CATÓLICA
ESCOLA DAS ARTES

PORTO

SOM PARA VIDEOJOGOS: ESTÁGIO NA VOLT GAMES

Relatório de Estágio apresentado à Universidade Católica Portuguesa
para obtenção do grau de Mestre em Som e Imagem

Vasco Tavares de Almeida

Porto, Julho 2024

CATÓLICA
ESCOLA DAS ARTES

PORTO

SOM PARA VIDEOJOGOS: ESTÁGIO NA VOLT GAMES

Relatório de Estágio apresentado à Universidade Católica Portuguesa
para obtenção do grau de Mestre em Som e Imagem

- Especialização em Sound Design-

Vasco Tavares de Almeida

Trabalho efetuado sob a orientação de

José Vasco Carvalho
Manuel Oliveira Silva

Porto, Julho 2024

Agradecimentos

À minha família que me apoiou ao longo de todo o meu percurso académico, sou vos eternamente grato pelo vosso apoio incondicional.

Agradeço ao Pedro Cabaço, Gonçalo Banha e João Albuquerque por me terem proporcionado esta oportunidade de estágio e pela experiência destes seis meses.

Um muito obrigado a toda a equipa da Volt Games , em especial ao Pedro Costa e Gonçalo Perpétua por serem a melhor equipa de design com quem já tive a oportunidade de trabalhar.

Agradeço de coração cheio aos meus tios Ana e Manuel e as minhas primas Matilde e Maria Ana que me receberam na sua casa de braços abertos.

À Lucía Beaumont, pela motivação para continuar e acabar este projeto.

Agradeço também ao meu primo e amigo Miguel Tavares, por ter sido uma cobaia para as gravações de voz.

E por fim quero agradecer aos meus professores José Vasco Carvalho e Manuel Silva que me orientaram ao longo deste projeto.

Resumo

Este relatório reflete a experiência e o trabalho desenvolvido na área de som para videojogos móveis durante o estágio curricular na empresa Volt Games. Ao longo de seis meses, tive a oportunidade de desenvolver as minhas competências em composição e design de som, adquirindo técnicas especificas de criação de áudio para videojogos. O objetivo com esta investigação é de compreender quais são as diferenças e terminologias necessárias, para produzir design sonoro nesta forma de media interativa. O relatório inclui uma análise de composição musical e de design de som adaptado para videojogos. A Volt Games, proporcionou-me a oportunidade de trabalhar num ambiente profissional aprimorando as minhas competências em som. O documento também inclui uma pesquisa e aplicação prática de implementação por FMOD e Unity, que são dois dos softwares mais utilizados para criação e implementação de som para videojogos, essencial para um designer de som que se queira especializar nesta área.

Palavras-Chave: Videojogos móveis, Design de som, FMOD, Composição musical**.**

Abstract

This report reflects about the experience and work developed in the field of sound for mobile video games during my internship in the company Volt Games. During the six months, I had the opportunity to strengthen my skills in composition and sound design, learning specific techniques to create audio for video games. The objective with this research is to understand what the differences and terminologies are needed to produce sound design in this interactive media. The report includes an analysis of the music compositions and the sound design adapted for video games. Volt Games provided me with the opportunity to work in a professional environment and to improve my competencies in som. The dissertation includes research on implementation using FMOD and Unity, as they are two of the most widely used softwares for creating and implementing video games, making them essential tools for a sound designer specializing in this area.

Keywords: Mobile Video Games, Sound Design, FMOD, Music composition.

Indice

Lista de Figuras

Glossário

Audio Clip - Consiste num segmento ou trecho de áudio.

Envelope AHDSR - Gerador de envelope de ataque (atack), de segurar (hold), decair (decay), suster (sustain) e libertar (release), que pode ser aplicado à amplitude ou a outros parâmetros de som.

"Build" no FMOD - O processo de exportar e compilar o conteúdo de áudio e configurações realizadas no FMOD.

C++ portable - Linguagem de programação orientada por objetos que é utilizada para o desenvolvimento de aplicações. É considerada uma linguagem de programação avançada, sendo utilizada em tarefas sensíveis à performance. O termo portátil (*portable*) é relacionado com o facto do código ser compilado para funcionar sem erros, em diversas plataformas.

C# - Linguagem de programação orientada por objetos. É considerada simples e de fácil aprendizagem. É utilizada para desenvolvimento de aplicações, videojogos e aplicações na web.

Delay (atraso) - Técnica de processamento de sinal em áudio. Consiste numa unidade de computação que grava o sinal de entrada e depois de um período de tempo, reproduz o mesmo sinal.

Efeito sonoro - Qualquer som que não seja música ou fala, criado através de recursos técnicos para representar um som real ou do imaginário.

FMOD Studio Listener - Componente utilizado como recetor de áudio, representando o ouvinte no videojogo.

FMOD Studio Event Emitter - Componente utilizado para integrar uma fonte sonora do FMOD para o Unity.

Github - Plataforma com base na web que disponibiliza aos programadores uma forma de partilhar e armazenar código.

Glockenspiel - Instrumento de percussão que produz um som metálico semelhante ao vibrafone.

Grau musical - Representa a função de cada nota musical numa escala musical e na sua tonalidade. Grau I é a tónica, representa a primeira nota da escala, grau V é a dominante, a quinta nota da escala.

Logic pro X - Estação de trabalho de áudio digital (DAW) desenvolvido pela Apple. Este

software é utilizado para gravação, edição, manipulação e produção de ficheiros de áudio.

Loop - Consiste num segmento de áudio ou secção de música que se repete de forma cíclica.

Middleware - Software que é utilizado como intermediário entre diferentes aplicações ou componentes.

MIDI - Protocolo de comunicação universal utilizado em instrumentos e DAWs.

Mistura - Processo de criar uma mistura coesiva de vários elementos de som e música.

Monday - Software utilizado no ambiente de trabalho que facilita a colaboração e organização de projetos entre diferentes membros.

Mono - Áudio monofónico, em que todos os sinais de áudio estão agrupados no mesmo canal.

Open source - Software disponibilizado gratuitamente e com código aberto, para que seja desenvolvido pela comunidade de utilizadores.

Prefabs - Objetos do Unity que são criados como modelos reutilizáveis.

Reverberação - Também referido como *reverb*, é um fenómeno físico em que o som proveniente de uma fonte sonora, reflete sobre superfícies de um espaço, reincidindo no ouvinte. Este efeito é representado digitalmente com o uso de algoritmos e técnicas de processamento de sinal.

Script - Sequência de instruções escritas numa linguagem de programação que o computador interpreta e executa.

Sampling - Sampling num contexto criativo, remete à ação de reutilizar uma porção de um áudio previamente gravado ou processado.

Slack - Plataforma de trabalho online utilizada para gerir e comunicar projetos.

Sweetener - Processo utilizado para enriquecer a qualidade e claridade do som a partir de pequenos ajustes, sem alterações à integridade do som.

Stereo - Processo de reprodução de som utilizando dois ou mais canais.

Surround 5.1 e 7.1 - Formato de áudio associado ao sistema de reprodução de som. Sistemas 5.1 representam o uso de seis canais de som, cinco canais direcionais e um canal para um baixo, enquanto que os sitemas de 7.1 representam o uso de oito canais de som, sete canais direcionais e um canal para o baixo.

Videojogo Open World - Vídeo jogo em que o jogador pode interagir com os objetivos livremente.

Visual studio - Ferramenta utilizada principalmente para desenvolvimento de software a partir de código.

1. Introdução

Desde muito jovem que tenho interesse por som e música para videojogos. Projetos independentes como, *Celeste* (Maddy Makes Games, 2018) e *Stardew Valley* (Barone, 2016) motivaram-me a aprofundar os meus conhecimentos nesta área. Ao longo da minha licenciatura, tive um primeiro contacto com a área de design sonoro e música para videojogos. Apesar de ter sido fundamental os três anos de licenciatura em música eletrónica e produção musical, procurei especializar-me numa prática que conecta os vários elementos do meu interesse, videojogos e som.

Para continuar com os meus estudos, decidi ingressar no mestrado de som e imagem com especialização em design de som. Durante o primeiro ano de mestrado a cadeira de som para videojogos, proporcionou as técnicas necessárias para ingressar nas minhas atividades práticas desta área. Dado isso, para o segundo ano de mestrado decidi fazer um estágio na Volt Games por ser uma empresa relevante em Portugal que me poderia oferecer uma experiência profissional no mundo dos videojogos móveis.

1.1 Sobre a empresa Volt Games

A Volt Games foi criada em Janeiro de 2020 por quatro fundadores e amigos que partilham do mesmo interesse e paixão por videojogos. A empresa tem como foco produzir videojogos relacionados com desporto, puzzle e hiper casuais para plataformas móveis. Neste momento a empresa encontra-se a trabalhar diretamente com uma das maiores editoras de videojogos móveis, Voodoo.io, sediada em Paris, França, para a qual produzem vários jogos hiper casual (Astle, 2022). Simultaneamente, estão a desenvolver projetos com a Lion Studios, criada em São Francisco, nos Estados Unidos. Além disso, a empresa está a criar e desenvolver videojogos originais.

A Volt Games está localizada em Alvalade, Lisboa sendo, atualmente, considerada uma das maiores empresas de videojogos móveis em Portugal.

A equipa de 15 trabalhadores é responsável por criar o design e desenvolver os videojogos (Figura 1). Os membros da equipa já participaram em vários eventos como WebSummit, Gaming Startup Retreat, e eventos em Singapura, como o Token 2049.

Figura 1 – Equipa da Volt Games.

1.2 Objetivos do estágio

O objetivo do estágio era o de qualificar as minhas competências em edição de som e composição musical, assim como adquirir novos conhecimentos na área de implementação de som por Unity. Tendo sido o único responsável pelo departamento de som durante os seis meses de estágio, foi possível pôr em prática técnicas adquiridas na minha licenciatura e mestrado, e ao mesmo tempo, estar em contacto com o processo de trabalho, e ambiente de uma equipa profissional na área dos videojogos. Esta experiência foi enriquecedora a nível pessoal e profissional sendo que, o facto de já ter conhecimentos em música clássica e eletrónica foi fundamental para poder comunicar as minhas ideias aos colegas da empresa.

Finalmente, o objetivo com o relatório é o de investigar e aprofundar conhecimento sobre implementação de som por Unity. Atualmente, no mercado de trabalho, aos designers de som e compositores, é exigida a compreensão de fundamentos na área de implementação de áudio, sendo necessário conhecimentos prévio prática em certas linguagens de programação e proficiência em software especializado como o Unity, FMOD, Unreal Engine e Wwise.

1.3 Metodologia

Esta investigação tem uma abordagem construtivista, por ser de carácter prático, ao desenvolver as minhas competências com projetos realizados no estágio. O design de investigação é qualitativo por agrupar a parte teórica e prática.

Para a parte prática da investigação de mestrado utilizei o Logic Pro como DAW, e também Unity e Visual Studio como ferramentas no processo de implementação de som. No contexto de implementação de som, fui acompanhado por colegas da equipa que são responsáveis pela componente de programação da empresa.

A investigação e experimentação desta pesquisa é acompanhada por referências bibliográficas, assim como algumas referências artísticas que auxiliam no processo de desenvolvimento do design sonoro e implementação, tais como: *Celeste* (Maddy Makes Games, 2018) um jogo de plataforma 2D, *Royal Match* (Dream Games, 2021) jogo de puzzle, *Match Factory* (Peak Games, 2023) jogo de puzzle, entre outras referências. A investigação consiste em trabalho prático com acompanhamento de equipa e pesquisa de documentação de referências para a implementação em Unity.

1.4 Estrutura do Relatório

A estrutura deste relatório está dividida em quatro capítulos. No capítulo de introdução apresento uma descrição da empresa, os objetivos gerais, a metodologia e a estrutura do relatório. O capítulo de desenvolvimento é dedicado ao progresso do estágio em que menciono o cronograma e as atividades realizadas durante os seis meses. O capítulo de investigação foca-se na pesquisa, explorando música, sonoplastia para videojogos e implementação em Unity. O quarto e último capítulo é reservado às conclusões do relatório em que reflito sobre planos futuros.

2. Plano do estágio

2.1 Atividades

No decorrer do estágio participei em todos os projetos realizados na Volt Games como designer de som e compositor. O escritório da empresa está localizado em Lisboa, isto não abstou a que a Volt Games oferece-se a possibilidade de colaborar com a empresa remotamente. Por não ter acesso a todo o equipamento e material necessário para trabalhar em Lisboa presencialmente, optei por realizar o estágio de forma híbrida.

Fui colocado na equipa de design, constituída pelos dois designers de videojogos, Gonçalo Perpétua e Pedro Costa. A equipa de design era também constituída pelos fundadores da empresa, Pedro Cabaço, João Albuquerque e Gonçalo Banha, responsáveis pela programação dos videojogos. Como único designer de som na empresa, tive principalmente reuniões e comentários destes colegas.

A equipa reunia-se todos os dias por chamada e especificamente às quartas-feiras havia reunião geral com todos os colegas da empresa, incluindo a equipa de arte e de programação. O software utilizado para organização e comunicação da empresa era o Slack, sendo que para as reuniões era utilizado o Google Meets. Utilizamos o Monday como calendário de entregas de trabalho. Na primeira semana de estágio fui também inserido no GitHub da empresa, de forma a facilitar o acesso aos projetos do Unity.

As entregas do trabalho eram efetuadas pelo Slack, no canal próprio para os projetos de som e música. A avaliação do trabalho era recebida durante as reuniões, e também por escrito no canal de som. Ocasionalmente, utilizei o Discord para discutir ideias e resolver problemas com colegas da Volt. Para a organização do trabalho foi utilizado uma ficha de observações com prioridades de sons e alterações necessárias a fazer (Apêndice A).

Trabalhei em vários projetos em simultâneo durante os seis meses, mas atribuíram-me primeiro o projeto, *Golf Stars*, um jogo relaxante de desporto, estilo simulador para o qual necessitei de criar som e música original.

Ao longo do estágio, foram-me atribuídos jogos do tipo hiper casual e puzzle, para a cliente Vodoo.io. Fui responsável pela criação de efeitos sonoros, gravação de vozes e composição de música original. Os jogos realizados durante o estágio foram:

Hotel Match 3D, *Hex em All*, *Brick Blast 3D*, *Bubble Blast 3D*, *Helix Drop 3D*, *Screw Puzzle 3D*, *Shoot Em Up*, *Word Chain 2*, *Craft Alchemy*, *Sort of War*, *Bubble Break*, *Merge Sort*, *Sorted Goods*, *Split Numbers*, *Seat Away* e *Sand Match*.

2.2 Cronograma

Durante o estágio compus música, criei efeitos sonoros, gravei vozes e implementei som por Unity. O estágio teve uma duração de 24 semanas começando no dia 6 de Novembro de 2023 e finalizando no dia 6 de Maio de 2024.

Em Novembro fui integrado na equipa e guiaram-me pelo software utilizado na empresa, comecei a trabalhar de imediato no primeiro projeto, *Golf stars,* fazendo design de som e composição musical para o mesmo.

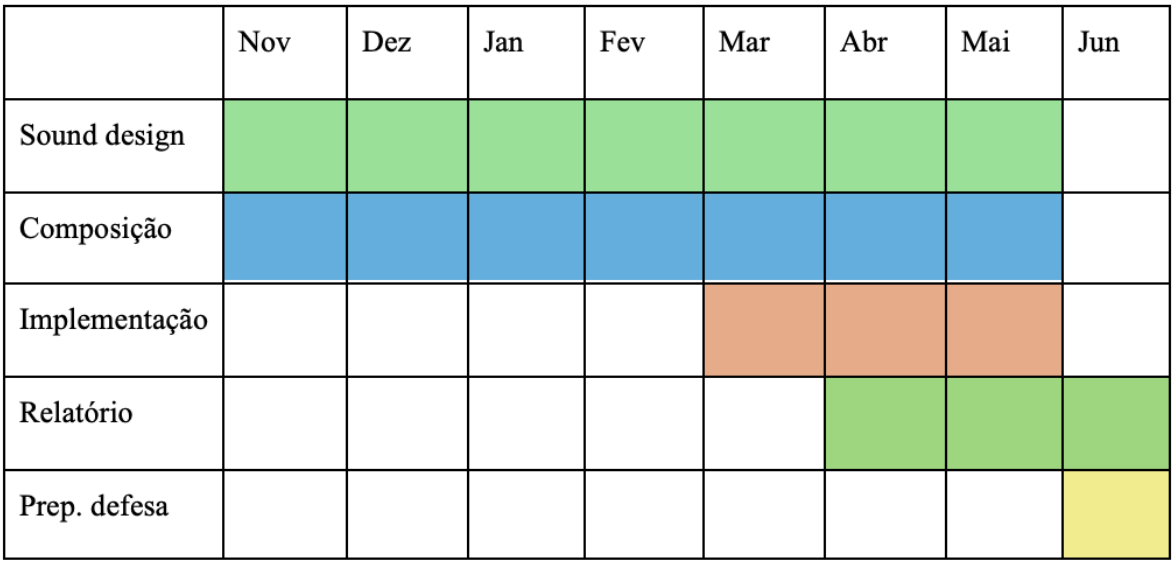


Figura 2 - Cronograma

O cronograma apresentado na figura 2 representa a distribuição do trabalho realizado por mim, de Novembro a Maio no qual desenvolvi o design de som e música dos videojogos. Em Março comecei com a implementação por Unity, assistindo a reuniões dos programadores e tendo algumas reuniões privadas com colegas. Em seguida de Março a Junho iniciei com o relatório e pesquisa. Terminando com a preparação para a defesa no último mês de Junho.

3. Desenvolvimiento do estágio

O trabalho realizado no estágio consistia em três tópicos, composição musical, design de som e implementação de som. A composição musical e design sonoro foram realizados em simultâneo com a mistura ao longo dos seis meses de estágio, e a implementação sonora foi realizada nos últimos meses de trabalho. Neste capítulo estão explicados os conceitos a forma como os apliquei no meu trabalho.

3.1. Música para videojogos

Música para videojogos é uma camada fundamental para imersão sonora sendo a principal responsável pela primeira reação do jogador. Especificamente para videojogos movéis, o som e a música são muitas vezes vistos como algo efémero, embora tenham um efeito significativo de satisfação para o jogador. Os videojogos móveis têm como principal qualidade a verstalidade da sua plataforma. Isto permite aos jogadores que joguem em qualquer lugar e momento mas torna a atividade passageira. Como consequência, o conteúdo dos videojogos é consumido de forma momentânea.

A composição de música para videojogos é um processo delicado no qual o compositor deve não só dominar várias técnicas, mas também necessita de ter a sensibilidade de se relacionar com o contexto do videojogo, e público-alvo para que o resultado seja coerente.

Para compor música de videojogos é relevante compreender os dois tipos de composição temporal: composição linear e não linear. Todas as músicas no geral, contêm componentes de linearidade e não linearidade. No contexto temporal, o aspeto linear da composição estabelece-se com a progressão da música no tempo (Kramer, 1988, p. 20 como citado em Villberg, 2022).

Estas terminologias estão também presentes em música para cinema. Bandas sonoras compostas para cinema são fundamentalmente lineares por seguirem uma narrativa que consiste no inicio de uma ação, desenvolvimento e por fim uma resolução final, isto numa ordem linear. Por outras palavras, a música para filme é criada de acordo com as variações da narrativa previamente associadas a um tempo. Isto faz com que a música seja linear de forma a complementar a linearidade da história. Para videojogos, a abordagem é predominantemente diferente, a temporalidade da ação depende do jogador. Para isso os compositores têm de adotar um sistema não linear que consiste em *loops* musicais que muitas vezes, são descontruções ou variações de uma só composição efetuada verticalmente (utilizando camadas). Isto permite que a música associada a esta multi-media, possa ser infinitamente ouvida sem causar incómodo ou cansaço no jogador (Villberg, 2022).

Por exemplo, no *Super Mario Odyssey* (Nintendo, 2017), um jogo de plataforma *open world*, é possível ouvir música constante durante a *gameplay*. Esta molda-se à medida que o jogador progride no seu percurso. O tema musical adapta-se de forma dinâmica, com o uso de elementos

musicais e texturas, de acordo com as decisões e com o espaço em que o jogador se encontra no videojogo, criando transições impercetíveis ao jogador. Isto é o resultado de uma composição de banda sonora complexa, misturada de forma vertical e eficazmente implementada. Esta é também designada por *non linear adaptive music* ou mistura vertical (Kähärä, 2018).

Adaptive music é a integração de composição não linear a partir de um sistema computacional dinâmico, que resolve às necessidades da imprevisibilidade de um jogador no videojogo. Na técnica de mistura vertical, o conteúdo musical é sobreposto verticalmente. Isto cria a possibilidade de alterar o arranjo musical em função das cenas ou níveis do videojogo. As transições são realizadas de duas formas diferentes. Primeiro, podemos adicionar ou remover elementos seguindo o mesmo ritmo e tempo musical, criando transições fluidas e mantendo a estrutura rítmica interrompida. Outra forma passa por inserir segmentos de áudio que gradualmente criam o final e o início da próxima sequência. Estes elementos de áudio são designados por *stingers*, elementos sonoros ou musicais introduzidos como resposta a uma ação do jogador, garantindo que a música do videojogo responda de forma dinâmica (Dubnob, 2021).

3.1.1. Exemplos de composições musicais

Durante o estágio, pude aplicar os conceitos mencionados anteriormente em diversos projetos. Em geral, para os projetos da Volt Games, compus principalmente música de forma não linear. Os requisitos dos videojogos da Volt Games eram relativamente limitados a nível musical isto porque a empresa produz essencialmente jogos hiper casual. Estes jogos são direcionados para consumidores que jogam esporadicamente durante o dia e por consequência, a música acaba por ser ignorada durante a *gameplay* (Horowitz & Looney, 2014). Como mencionado previamente, o estágio foi realizado em Lisboa de forma hibrida. A empresa, embora dispusesse de um escritório, não tinha possibilidade de ter material como microfones, monitores, sintetizadores, instrumentos musicais, entre outras ferramentas que são úteis para a criação de música. Por isso, as composições criadas ao decorrer do estágio foram elaboradas a partir do uso de MIDI e instrumentos digitais nativos do Logic Pro, entre outros plugins.

O jogo *Golf Stars*, foi o projeto inicial do meu estágio. Este videojogo, do estilo de desporto e casual, consistia num jogo simples e satisfatório, remetendo o jogador a uma experiência de relaxamento. Em conjunto com o meu colega da equipa de design, Gonçalo Perpétua, elaborei uma música de fundo para o menu do videojogo. As referências de jogos propostas pela equipa foram *Neko Golf* (CLOPL, 2022) e *Golf Clash* (Electronic Arts, 2021), para o qual se verificou mais tarde que, os jogos não encaixavam com estilo do *Golf Stars*. Procurei utilizar como referência, algo que remetesse à geração de videojogos da Playstation 2 e por isso, utilizei a música do *Grand Turismo 4* (Polophony Digital, 2005) como referência para o projeto.

Como processo de composição, comecei por desenvolver *loops* rítmicos e harmónicos, a partir de experimentação (Figura 3). Os instrumentos foram escolhidos pela estética que queria alcançar, principalmente sintetizadores, bateria e xilofones. Foram adicionadas e removidas

camadas, sempre utilizando uma base de *loop*, dando continuidade à ideia de composição não linear. Para este videojogo apresentei seis propostas, que essencialmente seriam trechos de 20 a 30 segundos de duração, adotando características distintas para cada um individualmente.

Uma vez concluído, o projeto acabou por ser direcionado num rumo diferente, sendo alterado para uma música mais simples e de carácter minimalista. A composição utilizada no projeto Golf Stars foi criada de forma não linear, na tonalidade de Ré maior, consistindo num *loop* de 44 segundos. Para compor a música utilizei instrumentos digitais que remetem a uma atividade de relaxamento: kalimba, guitarra acústica, pau de chuva, piano e um sintetizador suave com sustenção de notas. O ritmo da música é de carácter simples e o instrumento com maior componente rítmico é a guitarra acústica. A kalimba foi o instrumento responsável pela melodia devido à sua propriedade acústica suave e ressonante. Utilizei imensa reverberação nos instrumentos com o objetivo de criar uma sensação de leveza. À diferença da primeira opção, esta composição tem como foco o aspecto de relaxamento e a simplicidade.

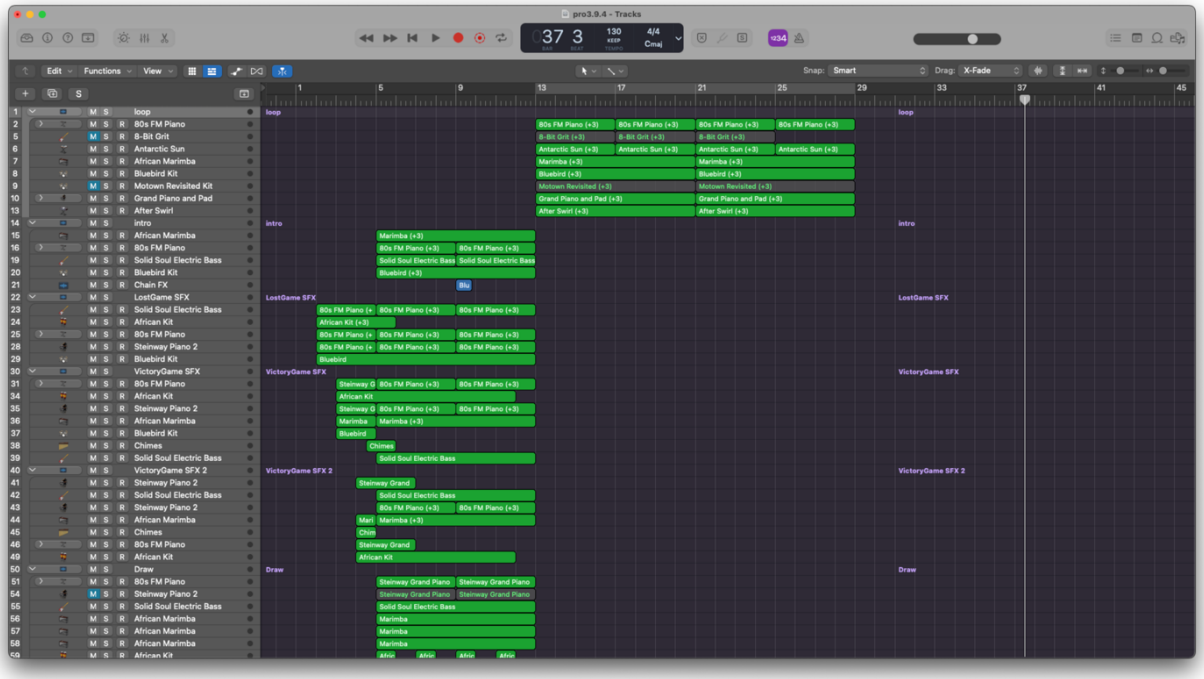


Figura 3 – Projeto Logic Pro X do *Golf Stars*.

A composição criada para o menu do jogo *Match Hotel 3D* é um exemplo de composição com características lineares e não lineares. Para este projeto foi me pedido um tema musical para menu e para *gameplay,* para o qual realizei quatro propostas. As composições consistiam em *loops* de uma harmonia que progredia com camadas suplementares, mantendo o mesmo tema musical de início ao fim.

Para compor a música para o menu do jogo *Hotel Match 3D*, utilizei instrumentos digitais do Logic Pro X. O instrumental era constituído por guitarra, baixo, sintetizador, kick, hi-hat e

vibrafone. A música contém, ritmo de carácter simples e a tonalidade é sempre Fá maior com variações de primeiro (I) para quinto grau (V) (Figura 4).

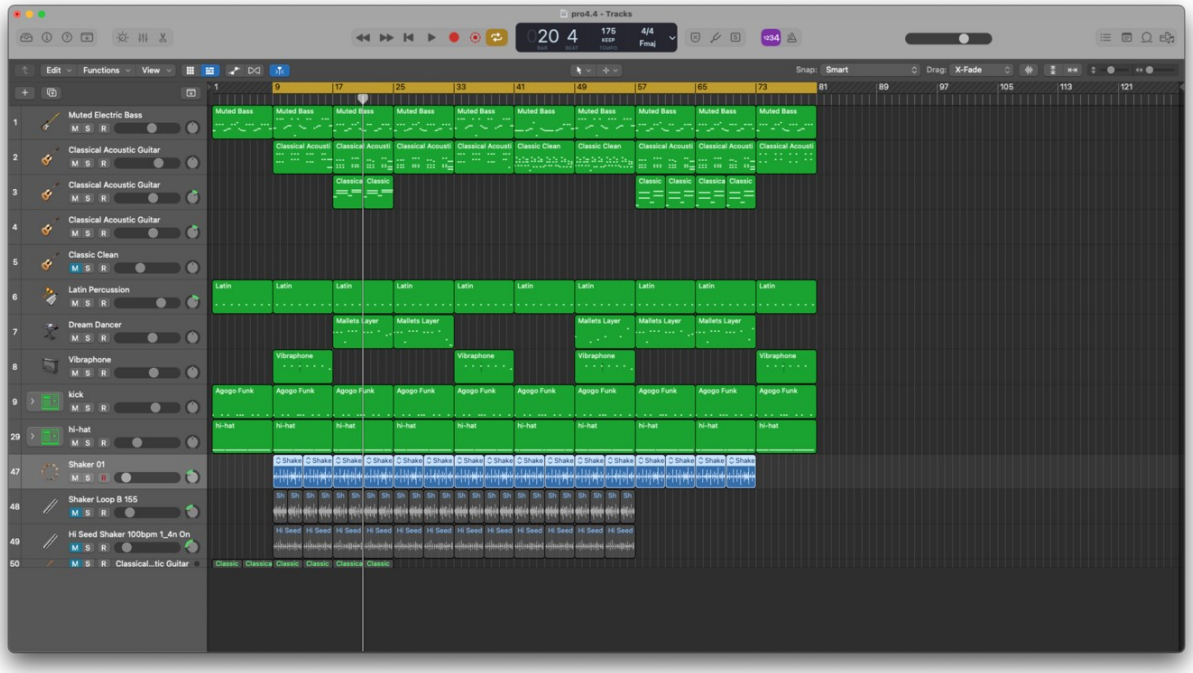


Figura 4 – Projeto Logic Pro X do *Hotel Match 3D*.

Os videojogos produzidos pela Volt são principalmente do estilo casual e hiper casual pelo que as músicas utilizadas nesses jogos são simples, minimalistas, genéricas e felizes. Em geral, para adequar a música a esse estilo, compus principalmente em escalas maiores e sem variações complexas.

Como designer de som foi-me requerido várias vezes "música genérica". Os exemplos propostos pela equipa de design, aproximavam-se do estilo *Lo-Fi* (estilo de música que utiliza sons simples, pouco complexos, suaves e de baixa qualidade). O meu processo para a criação de este tipo de música foi de minimizar todos elementos. Como exemplo, para o projeto "genérico 8" utilizei apenas cinco instrumentos, um sintentizador responsável pela harmonia em Sol maior, um ritmo simples de bateria digital, clavas para marcar o tempo, baixo a suportar a harmonia e por fim uma guitarra responsável pela melodia quase imperceptível. Apresentei várias propostas como música genérica sendo que a aprendizagem que retirei do processo é que quanto menos complexo e saturado, mais facilmente se adapta a música, a qualquer tipo de videojogo casual e hiper casual.

3.2. Design de som adaptado para videojogos

O cinema utiliza música e efeitos sonoros para acrescentar à experiência para o público, e o mesmo é aplicado para videojogos, em que o som é fundamental para absorver os jogadores na realidade do jogo (Horowitz & Looney, 2014).

Existem várias abordagens que um designer pode ter face ao áudio de um videojogo. O som, pode adicionar à experiência do jogador enriquecendo a qualidade e imersão no ambiente do videojogo. Como também, pode auxiliá-lo, dispondo de informação crucial que influência o jogador a tomar decisões ao longo da sua *gameplay*, criando uma mecânica dentro do jogo, a partir do uso de audição (Ng & Nesbitt, 2013). Um exemplo de jogo que utiliza o áudio para auxiliar o jogador é o *Counter Strike* (Valve, 2000). Este simula a perceção de profundidade e direcionamento dos efeitos sonoros para que se ouça sons de disparos com diferentes distâncias e direções. Esta mecânica é fundamental neste tipo de videojogos, pois facilita a localização de diferentes jogadores inimigos, sem ter total dependência da componente visual. A música, é também utilizada como ferramenta para auxiliar o jogador (Crathorne, 2010). Ainda no videojogo *Counter Strike* (Valve, 2000), se a bomba plantada pela equipa inimiga estiver perto de explodir, é audível uma música que assinala os últimos 10 segundos, indicando ao jogador a agir o mais rapidamente possível.

O design sonoro para os projetos da Volt Games era principalmente de carácter qualitativo, devido ao estilo de videojogos desenvolvidos na empresa. Uma outra abordagem possível, é de design informativo, em que o som tem uma função de auxiliar o jogador a realizar tarefas essenciais ao longo da sua jogabilidade. A adição de informação, utilizando o meio auditivo, adiciona mais à *gameplay* do videojogo.

O som para videojogos é categorizado em três partes: música, efeitos sonoros e vozes. Nos videojogos do tipo hiper casual e casual, os sons mais predominantes são os efeitos sonoros. O conceito de som diegético e som não diegético está presente não só no cinema como também nos videojogos. No contexto do estágio, um exemplo de som diegético num videojogo, é o som de impacto de uma tacada no *Golf Stars*, em que é possível observar a ação e associá-la à fonte sonora. Em contrapartida, a voz de comentário, "nice shot", é considerada como um som não diegético ou som de comentário, por não haver contexto, nem ser possível localizar a fonte sonora.

De acordo com o mencionado anteriormente, os videojogos são constituídos por elementos sonoros que enriquecem a experiência do utilizador. Os sons de primeiro plano (*foreground sounds*) e os sons de interface (*interface sounds)*, fazem parte de uma componente qualitativa para o design de som dos videojogos (Han & outros, 2022). *Foreground sounds* são sons que surgem de forma individual e direcionam a atenção do ouvinte quando ocorrem. Estes podem ser diegéticos ou não diegéticos. Por exemplo, uma explosão de um "pop" ou "woosh". *Interface sounds*, são sons que são atribuídos à interface do videojogo de forma que o utilizador tenha uma resposta auditiva enquanto utiliza os menus disponibilizados pelos programadores.

Um exemplo são os sons de *feedback* de volume ou os sons de "pop" ao abrir e fechar um menu. (Han & outros, 2022).

O som para videojogos pode ser aplicado como som dinâmico (*dynamic audio*) e som interativo (*interactive audio*). Nos jogos de plataforma móvel do tipo, casual e hiper casual , os sons são predominantemente interativos, isto é, por serem te caráter simples, em que o utilizador interage com uma ação e dessa ação resulta um som. Também poderá haver a presença de algumas camadas de som dinâmico como por exemplo, um "medidor de energia", em que o som de disparo de uma bola no nível máximo é diferente de um som de um disparo no nível médio. Som interativo é o som que resulta de uma ação realizada pelo jogador e o som dinâmico é o som que se adapta com a ação do jogador.

É relevante salientar que as equipas de design projetam os vídeos jogos para públicos-alvo, envolvendo vários fatores como por exemplo, o estilo do jogo, o estilo de arte e a *gameplay*. Os vídeosjogos hiper casual abrangem um público-alvo geral, e para que todos os utilizadores usufruam do conteúdo, a abordagem para o som costuma ser algo minimalista e satisfatória. Em outras palavras, o foco do designer de som para videojogos hiper casual*,* é de tentar segurar o utilizador ao telemóvel, criando sons satisfatórios para uma *gameplay* rápida. Não existe a preocupação de criar um tom emocional para o videojogo, sendo a única preocupação criar algo eficaz.

Existem vários tipos de jogos móveis, sendo que estes se dividem em casual, hiper casual, *mid-core* e *hardcore*. Os casual e hiper casual, são jogos em que o jogador pode jogar oportunisticamente e ou esporadicamente obtendo resultados instantaneamente. Os jogos *mid-core* e *hardcore* são jogos que em contrapartida, requerem algum tipo de habilidade e estratégia, para que o jogador possa progredir ou obter resultados. A principal diferença entres os diferentes tipos de jogos, é a necessidade de tempo investido (Zlobin, 2018).

3.2.1. Mistura

O processo de mistura é fundamental para o design de som de videojogos. No entanto, misturar para jogos de computador e consolas não é o mesmo que para as plataformas móveis. Isto devesse às colunas presentes nos telemóveis que são muito limitadas em termos de frequências e volume.

Neste contexto, foi em 2013 com o lançamento do "HTC One M7" e em 2016 com o "iPhone 7", que começaram a surgir telemóveis com sistema stereo embutido. Antes destes lançamentos, a única forma de ouvir áudio stereo seria com uso de uma coluna externa ou auriculares. Com o avanço tecnológico, os telemóveis com sistemas stereo tornaram-se mais populares e acessíveis a um público geral, mas ainda no ano de 2024 só os dipositivos topo de gama e média gama é que contêm essa funcionalidade. Estes fatores influenciam a mistura adotada pelos designers de som.

De acordo com o que menciono anteriormente, uma preocupação para os designers de som é de adaptar a mistura para estes dispositivos. As frequências mais predominantes nos telemóveis estão presentes num intervalo de frequências médias, entre 1000Hz e 4000Hz. Apesar de os telemóveis terem a capacidade de reproduzir altas frequências, a nitidez e energia é enfraquecida comparando com as frequências médias. Isto ocorre por consequência, do pequeno tamanho das colunas que ao mesmo tempo torna as baixas frequências (até 300hz) mais difíceis de reproduzir (Yang, 2019).

Relativamente à monotorização, os níveis estabelecidos na indústria para cinema é de 85db SPL enquanto que, para Blu-Ray e DVDs é de 79db SPL. Os videojogos para PC e consola geralmente seguem a norma do cinema, isto porque, os sistemas de reprodução são os mesmos. No entanto, devido a natureza do dispositivo, a mistura para telemóveis deve ser monitorizada com volumes mais baixos (Riviere, 2024, p.37).

O nível de intensidade média do som (LUFS) para a distribuição de um vídeo jogo de consola ou PC, deve de ser normalizado a -24 LUFS com 2 LUFS de tolerância. Esta tolerância de dois LUFS é necessária pela natureza não linear deste tipo de áudio. As composições não lineares têm um fluxo de intensidade variável consoante a ação. Para plataformas móveis o nível de intensidade média é por volta de -18 LUFS com 2 LUFS de tolerância (ASWG, 2013).

> *"I like to say that the **only** thing that matters is how it sounds on the device. [….] One of my favorite Duke Ellington quotes is 'You've got to write for your players', and that's true whether you're composing for Johnny Hodges or the Vienna Symphonic Library. For mobile gaming, your player is a speaker the size of your thumbnail (if you've got small hands), and so you need to design your sounds from the beginning with this frequency limitation in mind*." (Drescher 2014 p.70, como citado em Horowitz & Looney 2014).

Durante a experimentação do estágio, para evitar distorção harmónica nos telemóveis, ao misturar os sons e música, exportei o som entre -16 LUFS e -18 LUFS de intensidade. O processo decorria por verificar no meu dispositivo pessoal, se existia algum tipo de distorção ou efeitos sonoros não audíveis nos altifalantes. A minha abordagem de nível de frequências, era de manter os efeitos sonoros dentro do espectro audível dos telemóveis. Para executar isto utilizei, edição e manipulação de volumes, equalização, compressão e limitadores.

Para os projetos da Volt, os efeitos de som eram maioritariamente enviados em mono, apenas músicas e ambiências eram exportadas em stereo. Ao realizar a exportação dos efeitos sonoros em mono, tinha como objetivo, preservar a informação auditiva mais relevante para o videojogo. Com isto, os sons mais relevantes seriam audíveis em todos os dispositivos.

3.2.2. Exemplos de design sonoro

Ao longo do estágio tive a oportunidade de participar em mais de quinze projetos produzidos pela Volt Games. Como resultado, em seguida demonstro alguns projetos que podem ilustrar um pouco dos conceitos previamente mencionados.

No contexto do meu estágio, os efeitos sonoros foram responsáveis por construir a atmosfera dos videojogos, tornando a experiência imersiva, satisfatória e mantendo-a o mais minimalista possível. Esta abordagem designa-se de design sonoro qualitativo e é utilizada para conectar o jogador ao videojogo, tornando-o imerso na realidade virtual.

Um exemplo prático desta abordagem aplica-se no videojogo *Shoot Em Up* (Figura 5). Para este videojogo criei efeitos sonoros de impacto utilizando *sampling,* edição, manipulação com uso de envelope ADSR e aplicando efeitos de reverberação. Ao utilizar estas ferramentas, criei sons de impacto que adicionavam à qualidade do jogo, tornando-o estimulante e apelativo para o ouvinte. Este é um exemplo específico, para o qual necessitei de criar vários efeitos sonoros qualitativos. A primeira ação que realizei neste contexto foi utilizar sons de xilofone com processamento de reverberação e equalização para criar efeitos sonoros de interface minimalista. Subsequente, criei o efeito sonoro de sobreposição de fichas a partir de edição, *sampling,* equalização, reverberação e manipulação de envolpes ADSR. Por último, utilizei sons de finalização de sobreposição que consistiam num glockenspiel com reverberação e equalização.

Figura 5 – Jogo *Shoot Em Up*

O projeto *Hex Em All*, é um videojogo do tipo hiper casual e puzzle, que consiste na sobreposição de hexágonos de diferentes cores (Figura 6). Como designer de som, fui responsável pela criação de sons de primeiro plano e sons de interface. O som é de carácter interativo com alguns elementos dinâmicos, e a abordagem é de design sonoro qualitativo. O

som foi inicialmente dividido por quatro ações principais: o selecionamento de hexágonos, a sobreposição de hexágonos, super poderes e o explodir dos hexágonos.

Para criar o som da ação de selecionamento do hexágono, adotei um design tonal. Esta abordagem teve inspiração em pequena escala, do design sonoro dos videojogos da Nintendo. A Nintendo é reconhecida por desenvolver videojogos em que é prestada uma atenção especial ao detalhe do som sendo que, uma enorme coleção dos videojogos produzidos por esta empresa têm um carácter de design tonal, em que os efeitos sonoros possuem características musicais, consistindo em curtas frases melódicas ou rítmicas.

De forma a alcançar esse tipo de resultado, defeni uma sequência de intervalos melódicos que reproduzi ao utilizar sintetizadores com ataques suaves e reverberação. O efeito sonoro da sobreposição e da explosão consistiu na edição de samples de "pops" e de "suction pops" nos quais foram filtradas frequências com o uso de equalização e por fim, adicionada reverberação. Finalmente, o efeito sonoro do super poder consistiu numa escala crescente tocada em xilofone, pelo que adicionei um sample para a sensação de rotação.

Colocando em consideração que, este videojogo não utilizava música de fundo. Foi introduzida uma abordagem de efeitos sonoros tonais, para conseguir preencher o vazio que estava presente ao longo da *gameplay*. Mais tarde foram integrados novos elementos como, explosão de flores, buracos negros e outros elementos, que criei principalmente com uso de *sampling*.

Figura 6 – Jogo *Hex Em All*

Apesar de em pequena escala, o projeto *Hotel Match 3D* teve uma abordagem de design qualitativo e informativo. O videojogo tem como finalidade, a organização de vários objetos que se encontram amontados no ecrã, sendo que, apenas alguns objetos são necessários para finalizar a ronda (Figura 7). O som está dividido por ações: o selecionar do objeto, o agrupamento e os super poderes. Para criar o efeito sonoro de agrupamento, utilizei sobreposição de camadas de samples de "pops", com acordes perfeitos fabricados num

sintetizador. Para os objetos obrigatórios do nível, adicionei uma camada, com um sample de impacto de moedas reverberadas. O efeito sonoro de selecionamento de objeto consistia em "pops" que eram randomizados no Unity. Os super poderes foram criados a partir de sampling, edição e processamento de som.

Ao acrescentar uma camada de *sweetner* ao efeito sonoro de agrupamento, utilizo não só uma abordagem informativa, por indicar ao jogador que está a seguir o objetivo, como também através do uso de um som que remete a ganhar algo, estimulo o jogador de forma inconsciente à sensação de recompensa que por consequência, fixa o jogador ao videojogo.

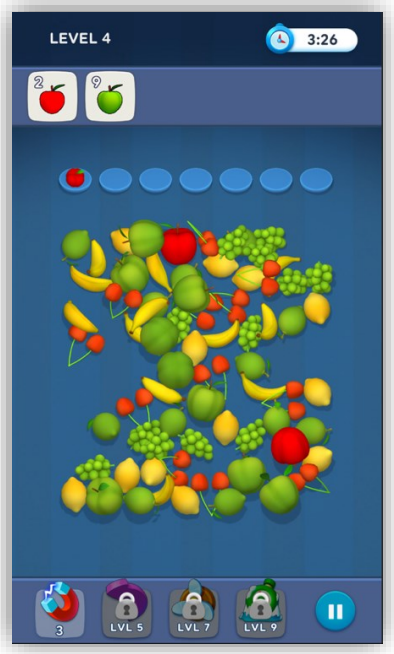


Figura 7 – Jogo *Hotel Match 3D*

O projeto *Golf Stars* teve uma abordagem de design qualitativo para o qual desenvolvi efeitos de interface e de primeiro plano (Figura 8). O videojogo utilizava sons, maioritariamente, diegéticos com exceção das vozes de comentário. O design sonoro teve uma aplicação interativa e dinâmica em que, para certas ações, o som adaptava-se de acordo com a decisão do jogador. Este projeto é categorizado por várias partes: como primeira ação o som de fundo consistia principalmente em samples de ambiência. Para cada piso era utilizado diferentes sons de relva ("fairway", "rough", "water hazard", "green" e "sand"). Os sons de impacto do taco variavam de acordo com a força exercida pelo jogador e por último, o jogo continha voz de comentário como "nice shot".

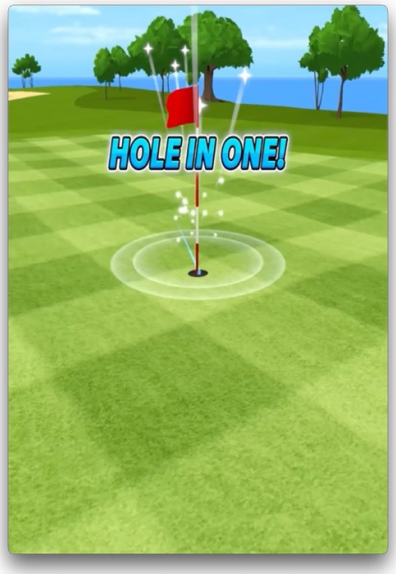


Figura 8 – Jogo *Golf Stars*

De acordo com o mencionado previamente, a implementação de uma lista de sons associados à força de tacada, representa uma utilização simples e eficaz de som dinâmico. Como designer de som fui responsável também por gravar vozes para os comentários do videojogo. Considero relevante mencionar que, para todos os projetos desenvolvidos na Volt Games utilizei *samples* de bibliotecas de som. Alguns exemplos de bibliotecas: Epidemic Sound, Freesound, Pixabay, Artlist.io, BBC sound effects, Macaulay Library e Soundly. As bibliotecas pagas, às quais não tive acesso total, foram uma referência para a criação de efeitos sonoros.

Ao longo desta experiência verifiquei que uma parte significativa do meu trabalho como designer sonoro foi a de elaborar sons aprazíveis ao ouvinte. Estes tipos de decisões de design são extremamente subjetivos, mas tive em consideração vários exemplos de outros projetos publicados, como referências. A minha abordagem foi minimalista visto que, utilizei como referência, um videojogo em que a simplicidade sonora é um componente essencial para a satisfação auditiva. No *Stardew Valley* (Barone, 2016), os sons para a ação de colheita são unicamente "pops" aos quais o designer sonoro, aplicou também variações aleatórias de *pitch* e reverberação, tornando o som extremamente aditivo. Os sons são extremamente simples, mas muito efetivos, consistindo apenas em um ou dois elementos. No contexto do estágio, uma grande parte dos projetos utilizaram sons de "pops" que gravei com articulações vocais e processei com variações de *pitch*. Ao utilizar sons que remetem ao ser humano, inconscientemente direciono o ouvinte a algo que lhe é familiar, criando uma sensação de conexão e conforto (Louwers & outros, 2022).

Apliquei a mesma técnica com uso de sons de moedas que embora não diretamente relacionadas com a ação, remetem o jogador a uma sensação de conquista e realização afixando-o ao videojogo (Louwers & outros, 2022).

3.3 Implementação por Unity e FMOD

Para que todo o trabalho sonoro seja representado num videojogo, é necessária a implementação do som a partir de software. Neste capítulo são mencionados os aspectos técnicos para este processo de forma a que, seja possível compreender como utilizar os programas necessários para implementação de som. Por fim, exemplifico a minha experiência com o uso de este tipo de software.

Antes de iniciar com a implementação de som, é relevante mencionar como é que a tecnologia se desenvolveu nos videojogos. O som para videojogos desenvolveu-se significativamente desde o primeiro surgimento em 1972 com o jogo *Pong* (Atari, 1972) (Paduano, n.d.). Com o avanço da tecnologia, surgiram chips especializados para processamento do som, sendo que o *Space Invaders* (Taito Corporation, 1978), foi o primeiro videojogo a adotar este tipo de chips. Em 1985 a Nintendo lançou a consola NES (*Nintendo Entertainment System*), um sistema já com cinco canais de áudio (três canais para síntese, um para ruído e um para uma samples). Sete anos mais tarde, com o lançamento da Sega CD System, o CD-ROM tornou-se o padrão e oferecendo 18 canais de áudio a um sample rate de 44100Hz. Em 1995 a Sony lançou o sistema Playstation com um chip de som de 24 canais e suporte integrado na consola para efeitos como reverb e delay. Dez anos mais tarde em 2005 a Microsoft lançou a Xbox 360 que já fazia o processamento do áudio a 32 bit de resolução e suportava som em surround, multicanal a 16 bit de resolução. (Horowitz, & Looney, 2014).

Atualmente, a tecnologia não é uma limitação para a criatividade e qualidade do som. O padrão de resolução do som para videojogos é de 24 bits e o sample rate é de 48000Hz. Esta é uma consequência do desenvolvimento da tecnologia, e da facilidade de acesso a softwares gratuitos ou *open source*.

Existem vários motores de jogos disponíveis de acesso gratuito, como o Unity, Unreal Engine e Godot (este motor de jogo é totalmente gratuito e *open source*). Atualmente os mais utilizados são, o Unity e o Unreal Engine. O software Unity é gratuito de forma não comercial, aplicando taxas a jogos cujo lucro é superior a 200.000 dólares (Unity, 2024). O Unreal é totalmente gratuito para particulares ou empresas que faturem menos de 1.000.000 dólares (Unreal, 2024).

O Unreal Engine, é um software de desenvolvimento de videojogos, criado pela empresa norte americana Epic Games. Este software é reconhecido pelas capacidades avançadas de renderizar gráficos, dispondo de ferramentas de edição de vídeo e som e também de ferramentas de renderização de animações.

O Unity é o software multiplataforma desenvolvido pela Unity Technologies. É caracterizado por ser de fácil aprendizagem e foi lançado inicialmente com a ideia de democratizar a indústria. Atualmente é o motor de jogos que domina a plataforma de videojogos móveis, e é também muito utilizado no mercado de videojogos indie para consolas e PC. Isto deve-se a uma interface intuitiva, que facilita a utilização do software, fornecendo documentação e tutoriais online.

Disponibilizam ferramentas de desenvolvimento de videojogos para multiplataformas, para que os programadores possam exportar os jogos sem limitações e também facultam o acesso à Unity Asset Store, que contem vários *assets* grátis e pagos, que facilitam e aumentam a produtividade de trabalho.

3.3.1. Motor de áudio

A implementação de som para videojogos é gerida por um *middleware*, e embora várias empresas como a Blizzard, Nintendo, Rockstar Games e Valve utilizem o seu próprio motor de som, outras empresas utilizam softwares como o FMOD e o Wwise.

O Wwise ou Wave Works Interactive Sound Engine, é um motor de áudio desenvolvido pela Audio Kinect no Canada. Este software é utilizado para implementação e gestão de som, em sistemas como Unity e Unreal engine. É considerado por vários profissionais na área, o melhor motor de som no mercado, devido à sua robustez e ferramentas de espacialidade de áudio (*Wwise Spatial Audio*). Este software não é gratuito, e por isso é maioritariamente utilizado em projetos com um orçamento alto.

No contexto das minhas práticas, a Volt Games implementava o som para os videojogos por Unity e por isso a investigação é baseada nos dois métodos possíveis de implementar som utilizando este software. Uma processo diretamente pelo Unity e outro método utilizando o software FMOD. Além disso, decidi investigar e utilizar o FMOD e Unity por este software ser gratuito e devido à formação adquirida no mestrado da Universidade Católica Portuguesa do Porto.

O FMOD é um software de motor de áudio desenvolvido pela Firelight Technologies na Austrália. Foi desenvolvido e escrito em "C++ portable", que proporciona compatibilidade com vários sistemas operativos diferentes. Este software é utilizado para facilitar a manipulação e integração de áudio nos videojogos, assim como auxiliar o trabalho dos designers de som e programadores. O FMOD é o software de gestão de som mais utilizado para projetos indie e de plataforma móvel, por ser parcialmente gratuito. O Unity, na sua essência, utiliza um motor de áudio que consiste numa versão antiga do FMOD, designada por FMOD Ex API.

Para utilizar FMOD é necessário ter alguns conhecimentos fundamentais em Unity, por isso menciono abaixo algumas terminologias fundamentais para o uso de áudio neste software:

- *Audio Listener*, é um componente integrado no Unity que é utilizado e aplicado para que o programa decifre qual é o ouvinte do videojogo. Pode ser aplicado na primeira pessoa ou na terceira pessoa sendo que, para jogos 2D é usualmente colocado na câmera do videojogo (Horowitz & Looney, 2014).

- *Audio Source*, é um componente do Unity que é aplicado, como uma fonte sonora. As propriedades do mesmo, adaptam-se para o tipo de jogo (2D ou 3D). Podem ser adicionados

vários *Audio Source* por projeto, sendo possível ouvir até trinta e dois em simultâneo (Horowitz, & Looney, 2014), (Tecnologies U. n.d.).

- *Audio Clip*, é um *asset* que representa o clip de áudio presente no banco de sons do *Audio Source* (Horowitz, & Looney, 2014).

- *Audio Mixer*, é um componente utilizado para gerir volumes e aplicar efeitos de som (Tecnologies, U. n.d.)

- *Game Object*, são considerados como "contentores", onde são colocados os componentes, scripts, entre outros elementos do Unity.

- Scripts, podem ser adicionados como componentes no Unity e são essenciais para execução de ações no programa. Essencialmente são sequências de instruções que estão escritas em C# às quais o programa decifra e interpreta (Brave Software, 2024).

Para a implementação de som por Unity utilizei principalmente os *Audio Listener* e *Audio Source* que são os elementos principais responsáveis pela saída do som. Para os projetos da Volt Games eram utilizados *Audio Sources* com saída de áudio 2D porque para os jogos casuais e hiper casuais não exista uma prespetiva de áudio espacial (áudio 3D). Os scripts não só são responsaveis por criar condições de saída de informação como também são responsáveis por manipular som, aplicar efeitos sonoros, criar automações entre outras técnicas de edição de som.

3.3.2. Iniciar som por Unity

Para despoletar o som no Unity, é necessária uma condição por script. Os scripts customizáveis têm como função, criar uma linha de comunicação entre os componentes do Unity, criando as condições necessárias para que tudo funcione em harmonia. É necessário ter alguns conhecimentos de linguagem C# para criar condições via script. Como utilizar um script:

- Criar um *Game Object*.

- Dentro do *Game Object* adicionar um *Audio Source* e um script.

- Adicionar a variável de *Audio Source* dentro do script.

No Apêndice B é representado de forma simplificada, como organizar um script dentro do Unity. Este script é responsável pela criação de uma condição para o playback de áudio. Também é demonstrado a possibilidade de tornar certos elementos do script visíveis, a fim de poder alterar valores, na interface do software (Apêndice B).

3.3.3. Integrar o FMOD no Unity

O FMOD Studio não vem por padrão integrado no sistema do Unity. Segundo Abbott (2018), de forma interligar os softwares é necessário:

- Fazer download do FMOD Studio no site oficial.
- Também fazer download do pacote de integração que vem em formato de *Unitypackage*.
- Importar o *Unitypackage* para o Unity ou utilizar a *Asset store* para o implementar.
- Seguir os passos dos FMOD *Setup Wizard.*
- E por fim, associar ao Unity o caminho da pasta do projeto do FMOD.

3.3.4. Utilização de FMOD

O FMOD tem como função a integração de som para aplicações e videojogos. Este software funciona como um DAW que disponibiliza várias ferramentas de manipulação, edição e processamento de som (Figura 9). De acordo com o autor Abbott (2018), existem alguns componentes fundamentais para a utilização de FMOD que são:

- *Bank*, é um ficheiro criado pelo FMOD, que contém os eventos do projeto comprimidos e formatados. Este depois é lido pelo motor de jogo para interpretar o que existe no projeto FMOD a nível sonoro.
- *Events* (eventos), são os pacotes que contêm o som do projeto.
- *Timeline*, é a interface visual utilizada para editar os clips de áudio do projeto.
- *Mixer*, interface visual onde é feita a edição de volumes e panorâmica e também onde são aplicados os efeitos de processamento.
- *Assets*, é um *browser* que dispõem das pastas e dos ficheiros de áudio utilizados no projeto.
- *Audio Bin*, é um *browser* que disponibiliza o acesso às pastas e ficheiros utilizados no projeto. Também possibilita o acesso a bibliotecas de som pagas.
- *Parameters*, são variáveis que podem influenciar o comportamento dos *Events* e do som dinamicamente.

- *Snapshots*, são ferramentas utilizadas para alterar algumas propriedades do som de forma temporal ou condicional, sem alterar a estrutura dos eventos de som na T*imeline*.

- *Effects*, são efeitos sonoros que estão presentes no software. Podem ser aplicados aos clips de áudio e ao Master.

- Automação, é uma ferramenta do FMOD utilizada para criar *adaptive audio* ou *adaptive music*, por exemplo, ao utilizarmos esta funcionalidade em conjunto com os *Parameters*, é possível criar variáveis que controlam efeitos automatizados pelo sound designer.

- *Instrument* (instrumentos), são regiões de ativação de som, ou clips de áudio. Existem vários tipos de instrumentos, single instrument, multi instrument, event instrument, scatterer instrument, programmer instrument, plug-in instrument e snapshot instrument.

- *Live Update*, é uma funcionalidade do FMOD que proporciona ao utilizador a habilidade de poder alterar propriedades do som em tempo real, ou em simultâneo com o Unity.

Alguns dos elementos mais importantes que utilizei na minha experimentação com FMOD foram a automação, *Live Update*, *Mixer e Timeline.* A ferramenta de *Live Update* foi essencial para experimentar o som do projeto em tempo real enquanto que a *Mixer e Timeline* foram as ferramentas visuais que utilizei para editar áudio, aplicar *loops* e regular volumes. Por fim, a ferramenta de automação é a meu ver essencial para criar variações temporais no som.

Figura 9 – Interface do FMOD

De acordo com o ritmo de trabalho e com os tipos de videojogos produzidos ao longo do estágio, a Volt Games procurou integrar o som a partir de scripts e de *prefabs*, directamente pelo Unity software. Como resultado, senti algumas limitações a nível de edição e mistura de som. Embora não tenha sido utilizado o FMOD no contexto prático de estágio, é igualmente importante compreender o processo de implementação e manipulação de som por FMOD de forma a poder retratar como designer a ideia sonora no videojogo. Em continuação menciono um exemplo de implementação diretamente por Unity, como realizado no estágio, e em seguida um exemplo de a minha experimentação e tentativa com o uso de FMOD.

3.3.5. Exemplo prático Unity

Como mencionado anteriormente, ao longo do estágio a implementação de som foi realizada pelos programadores diretamente no Unity. A função que desenvolvia neste processo consistia em seguir os requisitos sonoros, criar e editar os sons, e quando finalizados exportá-los para os programadores os implementarem no software. Devido ao meu interesse pela implementação, segui acompanhamento dos colegas da Volt Games, que me ensinaram as noções básicas do Unity. Desta forma fiquei responsável por adicionar elementos e trabalhar com o gestor de áudio *prefab* da empresa.

No videojogo Hotel Match 3D, de forma a auxiliar os programadores no processo de implementação, primeiro exportava os efeitos sonoros para uma pasta de *assets* no Unity. A pasta estava organizada em categorias como sons de *gameplay*, sons de construção, entre outros. Posteriormente, para adicionar novos sons abria um *asset* designado de "audio_data" onde verificava vários elementos na interface de utilizador. Cada elemento estava atribuído a uma ação e o respetivo clip de áudio (Apêndice C). O processo consistia em criar um elemento de "Audio Entries" na interface, abrir o script e iniciar o mesmo atribuindo-lhe um número de ordem. Por fim, introduzia o clip de áudio no novo elemento visível na interface (Figura 10).

Devido ao processo de implementação ser predominante em script, surgiram várias limitações a nível criativo. Por exemplo no vídeo jogo Hotel Match 3D, o uso de automações de panorâmica ou volume encontravam-se foram do alcance do designer, e mesmo ao comunicar as ideias aos programadores, o som resultava em som cru. Por isto, previamente à exportação os sons eram editados de acordo com as limitações presentes no momento de implementação: alguns sons eram editados com *fade in* e *fade out*, outros continham reverberação, delay e outros tipos de processamento prévio à implementação.

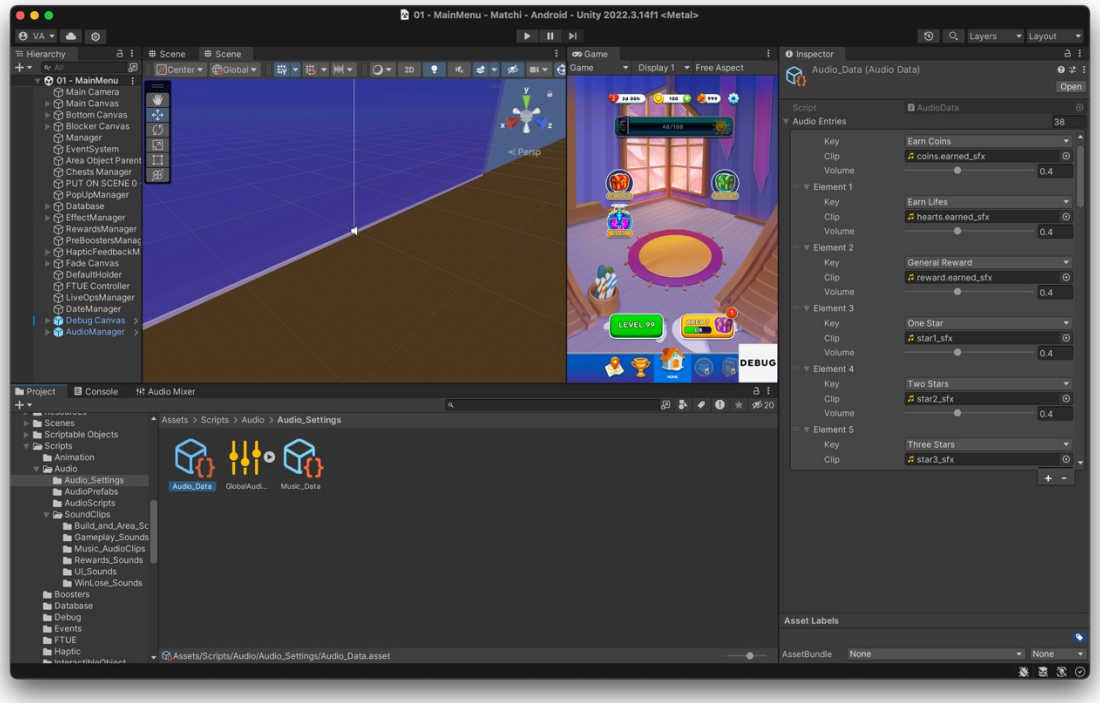


Figura 10 – Interface do Unity com elementos do “audio_data”.

3.3.6. Exemplo prático FMOD

Para pôr em prática os meus conhecimentos de FMOD decidi integrar este motor de áudio no projeto *Golf Stars* da Volt Games. Para esta experiência, o meu objetivo foi de colocar uma introdução e um *loop* no menu do videojogo. Comecei por criar um ficheiro novo no FMOD Studio, coloquei os clips de áudio no audio bin e arrastei os ficheiros de áudio para um evento. Introduzi uma região de *loop* no clip, e atribui o evento a um novo *Bank*. Este processo é importante porque, os *Banks* são o elemento responsável por comunicar com o Unity. Por fim, dei “build” do projeto e dei refresh no projeto do Unity.

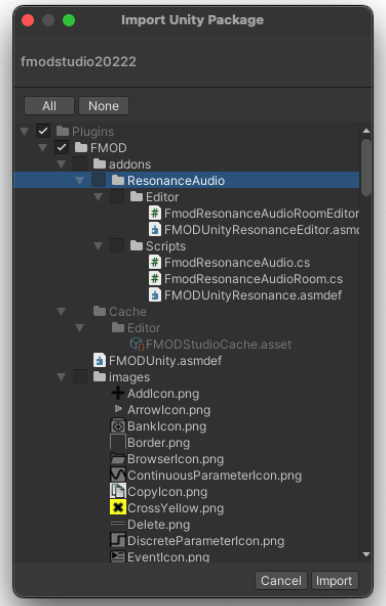


Figura 11 – Unity Package

Ao abrir o projeto no Unity verifiquei que, por não ter sido utilizado FMOD anteriormente, teria de carregar o *costum package* (Figura 11) e seguir os passos do *setup wizard*. Já com o *middleware* integrado, escolhi um caminho no computador para que o Unity acedesse aos *banks* do FMOD.

Para evitar conflitos de software, desativei o motor de áudio do Unity, e apaguei o componente *Audio Listener* da câmara de menu. Para substituir o *Audio Listener* introduzi o componente "FMOD Studio Listener" (Apêndice D).

Para iniciar o som, introduzi um "FMOD Studio Event Emitter" (componente responsável por iniciar e controlar os eventos), no objeto da câmara do jogo (Apêndice E).

Ao abrir o script, introduzi uma variável para o FMOD. Por fim, segui o sistema de instruções utilizado pela Volt Games e introduzi a variável do FMOD como *bgmusicaudio.Play()* e *bgmusicaudio.Stop()* respetivamente (Apêndice F).

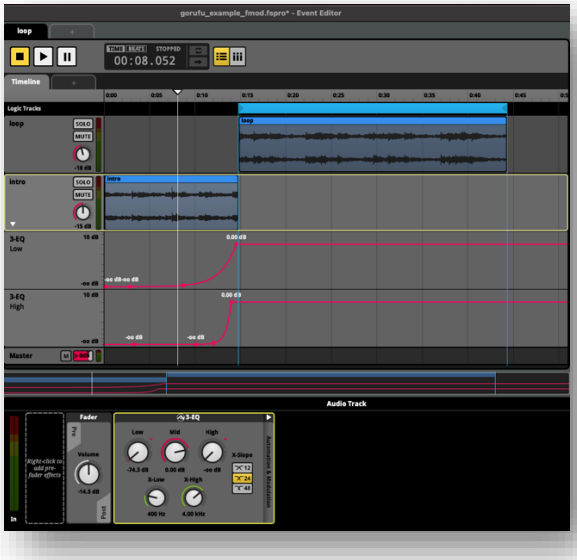


Figura 12 – Utilização de automação

Na janela do FMOD, adicionei um AHDSR como *fade in e fade out* no Master. Introduzi um equalizador no instrumento "intro" e utilizei automação de forma a alterar as propriedades do equalizador (Figura 12). A funcionalidade *Live Update* foi-me útil para alteração de propriedades do som em tempo real.

O FMOD permitiu-me aplicar efeitos e editar o som em tempo real, aumentando a produtividade e facilitando no processo criativo de implementação do som. A possibilidade de editar som, utilizar automações e o uso de efeitos resolvem situações como a re exportação de ficheiros do DAW por necessidade de alterar elementos.

3.3.7. Comparação dos softwares

Comparando a experiência de implementação por Unity e por FMOD concluo que, ambos os softwares oferecem as ferramentas necessárias para implementação de som num videojogo. O Unity oferece várias opções de implementação por uso de scripts, no entanto o FMOD oferece uma interface intuitiva, e componentes que contêm funções mais robustas e flexíveis. Como por exemplo "event editor", que permite editar e manipular áudio *assets* depois de exportados para o Unity. A colocação aleatória de uma lista de sons pode ser obtida por FMOD sem necessidade de programação, que ao contrário do Unity, é necessário que seja via script. Também é possível sequenciar música e efeitos sonoros, com o uso de ferramentas na *Timeline*, enquanto que no Unity isto é alcançado com o uso de vários *Audio Sources* e programação por script.

Um exemplo da diferença de qualidade sonora com o uso deste software é evidente no projeto Hex Em All. Neste projeto os programadores estiveram encarregues de realizar a implementação diretamente no Unity. O processo de lhes transmitir a ideia sonora para

colocarem os efeitos corretamente foi difícil e como consequência, isto fez com que certos sons se repetissem quando não era suposto e distorcessem. Tudo isto dificultou no processo criativo.

Se para o mesmo projeto fosse utilizado o FMOD, a disponibilidade de várias ferramentas integradas neste software tornariam possível, uma experimentação dos efeitos sonoros em tempo real de forma a perceber se eram necessárias alterações ou adaptações. A possibilidade de integrar reverberação e equalização como *plug-in* no FMOD, permitiria gerir certos parâmetros diretamente no projeto, criar elementos dinâmicos e ao mesmo tempo poupava armazenamento ao utilizar clips de áudio mais curtos. Por fim, ferramentas como automação e envelopes de AHDSR, seriam úteis para manipular o áudio tornando o jogo mais dinâmico.

4. Conclusões

Esta experiência foi enriquecedora a nível profissional e pessoal. Tive a oportunidade de trabalhar num espaço interdisciplinario, estar em contacto com profissionais da área e ter a responsabilidade total pelo departamento de som. Ao longo de seis meses, aprofundei os meus conhecimentos em âmbitos de composição musical, design de som e implementação de som para videojogos.

No decorrer do estágio coloquei em prática os meus conhecimentos de composição adaptando-os ao meio dos videojogos. Com os projetos realizados para a Volt Games adquiri novas perspetivas para compor, aplicando formas de composição linear e não linear.

Apliquei os conhecimentos de design sonoro adquiridos ao longo do mestrado como técnicas de edição, manipulação e mistura de áudio. Adquiri novos conhecimentos face ao design para videojogos como abordagens de design qualitativo e informativo, enriquecendo os videojogos, tornando-os mais imersivos e adicionando componentes às suas mecânicas fundamentais. Outras noções como, som interativo e som dinâmico, foram introduzidas de forma prática de experimentação no contexto dos videojogos em desenvolvimento.

Durante os seis meses, ultrapassei desafios relacionados com a simplificação de design e composição musical. Nas reuniões de design, fui designado para criar som e música de carácter simples e genérico. No início do estágio, existiram dificuldades em respeitar esses objectivos. Inicialmente compus de forma livre, mas ao verificar que a música não se adaptava à estética dos videojogos, alterei e simplifiquei as composições criando um carácter minimalista.

Numa procura de referências para projetos da Volt Games deparei-me com um vídeo designado de *Why Indie Games Always Seem To Have The Best Sound Design* (Mcgee, 2024). Este vídeo consiste numa análise a vários jogos em que, o criador retrata os problemas de design sonoro por profissionais da área. Uma abordagem do vídeo que considerei interessante foi a de criticar os sons que são complexificados sem necessidade. Ao efetuar uma análise introspetiva do meu trabalho e relacionando-o com o material da Volt, penso ter conseguido adaptar o meu design de som para diferentes níveis de complexidade, criando espaço para pequenos e simples sons.

Por fim, este estágio criou a oportunidade de avançar com a pesquisa no tema de implementação e uso de FMOD e Unity. Acompanhei o processo de implementação das minhas composições e analisei os benefícios de utilizar estes softwares para design sonoro.

O processo de implementação de som na Volt Games passava diretamente pela equipa de programação. Infelizmente, por não utilizarem FMOD, senti algumas limitações a nível de flexibilidade criativa e de independência na integração dos sons. Por seguirmos um cronograma de trabalho apertado, não foi possível um processo de experimentação com o software. Algumas ferramentas essenciais para a projeção e integração de som num videojogo estavam limitadas pelo uso de scripts no Unity.

Este estágio disponibilizou diferentes vertentes na área do design para videojogos, e criou a possibilidade de adquirir experiência neste mercado de trabalho competitivo. Tive a oportunidade de criar conexões nesta área e de perceber quais são os meus principais interesses.

Para um futuro próximo, estou muito interessado em criar o meu próprio videojogo desenvolvendo-o com a minha visão criativa. O meu objetivo é de expandir os meus conhecimentos sobre programação para áudio, composição e design para videojogos.

Referências

Abbott, D. (2018). Using FMOD Studio with Unity.

Astle, A. (2022), *Voodoo surpasses 6 billion downloads across games and apps.* PocketGamer.biz https://www.pocketgamer.biz/voodoo-passes-six-billion-downloads/

ASWG, (2013) *Average loudness and peak levels of audio content on Sony computer entertainment platforms*. Sony Computer Entertainment.

Brave Software (2023, August 24) Script. Brave. https://brave.com/glossary/script/

Crathorne, P. J. (2010). *Video game genres and their music* (Doctoral dissertation, Stellenbosch: University of Stellenbosch).

Dubnov, S. (2021). Adaptive Music.

Engine U. (2024). Licensing. https://www.unrealengine.com/en-US/license

Han, Z., Kang, J., & Meng, Q. (2022). The effect of foreground and background of soundscape sequence on emotion in urban open spaces. *Applied Acoustics*, *199*, 109039.

Horowitz, S., & Looney, S. (2014). *The essential guide to game audio: the theory and practice of sound for games*. Routledge.

Kähärä, L. (2018). Producing adaptive music for non-linear media.

Kanaris-Sotiriou, A. (2018, Oct. 26). What the flip is Non Linear Music? Polygon Treehouse https://www.polygon-treehouse.com/blog/2018/10/24/what-the-flip-is-non-linear-music.

Kellman, N. (2020). *The Game Music Handbook: A Practical Guide to Crafting an Unforgettable Musical Soundscape*. Oxford University Press.

Louwers, G. L. M., Vieira, E. O., Van Bommel, J., & Pont, S. C. (2022). Sounds that satisfy:: Describing the relationship between sound and need fulfilment. In DRS 2022.

Munday, R. (2007). Music in video games. *Music, sound and multimedia: From the live to the virtual*, 51-67.

Ng, P., & Nesbitt, K. (2013, September). Informative sound design in video games. In *Proceedings of the 9th Australasian conference on interactive entertainment: matters of life and death* (pp. 1-9).

Nguyen, J. (2019). 3D Platformer and Sound Design in Games. *Amsterdam University of Applied Sciences*.

Paduano, P. I. Music in early videogames.

Pitkänen, P. (2013). Creating and Designing Sound Effects for a Mobile Game.

Riviere, A. (2023). *Game Audio Mixing: Insights to Improve Your Mixing Performance*. Focal Press.

Technologies, U. (n.d.) Unity – Manual: An overview of the concepts and Audio Mixer. https://docs.unity3d.com/Manual/AudioMixerOverview.html

Technologies, U. (2024). The Unity Runtime Fee. https://unity.com/pt/products/pricing-updates

Villberg, A. (2022). *Composing nonlinear music for video games* (Master's thesis).

Yang, J. (2019) Loudness and frequency response on popular smartphones. AudioKinetic. https://blog.audiokinetic.com/fr/loudness-and-frequency-response-on-popular-smart-phones/

Zizza, K. (2023). *Game Audio Fundamentals: An Introduction to the Theory, Planning, and Practice of Soundscape Creation for Games*. Focal Press.

Zlobin, D. (2018). *Audio design in mid-core mobile games* (Master's thesis).

Referencias audiovisuais

Atari (1972) *Pong* [Videojogo].

Barone, E. (2016) *Stardew Valley* (Versão 1.6) [Videojogo].

COLOPL, Inc (2022) *Neko golf* (Versão 3.0.4) [Videojogo móvel].

Dream Games (2021) *Royal Match* (Versão 22272) [Videojogo móvel].

Electronic Arts (2017) *Golf clash* (Versão 2.51.2) [Videojogo móvel].

Maddy Makes Games. (2018) *Celeste* (Versão 2021) [Videojogo].

Mcgee, M. (2024) *Why Indie Games Always Seem To Have The Best Sound Design* [Vídeo]. https://www.youtube.com/watch?v=toEdi_wjTGM&ab_channel=MarshallMcGee

Nintendo (2017) *Super Mario Odyssey* (Versão 1.0.1) [Videojogo].

Ohira, I (n.d.) GT Mode 5. On Grand Turismo Soundtrack [Banda sonora de videojogo].

Peak Games (2023) *Match Factory* (Versão 1.19.61) [Videojogo móvel].

Polyphony Digital (2005) *Grand turismo 4* [Videojogo de consola].

Taito Corporation (1978) *Space Invaders* [Videojogo].

Valve (2000) *Counterstrike* (2023) [Videojogo].

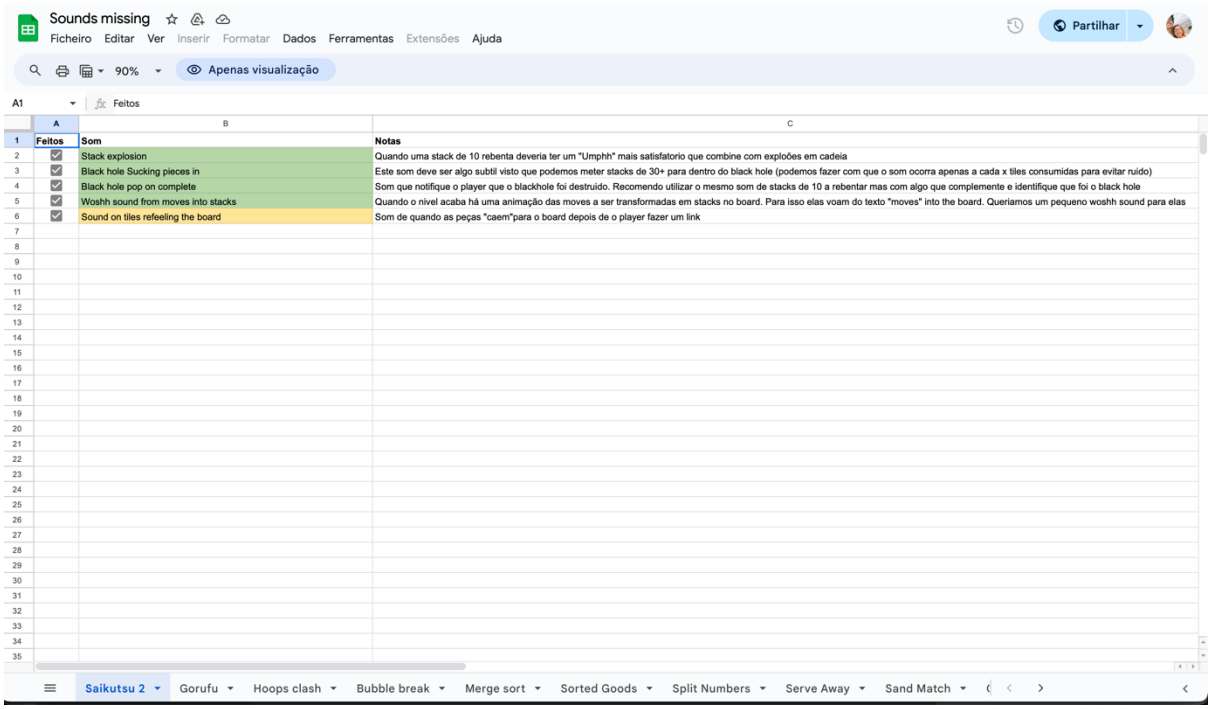
Volt Games (2024) *Golf Stars* (Versão 1.05) [Videojogo móvel].

Volt Games (2024) *Hex Em All* [Videojogo móvel].

Volt Games (2023) *Hotel Match 3D* (Versão 1.02) [Videojogo móvel].

Volt Games (2024) *Shoot Em Up* [Videojogo móvel em produção].

Apêndice A - Ficha Excel de observações para jogos *Saikutsu 2* e *Hoops Clash*.



Sounds missing

Ficheiro Editar Ver Inserir Formatar Dados Ferramentas Extensões Ajuda

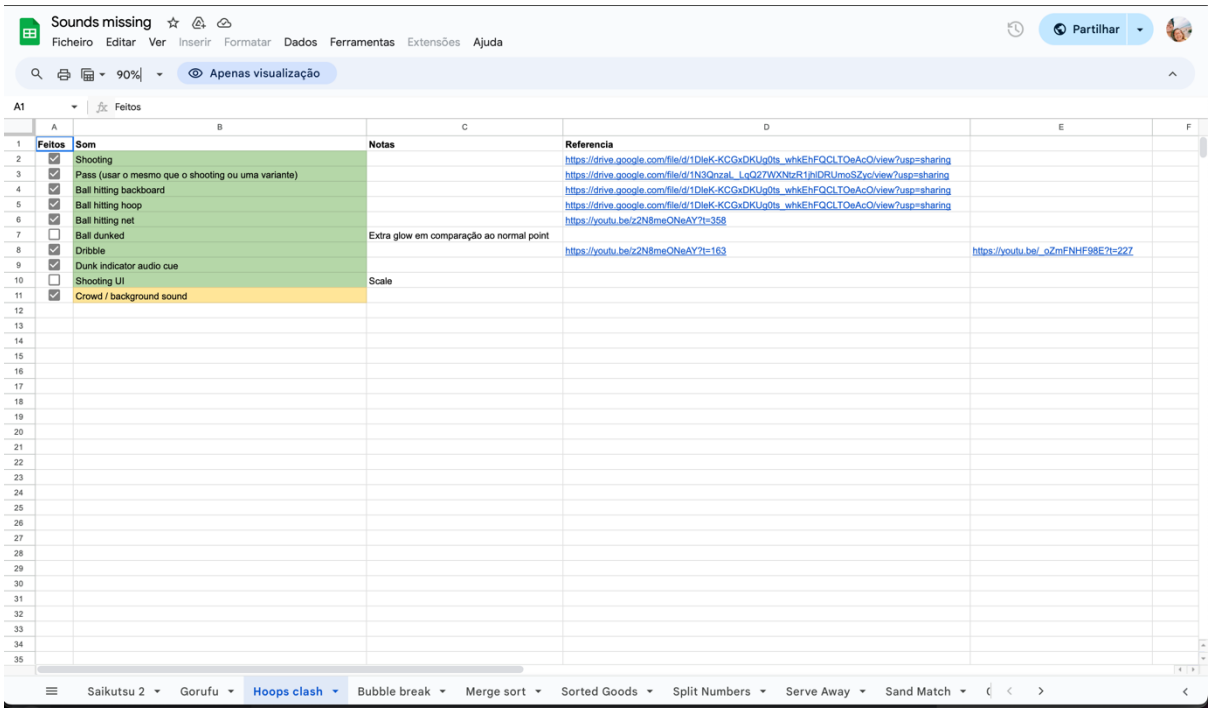
Partilhar

90% Apenas visualização

A1 Feitos

	Feitos	Som	Notas
2	☑	Stack explosion	Quando uma stack de 10 rebenta deveria ter um "Umphh" mais satisfatorio que combine com explodes em cadeia
3	☑	Black hole Sucking pieces in	Este som deve ser algo subtil visto que podemos meter stacks de 30+ para dentro do black hole (podemos fazer com que o som ocorra apenas a cada x tiles consumidas para evitar ruido)
4	☑	Black hole pop on complete	Som que notifique o player que o blackhole foi destruido. Recomendo utilizar o mesmo som de stacks de 10 a rebentar mas com algo que complemente e identifique que foi o black hole
5	☑	Woshh sound from moves into stacks	Quando o nivel acaba há uma animação das moves a ser transformadas em stacks no board. Para isso elas voam do texto "moves" into the board. Queriamos um pequeno woshh sound para elas
6	☑	Sound on tiles refeeling the board	Som de quando as peças "caem"para o board depois de o player fazer um link

Saikutsu 2 | Gorufu | Hoops clash | Bubble break | Merge sort | Sorted Goods | Split Numbers | Serve Away | Sand Match



Sounds missing

Ficheiro Editar Ver Inserir Formatar Dados Ferramentas Extensões Ajuda

Partilhar

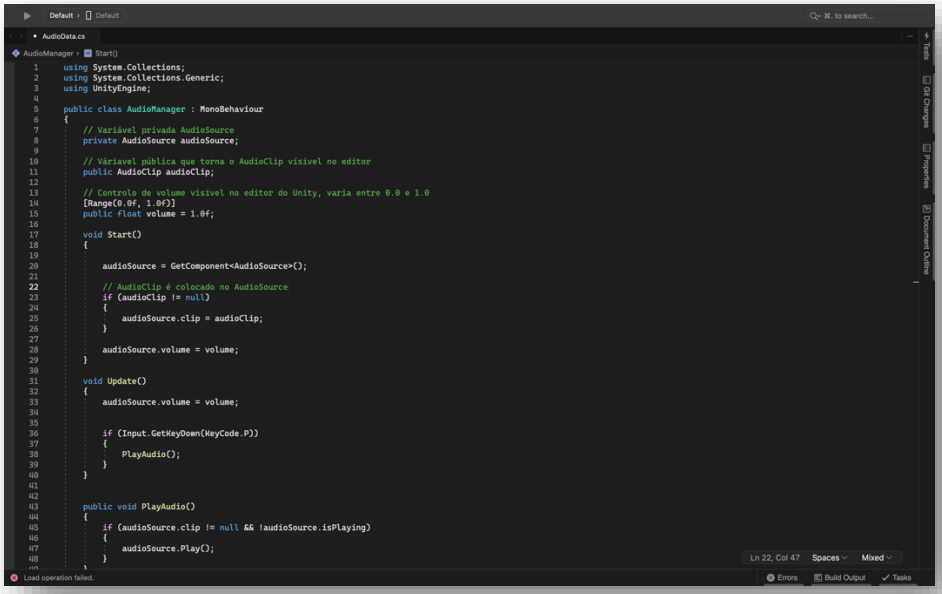
90% Apenas visualização

A1 Feitos

	Feitos	Som	Notas	Referencia		
2	☑	Shooting		https://drive.google.com/file/d/1DIeK-KCGxDKUg0ts_whkEhFQCLTOeAcO/view?usp=sharing		
3	☑	Pass (usar o mesmo que o shooting ou uma variante)		https://drive.google.com/file/d/1N3QnzaL_LqQ27WXNtzR1jhlDRUmoSZyc/view?usp=sharing		
4	☑	Ball hitting backboard		https://drive.google.com/file/d/1DIeK-KCGxDKUg0ts_whkEhFQCLTOeAcO/view?usp=sharing		
5	☑	Ball hitting hoop		https://drive.google.com/file/d/1DIeK-KCGxDKUg0ts_whkEhFQCLTOeAcO/view?usp=sharing		
6	☑	Ball hitting net		https://youtu.be/z2N8meONeAY?t=358		
7	☐	Ball dunked	Extra glow em comparação ao normal point			
8	☑	Dribble		https://youtu.be/z2N8meONeAY?t=163	https://youtu.be/_oZmFNHF98E?t=227	
9	☑	Dunk indicator audio cue				
10	☐	Shooting UI	Scale			
11	☑	Crowd / background sound				

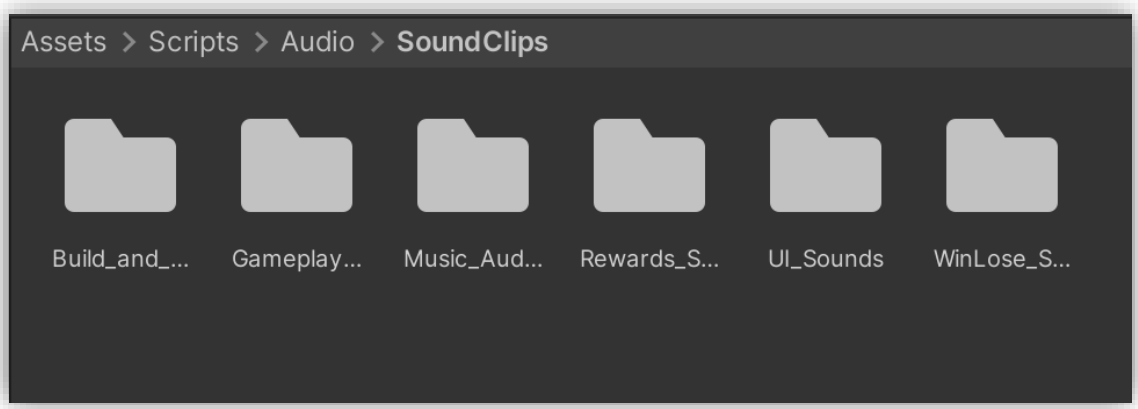
Saikutsu 2 | Gorufu | Hoops clash | Bubble break | Merge sort | Sorted Goods | Split Numbers | Serve Away | Sand Match

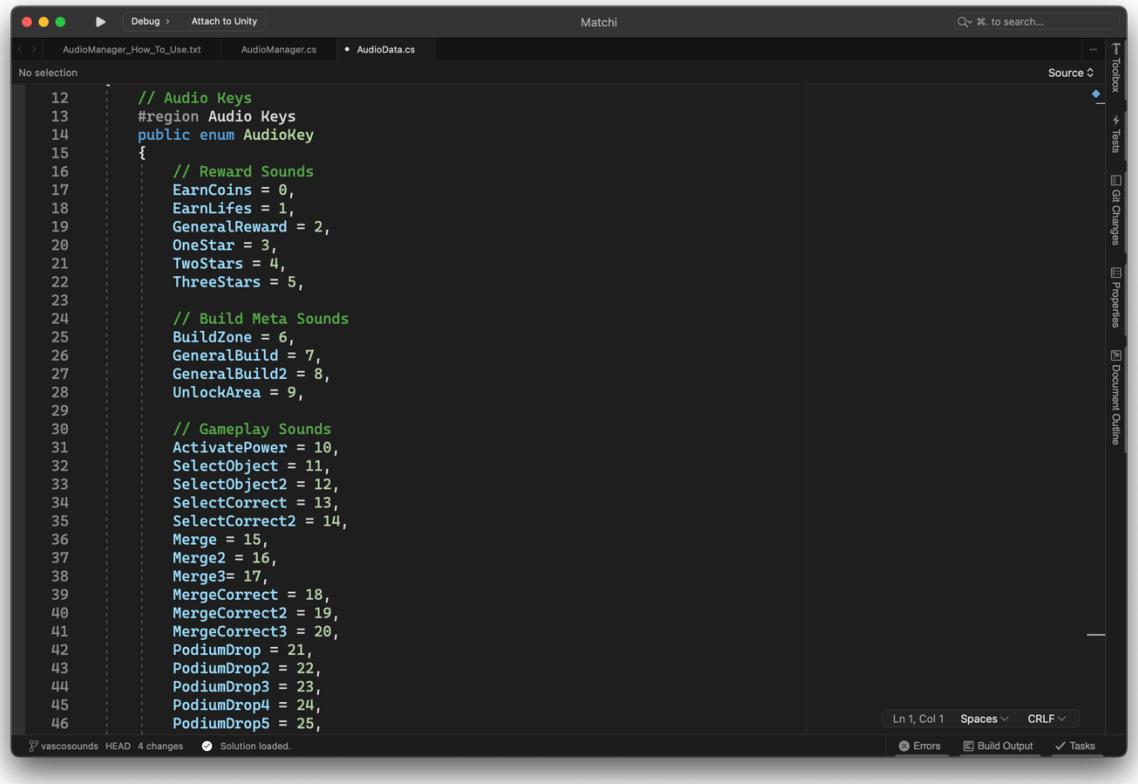
Apêndice B – Áudio script no Unity.



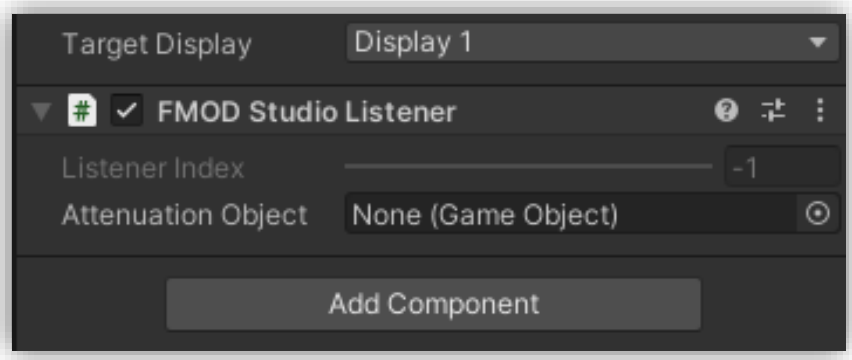
```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class AudioManager : MonoBehaviour
{
    // Variável privada AudioSource
    private AudioSource audioSource;

    // Váriavel pública que torna o AudioClip visivel no editor
    public AudioClip audioClip;

    // Controlo de volume visivel no editor do Unity, varia entre 0.0 e 1.0
    [Range(0.0f, 1.0f)]
    public float volume = 1.0f;

    void Start()
    {

        audioSource = GetComponent<AudioSource>();

        // AudioClip é colocado no AudioSource
        if (audioClip != null)
        {
            audioSource.clip = audioClip;
        }

        audioSource.volume = volume;
    }

    void Update()
    {
        audioSource.volume = volume;


        if (Input.GetKeyDown(KeyCode.P))
        {
            PlayAudio();
        }
    }


    public void PlayAudio()
    {
        if (audioSource.clip != null && !audioSource.isPlaying)
        {
            audioSource.Play();
        }
    }
```

Apêndice C – Unity Audio Assets e Script audio_data.



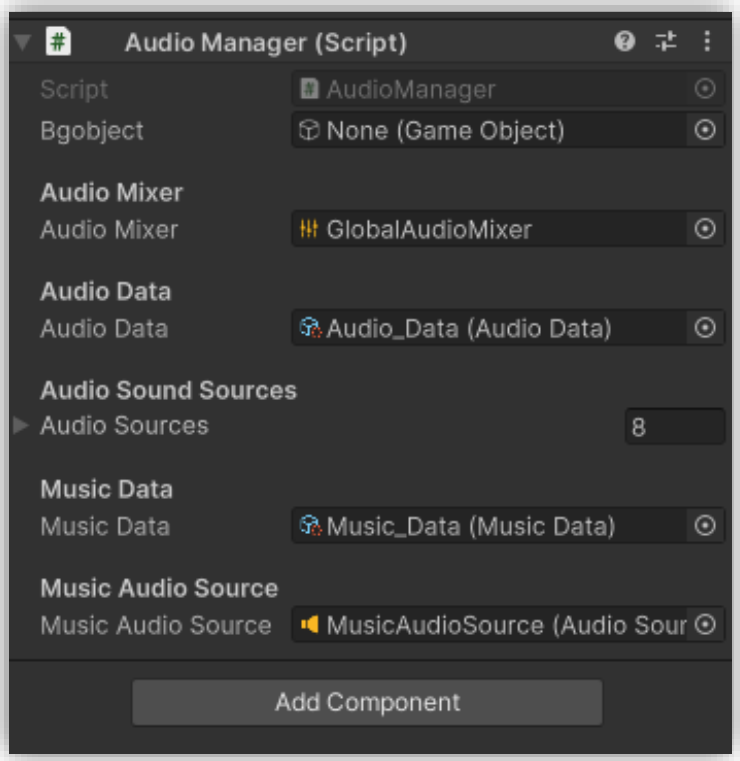


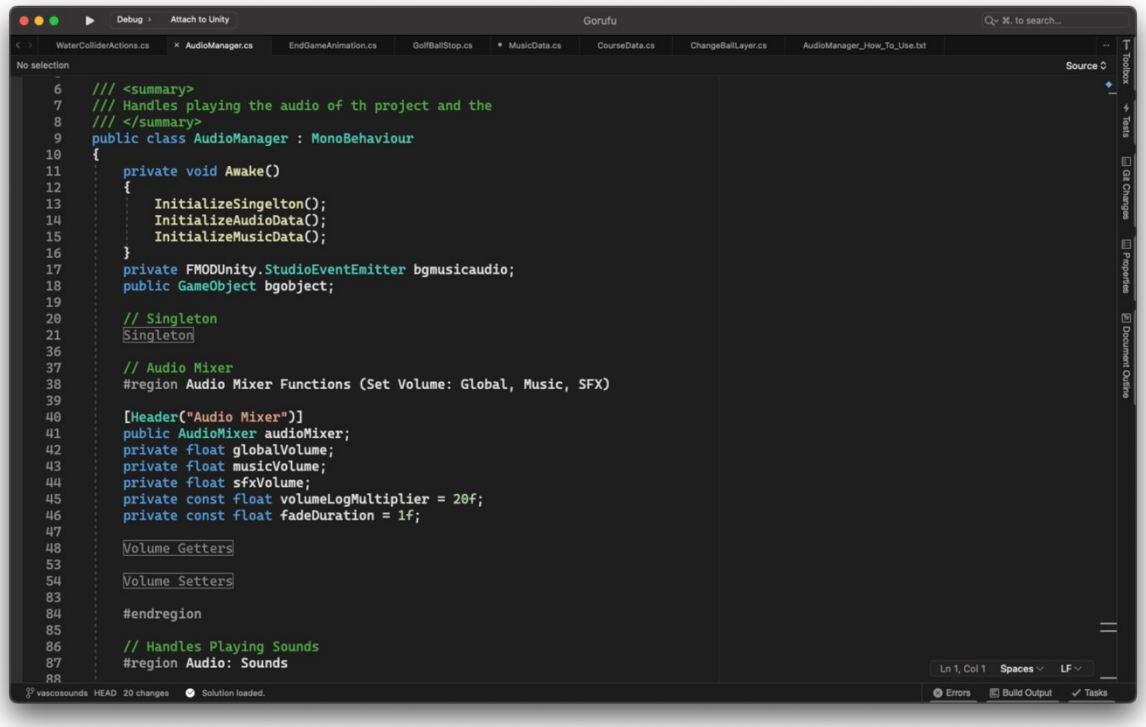
Apêndice D - FMOD Studio Listener.



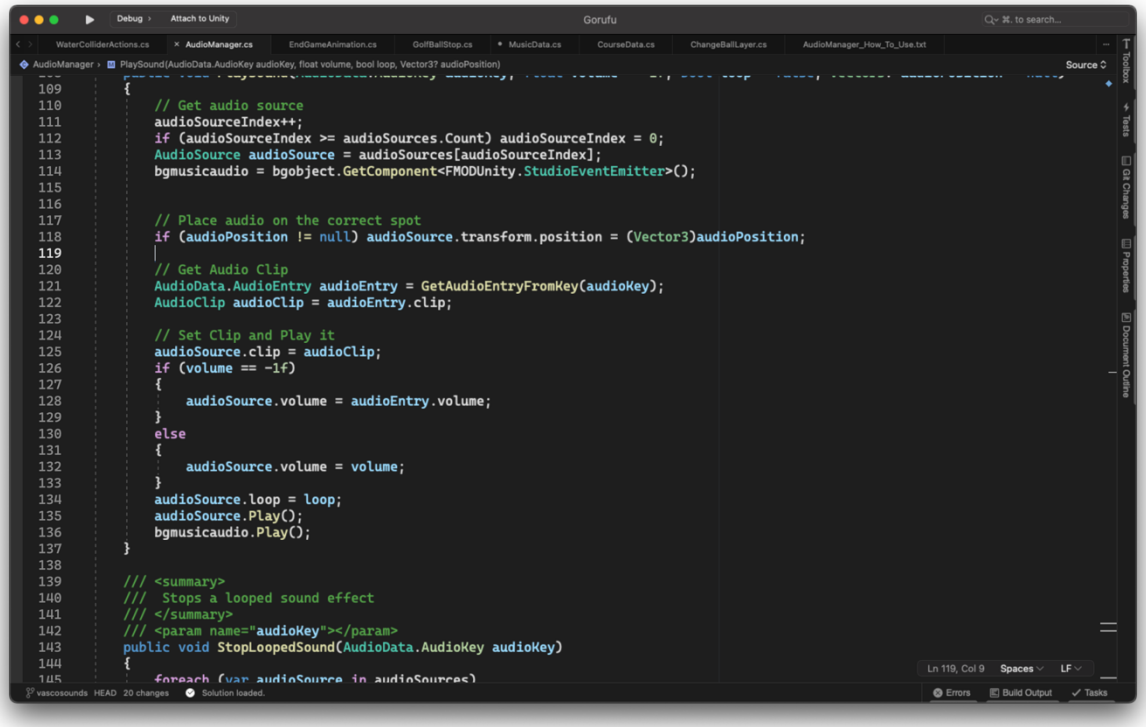
Apêndice E – FMOD Studio Event Emitter.

Apêndice F – Script EventManager.





```
/// <summary>
/// Handles playing the audio of th project and the
/// </summary>
public class AudioManager : MonoBehaviour
{
    private void Awake()
    {
        InitializeSingelton();
        InitializeAudioData();
        InitializeMusicData();
    }
    private FMODUnity.StudioEventEmitter bgmusicaudio;
    public GameObject bgobject;

    // Singleton
    Singleton

    // Audio Mixer
    #region Audio Mixer Functions (Set Volume: Global, Music, SFX)

    [Header("Audio Mixer")]
    public AudioMixer audioMixer;
    private float globalVolume;
    private float musicVolume;
    private float sfxVolume;
    private const float volumeLogMultiplier = 20f;
    private const float fadeDuration = 1f;

    Volume Getters

    Volume Setters

    #endregion

    // Handles Playing Sounds
    #region Audio: Sounds
```



```
    {
        // Get audio source
        audioSourceIndex++;
        if (audioSourceIndex >= audioSources.Count) audioSourceIndex = 0;
        AudioSource audioSource = audioSources[audioSourceIndex];
        bgmusicaudio = bgobject.GetComponent<FMODUnity.StudioEventEmitter>();


        // Place audio on the correct spot
        if (audioPosition != null) audioSource.transform.position = (Vector3)audioPosition;

        // Get Audio Clip
        AudioData.AudioEntry audioEntry = GetAudioEntryFromKey(audioKey);
        AudioClip audioClip = audioEntry.clip;

        // Set Clip and Play it
        audioSource.clip = audioClip;
        if (volume == -1f)
        {
            audioSource.volume = audioEntry.volume;
        }
        else
        {
            audioSource.volume = volume;
        }
        audioSource.loop = loop;
        audioSource.Play();
        bgmusicaudio.Play();
    }

    /// <summary>
    ///  Stops a looped sound effect
    /// </summary>
    /// <param name="audioKey"></param>
    public void StopLoopedSound(AudioData.AudioKey audioKey)
    {
        foreach (var audioSource in audioSources)
```