Universidade Estadual do Sudoeste da Bahia
Departamento de Ciências Exatas e Tecnológicas
Curso de Ciência da Computação

Desenvolvimento de um jogo educativo utilizando o Unity3D

Gabriel Almeida Dias

Orientador:
Prof. Dr. Marlos Marques

Vitória da Conquista - BA
Julho - 2017

Universidade Estadual do Sudoeste da Bahia
Centro de Ciências Exatas e Tecnológicas
Curso de Ciência da Computação

Desenvolvimento de um jogo educativo utilizando Unity3D

Gabriel Almeida Dias

Trabalho de conclusão de curso apresentado junto ao curso de Ciência da Computação da Universidade Estadual do Sudoeste da Bahia, na área de concentração de Ciências Exatas, como requisito parcial à obtenção do título de Bacharel.

Vitória da Conquista - BA
Julho - 2017

Sumário

Lista de Figuras

Resumo

A educação há alguns anos vem apresentando um crescente déficit em relação à aprendizagem, causada em grande parte pelo desinteresse dos estudantes na forma como os conteúdos são passados, abrindo assim espaço para recursos como os jogos, que veem como um objeto de aprendizagem inovador, prendendo a atenção e despertando o fascínio nas novas gerações, além de desenvolver habilidades importantes, indo além do mero divertimento e trazendo no bojo, em sua forma eletrônica, todos os recursos que um jogo comum traz e ainda mais, como a intensa imersividade que suas histórias e mecânica podem trazer, por isso nas escolas atuais vêm a ser uma excelente opção, resgatando o interesse dos jovens e trabalhando os conteúdos de forma mais interativa. Porém o desenvolvimento desses jogos não é algo trivial, envolvendo além de uma documentação extensa, própria de sistemas computacionais, adicionada a outro conjunto documental referente à área de jogos, também é necessária uma gama de conhecimentos em diversas áreas da computação como: programação, banco de dados, computação gráfica, interação humano-computador, dentre outras. Para auxiliar os desenvolvedores na criação desse tipo de sistema específico, que precisa possuir muitos elementos audiovisuais e uma grande interatividade com o usuário, foi criado um conjunto de ferramentas que dão suporte a todo o desenvolvimento, e ao coração desse conjunto de *sofwares* damos o nome de *engine de games*. Pretendemos com este trabalho criar um jogo eletrônico educacional (JEE) que sirva como um objeto de aprendizagem mais interessante ao aluno, potencializando as chances deste adquirir o conhecimento que o professor deseja trabalhar, para isso, iremos virtualizar uma sequência didática usada como jogo, já testada e validada em colégios municipais da cidade de Vitória da Conquista, chamada Fazendinha Matemática, sendo esta uma sequência voltada ao ensino e aprendizagem das operações básicas da matemática. No desenvolvimento utilizaremos a famosa *engine* Unity3D, mostrando cada aspecto do desenvolvimento desde a fase de concepção, documentação, até a programação em si e a vinculação dos elementos audiovisuais de forma detalhada, para que este trabalho sirva como referência para outros desenvolvedores que busquem o uso dessa ferramenta em futuros projetos de jogos. Obtemos ao final deste trabalho um jogo em fase beta que foi avaliado positivamente por professores em mais de uma fase de seu desenvolvimento.

Palavras Chaves: Unity, Desenvolvimento de Games, Jogos Educativos.

Abstract

In some years the education has been showing a growing shotfall in relation to learning, in large part caused by the lack of interest of students in the way content is taught, opening space for resources such as games, which they see as an innovative learning object, arresting The attention and the awakening to the fascination in the new generations, besides developing the important skills, going beyond the mere fun and bringing in the bulge, in its electronic form, all the resources for a common game and still more, like an intense emissividad that its stories and mechanics can bring, so in today's schools, offering an excellent choice, rescuing the interest of young people and working on content in a more interactive way. However, the development of these games is not trivial, involving besides an extensive documentation, own of computational systems, added to another documentary set related to the area of games, also it is necessary a range of knowledge in diverse areas of the computation like: Programming, database, computational graphic, human-computer interaction and others. In order to help developers in creating the specific system type, it is necessary to have many audio elements and great user interactivity, a set of tools have been created that support the whole development, and at the heart of this set of sofwares we give *engine of games* name. We intend with this work to create an electronic educational game that serves as a learning object more interesting to the student, potentializing as chances of it to acquire the knowledge that the teacher wishes teach, for this, we will virtualize a didactic sequence used as a game, Already tested and validated in municipal schools in the city of Vitória da Conquista, called Fazendinha Matemática, this is a transmission aimed at teaching and learning the operations of mathematics. In development we will use the famous Unity3D engine, showing every aspect of the development since the design phase, documentation, until the programming itself and the linking of visual audio elements in a detailed way, so that this work serves as a reference for other developers looking for the use of this tool in future game projects. We obtain at the end of this work a game in beta phase that was evaluated positively by teachers in more than one phase of its development.
Keywords: Unity, Games Development, Educational Games

1 Introdução

1.1 Motivação

A educação é uma das áreas mais importantes da vida de qualquer pessoa e a escola é o ambiente que maior personifica essa entidade social. Porém, em pleno século XXI, vemos o quão pouco a escola se renova perante a nova sociedade que se apresenta, e a velha fórmula tem cada vez mais mostrado suas falhas. A educação vem, ao longo dos tempos, sendo alvo de intensos debates e discussões. O sistema de ensino, diante dos dados revelados por pesquisas nacionais (SAEB, ENEN, INAF) e internacionais (PISA), vem sendo criticado em razão do baixo nível de qualidade apresentado (LACANALLO *et al.*, 2007). O que se percebe é que as gerações mudaram e possuem novas formas de ver o mundo, novas formas de se relacionar, novos objetivos e novas formas de aprendizagem.

Os últimos cem anos foram palco de diversas mudanças no mundo. Com duas grandes guerras e uma série de revoluções, tivemos uma alteração drástica no nosso modo de vida, e um dos principais causadores foram os recursos tecnológicos que se tornaram indispensáveis na sociedade e causaram mudança em todos os setores, modificando a forma de nos comunicarmos e nos relacionarmos uns com os outros. “Estas mudanças tem evidenciado a necessidade de repensar os ambientes educacionais: a escola pode se beneficiar da utilização das tecnologias digitais com novas abordagens para o ensino e aprendizagem, numa concepção mais ativa” (PINTO e BOTELHO, 2012, p. 135).

Apesar de ainda serem escassos os equipamentos tecnológicos dentro das salas de aula brasileiras, e de haver pesquisas como a de Ramos (2012) sobre o uso dos equipamentos próprios dos alunos (celulares, tocadores de música, entre outros) com uma finalidade pedagógica, segundo dados da Pesquisa TIC Educação de 2013, analisada por Santana (2014), os professores, mesmos sem esses aparatos tecnológicos dentro da sala, já se utilizam dos recursos tecnológicos para realizarem suas aulas, seja por meio de imagens, vídeos, textos, questões, jogos, programas educacionais ou *podcasts* obtidos da internet. Um destes recursos que cada vez mais ganha espaço é o jogo eletrônico educacional. Pinto e Botelho (2012) nos diz que o lúdico tem sido realmente uma possibilidade na educação, pois permite uma manifestação do potencial criativo, além de contribuir para o desenvolvimento do raciocínio e

melhorar aspectos sociais, pessoais e culturais de seus jogadores. Este é um recurso poderoso que é bastante familiar aos jovens atuais e uma ferramenta que pode auxiliar os professores a chegar mais próximo de seus alunos, além de desenvolver habilidades que a mera exposição de conteúdo não consegue chegar. “Seu uso nas atividades educativas demonstra uma clara percepção da natureza lúdica do ser humano” (PIETRUNCHINSKI *et al*, 2011, p. 477).

Para a criação desses jogos eletrônicos normalmente são reunidos profissionais em diversas áreas por conta da complexidade desses sistemas. Profissionais de artes são requisitados para desenvolver os desenhos que dão vida a personagens, cenários, menus, botões e toda sorte de elementos gráficos do jogo. Profissionais de música dão a ambientação sonora aos diversos cenários, fazendo com que o jogador tenha uma imersividade além da visual. Os roteiristas e *level-designers* criam a história e o fluxo do jogo, definindo cenários, personagens e objetivos. E por fim temos os programadores que dão vida e unem todos esses elementos em torno de um único produto, e para auxiliar esses profissionais a reunir as ferramentas necessárias que criam os jogos eletrônicos foram criadas as *engines de games*, que são verdadeiros motores de jogos, reunindo toda a criação em um único ponto em que vários profissionais podem trabalhar de forma integrada, e podem ser definidas como “um programa para computador ou um conjunto de bibliotecas capazes de juntar e construir todos os elementos de um jogo em tempo real” (DIAS, 2017).

Dentre as *engines* pesquisadas, a Unity3D foi a que melhor atendia as nossas necessidades visto que: Possui licença gratuita; É multiplataforma, de modo que não precisaremos fazer um novo código caso queiramos criar versões para Linux, Android e Web, por exemplo; Trabalha com a linguagem C#, sendo a mesma bastante difundida hoje e bastante parecida com a linguagem Java, diminuindo assim a curva de aprendizagem; Possui ampla documentação em seu site oficial, além de ser bem didática; Possui uma comunidade extremamente ativa, sendo hoje uma das *engines* mais utilizadas no mercado *indie*; E é de fácil manipulação para recursos áudio visual. Todas essas características nos levaram a eleger o Unity como nossa plataforma de desenvolvimento, aliado a outras ferramentas como o *MonoDevelop,* o *Paint*, o *Photoscape*, o *Adobe Flash*, e sites de edição de som e música como o (ONLINE IMAGE EDITOR, 2017) e o (MP3 CUT, 2017).

1.2 Objetivos

Este trabalho tem como objetivo a criação de um jogo eletrônico educacional com a *engine* de jogo Unity3D. Especificaremos todas as etapas de concepção do jogo confeccionando os diversos artefatos necessários para a documentação do mesmo, como o *Game Design Document*, ou GDD. Pretendemos também que este material sirva como base para outros projetos de jogos usando o Unity3D, por isso iremos especificar detalhadamente os recursos utilizados e todo o planejamento realizado.

O jogo confeccionado é baseado em uma sequência didática chamada Fazendinha Matemática (DIAS e GUSMÃO, 2015; DIAS, GUSMÃO, FREDINI e DE MOURA, 2015), que tem por meta auxiliar no ensino-aprendizagem das operações básicas da Matemática (adição, subtração, multiplicação e divisão) para alunos entre cinco e oito anos, e pretendemos que essa versão digital da sequência seja ainda mais efetiva na aprendizagem do que a sequência original, aproveitando-nos de toda a imersividade e vantagens de manipulação de informação que os jogos eletrônicos possuem. O jogo foi feito com base no estilo *point-and-click*, onde as imagens são em 2D e o *mouse* é o dispositivo mais utilizado.

Este trabalho também pretende ter uma avaliação constante tanto interna quanto externamente, pois além do autor desta monografia que é da área de tecnologia, participam também do projeto alunos das áreas de pedagogia e matemática, além da autora original do projeto que é de ambas as áreas, assim o projeto terá o suporte didático necessário para que não seja algo puramente lúdico, mas que ainda consiga manter a atenção dos seus jogadores. Para a avaliação externa buscaremos fazer reuniões com outros professores tanto durante o desenvolvimento quanto após o mesmo, para que o projeto seja desenvolvido com base naqueles que o utilizarão.

1.3 Estrutura do Trabalho

Este trabalho esta estruturado da seguinte maneira: No segundo capítulo explanaremos sobre o referencial teórico utilizado, detalhando a inserção tecnológica no meio educacional, as vantagens que os jogos possuem como objetos de aprendizagem e as várias ferramentas que o Unity3D nos oferece. No terceiro capítulo falaremos sobre jogos feitos em Unity3D e jogos educacionais feitos em outras plataformas, mostrando as diferenças no uso dos recursos.

No quarto capítulo discorreremos sobre o nosso trabalho, demonstrando em detalhes as ferramentas que utilizamos. No capítulo seguinte nós abordaremos os resultados que obtivemos a partir do nosso jogo e falaremos dos nossos projetos futuros para o jogo.

2 Referencial Teórico

2.1 Educação no Século XXI

Não é segredo que a qualidade na educação do Brasil não vai bem. Em estudo da SAEB (Sistema de Avaliação da Educação Básica) de 2015, como mostrado na Figura 01, analisado por Saldaña (2016), o aprendizado de matemática no 5º ano do Ensino Fundamental I, no 9º ano do Ensino Fundamental II e no 3º ano do Ensino Médio não atinge os níveis adequados, estipulados pelo movimento Todos Pela Educação, para que os alunos tenham os conhecimentos básicos na etapa em questão.

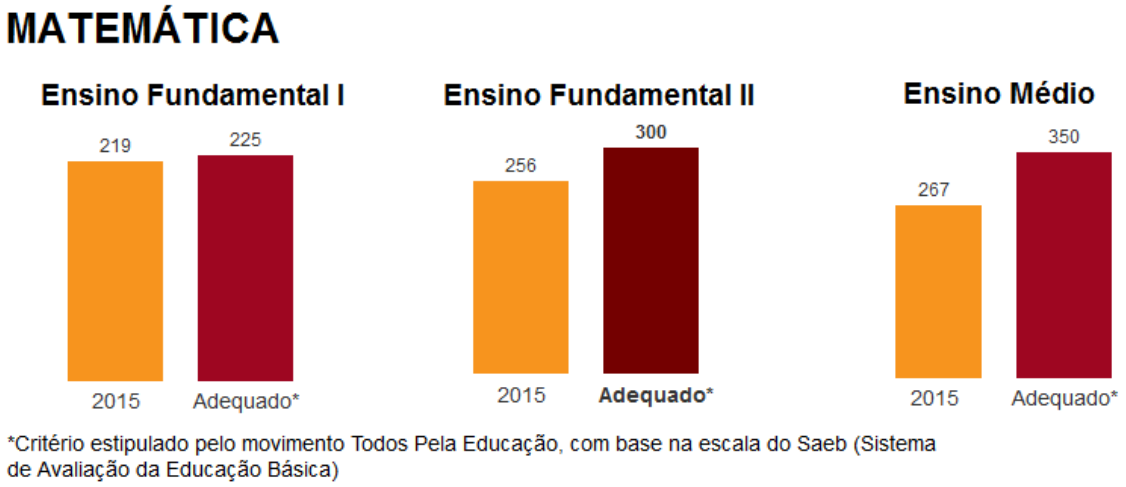


Figura 1 – Resultado das avaliações do Ideb 2015 comparado aos critérios estipulados pelo movimento Todos pela Educação . Fonte: (SALDAÑA, 2016)

Estes índices mostram que temos um *déficit* muito grande em nosso sistema educacional. Em comparação aos outros países estamos em ainda pior situação, ocupando as últimas posições nos *rankings* do PISA (Programa Internacional de Avaliação de Estudantes, traduzido), sendo esta a prova mais importante de avaliação de aprendizagem, organizada pela OECD (Organização para Cooperação e Desenvolvimento Econômico, traduzido), com 64 países participantes, abarcando provas nas áreas de Matemática, Leitura e Ciências, para alunos com 15 anos, matriculados a partir do 8º ano do ensino fundamental.

Além de diversos fatores que são conhecidos, como falta de infraestrutura nas escolas públicas, baixa renda entre muitos alunos e inicio da vida no mercado de trabalho muito cedo, ainda temos outro fator importante que é a falta de interesse pela escola. Em pesquisa realizada pelo Relatório Educação para Todos no Brasil 2000-2015, disponibilizado pelo Ministério da Educação, é possível observar (Figura 02) a queda no número de matrículas no Ensino Médio (de 9,07 milhões para 8,31 milhões em 10 anos).

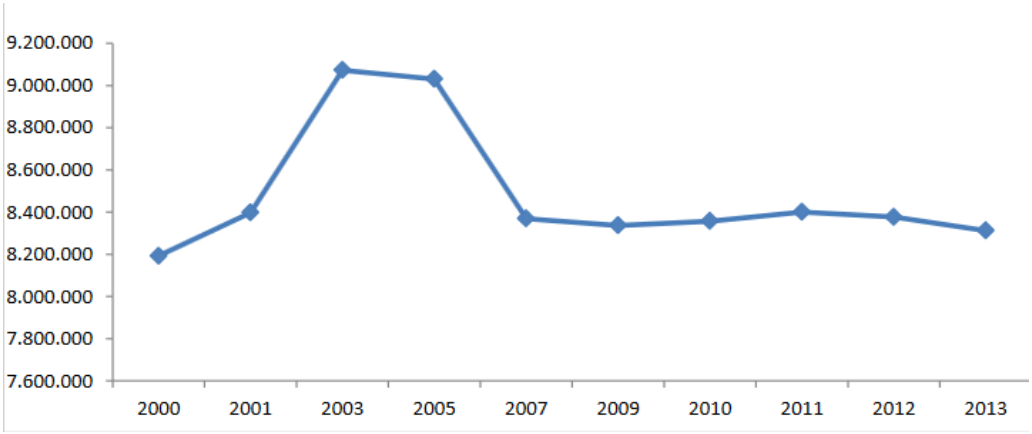


Figura 2 - Matrículas no Ensino Médio (2000 – 2013). Fonte: INEP, Censos Escolares

Nos estudos feitos pela Fundação Victor Civita (2013) também é possível identificar uma alta taxa de evasão que foi de 7,4% em 1999 para 16,2% em 2011 (Figura 03).

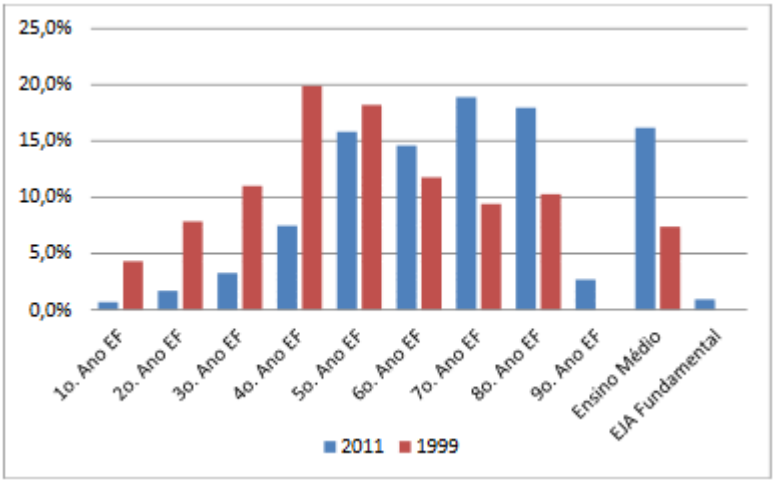
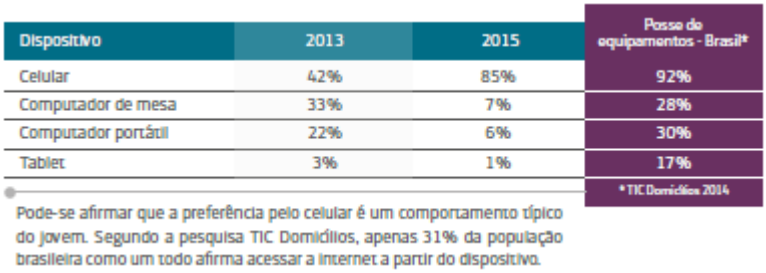


Figura 3 - Taxa de Evasão 1999-2011. Fonte: IBGE, PNAD 1999 e 2011

No estudo também é possível constatar que a maioria dos estudantes acha as disciplinas e conteúdos muito descontextualizados da vida prática, onde 36% dos alunos acham matérias como história, geografia, biologia e física como descartáveis e apenas 19,1%

dos jovens consideram o conteúdo de literatura como algo útil. A pesquisa também mostrou que apesar dos estudantes, mesmo com baixa renda, estarem conectados às novas tecnologias (70,7% dos mesmos tendo acesso a internet em casa), as escolas parecem não estar interessadas em utilizar esse recurso para manter o interesse do aluno, onde 73,8% dos entrevistados sendo de escolas equipadas com computadores, porém 37,2% destes afirmaram nunca terem usado os mesmos.

O fato é que o mundo mudou. Estamos na era em que crianças já crescem com um *tablet* na mão. De acordo a um estudo feito pela Fundação Telefônica Vivo (2016) com 1440 jovens entre 16 e 24 anos, 85% deles acessam a internet pelo celular (Figura 04), 54% usam a internet para se preparar para provas como o ENEM, vestibulares e concursos, e 99% usam o WhatsApp para se comunicar.



Dispositivo	2013	2015	Posse de equipamentos - Brasil*
Celular	42%	85%	92%
Computador de mesa	33%	7%	28%
Computador portátil	22%	6%	30%
Tablet	3%	1%	17%
			* TIC Domicílios 2014

Pode-se afirmar que a preferência pelo celular é um comportamento típico do jovem. Segundo a pesquisa TIC Domicílios, apenas 31% da população brasileira como um todo afirma acessar a internet a partir do dispositivo.

Figura 4 - Uso de maior frequência de acesso a internet. Fonte: Fundação Telefônica Vivo, 2016

São jovens que “falam” com naturalidade, sem “sotaque”, o idioma digital destes recursos eletrônicos, como se fosse a língua materna deles (CARNIELLO, RODRIGUES E MORAES, 2010). Marc Prensky (2001) denominou-os de nativos digitais, é a primeira geração que já nasceu imersa em tecnologia, por conta disso a relação dos mesmos com a tecnologia muda completamente a forma com que se comunicam, se divertem e se informam, em comparação às gerações anteriores, se refletindo também na forma como aprendem. “Os nativos digitais são acostumados a aprender rápido, fazem conexões randômicas, processam visualmente a informação dinâmica e aprendem através de atividade baseada em jogos” (PRENSKY *apud* LEMOS, 2009). Já Tapscott (2010) utiliza o termo geração digital ou Geração Internet, e nos fala sobre oito normas dessa geração que os diferenciam das demais gerações, sendo elas: a liberdade, a customização, o escrutínio, a integridade, a colaboração, o entretenimento, a velocidade e a inovação. Os professores, no entanto fazem parte dos chamados imigrantes digitais, pessoas nascidas antes dessa era, mas que tentam se adaptar as novas tecnologias. Segundo esses autores, os professores aprenderam a encarar os textos como fonte de informação e as imagens como fontes secundárias, e querem repassar esta

metodologia aos nativos que crescem “lendo” imagens em diversas telas, preferindo estas aos textos. Imigrantes são atraídos pela sistematização linear e ordenada das atividades, enquanto os nativos escolhem a agitação e a aparente inconstância dos hipertextos. “Os nativos usam *games* para promover aprendizado, já os imigrantes além de nem cogitarem esta possibilidade em sua maioria, censuram e condenam esta prática” (Carniello, Rodrigues e Moraes, 2010, p. 7). Por isso, para que tenhamos um ensino mais produtivo e uma aprendizagem mais efetiva, temos de sair do triangulo de mediação (composto por professor-saber-aluno) e passarmos para uma espiral de conhecimento em que o aprendizado é multidirecional e horizontal entre o professor, aluno, saber, tecnologias digitais, etc.

2.2 Jogos Eletrônicos

Os jogos são uma das formas mais antigas dos seres humanos se relacionarem e se divertirem. Ele é inerente à vida humana, necessário para o desenvolvimento das crianças e para o entretenimento geral das pessoas (DUFLOS, 1999). Ao longo da história dos jogos, muitas definições e conceituações sobre o que é um jogo surgiram, dentre elas podemos destacar a de Schell (2011), de que um jogo é uma **atividade de solução de problemas, encarada de uma forma lúdica**. Ele também faz análise de outras definições e retira dez qualidades (Figura 05) que os jogos possuem.

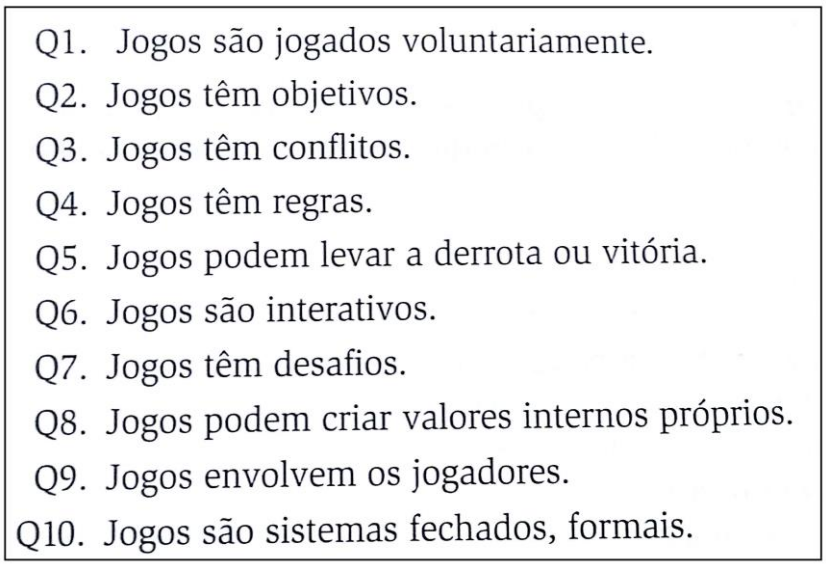


Figura 5 - Lista de qualidade dos Jogos. Fonte SHELL, 2011

Com o advento das tecnologias digitais, surgiram os jogos digitais (ou eletrônicos) que podem ser encontrados em diferentes plataformas (computadores de mesa, *notebooks*, *smartphones*) e jogados por todas as pessoas. Lucchese e Ribeiro (2012) analisam as diferenças que há entre os jogos eletrônicos e os jogos tradicionais, destacando nos primeiros a imersividade que o mundo do jogo cria nos jogadores por conta das imagens, sons e às vezes até reações táteis (controles com vibração, por exemplo), algo que os jogos tradicionais conseguem apenas através da imaginação. Outra característica destacada é a rigidez das regras nos jogos eletrônicos, que ao contrário dos jogos tradicionais, não pode se flexibilizar tão facilmente por conta das regras codificadas algoritmicamente. Crawford(1982) aponta quatro conceitos fundamentais para os jogos:

- **Representação**: O jogo eletrônico traz uma imersividade imensa por conta da combinação de recursos de áudio e vídeo, parecido com o que traz os filmes, porém com o acréscimo de possuir a interatividade do jogador com o jogo.
- **Interação**: A interação nos jogos pode se dar das mais variadas formas, principalmente por meio de comandos de teclado, *mouse*, controle, e hoje também por aparelhos como *kinect*. Ela é fortemente ligada às regras do jogo, podendo ser em tempo real ou não.
- **Conflito**: Os conflitos são criados por agentes que respondem às ações do jogador. Esses agentes possuem algum tipo de inteligência e representam um obstáculo aos objetivos do jogador.
- **Segurança**: Essa característica dos jogos digitais permite que os jogadores passem por enormes “riscos” sem estarem necessariamente em perigo, como por exemplo: Levar um tiro, capotar um carro, se afogar, entre outros. Ele pode fazer diversas atividades que na vida real seriam bem problemáticas de se realizar, sentir as sensações (de forma mais atenuada) que tais atividades podem acarretar, e mesmo assim terem a certeza que nada de mal ocorrerá a eles na vida real.

2.2.1 Estatísticas sobre Jogos Eletrônicos

Os jogos eletrônicos tem tido um crescimento sem igual nos últimos anos. Um relatório produzido pela Newzoo (2016) mostra que o mercado de jogos alcançou lucro de 99,6 bilhões de dólares em todo o mundo, sendo que quase um terço desse valor é dos jogos para computador. O Brasil ficou em 12º lugar, gerando um lucro de 1,27 bilhões de dólares nesse setor. Outra pesquisa, dessa vez com foco no Brasil, realizada pela Game Brasil (2016),

em parceria com a SIOUX, com a Blend e com a ESPM, onde participaram 2848 pessoas entre 14 e 84 anos de todas as regiões do país, mostra (Figura 06) que a maioria das pessoas costuma jogar alguma coisa, sendo que apenas 11,7% dos entrevistados disseram não jogarem nada.

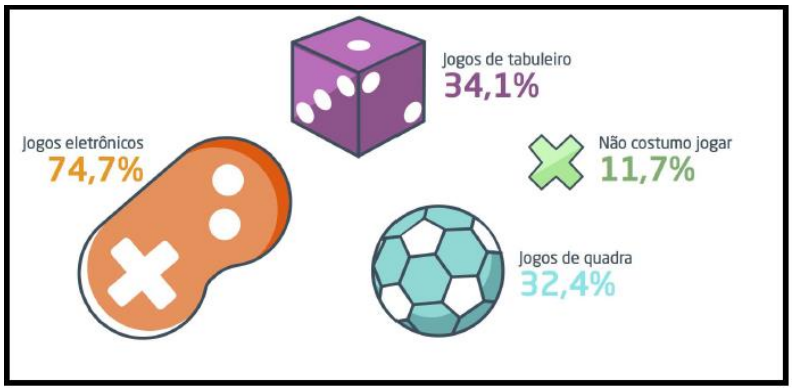


Figura 6 - Costume de jogar algum tipo de jogo. Fonte: Game Brasil (2016)

A pesquisa também levanta outros dados interessantes como que pouco mais da metade dos jogadores são do sexo feminino (Figura 07), está na faixa etária de 25 a 34 anos (Figura 08) e jogam em mais de uma plataforma (Figura 09).

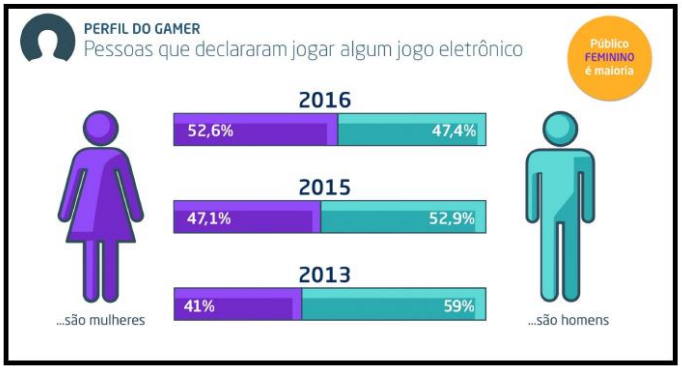


Figura 7 - Jogadores por sexo. Fonte: Game Brasil (2016)

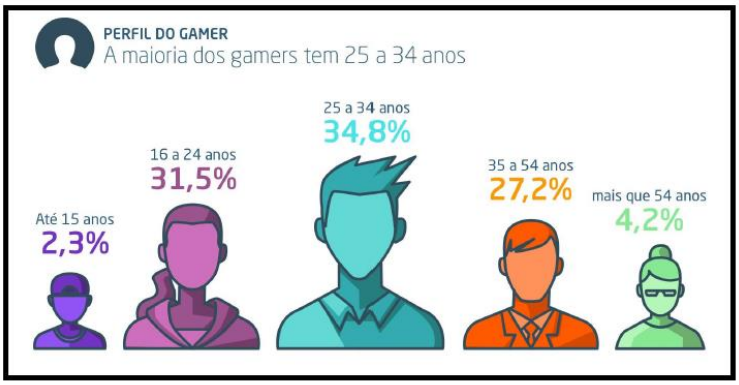


Figura 8 - Jogadores por idade. Fonte: Game Brasil (2016)

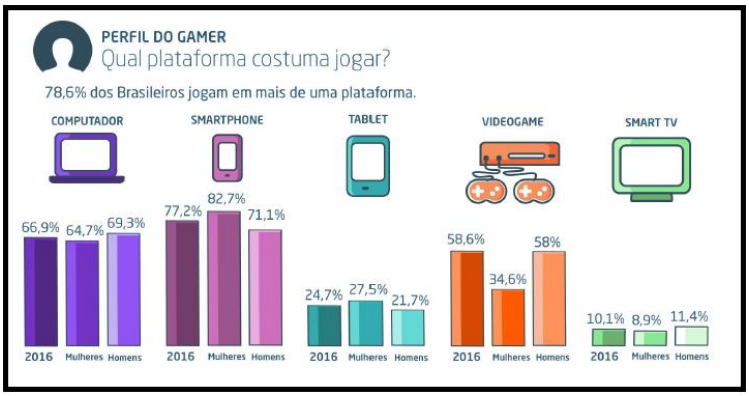


Figura 9 - Jogadores e plataformas usadas. Fonte: Game Brasil (2016)

Com esses dados é possível afirmar que jogar não é mais uma atividade exclusivamente infantil e masculina, adultos, jovens e pessoas de mais idade, assim como as mulheres, exploram cada vez mais os jogos para passar o tempo.

De Oliveira Medeiros e Schimiguel (2012) analisam quatro características levantadas por Jonh Huizinga (2000), um dos autores mais referenciados ao se tratar de jogos, em que o mesmo indica os motivos pelos quais os jogos são tão apreciados por pessoas de todas as idades:

- **Liberdade**: Significa que os jogos devem ser jogados de forma livre e espontânea por parte dos jogadores. Quando ela é feita obrigatoriamente, perde o caráter lúdico e prazeroso.
- **Não é a vida real**: Outra característica é que o jogo é um escape da "vida real", ele pertence ao mundo imaginário e somente a ele, nos absorvendo completamente.
- **Possui limitação de tempo e lugar**: O jogo é jogado em um determinado limite de tempo e em um determinado lugar, as regras são absolutas dentro desses limites. No jogo é possível tanto parar e continuar em outra hora, quanto começar tudo novamente.
- **Cria ordem e é ordem**: No espaço/tempo do jogo suas regras são absolutas, elas criam uma "perfeição temporária" no caos da vida real. Nenhum dos jogadores pode desobedecer tais regras para que o jogo seja "bem jogado".

2.2.2 Jogos Eletrônicos e a Educação

Como pudemos observar nas seções anteriores o jogo está intrinsecamente ligado ao homem e os jogos eletrônicos têm ganhado cada vez mais espaço na vida diária das pessoas, esta é uma tendência que não deve ser ignorada, principalmente pelos espaços educacionais que além de necessitarem buscar o interesse dos jovens, podem se aproveitar das várias características positivas que os jogos trazem aos jogadores.

"O jogo exige do jogador modalidades estruturais de inteligência para relacionar regras, conteúdo, estratégias, e esquemas" (BATISTA, NOVAES e FARBIARZ, 2009, p. 1). De acordo com Mattar (2010) e Prensky (2006), entre as várias habilidades desenvolvidas pelos jogos estão: a capacidade de aprender de forma rápida; facilidade para trabalhar em grupo; iniciativa, atitude e criatividade; capacidade de resolução de problemas e tomada de decisões mesmo com pouca informação disponível. Além disso, Wang (2005) destaca que os jogos eletrônicos possuem um melhor ambiente de aprendizagem, pois ajustam o nível de dificuldade de acordo as habilidades do jogador, proveem aos jogadores um *feedback* claro e imediato, dá escolhas e controle sobre suas ações, além de despertarem a curiosidade e a fantasia.

Logo é possível ver que os jogos não são simples perda de tempo, e se bem direcionados, podem trazer enormes benefícios para aqueles que o jogam. Batista, Novaes e Farbiarz (2009) vêm nos trazer as declarações do Dr. James Clarence Rosser Jr., chefe de cirurgia minimamente invasiva (videolaparoscopia) do Beth Israel Medical Center em Manhattan, que afirma que suas várias horas jogando videogames o ajudaram em sua excelente habilidade cirúrgica, tendo inclusive um videogame próximo à sala de cirurgia para aquecer suas habilidades antes de uma cirurgia.

Os jogos educacionais têm como finalidade potencializar a vontade dos alunos na criação de novos conhecimentos, além de mostrar novas formas de demonstrar o conteúdo que não a forma expositiva comum. Dohme (2008) aponta uma grande característica do jogo como ferramenta educacional, pois enquanto os alunos vêm os jogos como fim, voltados à diversão que o mesmo irá lhes proporcionar, os professores podem vê-los como "meio, um veículo que permite transmitir uma mensagem educativa de forma atraente e prazerosa, cabe ao professor escolher o jogo que melhor se aplica ao conteúdo que deseja" (MEDEIROS; SCHIMIGUEL, 2012, p. 3).

2.3 Desenvolvimento de Jogos Eletrônicos

O desenvolvimento de jogos eletrônicos é uma das áreas da computação que mais abrange diferentes especialidades e considerando o aspecto computacional tais aplicações requerem a adoção de sofisticadas técnicas que na maioria das vezes representam o "estado da arte" das pesquisas em Ciência da Computação principalmente as pesquisas relacionadas com Análise de Algoritmos, Computação Gráfica, Redes de Computadores e Inteligência Artificial. Para Laird e Van Lent (2001), os jogos computadorizados podem ser considerados uma *killer application* da computação, ou seja, uma aplicação modelo justamente pelo fato de possuir problemas significativos que ao serem solucionados irá impactar em outras aplicações (CLUA e BITTENCOURT, 2005).

Logo vemos a importância de um estudo abrangente pelas mais diversas áreas da computação na produção de bons jogos. Entre as subáreas, destacaremos aqui algumas das mais relevantes, mostrando os principais pontos a serem observados nas mesmas.

2.3.1 Programação

Começaremos pela programação, sendo esta o coração de qualquer sistema por tratar justamente da codificação de forma eficiente e organizada. Divério e Menezes (2000, p. 35) definem programa como "um conjunto estruturado de instruções que capacitam uma máquina a aplicar sucessivamente certas operações básicas e testes sobre os dados iniciais fornecidos, com o objetivo de transformar estes dados". A programação de jogos segue esses mesmos procedimentos e cuidados, devendo o programador estar atento às mesmas estruturas e paradigmas de uma aplicação comercial. Rabin (2013) cita algumas estruturas importantes na programação de jogos como as **listas conectadas**, frisando que tudo acaba em uma lista em um ponto ou outro: entidades do jogo em um mundo, projéteis no ar, jogadores no jogo, itens no inventário e assim por diante; e os **dicionários de dados**, lembrando que nos jogos as entidades estão constantemente interagindo umas com as outras: enviam mensagens para outras entidades, criam novas entidades e estão à procura de outras entidades.

Outro conceito importante da programação é o da orientação a objetos. Tendo sido iniciada após o início dos anos 90, ela é caracterizada pela junção de dados com o código. "A chave para a programação orientada a objeto é que os dados e o código são tratados como

uma unidade, em comparação aos dados sendo apenas uma consequência da execução do código, como no caso da programação procedural" (RABIN, 2013, p.213).

Os padrões de projeto (*Design Patterns*) são outro aspecto importante na programação de qualquer sistema computacional (os jogos eletrônicos incluídos), pois solucionam problemas comuns a um programa de uma forma limpa e organizada. Eles foram criados a partir da experiência de vários programadores ao se depararem com um mesmo tipo de problema diversas vezes, e foram reunidos pela *Gang of Four* (GOF), formada por Eric Gamma, Richard Helm, Ralph Johnson e John Vlissides, entre 1991 e 1994, no livro *Design patterns: elements of reusable object-oriented software*.

Dentre os principais padrões reunidos pelo GOF, alguns atendem bem às necessidades dos jogos, são eles: o **Singleton**, que é usado para classes que precisam ter exatamente uma única instancia no programa e estarem disponíveis de forma global para todas as partes do sistema. No desenvolvimento de jogos ele normalmente é usado em recursos de hardware, log de erros, fila de mensagens e o objeto que representa o próprio jogo; o **Object Factory**, para a criação de um objeto que cria outros, pois podemos não saber no momento da instância, o tipo de objetos que precisaremos criar. Nos jogos esse padrão é crucial por conta da programação de jogos ser direcionada a dados, ele ainda fala que o padrão ajuda na extensibilidade do código, fazendo com que novos tipos possam ser criados sem a alteração na criação do objeto; o **Observer** trata de um objeto (sujeito) notificar um determinado evento para outros objetos (observadores) que necessitam realizar alguma ação em razão desse evento. Na programação de jogos também é um padrão muitíssimo útil, por conta da extensa interação entre os objetos (um cavaleiro que deixa sua espada cair, um personagem que pisa em um alçapão, uma heroína que abre um baú, etc.), todos eles realizam eventos que outros objetos possuem interesse, e a codificação usando esse padrão torna as possíveis alterações e também a legibilidade mais limpa; e o **Composite**, que serve para fazer operações sobre um conjunto de objetos diferentes a partir de uma classe pai (por herança). Nos jogos podemos citar os elementos de menus e os elementos do *head-up* display (HUD), quer dizer a representação dos objetos do jogo, como vida, magia, mapas, armas etc. Seus nomes e valor variam de acordo com o jogo.

2.3.2 Computação Gráfica

Outra área importante da computação no desenvolvimento de jogos é a da computação gráfica sendo ela a responsável pela forma do jogo apresentada na tela, todos os personagens, cenários, itens, entre outros, sendo esta a parte mais pesada (sendo necessária muitas vezes uma placa de vídeo e processador gráfico a parte, apenas para rodar o jogo). Manssour e Cohen (2007, p. 1) definem a Computação gráfica como a área da Ciência da Computação "que se dedica ao estudo e desenvolvimento de técnicas e algoritmos para a geração (síntese) de imagens através do computador". Na programação de jogos 2D essa área se apresenta na disposição dos objetos em tela, nas relações de escala, altura e largura, na apresentação da ordem dos objetos (definindo qual irá se sobrepor ao outro), dentre outros detalhes. Já nos jogos tridimensionais a representação se torna mais realista, porém mais custosa em termos de processamento. O estudo de redução de polígonos é essencial para que o *game* não necessite de um *hardware* excessivamente potente, sendo os polígonos linhas e vértices que formam um objeto, personagem ou mapa. Quanto maior o número de polígonos, melhor a qualidade gráfica e maiores os requisitos de hardware para que o jogador possa jogar o game.

2.3.3 Banco de Dados

O banco de dados é um requisito importantíssimo, principalmente para os novos jogos. Armazenar os dados (seja em um arquivo, cartão de memória ou em um banco de dados interno) permite os jogadores poderem ter experiências mais duradouras e permite aos desenvolvedores construírem jogos para serem jogados por mais tempo. Segundo Elmasri e Navathe (2005), os **bancos de dados** podem ser definidos como uma coleção de dados relacionados. Já os **dados** podem ser definidos como fatos que podem ser gravados e que possuem significado implícito. Para facilitar a gerência desses dados é necessário um sistema a parte que é comumente chamado de Sistema Gerenciador de Banco de Dados (SGDB) que é uma coleção de programas que permite aos usuários criar e manter um banco de dados.

O uso puro de arquivos e outros tipos de armazenamento de memória, aos poucos vêm sendo abandonado principalmente por conta dos dados que os jogos representam terem se tornados maiores e mais complexos. Takai, Italiano e Ferreira (2005) citam algumas características importantes em relação às vantagens do uso de SGBDs, que são: **Controle de Redundância**, que impede que a informação esteja em dois lugares ao mesmo tempo,

causando inconsistências à informação; **Compartilhamento de Dados**, onde o sistema controla a concorrência de acesso aos dados, garantindo que qualquer processo de leitura/escrita seja sem erros; **Fornecimento de Múltiplas Interfaces**, sendo estas interfaces no modo de linguagem natural, acesso gráfico, em SQL ou via menus de acesso; **Reforçar restrições de integridade e fazer *backup* e restauração**, onde o SGBD impede que qualquer usuário (seja uma pessoa ou uma aplicação) possa realizar acessos que comprometam a integridades dos dados e deve apresentar facilidades na recuperação de falhas de *hardware* ou *software*.

2.3.4 Interação Humano-Computador

Além desses conceitos mais voltados para o sistema, temos uma área que visa o estudo da interação entre o usuário e o sistema: a Interação Humano-Computador (IHC). Essa área tem foco não apenas no projeto da interface, mas em todos os aspectos relacionados com interação entre usuários e sistema (PREECE *et al*, 1994). Estes aspectos incluem todas as ferramentas disponibilizadas ao usuário para que o mesmo tenha plena consciência das possiblidades do sistema, e também o *feedback* do estado do sistema, mostrado ao usuário de forma clara. Além disso, há requisitos de interação próprios de determinados sistemas, por exemplo, "em sistemas de apoio ao aprendizado o mais importante é saber se o aluno, usuário do sistema, consegue de fato utiliza-lo para aprender; ou em jogos o relevante é a sua propriedade de entreter e divertir os usuários" (PRATES e BARBOSA, 2007, p. 266).

Uma das principais preocupações ao se projetar interfaces interativas é com a qualidade de uso, e uma das suas propriedades mais importantes é a da **usabilidade**, que leva em consideração a facilidade e eficiência com a qual um usuário consegue utilizar um sistema (GOULD e LEWIS, 1985). Entre os fatores que devem ser levados em conta para se avaliar a usabilidade de um sistema, estão: facilidade de aprendizado, facilidade de uso, eficiência de uso e produtividade, satisfação do usuário, flexibilidade, utilidade e segurança no uso (PRATES e BARBOSA, 2007).

Um importante conceito dessa área é a Engenharia Semiótica que busca explicar os fenômenos envolvidos no *design*, no uso e na avaliação de um sistema interativo (ANDRADE, 2013). Em suma, ela vê a interface como um artefato de meta-comunicação, que envia e recebe mensagens do usuário e tem como papel a comunicar a funcionalidade da aplicação (o que a interface representa, que tipos de problema ela está preparada para

resolver) e o modelo de interação (como pode-se resolver um problema) (Souza, 1993 apud DE OLIVEIRA e BARANAUSKAS, 1998). Desta forma, os projetistas devem sempre se comunicar com os usuários e não apenas dizer como os botões funcionam, demonstrando assim porque a aplicação faz sentido para eles como também deveria fazer para os usuários (MATTOS, 2010). A Engenharia Semiótica trabalha com signos, que na visão de Peirce (2005) é tudo aquilo que significa algo para alguém, e em sistemas interativos podem ser divididos em três níveis: signos estáticos (que estimulam os usuários a interagir com o sistema), dinâmicos (que confirmam ou não a interação realizada pelo usuário) e os metalinguísticos (informam ou explicam os signos estáticos e dinâmicos).

Em um jogo eletrônico, independente da plataforma ou suporte, a interface deve ser desenhada tendo em vista o máximo de aproveitamento intuitivo por parte do jogador, além de funcional, e uma estética adequada ao tema e/ou grupo utilizador, não esquecendo a legibilidade e leiturabilidade. Esses conceitos significam que em jogos com boas interfaces há mensagens contidas, além de regras desenvolvidas, aliadas a informações só detectadas quando o jogador traz a experiência para o seu mundo (para o seu entendimento). Assim "o *design* da interface torna-se eficiente quando, além de funcionalidade, segurança, satisfação e produtividade, preocupa-se também com a estética com o fim não só de beleza, mas na conformação do jogador de assumir partes de um mundo que não é seu" (MACEDO FILHO, 2005, pp. 6-7).

Portanto, podemos ver o quão difícil é para uma única pessoa poder produzir um bom jogo. "A maioria dos jogos comerciais é criada por grandes equipes, desde centenas de programadores, dependendo do tamanho do grupo e do escopo do programa" (RABIN, 2013, p. 165). Porém as áreas de computação (apesar de extremamente importantes) não são as únicas para o desenvolvimento de um jogo. Como afirma Clua e Bittencourt (2005) os jogos, ao contrário de outros sistemas computacionais que devem apenas atender suas especificações de uma forma bem elaborada, precisam ser divertidos e agradáveis, pois seu principal objetivo é proporcionar entretenimento às pessoas. Os jogos computadorizados precisam criar a sensação de imersividade nos usuários, tal característica obtida pela combinação de aspectos artísticos e tecnológicos (BATTAIOLA; ELIAS; DOMINGUES, 2002). Para conseguir tal aspecto Clua e Bittencourt (2005) citam algumas áreas que ajudam na criação dos jogos: Educação, Psicologia, Artes Plásticas, Letras, *Design* Gráfico e Música. O estudo de tais áreas é comumente explorado pela etapa inicial do desenvolvimento de jogos, que é a de *Game Design*, que será detalhado na próxima seção.

2.4 Game Design

O processo de desenvolvimento de um jogo como o de qualquer outro sistema, deve começar pelo projeto. Como vimos nos tópicos anteriores há inúmeras questões a serem estudadas e decididas em relação ao desenvolvimento do *game* em seu contexto técnico, porém ainda precisamos desenvolver a parte principal que é a ideia do jogo, onde tudo se inicia. Basicamente o *design* de jogo é o ato de decidir o que um jogo deve ser, podemos colocar nessa lista a história do jogo, os cenários, os personagens, os desafios, o estilo de jogo, tudo isso e muito mais como estando previamente feito e decidido antes de se escrever a primeira linha de código.

Algo necessário a se saber antes de iniciar o desenvolvimento de jogos é que os mesmos possuem vários elementos. Schell (2011) elucida quatro elementos básicos (Figura 10) que guiarão toda a construção do jogo, são eles: Mecânica, Narrativa, Estética e Tecnologia.

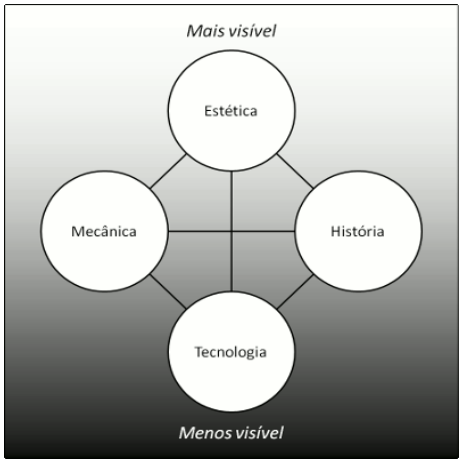


Figura 10 - Tétrade elementar dos jogos.
Fonte: Shell, 2011

Ele as define da seguinte maneira: **Mecânica**, esses são os procedimentos e as regras do seu jogo. A mecânica descreve o objetivo do seu jogo, como os jogadores podem ou não alcança-lo e o que acontece quando tentam; **Narrativa**, essa é a sequencia de eventos que se desdobram no seu jogo. Ela pode ser linear e previamente determinada (fechada), ou ramificada e emergente (aberta); **Estética**, isso tem a ver com aparência, sonos, cheiros,

sabores e sensações do seu jogo. A estética é um aspecto extremamente importante do *design* de jogos, uma vez que tem o relacionamento mais direto com a experiência de um jogador; **Tecnologia**, não estamos nos referindo exclusivamente à tecnologia "sofisticada" aqui, mas a quaisquer materiais e interações que tornem o jogo possível, como papel e lápis, peças de plástico ou *lasers* de alta potência.

2.4.1 Papel do *game designer*

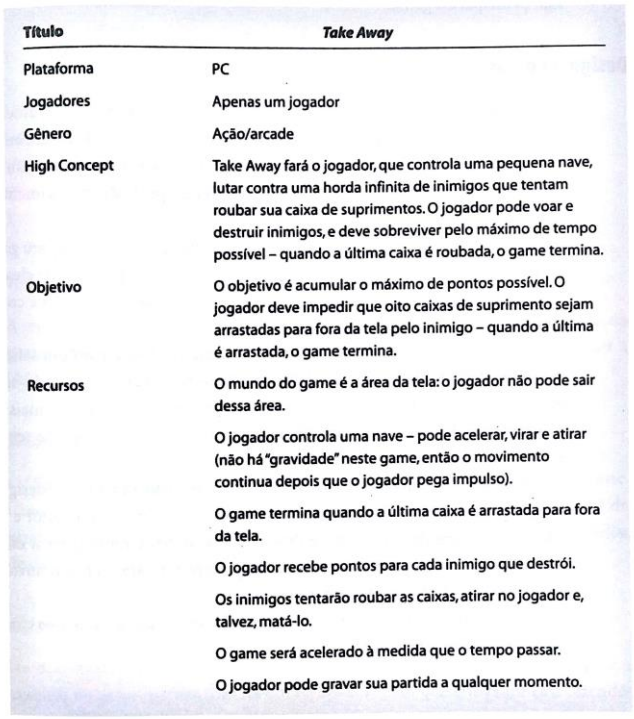
O *game designer* é o profissional responsável pela criação dos jogos. Normalmente é uma atividade atribuída a uma pessoa ou grupo específico dentro de uma empresa, onde as decisões mais importantes e de alto nível sobre os jogos serão tomadas. Por terem um papel tão grande no processo de criação dos jogos, esses profissionais requerem ter também as mais diversificadas habilidades, pois precisam fazer com que os outros entendam o que ele deseja criar. Logo os conhecimentos tecnológicos, hoje em dia, são importantíssimos para o *designer* que deverá ter noção das várias possibilidades que podem ser criadas para trazer aquela experiência que ele imaginou.

2.4.2 Documentação envolvida

Para estruturar o *design* precisamos primeiro organizar as ideias, e isso cria a necessidade de se fazer um documento de *design* é a planta baixa verbal de um jogo. Este documento pode estar tanto na forma física quanto na digital, podendo ser um pequeno texto com alguns tópicos até grandes *Wikis* bem detalhados. Ele define uma estrutura bem elaborada em relação aos documentos que devem conter um *game design*, e exploraremos uma parte da mesma nesse texto.

Começando pelo ***High Concept***, que é uma setença simples – ou duas – que descreve a essência de um game (SCHUYTEMA, 2008). Ou seja, ele descreverá de forma geral o que será aquele jogo, resumindo qual a ideia principal e o que se espera. O *High Concept* tem a função de delimitar e validar o jogo para os desenvolvedores. O primeiro documento a ser trabalhado (que na verdade é uma expansão do *High Concept*) é o **Escopo** (Figura 11), dando a ideia dos principais aspectos do game. O escopo pode ser dividido em: Plataforma (PC, PlayStation, Xbox, etc); Jogadores (*multiplayer* ou *single player*); Gênero (Ação, Aventura,

Tiro em Primeira Pessoa, Estratégia, etc.); O próprio *High Concept*; Objetivos do jogo, indicando o que o jogador tem que fazer para progredir e vencer no jogo; e Recursos, que resume todas as atividades e delimitações do jogo.



Título	*Take Away*
Plataforma	PC
Jogadores	Apenas um jogador
Gênero	Ação/arcade
High Concept	Take Away fará o jogador, que controla uma pequena nave, lutar contra uma horda infinita de inimigos que tentam roubar sua caixa de suprimentos. O jogador pode voar e destruir inimigos, e deve sobreviver pelo máximo de tempo possível – quando a última caixa é roubada, o game termina.
Objetivo	O objetivo é acumular o máximo de pontos possível. O jogador deve impedir que oito caixas de suprimento sejam arrastadas para fora da tela pelo inimigo – quando a última é arrastada, o game termina.
Recursos	O mundo do game é a área da tela: o jogador não pode sair dessa área. O jogador controla uma nave – pode acelerar, virar e atirar (não há "gravidade" neste game, então o movimento continua depois que o jogador pega impulso). O game termina quando a última caixa é arrastada para fora da tela. O jogador recebe pontos para cada inimigo que destrói. Os inimigos tentarão roubar as caixas, atirar no jogador e, talvez, matá-lo. O game será acelerado à medida que o tempo passar. O jogador pode gravar sua partida a qualquer momento.

Figura 11 - Exemplo de Escopo (jogo Take Away, desenvolvido para o livro Game development with Lua, da Charles River Media). Fonte: Schuytema, 2008

O segundo documento a ser apresentado é o ***Game Design Document*** (GDD, ou Documento de Design do Game), ele é o coração e a alma de todos os documentos que giram em torno de um game em desenvolvimento. Ele descreve, na medida do possível, todos os aspectos que o jogo deverá ter de forma detalhada. O documento poderá ter várias formas e estruturas e varia de *designer* para *designer*. Como mostrado na Figura 12, utilizaremos a estrutura de Schuytema (2008), em que ele busca generalizar o máximo possível todos os aspectos necessários em um jogo.

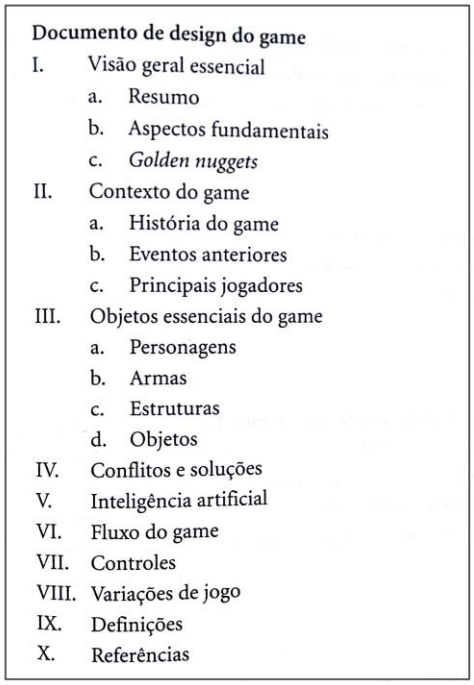


Figura 12 - Estrutura do GDD, por Paul Schuytema

Com cada um desses aspectos contemplados as equipes de desenvolvimento conseguem ter uma base satisfatória para poder desenvolver o *game*.

O terceiro documento a ser desenvolvido é o de **Dados do game**, "este documento existe para oferecer um local para documentar todas as informações específicas de *design* e do game sobre itens específicos" (SCHUYTEMA, 2008, p. 106). Estas informações normalmente são armazenadas em arquivos separados como os de Excel (que podem ser facilmente importados para o jogo), ou no próprio banco de dados.

O próximo documento é o **Design de ferramentas** que inclui todas as ferramentas externas que os desenvolvedores utilizam para desenvolver os recursos do jogo, como um modelador 3D, um programa de nivelação de água, ou mesmo um editor de imagens simples. E o último documento que descreveremos é a **Lista de recursos** que, como o próprio nome diz, é uma lista dos recursos utilizados no jogo como roteiros, imagens, sons, modelos 3D animados, entre outros. Lembrando que os documentos descritos aqui não representam o todo

do Game Design, podendo o mesmo ser bem mais complexo ou mais simples, variando conforme o jogo a ser desenvolvido e a equipe desenvolvedora.

2.5 Engine de Jogos

Como vimos nos tópicos anteriores, os jogos são sistemas complexos que reúnem diversos conhecimentos e exigem uma estrutura bastante sólida. Para o desenvolvimento destes sistemas poderíamos, tecnicamente, utilizar arquivos e o Bloco de Notas do Windows para fazer *scripts* de programação e executar nossos jogos, porém percebe-se facilmente que seria um trabalho hercúleo tal estratégia. Por isso que os desenvolvedores normalmente se utilizam de IDEs (*Integrated Development Environment*, ou, Ambiente de Desenvolvimento Integrado) para integrar diversas ferramentas que auxiliam no aumento de produtividade e organização. Segundo Reis (2009), o motor é responsável pelo funcionamento interno do jogo, retirando a complexidade de quem está desenvolvendo o game, proporcionando recursos que sejam reutilizáveis para a maior quantidade de jogos ele também nos diz que essa complexidade envolve entender-se com o *hardware* gráfico, controlar os modelos para serem renderizados, tratar da situação física e sonora, tratar das entradas de dados do jogador, tratar de todo o processamento de baixo nível, entre outras coisas. Na Figura 13 é mostrada uma forma de estrutura que as *engines* devem conter.

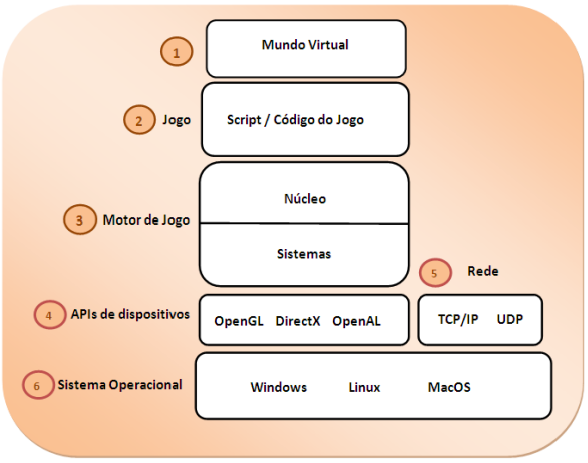


Figura 13 - Diagrama de um motor de jogo genérico.
Fonte Reis, 2009

Começando de cima para baixo, o **mundo virtual** é o cenário, em que o jogador interage. Este cenário normalmente é subdividido em várias partes e possui diversos objetos,

que podem ser objetos do mundo imaginário (como personagens, armas, itens, entre outros) quanto objetos de interação, tais como botões, menus, rótulos, etc. O **script** representa o código do jogo, onde o programador definirá o comportamento dos objetos de cena e a resposta da interação do jogador. O **motor de jogo** é responsável por coordenar as renderizações gráficas, física, colisão de objetos, sons, interface gráfica e outros elementos, dando as funcionalidades para o funcionamento do jogo. Em alguns jogos a **rede** fornece a experiência de multijogador. E todo o *software* interage com o **sistema operacional** do usuário.

Queiroz (2012) vem nos trazer uma análise sobre alguns *engines* de jogos: O primeiro deles é o **GameMaker**, que permite a confecção de jogos tanto com um modo *drag-and-drop* quanto com uma linguagem de programação própria do GameMaker, GML (*Game-Maker Language*, ou, Linguagem Game Maker), oferecendo assim um bom suporte para desenvolvedores iniciantes; A segunda é o **Genesis3D**, que é um motor para desenvolvimento de jogos com renderização para todos os programas em tempo real 3D. Ela é trabalhada com a linguagem de programação C++, e necessita estar em conjunto com um compilador para essa linguagem, um pacote de modelagem para os personagens 3D e um *Level Editor*, para a construção de mapas e fases; A terceira *engine* é o **Unreal Engine 3**, um motor mundialmente conhecido e completa para jogos profissionais em 3D, possuindo motor de física, gráfico e animação, de rede, de inteligência artificial, de áudio e possui ferramentas de suporte a iluminação, sombras, texturização, malhas, terreno, gerenciamento de cores e interface para programadores através de editores, bibliotecas e da linguagem Unrealscript.

2.6 Unity 3D

O Unity3D é uma das *engines* de jogos mais utilizadas hoje. Foi criada pela Unity Technologies em 2005 e é a mais popular *engine* de jogos, comandando cerca de 45% do mercado de desenvolvimento, de acordo a The Next Web (2016). Sua distribuição possui duas licenças, uma sendo a profissional e paga e a outra comum e grátis. É possível desenvolver e vender jogos com ambas, porém há uma regra dizendo que se você vender jogos com a licença comum e atingir determinado lucro, terá que obrigatoriamente comprar a versão profissional e paga (XAVIER, 2011). Até a versão 4 o Unity3D possuía importantes diferenças entre a versão livre e a paga, como as plataformas desenvolvimento, na versão grátis apenas era possível desenvolver para Windows, Mac e Web. Porém para manter a

concorrência com a Unreal Engine 4 (FLAUSINO, 2015), ela liberou todas as plataformas e outros recursos para a versão gratuita do Unity3D 5, mantendo como exclusivos da versão paga apenas um serviço de compilação na nuvem (*Cloud Build*), pacotes com estatísticas e licenças especiais para equipes.

Passos *et. al*. (2009) dizem que para uma *engine* de jogo ser considerada completa, ela necessita de alguns elementos básicos, como um sistema de renderização 3D, um sistema de simulação física, uma boa arquitetura para a programação de scripts e a capacidade de se importar modelos 3D, imagens e áudios diretamente. Todas essas características estão presentes no Unity de forma intuitiva, sendo natural sua aprendizagem. A Unity3D é uma poderosa ferramenta que nos permite a compilação de jogos e aplicativos em 2D e 3D com uma qualidade impressionante, a *engine* oferece a tecnologia mais avançada em termos de renderização, iluminação, terrenos, partículas, física, áudio, programação e *networking* (XAVIER, 2011). Outro destaque importante apontado é a possibilidade de fazer alterações nos objetos em plena execução, modificando valores e assim verificando erros e fazendo correções durante o processo de desenvolvimento.

2.6.1 Interface

A interface do Unity3D é projetada para dividir a atenção do desenvolvedor em algumas janelas principais, que são dispostas no sistema podendo as mesmas serem redimensionadas, mudada de posição e agrupadas de várias formas diferentes, de modo que o usuário desejar e que melhor lhe aprouver.

Figura 14 - Janela *Scene*

A primeira tela e mais utilizada é a ***Scene*** (cena) (Figura 14), nela o usuário poderá usar o recurso de arrastar-e-soltar para manipular graficamente objetos de cena como câmeras, cenários, personagens e todos os elementos que compõem a cena (PASSOS et al., 2009). Essa tela possui um recurso chamado *Scene Gizmo*, que tem a função de modificar a orientação da tela nas três dimensões: X, Y e Z, permitindo o usuário ver a *scene* por diferentes ângulos (função que existe somente em jogos 3D). O usuário também possui ferramentas que permitem que ele controle a posição, a escala e a rotação dos objetos apenas com o controle do mouse, isso permite um desenvolvimento mais rápido e mais visual, sem ter de depender de scripts de programação e tendo feedbacks imediatos.

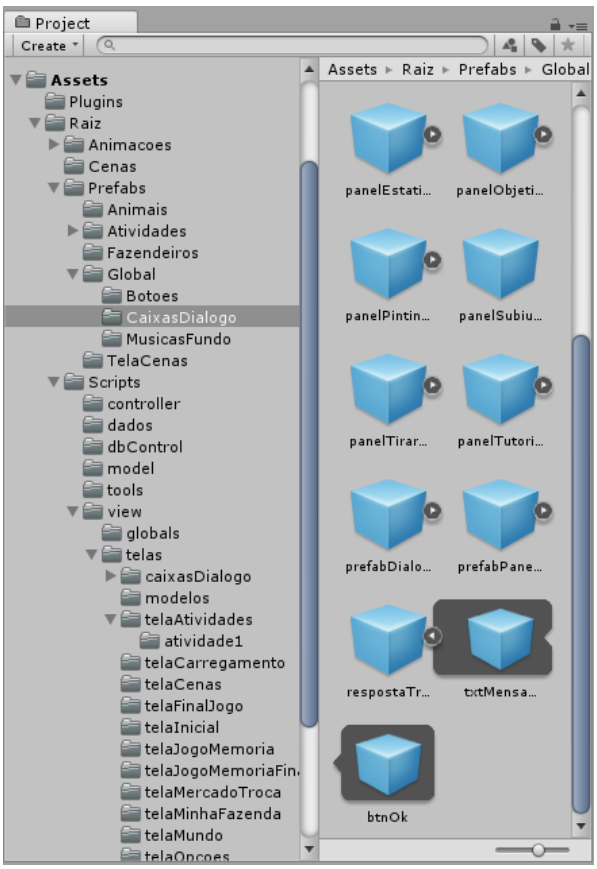


Figura 15 - Janela *Project View*

A segunda tela é a de ***Project View*** (visualização do projeto) (Figura 15), ela funciona exatamente como um explorador de arquivos de duas áreas, em que na primeira área existe uma organização hierárquica por pastas e arquivos, e mostrando o conteúdo da pasta selecionada na outra área. A pasta raiz é a "*Assets*" que fica dentro do projeto do Unity3D.

Um dado interessante sobre essa tela é que a mesma funciona conjuntamente com o explorador de arquivos do Sistema Operacional, assim se você inclui um arquivo no explorador de arquivos, ele será incluído no projeto e poderá ser visualizado no Unity3D, assim como o contrário também é verdade.

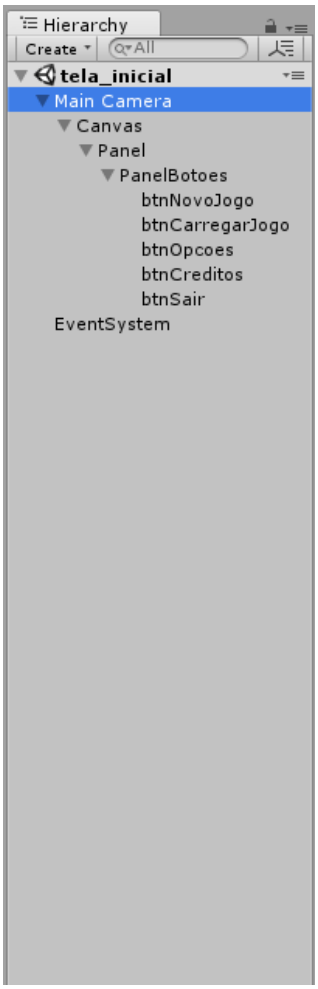


Figura 16 - Janela Hierarchy

A terceira tela é a de ***Hierarchy*** (hierarquia) (Figura 16) onde todos os objetos dispostos na tela (visualizáveis ou não) estão listados também em forma hierárquica, trabalhando com o conceito de pais e filhos. Este conceito torna-se importante no momento em que se pode aplicar certas transformações no pai a partir dos filhos, ou o contrário. A tela *Hierarchy* tem como objeto raiz a Cena, que são os arquivos que comportarão todos os outros e que constituirão aquilo que o jogo mostrará para o jogador.

Figura 17 - Janela *Game View*

A quarta janela é a ***Game View*** (visualização do jogo) (Figura 17), sendo a própria responsável pela simulação da execução do jogo tal qual o usuário irá vê-lo. Este é um recurso importantíssimo pois não é necessário construir o jogo para verificar o resultado final, permitindo também o usuário a testar o jogo em diferentes resoluções, com o aumento ou diminuição da tela do *Game View*. Além disso a Unity3D fornece a opção de paralisar (botão *pause*) a simulação enquanto ela estiver em depuração, de forma a possibilitar que os parâmetros dos vários dos vários elementos possam ser ajustados para experimentação. Lembrando que o ajuste desses parâmetros não necessita que a simulação esteja paralisada, podendo ser alterados inclusive enquanto a simulação esteja em execução (PASSOS *et. al*., 2009).

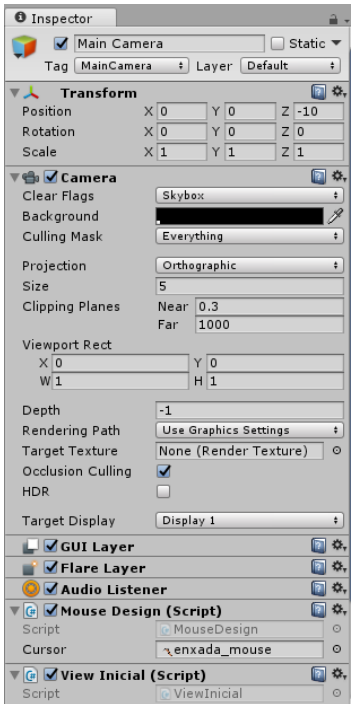


Figura 18 - Janela Inspector

A quinta tela é a ***Inspector view*** (visualização de inspeção) (Figura 18), que é usado para visualizar e editar as propriedades e configurações de quase tudo no Editor Unity3D, incluindo itens da física do jogo como *GameObjects*, *Assets* e *Materials*, bem como configurações e preferências do próprio editor. Entre essas configurações se pode configurar totalmente a posição e tamanho do objeto (apesar de ser mais recomendado fazer isso pelo *Scene View*, apenas fazendo ajustes muito precisos no *Inspector*), também há a inclusão e configuração de componentes como áudio e imagem dentro do objeto que se esta trabalhando, além da inclusão dos scripts de programação e a manipulação de suas variáveis públicas.

No Unity3D ainda há diversas outras telas como a de Animação, a *Asset Store* e o Console, porém elas serão abordadas conforme forem necessárias, para que não nos extendamos demais.

2.6.2 *GameObjects*

A Unity3D trabalha bastante com o conceito de hierarquia e passagem de comportamentos de pai para filho, porém essa característica não supriria as necessidades dos jogos, em que determinadas características deveriam ser compartilhadas e outras não. Nesse modelo, um objeto de jogo é especificado através da composição de várias funcionalidades, que são agregadas (ou removidas). Cada funcionalidade é implementada por um componente (classe que herda de um componente básico). Esse container genérico ainda é denominado de *GameObject* e funciona como um repositório de funcionalidades, ou mais especificamente, componentes (PASSOS *et al*., 2009). Esses componentes podem ser imagens, áudios, scripts de programação, componentes de física que orientam o jogo a entender como determinado objeto irá se comportar na física do mundo em questão, entre diversos outros tipos de componentes. O *Transform* é o único componente que já nasce presente nos *GameObjects* e ele é responsável pela posição, escala e rotação do objeto na cena, além de representar a hierarquia, contendo a informação de quem é o pai daquele objeto.

Outro importante conceito que podemos explorar em relação aos *GameObjects* é a criação de *Prefabs*, que nada mais são do que a representação de um modelo de um *GameObject* criado, que são vinculados aos scripts de programação para serem chamados à cena em tempo de execução, podendo ser reproduzidos fielmente ao modelo ou modificados de acordo a necessidade. Os *prefabs* agem como um *template* em que você pode criar novas instancias de objetos na cena. Por exemplo, podemos ter o *prefab* de um adversário padrão

para ser instanciado na cena, porém podemos instanciar vários desses adversários modificando algumas características do mesmo, como força, cor do cabelo, altura, etc. Uma característica bastante útil é a possibilidade de pode instanciar o *prefab* na cena em tempo de projeto e qualquer alteração feita no *prefab* é automaticamente reproduzida em suas instancias.

Os *GameObjects* também possuem outros dois recursos interessantes. O primeiro deles é a possibilidade de ativar e desativar o objeto em tempo de projeto ou de execução, essa característica é útil por exemplo em jogos de tiro em que as balas são disparadas e ficam em cena, sendo renderizadas (por menor que sejam, ainda necessitam de algum processamento) e se tornam inutéis para a experiência geral do jogo, logo o melhor é desativa-las, fazendo com que saiam de cena. Outro uso relevante para este recurso é a chamada e desativação de caixas de dialogo com o usuário. O segundo recurso é a inclusão de *Tags*, que nada mais são do que marcadores usados nos objetos para que possam ser facilmente identificados como grupo, ou até mesmo serem usados isoladamente, pelos scripts, com o uso de *Tags* cessa-se a necessidade de criar variáveis públicas no script para poder manipular esses objetos, sendo os mesmos buscados através das *Tags*.

2.6.3 *Scripting*

Os scripts no Unity3D são o coração do comportamento dos objetos nas cenas, eles são componentes e devem ser vinculados a um objeto para que possam ser usados pelo jogo. Na versão atual o Unity3D suporta scripts em duas linguagens: Unity Script e C#. Elas possuem a mesma capacidade e funções, tendo algumas diferenças de sintaxe e o C# tendo a vantagem de ser orientada a objeto. Porém uma das grandes diferenças entre elas é a comunidade de desenvolvedores do Unity3D que utilizam uma linguagem e outra, onde de acordo com UnityBlogs (2014) o C# mostra ser a linguagem mais utilizada com 80,4% de uso. Outro dado relevante que podemos medir é o desempenho, apesar de ser pequena a diferença, em uma medição feita por DentedPixel (2013) com um mesmo script feito em UnityScript e feito em C# com teste de stress em um total de 10 vezes, o C# rodou o script um pouco mais rápido na média, atingindo 8,075ms, enquanto o UnityScript atingiu 8,142ms.

Independente da linguagem utilizada, na criação do script a classe é imediatamente ligada à biblioteca Mono, esta biblioteca possui vários métodos que podem ser usados em

consonância com os eventos do Unity3D. Estes métodos bastam ser criados no script com o retorno “void” e sem passagem de argumentos. São eles:

- Start: Criado junto com o script, essa função é executada automaticamente e apenas uma vez assim que o objeto ao qual o script esta vinculado é instanciado;
- Update: Criado também junto com o script, ela é executada a cada chamada de *frame* renderizada no jogo. Ou seja, se o jogo esta sendo executado a 60 fps (*frames-por-segundo)* essa função será executada 60 vezes a cada segundo;
- Awake: Essa função é chamada a cada vez que o objeto é habilitado na tela. Apesar de ser apenas uma vez também, a cada vez que o objeto ser desativado e ativado novamente, a função também será chamada;
- OnGUI: Função utilizada para renderizar e manipular eventos de GUI (*Graphical User Interface*). Ela é chamada várias vezes por *frame* e fica escutando para quando os eventos ocorrerem, a função poder trata-la;
- OnDestroy: Função chamada quando o objeto for destruído;
- OnEnabled: Função chamada quando o objeto for habilitado;
- OnMouseEnter: Função chamada quando o ponteiro do mouse entrar na área do objeto GUI ao qual o script está vinculado;
- OnMouseExit: Função chamada quando o ponteiro do mouse sair na área do objeto GUI ao qual o script está vinculado;

2.6.3.1 Corotinas

Normalmente as funções são executadas imediatamente assim que chamadas, porém algumas vezes precisamos que uma função tenha um tempo de delay, ou espera, antes de ser executada, como exemplo, temos um efeito de esmaecimento em um objeto em tempo de execução, onde se precisa modificar o valor alpha do objeto aos poucos, de 0 até chegar em 1, caso feito com uma chamada comum o usuário não verá o efeito aparecendo aos poucos, e

sim como se o objeto tivesse aparecido do nada, mas se colocarmos um tempo de espera a cada aumento do valor, então o jogador poderá ver claramente esse efeito.

Para utilizar a corotina é necessário passar o nome do método, como *String*, para o método “StartCoroutine”. No método a ser chamado o retorno terá de ser do tipo IEnumerator e no corpo do método deverá ter uma linha como a seguinte: “yield return;”, será exatamente nesse ponto em que o método irá pausar e ser retomado.

2.6.4 Pastas especiais e ordem de compilação dos scripts

O Unity3D possui alguns nomes de pasta que são reservados e influenciam na ordem em que essas pastas são compiladas. Esses nomes de pasta são:

- Assets: que é a pasta principal do projeto, serve normalmente de raiz para todas as pastas do projeto;
- Editor: Essa pasta serve para guardar scripts usados no desenvolvimento e não são necessários na hora da execução do jogo, por isso eles são deixados para serem compilados por último;
- Editor Default Resource: Que irá guardar scripts de Editor que usarem a função EditorGUIUtilily.Load;
- Resources: Pasta que guardará Assets (imagens, áudios, modelos 3D, etc.) em que se desejará carregar sem a necessidade da vinculação com um script, e chamando tal recurso com o método Resource.Load;
- Standard Assets: Pasta em que ficarão os Assets que o usuário importar para o projeto (com o uso do Assets Store, por exemplo);
- Streaming Assets: Pasta usada para armazenar determinados arquivos de mídia que se deseja passar para um outro local de arquivos na máquina de destino, fazendo o mesmo ser acessível por outro aplicativo;
- Hidden Assets: Pasta onde se coloca os arquivos que se deseja que o Unity ignore na compilação;

2.6.5 Controle de Versão

Um problema que muitas equipes de programação iniciantes têm ao iniciarem um projeto é como organizar o projeto para que várias pessoas possam trabalhar nele sem um afetar o que o outro está fazendo. Na programação profissional isso é chamado de Controle de Versão e existem no mercado diversas soluções para tratarem desse problema. No geral o controle de versão usa um projeto guardado em um servidor como base e todas as máquinas dos programadores possuirão uma cópia desse projeto. O programador então faz alterações no projeto em sua máquina e envia todo o projeto para o servidor com o projeto base, onde o mesmo irá comparar os arquivos alterados e atualizar de acordo a versão que o programador enviou, tratando de possíveis conflitos com o trabalho enviado por outros programadores concomitantemente.

O Unity3D trabalha com o Controle de Versão de forma interna e utiliza dois sistemas de servidores para controlar esse recurso: o *Perforce* e o *Plastic* SCM. Porém o Unity3D não se restringe apenas aos dois, com a devida configuração é possível vincular outros sistemas de controle aos arquivos Unity3D, como o *Subversion* e o *Bazzar*. Também existe no Unity3D uma ferramenta chamada *Smart Merge*, que pode mesclar duas alterações feitas em um mesmo arquivo de cena ou *prefab*, algo um pouco mais difícil por estar se tratando de comparação de *bytes* e não uma comparação de texto.

2.6.6 Controle de Áudio

Um jogo não é a mesma coisa sem sons, seja uma música de fundo ou apenas de efeitos de cena, efeitos sonoros fazem parte da experiência criada pelos jogos. O Unity3D essa questão a partir de tipos específicos de componentes chamados de AudioSource (Origem do som) e AudioListener (Ouvinte do som), partindo da premissa e da forma natural de que sempre haverá algo que produz o som e outra coisa que ouvirá o som produzido, que por padrão será a própria câmera do jogo. O Unity3D também possui dois objetos chamados de AudioFilter e AudioMixer, que procuram dar efeitos no som em busca de maior realismo.

O Unity3D possui suporte a vários formatos de arquivos de áudio, dentre eles podemos citar: MPEG layer 3 (.mp3), Ogg Vorbis (.ogg), Microsoft Wave (.wav), Audio Interchange File Format (.aiff/.aif), Ultimate Soundtracker module (.mod), entre outros. Esses arquivo ao serem importados para o Unity ficam como AudioClips que inclui

metainformações no arquivo indicando como os mesmos serão executados e compilados dentro do programa.

Um interessante recurso que o Unity3D também oferece é a possibilidade de acessar o Microfone em pleno jogo (ou antes) e gravar um áudio a partir do mesmo para que se possa utilizar.

2.6.7 UI (*User Interface)*

O tratamento de GUI é muito importante para uma *engine* de jogos, principalmente ao se tratar de jogos 2D. A Unity3D em versões anteriores a 4.6 oferecia uma biblioteca GUI que apesar de possuir vários recursos na criação de componentes não tinha correspondente na janela *Scene*, ou seja, você só conseguiria ver o resultado na simulação. Após a versão 4.6 a Unity3D trouxe uma nova biblioteca chamada de UI, trazendo todos os recursos de uma forma fácil e intuitiva. Os *widgets* são manipulados por um *RectTransform*, o qual é um *Transform* comum porém com a possibilidade de alterar as dimensões de um botão de forma 2D. Isto possibilita ao usuário que seus componentes tenham sempre a mesma dimensão, independente da tela do usuário ou tenham uma dimensão fixa (DESENVOLVIMENTO DE JOGOS WIKIDOT, 2017).

O elemento raiz do sistema UI é o *Canvas* que comportará todos os outros elementos na Hierarquia dos objetos, ele é mostrado como um retângulo vazio na janela *Scene* e os outros elementos serão colocados dentro dele através do recurso de arrastar-e-soltar. Os elementos dentro do *Canvas* são desenhados conforme sua ordem na hierarquia, mostrada na janela *Hierarchy*, caso se queira mudar essa ordem (objetos sobrepostos) basta trocar a ordem arrastando o elemento para um ponto mais alto do que se encontra na *Hierarchy*, ou controlar tal propriedade através dos métodos: *SetAsFirstSibling, SetAsLastSibling*, e *SetSiblingIndex*.

Em relação ao seu posicionamento, rotação, tamanho e escala, os objetos UI dentro do *Canvas* possuem um componente chamado *RectTransform* que possui todos esses dados para serem alterados conforme o usuário desejar, podendo ser feito através da *Scene View* com o mouse e uma barra de ferramentas para as alterações chamada de *RectTool*, ou através da *Inspector View*, colocando diretamente os valores que se deseja. Outro recurso interessante são os *Anchors*, que são uma marcação nos quatro cantos do objeto UI que definem como o objeto se comportará com uma resolução diferente da que se esta modelando na *SceneView*.

Entre os objetos UI podemos destacar: o ***Text***, que serve como um texto puro no meio da tela, podendo ser usado como título ou rótulo. Em suas propriedades no *Inspector* pode-se notar que o mesmo é criado com um script já vinculado de mesmo nome (*Text*) que possui diversas propriedades como tipo de fonte, estilo, tamanho, alinhamento, dentre outros; ***Image***, sendo o componente em que se pode incluir uma imagem e definindo também algumas propriedades como o modo como a mesma irá aparecer (em relação a sua escala), a cor de fundo em que a imagem estará, dentre outros; ***Button***, que é um componente de interação, onde o script vinculado ao mesmo possui propriedades para que se possa indicar qual ação irá ser realizada quando o jogador clicar no botão; ***Slider***, sendo o mesmo um componente que se pode arrastar um objeto interno que será criado junto com o *Slider*, e internamente mudará o valor do *Slider*, que poderá ser percorrido de um valor mínimo até um valor máximo, definido pelo usuário nas propriedades do *Slider*; e ***Input Field***, sendo um campo de texto em que o jogador poderá inserir algo através do teclado. Possui as mesmas propriedades do componente *Text* em relação à alteração de fonte.

O Unity3D também oferece um recurso de animação por mudança de estado para esses componentes UI, para isso basta que se crie um *Animator Component* para o objeto e o recurso será disponibilizado, podendo gerar vários efeitos de acordo aos seguintes estados: *Normal, Highlighted, Pressed* e *Disabled*. Os efeitos podem ser realizados através do próprio *Scene View*, pois na janela de Animação existe um botão de *Record* (gravar), em que ao se clicar o usuário pode fazer determinadas alterações no objeto selecionado e ao se clicar no *Record* novamente o objeto voltará ao estado original, porém gravará as modificações que foram feitas quando o estado selecionado for alcançado.

2.7 Estudo de caso: Fazendinha Matemática

Agora que já vimos um pouco sobre várias das partes do desenvolvimento de um jogo, precisamos discorrer sobre que jogo será criado como estudo de caso do nosso desenvolvimento de um jogo educacional com a ferramenta Unity. O jogo a ser desenvolvido faz parte de um projeto iniciado pela Profa. Dra. Tânia Cristina Rocha Silva Gusmão, que atua nas áreas de Pedagogia e Matemática pela Universidade Estadual do Sudoeste da Bahia.

Este projeto de nome Fazendinha Matemática visa desenvolver a compreensão das operações fundamentais da matemática (adição, subtração, multiplicação e divisão) e de seus algoritmos. A estratégia usada para potencializar esse aprendizado é utilizando uma história

lúdica (voltada para crianças entre 5 e 8 anos) e um sistema de trocas, aludindo às mudanças de casas numéricas (unidade, dezena, centena, milhar etc.).

A Fazendinha Matemática já está presente e vem sendo aplicado a mais de dez anos em várias escolas no município de Vitória da Conquista-BA, e já mostrou resultados eficientes no aumento de rendimento dos alunos ao qual foi aplicado. A proposta desta monografia é a criação de um jogo virtual sob a mesma proposta da Fazendinha Virtual, porém trazendo as vantagens que a utilização de um jogo eletrônico tem em relação aos jogos comuns, já discutidas em capítulos anteriores.

2.7.1 Sequências Didáticas e Desenho de Tarefas

A Fazendinha Matemática é baseada em desenho de tarefas e sequências didáticas, dois conceitos pedagógicos que buscam uma aprendizagem alternativa do que a simples passagem de conteúdo e a aplicação de uma prova. Por isso inicialmente precisamos entender o que são esses dois conceitos e como eles se relacionam com os jogos eletrônicos.

Segundo Gusmão (2014) as tarefas podem ser entendidas como um conjunto de atividades pensadas e desenhadas para professores (e estudantes), cujo objetivo é o de desenvolver e avaliar destrezas cognitivas e metacognitivas desses professores (e estudantes). Podemos então notar que cada atividade passada para os estudantes deve ser desenhada com um planejamento em mente, buscando sempre o desenvolvimento e avaliação. Já as sequências didáticas são um conjunto de atividades ligadas entre si, planejadas para ensinar um conteúdo, etapa por etapa, organizadas de acordo com os objetivos que o professor quer alcançar para a aprendizagem de seus alunos (COSTA; PERETTI, 2013, p. 6). As sequências didáticas utilizam-se muitas vezes de atividades lúdicas, que envolvem o aluno em situações práticas com aporte teórico para realizar os desafios propostos. Esses desafios começam de maneira mais simples, e vão aumentando a complexidade aos poucos, seguindo a evolução do próprio aluno (DIAS; GUSMÃO, 2015).

Logo se pode notar a relação destes conceitos com os jogos de maneira em geral e principalmente aqueles em que os jogos eletrônicos são mais atuantes, pois é característica marcante nos jogos a nivelação da dificuldade a partir da evolução do jogador dentro do jogo. Normalmente os jogos iniciam com níveis de dificuldade fáceis, para que o jogador se acostume com os comandos do jogo e com a história, assim que ele começa a passar de fase, o nível de dificuldade também aumenta, o que exige do mesmo maior habilidade para

conquistar os novos desafios, e assim prossegue até as etapas finais. Podemos dizer assim que um jogo tem uma correlação muito grande com uma sequência didática, e as tarefas têm uma correlação muito grande com os desafios relacionados nos jogos. Em se tratando de jogos eletrônicos, podemos entender estes como sendo situações de aprendizagem lúdica utilizando tecnologias digitais (DIAS; GUSMÃO, 2015).

Outro conceito que se alinha em relação às sequências didáticas e desenho de tarefas é o de redesenho de tarefas, que segundo Pochulu, Font e Rodríguez (2013) faz referência ao processo de adaptação, adequação e ajustes das tarefas. Sendo assim, dizemos que este trabalho tem como objetivo fazer um redesenho das tarefas e da própria sequência didática para adequa-la às tecnologias digitais e a forma como os jogos eletrônicos trabalham tanto a aprendizagem quanto a ludicidade.

O principal motivo da realização desse projeto é justamente trazer os alunos e professores a pensarem como realmente funcionam as operações básicas da matemática, utilizando do artifício lúdico para prender a atenção dos alunos e fazer com que os mesmos tenham outro olhar em relação à matemática, considerada por muitos como algo aterrorizante e difícil.

2.7.2 História e Jogo Analógico

O jogo da Fazendinha Matemática é dividido em quatro momentos principais. No primeiro momento o professor conta a história para os alunos. A história é resumida da seguinte forma: Havia um vilarejo com várias fazendas e fazendeiros que cuidavam de seus animais e viviam felizes, quando um dia uma grande ventania praticamente destrói o vilarejo e leva muitos dos animais embora. Porém, um fazendeiro chamado Seu Toninho encontra alguns pintinhos coloridos que ninguém nunca havia visto, e tem uma ideia: montar um sistema de troca de animais, baseado em uma tabela (Figura 19) onde um animal vale dois de outro, para que todos pudessem erguer suas fazendas e ganhar novamente os animais. Os alunos então são convidados a virar fazendeiros e a erguer suas fazendas, seu objetivo principal é conquistar todos os itens da tabela até o último: um lote de terra.

Figura 19 - Tabela de Trocas

O segundo momento é composto por um jogo da memória (Figura 20), em que todos se sentam para jogar. As cartas são os pintinhos coloridos que serão “caçados” pelos jogadores para que se possam fazer as trocas.

Figura 20 - Alunos jogando o Jogo da Memória. Fonte: (Vieira, 2015)

O terceiro momento é o de fazer as trocas, e que representa o momento mais importante do ponto de vista pedagógico, pois é ali que serão trabalhadas as habilidades cognitivas que se deseja que o aluno obtenha. Nesse momento de trocas o professor será o “mercado” e terá as cartas de todos os outros animais, e o aluno deverá indicar quantos pintinhos ele deseja trocar, por qual animal do mercado ele deseja trocar e por quantos ele deseja trocar. Como o sistema de trocas é dois por um em níveis de menor valor para o maior

valor, Dias et al.(2015) explicam como esse sistema alude à noção de divisão dos números binários, em que a quantidade de pintinhos a ser trocada é o dividendo, a base utilizada (no caso de trocas dois por um, a base será 2) será o divisor, e o resultado será a quantidade de animais do próximo nível que o jogador receberá na troca (Figuras 21 e 22).

Ainda temos ai o conceito de resto, quando, por exemplo, tentarmos trocar 3 sacos de milho por porcos, a divisão será de 3 por 2, que resultará em 1, sobrando 1, isso significará que o jogador receberá 1 porco e ainda continuará com 1 saco de milho sobrando (Figura 23).

Figura 21 - Primeiro exemplo de troca

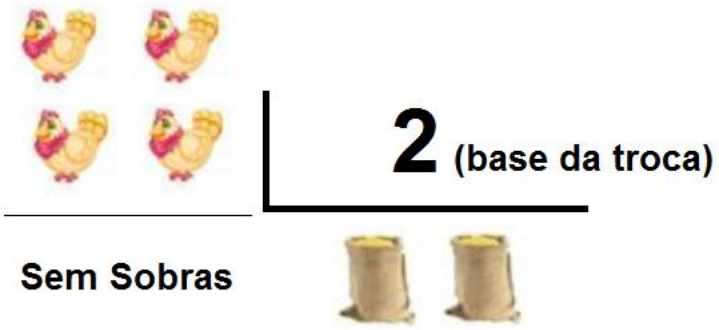


Figura 22 - Segundo exemplo de troca

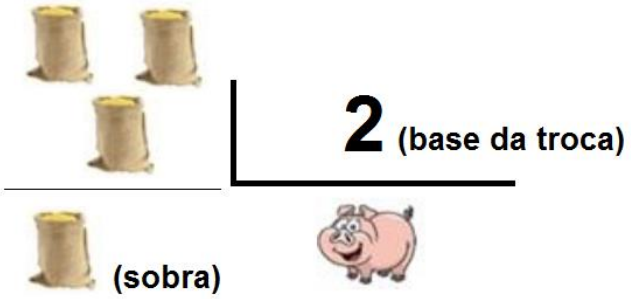
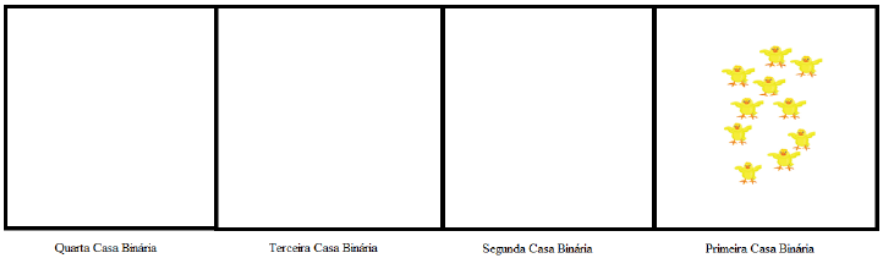


Figura 23 - Terceiro exemplo de troca

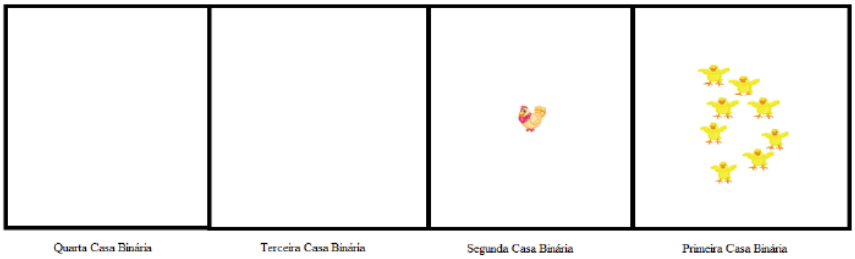
Dias et al. (2015) também nos explicam que a operação de adição (Figura 24) também é trabalhada conjuntamente com a da divisão, pois o que se busca nesse projeto é a desvinculação da ideia de que as operação precisam necessariamente ser pensadas em separado, na verdade, não existe uma proposta de fragmentação da aprendizagem de operações, a ideia é trabalhar com todas elas de modo integrado.

Em relação à adição, o jogo trabalha no sentido de mudança de casa e na passagem do "vai um", tão comumente utilizado nas operações de soma.



ANTES

OPERAÇÃO



DEPOIS

PARIDADE NA MATEMÁTICA

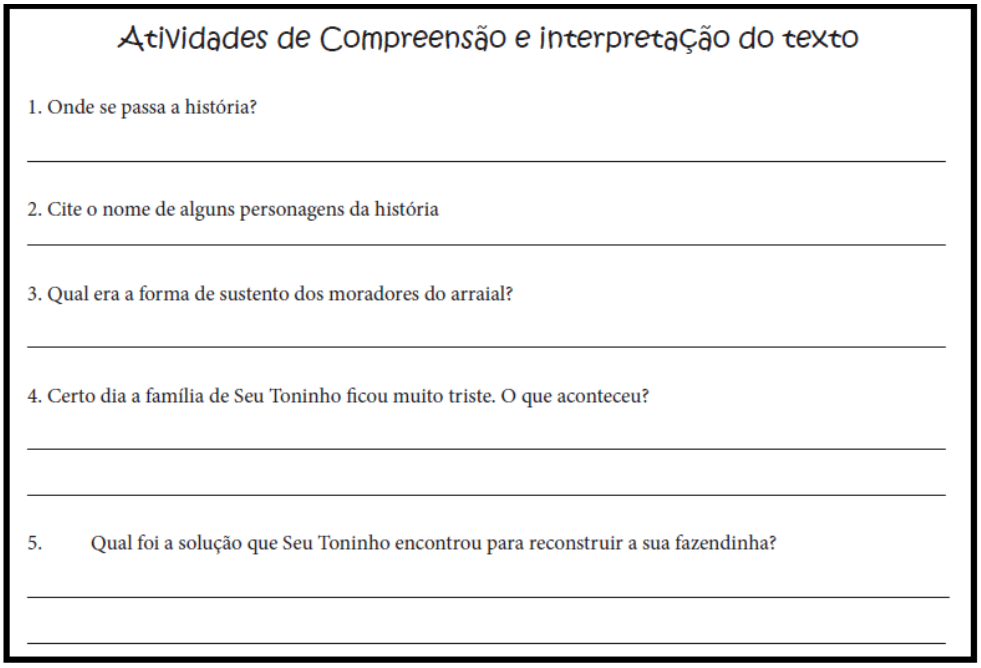
```
 1
0001
0001 +
------
0010
```

Figura 24 - Primeiro exemplo de troca por adição

Essa passagem de “casa” foi questionada por Vieira (2015) em relação a estudantes e futuros professores da rede de ensino, porém muitos não souberam responder ao questionamento do porque havia a mudança de casa quando o número ultrapassava a base em determinada casa (no caso da base binária, quando um número chega a 2, ele deve ser passado com o “vai um” para a próxima casa. No caso da base decimal, quando um número chega a 10, então ele deve ser passado para a próxima casa), isso nos mostra que não apenas os alunos, mas também os professores necessitam de outras formas de representação das operações para melhor compreenderem os conceitos formais. As outras operações

(multiplicação e subtração) não são exploradas no jogo principal analógico nessa parte de trocas, mas sim na etapa seguinte.

O quarto e último momento explorado na Fazendinha Matemática é o de Atividades (Figuras 25, 26 e 27), em que o professor elabora perguntas sobre as trocas e sobre a história para ver como os alunos absorveram os conhecimentos trabalhados e se os mesmos estão conseguindo acompanhar a ideia do jogo. Com essas atividades é possível avaliar de modo progressivo os alunos e verificar seu rendimento, dando uma visão mais clara para o professor se o objetivo do jogo está ou não sendo atingido para cada aluno.



Atividades de Compreensão e interpretação do texto

1. Onde se passa a história?

2. Cite o nome de alguns personagens da história

3. Qual era a forma de sustento dos moradores do arraial?

4. Certo dia a família de Seu Toninho ficou muito triste. O que aconteceu?

5. Qual foi a solução que Seu Toninho encontrou para reconstruir a sua fazendinha?

Figura 25 - Primeiro exemplo de atividade

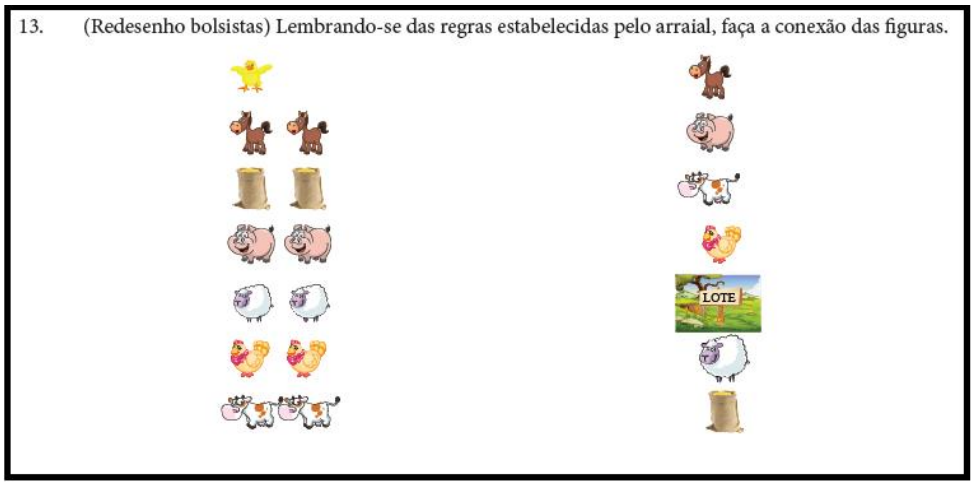


Figura 26 - Segundo exemplo de atividade

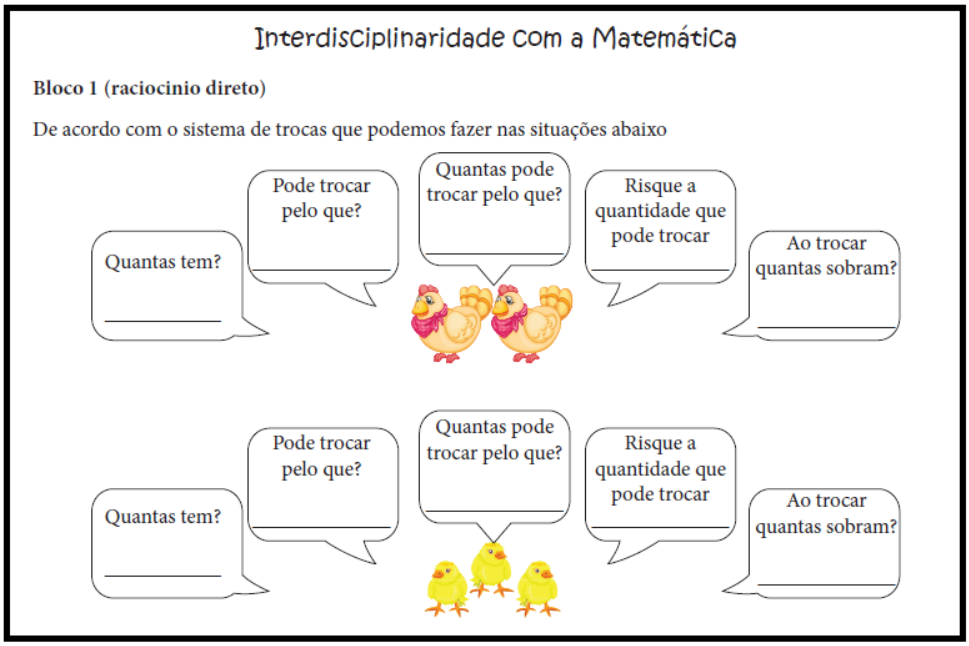


Figura 27 - Terceiro exemplo de atividade

3 Estado da Arte

Os jogos em si têm como objetivo levar uma experiência única aos seus jogadores, estando no mesmo patamar dos livros e das músicas, porém diferenciando-se dos mesmos pela sua interatividade com o público consumidor. Os jogos educacionais por sua vez tem esse objetivo ampliado, devendo levar além de uma ótima experiência, terem um aprendizado significativo, seja no nível cognitivo ou metacognitivo, de um ou mais temas/assuntos específicos. Por esse caráter de singularidade é realmente difícil encontrar jogos educacionais que se proponham a resolver exatamente o mesmo problema de outro, por isso fizemos uma pesquisa em busca de jogos eletrônicos voltados a temas educacionais apenas.

A partir de uma busca na ferramenta de busca Google, vê-se que há diversos sites que possuem grande repositório de jogos educativos, a esmagadora maioria destes voltadas ao público infantil e sendo compostos de um único mini-game. Porém existem grupos nascidos no meio acadêmico que estão buscando desenvolver propostas inovadoras de jogos completos (com enredos e vários minigames em seu progresso), aproximando mais o universo dos jogos educativos daqueles comerciais, que tanto atraem jovens e adultos.

Destacamos aqui propostas do grupo brasileiro Centro de Pesquisa e Desenvolvimento Comunidades Virtuais que se dedica a criação de games voltados à educação, onde há uma proposta de cunho mais científico e a participação tanto de estudiosos de jogos e tecnólogos, quanto de educadores, resultando em objetos de aprendizagem significativos, que é onde pretendemos chegar com o presente trabalho.

3.1 Centro de Pesquisa e Desenvolvimento Comunidades Virtuais

O Comunidades Virtuais é um centro de pesquisas da Universidade do Estado da Bahia (UNEB), em Salvador. Coordenado pelas professoras Lynn Alves e Carmen Lima, ele foi criado em 2002 e já conta com onze jogos desenvolvidos, constando como o grupo que mais desenvolveu jogos dentro de uma universidade pelo Relatório do Mapeamento da Industria de Games do Brasil 2014, financiado pelo BNDES (Fleury, 2014). Eles

desenvolvem jogos tanto para PCs quanto para dispositivos móveis e abordam temas como o folclore, história da Bahia, funções quadráticas, Revolução Francesa, dentre outros. O Comunidades Virtual além de muito respeitada no Brasil pelo seu trabalho possui uma documentação bastante consistente de seus jogos, destacando suas fontes e escolhas durante o projeto, além de possuir também orientações de como o professor deve abordar o jogo em sala de aula.

Aqui apresentaremos três dos jogos desenvolvidos pelo Comunidades Virtuais, destacando um pouco sobre o processo e ferramentas de desenvolvimento, as histórias e elementos pedagógicos dos jogos, além de mostrar as telas do jogo e as decisões que os mesmos tomaram para tornar esses objetos de aprendizagem o mais efetivo possíveis.

3.1.1 Tríade – Liberdade, Igualdade e Fraternidade

Figura 28 - Jogo Tríade

Tríade é o primeiro jogo criado pelo grupo Comunidades Virtuais, é vinculado ao Mestrado de Educação e Contemporaneidade do Departamento de Educação do campos I da UNEB e contou com uma equipe de mais de 40 pessoas, dentre elas game designers, programadores, roteiristas, historiadores, artistas, dentre outros, durando cerca de 33 meses para ser desenvolvido. Ele é um jogo para PC, sendo suportado nos Sistemas Operacionais Windows e Linux, possui como gênero o estilo de aventura com elementos de RPG e é feito para ser jogado por apenas um jogador. Seu público alvo são estudantes a partir do 8º ano até

a 2ª série do ensino médio e tem como objetivo resgatar, de forma lúdica e interativa, episódios relacionados com o período de 1774 a 1793, aliando elementos ficcionais a fatos históricos.

3.1.1.1 Desenvolvimento

O jogo foi desenvolvido utilizando a Torque Game Engine, aliado a diversos outros programas como Photoshop (ilustrações e texturizações), 3D Studio Mark (modelagem), Adobe Flash (animações), entre outros. A linguagem de programação usada foi o C++, tanto para o desenvolvimento do jogo quanto para a adaptação do Torque Game Engine para as necessidades surgidas no jogo (uma das vantagens dessa engine é ser de código aberto). O Torque tem também como vantagem a disponibilização dos **Resources**, que são artigos disponibilizados na comunidade Torque que demonstram como implementar certas funcionalidades na mesma. Entre as utilizadas no jogo, estão: RPGDialog, que implementa um sistema de dialogo ao se aproximar de um NPC; immersiveAI, que implementa um sistema de inteligência artificial; Melee Combat, que implementa um sistema de combate com armas de mão; e Inventory System, que implementa um sistema de inventário.

3.1.1.2 História

A história faz o jogador encarnar na primeira parte em um nobre aristocrata chamado Henri Valois, e na segunda etapa em sua filha Jeanne Valois (personagens fictícios). Eles serão protagonistas nos acontecimentos que antecedem a Revolução Francesa, participando das motivações que levaram a Revolução que marcou o fim do Absolutismo no mundo.

Figura 29 - Forma usada no jogo Tríade para contar a história

O jogo é dividido em nove fases, que vai desde a primeira reunião de Henri Valois com apoiadores que também estavam cansados dos privilégios dados ao Primeiro e Segundos Estados, até a invasão e queda da Bastilha que tem Jeanne como participante. Uma particularidade nesse jogo, que o torna diferente até mesmo de vários grandes jogos comerciais, é a diferenciação na história por conta de uma decisão do jogador no meio do jogo, fazendo com que haja uma ramificação no enredo e finais diferentes.

Figura 30 - Fluxograma das fases do jogo Tríade

3.1.1.3 Telas do jogo

A primeira tela do jogo é um menu inicial que possui seis opções: Um “Novo Jogo”, onde o jogador pode iniciar uma nova aventura do início; “Jogo Salvo”, onde o jogador pode continuar um jogo já iniciado anteriormente, não perdendo seu progresso; Opções, onde se poderá mudar configurações de Controle, Áudio e Video; “História”, em que poderá ser revisitado as narrações que embasam todo o gameplay do jogo; Créditos, onde são mostradas as pessoas que ajudaram o projeto a vir a tona; e o “Sair”, em que se pode sair completamente do jogo.

Figura 31 - Menu do Jogo Tríade

A tela principal do jogo mostra as costas do personagem jogável, em um estilo conhecido como 3ª Pessoa, possuindo alguns elementos nas bordas da tela como a barra de vida (no canto superior direito) e uma barra de itens que o jogador possui no momento (na parte inferior).

Figura 32 - Gameplay do jogo Tríade

Durante o jogo também é possível abrir caixas de diálogo referentes ao Inventário do jogador, que mostra com mais detalhes os itens que o mesmo possui, além de peças de vestimenta chamada de Equipamentos, em que o personagem pode ser fortalecido com chapéus, espadas, escudos, botas, camisas, etc. Outra caixa de diálogo é referente às características do personagem como: nível, experiência, pontos de vida, dinheiro e habilidades, elementos muito comuns em jogos de RPG.

Figura 33 - Inventário e Características do jogo Tríade

3.1.1.4 Considerações sobre o jogo

O jogo Tríade alcança os objetivos a que se propõe, em nossa experiência de gameplay pudemos realmente sentir que estávamos jogando um jogo como qualquer outro jogo comercial, porém nos sentindo impelidos a saber mais tanto sobre essa parte da História, quando sobre seus personagens. Cremos que este projeto do Tríade é um marco para grupos brasileiros que visam desenvolver grandes jogos que possuem além do elemento lúdico, também o pedagógico, atraindo alunos e professores para outras fontes de informação e debates.

3.1.2 Búzios – Ecos da Liberdade

Figura 34 - Jogo Búzios Ecos da Liberdade

O jogo Búzios é um projeto que também concilia a ludicidade dos jogos com os conhecimentos sobre História. Dessa vez o grupo Comunidades Virtuais resolve explorar a Revolta dos Alfaiates, ocorrida no século XVIII na Bahia, um acontecimento que refletiu as ideias trazidas pela Revolução Francesa. De acordo com Souza, Rios e Alves (2010), o jogo foi desenvolvido tendo em vista a lei 10.639/03 que torna obrigatória a temática historia e cultura afro brasileira no currículo oficial da rede de ensino. O jogo é no estilo *adventure*, jogado por apenas uma pessoa e tem como único meio de entrada de dados o mouse, classificado assim como *Point-and-Click.* É disponibilizado nos Sistemas Operacionais Windows e Linux e tem como público alvo estudantes a partir do 5º ano até a 2ª série do ensino médio.

3.1.2.1 Desenvolvimento

O motor do jogo foi escrito sob a tecnologia Adobe Flash com a linguagem de programação Action Script 3, esta é uma ferramenta que permite a criação de recursos interativos que reúne ferramentas de desenho, animação e programação. Apesar de tecnologia ser largamente utilizada para aplicações Web, o jogo Búzios foi desenvolvido para ser jogado no modo Desktop.

O roteiro do jogo é definido através de arquivos XML, que são acessadas pela linguagem Action Script para carregar certos dados após determinados eventos que acontecem no jogo. Esses arquivos XML são divididos em: Arquivos de definição de cena, arquivos de definição de objetos e arquivos de definição de personagens. Os arquivos de personagem e objetos ainda são divididos em duas partes: Diálogo e Interação. A primeira parte contém todos os diálogos que podem ser feitos com aquele personagem ou objeto, a segunda parte define as ações que podem ser feitas com os mesmos, e sob quais condições elas podem ocorrer.

O Adobe Flash, apesar de ser uma ferramenta bastante versátil, não foi criada especificamente para games, por isso não possui alguns recursos essenciais para esse tipo de aplicação como a localização de rotas, que nada mais é do que identificar o caminho que um objeto deve percorrer de um ponto para outro. A equipe de desenvolvimento do Búzios teve de utilizar-se de alguns artifícios que o Flash disponibiliza aliados a criação de uma matriz de colisão para simular essa localização de rotas, permitindo ao personagem conseguir mover-se pelo cenário a partir do clique do mouse do jogador em determinados pontos, onde é possível que o mesmo passe.

Em relação aos salvamentos do jogo, o Búzios não se utiliza de um banco de dados ou SGBD, mas sim de um arquivo de texto que grava os estados de personagens, objetos e da cena atual, com o uso de uma extensão do Flash chamada Zinc.

Os desenvolvedores do Búzios relatam ao final da criação do jogo, dois problemas principais que enfrentaram com o Adobe Flash: O primeiro é em relação a quantidade de objetos interativos que eles poderiam colocar em cada cena, pois como todo bom jogo adventure, o jogador deseja poder ter o máximo possível do cenário disponível para que possa agir sobre. Porém havia um tempo de resposta muito grande do Adobe Flash quando haviam muitos objetos assim em cena, de modo que tiveram de reduzir para não prejudicar o desempenho. O segundo problema foi a dificuldade em “destruir” cenas da memória que não estão sendo usadas, pois o Flash mantém na memória qualquer cenário que tenha pelo menos

alguma variável ainda pendente apontando para o mesmo, logo os desenvolvedores tiveram de tomar o máximo cuidado para desvincular tudo que estivesse ligado a um cenário quando o mesmo fosse substituído por outro.

3.1.2.2 História

A história do jogo ambienta-se no período que antecede a Revolta dos Búzios, em que vários cidadãos da Bahia revoltam-se com a Coroa Portuguesa por conta da exploração e abusos que os mesmos estavam sendo submetidos. O jogador encarna na pele de Francisco, um advogado recém formado na Universidade de Coimbra que chega à Salvador com ideias revolucionárias sobre igualdade, liberdade e fraternidade.

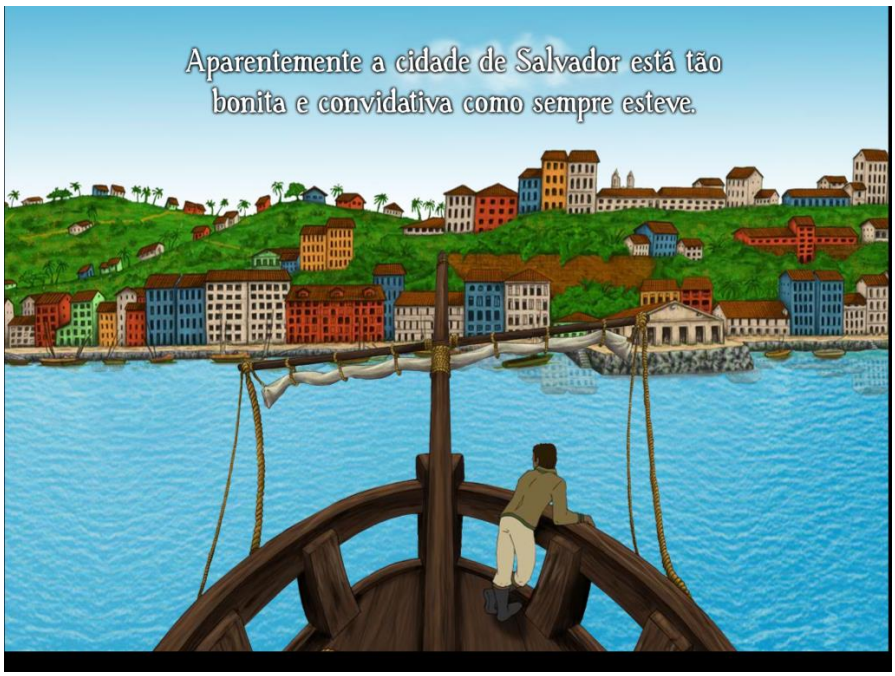


Figura 35 - Apresentação da História do jogo Búzios

O jogo é dividido em três atos principais: O primeiro será o de Francisco retornando a Salvador, em que o mesmo terá de solucionar alguns problemas que ocorrem no barco em que esta viajando para não ter seus pertences roubados. No segundo ato Francisco terá chegado a Salvador e fica sabendo sobre uma reunião de alguns cidadãos descontentes com a situação atual e precisará se preparar para poder participar da mesma. No terceiro e último ato, Francisco deverá primeiro provar seu valor e entrar para o movimento democrático, e depois de ingresso realizar certas missões para o mesmo.

3.1.2.3 Telas do Jogo

O menu principal do jogo possui cinco botões: “Novo Jogo”, “Carregar Jogo”, “Opções”, “Créditos” e “Sair”.

Figura 36 - Menu Principal do Búzios

A tela de *gameplay* do jogo mostra um cenário 2D em que estão dispostos o personagem principal (Francisco), personagens secundários e objetos. O jogador poderá, através do mouse, interagir com os personagens e alguns objetos em cena, clicando sobre os mesmos. Na parte superior a esquerda estará um botão de “Inventário”, em que ele poderá acessar os objetos que estão em sua posse. Ele também terá a opção de apertar a tecla “Esc”, entrando assim no menu de pause, podendo realizar algumas ações.

Figura 37 - Gameplay do jogo Búzios

A parte de Inventário pode ser acessada através do botão acima, e contém objetos que podem ser coletados no cenário. Esses objetos podem ser combinados entre si para criarem novos “objetos” que terão algum objetivo durante o jogo. Para aciona-los o jogador deverá arrasta-los para o ponto do cenário onde devem ser usados.

Figura 38 - Inventário do jogo Búzios

No menu de pause, o jogador terá as opções de Salvar o jogo ou Continuar um outro jogo já salvo, Alterar as opções de configuração e Sair de jogo.

Figura 39 - Jogo em Pausa

3.1.2.4 Considerações sobre o jogo

Mesmo sendo um game *point-and-click* os desenvolvedores conseguiram manter uma boa jogabilidade utilizando os recursos do Flash para simular a trajetória de rota, fazendo com que o mesmo fosse tão imersivo quanto qualquer adventure com comandos no teclado. Outro ponto interessante é o uso da combinação de objetos, apesar de os mesmos terem finalidade única, o que limita um pouco o jogo. As contribuições históricas ficaram bastante naturais e realmente fazem o jogador ter um interesse pela história contada.

3.1.3 Gamebook Guardiões da Floresta

Figura 40 - Jogo Gamebook Guardiões da Floresta

Este jogo constituiu-se de um novo projeto do grupo Comunidades Virtuais, feito em 2015, para fazer um game que relacionasse também elementos de *appbook* (livro com narrativa interativa). Este jogo tem como objetivo o estímulo às funções executivas da pessoa, tais como: memória de trabalho, planejamento, flexibilidade cognitiva, atenção seletiva, monitoramento e controle inibitório, em crianças com e sem indicação de Transtorno do Deficit de Atenção e Hiperatividade (TDAH), na faixa etária de 8 a 12 anos.

3.1.3.1 Desenvolvimento

O jogo foi feito na plataforma mobile e foi utilizado a engine Unity 3D como ferramenta desenvolvedora principal, sob a linguagem C#. Nery Filho (2015), em sua dissertação de mestrado, descreve as etapas de desenvolvimento desse jogo, desde sua concepção até os testes finais. A produção é dividida em: Pré- Produção, Produção e Pós-Produção.

A etapa de Pré-Produção se constituiu de reuniões e decisões de projeto, que envolveram game designer, roteiristas, artistas, bem como profissionais das áreas de saúde e psicologia para o auxílio sobre o TDAH e sobre as funções executivas. Além dos

profissionais, foram consultados também portais web que tratam do assunto de funções executivas. Algumas decisões importantes também foram tomadas nessa fase, como a mudança de tecnologia para o motor de jogo, passando de Flash (que foi descontinuado) para Unity, onde foram elencadas diversas vantagens do mesmo como: produzir executáveis para Android, Windows e Linux sem alteração de código (característica multiplataforma), possui funcionalidades em 2D, e o suporte a óculos de realidade virtual e geração de ambientes de realidade aumentada. O produto final desta etapa foi o GDD (*Game Design Document*) do jogo, em que todos os elementos do game foram definidos.

Na etapa de Produção foram criados os documentos e diagramas referentes a modelagem UML, para tais diagramas foi utilizado o programa ASTAH. Após serem analisados nos diagramas, aquelas partes do sistema eram então prototipados na ferramenta Unity. Os tipos de diagrama utilizados para a modelagem foram: Diagrama de Componentes, Diagrama de Fluxo, Diagrama de Caso de Uso, Diagrama de Atividades e Diagrama de Sequência.

Na última etapa, de Pós-Produção, os protótipos criados na etapa anterior eram avaliados a partir de instrumentos criados pela própria equipe para saber se os mesmos atingiam os objetivos esperados em relação ao trabalho das funções executivas.

3.1.3.2 História

A história do jogo conta a aventura de Lyu, uma corajosa garotinha, que se separa dos pais quando os mesmos viram prisioneiros de um grupo de desmatamento ilegal durante uma expedição na floresta amazônica. Lyu acaba perdida até se encontrar com os Guardiões da Floresta, e juntos eles tem o dever de deter a fabrica Aragon, a simbólica vilã do jogo, destruidora da natureza.

Figura 41 - História do jogo Gamebook Guardiões da Floresta

Há um total de 8 minigames no jogo, cada um trabalhando alguma das funções executivas de forma lúdica e contextualizada com a história, trazendo significação à atividade. A primeira atividade (Figura 43) chama-se Vitória Régia, e trabalha com a atenção seletiva. Para cumprir o minigame o jogador deverá selecionar s flores que passam ao lado e arrasta-las para os espaços corretos no meio. O segundo minigame (Figura 44) é a Flor da Lua, em que o jogador deverá se lembrar da posição de flores que aparecerão randomicamente em algumas caixas. Este minigame trabalha com a memória de trabalho. O terceiro chama-se Replantio (Figura 45) e também trabalha com a memória de trabalho, nele o jogador deverá se lembrar da sequência correta em que deve associar árvores aos espaços a que elas devem ser plantadas. No quarto game (Figura 46), o jogador irá treinar a atenção seletiva e o controle inibitório, nele será apresentado várias jaulas em que será necessário libertar os animais que ali estão, e para isso o jogador deverá clicar na tela somente quando um indicador que estará caminhando de um lado para o outro, passar por uma área verde. No quinto minigame (Figura 47), chamado de Cartas, o treinamento é em relação a atenção seletiva e a memória de trabalho. O jogador deverá seguir as regras de um jogo de cartas, relacionando cartas que aparecem aleatoriamente na tela. O sexto é o minigame Escondidos (Figura 46) e trabalha com a atenção seletiva e o controle inibitório. Neste jogo o jogador deverá clicar apenas no personagem indicado, enquanto vários outros aparecem. O sétimo jogo é o Tubulações

(Figura 49), onde há o destaque das funções de planejamento e memória de trabalho. Nele o jogador deve criar um caminho formado por tubos, de um ponto inicial até um ponto final. O último mini-game é chamado de Fábrica (Figura 48), e trabalha com a atenção seletiva e o planejamento. Aqui o jogador precisa mover certas peças a partir de estímulos mecânicos recebidos.

Figura 43 - MG 02 – Vitória Régia

Figura 42 - MG 02 – Flor da Lua

Figura 45 - MG 03 – Replantio

Figura 44 - MG 04 – Jaulas

Figura 47 - MG 05 – Cartas

Figura 46 - MG 06 – Escondidos

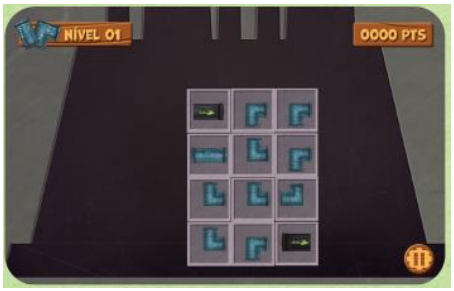


Figura 49 - MG 07 – Tubulações

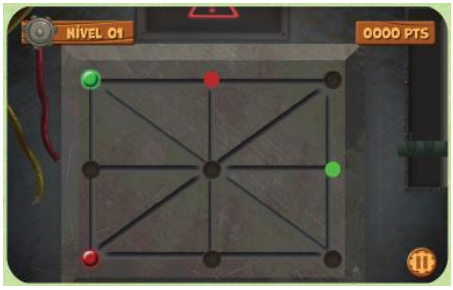


Figura 48 – MG 08 – Fábrica

As avaliações de desempenho no jogo são marcados em dois momentos: No primeiro serão dadas estrelas a cada mini-games finalizado pelo jogador, avaliando a forma e perícia com que a tarefa foi executada

Figura 50 - Resposta de desempenho do jogo Gamebook Guardiões da Floresta

No segundo será contabilizado todo o progresso do aluno em todas as funções executivas trabalhadas, e os resultados serão postos em forma de um gráfico de seis vértices (um para cada função), onde quanto maior for o afastamento do centro, maior será a pontuação do jogador naquela função.

Figura 51 - Gráfico das funções executivas do jogo Gamebook Guardiões da Floresta

3.1.3.3 Considerações sobre o jogo

Percebe-se que neste jogo há uma preocupação maior com as avaliações feitas sobre o jogador em relação ao aprendizado adquirido, que tem relação com envolver além de uma questão de educação, também uma questão de saúde. Há também nas documentações encontradas sobre este jogo, maior foco em relação a pré-produção, focando na modelagem UML, e revalidando a quantidade de vezes necessárias antes de ser passada para o protótipo (onde torna haver mais validações), cremos que isso seja fruto do amadurecimento da equipe e do ganho de experiência com os jogos criados anteriormente. Em nosso gameplay do Gamebook – Guardiões da Floresta vimos que os jogos mantem-se na mesma linha dos outros de manter uma história bem coesa, com minigames bastante contextualizados e desafiadores, o que mantem o jogador cada vez mais interessado em chegar ao final.

4 Fazendinha Matemática Virtual

Neste capítulo apresentaremos todo o processo que foi feito no trabalho, além de todos os recursos utilizados e a organização e uso das ferramentas disponibilizadas pelo Unity. Começaremos pela explicação da idealização do jogo e a divisão em alguns dos vários documentos expostos no referencial teórico, frisando diferenciação do jogo analógico, que serviu de base para o trabalho. Após isso, mostraremos como foi a modelagem de entidades e a organização de banco de dados (percebe-se assim que optamos por uma estrutura bottom-up para o projeto). Logo em seguida pontuaremos a estrutura MVC feita no código, ressaltando algumas classes principais e ferramentas que utilizamos. Finalizamos então com a divisão das cenas (que em resumo dividem o jogo) e explicamos como cada uma foi feita e integrada ao todo.

4.1 Idealização da Fazendinha Matemática virtualizada

Este trabalho tem como ponto de partida o projeto “Fazendinha Matemática”, uma sequência didática que visa potencializar o ensino/aprendizagem das operações básicas da matemática através de um jogo, que se baseia em uma história, um jogo da memória, uma ação de troca e uma série de atividades relacionadas às trocas. Essa sequência didática já vem a muito sendo testada em diversas escolas e já se provou ser eficaz em melhorar o desempenho dos alunos durante as aulas de matemática, por isso a idealizadora do projeto, Profa. Dra. Tânia Gusmão, teve a ideia de transformar toda essa sequência em um jogo eletrônico para que pudesse ver o impacto que isso poderia trazer aos alunos, visto a revolução tecnológica presente nos dias atuais e a proximidade que essa nova geração tem com esses equipamentos. Em parceria com o discente e orientador do presente trabalho, além de alunos dos cursos de Matemática, Pedagogia e Ciência da Computação da Universidade Estadual do Sudoeste da Bahia (UESB), foi pensado em como o jogo deveria ser na plataforma eletrônica, visto que a mudança de “mídia” alteraria a experiência que o jogador/aluno teria com a sequência didática, tomando cuidado para que o jogo não perdesse as características pedagógicas que o tornam uma sequência didática, mas ao mesmo tempo tentando aproxima-lo do modo de interação dos jogos eletrônicos.

Primeiramente precisávamos tomar algumas decisões sobre a base do jogo, dentre elas podemos destacar algumas: o jogo será na plataforma PC, decisão tomada para aproximar o jogo da sala de aula, visto que é o equipamento mais fácil de ter nas escolas públicas brasileiras; o jogo foi desenhado inicialmente para ser *single-player* (jogador único), essa decisão foi tomada para facilitar o desenvolvimento como um todo, visto que ninguém da equipe tinha experiência com jogos eletrônicos, além da experiência de usuário, e a inclusão de um sistema em rede poderia levar um tempo muito grande para ser confeccionado; o jogo precisaria de inicio captar o mais próximo possível a forma como o jogo analógico é, em que isso significa que decidimos modificar o mínimo possível dos elementos que a sequência original possuía para observar como a passagem de mídia iria afetar o desempenho da mesma, e após isso poder acrescentar novos elementos. Mesmo tomando essa decisão, acabamos por acrescentar algumas pequenas coisas não presentes no jogo original, devido a perda de interação social que acarretou da primeira decisão (de torna-lo *single-player*), acrescentamos alguns elementos que podem tornar o jogo mais interessante e com objetivos secundários; e O jogo será no estilo *point-and-click* (apontar e clicar), significando que o *mouse* será praticamente o único dispositivo utilizado pelo jogador para realizar todas as interações com o jogo, tal decisão foi tomada para aproximar o jogo eletrônico da sequência original em que as interações são bem pontuais e não requerem a criação de um personagem, ao menos nesse ponto do projeto.

Após isso foi necessário escolher quais as ferramentas que utilizaríamos para o desenvolvimento do jogo eletrônico. Apesar de ter visto alguns softwares voltados para a documentação, como o *Dundoc*, *Gamescrye* e o *Gambit Projet*, além da ferramenta *Wiki*, também bastante utilizada para essa etapa, optamos por usar um editor de texto simples, devido aos poucos elementos que estariam presentes nessa primeira versão do jogo. Após isso teríamos de eleger uma *engine* de jogos que seria a principal ferramenta de desenvolvimento, e após analisar algumas, optamos pela Unity3D, pois a mesma é largamente usada e com isso teríamos como tirar vários tipos de dúvida em *sites*, blogs, fóruns e afins, e também é bastante completa, possuindo todos os recursos que necessitaríamos para o desenvolvimento, com exceção de um editor de imagens. O Unity3D em si possui dois editores de código que já vêm instalados com o pacote, são eles: VisualStudio e MonoDevelop, optamos por utilizar o MonoDevelop por ser mais simples e leve para o desenvolvimento. Outra questão foi a questão das imagens do jogo, em que parte foi produzida exclusivamente para o jogo, e a outra parte foi captada de repositórios da internet, em que algumas tiveram de serem editadas com o uso do Paint e do Photoscape. Em relação aos áudios utilizados, a maior parte foi

importada do repositório do próprio Unity3D, o *Assets Store*, em que qualquer desenvolvedor pode postar áudios, imagens, modelos 3D, *sprites*, dentre vários outros elementos de jogos, podendo ser fixados preços ou liberados gratuitamente, e uma outra parte (principalmente as músicas de fundo de cada tela) foi buscada na internet, colocando os devidos créditos por todo o material utilizado de outros.

4.2 Documentação do Jogo

4.2.1 Escopo do Jogo

4.2.1.1 Plataforma

A Plataforma inicialmente será voltada para PC apenas.

4.2.1.2 Jogadores

Apenas um jogador.

4.2.1.3 Gênero

Point-and-Click

4.2.1.4 High Concept

A Fazendinha Matemática Virtual é um jogo que fará do jogador um aspirante a fazendeiro, em que o mesmo deverá ajudar um Vilarejo, onde aconteceu um terrível desastre e um dos moradores, Seu Toninho, perdeu sua fazenda com todos os seus animais. O novo fazendeiro deverá ajudar o Seu Toninho capturando pintinhos coloridos na floresta do Vilarejo e trocando-os por animais e mercadorias dos outros fazendeiros.

4.2.1.5 Objetivo

O objetivo do fazendeiro é a partir da captura dos pintinhos e das trocas, conseguir conquistar um lote de terra, para assim ele e o Seu Toninho, conseguirem conquistar a fazenda que perderam com o desastre. Como o jogo também tem o cunho educacional, o jogo também buscará que o jogador consiga compreender completamente como as trocas são feitas e porquê são feitas dessa forma, para que ele possa criar associações quando for lidar com as operações básicas da matemática, tendo mais habilidade na hora de executar os algoritmos dessas operações.

4.2.1.6 Recursos

Para realizar a captura dos pintinhos, o jogador deverá jogar um jogo da memória, em que ele terá um tempo para encontrar quantos pares de pintinhos forem possíveis. Após o jogo da captura, ele deverá ir ao mercado de trocas, onde escolherá de um lado os animais que tem e que deseja trocar, e escolher do outro lado o fazendeiro com o qual deseja fazer a troca. Cada um desses fazendeiros cuidará de apenas um tipo de animal e possuirá virtualmente uma quantidade infinita do mesmo. Após escolher os animais de seu lado, e do lado do fazendeiro, ele tentará fazer a troca, e o jogo dará como resposta se o mesmo acertou ou não. O jogador ganhará experiência a cada novo animal conquistado, pintinhos capturados e objetivos ocultos realizados (sendo estes últimos certas ações que mostrarão que o mesmo está ganhando domínio nas trocas). Com a experiência o jogador poderá passar de nível. A cada nível conquistado o jogador será desafiado a responder algumas perguntas sobre as trocas, para que ganhe cada vez mais habilidade com elas.

4.2.2 Game Design Document (GDD)

4.2.2.1 Visão Geral Essencial

4.2.2.1.1 Resumo

O jogador deverá ajudar o fazendeiro Seu Toninho, morador de um Vilarejo, a recuperar sua fazenda e animais, que foram perdidos devido a uma forte ventania que por ali passou. Ele deverá então caçar pintinhos coloridos na floresta para fazer trocas com os fazendeiros e assim poder recuperar o lote de terra para que ele e seu Toninho possam começar a reconstruir o que perderam.

4.2.2.1.2 Aspectos Fundamentais

O jogador terá três ações principais. A primeira é a caça dos pintinhos, em que ele deverá jogar um jogo da memória em que as cartas serão os pintinhos a serem capturados, par a par. A segunda ação é a do Mercado de Trocas, onde o mesmo deverá selecionar os animais que acumulou e os animais dos fazendeiros para que as trocas sejam feitas. A terceira é, ao passar de nível, realizar atividades em que o mesmo será desafiado em relação ao seu conhecimento da tabela de trocas do jogo.

4.2.2.1.3 Golden nuggets

O principal aspecto de destaque nesse jogo é o seu fundo educacional, que busca por meio de habilidades ganhas com as trocas de animais, fazer o jogador relacionar e entender como funcionam as operações básicas da matemática e seus algoritmos.

4.2.2.1.4 História do Game

A História é sobre um Vilarejo em que havia vários fazendeiros e cada um cuidava de vários animais e um fazendeiro, Seu Toninho, que cuidava de vários e sua fazenda era muito querida por todos. Porém em um dia de muita ventania e furacões, a fazenda de seu Toninho foi completamente arrasada, e o mesmo perdeu todos os seus animais e a própria fazenda. Todos os moradores do Vilarejo se abateram em tristeza, em solidariedade ao seu Toninho. Seu Toninho, junto com o jogador principal e seu cachorro Pipo, vários pintinhos coloridos que vieram trazidos da ventania e foram parar na floresta do Vilarejo. Ele então tem uma ideia de realizar um sistema de trocas de animais, em que os animais maiores valeriam dois animais menores, em uma tabela hierárquica de oito níveis (pintinho, galinha, saco de milho, porco, ovelha, cavalo, vaca e lote de terra). Todos os outros fazendeiros então concordam com a ideia e se

cria o mercado de trocas, para que todos pudessem ajudar o Seu Toninho a reconstruir sua fazenda.

4.2.2.2 Objetos essenciais do game

4.2.2.2.1 Personagens

Personagem Principal: o personagem principal será um forasteiro no Vilarejo que será convidado pelo Seu Toninho para ajudar a salvar o que foi perdido com a ventania. Ele terá o nome que o mesmo atribui no inicio do jogo. Ele será requisitado pelo seu Toninho a caçar os pintinhos coloridos e a fazer as trocas, para que assim consigam reviver tanto a fazenda.

Seu Toninho: o Seu Toninho é o fazendeiro que perdeu tudo, ele requisita ao jogador uma ajuda para reerguê-la. Quando o cachorro Pipo encontra os pintinhos coloridos, o seu Toninho é quem tem a ideia de fazer o sistema de trocas.

Dona Gertrudes: é uma adorável senhora que é a principal criadora de galinhas da região. Após a ventania, ela passou a ajudar no mercado de trocas fornecendo galinhas para todos.

Seu Joaquim: o seu Joaquim é um senhor que cuida do milharal no Vilarejo. As galinhas de dona Gertrudes se alimentam desse milho. Ele ajuda a todos fornecendo sacos de milho no mercado de trocas.

Seu Zé: é o fazendeiro que mais possui porcos no Vilarejo e por isso passa a oferecê-los no mercado de trocas após a ventania.

Dona Maria: é uma senhora que todos gostam e cria muitas vacas. Os porcos do seu Zé se alimentam do soro do leite das vacas de Dona Maria. Ela fornece no mercado de trocas suas adoráveis vaquinhas.

Chica: Chica é a criadora de ovelhas da região, fornecendo lã para a roupa de todas por ali. Após a ventania ela as colocou no mercado para que pudesse ajudar seu Toninho.

Miguel: é o irmão de Chica e possui os cavalos mais bonitos de todo o Vilarejo. Ele colocou todos no mercado vendo o quanto as pessoas ficaram tristes após o seu Toninho perder tudo.

4.2.2.2.2 Objetos

Coleção de Pintinhos Coloridos: durante as caçadas (jogos da memória) o jogador se deparará com vários tipos de pintinhos diferentes, incluindo pintinhos especiais que só aparecerão raras vezes. Na parte da "Minha Fazenda", o jogador poderá acompanhar quais desses pintinhos o mesmo já capturou, fazendo com que completar todos os pintinhos seja um desafio a parte para os jogadores.

4.2.2.3 Conflitos e Soluções

São basicamente três os conflitos apresentados no jogo que o jogador deve enfrentar:

4.2.2.3.1 Caça aos Pintinhos

A caça aos pintinhos coloridos é o que permitirá o jogador possuir pintinhos para poder fazer as trocas e avançar no jogo. Na história, a caça aos pintinhos se da em meio à floresta. A primeira caçada que o jogador deve fazer ocorrerá no meio da história inicial, quando o Pipo, o cachorro do Seu Toninho, encontra os pintinhos pela primeira vez. Objetivamente a caçada aos pintinhos é um jogo da memória em que os arbustos representarão as cartas viradas que ao serem clicadas revelarão o pintinho escondido naquele arbusto. Por conta de o jogo ser individual, um jogo da memória ficaria pouco desafiador, já que bastaria o jogador tentar N vezes todas as combinações para que ele sempre virasse todas as cartas, por isso que para diferenciar e aumentar a dificuldade foi acrescentado o elemento do tempo nesse jogo da memória, assim o jogador deve achar os pares de pintinhos o mais rápido possível antes que o tempo acabe. Para virar as cartas basta que o jogador clique nos arbustos que logo revelarão o pintinho escondido, e então ele clicará em outro arbusto, para ver se os pintinhos são idênticos. Caso não o sejam o jogo irá desvirar ambos os pintinhos e permitirá que o jogador faça nova tentativa. A quantidade de cartas a ser virada e o tempo do jogo dependerão do nível que o jogador se encontra no momento: Até o nível cinco (5) o jogo disponibilizará 12 cartas (6 pares de pintinhos) e o tempo de 20

segundos (tempo padrão); Do nível cinco (5) até o nível dez (10), o jogador contará com 18 cartas (9 pares de pintinhos) e o tempo de 30 segundos (tempo padrão); a partir do nível dez (10) o jogador terá 24 cartas (12 pares de pintinhos) e o tempo de 40 segundos para fazer sua caçada.

4.2.2.3.2 Mercado de Trocas

O mercado de trocas é onde o jogador levará os pintinhos capturados para que possa ser trocados pelos animais de valor mais alto, e assim poder chegar ao lote de terra, objetivo principal do jogo. O mercado de trocas contará com três áreas principais: A área de animais do jogador, a área dos fazendeiros e a área de trocas. Na primeira área aparecerão todos os animais com a respectiva quantidade do mesmo que o jogador possui no momento, e ao clicar no animal os animais aparecerão na área embaixo para que possam ser trocados. Caso o jogador possua uma quantidade que ultrapasse o número de casas possíveis (10 casas) o jogo carregará apenas os 10, já que não afetará na troca. Na segunda área estarão os fazendeiros, que aparecerão conforme o jogador tiver os respectivos animais para trocar com eles (por exemplo, o Seu Joaquim que tem os sacos de milho só aparecerá quando o jogador conquistar sua primeira galinha, já que são necessárias duas galinhas para trocar por sacos de milho). Ao clicar no fazendeiro aparecerá dez de seu respectivo animal. Os fazendeiros terão sempre a quantidade máxima de animais, mesmo após as trocas, para que o jogador tenha sempre a condição de poder fazer a troca independente de quantos animais ele possua. A terceira área é a área de trocas, ao clicar nos animais que aparecem nas duas primeiras áreas, os mesmos serão deslocados para a área de trocas e ao ter a quantidade proporcional o jogador clicará no botão de trocar, dessa forma o jogador deve sempre ter em mente a tabela de trocas e a proporcionalidade entre os animais. É possível além das trocas de dois por um (duas galinhas por um saco de milho), também realizar trocas quatro por dois (quatro ovelhas por dois cavalos) entre animais de relação direta, ou fazer trocas quatro por um (quatro sacos de milho por uma ovelha) entre animais de relação indireta. Também será possível fazer uma troca inversa (um cavalo por duas ovelhas), pois embora não seja um progresso para cumprir o objetivo final (conquistar o lote de terra) demonstrará que o jogador consegue ver que a relação não é apenas em um sentido.

4.2.2.3.3 Desafios

Os desafios aparecerão sempre que o jogador passar de nível. Neles o jogador será desafiado com uma quantidade aleatória de um determinado animal e por quatro perguntas. A primeira pergunta é a quantidade dos animais que aparecem na tela (lembrando que o jogo é voltado para crianças das séries iniciais). A segunda pergunta é pelo que se pode trocar diretamente (os animais de quantidade aleatória que apareceram inicialmente), onde aparecerá um botão para cada um dos animais da tabela e o jogador deverá clicar no que esta diretamente acima ou abaixo do animal que apareceu (tecnicamente ele poderia trocar por qualquer um, porém isso atrapalharia a terceira pergunta por conta da quantidade de animais que poderiam aparecer, logo foi decidido que apenas aqueles relacionados diretamente para cima ou para baixo da tabela é que seria a resposta esperada para essa pergunta). A terceira pergunta é por quantos do animal escolhido na segunda pergunta o jogador poderá trocar pelos animais que apareceram inicialmente, o jogador deverá selecionar os animais de dez que aparecerão para ele. A quarta e última pergunta é referente a se sobra algum animal na troca, isso é possível caso haja um número impar de animais que apareceram inicialmente e caso na segunda pergunta o jogador escolher fazer a troca por um animal relacionado acima na tabela (por exemplo, caso apareça inicialmente três galinhas e o jogador selecionar para trocar por saco de milho, ele deverá selecionar apenas um saco de milho e dizer que sobrou uma galinha na troca).

4.2.2.4 Fluxo do Game

O jogo inicia com a História inicial, em que o Seu Toninho aparece se apresentando e apresentando os demais fazendeiros do Vilarejo, mostrando como cada um cuidava principalmente de um tipo de animal e ele cuidava um pouco de todos, sendo a sua fazenda uma atração a parte para os moradores, principalmente para as crianças do Vilarejo. Então ele conta sobre o que aconteceu a pouco, uma forte ventania que destrói muita coisa e leva embora todos os animais da sua fazenda, fazendo todos ficarem desolados com a situação. Seu Toninho então pede ajuda ao personagem principal e pergunta seu nome, após o jogador digitar como quer ser chamado começará de fato o jogo.

A primeira coisa que ocorre é Seu Toninho estar pensando junto ao jogador como resolver o problema do Vilarejo quando ele percebe que o cachorro dele, o Pipo esta agindo estranhamente, então ambos o seguem para a floresta. Eles percebem que Pipo percebeu algo

nos arbustos e eles então devem investigar o que é, e ai acontece a primeira caçada do jogo, que é o jogo da memória. Porém, ao contrário da caçada comum, esta primeira não irá ter o tempo regressivo para o jogador, pois servirá como tutorial e para o jogador se acostumar. Ele então virará três pares de pintinhos (dos 6 pares que estão ali) e o jogo imediatamente voltará para a história, em que eles admiram os pintinhos que conseguiram e o Seu Toninho diz ter uma ideia e fala para eles se encontrarem com os outros fazendeiros para discuti-la.

Seu Toninho então explica sua ideia para todos do Vilarejo, dizendo que eles poderiam montar um sistema de troca de animais. O sistema de troca é baseado nos animais e em seu valor, no qual dois animais de menor valor seriam equivalentes a um de maior valor, em uma sequência hierárquica, na seguinte forma: Pintinho – Galinha – Saco de Milho – Porco – Ovelha – Cavalo – Vaca – Lote de Terra. Cada par desses animais valeriam um único da casa seguinte da sequência, assim o seu Toninho poderia recuperar os animais e a própria fazenda capturando pintinhos e trocando com os outros fazendeiros. A ideia é bem recebida por todos, que decidem aderir ao sistema de trocas. Logo então seu Toninho pede ajuda novamente ao jogador, para que o mesmo possa caçar os pintinhos e fazer as trocas, para que eles pudessem recuperar o que havia sido perdido.

O jogador então se dirige ao mercado de trocas a partir do mapa do Vilarejo, para que possa trocar os seis pintinhos que ele havia conseguido inicialmente. A Dona Gertrudes então se apresenta ao mesmo, dizendo ela ser a criadora de galinhas e que ela estaria disponível caso o jogador precisasse de alguma. Após ele fazer a troca dos pintinhos pelas galinhas, o personagem do Seu Joaquim se apresentará com os sacos de milho para que o jogador possa trocar, e assim acontecerá sempre que o jogador conseguir um animal novo. Após realizar todas as trocas o jogador fica liberado para ir novamente caçar os pintinhos, para que possa realizar novas trocas.

Além desse fluxo principal o jogador também poderá visitar a “Minha Fazenda”, onde encontrará todos os seus animais (ainda em um espaço sem fazenda, pois o mesmo está buscando conquistar o lote de terra) e poderá ver também os tipos de pintinho que o mesmo já capturou, para que possa completar a coleção.

Ao alcançar o Lote de Terra, o Seu Toninho agradecerá e dirá que ali era apenas o começo e que a partir da fazenda recuperada, eles poderiam ter várias outras ideias que ajudariam e muito o Vilarejo. Então o jogo é encerrado trazendo uma estatística de como foi o desempenho do jogador durante o jogo.

4.2.2.5 Referências

Este jogo é baseado em um projeto coordenado pela Profa. Dra. Tânia Gusmão para o ensino e aprendizagem das noções de operações básicas da matemática, aplicado em colégios municipais de Vitória da Conquista. A Fazendinha Matemática, como é chamada, é totalmente analógica, sendo que a história do Vilarejo é contada pelo professor em sala de aula para então os alunos sentarem-se em grupos para jogar o jogo da memória (com cartas físicas) e "capturar" os pintinhos e então trocarem os mesmos com o professor no "mercado", eventualmente também fazendo atividades e sendo "desafiados" pelo professor à responder questões relacionadas às trocas. Este trabalho no formato digital nasceu justamente do desejo da professora Tânia em se atualizar visto a abrangência e popularidade que ganham os jogos digitais hoje em dia, além de um maior controle e possibilidades que as ferramentas eletrônicas possuem em relação aos jogos "tradicionais". Um elemento que foi acrescentado apenas para dar maior ludicidade ao jogo foi a coleção de pintinhos coloridos, uma ideia tirada do jogo Pokemon Go, um dos maiores sucessos atuais em termos de jogos.

4.2.3 Dados do Game

4.2.3.1 Roteiro da história do jogo

Cena 1 (Seu Toninho aparece para introduzir o jogador no Vilarejo)

- Em um lugar não muito distante, existe um arraial muito alegre e tranquilo;
- Meu nome é Toninho, e eu moro nesse lugar. Aqui vivem várias pessoas, e cada uma tem sua forma de viver.

Cena 2 (Aparece a dona Gertrudes, cuidando de suas galinhas)

- A dona Gertrudes gosta de criar galinhas. As crianças adoram correr atrás delas.

Cena 3 (Aparece seu Joaquim, no milharal)

- As galinhas da Dona Gertrudes se alimentam do milho das terras do seu Joaquim.

Cena 4 (Aparece Seu Zé com seus porcos)

- Outro morador é Seu Zé, criador de porcos. Os porquinhos do Seu Zé são tão engraçadinhos.

Cena 5 (Aparece Dona Maria em meio às suas vacas)

- Seus porquinhos se alimentam do soro do leite das vaquinhas de Dona Maria. Elas são todas pintadinhas.

Cena 6 (Aparece a Chica, das ovelhas)

- Tem a Chica que cria ovelhas, e fornece a todos lã para suas roupas.

Cena 7 (Aparece Miguel, com seus cavalos)

- Seu vizinho é seu irmão Miguel, que tem os cavalinhos mais bonitos da região.

Cena 8 (Aparece Seu Toninho cuidando de sua fazenda)

- E esta era a minha fazenda. Eu criava de tudo um pouco. Galinha, porco, ovelha, vaca e cavalo.
- As crianças adoravam esta fazenda e sempre viam brincar com meus animais. Todos éramos muito felizes.

Cena 9 (Mostra a cena do tempo escurecendo no Vilarejo)

- Tudo estava na mais perfeita harmonia quando de repente o tempo escureceu.

Cena 10 (Mostra o tempo)

- Uma ventania muito forte e com chuvas desceu tomando conta da minha fazenda. Todos se assustaram com o tempo.
- A ventania se tornou um grande furacão que passou girando e girando, crescendo e crescendo e engolindo tudo que via pela frente.

Cena 11 (Mostra os fazendeiros desolados)

- Quando tudo se acalmou, todos viram o ocorrido. Foi uma tristeza só! Nada sobrou da fazenda.

Cena 12 (Retorna ao Seu Toninho inicial)

- Estou muito triste por ter perdido todos os meus animaizinhos.
- Será que você poderia nos ajudar a reconstruir minha fazenda?
- Primeiro me diga seu nome.

Cena 13 (Aparece Seu Toninho com o cachorro Pipo)

- Agora temos de pensar em um jeito de salvar o Vilarejo;
- Olha só o Pipo, ele está agindo muito estranho;
- Parece que ele quer que nós o sigamos;
- Vamos lá!

Cena 14 (Mostra na floresta alguns arbustos)

- Parece que há algo escondido naqueles arbustos. Temos de investigar o que é
- O Pipo irá te ajudar. Você consegue!

(Jogo da memória da caçada aos pintinhos, modo tutorial)

Cena 15 (Mostra os pintinhos conseguidos na caça)

- Muito bem! Conseguimos alguns pintinhos. Agora, o que faremos com eles?
- Tive uma grande ideia. Vamos reunir os moradores, acho que conseguiremos salvar a fazenda e recuperar os animais.

Cena 16 (Mostra uma reunião dos fazendeiros do Vilarejo)

- Meus amigos e amigas, encontramos estes pintinhos coloridos logo ali na floresta e tivemos uma ideia;
- Que tal se fizermos um sistema de trocas de animais? Dois por um, essa é a regra. Do de menor valor, para o de maior valor;
- Assim vocês poderiam me ajudar a recuperar a fazenda que tanta alegria trazia a todos, e também poderão trocar animais entre si;
- Podemos utilizar a seguinte tabela para auxiliar nossas trocas...

Cena 17 (Mostra a tabela de trocas)

- É uma tabela simples, em que se pode trocar dois pintinhos por uma galinha...
- Duas galinhas por um saco de milho, dois sacos de milho por um porco, e assim por diante;
- Decorrem ela e sempre que precisar a consultem para que possamos acertar cada vez mais trocas.

Cena 18 (Retorna para a reunião de fazendeiros)

- E então pessoal, podemos fazer assim?
- Dona Maria: Claro Seu Toninho, gostei muito da ideia;
- Seu Zé: Opa, uma luz no fim do túnel. Vamos ajuda-lo Toninho, começaremos os preparativos agora mesmo.
- Miguel: Vamos lá!

Cena 19 (Retorna para o Seu Toninho conversando com o jogador)

- Pronto. Agora conseguimos convencer os moradores, acredito que a fazenda será salva;
- Gostaria de te pedir que continue caçando os pintinhos para irmos ao mercado troca-los com os outros;
- Se conseguirmos pintinhos suficientes, conseguiremos reerguer a fazenda.
- Espero contar com seu apoio.

(Jogo)

Cena 20 (Cena final. Aparece Seu Toninho para agradecer após a conquista do Lote de Terra)

- Muito bem caro amigo! Você conquistou o Lote de Terra que tanto precisávamos;
- Agora podemos reconstruir nossa fazenda, que também já é sua;
- Muito obrigado pelo esforço. Ainda iremos nos encontrar bastante, tenho muitas ideias para que possamos melhorar o nosso Vilarejo;
- Fique bem.

(Encerramento)

4.2.3.2 Fala de Apresentação dos personagens

Dona Gertrudes: Bem vindo ao Mercado de Trocas. Eu sou a Dona Gertrudes, a cuidadora de galinhas da região. Caso queira uma é só me procurar. Estarei te esperando.

Seu Joaquim: Olá pequeno(a) fazendeiro(a). Sou o seu Joaquim e cuido dos milhos da região. Se precisar de alguns sacos de milho, não hesite em me pedir.

Seu Zé: Tudo bem meu(inha) jovem? Sou o seu Zé e tenho aqui vários porquinhos se tiver interesse. Vamos trocar?

Chica: Olá, como vai? Vejo que esta fazendo várias trocas. Parabéns. Agora já pode pegar algumas ovelhas, vamos lá.

Miguel: E ai amiguinho(a), sou o Miguel. Meus cavalinhos vão te ajudar a cumprir seu objetivo. Já estou ótimo nas trocas, e você?

Dona Maria: Sou a Maria, ou Dona Maria se preferir. Consiga duas de minhas vaquinhas e o lote de terra é seu. Mais algumas trocas e vai reconquistar a fazenda. Você consegue.

Seu Toninho: Finalmente poderemos recuperar a fazenda. Achei esses lotes de terra que poderemos trocar. Você é demais.

4.2.3.3 Configuração do jogo por nível

Até o nível 5:

- Quantidade de Cartas com suas respectivas posições na tela: 12 cartas;
- Acréscimo no tempo da caçada dos pintinhos: 0 segundos;
- Chances de aparecer algum pintinho de segunda classe: 1 em 50;
- Chances de aparecer algum pintinho de terceira classe: 1 em 100;

Do nível 5 até o nível 10:

- Quantidade de Cartas com suas respectivas posições na tela: 18 cartas;
- Acréscimo no tempo da caçada dos pintinhos: 20 segundos;
- Chances de aparecer algum pintinho de segunda classe: 1 em 20;
- Chances de aparecer algum pintinho de terceira classe: 1 em 40;

A partir do nível 10:

- Quantidade de Cartas com suas respectivas posições na tela: 24 cartas;
- Acréscimo no tempo da caçada dos pintinhos: 40 segundos;
- Chances de aparecer algum pintinho de segunda classe: 1 em 10;
- Chances de aparecer algum pintinho de terceira classe: 1 em 20;

4.2.4 Lista de Recursos

Este jogo ainda esta em desenvolvimento e por isso estamos aos poucos construindo tanto a mecânica e a história, quanto à estética, por isso precisamos mesclar as imagens próprias do jogo com imagens buscadas de outras fontes. As imagens principais do jogo (Figura 52) foram desenvolvidas dentro do próprio grupo, sendo as mesmas os fazendeiros, uma parte dos pintinhos, as tábuas e fundo dos botões, bem como as placas utilizadas no menu inicial.

Figura 52 - Imagens produzidas para o jogo

Algumas das imagens (Figura 53) foram retiradas dos desenhos originais da fazendinha matemática, como os animais e o *background* inicial.

Figura 53 - Imagens retiradas da fazendinha original

Outras imagens (Figura 54) foram retiradas de outros repositórios encontrados na Internet como o (FREE PIK, 2017), (KIDS PLAY COLOR, 2017), (ROLOFF FARM, 2017), (GET COLORING PAGES, 2017), (PRINTABLE FREE COLORING, 2017), (HAVE GO NAVEGAN, 2017) e (DRAWING NOW, 2017).

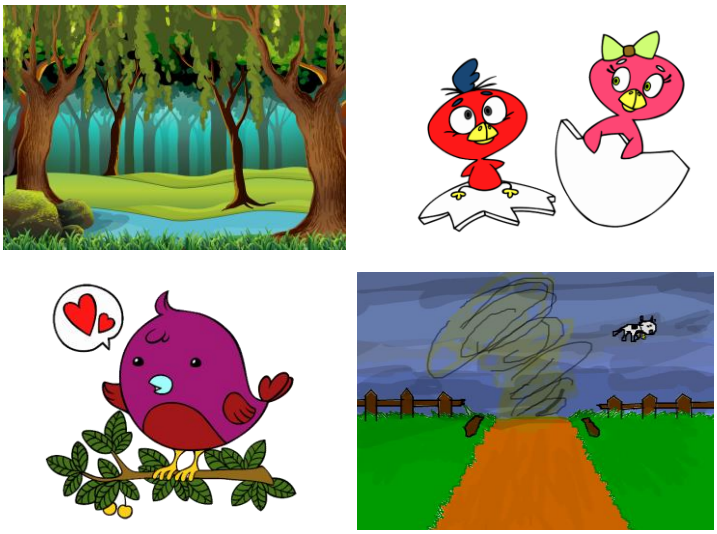


Figura 54 - Imagens de outras fontes

Outro recurso bastante utilizado no jogo foram as músicas e os sons. Para o uso de tais recursos foi utilizado o repositório do próprio Unity3D, o *Assets Store*, além de músicas de alguns vídeos do YoutTube.

4.3 Modelagem

4.3.1 Idealização das entidades para o jogo digital

Ao montar qualquer sistema é necessário montar uma estrutura, e nos sistemas voltados a orientação a objeto, o modelo de entidade e relacionamento é o mais natural. Na fazendinha matemática digital podemos pensar em algumas entidades principais: Jogador, Jogo, Animais, Cartas, Atividades, Objetivos, etc.

Inicialmente dividimos a entidade de jogo em duas: **Jogo** e **JogoConfiguração**, para separar as informações que poderão, futuramente, ser configuradas pelo professor. A entidade que representará as informações do jogador ficaram na entidade de **Registro**, pois mais tarde o jogador poderá salvar mais de um jogo ao mesmo tempo. Para representar os animais foi criado três entidades: **Item**, **Prototipo_Item** e **Tipo_Item**. O Tipo de Item representa os animais apenas (inicialmente), nas próximas atualizações serão criadas outras classes do que poderá ser trocado no mercado, como será explicado no tópico de Conclusões e Trabalhos Futuros. O protótipo item irá guardar os animais em si: Pintinho, Galinha, Porco, etc. Já a entidade de itens guardará os animais separados individualmente, em relação aos pintinhos, irá registrar aqueles pintinhos que já foram capturados para a coleção.

Os **Objetivos** e as **Atividades** (desafios) também se dividem com entidades de tipo, porém com a função de registrar explicações teóricas em relação a cada grupo das entidades mencionadas em relação a questão pedagógica. Os objetivos possuem, além disso, uma tabela extra chamada de **Objetivo_Principal_Secundario**, servindo a mesma para relacionar um objetivo principal (de mudança de fase) em relação aos objetivos complementares, necessário para chegar no mesmo, sendo isso aplicado à conquista do lote de terra (objetivo principal) sendo necessário antes passar pelos objetivos complementares, que são os de capturar os outros animais. Já as atividades possuem a entidade de **Etapa_Atividade**, que registra o progresso do jogador ao responder as atividades (inicialmente apenas uma atividade esta no jogo). Na atividade que se encontra no jogo, essas etapas serão cada uma das perguntas que são feitas ao jogador.

Para abranger os múltiplos registros, foi criado também a relação da entidade de Registro com os itens, objetivos e atividades, nas entidades de **Registro_Item**, **Registro_Objetivo** e **Registro_Atividade.** A relação de registro com item ainda possui a entidade **Registro_Quantidade_Item**, pois enquanto a Registro_Item marca se determinado item (principalmente pintinho) já foi capturado, com um atributo lógico (Sim ou Não), a entidade de Registro_Quantidade_Item marca a quantidade de itens que o jogador possui no momento. Já a relação de registro com atividade possui além a entidade de Registro_Etapa_Atividade, para guardar, como já dito, o progresso do jogador em cada uma das partes da atividade.

Duas outras entidades foram criadas com intuito de apenas auxiliar. A primeira é a de **Configuracao_Cartas** que registra como as cartas serão dispostas a depender do nível do jogador, e a segunda é a **Generator** que serve apenas para registrar o próximo valor da chave-

primária no banco de dados. Na próxima sessão é mostrado como ficou o Diagrama de Entidade-Relacionamento.

4.3.2 Entidade Relacionamento

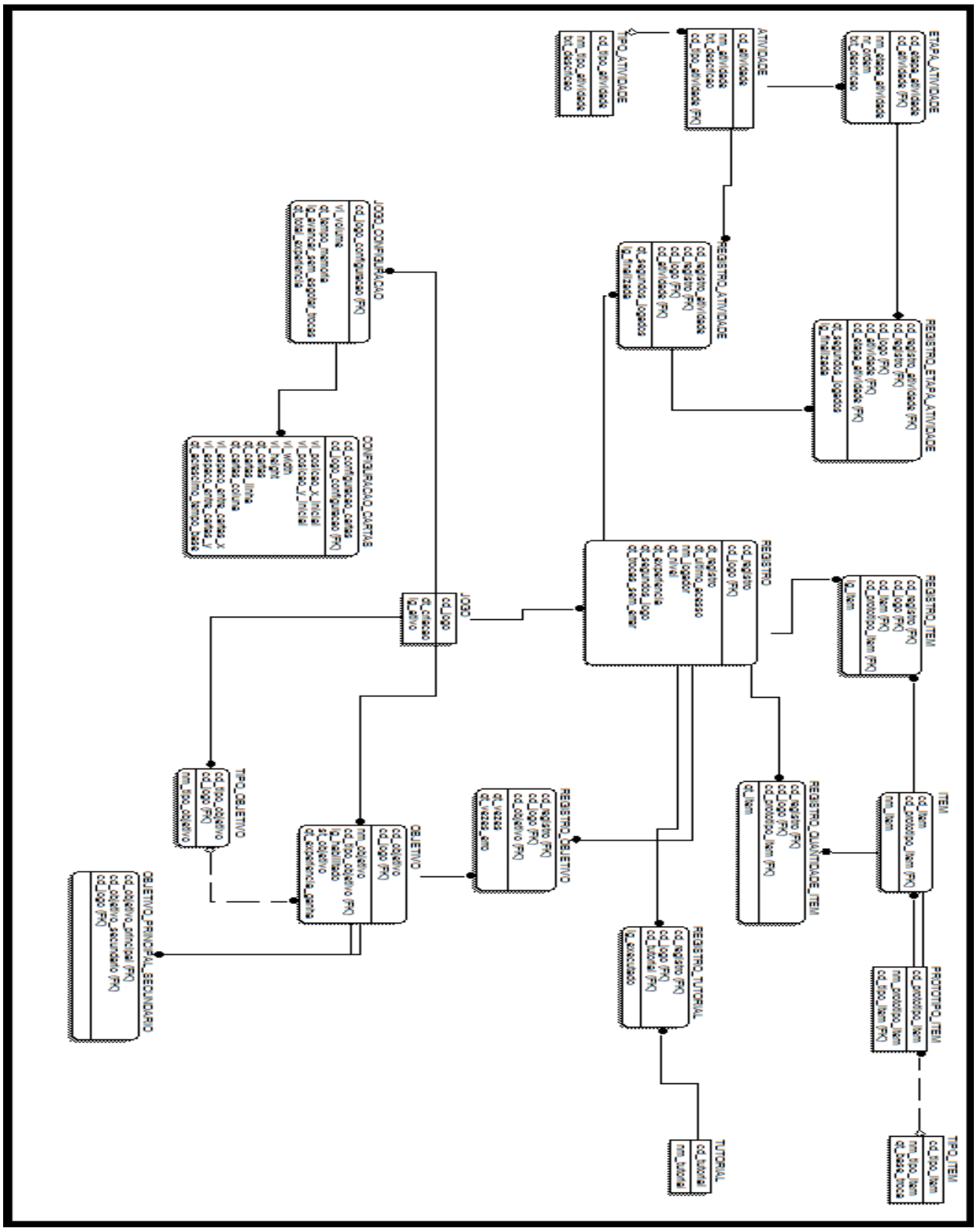


Figura 55 - Diagrama Entidade-Relacionamento do jogo

4.3.3 Conversão para o Banco de Dados

Para o controle de banco de dados foi utilizado o SGBD SQLite, por ser um sistema leve e simples, além de sua integração já bastante usual para jogos feitos em Unity facilitar a procura por informações específicas na internet. A ferramenta utilizada para o controle do SGBD foi o SQLite Expert Personal 4.2 (*free licence*), que possui uma interface de boa usabilidade e os principais recursos necessários para a manutenção de um banco de dados: Editor SQL; Controle dos Dados via interface, podendo adicionar, alterar e excluir dados por ali mesmo; Controle de DDL; e Controle de Design dos campos da tabela, permitindo ao usuário criar suas tabelas fora da codificação DDL, apenas por meio de interface.

4.4 Organização do Código

Para a codificação do jogo no Unity3D foi utilizado a linguagem C#, preterindo a opção do JavaScript por conta da primeira ter uma referência muito maior de outros projetos na internet, sendo assim mais fácil pesquisar soluções e comparar erros e dificuldades, e também pela sua estruturação interna voltada à orientação a objeto. Os scripts foram todos colocados dentro da pasta “Scripts” do projeto e se dividem em seis pastas: **controller**, **dados**, **dbControl**, **model**, **tools** e **view** (Figura 56), cada uma delas com uma função diferente dando estrutura para o funcionamento geral do jogo. Nas subseções seguintes abordaremos como os scripts foram utilizados para o controle do comportamento dos objetos do jogo bem como é sua interação com o banco de dados.

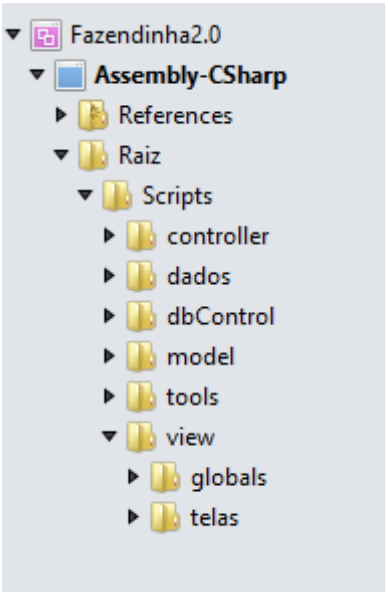


Figura 56 - Organização das pastas principais dos scripts do projeto, no programa MonoDevelop

4.4.1 Arquitetura MVC

4.4.1.1 Model

4.4.1.1.1 Classes de interação com Banco de Dados

Inicialmente necessitamos incluir no projeto (na pasta "Plugins") três arquivos: **Mono.Data.Sqlite.dll**, **sqlite3.dll** e **system.data.dll**. Estes três arquivos são bibliotecas dinâmicas, que fornecem os métodos necessários para que o Unity possa trabalhar com o SQLite. A Figura 8 mostrará como é feito o acesso ao banco de dados a partir de uma classe de acesso direto e outra classe que reúne os métodos CRUD (*create-read-update*-delete), usando assim os conceitos de encapsulamento. A primeira das duas classes base para a estrutura dessa comunicação entre o Jogo e o Banco de Dados é chamada de **DbAcess.cs**, sendo ela quem indicará onde se encontra o arquivo de banco de dados (no caso desse jogo é chamado de Fazendinha20DB) e se apoiará em dois métodos principais: **executeCommand** e **returnQuery**. O executeCommand realiza uma ação DML (*Data Manipulation Language*) sobre o banco de dados, ou seja, ações que modificam os dados e não a estrutura do mesmo, como comandos INSERT, UPDATE e DELETE. Já o método returnQuery realiza uma ação DQL (*Data Query Language*), onde a mesma consulta dados do banco de dados, sem altera-

lo, com o comando de SELECT. A segunda classe que faz essa estrutura para a interação do Banco de Dados é chamada de **Dao.cs**, ela sendo a classe que normalmente conversará com o DbAcess para realizar os comandos para o Banco de Dados, sendo dividida basicamente nos seguintes métodos:

- Insert: Insere dados em uma determinada tabela;
- Update: Atualiza dados de um registro de uma tabela;
- Delete: Exclui um registro do Banco de Dados;
- Get: Busca um determinado registro de uma tabela no Banco de Dados;
- GetAll: Busca todos os registros de uma determinada tabela;
- DeleteAll: Exclui todos os registros de uma tabela no Banco de Dados;
- Find: Busca alguns registros específicos de uma tabela do Banco de Dados, seguindo alguns critérios específicos;
- GetNextCode: Retorna o próximo código a ser utilizado como chave de um determinado registro.

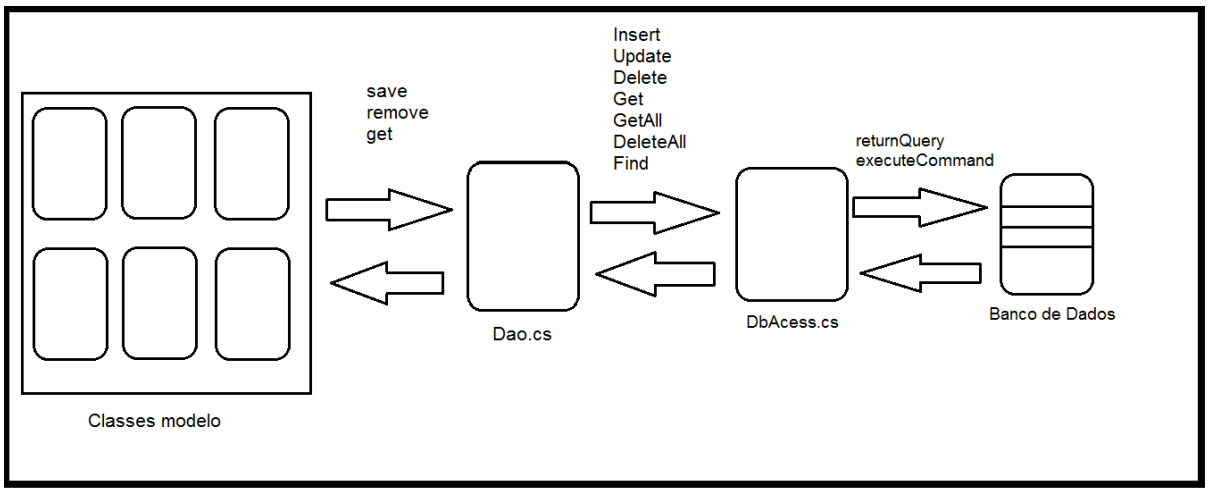


Figura 57 - Fluxograma de acesso ao Banco de Dados

4.4.1.1.2 Classes principais do modelo

As classes principais representam as entidades do modelo de entidade-relacionamento bem como as tabelas do Banco de Dados. Elas são a base para a Orientação a Objeto, possuindo atributos e métodos para as trocas de mensagem entre as mesmas e entre os componentes de tela do jogo. Essas classes são divididas basicamente em: **Atributos**,

Construtores, **Métodos Acessores** (*Getters e Setters*), **Métodos CRUD** (*Create-Read-Update-Delete*) e **Métodos Auxiliares**. Os métodos CRUD é que se relacionam com a classe Dao.cs, mencionada na subseção anterior, e possui os métodos:

- Save (Figura 58): Realiza o salvamento do registro em questão (o registro representado pelo objeto que esta chamando o método) analisando se a chave-primária esta ou não vazia. Caso esteja vazia, indica que o registro ainda não existe no banco de dados, usando o Dao.insert, caso já haja algum valor, o sistema irá entender que o registro foi previamente selecionado do banco de dados, usando o Dao.update;
- Remove: Realiza a retirada do registro no banco de dados, usando o Dao.delete;
- Get: Faz a busca do registro no banco de dados e popula os atributos do objeto com o valor do registro.

Já os métodos auxiliares são:

- ToString: que imprime no console o valor dos atributos do objeto;
- Clone: Onde um novo objeto é criado com os mesmos valores de atributo do objeto original;
- GetItensDataBase: Usado para converter os atributos do objeto em um array compreensível aos métodos de acesso ao banco de dados, para que se possa realizar consultar, inserções ou alterações de dados com os atributos do objeto.

```
67      /// <summary>
68      /// Método que salvará o objeto de Atividade no banco de dados
69      /// </summary>
70      override public Result save(){
71          int ret = 0;
72          //Caso a chave primária esteja vazia, então fará um insert no objeto, colocando a chave gerada no campo
73          //caso contrário, fará um update, atualizando os dados no banco
74          if (getCdAtividade () == 0) {
75              setCdAtividade (Dao.getNextCode ("atividade", "cd_atividade"));
76              ret = Dao.insert ("atividade", getItensDataBase ());
77          }
78          else
79              ret = Dao.update ("atividade", getItensDataBase ());
80
81          return new Result (ret, (ret <= 0 ? "Erro ao tentar salvar atividade" : "Sucesso ao salvar atividade"));
82      }
83
```

Figura 58 - Método save da classe Atividade.cs

4.4.1.1.3 Classes Jogo e Jogo Configuração

As classes Jogo e JogoConfiguração possuem uma importância a parte por representarem informações de configuração que foram utilizadas em vários pontos do código. Dentre essas informações estão informações comuns como o **volume de jogo**. Outras informações, no entanto, são de cunho educacional, pois a ideia inicial é que o jogo possa ser utilizado por professores e professoras dentro da sala de aula, orientando os alunos enquanto jogam e também configurando o jogo de cada um. Assim há também a informação sobre a quantidade de **tempo inicial do jogo da memória**, que poderá ser aumentada ou diminuída com base na turma e aluno que estiver jogando na hora. Outro item configurável é a **condição para continuar caçando os pintinhos**, pois em nossas pesquisas sobre o jogo analógico, aplicado nas escolas por diversos professores e professoras, nos foi relatado que a depender do professor havia duas formas de aplicar as rodadas de caça e troca, uma em que os alunos poderiam jogar repetidas vezes o jogo da memória e só então, com uma grande quantidade de pintinhos, troca-los no mercado, e outra forma em que a cada caçada os alunos deveriam fazer todas as trocas possíveis para então continuar. Essa proposta de flexibilidade foi absorvida pelo jogo digital, permitindo o professor seguir tanto por um caminho quanto pelo outro, bastando configurar o jogo do aluno para tal.

```
17          if (prefabMusicaFundo != null) {
18              setMusicaFundo (Instantiate (prefabMusicaFundo));
19              getMusicaFundo ().volume *= (float)Jogo.getConfiguracao ().getVlVolume ();
20          }
```

Figura 59 - Exemplo de uso de JogoConfiguração, na classe View.cs, regulando o volume de um AudioSource

4.4.1.1.4 Classe Registro

A classe de Registro, assim como Jogo e JogoConfiguracao, também possui uma representação forte em todo o código pois a mesma guardará todo o estado do jogador no jogo e não apenas os campos que fazem parte da tabela Registro, mas também de todas as outras tabelas relacionadas (RegistroItem, RegistroQuantidadeItem, RegistroObjetivo, RegistroAtividade). Esse modo de organização foi escolhido para se evitar chamar muitas classes para obter as informações referentes aos ganhos do jogador, assim sempre que se quiser saber quantos pintinhos o jogador já capturou, recorre-se ao Registro; do mesmo modo, se queira saber quantas atividades (ou desafios) respondidos corretamente o jogador já

realizou, também recorrer-se ao registro. Deste modo a organização do código fica mais simples e intuitiva.

```
172                 Registro.getRegistro ().setQtTrocasSemErrar (Registro.getRegistro ().getQtTrocasSemErrar() + 1);
173                 Result result = Registro.getRegistro ().save ();
174                 if (result.getCode () < 0) {
175                     Debug.LogError (result.getMessage ());
176                 }
```

Figura 60 - Exemplo 1 do uso de Registro, atualizando o valor de quantidade de trocas sem errar, ao fazer uma troca correta, na classe Troca.cs

```
114     /// <summary>
115     /// Busca o tempo médio usado pelo jogador para realizar os desafios
116     /// </summary>
117     /// <returns>A média de tempo em forma de texto.</returns>
118     public static string getDesafiosRealizadosMediaTempo(){
119
120         //Busca todas as atividades feitas pelo jogador
121         List<RegistroAtividade> conjuntoRegistroAtividade = Registro.getRegistro ().getAtividades ();
122
123         int qtMinutosJogados = 0;
124         int qtSegundosJogados = 0;
125
126         //Soma todos os tempos marcados de cada atividade
127         int tempoQtSegundosJogados = 0;
128         foreach (RegistroAtividade registroAtividade in conjuntoRegistroAtividade) {
129             tempoQtSegundosJogados += registroAtividade.getQtSegundosJogados ();
130         }
131
132         //Tira a média de tempo (em segundos) a partir de uma divisão pela quantidade de atividades buscadas
133         qtSegundosJogados = tempoQtSegundosJogados / (conjuntoRegistroAtividade.Count > 0 ? conjuntoRegistroAtividade.Count : 1);
134
135         //Contabiliza os minutos a cada 60 segundos
136         while(qtSegundosJogados > 60){
137             qtMinutosJogados++;
138             qtSegundosJogados -= 60;
139         }
140
141         return qtMinutosJogados + "m " + qtSegundosJogados + "s (média)";
142     }
```

Figura 61 - Exemplo 2 do uso de Registro, onde a função irá retornar a quantidade média de tempo em que o jogador realizou os desafios de atividade, buscando todas as atividades realizadas a partir do uso de Registro

4.4.1.1.5 Uso do Padrão Singleton

Tanto na classe de Jogo quanto da de Registro foi utilizado o padrão Singleton para o controle dessas duas classes em todo o código. Para implementar tal padrão é criado uma variável estática em cada representando a própria classe. Então os construtores de ambas as classes foram declaradas com o modificador de acesso **private**, assim apenas os métodos internos das mesmas poderiam acessar os construtores e fazer novas instancias das classes. Logo então é criado um método estático (getJogo e getRegistro) que verifica a variável representante da classe para caso a mesma seja nula, o método então instancia com o construtor aquela variável e inicializa todos os recursos que a mesma necessita (no caso de

Jogo é realizar a inicialização também das configurações com o JogoConfiguracao, e no caso de Registro é a inicialização das tabelas relacionadas de Atividade, Objetivo e Itens), caso ela não seja nula, o método simplesmente retorna a variável, assim o formato Jogo.getJogo e Registro.getRegistro serão sempre utilizados no código para referenciar a instância única que há dessas classes em toda a execução.

4.4.1.2 Controller

4.4.1.2.1 Classes de Dados

O jogo atualmente possui três classes de dados que representam as falas dos personagens ao se apresentarem pela primeira vez no mercado de trocas (ApresentacaoFazendeiro.cs), as caixas de texto da história inicial (FalasCenas.cs) e os textos de tutorial ao abrir novas telas (MensagensTutorial.cs). Essas classes são acessadas pelas Views e possuem estrutura separada da mesma para que possam ser facilmente alteradas, respeitando também os princípios de dividir para conquistar, tão falados na teoria de Programação.

```
1 using UnityEngine;
2 using System.Collections.Generic;
3
4 public class ApresentacaoFazendeiros
5 {
6
7     public static List<string> arrayTexto;
8
9     public static void inicializarApresentacao(){
10         arrayTexto = new List<string> ();
11
12         arrayTexto.Add("Bem vindo ao Mercado de Trocas. Eu sou a Dona Gertrudes, a cuidadora de galinhas da região. Caso queira uma é só me procurar. Estarei te esperando.");
13         arrayTexto.Add("Olá pequeno(a) fazendeiro(a). Sou o seu Joaquim e cuido dos milhos da região. Se precisar de alguns sacos de milho, não hesite em me pedir.");
14         arrayTexto.Add("Tudo bem meu(inha) jovem? Sou o Seu Zé e tenho aqui vários porquinhos se tiver interesse. Vamos trocar?");
15         arrayTexto.Add("Olá, como vai? Me chamam de Velho Chico. Vejo que esta fazendo várias trocas. Parabéns. Agora já pode pegar algumas ovelhas, vamos lá.");
16         arrayTexto.Add("E ai amiguinho(a), sou o Miguel. Falta pouco agora para terminar. Meus cavalinhos vão te ajudar. Já estou ótimo nas trocas, e você?");
17         arrayTexto.Add("Sou a Maria, ou Dona Maria se preferir. Consiga duas de minhas vaquinhas e o lote de terra é seu. Mais algumas trocas e vai reconquistar a fazenda. Você consegue.");
18         arrayTexto.Add("Finalmente poderemos recuperar a fazenda. Achei esses lotes de terra que poderemos trocar. Você é demais.");
19
20     }
21
22 }
23
24
```

Figura 62 - Classe ApresentacaoFazendeiros.cs

4.4.1.2.2 Classes de Ferramentas

Para o uso de métodos utilizados por diversas classes que tratam de tratamento ou conversão de dados, além de evitar repetição de grandes blocos de código, foi utilizada a classe Util, que a partir de métodos estáticos conseguem prover uma boa base para todas as camadas em relação a esses casos, tendo, por exemplo, métodos de conversão de data, de verificação de determinado elemento em um array, de mudança de um determinado valor no pai e nos filhos do mesmo, entre outros métodos.

```
89              setCdJogo(System.Convert.ToInt32(arrayObjeto["cd_jogo"]));
90              if(arrayObjeto["dt_criacao"] != null)
91                  setDtCriacao(Util.convStringToDateTime((string)arrayObjeto["dt_criacao"]));
92              if(arrayObjeto["lg_ativo"] != null)
93                  setLgAtivo(System.Convert.ToInt32(arrayObjeto["lg_ativo"]));
```

Figura 63 - Exemplo de uso da classe Util na classe Jogo.cs, para realizar a conversão de data, recebendo a string de uma data e convertendo-a para o objeto DateTime

Outra classe de ferramentas usada é a de Parametros, que guarda os códigos padrão de todos os registros utilizados pelo jogo para que as outras classes possam fazer tratamentos adequados de comportamento a depender do dado, como por exemplo relacionar a Dona Gertrudes da história com o animal Galinha, e o Miguel com o animal Cavalo, também fazer a enumeração das cenas da história inicial identificando, por exemplo, o momento em que se deve trocar a cena para o jogo tutorial da caçada, e em que ponto da história deve retornar após finalizada.

```
41          if(TelaCenas.getTela().getNrCenaHistoriaInicial() == Parametros.NR_CENA_PIPO_AGINDO_ESTRANHO){
42              TelaJogoMemoria.getTela (true);
43              ViewCarregamento.direcionadorTelaCarregamento = Parametros.TELA_JOGO_MEMORIA;
44              SceneManager.LoadScene("tela_carregamento");
45          }
46          else if(TelaCenas.getTela().getNrCenaHistoriaInicial() == Parametros.NR_CENA_ENCONTRO_MORADORES_PARTE_FINAL){
47              TelaCenas.destroyTela ();
48              TelaMercadoTroca.getTela (true);
49              ViewCarregamento.direcionadorTelaCarregamento = Parametros.TELA_MERCADO_TROCA;
50              SceneManager.LoadScene("tela_carregamento");
51          }
52          else if(TelaCenas.getTela().getNrCenaHistoriaInicial() == Parametros.NR_CENA_ENCERRAMENTO_CENA_INICIAL){
53              TelaCenas.destroyTela ();
54              TelaFinalJogo.getTela ();
55              SceneManager.LoadScene("tela_final_jogo");
56          }
57          else
58              Camera.main.GetComponent<ViewCenas>().chamarCena();
```

Figura 64 - Exemplo de uso da classe Parametros na classe btnPularCena.cs, onde os parâmetros de número de cena são usados para definir para que tela o jogo será direcionado

4.4.1.3 View

4.4.1.3.1 Classes Principais

No controle de frente do jogo há dois scripts que servem como representação das cenas em questão e são eles a **View** e a **Tela**. As duas servem como modelo e cada uma das partes do jogo possuem uma filha de cada uma delas (ViewJogoMemoria e TelaJogoMemoria, ViewMercadoTroca e TelaMercadoTroca, ViewTelaInicial e TelaMercadoInicial, etc.).

A Tela é a classe que guardará as variáveis locais que serão manipuladas nas cenas, mas que não são objetos representados nas cenas, como por exemplo o ponteiro da cena da história inicial que indica qual a cena que esta sendo trabalhada naquele momento, outro exemplo que podemos utilizar é a do jogo da caçada em que o script de TelaJogoMemoria guarda se aquela chamada em questão será de um jogo de tutorial ou não. Um terceiro exemplo é o de JogoMemoriaFinal, sendo esta uma cena posterior ao jogo para poder mostrar os resultados do mesmo, em que a TelaJogoMemoriaFinal guarda a quantidade de cartas viradas, a quantidade de tempo que sobrou, a quantidade de pontos ganhos nas cartas e a quantidade de pontos ganhos pelo tempo restante.

Os scripts de View no entanto lidam diretamente com os objetos que estão nas cenas como Botões, Imagens, Caixas de Texto, Ponteiro do mouse, Camera, entre outros, eles que inicializarão as coisas na tela do computador bem como verificarão as mudanças através do método **Update**, que é realizado a cada mudança de frame. Ambas também seguem o padrão Singleton e cada uma (de cada cena) possui apenas uma instância por jogo, assim como é natural que não haja duas cenas iguais executando ao mesmo tempo.

4.4.1.3.2 Cenas

As cenas foram divididas em:

- Historia_inicial;
- Tela_atividade1;
- Tela_carregamento;
- Tela_inicial;
- Tela_jogo_memoria;
- Tela_jogo_memoria_final;

- Tela_mercado_troca;
- Tela_minha_fazenda;
- Tela_mundo;
- Tela_opcoes;
- Tela_principal.

Elas são vistas pelo jogador em um formato 2D. Elas possuem uma câmera principal (*MainCamera*) onde são adicionados os componentes de View daquela cena, além de um script especial chamado de **MouseDesign** que tem como função modificar o ponteiro do mouse. Ela também possui um script de **AudioListener** que servirá para "escutar" os sons emitidos pelas músicas de fundo, pelos botões e outros eventos no jogo. Além da câmera, as cenas possuem um objeto chamado **Canvas** que serve de repositório onde os objetos da biblioteca **UI** trabalham e vários objetos **Panel**, sendo estes responsáveis por duas funções: A de armazenar os Sprites, colocando as imagens dentro do jogo, e a de separar em áreas específicas os objetos em cena, fazendo possível a execução de uma determinada ação em vários objetos ao mesmo tempo pelas ligações hierárquicas. O Panel logo abaixo do Canvas na hierarquia é o background, que servirá de imagem de fundo em todas as cenas, todos os outros objetos em cena são filhos dele.

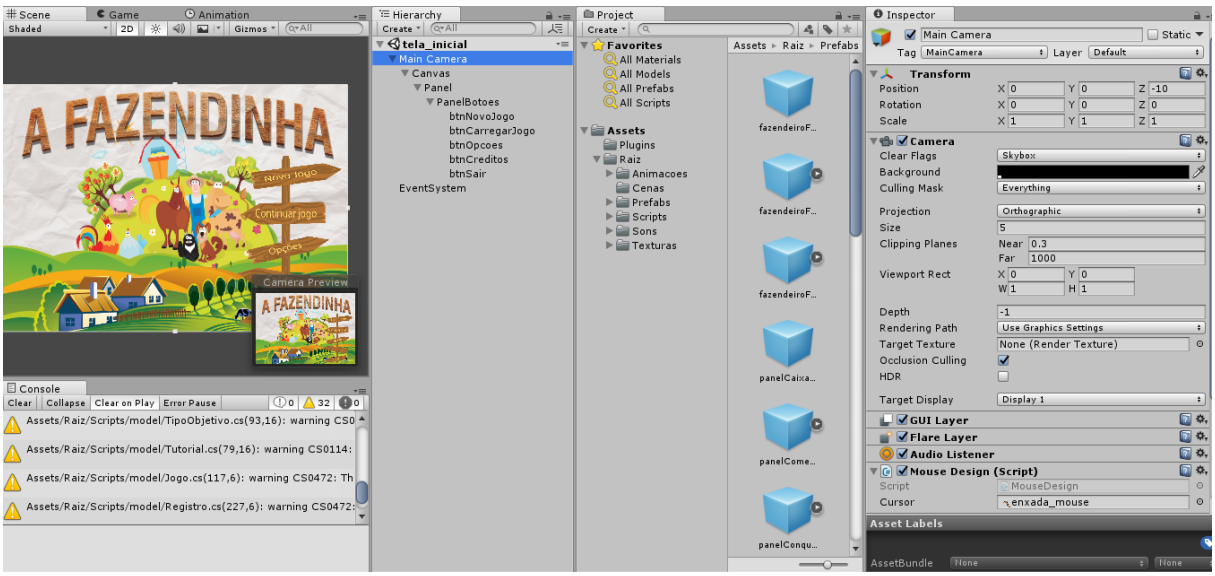


Figura 65 - Exemplo de Cena, mostrando seus elementos no Unity3D

4.4.1.3.3 Botões

Os botões são as principais áreas de interação do jogador com o jogo, é a partir deles que o jogador irá ir de uma cena pra outra, capturar os pintinhos, fazer as trocas, e todas as outras ações. Normalmente têm uma imagem principal que representará o botão, em seu componente **Imagem**, e possuem também um script chamado **MouseControl** que tem como variáveis um AudioSource para o clique do botão e um para quando o mouse passa sobre o botão, e para utilizar esses sons é ligado a cada botão um componente chamado **EventTrigger**, que possui a função de fazer eventos relacionados ao uso do mouse sobre o componente. Nos botões do jogo são utilizados os eventos **Point Enter**, **Point Exit** e **Point Click**. Outra característica colocada nos botões foi a animação ao se passar o mouse por cima do mesmo a partir da variável de **Transition** do componente Button e do componente **Animator**, que realiza uma "gravação" de determinadas mudanças que podem ser feitas no objeto da cena e serão realizadas no momento da ação programada. No caso dos botões deste jogo foi utilizada a alteração de escala, fazendo os botões ficarem maiores quando o mouse passa por cima, e retornando ao tamanho original quando o mouse deixa o objeto.

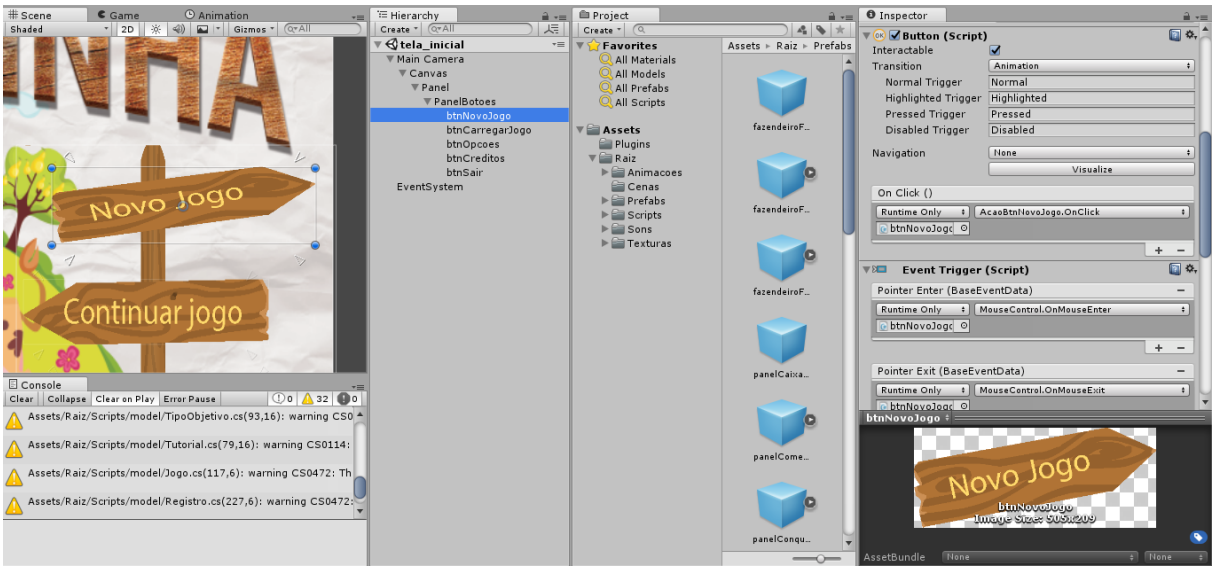


Figura 66 - Exemplo de botão, frisando os componentes de Button e EventTrigger no Inspector

A Figura 67 mostra a organização básica dos objetos dentro da Cena, destacando a importância dos objetos **View** e **Tela** dentro do MainCamera, e do **MouseControl** e dos **EventTriggers** dentro dos botões, que dão as interações principais do jogador com o jogo.

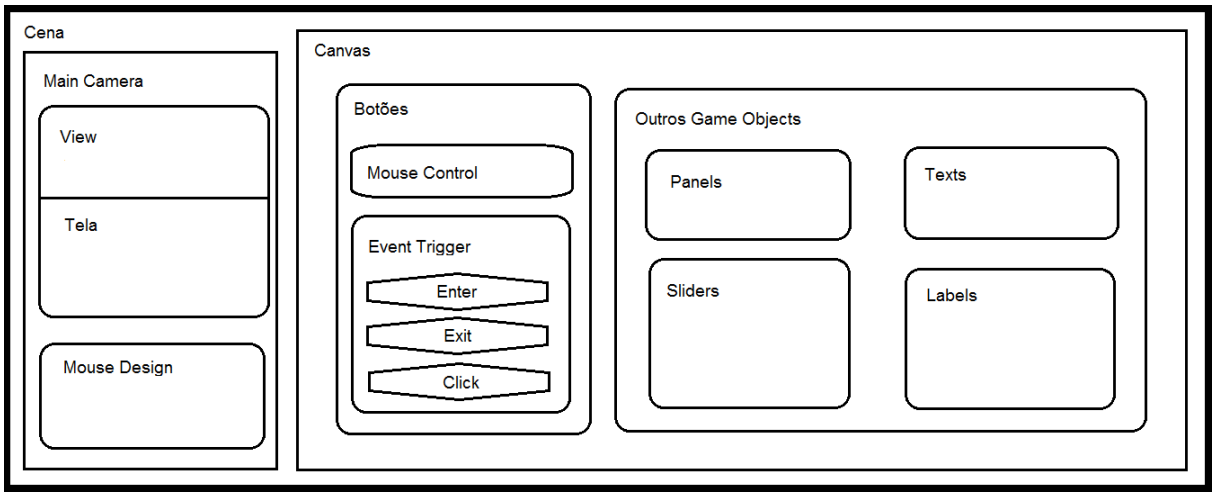


Figura 67 - Organização padrão dos objetos de cena do jogo

4.4.1.3.4 StartCoroutines

O uso das StartCoroutines, como explicado na sessão do Referencial Teórico, é usado para realizar um *delay* na execução do código de forma programada. No jogo em questão, seu principal uso foi o de criar uma espera para que o som do "clicar" do botão fosse ouvido antes que o componente fosse inativado ou que a cena fosse retirada, nos botões de "Sair" ou "Voltar", por exemplo. Normalmente foi utilizado o tempo de 0,7 segundos para que o som fosse ouvido antes que o componente sumisse. Além de botões, ele também foi utilizado na chamada final de uma atividade, quando o jogador faz o último acerto ou erra alguma das etapas da atividade, um quadro é mostrado indicando quantas vezes o mesmo já ele fez aquela atividade e quantas ele acertou, é dado então um tempo de 3 segundos para que o jogador possa visualizar o quadro para então o mesmo desaparecer e voltar ao mercado de trocas. Uma terceira função do StartCourotines no jogo foi no **Slider** (sendo este o componente responsável por montar a barra de progresso de experiência, usada para alcançar novos níveis), onde para que a barra cresça progressivamente é necessário um certo *delay* para que se possa aumentar o valor de experiência aos poucos, ou então o jogador iria ver apenas a quantidade final de experiência, sem aquela progressão que é uma característica bastante utilizada nos games eletrônicos.

```
7     //Função de clique no botão
8     public void OnClick(){
9         StartCoroutine(delayOnClick(0.7F));
10    }
11
12    //Função que irá ser chamada com um delay
13    private IEnumerator delayOnClick(float waitTime){
14
15        //O código irá ficar em delay exatamente nesse ponto pela quantidade de segundos passada
16        yield return new WaitForSeconds (waitTime);
17
18        //Faz a inicialização das classes de dados
19        ApresentacaoFazendeiros.inicializarApresentacao ();
20        FalasCenas.inicializarFalas ();
21        MensagensTutorial.inicializarMensagens ();
22
23        //Atualiza a data de último acesso no jogo
24        Registro registro = Registro.getRegistro ();
25        registro.setDtUltimoAcesso (System.DateTime.Now);
26        Result resultado = registro.save ();
27        if (resultado.getCode () < 0) {
28            Debug.LogError (resultado.getMessage ());
29        }
30
31        //Destroi o objeto de tela inicial, constroi o da tela principal e chama a tela principal
32        TelaInicial.destroyTela ();
33        TelaPrincipal.getTela ();
34        ViewCarregamento.direcionadorTelaCarregamento = Parametros.TELA_PRINCIPAL;
35        SceneManager.LoadScene("tela_carregamento");
```

Figura 68 - Exemplo de uso do StartCoroutine para clicar no botão de “Continuar Jogo” na tela inicial

4.4.1.3.5 Prefabs

Os prefabs são largamente usados durante todo o jogo. Um de seus usos mais frequentes é para criar caixas de diálogo com o jogador, criar os animais nas telas de Mercado de Trocas e de Atividade, criar os painéis dos Fazendeiros ao se apresentarem no Mercado de Trocas e criar as cartas na Caçada aos Pintinhos. Para fazer a criação de um prefab é utilizado o método **Instantiate**, que tem por função Inicializar objetos na cena, porém ele necessita ser vinculado a algum objeto pai para que o mesmo seja visível (normalmente é utilizado o componente **RectTransform** que se encontra no canvas principal e o objeto recebe este componente como pai), ou necessita que seja colocado as coordenadas exatas em que ele estará na tela, como é feito com as cartas da Caçada. Os prefabs também foram úteis na criação de um objeto que é utilizado várias vezes em várias cenas diferentes, como é o caso dos prefabs responsáveis pelos sons de clique do botão e o passar do mouse sobre o mesmo.

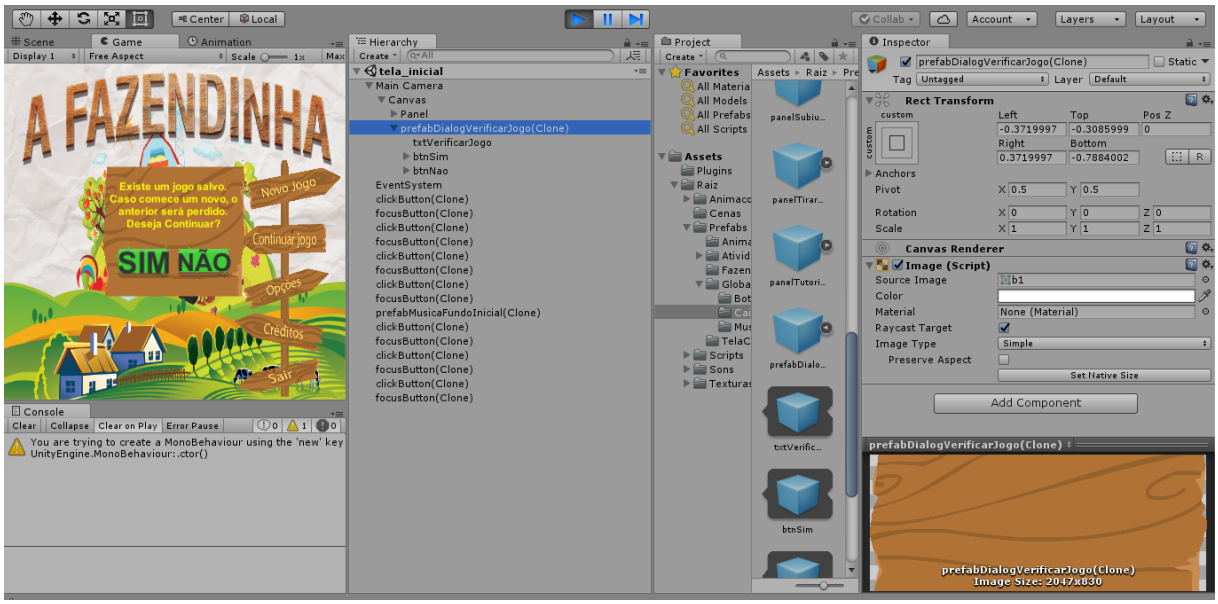


Figura 69 - Exemplo de uso do Prefab, fazendo uma chamada no botão “Novo Jogo” que verifica se já existe um jogo em execução, caso exista lança essa caixa de diálogo para perguntar ao jogador se o mesmo deseja continuar

4.4.2 Tela Inicial

Figura 70 - Tela Inicial

A tela inicial (Figura 70) é a tela mostrada no jogo assim que o jogador entra. Nela é mostrado o fundo do jogo retirado dos cadernos do jogo analógico, trazendo assim familiaridade com a Fazendinha Matemática Original. Dentro do Panel de *background* há um panel que englobará os botões utilizados. O Panel e os botões foram criados para darem ideia de placas pregadas em direções diferentes, remetendo a forma como é feita a sinalização em áreas rurais. Nas placas (que são botões) foi utilizada a mudança de cor do texto do botão, além da animação padrão de alteração de escala, quando o mouse passar por cima do mesmo (*highlighted).*

O primeiro botão a ser mostrado é o de "Novo Jogo", nele o jogador iniciará a sua experiência de jogo, direcionando-o à tela de História Inicial. Um tratamento utilizado nesse botão ocorre quando o jogador já tenha começado um jogo previamente e deseje começar outro, o jogo então chamará o *prefab* **prefabDialogVerificarJogo**, sendo esta uma janela de diálogo informando ao jogador que já existe um jogo salvo e que caso o mesmo continue, o jogo será sobrescrito.

O segundo botão é o de "Continuar Jogo" e levará o jogador para a tela principal do jogo, caso já exista um jogo salvo, e o fará ir para a História inicial caso não haja nenhum registro.

O terceiro botão é o de "Opções", que direcionará o jogador à tela de onde o mesmo poderá fazer as configurações de: **Volume**, **Tempo Inicial do Jogo da Memória** e **Poder**

avançar sem realizar todas as trocas possíveis. Nas configurações de Volume e Tempo foi utilizado o objeto **Slider**, que representa uma barra de progresso que pode ser avançada ou retrocedida pelo usuário. No volume é utilizada uma faixa de valores entre zero (0) e um (1), com o uso de números decimais, para que o usuário possa configurar com maior precisão qual o volume que melhor lhe agrade para jogar, esse volume será utilizado com a chamada **Jogo.getConfiguracao().getVlVolume()**, acessando o campo de volume através do método assessor do mesmo. Esse valor será multiplicado pela propriedade de **volume** dos componentes de AudioSource dos objetos, definindo assim o volume final proporcional ao volume escolhido nas configurações. Já o tempo inicial do jogo da memória possui uma faixa de dez (10) até cento e vinte (120), apenas de valores inteiros, e mostrando através de um objeto **Text** o valor atual (a mudança de valor feita pelo usuário ativará o método **OnValueChanged** do s*lider*, e este possuirá uma referência ao *text* para mudar seu valor conforme o novo valor do Tempo será alterado. O valor do tempo será utilizado na tela de Jogo da Memória a partir da chamada **Jogo.getConfiguracao().getQtTempoMemoria()**. O outro objeto de configuração para poder avançar sem realizar todas as trocas é um **Toggle**, sendo esta uma caixa de seleção “Sim” ou “Não”. Este usará o método **OnValueChanged** para fazer a alteração no banco de dados. No uso do código é utilizada a chamada **Jogo.getConfiguracao().getLgAvancarSemEsgotarTempo().**

O quarto botão é o de “Créditos”, em que serão mostradas todas as pessoas envolvidas no projeto, bem como autores de imagens, músicas e sons utilizados. E o quinto e último é o de “Sair” que saíra do jogo completamente.

4.4.3 História do jogo

Figura 71 - Tela da História do Jogo

Nessa parte do jogo será contada toda a história, introduzindo tanto os personagens quanto o próprio jogador, indicando qual será o objetivo do mesmo bem como a forma com que ele irá atingir o objetivo. A cena conta com um Panel principal, com o background da cena, e um Panel para a caixa de texto, ambos mudarão conforme a história passa. Essas mudanças serão realizadas através de um *array* de *sprites* no **ViewCenas**, vinculado a câmera principal, chamado **Background** e através da classe de dados **FalasCenas.cs**, que conta com um *array* bidimensional de *strings*, sendo que o nível mais externo (à direita) fará as passagens de fala na caixa de texto, e o nível mais interno (à esquerda) fará as passagens dos backgrounds das cenas. Para realizar essa dinâmica é utilizado no ViewCenas dois métodos: **chamarCena** e **delayControlePassagemTexto**, o primeiro será a chamada de cada background em que constará uma ou mais falas armazenadas no *array* de FalaCenas, o segundo será um *loop* que se repetirá para colocar cada uma das falas na caixa de texto.

Para ativar cada uma das passagens de texto é utilizado o botão de Pular, a direita da caixa de texto. O mesmo possui o script **btnPularCena** que tem como função passar para o próximo texto, mas também pode adiantar o texto ao ser clicado antes que o mesmo termine de ser escrito na caixa. Internamente a função de pular apenas controla uma variável booleana que é utilizada como condição em um ***while*** infinito, fazendo com que a mesma se torne verdadeira (*true*), para passar no *while*. A função do btnPularCena só apresentará comportamento diferente em algumas situações, como quando o jogador precisar digitar seu nome na tela, fazendo então a reinicialização efetiva de todos os dados do banco de dados que tenham relação com Registro, e também quando o jogo precisará passar pela caçada de pintinhos tutorial, no meio das passagens de cena.

Outro recurso interessante dessa parte do jogo foi a passagem de texto, em que é usado o script **AutoTextCenas**, que se encontra no próprio objeto Text da cena. Este script tem como variável o texto a ser colocado na caixa e utiliza uma StarCoroutine para dar um *delay* a cada letra colocada na caixa, dando um aspecto de leitura interessante e progressiva. Também é utilizado no *text* um AudioSource que simula uma máquina de escrever, dando uma maior imersividade à história no momento da execução.

4.4.4 Tela Mundo

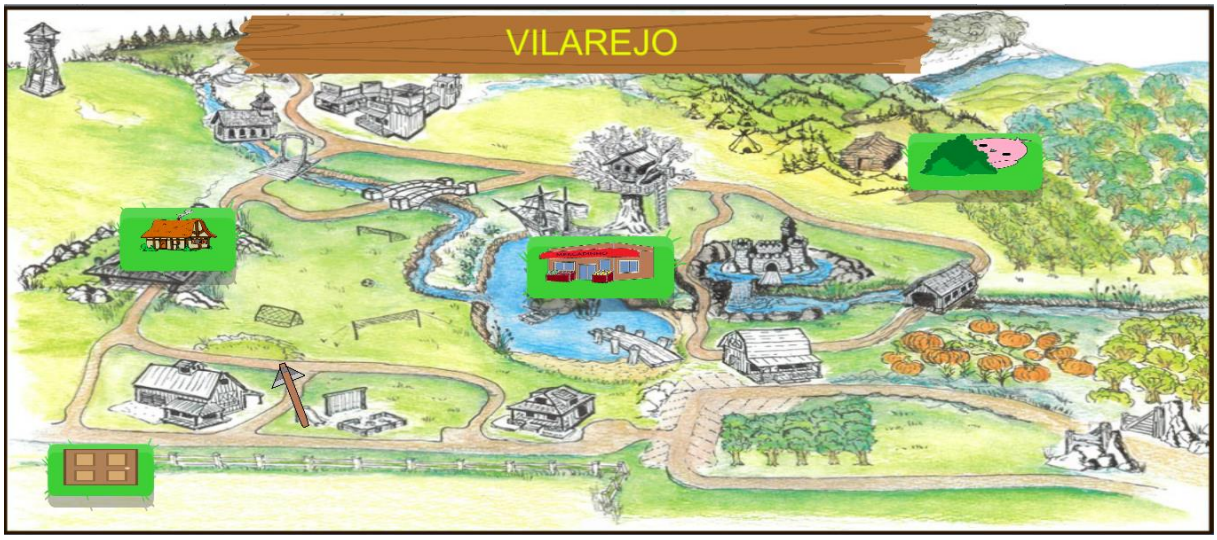


Figura 72 - Tela do Mapa do Vilarejo

A tela mundo (Figura 72) representará o mapa do Vilarejo, e será usado para simular uma “mobilidade” no jogo, em que o jogador deverá acessar sempre essa tela para mudar de um lugar para o outro, criando maior imersividade no sentido de ele realmente estar no Vilarejo caçando os pintinhos na floresta e indo trocar no mercado, e colocará como botões os lugares em que o jogador poderá ter alguma ação. Nessa versão do jogo estão habilitados apenas os locais do Mercado de Trocas, da Caçada de Pintinhos, e da Minha Fazenda.

4.4.5 Jogo da memória

Figura 73 - Tela do Jogo da Memória/Caçada de Pintinhos

O jogo da memória (Figura 73) representa a caçada dos pintinhos na história e é o momento em que o jogador pode obtê-los para então poder realizar as trocas. Essa parte é dividida em duas: o jogo da memória em si e o resultado final, mostrando quanto foi obtido, em que tempo e quanto de experiência foi ganha. O jogo da memória possui um *background*, uma área em que mostra a quantidade de pintinhos (ou cartas) já capturados, e uma área mostrando uma contagem regressiva que é iniciada com o tempo-base configurado previamente na tela de opções (por padrão de 20 segundos) e acrescentado um tempo de acordo ao nível do jogador e número de cartas postas na tela.

Na área principal é posta as cartas dos pintinhos, porém ao contrário dos outros objetos de cena, estes são criados em tempo de execução. A tabela de **Configuracao_Cartas** guardará as posições de referência na tela para que se disponha a quantidade de cartas para o nível em que o jogador se encontra, entre as informações guardadas estão: As posições de X e de Y do canto superior esquerdo da primeira carta, os valores de largura (*width*) e altura (*height*) de cada carta, a quantidade de cartas, as quantidades de cartas por linha e coluna, e o espaçamento entre as cartas na vertical e na horizontal. Para criar as cartas, primeiro é verificado o nível do jogador através do método **verificarNivel**, e fazer a busca no banco de dados de qual registro de configuração o jogo irá utilizar naquele momento.

Logo após as cartas são distribuídas com o método **distribuirCartas**, essa distribuição consiste em fazer um array com a quantidade de cartas que serão utilizadas naquele momento

(12, 18 ou 24 cartas) e sortear as posições das cartas para que a cada partida haja uma distribuição diferente dos pintinhos na tela.

O próximo momento é a inclusão das cartas especiais com o método **incluirCartasEspeciais**, para tal é utilizado um valor de **chance** (que também dependerá do nível do jogador, sendo menor a medida que o jogador aumenta seu nível) em que aleatoriamente o algoritmo irá sortear um número em uma faixa de 1 até o valor da chance, caso o número sorteado seja metade do valor da chance, então é incluída uma carta especial, que substituirá uma das cartas comuns naquela rodada. O método de inclusão das cartas especiais é usado duas vezes pois são duas as classes de "pintinhos especiais", sendo que há cinco pintinhos de segunda classe (a primeira classe são os pintinhos comuns que sempre aparecerão) e há um único pintinho de terceira classe, o mais difícil de aparecer.

Então finalmente são inseridas as cartas na tela com o método **incluirCartasView**. Nesse método é feito um laço que irá construir carta a carta que deverá ser incluída na cena. A partir do método **AddComponent** são incluídos os componentes que as cartas precisarão, que são: **SpriteRenderer**, que guardará o Sprite da carta virada de face para baixo, ou seja, será o mesmo Sprite para todas as cartas nesse momento de construção; **BoxCollider2D**, sendo responsável por ser possível identificar um clique do mouse quando o jogador selecionar a carta, tratando tal evento; e **Carta**, que guardará as informações da posição da carta que foi distribuída aleatoriamente entre as 12, 18 ou 20 cartas dispostas, e também guardará a informação da posição do par da carta.

Após isso o objeto então usada a chamada **.tranform.localScale** para indicar a largura e altura do objeto em cena, utilizando os valores buscados em **Configuracao_Cartas**, para então usar o **Instantiate** e instanciar o objeto em cena, porém ao invés de passar o pai do objeto, como normalmente foi feito em outras chamadas, o algoritmo utiliza dois outros parâmetros do método Instantiate, que são um **Vector2D**, que indica as posições em X e em Y em que o objeto será instanciado, e um **Quartenion**, que é uma classe que indica a rotação de um objeto em tela, sendo usado neste caso o valor de **Quartenion.identity**, que corresponde a uma "não-rotação". Após isso o objeto original é destruído e os valores reiniciados para a próxima carta.

O jogo é iniciado com um painel (originado de um prefab) que diz quando o jogo deseja começar o jogo, ao iniciar a contagem de tempo vai sendo decrescida com a função **Time.deltaTime**, que é a contagem de tempo em que um frame inicia e termina. O algoritmo então vai somando os valores a cada frame que passa (dentro do método **Update** do ViewJogoMemoria), quando atinge 1, então o tempo é decrescido em 1 segundo. O jogador

então contará com o mouse e deverá clicar nas cartas para que as mesmas sejam “viradas”, pois o que acontece internamente é o evento de clique do mouse ser disparado através do componente SelecionarCarta (acrescentado anteriormente em todas as cartas da cena), o algoritmo então identifica que a carta foi clicada e busca o componente Carta, que guarda o índice de posição que foi sorteado no inicio da chamada do Jogo da Memória. Então o array inicial é consultado com a posição buscada, e o s*prite* (ou imagem) de “carta virada” é substituída pela da posição do array escolhido.

O próximo passo é esperar que o jogador clique em mais uma carta e comparar o campo **par** de ambos, para saber se eles fazem um par de cartas “iguais”. Caso sejam, ambas são destruídas, sumindo da cena, e a área de pintinhos capturados registra +2 em sua conta. Caso não sejam, o algoritmo, após um pequeno delay, “desvira” as cartas substituindo as imagens dos pintinhos pela da “carta virada” novamente. E assim acontece até o final do tempo ou até que o jogador desvire todas as cartas antes do tempo acabar, o jogo então passará para a tela de Jogo Memória Final.

Na tela de Jogo Memória Final (Figura 74) é apresentado um painel que indica quantos pintinhos foram capturados naquela caçada, quanto tempo ficou faltando para terminar a contagem, e a experiência ganha com esses resultados. O objeto usado para mostrar a experiência é o Slider, que possuirá uma faixa de zero (0) até dez (10), e poderá ser qualquer valor decimal. O script ViewJogoMemoriaFinal, que é o View principal dessa parte, terá em seu algoritmo a inicialização dos valores de quantos pintinhos foram capturados e quanto tempo sobrou. Esses valores serão buscados a partir da classe TelaJogoMemoriaFinal, que terá tais campos e serão inicializados ainda na tela de Jogo da Memória, passados antes do jogo mudar a tela. O Valor da experiência então será acrescido automaticamente em um loop controlado pelo método **FillSlider**, que usará do StartCoroutine para ir aumentando o valor do Slider aos poucos, e a cada vez que chegar no valor máximo (10), modificar á o valor de nível do jogador.

Essa tela contará com três botões na parte de baixo: O primeiro é o de saída, em que o jogador voltará para a Tela Principal; O segundo botão é o retornar ao jogo da memória, porém esse botão aparecerá somente quando a configuração permitir que sejam feitas caçadas sem esgotar todas as trocas possíveis; O terceiro botão é o que levará o jogador de volta ao Vilarejo. Um evento que poderá acontecer é quando o jogador tiver subido de nível, o jogo transportará o jogador para a tela de atividades, desafiando ele com perguntas sobre as trocas.

Figura 74 - Jogo Memória Final, onde serão contabilizados os ganhos do jogador na rodada

4.4.6 Mercado de Trocas

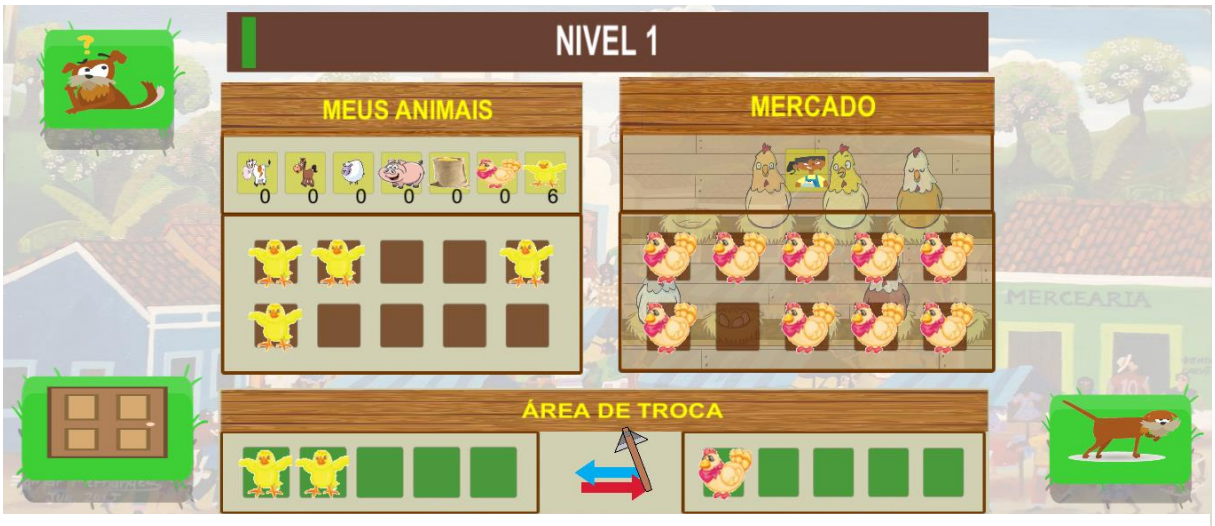


Figura 75 - Tela do Mercado de Troca

O mercado de trocas (Figura 75) é o espaço principal do jogo, onde o jogador irá desenvolver seu conhecimento sobre as trocas e atingir os objetivos, até conquistar o item final: Um lote de terra. Essa tela é separada em três botões, cada um em um dos cantos. No canto superior esquerdo está o botão de tirar dúvidas, que chama um *prefab* e instancia uma

tela pra o jogador. Nessa tela se encontra a tabela completa para que o jogador possa consultar caso tenha dúvidas. Os animais aparecerão inicialmente todos em preto-e-branco, e irão ficando coloridos conforme o jogador for conquistando novos animais, para isso será consultada a tabela de **Registro_Item**, que conterá um campo lógico, definindo se aquele animal já foi capturado alguma vez. Essa janela também mostrará os objetivos que o jogador já conquistou, sendo estes: Troca Direta (quando o jogador faz uma troca qualquer), Troca Inversa (quando o jogador realiza uma troca um por dois, ou seja, um animal de maior valor por dois de menor valor do fazendeiro), Troca 4 por 1 (quando o jogador faz uma troca de quatro animais por um, pulando um nível da tabela) e Troca 4 por 2 (quando o jogador faz uma troca de quatro animais por dois do nível seguinte). O segundo botão que se encontra no canto inferior esquerdo, é o botão de voltar, retornando para a tela principal. O terceiro botão, no canto inferior direito, é o que irá novamente para o Vilerejo.

A parte central desta tela é dividida em três áreas: Na primeira área há na parte de cima os animais que o jogador pode ter, acompanhado de uma numeração (a quantidade daquele animal que o jogador possui). Os objetos que representam os animais serão clicáveis (e também terão a animação de aumento de escala), ao clicar no animal o componente **Selecionar** será ativado. Esse componente utilizará o recurso de **tags** para saber qual animal deve ser colocado na segunda área, fazendo um encadeamento de if-else para saber qual animal será colocado na parte de baixo. Haverá apenas dez quadros que poderão ter animais, e caso o número de animais que o jogador possua seja maior, apenas dez serão instanciados. Para organizar os quadros de todas as partes é utilizado o componente **GridLayoutGroup**, que posiciona automaticamente os elementos dentro do objeto (no caso, dentro das partes das áreas).

Na segunda área aparecerão os fazendeiros na parte de cima, porém cada um só aparecerá quando o jogador conquistar o animal de nível inferior ao que o fazendeiro oferece. Por exemplo: Miguel que possui cavalos só aparecerá quando o jogador tiver conquistado alguma ovelha. Nessa área os fazendeiros também serão clicáveis, e irá popular a parte de baixo da segunda área com os animais daquele fazendeiro. Ao contrário da primeira área em que o algoritmo contará quantos animais o jogador possui daquele animal, para encher a parte de baixo, na segunda área todos os 10 quadros disponíveis serão preenchidos com o animal do fazendeiro. Isso ocorrerá para que o jogador não precise se preocupar com quantos animais o fazendeiro tenha. Após o jogador clicar no animal que deseja trocar e no fazendeiro com quem ele quer trocar, ele precisará selecionar os animais que deseja trocar, jogando-os na área de troca, a terceira área.

Essa terceira área contará apenas com 5 quadros do lado do jogador e 5 do lado do fazendeiro. O jogador quando passar o mouse por cima dos animais verá aparecendo uma Seta Azul. Essa seta será ativada pelo componente **EntrarAnimal** que utilizará a variável **seletor** do componente **ClicarAnimal** (ambos os componentes estarão vinculados aos animais preenchidos nas áreas) para saber se a seta é para baixo (caso o animal esteja na primeira ou segunda áreas) ou para cima (caso o animal esteja na terceira área, de trocas). Para colocar o animal na área de trocas, bastará clicar nele, e clicar de volta para fazê-lo voltar à primeira ou segunda área.

Quando o jogador tiver escolhido os animais e colocados nas áreas de troca dos dois lados, ele terá um botão para realizar a troca, que ativará o script **Troca**. Esse *script* irá fazer a verificação do nível do animal com a quantidade colocada, em uma equação definida. Caso não haja igualdade, o algoritmo irá chamar um *prefab* que alertará o jogador que ele não acertou a troca, para tentar novamente. Caso haja igualdade o algoritmo verificará se algum dos objetivos foi atingido (e se foi atingido pela primeira vez), verificará se algum novo fazendeiro será acrescentado (quando um novo animal for obtido) e verificará se o jogador atingiu um novo nível, com a experiência ganha. Então irá ser atualizado a quantidade dos animais trocados no banco de dados, atualizado a quantidade de animais mostrada nos objetos **Text** na primeira área, limpado os animais da área de trocas, atualizado a quantidade de animais na primeira área e preenchido novamente todos os espaços na segunda área, com os animais do fazendeiro. Caso se tenha conseguido um novo nível, um novo objetivo ou um fazendeiro novo se apresentar, um *prefab* especial será chamado para mostrar as informações novas para o jogador.

4.4.7 Tela de Desafios

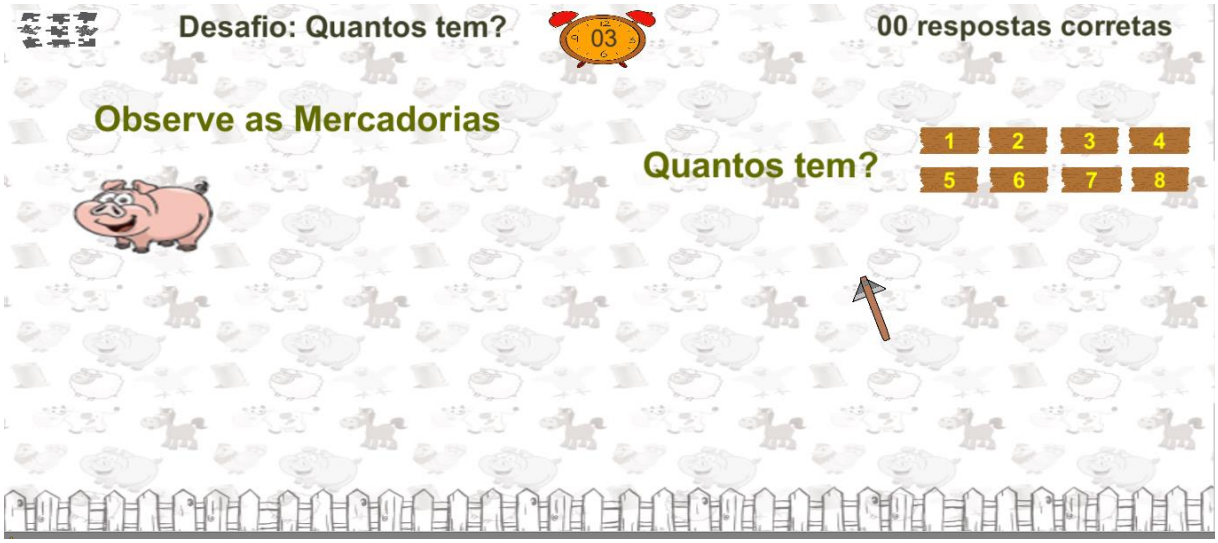


Figura 76 - Tela de Desafio

A tela de desafios/atividades (Figura 76) será chamada sempre que o jogador subir de nível, ela fará perguntas ao jogador desafiando sobre o que o mesmo já aprendeu da tabela de trocas com quatro perguntas. A tela possui na parte de cima algumas informações, sendo a primeira delas o nome do desafio **Quantos tem?** (em versões posteriores haverá mais desafios). No centro dessa parte também é mostrado um relógio, mas a contagem para este é progressiva, ao invés da regressiva utilizada no Jogo da Memória. Servirá para que o professor tenha posteriormente noção de quanto tempo o jogador leva para realizar a atividade. Na parte de cima no lado direito há o número de desafios completos que o jogador já cumpriu. Em baixo será mostrado, inicialmente, as mercadorias e a pergunta de quantos tem. A quantidade de mercadorias (quantMercadoria) será escolhida aleatoriamente no **Start** do ViewAtividade1, bem como a mercadoria a ser sorteada (numMercadoria). Esse valor da quantidade é guardado para ser comparado ao botão da pergunta de "Quantos tem?", em que cada botão terá o mesmo script **btnQuantidadeMercadoria** como um de seus componentes, porém terá o valor de **quantMercadoria** inicializado de acordo ao seu número (de 1 até 8). Quando o jogador clicar em um dos botões, os valores serão comparados para saber se o jogador acertou a quantidade sorteada. A cada resposta do jogador, o algoritmo chamará um prefab com um feedback, dizendo se o jogador acertou ou não a resposta. Caso não tenha

acertado, o jogo irá imediatamente para a tela de Mercado de Trocas. Caso tenha acertado, mostrará a próxima pergunta.

A segunda pergunta é “Pelo que pode trocar diretamente?” e nela estará um botão para cada mercadoria. O algoritmo analisará o botão que foi clicado a partir do valor de **cdAnimalEscolhido**, inicializado no componente **btnPodeTrocar**, que estará em cada um dos botões. O algoritmo comparara o valor do cdAnimalEscolhido do botão clicado com o valor de numMercadoria, sorteada no inicio do ViewAtividade1. Caso o valor de cdAnimalEscolhido seja igual a numMercadoria mais ou menos 1, então a resposta estará correta, e passará para a próxima pergunta, caso contrário a resposta não estará correta, e voltará para a Tela de Mercado de Trocas.

A terceira pergunta não é exatamente uma pergunta, mas pede para o usuário selecionar quantas mercadorias do tipo escolhido na pergunta 2, ele irá precisar para trocar pelas mercadorias que estão sendo analisadas (sorteadas inicialmente). Para fazer a seleção e desseleção, são usadas duas imagens, uma com a mercadoria com fundo transparente, e outra com a mercadoria com o fundo verde. Ao clicar o jogador ativará o script **Selecione**, que será componente de todos os objetos ali. O algoritmo verificará se o botão já está selecionado ou não através da variável **selecionado**, e fará a troca do *sprite* (ou imagem) do botão e também da variável. Após isso o jogador deverá clicar no botão de trocar, que irá contabilizar quantas mercadorias o jogador pôs na troca e se essa quantidade esta correta ou não, para a troca pelas mercadorias iniciais. Caso esteja, o jogador ainda deverá responder a última pergunta, que é “Algum sobra?”, essa pergunta é necessária pois caso seja sorteado inicialmente uma quantidade par de mercadorias, e o jogador escolher na segunda pergunta uma mercadoria de nível superior, a troca não será exata. Logo o jogo disponibilizará na quarta pergunta dois botões: Sim ou Não, para que o jogador analise mais essa possiblidade, a partir do componente **btnSobra**. Após essa pergunta, acertando ou errado, o jogador terá um *feedback* geral sobre como seu progresso com esse desafio e será direcionado à tela de Mercado de Trocas.

4.4.8 Minha Fazenda

Figura 77 - Tela Minha Fazenda

A tela de Minha Fazenda (Figura 77) representa o local onde o jogador e Seu Toninho guardam os animais enquanto não conquistam o lote de terra. Nela o jogador também terá um feedback em relação a outras informações de seu progresso no jogo. A tela terá como painel de fundo um local desolado, na parte de cima, à esquerda, o jogador terá o botão de estatísticas, que informará, por meio de um prefab instanciado, a quantidade de pintinhos já capturados pelo mesmo, a quantidade de desafios ocultos já realizados, a quantidade de tempo gasta, e o número de atividades feitas por ele corretamente. Na parte inferior esquerda estará um botão de voltar, que utilizará não uma chamada de cena (como em todas as outras telas), mas um desempilhamento daquela cena, mantendo a cena original que chamou a "Minha Fazenda", a partir da chamada **SceneManager.UnloadSceneAsync**. Já na parte inferior à direita, estará o botão da coleção de pintinhos, em que serão mostrados todos os tipos de pintinhos que o jogador já capturou e quais faltam ele capturar. Espalhados no resto da tela estarão os animais que o jogador possui no momento, com a quantidade que o jogador possui.

4.4.9 Final de Jogo

Figura 78 - Tela Final de Jogo

A tela de final de jogo (Figura 78) aparecerá logo após o jogador conquistar o Lote de Terra, último item da tabela de trocas. Ela terá como função dar as estatísticas finais do jogador ao término do jogo. Ela é dividida em três partes. Na primeira dará a quantidade de tempo total que o jogador teve no jogo, esse tempo será contabilizado apenas das partas essenciais do jogo, tais como: A caçada aos pintinhos, o mercado de trocas e a tela de desafios. Isso acontecerá, pois o professor necessitará apenas da quantidade de tempo que o jogador estará em atividade nessas telas pensando sobre o qual será o seu passo seguinte. Também será disponibilizado nessa primeira tela a quantidade total de pintinhos que o jogador capturou e o total de desafios ocultos realizados por ele. Na segunda tela (Figura 73) o jogo dará as estatísticas dos desafios realizados, mostrando quantos foram os acertados, qual o tempo médio geral que o jogador levou para responder, e qual o tempo médio que ele demostrou em cada uma das perguntas. Na terceira parte serão mostradas quantas trocas de cada animal o jogador realizou corretamente. Essa tela foi pensada para que o professor pudesse trabalhar as dificuldades gerais de cada aluno individualmente, vendo tempos e acertos para sanar as dúvidas pontuais que os mesmos venham a ter. Internamente todas as funções foram colocadas na classe Util.cs para que pudessem ser acessadas posteriormente de outros pontos do sistema.

4.4.10 Elementos interativos e educacionais utilizados no jogo

Durante o jogo foram utilizados diversos elementos necessários para manter o jogo com boa interatividade e com os elementos educativos que o tornam um jogo educacional. Os primeiros elementos que podemos analisar são os botões, que foram criados todos de uma mesma forma, com um fundo verde e com uma placa de madeira aparecendo uma metainformação sobre o que aquele botão representa (Figura 79). Eles foram criados de tal forma que os jogadores possam associar rapidamente que aquele é um ponto de interação com o jogo quando virem elementos nessa forma, criando o signo do que significa “botão” no jogo. As caixas de diálogo em forma de uma tábua grande com elementos em amarelo dentro dela também criam essa familiaridade com as mesmas. Percebe-se ai o uso de signos estáticos, que indicam ao usuário as interações que estão disponíveis a eles ao jogarem.

Figura 79 - Botões padrão do jogo

As caixas de tutorial (Figura 80) são outro elemento importantíssimo durante a interação inicial com o jogo, pois indicam o que os jogadores podem fazer em cada tela. Essas caixas são signos metalinguísticos informando ao jogador o sentido de outros signos. Do ponto de vista educacional elas também são importantes, servindo de mediadoras do jogador com o objeto de aprendizagem ajudando-o a manipula-lo e assim obtendo os conhecimentos que o objeto que trazer a tona.

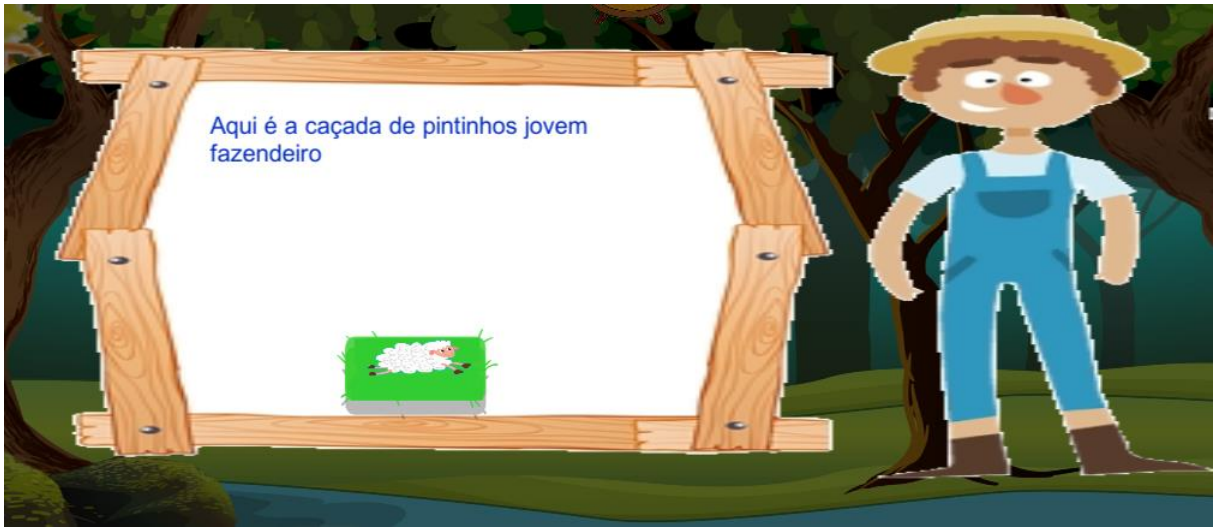


Figura 80 - Caixas de Tutorial do Jogo

Outro recurso importante acrescentado é a barra de experiência e a contagem de nível (Figura 81). Esse elemento remete à teoria de sequências didáticas que indicam que o jogador deve receber desafios mais fáceis que vão sendo dificultados conforme o mesmo for aprendendo e se desenvolvendo na sequência, e é exatamente isso que acontece quando ele sobe de nível e o número de cartas para o jogo da memória aumenta, e quanto mais pintinhos tiver mais trocas ele terá de realizar, passando por todos os animais, ajudando-o a fixar melhor a mecânica das trocas e ajudando-o a entender metacognitivamente os conceitos de mudança de "casa" nas operações matemáticas.

Figura 81 - Barra de Experiência e Nível

Os dois últimos recursos que comentaremos aqui são o jogo da memória, que com o adicional do tempo regressivo torna o jogo tão interessante e desafiador quanto um jogo da memória comum com vários jogadores, onde o jogador deverá criar estratégias de modo rápido para poder acertar a maior quantidade de pares dentro do tempo. E por fim o recurso de "Tipos de Pintinhos" (Figura 82), onde o jogador poderá colecionar os pintinhos capturados e completar uma espécie de álbum de figurinhas, servindo unicamente para manter um objetivo

secundário no jogo. Isso foi incluído seguindo as orientações de *Game Design* que indicam que jogos com apenas um objetivo se tornam facilmente entediantes.

Figura 82 - Coleção de pintinhos capturados

5 Avaliação e Resultados

O resultado do jogo (na versão apresentada neste trabalho) foi avaliado em dois momentos diferentes. Essa forma de avaliação foi escolhida por conta da natureza inclusiva de desenvolvimento deste jogo, onde profissionais de educação, tecnologia e *game design* atuam juntos na construção de uma ferramenta que leve aprendizagem e divertimento. A primeira avaliação, portanto, ocorreu no dia 25 de novembro de 2016 com três professoras de longo tempo de ensino na rede municipal de Vitória da Conquista, quando o jogo estava na metade de seu desenvolvimento e vários de seus recursos não estavam implementados e foram alterados devido a essa avaliação, enquanto a segunda avaliação ocorreu no dia 27 de junho de 2017, com duas das mesmas professoras que participaram da primeira e mais algumas professoras e futuras professoras, o jogo já estando na mesma versão em que esta apresentada no trabalho.

A forma de avaliação foi um questionário com questões abertas dividido em duas partes, uma analisando as expectativas das professoras e alunos em relação ao uso de jogos educacionais em sala de aula e recursos tecnológicos e outro questionário aplicado após jogarem o jogo em grupo (mesmo sendo um jogo individual), perguntando algumas questões relacionadas à jogabilidade e mecânica, além da diferença entre o jogo analógico e do jogo virtualizado (para aquelas que conheciam as duas versões) e questões sobre como as mesmas aplicariam esse jogo em sala de aula.

Tivemos como intenção principal para esses questionários a avaliação durante o desenvolvimento do jogo, fazendo dos professores parceiros no resultado final de um objeto de aprendizagem significativo em que as mesmas teriam voz no processo de confecção. Após esses diálogos aplicaremos então o jogo aos alunos, principais “alvos” de todo esse processo, vendo se os mesmos estão conseguindo aprender o que esta sendo proposto pela ferramenta e se a mesma esta com bons níveis de envolvimento, criando uma boa experiência para os alunos, fazendo-os querer jogar mais.

Sabemos que nem entre as professoras nem entre os alunos conseguiremos ter 100% de aprovação nem ser eficiente em relação a aprendizagem de 100% dos alunos, pois a educação é uma área que trabalha com organismos heterogêneos, sendo sempre necessária a adaptação do agente modificador (no caso, o professor). Logo pretendemos que essa ferramenta atinja pelo menos uma boa parcela dos alunos, principalmente aqueles que

possuem dificuldades no método tradicional, para sim ser uma alternativa complementar para o professor.

5.1 Primeiro Questionário Aplicado – Fazendinha em versão preliminar

O primeiro questionário (Anexo I) foi aplicado dia 25 de novembro de 2016 e contou com a participação de três professoras da rede municipal de educação da cidade de Vitória da Conquista, duas atuando na área de Matemática e uma na área de Ciências Sociais, cada uma delas contando com mais de 20 anos de carreira, tendo muita experiência em questões escolares e tendo visto a influência que a tecnologia teve ao longo dos anos na sociedade e na escola.

A primeira parte do questionário teve como foco a expectativa das professoras em relação a tecnologias e jogos em sala de aula, onde as três participantes concordam que os recursos escolares vigente são muito importantes para a prática escolar do professor, mas eles podem ser incrementados com elementos que trazem uma diversificação dentro da rotina de sala de aula. A participante 1 fala que apesar de não utilizar esses recursos, se sente bem quando consegue "mergulhar" um pouco no mundo do aluno. A participante 2 fala sobre o uso da calculadora, que deveria ser comum em sala mas há uma certa "proibição" na maioria das vezes durante as aulas de matemática, que já despertam o interesse dos jovens. A participante 3 diz que permite até mesmo o uso de celular e acesso à internet em algumas aulas, explorando o vasto mundo de informações que a mesma permite.

Elas também foram perguntadas como foi a sua experiência com sequências didáticas, principalmente em relação a Fazendinha Matemática analógica. As adaptações das atividades é o que mais chamam a atenção, pois são extremamente necessárias levando-se em consideração que as turmas são heterogêneas. Sobre se jogavam e se aplicariam jogos, a participante 1 afirmou que não jogava, porém que aplicaria jogos por questão metodológica, já que os mesmos a levavam para "mais perto" dos alunos. A participante 2 brincou dizendo que tinha parado por estar "viciada" em um determinado jogo, e concorda dizendo que os jogos ajudam na fixação do conteúdo além de trazerem o conteúdo de forma mais lúdica. A participante 3 diz que é uma jogadora assídua, e possui vários jogos em seu celular, ela diz que precisa se sentir "interessante" aos alunos para estar satisfeita como professora.

Na segunda parte foi feita uma observação com as professoras jogando o jogo. Nessa etapa o jogo não possuía a história, porém não houve prejuízo já que as mesmas já conheciam o enredo. Ao jogarem percebeu-se que as mesmas se envolveram bastante nas duas atividades principais, onde o jogo da memória trouxe uma dinamicidade muito parecida com a de jogos comerciais e o desafio do tempo gerou uma animosidade entre as jogadoras. Já na parte das trocas elas mesmo conhecendo já a fazendinha tiveram alguns problemas por conta da mecânica de ter que arrastar e soltar os animais (como era nessa versão preliminar), em que acharam não muito natural, e tiveram dificuldades em saber o que fazer em seguida, além de alguns botões não terem sido notados e algumas imagens estarem confusas.

Fizemos então a finalização do jogo e levamos em consideração alguns dos questionamentos feitos durante a fase preliminar. As principais alterações foram em relação aos botões, que foram redesenhados com um estilo mais próximo de objetos de fazenda, além de ganharem um padrão de fundo verde e uma placa que aparece quando o *mouse* passa por cima do botão, para identificar para que serve o botão. Outra alteração foi a retirada do modo de arrastar e soltar para levar os pintinhos e outros animais de uma área para outra, sendo convertido para um clique de botão apenas e setas para baixo e para cima transferindo os animais para a área de troca e de volta para as áreas iniciais. Também incrementamos as imagens do jogo, com desenhos próprios criados exclusivamente para essa história. Além dessas alterações foi acrescentada a parte de "contação" da história no inicio de jogo e elementos como o mapa do vilarejo e a parte de "Minha Fazenda".

5.2 Segundo Questionário Aplicado – Fazendinha em versão finalizada

No dia 27 de junho de 2017 foi aplicado novamente um questionário (Anexo II) com três alunas do curso de pedagogia da UESB, do quinto e sexto semestre. Uma (Estudante 1) delas já tendo trabalhado com a fazendinha em escolas municipais, uma (Estudante 2) tendo trabalhado durante o processo de desenvolvimento do jogo virtualizado e a outra (Estudante 3) não tido nenhum contado nem com o jogo virtualizado nem com o jogo analógico.

As três estudantes foram primeiramente questionadas sobre o que entendiam por sequências didáticas e desenho de tarefas, onde a Estudante 1 colocou que sequências didáticas eram conjuntos de tarefas que buscavam levar o aluno a atingir um conhecimento sobre algum conteúdo, e o desenho de tarefas possibilitava ao professor adaptar a sequência de acordo ao perfil de cada turma, ao que as outras estudantes concordam. Em relação a costume de jogar e ao que acham de jogos e recursos tecnológicos em sala a Estudante 1 diz que tem bastante costume de jogar no celular e pela internet, pensa nos jogos como recursos importantes para mudar a rotina do ensino tradicional e fala que as ferramentas tecnológicas são uma tendência que deve começar a ser abarcada pelas escolas, pois é inevitável. A Estudante 2 não tem muito costume como jogadora mas também vê os jogos como grandes objetos de aprendizagem, frisando que não há necessidade dos jogos serem muito elaborados para prenderem a atenção dos estudantes. A Estudante número 3 diz que em especial os jogos de perguntas e respostas a atraem bastante, e que considera tanto os jogos como as ferramentas tecnológicas como promotoras e como tendência, e especifica o uso do celular como um bom recurso que já é familiar ao estudante, e que pode ser utilizado pelo professor.

Elas jogaram o jogo em apenas um computador, as três em um modo cooperativo. Elas disseram que se divertiram bastante e as dificuldades foram mínimas, ocorrendo principalmente na Estudante 3 que nunca havia visto nada da história original, mas que logo pegou o sentido do mesmo. Elas aprovaram o jogo da memória com tempo e só apontaram alguns erros de ortografia, interpretação e coerência do texto com a imagem mostrada na história inicial. Elas também frisaram o bom uso que poderiam fazer das estatísticas finais mostradas no jogo, até mesmo levando em consideração (Estudante 1) que a própria estatística pode ser usada como uma ferramenta do assunto de estatística. A Estudante 2 diz que o jogo é auto-explicativo.

5.3 Terceiro Questionário Aplicado – Fazendinha em versão finalizada

Esse terceiro questionário (Anexo III) foi aplicado com duas das professoras que participaram do primeiro (Participantes 1 e 2), mais uma professora da área de linguagem (Participante 4) e um estudante do curso de Odontologia da UESB de Jequié (Participante 5). O questionário foi aplicado com eles apenas após o jogo, sendo que para os Participantes 4 e 5

toda a história era completamente nova. O jogo também foi jogado com todos os participantes em apenas um computador com todos dando opiniões sobre o que fazer a seguir.

A primeira consideração deste questionário foi feita com as Participantes 1 e 2, sobre quais as diferenças que sentiram da versão preliminar para essa versão finalizada. A participante 1 fala que houve uma melhora muito grande em relação a imagem e som, além da logística do jogo. A Participante 2 acrescenta que a contação da história no inicio deu outra cara ao jogo, no que a Participante 1 complementa com a ideia de pertencimento que a história inicial deu na experiência como um todo. O Participante 5 diz que gostou bastante do jogo, mas durante o *gameplay* percebeu alguns *bugs* que ocorreram.

Em relação às dificuldades sentidas a Participante 2 comenta que os recursos do jogo ajudaram bastante a aliviar as dúvidas sobre o próximo passo, em contrapartida com a versão preliminar do jogo. O Participante 5, que nunca havia ouvido falar sobre a fazendinha anteriormente, fala que de inicio sentiu um pouco de dificuldade, porém que não era diferente da dificuldade sentida em outros jogos, e concorda com a Participante 2 que os recursos do jogo foram sanando as dúvidas aos poucos. A Participante 4 fala que vê como importante um jogo com sentido educativo onde não há perdedores, apenas ganhadores que conseguem terminar as coisas em seu próprio tempo. A Participante 1 diz que gostou do elemento de tempo no jogo da memória, comentando que achou mais instigante do que um jogo da memória comum. O participante 5 fala que achou mais difícil mas que também gostou, no que as Participantes 2 e 4 concordaram.

Sobre as estatísticas finais como uma forma de avaliação do desempenho do aluno, a Participante 2 foi categórica em dizer que apesar de serem dados relevantes que podem ser trabalhados, deve-se tomar cuidado para não classificar os alunos, pois cada qual tem o seu próprio tempo de aprendizagem. Sobre as dificuldades em se aplicar o jogo em sala de aula, as três professoras concordam que a parte estrutural complicaria, pois não há computadores suficientes e nem em boas condições dentro das escolas para se fazer atividades do tipo. A Participante 1 fala também sobre a maturidade do aluno em usar tais recursos, por que apesar da riqueza que traz para a aula, ferramentas como o celular podem destruir completamente a atenção do aluno no assunto a ser trabalhado. Ela também acrescenta sobre o balanceamento que se deve ter com o sistema chamado “escola” e as novas tecnologias, que devem ser pensadas dentro desse sistema que apesar de antigo e por vezes obsoleto, é o que esta vigente hoje e que de certa forma tem funcionado. A Participante 2 comenta sobre a falta de apoio que o professor tem em relação a essas questões, pois muitas vezes o mesmo não tem conhecimento para manipular tais ferramentas e não conseguiria conduzir completamente a

aula nem se adaptar a possíveis dificuldades, sendo necessário um profissional da área para auxilia-lo. O Participante 4 concorda com o que foi dito porém acrescenta que mesmo com as dificuldades vale a pena aplicar o jogo pois é algo que além de ter todas as potencialidades já comentadas, pode também servir de estímulo para que outras ferramentas possam ser desenvolvidas até mesmo pelos próprios alunos.

5.4 Resultados obtidos

Mesmo não tendo tido uma base grande de análise para relacionar dados estatísticos, pudemos observar com essas aplicações que há espaço para jogos eletrônicos em sala de aula tanto por parte de professores da geração atual, quanto das novas gerações que ainda virão a estar em sala de aula. Vimos também que há duas grandes dificuldades na aplicação dos jogos em sala de aula, que são: a falta de bons equipamentos em escolas públicas no geral, e a falta de formação dos professores para com a manipulação dessas ferramentas. Portanto, apesar de o jogo ser pensado para o exercício em sala de aula, ele pode ser mais generalizado para que qualquer pessoa possa tê-lo em casa e joga-lo como qualquer outro jogo de computador, além de podermos abrir uma frente para uma versão *mobile*, bastante citada durante as entrevistas.

Em relação às entrevistas, percebemos que a aplicação de versões preliminares em muito contribuiu para a evolução do jogo, sendo que as professoras deram várias opiniões sobre como o jogo deveria ser feito e como a maleabilidade com o recurso ajuda na adaptação do mesmo às turmas. Percebemos que dessa forma conseguimos seguir a proposta inicial de desenvolver junto às especialistas da área educacional e o quão rico o jogo torna-se com esse cuidado.

Sobre o jogo em si, vimos que apesar de estarmos seguindo uma boa evolução, ainda precisamos corrigir e melhorar alguns pontos necessários. Os *bugs* encontrados, apesar de não terem afetado diretamente a experiência de jogo precisam ser imediatamente localizados e resolvidos. Outro problema percebido foi com as caixas de diálogo, pois muitas delas aparecem por cima da tela, mas não desabilitam os componentes por trás, fazendo com que os usuários cliquem em elementos que eles deveriam interagir somente após a caixa de diálogo desaparecer, esse problema também afetou a aparição das atividades, pois as mesmas necessitam de um tempo após a subida de nível para aparecer, porém alguns jogadores clicavam rapidamente no botão de continuar jogo e acabavam por impedir que a janela de atividades aparecesse. Alguns elementos acabaram por serem extras sem sentido no jogo

como a coleção de “Tipos de Pintinhos” e a “Minha Fazenda” que foram acessadas somente uma vez durante todo o *gameplay*, mas que não traziam motivação suficiente aos jogadores para que os mesmos desfrutassem do intento de criar objetivos diferentes. Porém alguns elementos ficaram tão bons ou superaram as nossas expectativas, como o jogo da memória com tempo que todos pareceram gostar bastante e a história inicial, em que todos elogiaram os desenhos e a apresentação.

6 Conclusões e Trabalhos Futuros

Em nossas considerações finais, percebemos que apesar do enorme potencial ainda a ser explorado por esse jogo, conseguimos cumprir o nosso intento na virtualização da sequência didática em formato de jogo eletrônico, sendo bem recebido por professoras e futuras professoras da rede de ensino. Vimos que a virtualização de sequências didáticas já consolidadas e a participação de especialistas na área de educação dá maiores possibilidades ao jogo educacional, além de terem maior aceitação. Concluímos também que a ferramenta Unity3D é bastante abrangente na criação do jogo, facilitando bastante o uso de recursos multimídia, além de ter uma curva de aprendizagem pequena para pessoas já acostumadas a manipulação de ferramentas virtuais.

Pretendemos agora trabalhar nas correções já apontadas no capítulo anterior além de expandir o jogo, tornando a versão atual na primeira fase de um jogo maior, que continua com a fazenda conquistada, criando outros elementos de conflito na história para que possamos dar mais objetivos ao jogador, sem nos descuidar da parte educacional, com a criação de *puzzles* e atividades significativas para o aprendizado. Em versões posteriores também pretendemos inserir o jogo em forma *multiplayer* para que os alunos possam atuar no mesmo mundo, fazendo com que a interação social seja mais visível durante o *gameplay*.

Um trabalho maior, que deverá ser aplicada em um projeto de mestrado do presente autor, consistirá em uma ferramenta de construção do jogo, baseado em informações do banco de dados, permitindo ao professor alterar os diálogos, imagens e sons presentes no jogo, para que o mesmo possa adaptar a história para diferentes faixa-etárias, como também selecionar

atividades e desafios dentro de um conjunto maior, permitindo também configurar os níveis e a progressão dos alunos durante o jogo. A ideia é que o professor possa “construir” sua própria fazendinha de acordo ao perfil de sua turma, aproximando o mesmo ainda mais do desenvolvimento.

Consideramos este um trabalho de grande valia tanto no tocante ao desenvolvimento com a ferramenta Unity3D, permitindo que tal trabalho seja um ponto de partida para futuros desenvolvedores, quanto na construção de jogos educacionais, mostrando os cuidados que se deve ter com esse tipo de jogo e como os mesmos poderão ser avaliados.

Referências Bibliográficas

ANDRADE, Vinicius Silva. **AVALIAÇÃO DA COMUNICABILIDADE EM JOGOS DE DISPOSITIVOS MÓVEIS: um estudo da relevância dos signos em jogos Tower Defense**. Projetos e Dissertações em Sistemas de Informação e Gestão do Conhecimento, v. 2, n. 2, 2013.

BATISTA, Gabriel; NOVAES, Luiza; FARBIARZ, Alexandre. **Jogos: desenvolvendo competências e habilidades**. In: VIII Brazilian Symposium on Games and Digital Entertainment. Rio de Janeiro, 2009.

BATTAIOLA, André L.; ELIAS, Nassim C.; DOMINGUES, Rodrigo G. et al **Desenvolvimento de um Software Educacional com Base em Conceitos de Jogos de Computador**. In: XIII Simpósio Brasileiro de Informática na Educação. São Leopoldo: SBC, 2002

CARNIELLO, Luciana Barbosa Cândido; RODRIGUES, Bárbara Mônica Alcântara Gratão; MORAES, Moema Gomes. **A relação entre os nativos digitais, jogos eletrônicos e aprendizagem**. SIMPÓSIO DE HIPERTEXTO E TECNOLOGIAS NA EDUCAÇÃO, v. 3, 2010.

CLUA, E., BITTENCOURT, J. **Desenvolvimento de Jogos 3D: Concepção, Design e Programação**. Anais da XXIV Jornada de Atualização em Informática do Congresso da Sociedade Brasileira de Computação, São Leopoldo, Brazil, Julho de 2005.

COSTA, G. M. T.; PERETTI, L. **Sequência didática na Matemática**. *REI. Revista de Educação do IDEAU*, Getúlio Vargas, v. 8, p. 01-14, jan. 2013.

CRAWFORD, C. (1982). **The Art of Digital Game Design**, Washington State University, Vancouver, 1982.

DE OLIVEIRA, Osvaldo Luiz; BARANAUSKAS, M. Cecília Calani. **A Semiótica e o Design de Software**. 1998.

DE OLIVEIRA MEDEIROS, Maxwell; SCHIMIGUEL, Juliano. **Uma Abordagem para avaliação de jogos educativos: ênfase no ensino fundamental**. In: Brazilian Symposium on Computers in Education (Simpósio Brasileiro de Informática na Educação-SBIE). 2012.

DESENVOLVIMENTO DE JOGOS WIKIDOT. Trabalhando com GUI na Unity, 2017. Disponível em: <http://desenvolvimentodejogos.wikidot.com/unitygui>. Acesso em: 05/05/2017

DIAS, Gabriel Almeida; GUSMÃO, Tânia Cristina Rocha Silva. **Virtualização da Sequência Fazendinha Matemática**. In: Anais do Simpósio Internacional de Pesquisa em Educação Matemática, 4º, 2015, Ilhéus. p.282-287.

DIAS, Gabriel Almeida; GUSMÃO, Tânia Cristina Rocha Silva; FREDINI, Pablo Sebastian Rojas; DE MOURA, Humberto Plácido Gusmão. **Fazendinha Matemática do jogo físico ao jogo virtual: Trabalhando as operações fundamentais da Matemática**. In: *Anais* do XI

nacional e IV Colóquio Internacional do Museu Pedagógico, 2015, Vitoria da Conquista. Crise, conflito e conhecimento no mundo contemporâneo, 2015.

DIAS, Raphael, Game Engine: o que é, para que serve e como escolher a sua, 2017, Disponível em: <http://producaodejogos.com/game-engine/>. Acesso em: 01/05/2017.

DIVERIO, Tiarajú Asmuz. e MENEZES, Paulo Blauth. **Teoria da computação: máquinas universais e computabilidade**. Porto Alegre, Editora Sagra Luzzatto, 2000.

DOHME, Vania. **Atividades lúdicas na educação: o caminho de tijolos amarelos do aprendizado**. 4ed. – Petrópolis, RJ: Vozes, 2008

DRAWING NOW, 2017. Disponível em: < www.drawingnow.com >. Acesso em: 20/05/2017

DUFLOS, C**. O jogo: De Pascal a Schiller**. Porto Alegre: Artmed, 1999.

ELMASRI, Ramez; NAVATHE, Shamkant B.; DE OLIVEIRA MORAIS, Rinaldo. **Sistemas de banco de dados**. 2005.

FLAUSINO, Rodrigo. Versão gratuita da "Unity 5" é lançada com todas as funcionalidades da versão Pro. 2005. Disponível em: <http://selectgame.gamehall.uol.com.br/versao-gratuita-da-unity-5-e-lancada-com-todas-funcionalidades-da-versao-pro/>. Acesso em: 02/05/2017

FLEURY, Afonso; SAKUDA, Luiz Ojima; CORDEIRO, José Henrique Dell Osso. **I Censo da Indústria Brasileira de Jogos Digitais**. NPGT-USP e BNDES: São Paulo e Rio de Janeiro, 2014.

FREE PIK, 2017. Disponível em: < br.freepik.com>. Acesso em: 05/05/2017

GET COLORING PAGES, 2017. Disponível em: < www.getcoloringpages.com>. Acesso em: 10/05/2017

GOULD, J.; LEWIS, C. **Designing for usability: key principles an what designers think. Comunicattions of the ACM**, Volume 28, Issue 3, 1985, pp 300-311.

GUSMÃO, Tânia C.R.S. **Desenho de tarefas para o desenvolvimento da cognição e metacognição matemática**. Anais do I Colóquio Internacional sobre Ensino e Didática das Ciências: Contribuições e Perspectivas. Feira de Santana, 27 a 31 de outubro de 2014b.

HAVE GO NAVEGAN, 2017. Disponível em: < www.havegonevegan.com>. Acesso em: 17/05/2017

HUIZINGA, Johan. **Homo ludens: o jogo como elemento da cultura**. 4ed. São Paulo: Perspectiva, 2000.

KIDS PLAY COLOR, 2017. Disponível em: < www.kidsplaycolor.com>. Acesso em: 13/05/2017

KISHIMOTO, André. **Inteligência artificial em jogos eletrônicos**. Academic research about Artificial Intelligence for games, 2004.

LACANALLO, Luciana Figueiredo; SILVA, Sandra Salete de Camargo Silva; DE OLIVEIRA, Diene Eire de Mello Bortotti; GASPARIN, João Luiz; TERUYA, Teresa Kazuko. **Métodos de ensino e de aprendizagem: uma análise histórica e educacional do trabalho didático**. VII Jornada do Histedbr-O trabalho didático na história da educação. Atas do Evento, Campo Grande, 2007.

LAIRD, John. VAN LENT, Michael. **Human-level AI's Killer Application: Interactive Computer Games** In: AI Magazine, v.22, n.2, 2001, pp. 15-25.

LEMOS, Silvana. **Nativos digitais x aprendizagens: um desafio para a escola**. Boletim Técnico do Senac, v. 35, n. 3, p. 38-47, 2009.

LUCCHESE, Fabiano; RIBEIRO, Bruno. **Conceituacao de jogos digitais**. Sao Paulo, 2009.

MACEDO FILHO, M. D. **A relevância de interfaces gráficas amigáveis**. I seminário de jogos eletrônicos, eduação e comunicação–construindo novas trilhas, no GT–Desenvolvimento de Games/UNEB, n. 1. 2005.

MANSSOUR, Isabel Harb; COHEN, Marcelo. **Introdução à Computação Gráfica**. RITA, v. 13, n. 2, p. 43-68, 2006.

MATTOS, B. A. M. **Uma Extensão do Método de Avaliação de Comunicabilidade para Sistemas Colaborativos**. Dissertação de Mestrado. Belo Horizonte: [s.n.], 2010

MATTAR, João – De Mattar, 2009. Disponível em: <http://blog.joaomattar.com/>. Acesso em: 22 mar. 2017.

MENEZES FILHO, Naércio Aquino. **Os determinantes do desempenho escolar do Brasil**. IFB, 2007.

MP3 CUT, 2017. Disponível em: <http://mp3cut.net>. Acesso em: 05/06/2017

NERY FILHO, Jesse. **Prototipagem de um gamebook para potencializar as funções executivas**. 2015.

NEWZOO. **2016. Global Games Market Report: An Overview of Trends & Insights**. Junho, 2016.

QUEIROZ, Heverton Santos. **DESENVOLVENDO JOGOS COM O GAME MAKER**. 2012.

ONLINE IMAGE EDITOR, 2017. Disponível em: <www.online-image-editor.co>. Acesso em: 05/06/2017

PEIRCE, C. S. **Semiótica**. 2005. Editora Perspectiva, São Paulo, 2005.

PIETRUCHINSKI, Mônica Hoeldtke; NETO, João Coelho; MALUCELLI, Andreia; REINEHR, Sheila. **Os jogos educativos no contexto do SBIE: uma revisão sistemática de Literatura**. In: Brazilian Symposium on Computers in Education (Simpósio Brasileiro de Informática na Educação-SBIE). 2011.

PINTO, Ivete Martins; BOTELHO, Silvia Silva da Costa. **Ambientes Tecnológicos Lúdicos de Autoria (ATLA): criando espaços de ensino e aprendizagem**. 2012.

POCHULU, M. FONT, V. e RODRIGUEZ, M. **Criterios de diseño de tareas para favorecer el análisis didáctico em la formación de profesores.** In: *Actas* del VII CIBEM. Montevideo: Uruguai. 2013.

PRATES, R. O; BARBOSA, S. D. J. **Introdução à Teoria e Prática da Interação Humano-Computador fundamentada na Engenharia Semiótica**. In T. Kowaltowski & K. Breitman (orgs.) Jornadas de Atualização em Informática, JAI 2007, pp. 263-326.

PREECE, J.; ROGERS, Y.; SHARP, H.; BENYON, D.; HOLLAND, S.; CAREY, T. **Human-computer interaction**. Reading, MA. Addison-Wesley, 1994.

PRENSKY, Marc. **Digital natives, digital immigrants part 1**. On the horizon, v. 9, n. 5, p. 1-6, 2001.

PRINTABLE FREE COLORING, 2017. Disponível em: < printablefreecoloring.com>. Acesso em: 12/05/2017

RABIN, S. Jogos e a Sociedade. In: RABIN, S. **Introdução ao Desenvolvimento de Games**. São Paulo: Cengage Learning, 2011.

RAMOS, Márcio Roberto Vieira. **O uso de tecnologias em sala de aula**. V Seminário de Estágio do Curso de Ciências Sociais do Departamento de Ciências Sociais-UEL. Londrina, v. 11, p. 2012, 2012.

REIS, FVDOS. **Jogo da cabanagem: projeto e implementação**. Monografia (Graduação em Ciência da Computação). Instituto de Tecnologia. Universidade Federal do Pará. Belém–Pará, 2009.

RIOS, Ivana Souza Vanessa; ALVES, Lynn. Games e cultura: **Búzios: ecos da liberdade–uma leitura da história da Bahia**. 2010.

PRINTABLE FREE COLORING, 2017. Disponível em: < www.rolofffarms.com>. Acesso em: 18/05/2017

SANTANA, Bianca. **Panorama setorial da Internet. Recursos digitais na escola: repensando caminhos**. Tecnologia e Educação. Ano 6. Número 3, Outubro, 2014.

SCHELL, Jesse. **The Art of Game Design: A book of lenses**. CRC Press, 2011.

SCHUYTEMA, Paul. **Design de games: uma abordagem prática**. Cengage Learning, 2008.

SOUZA, C. S. **The Semiotic Engineering of User Interface Languages**. *International Journal of Man-Machine Studies*, v. 39, 1993, p. 753-773.

TAPSCOTT, D. **A hora da geração digital: como os jovens que cresceram usando a internet estão mudando tudo, das empresas aos governos**. Rio de Janeiro: Agir Negócios, 2010.

TAKAI, Osvaldo Kotaro; ITALIANO, Isabel Cristina; FERREIRA, João Eduardo. **Introdução a banco de dados**. DCC-IMEUSP: Fevereiro, 2005.

VIEIRA, K. L. A. S. **As contribuições formativas de uma sequência didática para atuação dos pedagogos no ensino da matemática nos anos iniciais**. 2015. f. 175. Dissertação de Mestrado em Educação Científica e Formação de Professores – Universidade Estadual da Bahia – UESB, Jequié.

WANG, W. S. **O aprendizado através de jogos para computador: por uma escola mais divertida e mais eficiente**. Disponível em: <http://www.portaldafamilia.org/arqs/Aprendizado_atraves_de_jogos_para_computador.pdf>. Acesso em: 06 mai. 2017.

XAVIER, Thomaz Canali; DOS SANTOS, Maurício; CARUSO, André Luis Macedo. **ESTUDO E DESENVOLVIMENTO DE JOGOS PARA INTERNET UTILIZANDO UNITY 3D**. 2011

Anexo I

Questionário aplicado no dia 25 de novembro de 2016 com três professoras da rede pública de ensino:

1 – Qual seu nome?
2 – Qual sua idade?
3 – Qual disciplina leciona?
4 – Leciona para alunos de que idade?
5 – Quantas horas/dias por semana em cada turma?
6 – Além do método tradicional de ensino (Quadro, explicação, testes, provas, apresentação de trabalhos), quais outros recursos você já utilizou em sala de aula?
7 – Já utilizou alguma sequência didática em suas aulas? Se sim, como foi a experiência?
8 – Você costuma jogar algum jogo em suas horas vagas?
9 – Utilizou ou utilizaria algum jogo pedagógico em sua aula? Caso já tenha utilizado, como foi a experiência? Caso pense, ou caso não pense, em utilizar, por qual motivo?

Anexo II

Questionários aplicados no dia 27 de junho de 2017 com três alunas do curso de pedagogia da UESB.

1 – Qual seu nome?
2 – Qual sua idade?
3 – Qual seu curso?
4 – O que você conhece de sequências didáticas? Já utilizou alguma? E de desenho de tarefas?
5 – Você costuma jogar algum jogo em suas horas vagas?
6 – O que pensa sobre jogos como objetos promotores de aprendizagem? Houve algum estudo durante o curso?
7 – Já utilizou (em estágios) ou participou de algum jogo pedagógico em aula? Como foi a experiência?

Após o jogo

8 – Sentiu alguma dificuldade em saber o que fazer?
9 – Você pode comentar um pouco sobre a história do jogo?
10 – Achou esse estilo de jogo da memória (com tempo) mais ou menos difícil do que o jogado com várias pessoas? Alguma preferência entre eles?
11 – O que achou das estatísticas finais do jogo? Conseguiria trabalhar com elas? O que mudaria?
12 – As trocas te fizeram lembrar algum conceito matemático?

(Explicação dos conceitos matemáticos que envolvem a fazendinha matemática)

13 – Depois de ver o jogo, você aplicaria o mesmo em sala de aula?
14 – Quais dificuldades você sentiria em coloca-lo aos alunos?
15 – Quais estratégias você adotaria para conciliar o jogo às aulas?
16 – O que você acha que falta no jogo? Quais modificações você faria?

Anexo III

Questionários aplicados no dia 27 de junho de 2017 com três professoras da rede municipal de ensino e um aluno do curso de Odontologia da UESB (campus de Jequié).

1 – Quais mudanças mais relevantes você sentiu da primeira vez em que jogou para agora?
2 – Sentiu alguma dificuldade em saber o que fazer?
3 – Você pode comentar um pouco sobre a história do jogo?
4 – Achou esse estilo de jogo da memória (com tempo) mais ou menos difícil do que o jogado com várias pessoas? Alguma preferência entre eles?
5 – O que achou das estatísticas finais do jogo? Conseguiria trabalhar com elas? O que mudaria?
6 – Depois de ver o jogo, você aplicaria o mesmo em sala de aula?
7 – Quais dificuldades você sentiria em coloca-lo aos alunos?
8 – Quais estratégias você adotaria para conciliar o jogo às aulas?
9 – O que você acha que falta no jogo? Quais modificações você faria?