Instituto Superior de Tecnologias Avançadas

Licenciatura em Informática

ICYMAZE

Jogo de puzzle para plataformas Android

Realizado por - José Pedro Martins Pinto – 45156

Coordenador – Professor Dr. Pedro Brandão

Orientadora - Professora Dra. Dulce Mourato

Lisboa, Ano Letivo 2022/2023

Instituto Superior de Tecnologias Avançadas

Licenciatura em Informática

ICYMAZE

Jogo de puzzle para plataformas Android

Realizado por - José Pedro Martins Pinto – 45156

Coordenador – Professor Dr. Pedro Brandão

Orientadora - Professora Dra. Dulce Mourato

Lisboa, Ano Letivo 2022/2023

Dedicatória

Dedico em especial ao meu pai e à minha mãe pelo apoio pessoal que me deram, de forma a ter espaço e condições na minha vida para executar este projeto e ao meu avô que já não se encontra entre nós.

Dedico este trabalho à minha família e aos meus amigos.

Agradecimentos

Tenho a agradecer à minha família, aos meus colegas de curso, aos meus colegas de trabalho e aos professores que lecionaram as unidades curriculares da minha licenciatura, por todo o conhecimento transmitido e por todo o apoio dado na execução deste projeto.

Resumo

Este projeto pretende criar um jogo de puzzle, em 2D, com pixel *art*, no *engine* de jogos Unity, para dispositivos móveis. A mecânica principal do jogo é deslizar a personagem principal pelo gelo, entre obstáculos, de forma a chegar à meta com o mínimo número de ações possível.

Foi possível verificar na prática algumas fontes bibliográficas sobre criação de jogos, diversos registos publicados sobre o tema, tópicos publicados oficialmente que demonstrassem a validade ou invalidade de decisões a tomar no desenvolvimento, assim como objetivos e metodologias utilizados. As recolhas de dados e observações de desenvolvimento deste tipo de jogos, levaram às escolhas das ferramentas, plataformas ou aplicações utilizadas para a realização do projeto.

O conhecimento obtido pela preparação e execução do jogo ICYMAZE, das quais: técnicas de desenvolvimento de projeto, gestão de tempo, programação em C#, desenvolvimento em Unity e gestão de dados em formato CSV foram fundamentais para a realização do projeto.

Palavras-Chave: Unity, C#, CSV, Videojogo, Puzzle.

Abstract

This project aims to create a 2D puzzle game with pixel art, using the Unity game engine, for mobile devices. The main mechanic of the game is to slide the main character across the ice, between obstacles, in order to reach the goal with as few actions as possible.

It was possible to verify in practice some bibliographic sources on game development, various published records on the subject, officially published topics that demonstrated the validity or invalidity of decisions to be made in development, as well as objectives and methodologies used. The collection of data and observations of the development of this type of game led to the choices of tools, platforms, or applications used for the project's realization.

The knowledge obtained through the preparation and execution of the game ICYMAZE, such as project development techniques, time management, C# programming, Unity development, and data management in CSV format, were fundamental for the realization of the project.

Keywords: Unity, C#, CSV, Videogame, Puzzle.

Índice

Introdução

Este projeto tentou responder ao desafio duplo de conceber um artefacto digital de final de curso e dar resposta a um gosto pessoal por videojogos de entretenimento.

O desenvolvimento deste jogo de puzzles tem como objetivo entreter quem o joga e utilizar mecânicas previamente usadas como-elementos visuais apelativos retro. Ao pesquisar sobre o tema foi entendido, que um jogo com estas mecânicas e gráficos é difícil de encontrar no mercado. O IcyMaze veio assim tentar colmatar essa inexistência.

Ao realizar alguma pesquisa bibliográfica e prática foi identificada a necessidade de utilização de um *engine* de jogos, que facilitasse este desenvolvimento, tal como um programa de design gráfico, entre outros. A pesquisa foi feita online, de forma a apresentar os pros e contras de cada tecnologia possível de encontro com as necessidades dinâmicas de um ambiente de desenvolvimento. Com isto, foi planeado que o jogo a realizar terá um jogador principal que percorrerá um labirinto subterrâneo de forma a chegar ao fim do mesmo, o personagem irá deslizar no gelo de forma a navegar este labirinto. Várias formas de criação e locomoção deste labirinto foram estudadas e consideradas, mas no final esta foi a direção escolhida, dado a sua unicidade e caráter.

Em suma, este projeto teve por objetivo principal a criação de um jogo de puzzles, destinado ao entretenimento, desenvolvido para plataformas Android. Os outros objetivos secundários procuraram aumentar as competências de conceção, de design, de desenvolvimento sobre o tema do jogo e sobre as tecnologias a utilizar para a sua realização, como também entender quais as melhores formas de prender o foco do jogador segundo as mecânicas da gamificação.

Revisão da Literatura

O *Game Engine*, a categoria do jogo de puzzles escolhida e o modo como se apresenta habitualmente aos utilizadores e potenciais jogadores foram algumas preocupações que orientaram o capítulo de revisão de literatura, como o ponto de partida para responder à motivação da realização da parte prática: Qual foi a inspiração para a criação do jogo e quais as mecânicas utilizadas?

- Antes de mais importa esclarecer o conceito de *Game Engine*, segundo Haas (2014) Os motores de jogo ou Game Engine podem ser descritos como softwares que servem para criar jogos, de modo simplificado, integrados num ambiente de desenvolvimento próprio (IDE), cuja tecnologia garante a execução do áudio, dos aspetos físicos, da interação e das redes onde se desenrolam. Game Engine refere-se não só à estrutura (framework) que consiste em bibliotecas básicas para elaboração de jogos, mas também a conjugação entre uma framework e o respetivo editor visual.

No livro 'Game Engine Architecture' de Jason Gregory (2019) é identificado o *game engine* e chamado de Unity, que foi o escolhido para executar o projeto que condiz com os parâmetros necessários para o desenvolvimento de um jogo 2D, que pode ser jogado em dispositivos móveis. Procurou-se então o desenvolvimento de um *game engine* versátil, que facilitasse a escalabilidade do projeto, seguindo as coordenadas de Haas (2014) e Gregory (2019) usando Unity. Unity surge como um ambiente de desenvolvimento de jogos multiplataforma, que permite regular mecanismo de tempo de execução e disponibiliza suporte a uma ampla variedade de plataformas. O Unity possibilita aos desenvolvedores inscrever os seus jogos em plataformas móveis (como a Apple IOS, o Google Android), consolas (Microsoft Xbox 360 e Xbox One, Sony PlayStation 3 e PlayStation 4 e Nintendo Wii, Wii U), plataformas de jogos portáteis (como é o caso da Playstation Vita, Nintendo Switch), computadores desktop (Microsoft Windows, Apple Macintosh e Linux), boxes de TV por cabo ou por satélite (Android TV e TV OS) e sistemas de realidade virtual (VR) (por exemplo, Oculus Rift, Steam VR, Gear VR), entre outros.

Os mesmos autores Haas (2014) e Gregory (2019) descrevem algumas das potencialidades da Unity: o desenvolvimento simplificado num ambiente de editor integrado (onde se pode manipular os assets e as entidades que fundam o Mundo do Jogo) que o permitem incorporar em várias plataformas, podendo a ação em cada momento ser previsualizada quer em termos de software quer em termos de hardware

Emel Altiner, sobre a versatilidade dos jogos desenvolvidos em Unity, conclui:

> It can be said that spatial games, talking about mathematics, activities involving visual puzzles, and experiential teaching methods will also benefit students in gaining proficiency in mathematics and geometry. By adding Legos, puzzle games, blocks, tangrams, and wooden blocks to school syllabi. (Altiner, 2018. p.76)

Mediante o excerto acima referido, conclui-se que jogos de puzzle podem contribuir para o desenvolvimento lógico e visual de crianças na escola primária, que ultrapassa o simples poder de entretenimento e entra na área do crescimento pessoal.

A principal diferença entre os jogos de puzzle e os restantes, é a utilização de mecânicas que desafiam o jogador a resolver problemas, é um desafio mais lógico do que prático. Por exemplo, os jogos de ação são um exemplo de género, em que as suas mecânicas principais são por norma práticas. Apesar disso existem exceções sendo que o que está descrito é o comum.

No livro 'Moving from Unity to Godot' o autor Alan Thorn (2020) fala sobre várias razões para usar o *game engine.* Godot ao invés do Unity, é completamente grátis, mesmo para uso comercial. O Godot é *open source,* o que significa que qualquer pessoa pode analisar o código, usá-lo, descarrega-lo para uma cópia local, compilá-lo ou até mesmo acrescentar algo ao programa. O Godot suporta iluminação global, mesh de navegação, determinação de caminho, programação visual e ferramentas 2D para sprites e interface de utilizador (Thorn, 2020).

Para responder à questão de quais as recompensas que devem ser dadas e disponibilizadas ao utilizador, após completar um nível de jogo com sucesso, foram aplicadas algumas das funcionalidades, descritas por Tyni et al. (2023), que podem servir para prender a atenção do utilizador às expetativas de sucesso, que pode vir a ter, se completar todas as etapas do jogo.

> Sensory feedback was found in every game, usually in the form of a sound or a visual effect after a player action was performed. The second popular reward type was glory, which came in the form of points or achievement systems. This was closely followed by access because most of the games had various levels to play. The number of sustenance rewards was found to be steadily increasing in the data set. Praise was usually given to the player in the form of verbal and written praise after successfully completing a task or a level, but it still did not see much appearance in the data set when compared to other reward types. (Tyni et al., 2023. p.3)

Houve também vantagem em efetuar uma pesquisa sobre as especificidades dos jogos de puzzle, que já tinham sido desenvolvidos com algum sucesso. Foram pesquisados e reunidos os principais jogos que inspiraram o IcyMaze e tiveram influência sobre as decisões de design e mecanismos deste projeto.

O primeiro jogo icónico e clássico (denominado Portal 2) consiste em criar portais de um local para o outro, num espaço tridimensional, de forma a conseguir com outros mecanismos adquiridos ao jogar, chegar ao final do nível. No site https://www.thinkwithportals.com/ é possível conhecer todas as suas características e perceber quais as funcionalidades que realmente funcionavam num jogo semelhante.

O projeto atual utiliza um aspeto similar de resolver o puzzle, com os vários elementos presentes de forma a chegar ao fim do nível.

Figura 1 - Portal 2 (Lund-Hansen, 2011 p.4)

Um clássico de RPG, consiste em capturar Pokemon's (animais fictícios) e ir avançando na história do jogo, através de lutas com esses animais. O projeto atual utiliza umas frações

deste jogo, que consistem em navegar uma gruta de gelo, em que o jogador tem de planear qual a rota indicada para chegar ao fim. No site https://www.pokemon.com/us/pokemon-video-games é possível conhecer todas as suas características.

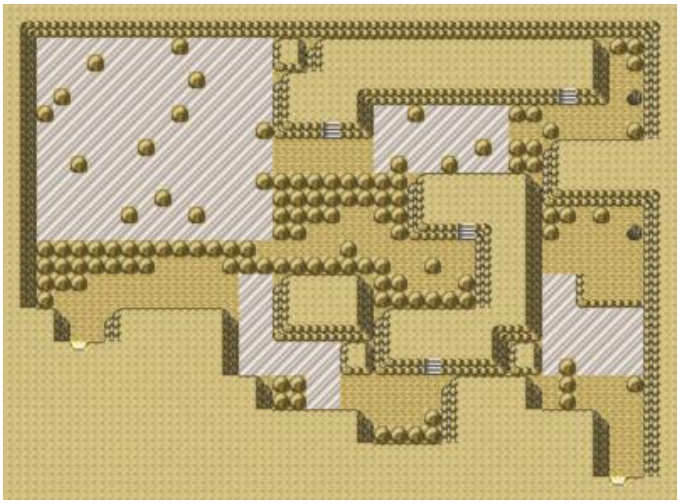

Figura 2 - Pokemon Gold (Game Freak, 2004 p.4)

Uma destas escolhas foi especialmente interessante, a série de jogos Pokémon foi uma grande influencia sobre este projeto académico, porque o mecanismo de puzzle principal tem como base os pequenos puzzles encontrados nesta série de jogos. Os restantes exemplos foram excelentes para fundamentar o design de níveis (Portal 2).

Numa outra fase, procurou entender-se como é que no *game engine* o jogo seria realizado, como seriam executados os gráficos (em que programa) e quais as melhores práticas para a execução de cada elemento do jogo? (inclusive a investigação de mecânicas e a sua execução).

Começando assim por o essencial, a escolha do *game engine*, esta foi feita de forma a responder aos seguintes requisitos:

- Grátis
- Acessível
- Gráficos 2D
- Desenvolvimento para dispositivos móveis

Foi então feita a pesquisa que concluiu-se que de todos os *game engines* grátis o mais acessível era o Unity, pela enorme comunidade, extensa documentação e conhecimento prévio pessoal.

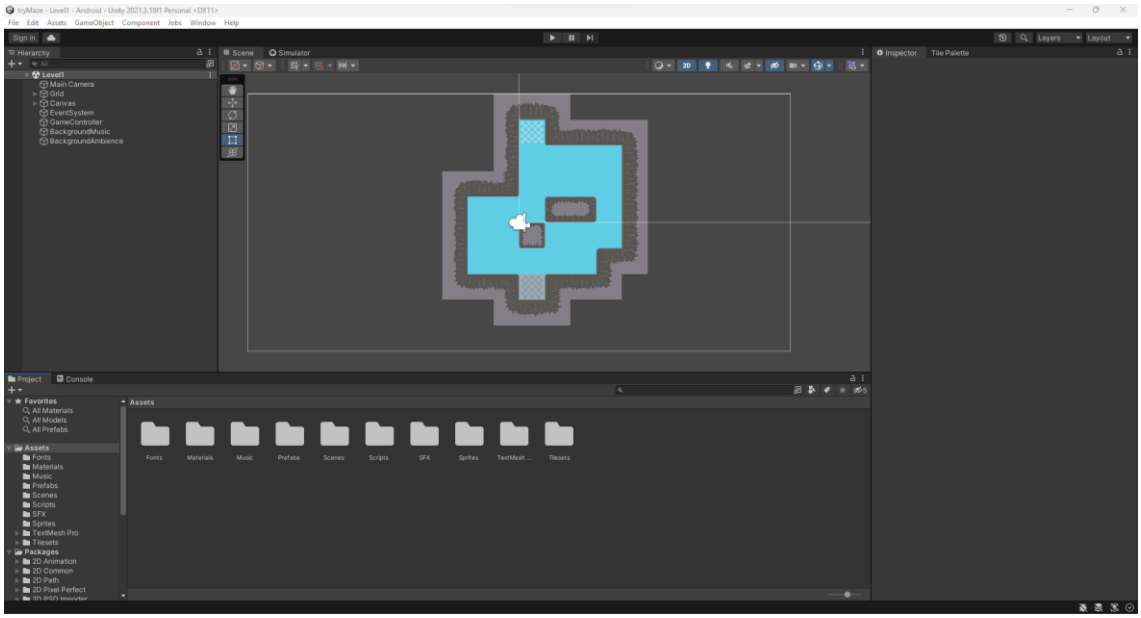

Figura 3 - Projeto Unity (Pinto, 2023 p.4)

O *game engine* Godot também seria uma opção viável para este projeto, apesar de que a escolha do Unity não foi apenas pela sua capacidade e compatibilidade, mas também por pessoalmente ter experiência com o mesmo, no entanto o aspeto principal por não ter escolhido o Godot foi a dimensão e quantidade de recursos existentes para o Unity em comparação com o Godot.

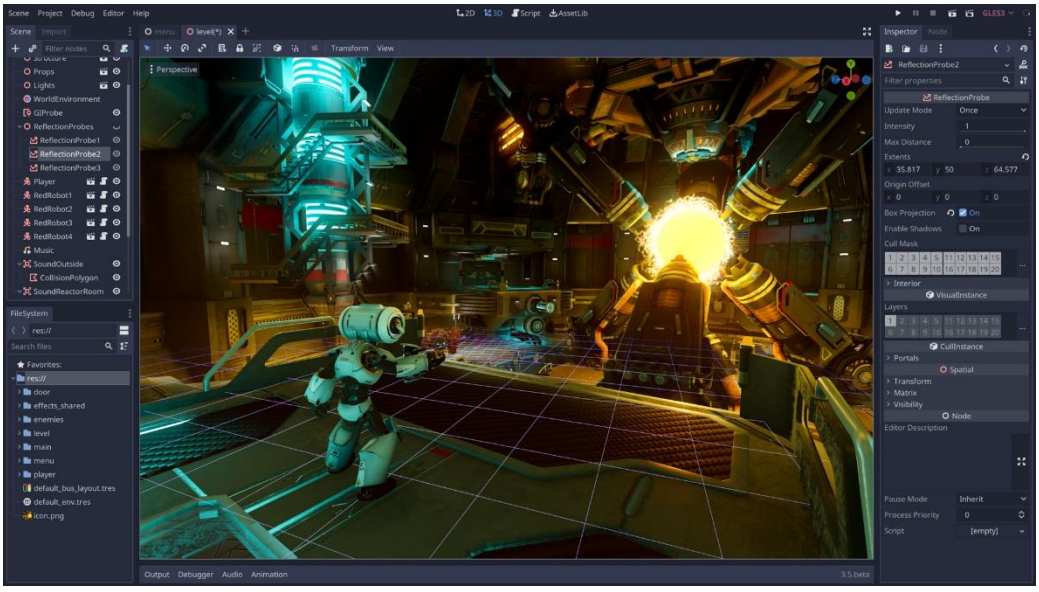

Figura 4 - Game Engine Godot (Godot, 2023 p.5)

Materiais e Métodos de Desenvolvimento

A questão base proposta no início deste projeto foi: Como fazer um jogo de puzzle inovador e interessante? – O desafio implicou uma série de etapas referidas ao pormenor no capítulo dos Resultados, que resultaram da observação e revisão de literatura, pesquisa em Websites oficiais de diversos jogos, experimentação e Q&A/Vídeos online, entre outra documentação que acabou por inspirar e condicionar certas escolhas.

Foi assim realizada uma investigação qualitativa, exploratória que possibilitou ir concretizando cada etapa do jogo, através da experimentação, adicionando mais características sempre que necessário. Pathak; Jena & Kalra (Jul–Sep 2013) explicam que a investigação qualitativa pretende entender uma questão de pesquisa em perspetivas humanísticas ou idealistas.

Foi também apresentado um pequeno cronograma de investigação do projeto, de forma a definir as datas gerais para as atividades e trabalhos a desenvolver:

Datas	Atividades/trabalhos a desenvolver
Outubro 2022 - Março 2023	Apresentação e aceitação do tema. Desenvolvimento do pré-projeto. Desenvolvimento e apresentação do Pitch.
Março 2023 – Setembro 2023	Entrega do trabalho final: parte prática e parte teórica baseada no anexo do Projeto Académico, com a declaração do Orientador e a declaração de autoria do aluno.

Nenhum código de programação utilizado no corrente projeto tem direitos de autor, optou-se, por isso, aceder a recursos *open source*, de modo a garantir flexibilidade de escolhas, sem qualquer tipo de restrições.

De seguida, apresenta-se o script do jogo, que tem por objetivo dar uma visão geral do ambiente de jogo e dos seus objetivos de entretenimento.

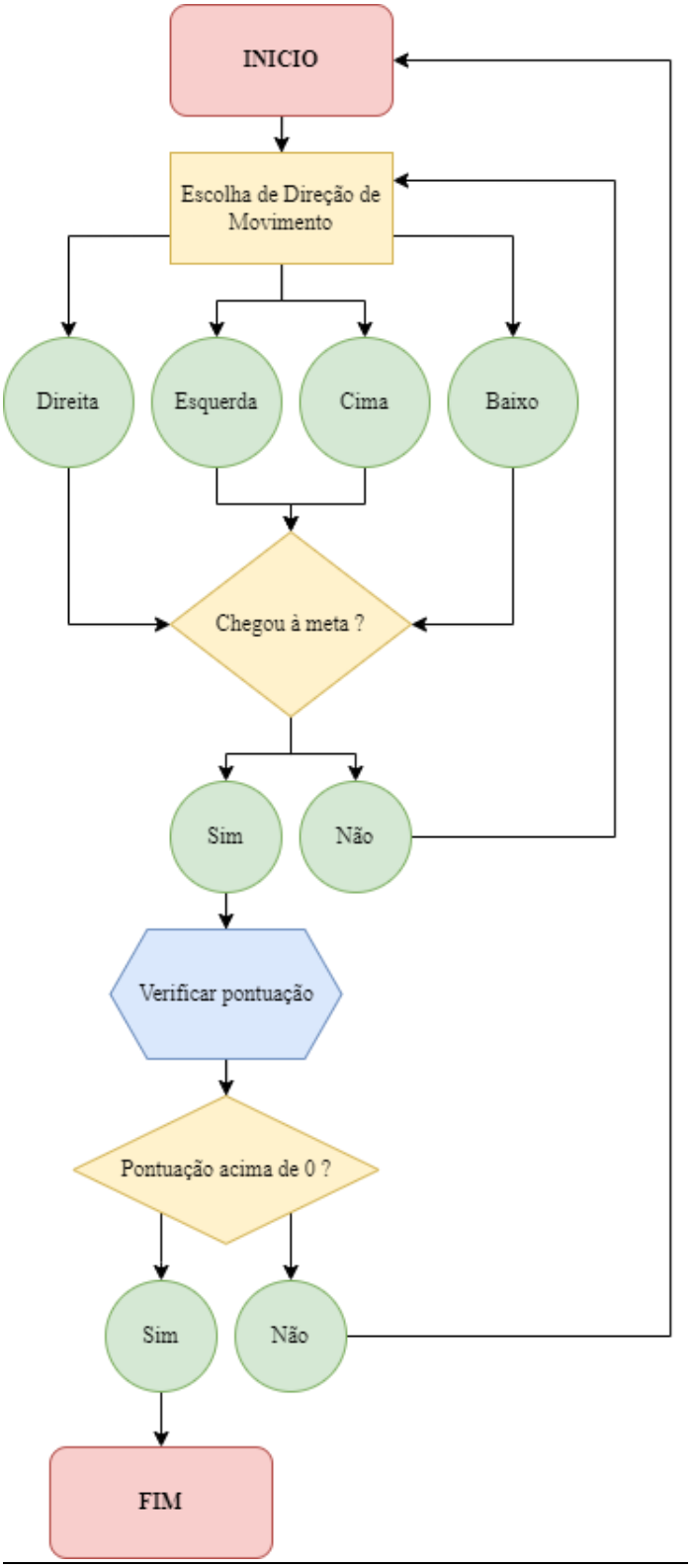

Figura 5 - Diagrama Funcional (Pinto, 2023, p.1).

Como o público-alvo deste jogo são pessoas dos 8 aos 80 não foi necessário estabelecer quaisquer restrições de acesso.

Resultados

No decorrer do desenvolvimento do jogo houve várias questões/dificuldades, as principais que mais sobressaíram foram:

Como pintar um tile (um pequeno bloco gráfico usado para construir mapas e cenários) tendo vários componentes próprios distintos dos restantes num *tilemap*, esta questão teria sido fácil de resolver, mas infelizmente a versão do Unity utilizada tinha deprecado essa funcionalidade que ainda estava por refazer, assim, utilizando o modo de *debug* do Unity foi possível inserir um *prefab* em qualquer tile individual, resolvendo assim o problema, apesar de não ser uma opção recomendada oficialmente por o *engine*.

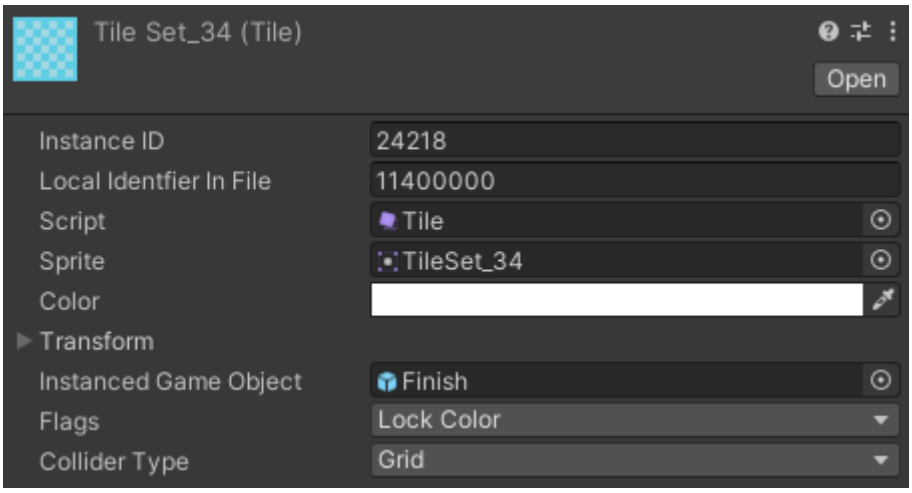

Figura 6 - Propriedades de um Tile no Unity (Pinto, 2023 p.2)

Como mover e limitar elementos?

Para resolver esta questão foi utilizado o que se chama um Rigidbody2D (um elemento que dá propriedades físicas reais ao objeto que foi adicionado) no *Prefab* (Objeto criado, editado e configurado que é guardado para uso futuro) do jogador, que através da utilização de força simulada num espaço 2D sem gravidade foi possível criar o tipo de movimento necessário para o jogador.

Outro problema relacionado foi a forma como o jogador era limitado por paredes e obstáculos ao longo dos níveis foi criar esse ambiente da melhor forma possível dada a construção dos níveis utilizando *Tilemaps* (Um tilemap é um mapa composto por um plano

quadriculado, do qual cada bloco é chamado de "tile".), para resolver esta questão foram adicionados elementos de colisão a um *tilemap* criado para tal em cada nível, ao que assim todos os Tiles pintados nesse *tilemap* tem colisão com o *Player*.

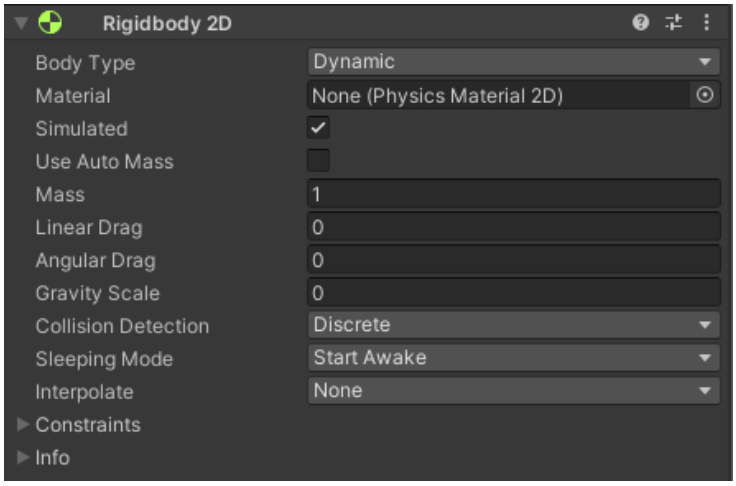

Figura 7 - Elemento Rigidbody2D do Player (Pinto, 2023 p.2)

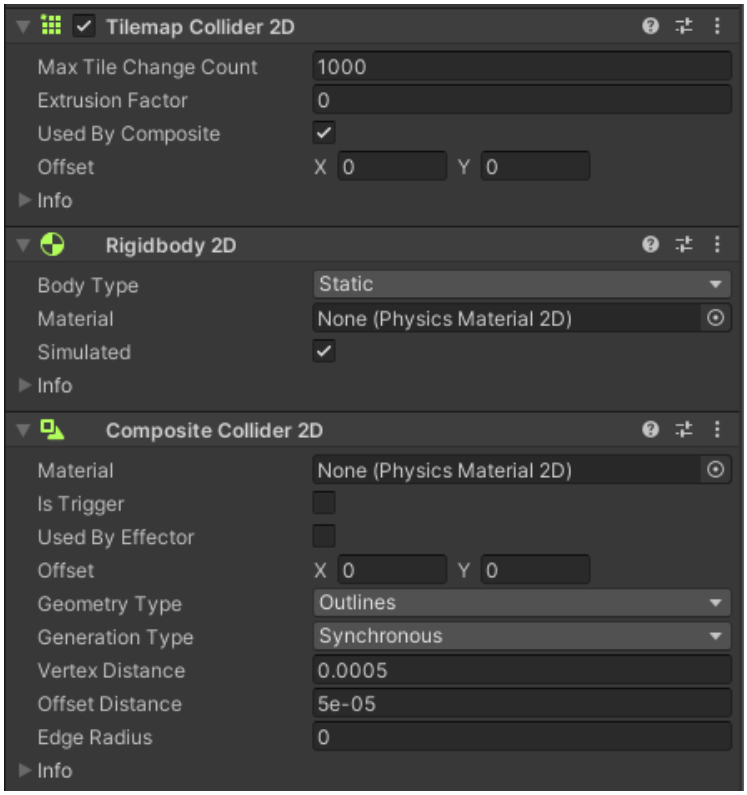

Figura 8 - Elementos do Tilemap com colisão (Pinto, 2023 p.3)

Como contruir um UI?

Para esta questão fui entender como se construía UI no Unity, quais as ferramentas disponíveis para tal, ao que entendi que existem 2 elementos essenciais que independentemente do tamanho do ecrã conseguem escalar de forma correta e sistemática os elementos presentes, estes são 'Vertical Layout Group' e 'Horizontal Layout Group'. Apesar disso. também foi necessária alguma manipulação por script desses elementos de forma a construir um UI mais genérico que desse para "reciclar" noutros casos e também para poupar tamanho e performance no projeto.

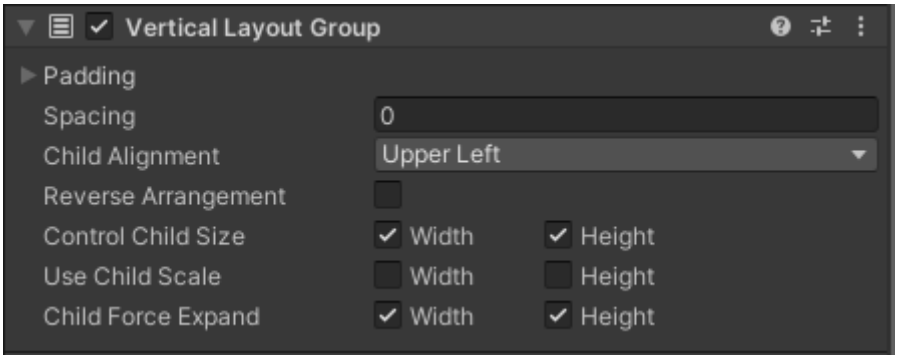

Figura 9 - Vertical Layout Group (Pinto, 2023 p.2)

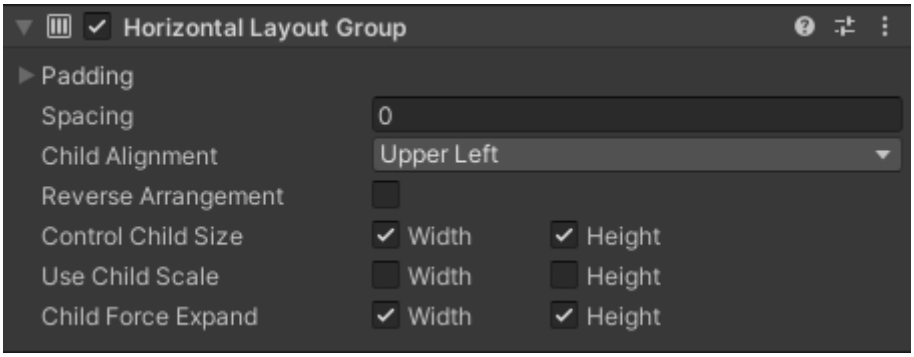

Figura 10 - Horizontal Layout Group (Pinto, 2023 p.2)

Como guardar o jogo?

Para responder a esta questão foi bastante simples, apesar de inicialmente parecer abstrata, foi resolvida da seguinte forma, dado que o jogo tem dois elementos de progressão, o nível em que o jogador está e quantas estrelas conseguiu por nível (0 a 3) são os únicos dados essenciais para guardar o avanço do jogador, assim é guardado em formato csv usando o numero do nível primeiro e de seguida as estrelas que teve, desta forma o ultimo nível disponível é aquele que tem 0 estrelas após o que tem mais que 0, assim não só é possível dar essa informação ao jogador como também guardar e manter registado para futuras sessões.

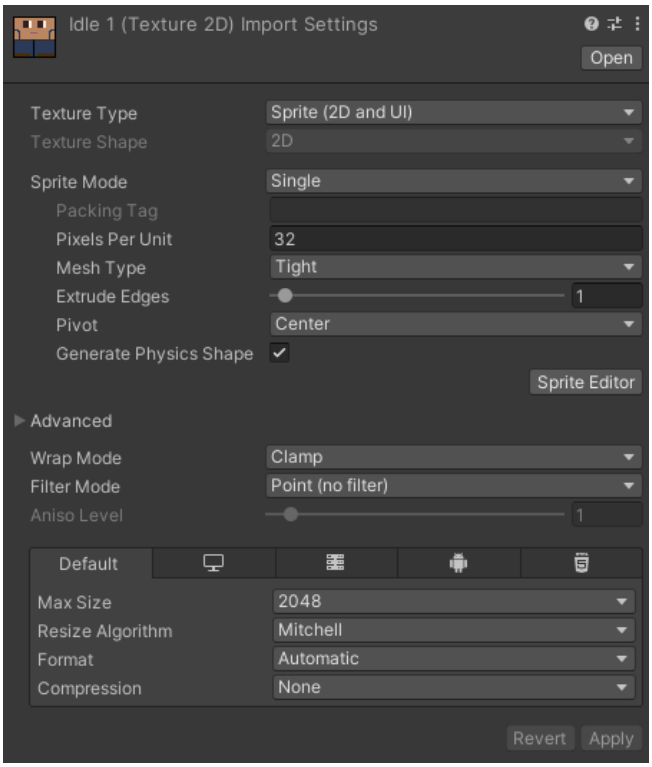

Figura 11 - Propriedades Sprite (Pinto, 2023 p.3)

Como arranjar as *sprites*?

Existia um problema com as *sprites* do jogo, o facto de ficarem "desfocadas" ao serem aplicadas no jogo, que levou-me a procurar as definições de renderização de *sprites* do Unity e alterar algumas configurações do projeto para corrigir este problema.

De seguida foi necessário identificar qual o programa de design gráfico 2D seria o ideal para o projeto, para isso a escolha teria de responder aos seguintes requisitos:

- Fácil usabilidade
- Múltiplas *layers*
- Esquema de cores
- Exportação em vários formatos comuns

Com estes requisitos em mente o programa escolhido foi o Aseprite, apesar de ser pouco conhecido, servia perfeitamente a necessidades acima descritas e facilitava a criação de *spritesheets* em pixel *art*, devido a ser criado com esse objetivo em mente, para alem disso também é conhecido no mundo de videojogos em que é utilizado por bastantes criadores gráficos na área tendo assim uma boa base de documentação e pesquisa para qualquer dúvida necessária.

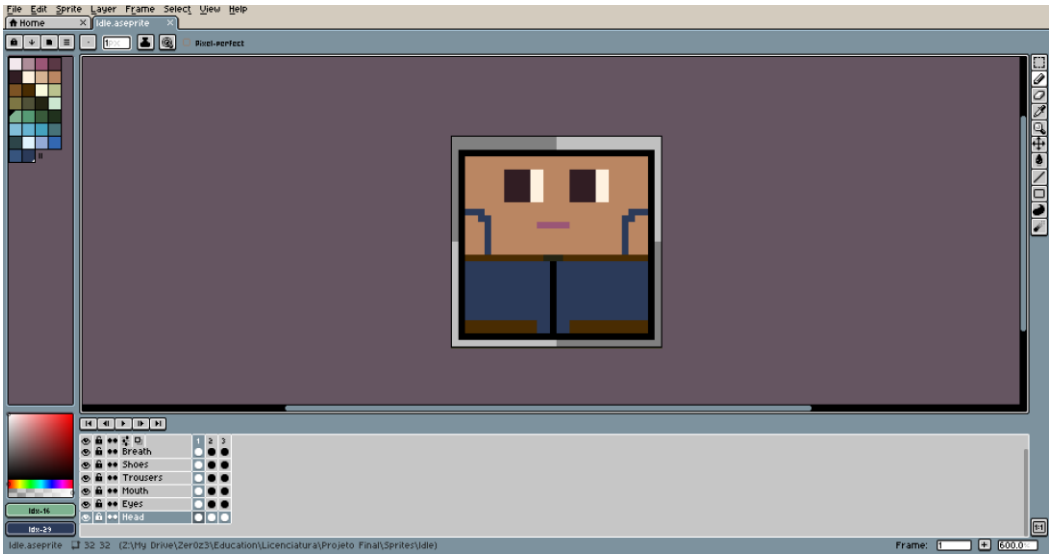

Figura 12 - Programa Aseprite (Aseprite, 2023 p.5)

Discussão de Resultados

Os resultados obtidos são favoráveis ainda que com alguns senãos e questões recorrentes, o projeto está concluído e conseguiu atingir os objetivos pretendidos, um jogo de puzzle com alguma dinâmica e desafio, feito em Unity e desenhado em pixel *art*.

Figura 13 - Gameplay Projeto (Pinto, 2023 p.5)

O uso da palete de cores criou uma consistência homogenia ao longo da construção da arte para o projeto que foi a razão exata para a escolha da mesma, como também o apoio visual que dá a todo o jogo, com o seu tema de cores mais frio, assim homogeneizou os visuais do jogo, de forma a que todo o projeto fosse mais conciso e consistente.

Desta forma foi escolhida a seguinte palete de cores de forma a refletir o tema do jogo:

Figura 14 - Palete de cores utilizada (Pinto, 2023 p.5)

Depois veio a questão do tamanho dos *sprites* onde foi escolhido 32x32px, esta escolha foi para dar um look retro ao jogo e também para facilitar o desenvolvimento dos gráficos/*sprites*.

Figura 15 - Cadeado (Pinto, 2023 p.6)

A criação dos gráficos a 32x32 pixéis deu o resultado retro esperado, que apesar de ter menos detalhe visual, a qualidade contida é a esperada, criando um look retro e simples de visualizar.

O *engine* escolhido foi o correto, pois conseguiu albergar todos os requisitos, tanto visuais como técnicos, ainda que teve alguns problemas e dificuldades na flexibilidade de uso do Tile Palette, que tinha poupado tempo e pesquisa se a função necessária estivesse disponível como em verões anteriores, ou se houvesse uma alternativa viável suportada oficialmente.

A construção dos scripts em C# foram relativamente acessíveis na sua criação pois a documentação do Unity é bastante completa e acessível, criando um ambiente de programação para o *engine* fácil e flexível.

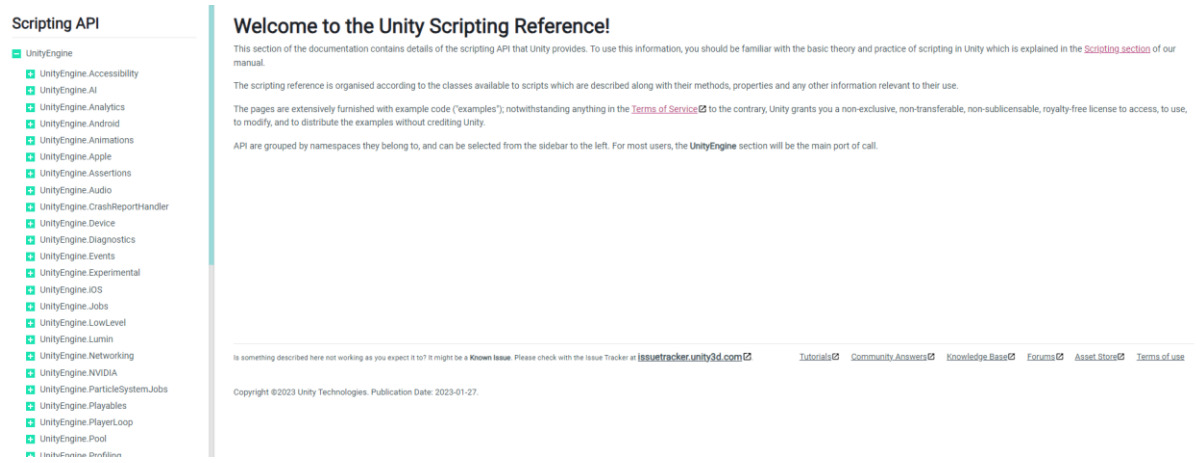

Figura 16 - Documentação Unity (Unity, 2023 p.6)

Apesar de todos estes pontos positivos, teria sido interessante experimentar o outro *engine* de jogos concorrente, o Godot, que podia ter facilitado algumas das dificuldades encontradas e questões mais complicadas de resolver, visto existir mais informação por ser *open source*.

O programa do Aseprite foi adequado, porque não houve qualquer tipo de dificuldade ou limitação no seu uso, serviu para criar os gráficos do jogo com a palete limitada de cores e em 32x32px.

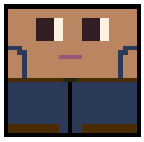

Figura 17 - Personagem principal (Pinto, 2023 p.6)

Conclusão

Em conclusão o projeto cumpriu todos os objetivos da sua construção, em que todas as limitações e dificuldades encontradas, como as mencionadas no capítulo dos resultados, foram superadas da melhor forma possível, analisando o problema e verificando o melhor caminho para a sua resolução.

Toda a pesquisa feita foi através de documentações e livros publicados, como também vídeos no Youtube e alguns artigos online, ambos de especialistas na área/material.

Em retrospetiva este projeto mostrou algum desafio, apesar do conhecimento prévio em gaming, assim como também projetos realizados em Unity, especialmente na área mais técnica do mesmo, ainda que as dificuldades tenham sido superadas, com normalidade.

Aprendi bastante sobre o *game engine*, as suas funcionalidades e as suas limitações, especialmente a utilização dos gráficos em 2D e as melhores práticas para a criação de scripts em C# direcionados ao ecossistema do Unity.

Tenho também bem presente a realidade do desenvolvimento de videojogos, que não tinha anteriormente, melhorei significativamente o conhecimento técnico em múltiplas áreas, como programação, design, som, base de dados e também nas diversas tecnologias utilizadas.

De forma a dar continuidade a este projeto, seria interessante um desenvolvimento mais aprofundado sobre o tema e sobre as mecânicas do próprio jogo, que aumentasse a dimensão e complexidade.

Referências Bibliográficas

Unity Technologies (2023) *Blurry Sprites*. Unity Discussions. Unity. (2014, October 9). https://discussions.unity.com/t/blurry-sprites/118978

Çilingir Altıner, E. (2018). Relationship between Spatial Thinking and Puzzle Games of Elementary School Students. *International Online Journal of Educational Sciences*, *10*(1). https://doi.org/10.15345/iojes.2018.01.008

Coco Code. (2020, November 9). *Master Unity Button OnClick actions! - Unity UI tutorial*. Www.youtube.com. https://www.youtube.com/watch?v=Ky-bzQFxV2U

French, J. (2021a, March 21). *Input in Unity made easy (complete guide to the new system)*. Game Dev Beginner. https://gamedevbeginner.com/input-in-unity-made-easy-complete-guide-to-the-new-system/#how_to_manage_input

French, J. (2021b, October 14). *How to move objects in Unity (3 methods with examples)*. Game Dev Beginner. https://gamedevbeginner.com/how-to-move-objects-in-unity/

Game Dev Guide. (2020, February 18). *Making UI That Looks Good In Unity using Color Palettes, Layout Components and a Blur Panel*. Www.youtube.com. https://www.youtube.com/watch?v=HwdweCX5aMI

Game Dev Guide. (2021, July 30). *Making a Modal Window in Unity*. Www.youtube.com. https://www.youtube.com/watch?v=SzQABx2YTJA&list=PLXD0wONGOSCKcUJHc4-7LIkEgvFvJ-nl1

Gregory, J. (2019). *Game engine architecture*. Crc Press, Taylor & Francis Group.

Haas, J. K. (2014). *A History of the Unity Game Engine*. https://digital.wpi.edu/pdfviewer/2f75r821k

Meebou. (2021, April 30). *2D Extras - Prefab Brush gone??* Unity Forum. https://forum.unity.com/threads/2d-extras-prefab-brush-gone.1102141/

Nintendo/Creatures Inc./GAME FREAK inc. TM, ®Nintendo. (2023). *Pokémon Video Games | Pokemon.com*. Www.pokemon.com. https://www.pokemon.com/us/pokemon-video-games/all-pokemon-games/

Pathak, V.; Jena, B. & Kalra, S. (Jul–Sep 2013). Qualitative research. Perspectives in Clinical Research 4(3):p 192. DOI: 10.4103/2229-3485.115389. URL: https://journals.lww.com/picp/fulltext/2013/04030/qualitative_research.10.aspx

SpeedTutor. (2021, July 29). *How To Create Buttons in Unity | Beginner UI Tutorial*. Www.youtube.com. https://www.youtube.com/watch?v=gSfdCke3684

Sunny Valley Studio. (2021, November 22). *How to use and adjust 2D collider in Unity*. Www.youtube.com. https://www.youtube.com/watch?v=eDOxDJEtE14

Technologies, U. (2023, July 7). *Unity - Scripting API: Rigidbody*. Docs.unity3d.com. https://docs.unity3d.com/ScriptReference/Rigidbody.html

Thorn, A. (2020). *Moving from Unity to Godot*. Apress.

Thorson, M., Berry, N., & Raine, L. (n.d.). *Celeste*. Celeste. https://www.celestegame.com/

Tyni, J., Turunen, A., Kahila, J., Bednarik, R., & Tedre, M. (2023). Reward Types in Popular Recreational and Educational Mobile Games. *IEEE Access*, *11*, 1166–1174. https://doi.org/10.1109/access.2022.3231936

Valve Corporation. (2011). *Official Portal 2 Website*. Www.thinkwithportals.com. https://www.thinkwithportals.com/

Zotov, A. (2020, September 8). *How To Set A Sprite To Particles In Unity Software Particle System | Particles Material Tutorial*. Www.youtube.com. https://www.youtube.com/watch?v=7q6GSVfyUbQ

Anexos

Manual de Projeto.docx