invaders and Asteroids. These games used sound to make the already high-pressure gameplay feel even more hectic and intense. The first game to feature a talking voice was Major League Baseball for the Mattel Intellivision system, both released in 1979. The voice was computer generated and could imitate simple baseball terms used in the game such as “Strike”, “You’re out” and “Ball”. (McDonald 2005.)

Moving into the 1980s, the next major development in sound design was the arcade game Berzerk, developed by Stern Electronics and released in 1980. Berzerk featured sophisticated synthesised speech which consisted of full sentences. (McDonald, 2005.) However, the use of speech synthesis was quite expensive, costing around $1000 per word to compress using LPC coding (Arcade museum). In 1980, another remarkable game in video game history was released, Pac-Man. It shipped over 100,000 units in the United States alone and became one of the most popular video games in the world. Pac-Mans sound design and music are recognizable around the world (McDonald 2005). Pac-Man also helped to kickstart the video-game industry, sparking the spread of arcades all around the world (Matthew Partridge 2015).

Another notable release was in 1981, when the arcade game Donkey Kong was released. However, the next milestone in video game audio happened the same year, when the Atari 5200 was released. It featured a separated audio component, the POKEY chip. The chip has four separate channels for audio, and the pitch, volume and distortion of each track could be controlled separately. The Atari 5200 featured two of these chips, creating a total of eight tracks to create video game music with. This was a remarkable development as a game called Tempest: Sound and Fury was released for the Atari 5200. It was one of the first games to use the POKEY chip. Tempest: Sound and Fury is recognized as the first game to have a stand-alone audio soundtrack release. In 1983, the first game featuring stereo sound and real human voices was released, called “Into the Dragon’s Lair”. This feat was achieved by using LaserDisc technology. (McDonald 2005.) As well as technological breakthroughs, the 1980s featured many iconic video game soundtracks, from video games such as Tetris by Alex Pajitnov, and Super Mario Bros, the soundtrack for the latter was composed by Koji Kondo. (McDonald 2005) Super Mario Bros was released in 1985 for the Nintendo Entertainment System (NES), a home video game system that helped the video game industry expand rapidly in the United States, and later the rest of the world. (Kohler 2010.)

In 1986 the first optical disk drive for home video game systems was released, the Famicom Disk System for Famicon, the Japanese version of the Nintendo Entertainment System. It never reached the market in the United States but it marked the start of the shift from video game cartridges to digital discs. The NES featured games with remarkable soundtracks, with starts of two video game franchises, the Legend of Zelda and Final Fantasy. In 1989, several console releases advanced the video game audio techniques; the TurboGrafx-16 by NEC which had a separately bought CD player attachment, and the Sega Genesis, which featured six stereo audio channels. Nintendo’s first handheld console, the Game Boy was also released the same year, and it featured two speakers and four mono sound channels. (McDonald 2005.)

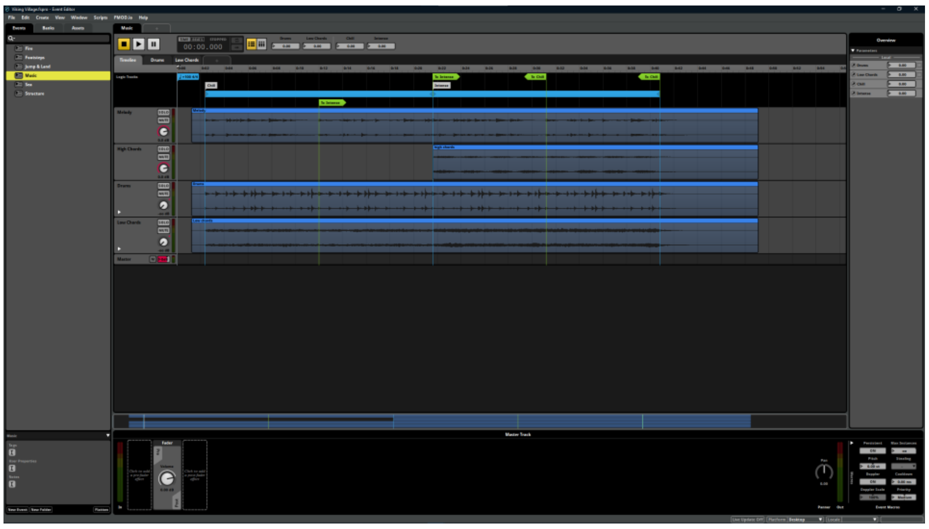
PICTURE 2. The Famicon Disk System (The Vanamo Online Game Museum 2017)

The early 1990s featured many releases with large advances in video game audio technology, such as the Super FamiCom in 1990, its release in the US with the name Super Nintendo Entertainment system (SNES) in 1991 as well as the NeoGeo in 1990. The NeoGeo featured an 8-bit sound processor with 15 separate channels (McDonald, 2005). The SNES featured the Sony SPC700 audio processor which had 8 stereo channels and could playback “near CD-quality” PCM files. This meant that the SNES games could feature actual human voices and real samples, although the memory limitations of the cartridge-based system would mean that the sound engineers working on the game would have to come up with creative ways to save space on the game cartridge. One key technique to deal with the limited sound memory on the SNES was to have the audio samples on a high pitch which then would be slowed down in software. (Joey Pendergrass 2015, 23-24.) Utilising the SNES’ technology in 1991 one of the first video game titles to use a symphonic musical score, ActRaiser was released. The score featured a more cinematic style that took influence from Hollywood film scores. Another video game that was released in 1991, “Joe Montana Sportstalk Football“ for the Sega Genesis featured another advancement in video game audio techniques. It was the first sports game to feature continuous play-by-play commentary, where the in-game announcer would commentate on the action in real time. Although it would be a few more years before the CD would reach its peak in the video game market, in 1992 the Sega CD was released, furthering the migration from cartridge media to optical media. (McDonald 2005.)

The breakthrough of using optical discs to store video game data came with the release of the Sony Playstation in 1994. It was the first commercially successful video game console to have full 16-bit 44.1kHz CD-quality audio. This allowed the video game developers to have music in their games technically on par with commercial music releases. The Playstation also had more processing power than its predecessors thanks to its 32-bit architecture. This allowed the Playstation to produce 24 channels of PCM files and MIDI-playback. The console also pioneered many of modern audio processing features such as real-time pitch modulation and digital reverb. (Pendergrass 2015, 24-25.) The powerful features of the Playstation allowed game developers to implement licenced music from famous musical artists, bringing video games even further in the public knowledge with releases like “Tony Hawk’s Pro Skater”(1999) and “Wipeout” (1995) (McDonald 2005; Pendergrass 2015, 25). Other innovations for the Playstation included “Dance Dance Revolution” by Konami in which the player had to match their movements on an external dance pad to the action happening on screen. “Vib-Ribbon” was another innovative game where players could insert their own audio CDs into the Playstation and play music, of which the game would create unique stages. (McDonald 2005.)

The release of the sixth generation of video game consoles in the beginning of the 2000s brought video game audio close to its current form. The sixth generation of consoles included the Sony Playstation 2, Nintendo Gamecube and Microsoft Xbox all of which boasted powerful technical capabilities. This increase in processing power meant that the audio programmers were not constricted by the technical capabilities of the consoles like in the console generations before, and could create an even more immersive audio systems such as dynamic mixing, where the game controls the audio in real time according to the players actions. In earlier generations of consoles, audio implementation was more basic as the audio mix was programmed by the audio programmer. In dynamic mixing, the actions of the player affect the mix of the game, not just the sound effects themselves but music, dialog, ambience all could be programmed to react to the players presence in real-time. (Pendergrass 2015, 26-27.) Nintendo Gamecube, released in September 2001 was the first console to have a separate digital signal processor (DSP) and the possibilities for surround sound. Microsoft Xbox quickly followed in November 2001 with support for surround sound as well as the capability for 256 separate stereo voices through 64 channels compared to the Gamecubes 64 voices and the Playstation 2's 48 voices. The Playstation 2 did not support surround sound. (Kyusik Chang, Taeyong Kim 2007, 5.)

Along the release of the sixth console generation, audio middleware started becoming commercially viable with the first release of a programmers API for FMOD (Guy Somberg 2016, 98) and development of Wwise starting in 2000. The first commercial version of WWise was released on 2006 and the first game that utilised it was Microsoft Games Studios' Shadowrun (AudioKinetic n.d.). Audio middleware is a program that works in conjunction with the game engine between the programmer and the audio designer. Audio middleware helps the audio team working on the game by providing a graphical user interface similar to DAW applications more familiar to audio personnel. Audio middleware also allows the sound designer to utilise DSPs through the middleware without having the programmer hard code it into the game source code. (Horowitz, Looney 2014, 124-125.)


PICTURE 3. A screen capture of FMOD (FMOD 2022)

The seventh generation of consoles, starting with the Microsoft Xbox 360 in 2005, brought with it the complete switch to surround sound. All Xbox 360 games were required to support 5.1-channel Dolby Digital surround sound. Nintendo Gamecubes successor Nintendo Wii had similar audio capabilities to the Gamecube and Sony Playstation 3 featured full surround sound support. (Chang 2007, 6.) During the seventh generation and the implementation of surround sound into video games, dynamic and interactive audio became the industry standard in video game audio development (Marks 2008, 5).

Moving on to the 2010s, Audio middleware started becoming more popular, with Audiokinetics Wwise already having been shipped with 100 games by the year 2010 (Audiokinetic n.d.). Its contender FMOD released FMOD Studio in 2013, a brand new take on the original FMOD with significant technical improvements on its predecessor. (Horowitz 2014,133.)

2.2 Sound design techniques for games

Sound design for games can be considered similar to musical composition. To make a musical piece, you can choose different instruments and tempo for your composition. The same applies to making sound effects, where you can choose what sounds fit together as well as the rhythm of the sounds. The main difference being that the sound effect is usually only a few seconds long. The production process for both musical pieces and sound effects is similar as well. Recording, editing, mixing and mastering are essential in both cases. (Marks 2008, 34-35.)

2.3 Audio implementation methods in games

There are basically two methods of implementing audio into a game. For the developers, the options are to create their own in-house audio engine or to use audio middleware such as FMOD or Wwise. Using audio middleware has become the more popular option in the video game industry as it is much simpler to use systems that are already developed and standardized than it is to program them from scratch. (Guy Somberg 2018, 24) Using audio middleware also speeds up the development process, especially in indie games where there might be just one main programmer, as the programmer doesn’t have to focus on tweaking and mixing the sounds and instead can just create the hooks, the spots that the sounds are played, in the game. With middleware, the audio designer can focus on the sound design and tweaking the audio. The programmer and audio designer have to agree on specific parameters and hook and the audio designer can do all his work in the middleware. (Horowitz 2014, 125.)

2.4 Middleware

Audio middleware is a piece of software that works between the game engine, programmer and the audio designer (Horowitz 2014, 124). Most audio middleware can do the same basic tasks. The most basic task is playing sounds. With middleware, the audio designer can decide when, where and how loud the sounds are playing. A key element to creating a more organic soundscape is randomizing different sound parameters such as volume and pitch and these are the most important features middleware can offer. Other important tasks include; creating dynamic music and ambience which reacts to what the player is doing in the game; real time audio mixing and the ability to use different audio file compression methods for different gaming platforms. (Horowitz 2014, 127-128.)

2.4.1 Wwise

The ”Wave Works Interactive Sound Engine” or Wwise for short is an audio middleware developed by Audiokinetic. It offers a versatile event system and an interactive music engine. It also offers audio designers the ability to create abstract prototypes of the sound systems for a game even without a working build of the game by using the Soundcaster features. (Horowitz 2014,134-135.)

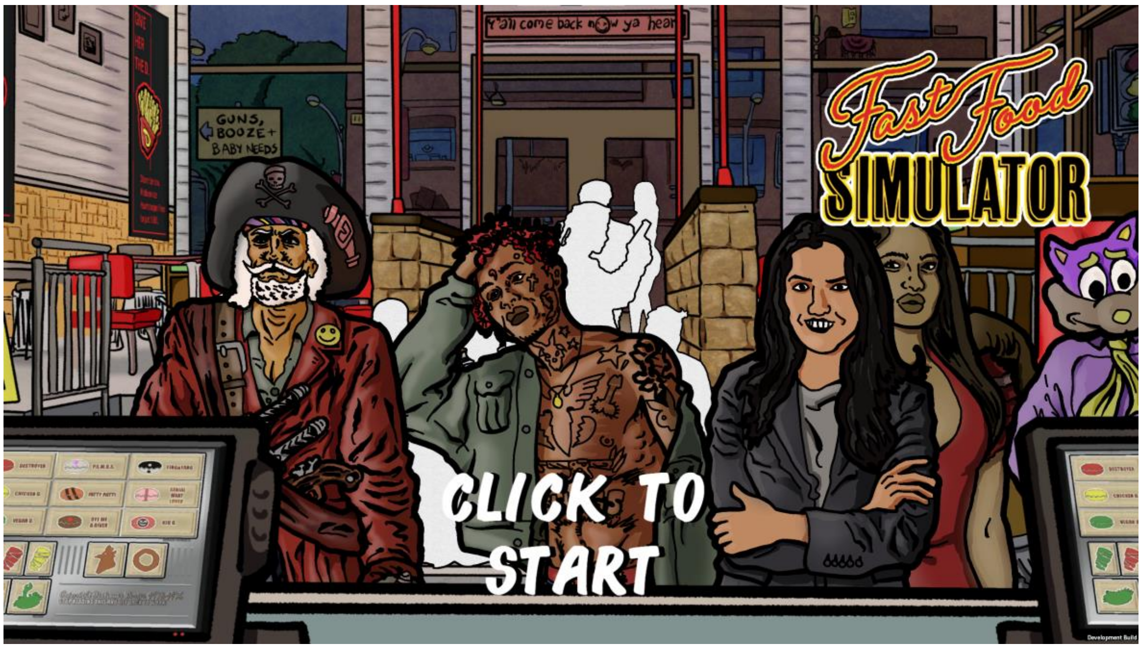
2.4.2 FMOD Studio

FMOD Studio by Firelight Technologies was released in 2013. It offered significant improvements to their previous engine that was initially released in 2002 but updated over the years. The improvements included sample-accurate audio triggering, better file management and a new audio mixing system that allowed the use of busses, sends and returns. (Horowitz 2014, 133.)

2.4.3 Other options?

The two middleware mentioned above are the two most commonly used systems but other options are available. The Miles Sound System by John Miles, first released in 1991, is one of the most popular options with over 5000 games shipped over the years. Fabric is a high-level audio middleware, meaning that it offers functions similar to middleware with a lesser need to learn code. Fabric is designed to be used with the game engine Unity. (Horowitz 2014, 135-136.)

3 THE PROJECT



PICTURE 4. A screen capture of the start screen of Fast Food Simulator

Fast food simulator is an indie video game project by a small team consisting of a game designer, a programmer and a sound designer. The synopsis of the game is that the player is at his first summer job in a fast food restaurant and has to deal with the high-pressure work environment of the restaurant industry. The player has to get through 30 workdays to finish the game. The workdays will vary in content slightly as every other day the player will be working on the front desk of the restaurant and the kitchen. (Fast Food Simulator 2022)

3.1 Core gameplay loop

The core gameplay loop is a concept of what the player is doing in the game on a moment by moment basis. There can be multiple layers of gameplay loops such as primary, secondary, tertiary et cetera. For a basic example of this, in the arcade game Pac-Man (1980) the primary gameplay loop is eating pills and avoiding ghosts. The secondary is to eat all the pills in the current level, thus getting all the points available in the current level. The tertiary is getting through enough levels to achieve a high score. (Ben Croshaw, 2019.)

The primary gameplay loop of Fast Food Simulator is making fast food meals from ingredients. The secondary loop is completing the full orders of the customer correctly. The tertiary loop is completing enough orders correctly to earn as much money as possible during the workday. The quaternary loop is to get through all the workdays to finish the game. Every workday has a rush hour, where the player has to deal with many more customers than during the rest of the workday.

3.2 Sound design

The sound design for Fast Food Simulator should reflect the chaotic nature of fast food restaurants, while still maintaining a clear enough aural environment so the player knows what is happening through the audio cues. The purpose for the sound design was to create a unique sound for every action of the player as well as clear notifications on other events happening in the game such as when a new order is received from a customer and when the rush hour starts or ends.

3.3 Music

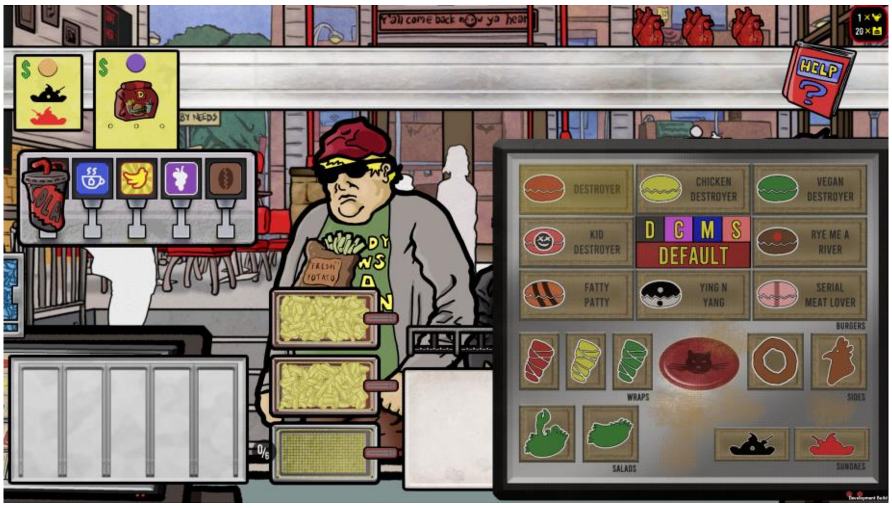
The music for Fast Food Simulator is an important part of creating the aural atmosphere. The music should reflect the in-game events so the player feels the difference when working on the front desk and in the kitchen, as well as the difference of the rush hours compared to the normal work time. Stylistically, the music takes influence from 80s and 90s power rock especially in the main themes of the game. Other influences include the whimsical soundtrack of Conker's Bad Fur Day (2001), Dark Souls (2011) for the secret final boss, and various funk bands. The front desk music is more light-hearted and feelgood music that sounds like its playing on the in-game radio. When playing in the kitchen, the music is more gritty and dirty funk, reflecting the shady business that could be happening in the back of a fast food restaurant.

3.4 Gameplay features

Fast Food Simulator is divided into two types of gameplay. When the player starts their playthrough, on the first in-game day, they work the front desk, taking in orders, ordering things from the kitchen and making the meals. After a few in-game days, the player gets to work in the kitchen, making the individual burgers, burritos, fries and so on. Between work days, the player can choose what to do after work, and these choices affect the gameplay in different ways, such as making the customers more patient and wait longer for their orders or the ability to make fries faster.

3.4.1 Front Desk

The front desk is modelled after a generic fast food restaurant and it features a touch panel, where burgers and tortillas are ordered from the kitchen; a soda fountain; a frying station; a holding station and a freezer for desserts. Customers arrive at pseudo-random times to order different food items. The player must complete the orders correctly or the game will end after 3 failures. All of the different features have their own distinct sounds. The overall goal of the sound design in Fast Food Simulator was to create impactful and high quality sounds and more importantly, for the player to be able to decipher what they are doing even if they were playing with their eyes closed. In practice this was achieved by having distinct sounds for each food item as well as sounds for when the player hovers over different options with the mouse.

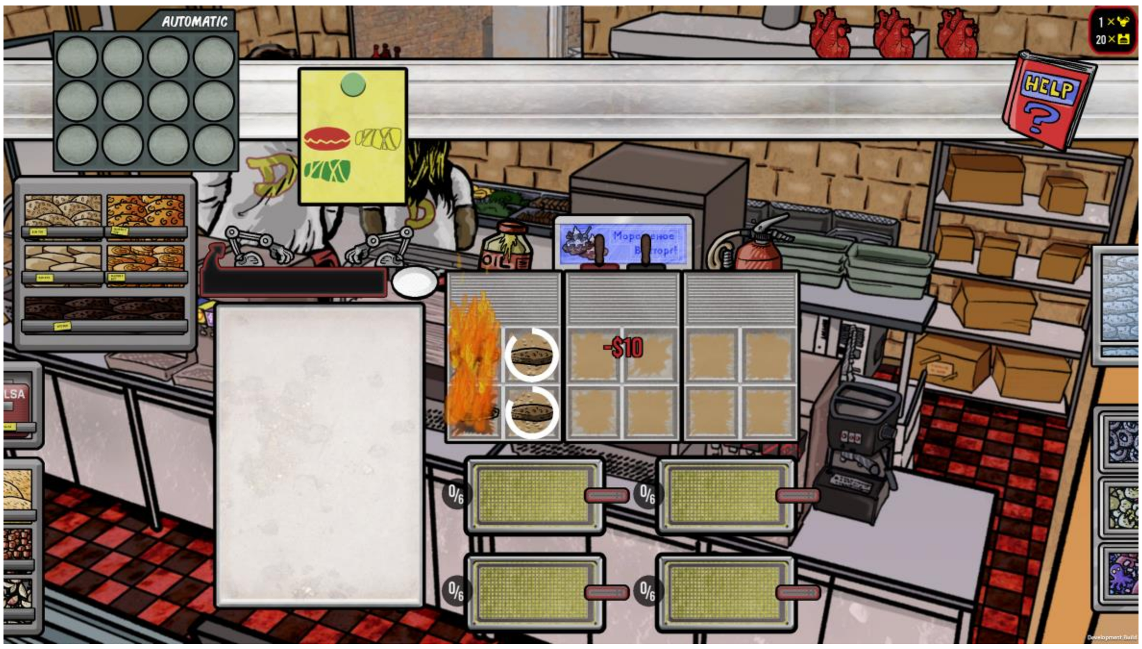


PICTURE 5. A screen capture of the front desk.

3.4.2 Kitchen

The Kitchen is the second gameplay type in Fast Food Simulator. After the player has gone through the first few days working the front desk, the player moves on to work in the kitchen. After working in the kitchen for the first time, the player will work on the front desk and in the kitchen on alternating days until the end of the game.

The main gameplay features of the kitchen are the grill where the player cooks the food and the assembly station where the player assembles the burgers. On the first day in the kitchen the player only has to make the regular Destroyer burgers but like on the front desk, every day there is a new feature that adds challenge to the gameplay.



PICTURE 6. A screen capture of the kitchen showing what happens when the player fails to flip the burgers.

3.4.3 Extras

By doing their work successfully, the player earns money. On the end of each day, the player returns to his home. There, the player can check their calendar and financial situation, buy furniture for their apartment and purchase upgrades which give the player options to optimize the gameplay for their playstyle. For example, the upgrades can make the customers less patient but give bigger tips, so if the player enjoys a challenge with bigger rewards, they can choose to buy those upgrades. The game features a final boss; a final challenge that appears at the end of the last workday. This adds a twist to the gameplay as now the player has to be able to decipher the orders of the annoyed customer from the riddle-like messages that appear on the screen. For dramatic and comic effect, the sound design completely changes for this final battle and the music and sound effects become more cinematic with the music featuring church organs, a dramatic choir and bombastic drums.

4 PRE-PRODUCTION

On his Sound on Sound article on pre-production, Neil Rogers defines pre-production as a broad term that refers to the work done on a project before the actual production begins. Pre-production usually includes planning, scheduling and creating demos. (Neil Rogers 2017.) Pre-production for Fast Food Simulator started in the summer of 2019. The pre-production included planning meetings, creating design documents and creating demos for the different audio parts.

4.1 Planning

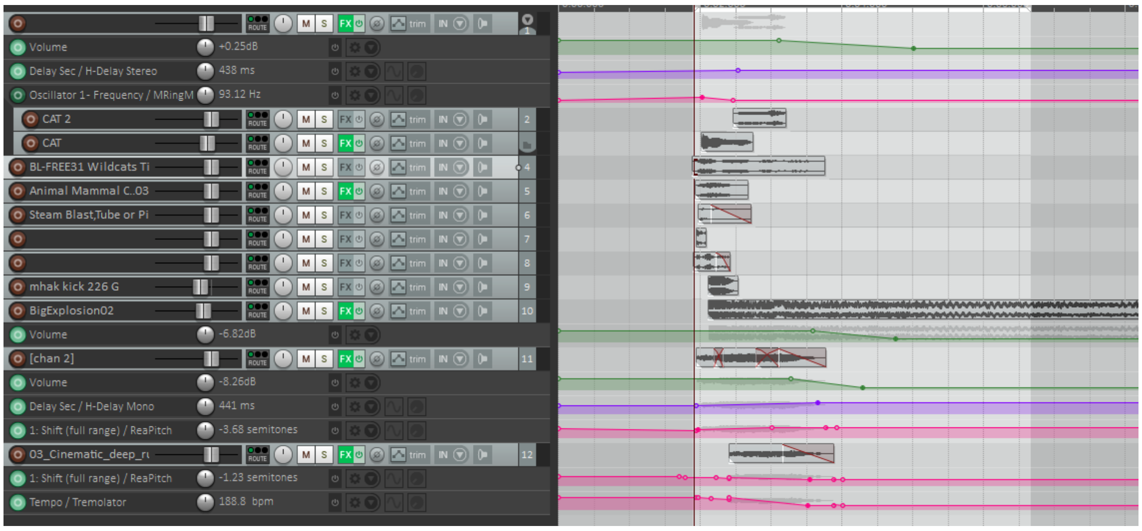
The original plan for the audio for Fast Food Simulator was to create six music tracks that would be used in game, three for when the player is playing the Front Desk and three for the Kitchen. In addition to that there would be a track for the main menu as well as for the trailers and promotional material. Sound effects and general sound design should be impactful and unique as well as of professional quality. The planning included creating sound design and music demos for the sketches that were provided by the visual department. When development began in August 2019 it quickly became clear that there would be a need for more music tracks as the programmer and game designer started developing new features for the game. The new features would require different styles of music as the gameplay would be different than the core gameplay loop

4.2 Work methods / Software & hardware / Tools

The basic workflow on the audio production of Fast Food Simulator was the following. First, the programmer and visual designer work together to create new features for the game. When the new feature is implemented into a build of the game and animated to be in its final form, a recorded video of the new gameplay feature would be created. Then the video is imported into Reaper for the sound designer to create sound effects for the new feature. When the sound effect is finalized, it is exported in WAV-format. The exported sound effects are then imported in Wwise where the audio is finally implemented into the game using events that the programmer has implemented into the game code.

4.3 Reaper

Reaper is a digital audio workstation (DAW) developed by Cockos, Inc. The main pros of using Reaper compared to other DAWs are its accessibility, price and being light-weight, in other words, it can run on most computers. (Cockos Website 2021.) It also comes with free basic plugins which are usable even in professional quality productions, including an EQ, compressor, multiband compressor, delay, pitch shifter and reverbs and more. Reaper also supports 3rd party VST-plugins which can be found online for free.



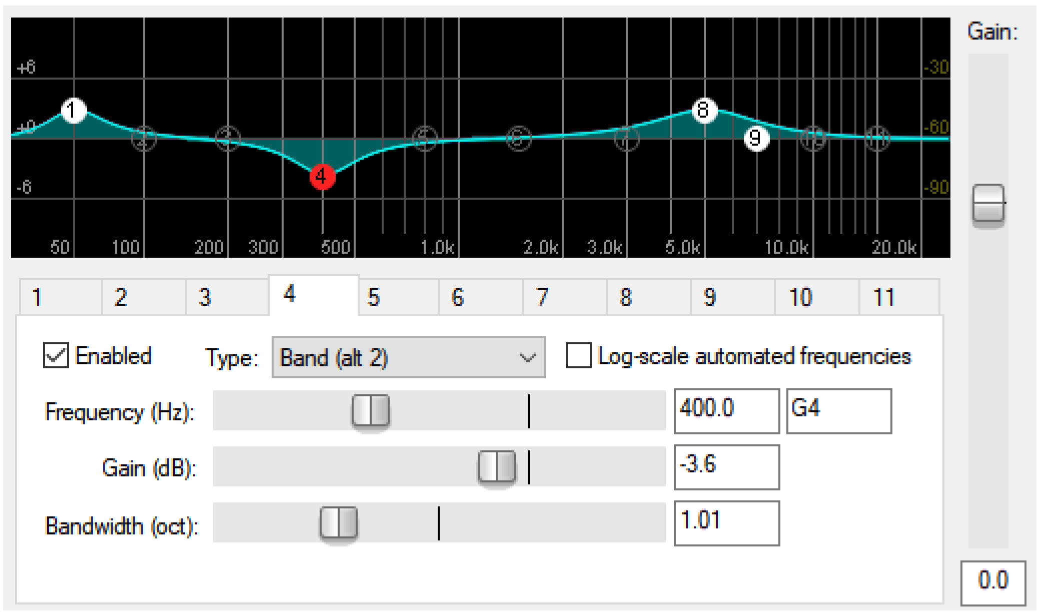
PICTURE 7. Screen capture of the timeline in Reaper. (Reaper 2022)

4.4 Audio Plugins

Audio plugins are additional programs, usually used inside a DAW or middleware programs to affect the audio signal in different ways. There are three main categories of what plugins are used for: Audio signal processing, audio analysis or sound synthesis. (What are DAWs, Audio Plugins and MIDI Controllers? N.d.). These three categories can be further divided into sub-categories which are listed below.

4.4.1 Equalizers

Equalizers or EQs as they are often shortened are tools used for controlling the frequency response of an audio signal (Picture 1). The frequency response areas are usually referred to as Hertz (Hz). Equalizers are used in many phases, recording, mixing and mastering and in all kinds of audio production, whether it be music, sound design or live sound reinforcement. EQs have many uses such as eliminating unwanted frequencies in an audio signal or to reinforce other frequencies to make instruments or voices easier to hear. The human hearing range is around 20hz to 20,000hz so most equalizers work around that range. (Rémy Pujol 2018.)


Enabled ✓ Type: Band (alt 2) Log-scale automated frequencies ☐
Frequency (Hz): 400.0 G4
Gain (dB): -3.6
Bandwidth (oct): 1.01
Gain: 0.0

PICTURE 8. Screen capture of ReaEQ, a graphic equalizer plugin. (Reaper 2021)

4.4.2 Dynamic processors

Where Equalizers are used to balance the frequency responses of the audio signal, compressors are used to balance the volume levels of the signal, making louder parts of the signal quieter and the quiet parts of the signal louder. In other words, compressors are used to reduce the dynamic range of the audio signal. This can be very useful in audio mixing as when the number of audio signals increases, so does the amount of volume variation. Compressors are used to get rid of the unnecessary volume variation, making the audio signals easier for the mixing engineer to control.
Compressors usually have many different parameters to change. Input gain controls the level of the audio signal that is going into the compressor. Threshold is used to set the audio level at which the compressor starts effecting the signal.

Ratio effects the amount of compression of the signal after it goes over the threshold level. 1:1 ratio means there is no compression at all. A ratio of 2:1 means for every 2 dB of the signal that goes over the threshold, there is only 1dB of output. Compressing an audio signal with a compression ratio of 10:1 or more is more commonly referred to as limiting. Limiters are just compressors with a high compression ratio even though they are usually being marketed as separate plugins. Limiting is usually used in the very last spot on the audio signal processing chain as a safeguard against sudden peaks in the audio signal.
(Mike Senior 2018, 173-176; What is an Audio Compressor 2016.)

Multiband compressors, also known as dynamic equalizers are compressors that can control the dynamic range of specific frequencies. They are essentially a combination of a compressor and an equalizer. So instead of compressing the audio signal with the width of the entire frequency spectrum, multiband compressors can affect specific frequency ranges. For example, if there is a need to compress the low end of the audio signal but not the high ends, a multiband compressor can be used to select the specific frequencies where the compression starts and when it should stop. (Bradley Marcus 2019.)

4.4.3 Delay

According to an article about delay on the website Teach Me Audio (2020), delay is an effect where the same audio signal is repeated at a later time, also sometimes referred to as an echo effect. Usually a delay effect plugin has different parameters for the time and volume of the echo as well as how many times it is repeated.

4.4.4 Reverb

Reverb is an effect that is created in the real world when soundwaves reflect off different surfaces in a space. Every one of these reflections is basically creating a simple delay effect and when multiple reflections are combined, a reverberation effect is created. There are multiple types of reverbs, most of which can be replicated with digital plugins. The digital plugins are used to emulate either natural reverbs such as a rehearsal room or a concert hall; physical artificial reverbs such as plate or spring reverbs or they can be completely digital artificial reverbs. (Ken Pearsall 2020.)

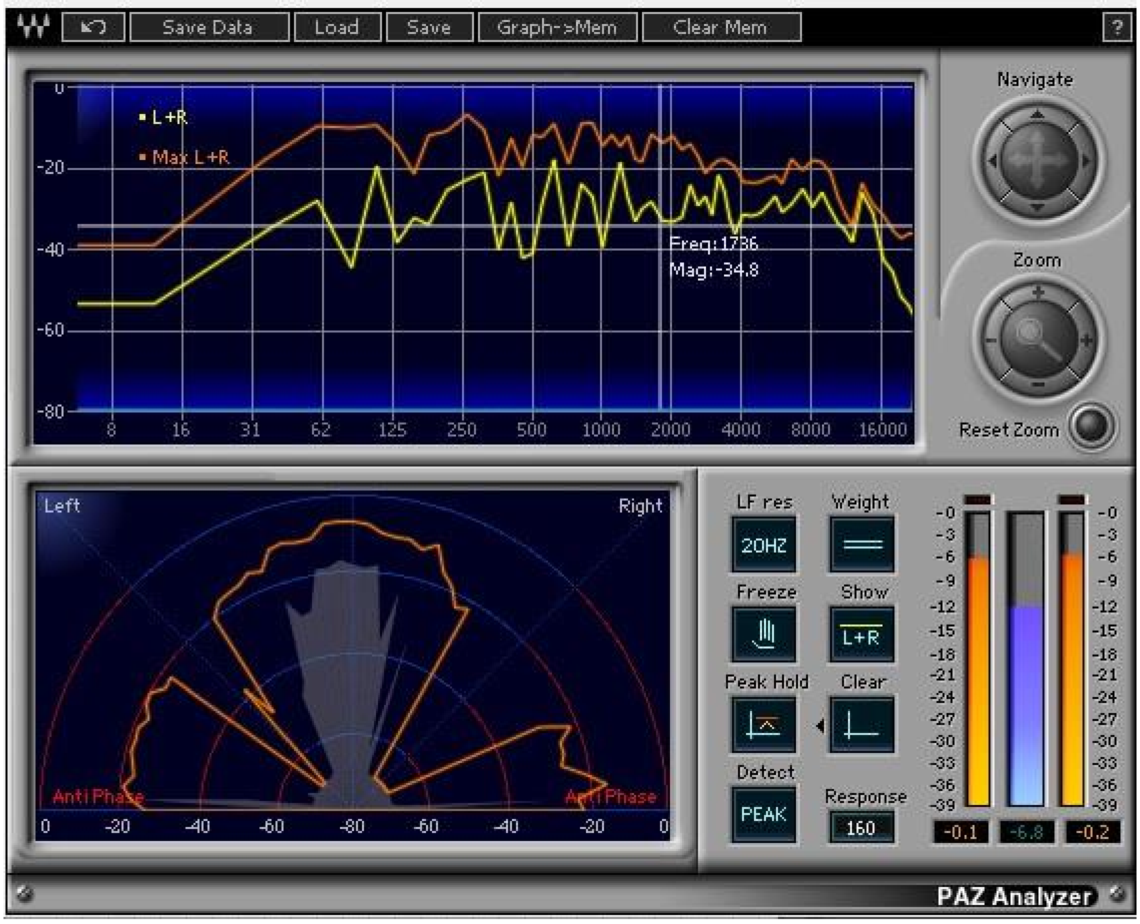
4.4.5 Pitch shifters

Pitch shifters are tools used to change the pitch and formants of an audio signal. The formants of the audio signal are determined by the source of the signal. The formants are the frequency regions created by the resonances of the audio signal source. For a musical instrument, a guitar for example, the formants come from the resonances of the body of the guitar, the types of strings used and all other distinct characteristics the guitar has.
In music, pitch shifting is mostly used to fix vocals that are sung slightly out of tune or instruments that have not been tuned correctly. In sound design there are more possibilities for pitch shifting, such as creating unnatural sounds from natural sources like lowering human speech to make it sound like a fictional monster. (Computer Music 2015.)

4.4.6 Virtual instruments

Virtual instruments are plugins that are used to create audio signals. They can be emulations of real-world instruments or synthesizers, or they can be completely digital synthesizers that generate their own sounds from scratch without having to use samples. For sound designers, virtual instruments can be useful when creating sounds as a sound designer can create his own instruments from the samples he has recorded. (Computer Music 2008.)

4.4.7 Analyzers



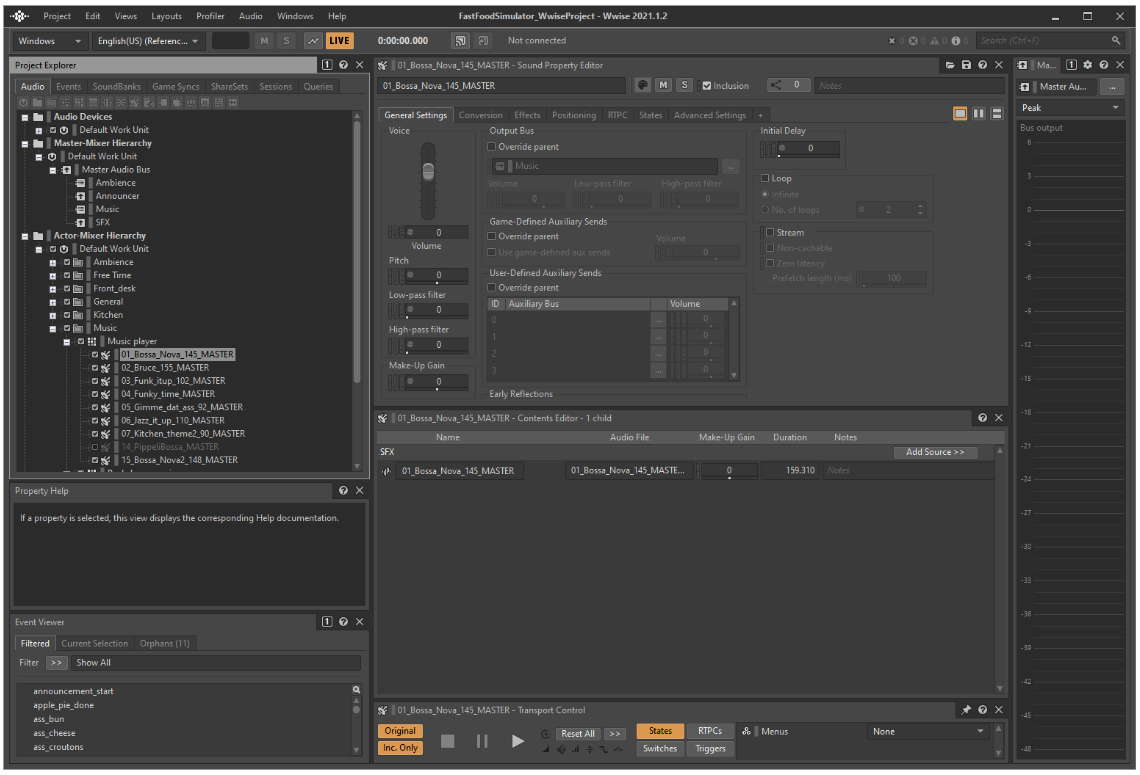
PICTURE 9. Screen capture of PAZ Analyzer. (Reaper 2021)

Audio analyzers are used to measure different parameters of an audio signal. The main parameters are level and gain, frequency response, phase and cross-talk. (Mike Senior 2018, 370-371.)

4.5 Wwise

Wwise is an audio middleware program used to implement audio into video games. Audio middleware is a tool used by the sound designer to integrate sounds more easily to the game engine. Audio middleware programs can provide the game with functionality that is needed for modern video games such as pitch and volume randomization, dynamic or procedural music creation or audio sample randomization. (Horowitz 2014, 134-135.)

Wwise works with the game engine with Events. The Events are created in Wwise and then integrated into the game engine. When the desired action happens, the game engine calls for the event and the event is played through Wwise. The sound designer can continue to work with fine tuning the events without having to integrate it into the game engine again. An event can be very basic such as playing or stopping a sound or more complicated like removing specific frequencies from the sound. For example, when a player is moving to a new area in an adventure game, an Event can be triggered that starts playing new ambient sounds as well as changing the reverb of the environment. (AudioKinetic WWise Manual n.d.)



PICTURE 10. Screen capture of Wwise. (Wwise 2022)

4.6 Scheduling

In large scale video game projects, audio is often pushed to the end of the development cycle. Even large video game companies outsource their audio to be done by a different company. However, it might be useful to have the audio designer or design team working on the project even in the early days of development. A well thought-out audio pipeline and a cohesive vision can help bring a higher quality end product as the audio team and programmer team have a clear picture of what they are working towards and are able to collaborate seamlessly. Having the audio team available right from the start gives the developer a chance to plan the schedule more accurately as the audio team provides expertise on how much time the production and implementation of the audio will take. (Somberg Guy 2016, 166-169.)

When planning the schedule, it is pertinent for the development team to create clear goals and milestones for different stages of the game development. For example, when the audio pipeline should be ready and when all of the audio assets should be available. This helps the programming team as well as the audio team plan their schedule better and create more cohesive builds of the game and what is going to be in the next build. (Somberg Guy 2016, 166-169.)

5 PRODUCTION

The production for Fast Food Simulator started in the Fall of 2019. According to the plans made in pre-production, the game production started by focusing on the Front Desk. The audio production started by creating distinct musical themes for each section of the game. The main sections include, the Front Desk, the Kitchen, main menu and the final boss. The audio production work was periodical, when a new game build was ready, new music and sound effects could be created. According to Monika Rani (2019) a software or game build is the act of translating the source code made by the programmer to create an executable program (Rani 2019).

5.1 Recording

Fast Food Simulator had multiple recording sessions as there was a need for in-game sounds such as human voices and music. The in-game audio contained human voices in the in-game radio shows as well as the advertisements inside the workplace as well as the announcers. All the recordings were done using Reaper.

The idea for the announcers was two-fold, the male announcer performing "pre-recorded" advertisements with high energy and the female announcer being the total opposite, an actual employee of Destroyer burger, a jobsworth with no passion or energy. The announcements occur randomly throughout the day, randomized with Wwise. The in-game announcements sound like they are coming from a low-quality loudspeaker. The announcements were recorded at a home-studio with a Shure SM58 microphone and they were recorded on the same sitting which was helpful for the voice actors as they could stay in character throughout the session.

For advertisement purposes, a few podcast-style radio shows were also recorded. The in-universe purpose of the radio shows is to work as positive PR for the Destroyer Burger corporation. They feature interviews with the mascot of Destroyer Burger as well as customer confessionals. The radio shows were recorded with the help of some external voice actors but for budget reasons, most of the voice acting was done by the development team. These radio shows are found inside the game as well as used for advertisement purposes in the real word on social media. The radio shows were recorded with Shure SM58 microphones.

The recording sessions for the music blended into the composing process as all music was composed, recorded, produced and mixed all in the box. Working “in the box” means doing all the work on a computer using software (Mike Senior 2018, 80.). The melodies and virtual instruments were recorded with a MIDI-keyboard through various virtual instruments depending on the song in question. Recording with a MIDI-keyboard is very useful as the melodies are saved as MIDI. MIDI is easy to edit, thus saving time as the parts don’t have to be played perfectly on the recording an any mistakes or wrong notes can be fixed quickly in editing. Guitars were the only real instrument that was recorded and they were recorded through a Line 6 UX2-sound card.

5.2 Editing

Sound editing includes cleaning up the recorded audio files by cutting out unnecessary parts and removing possible recording errors such as digital glitches or unneeded noises. Editing can also be used to change the tempo or timing of the recording which is key to synchronising the sound effect to the action happening in the game. Naming the audio files is an important part of the editing process. The files should be named according to what they will be used for in the game, not what the original unedited file was. (Viers 2008, 136-144.)

In Fast Food Simulator, all the editing was done using Reaper. The editing process included cutting and trimming the audio files as well as cleaning the audio files with the iZotope RX7 audio plugins. The editing process for Fast Food Simulator was not linear as the editing was done in the same Reaper project files as the sound design and mixing but the key rule of editing was to have the files as ready as possible when they were imported to the Wwise session.

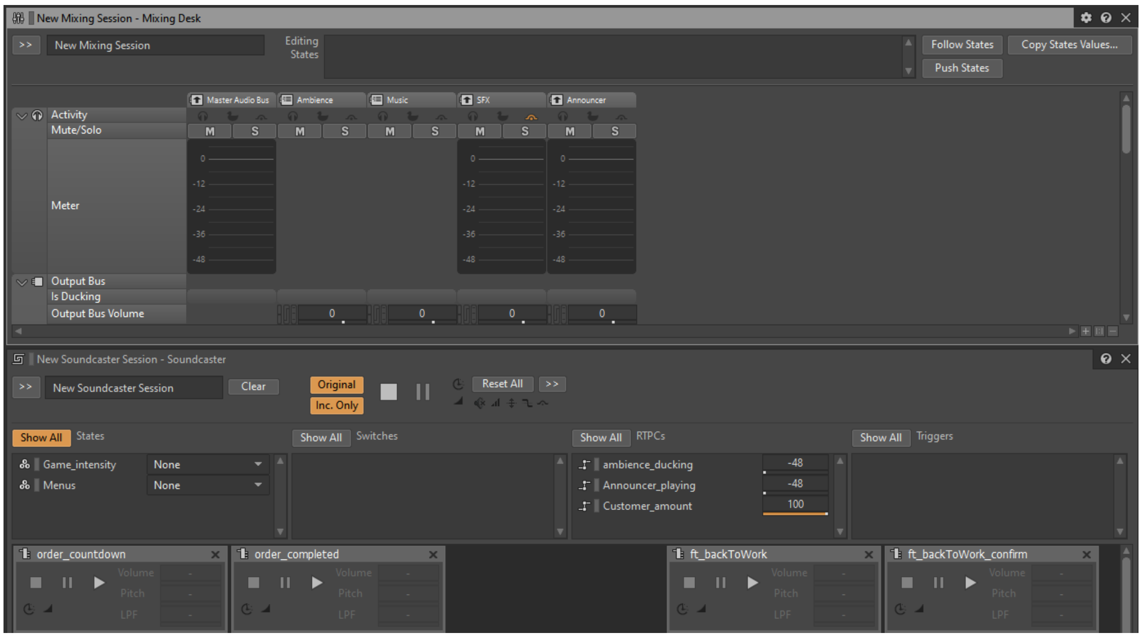
5.3 Designing

While recording and editing is a technical process, sound design is a more creative process. In essence, sound design for video games is creating the sounds for the actions happening in the game. Sound design is more art than science and it includes as many techniques as there are sound designers but it can be boiled down to a few key elements: Dynamics, layering, pitch shifting and sweetening. Dynamics is the difference of the highest volume and the lowest volume of the sound effect. Compression can be used to lower this difference. Layering is the act of playing multiple sounds on top of each other. Sweetening is changing a realistic sound to fit better in a game by adding additional elements to the sound i.e. adding a kick drum sample to a punch sound effect to give it a more impactful sound. (Viers 2008, 154-160.)

5.4 Mixing and mastering

Mixing is an essential part of video game audio. Proper audio mixing helps the player determine what is going on in the game. With music, ambience, voice-overs and sound effects, there can be dozens or even hundreds of sound layers playing at the same time. Even with all of those layers, the audio shouldn't overpower the player and instead should serve the gameplay experience. In-game music can be especially challenging to mix, if its not composed to fit the game, for example a background music track can't have too many melodical elements as that might interfere with crucial audio information such as voice-overs. (Marks 2008, 323-324.)

Mixing in Wwise can be done using the mixing desk (Picture 5). The mixing desk works similar to a traditional DAW, where you can change the volume and panning of Wwise events as well as route them to the different mix busses.
(Audiokinetic Wwise manual, Mixing. N.d.)



Game_intensity None
Menus None
ambience_ducking -48
Announcer_playing -48
Customer_amount 100

PICTURE 11. Screen capture of the mixing desk in Wwise

Mastering game audio is similar to mastering a music album, although mastering sound effects is closely related to the mixing process. The key element is to listen to all the finished sound effects in relation to each other and make sure their perceived loudness and relative volume are well balanced. For example, an explosion should be loud and a button click should be quiet but voice-overs should be consistent in volume. (Marks 2008, 332-333.)

6 IMPLEMENTATION

6.1 Working with the programmer

Historically, the primary task of the video game audio designer has been to design the sound effects for the game in a traditional DAW and deliver them to the programmer in an uncompressed format. It was then the responsibility of the programmer to implement the playback behaviour for the sound in the game as well as the sample rate and compression settings. Audio middleware tools have since moved these responsibilities from the programmer to the audio designer. This saves programming resources while increasing the overall quality of the game. (Somberg 2016, 165.)

While large scale game projects might have a separate audio programmer and multiple audio designers, in the three man team of Fast Food Simulator all the audio responsibilities are on one person. These responsibilities include the sound design and implementation of all the sound effects in the game as well as the composition, mixing, mastering and implementation of the soundtrack. This combined with the need to provide music and sound design to the promotional material as well meant that communication and organization is the key element to keep things working smoothly.

To get Wwise to work in the game, the programmer has to integrate Wwise into the game. This integration program can be broken into four separate functions. The initialization function loads the Wwise modules, such as the sound engine, and soundbanks into the game. The Playsound function ties the Wwise event into a game object, this is what actually triggers the sound to play. The game loop function sends messages to Wwise telling it to process the next audio frame such as play events and game object positions. Game object position in a 2D game basically means the left-right panning of a sound. The last function needed is the termination function which unloads the soundbanks and the modules which were loaded in the initialization function. (Somberg 2016, 77-78.) This integration gets complicated quick and this is one of the key reasons why when communicating with the programmer, it is important to keep track on what features will be coming to the next build of the game and how they will be implemented. For example, things such as the randomization of the messages the in-game announcer delivers has to be discussed as it can be done through Wwise or through the source code. There are various tools that can be used to communicate effectively such as using an audio asset list and using Trello.

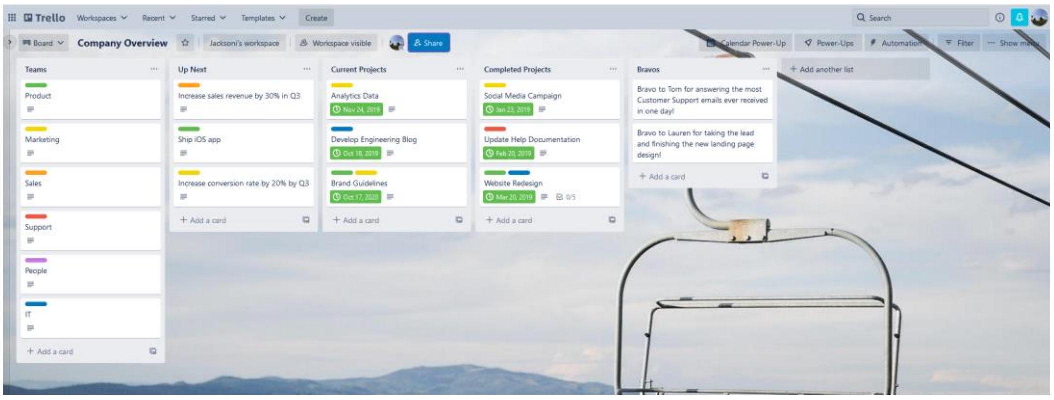
6.1.1 Audio asset list

When working on a video game, whether it is an Independent title or a big budget AAA production, it is very important to keep your assets organised. An essential tool that is used for organising assets is an Audio Asset list. An audio asset list is a spreadsheet or a document that helps the sound designer and development team keep track of all the audio files that are going to be used in the game. The asset list should have helpful information of all the assets such as; the Wwise event name, the scene or situation where the audio file will be called, a description of what the file contains, a notes section where the sound designer or programmer can leave comments and the design and integration status. The design and integration status can be colour coded in the spreadsheet to represent what state the audio file is in. (Anne Sophie Mongeau 2016.) For Fast Food Simulator these states were marked with a traffic light pattern; red meaning that the work for the asset has not started, yellow meaning the work is ongoing or it has a problem and finally green meaning the asset is ready and implemented into the game properly.

6.1.2 Trello

Trello is an online project management application that is used to organize the workflow of a project between multiple people (Kyle Wiggers 2019). It is used to show an overview of a project, who is working on what and at what state the different parts of the project are in (Trello Help Guide n.d.). Trello is used by many industry professionals such as Tom Salta, a composer who has worked on such titles as Tom Clancy’s Ghost Recon Advanced Warfighter 1 & 2 by Ubisoft and Halo Master Chief Collection by Microsoft Game studios and Chris Rickwood, known for his work in Age of Empires Online by Microsoft and the Orcs Must Die!

Series (Marks 2008, 148, 267). Trello is used in the development of Fast Food Simulator.



Teams: Product; Marketing; Sales; Support; People; IT

Up Next: Increase sales revenue by 30% in Q3; Ship iOS app; Increase conversion rate by 20% by Q3

Current Projects: Analytics Data (Nov 24, 2019); Develop Engineering Blog (Oct 18, 2019); Brand Guidelines (Oct 17, 2020)

Completed Projects: Social Media Campaign (Jan 23, 2019); Update Help Documentation (Feb 20, 2019); Website Redesign (Mar 20, 2019) 0/5

Bravos: Bravo to Tom for answering the most Customer Support emails ever received in one day!; Bravo to Lauren for taking the lead and finishing the new landing page design!

PICTURE 12. Screen capture of Trello.

6.2 File naming methods

Along with having an audio asset list, a good file naming convention is essential in a project of any scale. Having a consistent naming convention for all your files, whether it be session files, game assets or source material, is key to keeping a project organized. There's always a chance that some files will need to be improved later down the line and then it is crucial to be able to decipher what files relate to what. (Marks 2017, 446.) In some projects, the development team might already have written file names for the audio assets into the code. (Marks 2017, 506.) Using an audio asset list and agreeing with the development team about file names helps tremendously with keeping things organized (Mongeau 2016). The audio asset list for FFS contains only the WWise event name, which is the only relevant name for the programmer to implement the events into the code.

It is the sound designers job to keep his own recorded assets organized. Organizing is imperative as even during one project the recorded asset count can reach thousands. A conventional method to name files is using the following order.

1. Category
2. Noun
3. Verb
4. Description
5. Number

For example, if you have a recording of a seagull screeching on a beach, the file could be named as follows: Animals_seagull_screech_beach_01. A good naming method will help the sound designer find the files they are looking for quickly. (Ric Viers 2008, 149.)

6.3 Formats, sample rate and bit depth

There are three audio file formats that are usually used when recording, editing and mastering audio. These formats are .WAV (Waveform Audio Format), .AIFF (Audio Interchange File Format) and .BWF (Broadcast Wave Format). These three formats use uncompressed pulse code modulation (PCM) to achieve a lossless quality in audio files. (Viers 2008, 113.) The problem with using lossless or uncompressed audio files in video game implementation is that the file size will be large. The solution for this problem is audio file compression. An industry standard CD-quality audio file format is 44.1 kHz, 16-bit, stereo .WAV. A one minute long audio file of this format takes 10.6MB of space. Popular audio file compression formats such as .OGG (Vorbis) and .mp3 can shrink the same file size to around one tenth of the original size, to 947kb and 940kb respectively. The problem with compressing the audio file is the loss of sound quality. (Marks, p.222-223, 2017.) In Wwise, imported audio files have to be in .WAV format. Wwise can then convert the audio files to the format the user chooses. The conversion formats available in Wwise are: PCM, Vorbis, Opus, ADPCM and AAC. The two most popular formats for Wwise are uncompressed PCM and Vorbis, the tradeoff between them being that PCM uses more memory but less CPU power, while Vorbis uses more CPU as the files need to be compressed but has a smaller file size so it uses less memory. (Wwise certification 251 course materials n.d.)

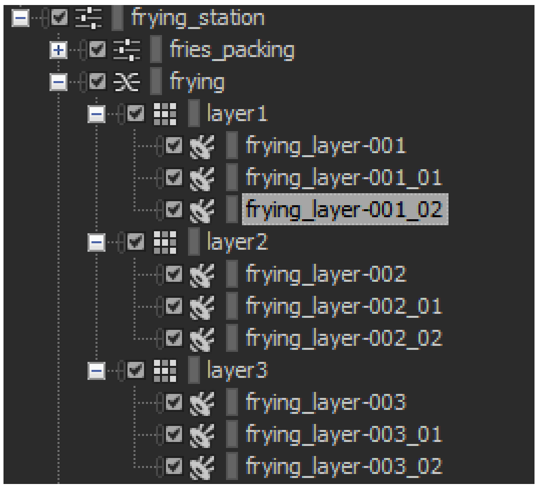
6.4 Implementation with Wwise

With Wwise, importing audio files is very simple. First, the audio files must be imported into the project by going into the “Project”-menu and selecting the option “Import Audio Files”, clicking “Add Files” or “Add Folders”, selecting the audio files you want to import and then clicking import. Wwise then creates copies of the original audio files so no changes to the original files are made. The audio files are then displayed as Sound SFX containers in Wwise, in the Default Work Unit of the Actor-Mixer Hierarchy. The imported Sound SFX can be placed in different kinds of containers. (WWise Help – Importing Media Files for SFX n.d.)

6.4.1 Containers

In Wwise, containers help you group sound files together. There are four different container types, all of which have different settings for playing back the grouped sounds. Random containers can be used to play a different random sound each time the container is triggered. Sequence containers can be used to play the grouped sounds in a specific order. Switch containers play the grouped sounds that correspond to the Switches that are active in the game. Blend containers can play many grouped sounds simultaneously. Blend containers can be combined with random containers to create realistic sounds as every time the sound is played, it sounds slightly different. (Wwise Help – Types of containers n.d.)

In the Wwise project for Fast Food Simulator, random containers were used with blend containers to create more variance in the sound design. For example, in picture 2, the individual audio files are named as “frying_layer” with the number differentiating the unique files. These three files are contained in random containers named layer1, layer2 and layer3. The three random containers are located in a blend container called “frying” which is connected to the in-game event. The event is triggered in-game when the player puts french fries in the fryer. Wwise then plays the event which triggers the blend container, which triggers the three random containers to be played simultaneously. Each of the three random containers trigger one of the frying_layer audio files to be played. For additional randomization, Wwise offers a randomizer in each container for volume, pitch, low-pass filter and high-pass filter. In this example, the randomizer for the pitch was used in the random containers. By using this method, the sound that is heard in-game is always slightly different, creating a more natural soundscape.



PICTURE 13. Screen capture of the Wwise container structure (Wwise 2019.2.7.)

6.4.2 Events

When using middleware, in order for a sound to be able to be heard when playing the game, an event has to be created. Events in middleware are the tools used to communicate with the game code. The programmer creates a line of code relating to an action happening on screen. In that line of code, the programmer creates a hook that the audio implementer will use to link the event to. In Wwise, events can be used to change various parameters of the sound effect, such as starting and stopping a sound and changing the volume, pitch and randomization. (Horowitz 2014, 126.)

For example, in Fast Food Simulator, when starting a new day an event is triggered that starts playing music and background ambience. When a rush hour starts, another event is used to stop playing the regular music and to start the rush hour music as well as playing the sound that signals the start of the rush hour. When the rush hour ends, an event is triggered that plays sound signaling the end of the rush hour, stops the rush hour music and starts up the regular music once again. An another example is the in-game announcements. The announcements are programmed in the game code to play at random times during the day. When the announcement is triggered in the game, an event will start that plays the announcer jingle as well as the announcement itself, while also lowering the volume of the music. When the announcement ends, the volume resets back to normal after a few seconds. (Horowitz 2014, 134.)

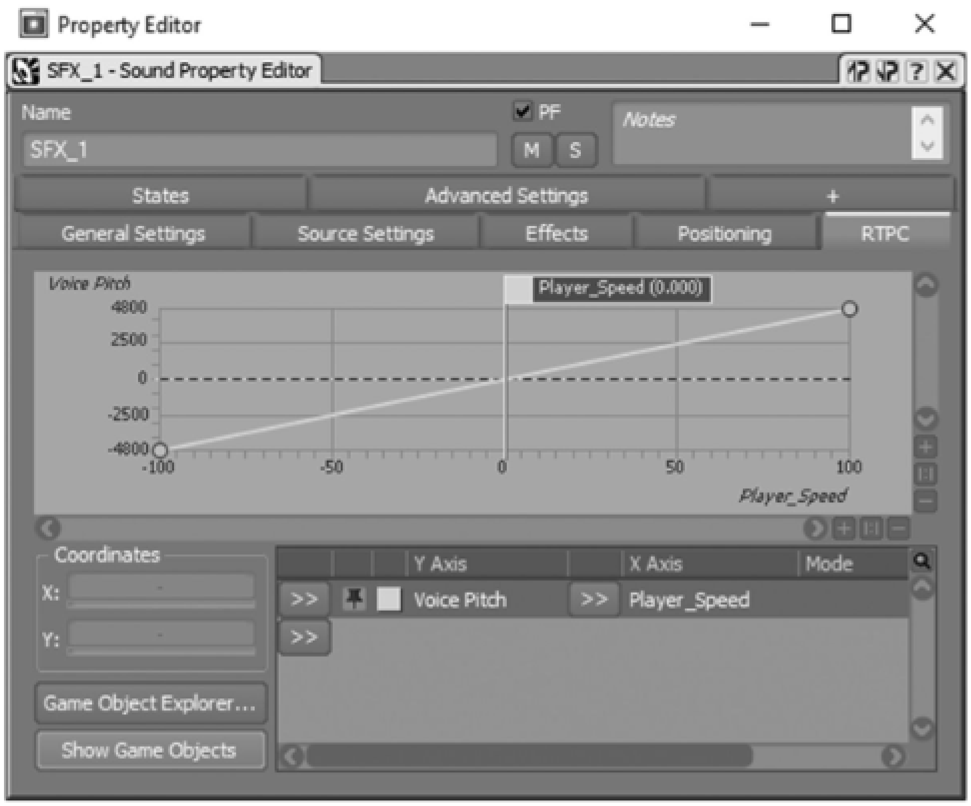
6.4.3 Generating Soundbanks

In Wwise, a soundbank is the way to manage the audio components of a game. A soundbank is a file that contains the Event data, audio files as well as audio data and motion structure and other media files. Soundbanks can be used to optimize how much computer memory the game is using at different points in the game by only loading the files that are necessary. Wwise offers a soundbank layout where different soundbank parameters, such as what audio files and events are included in the bank, can be configured. (Understanding Soundbanks 2021.) In Fast Food Simulator, different soundbanks were used for the Front Desk and the Kitchen. Each time a new sound or event is implemented in the Wwise project, the soundbanks have to be generated for the sound to be heard in the game.

6.4.4 Real time parameter control

Real-time parameter controls or RTPCs are used to control various game parameters. For example, if a parameter is used to move the player in-game, this parameter can be tied to an effect in Wwise. If the players move speed is a value from 0 to 50, this value can be tied to the pitch of the sound effect so when the player moves faster, the pitch of the sound will be higher. (Somberg 2016, 85-88.)

In Fast Food Simulator, RTPCs are used for changing the audio bus volumes so that the most important sounds can always be heard. In practice this means that when an in-game announcement is made, the RTPC is used to lower the volume of the music. This is done by side-chaining the volume of the announcer to the music bus, so when the announcer is being played, the volume of the music is lowered relative to the volume of the announcer.



Picture 14. RTPC controls in Wwise. (Somberg 2016, 87.)

6.4.5 Licencing

There are several options for licencing in Wwise depending on the project scale. For small scale projects, a Starter-license can be used. The Starter license requires the production budget of the project to be less than $150,000 and it offers access to all the Wwise features for free, the only limitation being that a maximum of 500 audio files can be used. The paid licenses are required for games with bigger budgets and the pricing for paid licenses starts at $750. Compared to the free license, the paid licenses offer unlimited audio assets as well as a technical support plan with Audiokinetic. (Audiokinetic Pricing 2021.) A Starter-license was used for Fast Food Simulator as it offered the best value and the audio file requirement was enough to get all the required assets in the game.

7 CONCLUSION

The end results of this thesis show that even a small-scale video game project can be positively affected by the use of audio middleware. The use of WWise in addition to other production techniques such as extensive planning and design documents proved very useful over the course of the production. It helped spread the workload more evenly and gave more insight to the sound designer as well as the programmer as to how audio should work in a game. It also helped in the creative aspects of the game, to create an immersive experience for the player with the use of variable ambient sounds and dynamic music.

The prospects of audio design and implementation for video games are looking bright for small game companies. The low licensing fees and purchase prices of software such as Reaper and Wwise make using them an appealing option no matter the budget. The Wwise framework being easy to implement into the Unity game engine also helps as there are plenty of learning resources on how to do it. Audio middleware also leaves room for more creativity as there already are many different parameters and effects that affect audio and they don’t have to be programmed from scratch.

In conclusion, the key elements in video game audio design are planning, communication, creativity and production. Audio middleware can help with creativity and production but the planning and communication have to be in good order to make full use of the middleware.

References

ONLINE:

Glenn McDonald. 2005. A timeline of videogame music. Read on 4.2.2021. https://glennmcdonald.wordpress.com/2005/12/09/a-timeline-of-videogame-music/

Al Alcorn interview. 2008. Read on 4.2.2021. https://www.ign.com/articles/2008/03/11/al-alcorn-interview

Brookhaven National Laboratory. 2008. The first video game? Read on 4.2.2021. https://www.bnl.gov/about/history/firstvideo.php

Paul Scolum. 2003. Atari 2600 music and sound programming guide. Read on 5.2.2021. http://www.qotile.net/files/2600_music_guide.txt

Arcade museum. N.d. Berzerk description. Read on 5.2.2021. https://www.arcade-museum.com/game_detail.php?game_id=7096

Matthew Partridge. 2015. 22 May 1980: Pac-Man hits the arcades. Read on 5.2.2021. https://moneyweek.com/392564/22-may-1980-pac-man-hits-the-arcades

Chris Kohler. 2010. Nintendo Entertainment System launches. Read on 5.2.2021. https://www.wired.com/2010/10/1018nintendo-nes-launches/

Cockos Reaper Website. 2021. About Reaper. Read on 3.3.2021 https://www.cockos.com/reaper/about.php

What are DAWs, Audio Plugins and MIDI Controllers? N.d. MathWorks Help Center. Read on 4.3.2021. https://se.mathworks.com/help/audio/gs/what-are-daws-audio-plugins-and-midi-controllers.html

Rémy Pujol. 2018. Human auditory range. Read on 11.11.2020. http://www.cochlea.org/en/hear/human-auditory-range

What is an Audio Compressor. 2016. Practical music production website. Read on 3.3.2021

https://www.practical-music-production.com/audio-compressor/

Bradley Marcus. 2019. ICON collective music blog. Read on 2.2.2021.

https://iconcollective.edu/multiband-compression-tips/

Teach Me Audio. 2020. Article on Delay. Read on 2.2.2021

https://www.teachmeaudio.com/mixing/equipment/effects/delay

Ken Pearsall. 2020. Effects guide: What is reverb?. Read on 3.3.2021 https://www.fender.com/articles/tech-talk/pedal-board-primer-reverb,

Computer Music. 2015. The beginners guide to pitchshifting. Read on 4.3.2021 https://www.musicradar.com/tuition/tech/the-beginners-guide-to-pitchshifting-626133

Computer Music. 2008. The beginners guide to: virtual instruments. Read on 3.2.2021. https://www.musicradar.com/tuition/tech/the-beginners-guide-to-virtual-instruments-179640

Audiokinetic. N.d. About Audiokinetic. Read on 3.3.2021. https://www.audiokinetic.com/about/

AudioKinetic Wwise Manual. N.d. Understanding Events. Read on 23.11.2020. https://www.audiokinetic.com/library/edge/?source=WwiseFundamentalApproach&id=understanding_events_understanding_events

Neil Rogers. 2017. Sound on Sound article on pre-production. Read on 16.3.2021. https://www.soundonsound.com/techniques/pre-production

Monika Rani. 2019. What is a software build. Read on 16.3.2021 https://medium.com/webgentle/what-is-the-software-build-all-you-need-to-know-4046b0e674bb

http://www.yannisbrown.com/about-audio-middleware/, Yannis Brown,

https://www.asoundeffect.com/10-inventions-that-changed-the-history-of-game-sound/, Asbjoern Andersen, 2014

Anne Sophie Mongeau. 2016. Game audio asset naming and organisation. Read on 23.11.2020. https://annesoaudio.com/2016/07/07/game-audio-asset-naming-and-organisation/

https://www.wired.com/2017/01/trello-simple-app-worth-425-million-dollars/, Klint Finley (01.09.2017) Avattu 23.11.2020.

Kyle Wiggers. 19.3.2019. Trello limits teams on free tier to 10 boards, rolls out Enterprise automations and admin controls. Read on 23.11.2020.

https://venturebeat.com/2019/03/19/trello-limits-free-users-to-10-boards-rolls-out-enterprise-automations-and-admin-controls/,

Trello Help Guide. N.d. What is Trello? Read on 24.11.2020. https://help.trello.com/article/708-what-is-trello

Wwise certification 251 course materials. N.d. Audio File Compression. Read on 25.11.2020. https://www.audiokinetic.com/courses/wwise251/?source=wwise251&id=Lesson2_Audio_File_Compression#read

WWise Help – Importing Media Files for SFX. N.d. Read on 15.3.2021. https://www.audiokinetic.com/library/edge/?source=Help&id=importing_media_files_for_sfx

Wwise Help – Types of containers. N.d. Read on 15.3.2021 https://www.audiokinetic.com/library/edge/?source=Help&id=grouping_sound_and_motion_objects_to_create_actor_mixer_hierarchy_types_of_containers

Understanding Soundbanks. 2021. Read on 3.3.2021 https://www.audiokinetic.com/library/edge/?source=WwiseFundamentalApproach&id=understanding_soundbanks_understanding_soundbanks, Understanding Soundbanks, 3.3.2021

Audiokinetic Pricing. 2021. Read on 3.3.2021. https://www.audiokinetic.com/pricing/

Audiokinetic Wwise manual, Mixing. N.d. Read on 26.4.2022. https://www.audiokinetic.com/library/edge/?source=Help&id=mixing_audio_using_mixing_desk

Tennis for two on an oscilloscope, Brookhaven National Laboratory 2013. Opened on 4.5.2022. https://upload.wikimedia.org/wikipedia/commons/5/50/Tennis_For_Two_on_a_DuMont_Lab_Oscilloscope_Type_304-A.jpg

The Famicon Disk System,The Vanamo Online Game Museum 2017. Opened on 4.5.2022 https://commons.wikimedia.org/wiki/File:Nintendo-Famicom-Disk-System.jpg

BOOKS:

Aaron Marks – The complete Guide to Game Audio, 2008

Steve Horowitz, Scott R. Looney - The Essential Guide to Game Audio - The Theory and Practice of Sound for Games-Focal Press (2014)

Ric Viers – The Sound Effects Bible (2008)

Mike Senior – Mixing Secrets for the Small Studio (2018)

Theses:

Video Game Console Audio: Evolution and Future Trends, Kyusik Chang, Taeyong Kim, 2007

Joey Pendergrass. 2015. The Rise of Reactive and Interactive Video Game Audio. California State University, Monterey Bay. Master's Thesis; 484. https://digitalcommons.csumb.edu/caps_thes/484

Video Essays:

Ben Croshaw. 2019. Yahtzee’s Dev Diary Episode 4: Shooter Loopy. Youtube video. Published on 25.6.2019. Referred on 11.11.2020. https://youtu.be/yh26jd9UqRw

Games:

Fast Food Simulator. 2022.