UNIVERSIDADE FEDERAL DO PIAUÍ
CAMPUS SENADOR HELVÍDIO NUNES DE BARROS
CURSO DE BACHARELADO EM SISTEMAS DE INFORMAÇÃO

+MUNDO: UM JOGO EDUCATIVO SOBRE DESENVOLVIMENTO SUSTENTÁVEL

JULIANO VIEIRA ROCHA

PICOS – PIAUÍ
2016

JULIANO VIEIRA ROCHA

+Mundo: Um Jogo Educativo sobre Desenvolvimento Sustentável

Monografia submetida ao Curso de Bacharelado de Sistemas de Informação como requisito parcial para obtenção de grau de Bacharel em Sistemas de Informação.

Orientador: Prof. Esp. Ismael Leal Holanda

PICOS – PIAUÍ
2016

FICHA CATALOGRÁFICA

Serviço de Processamento Técnico da Universidade Federal do Piauí

Biblioteca José Albano de Macêdo

R672m Rocha, Juliano Vieira.

+ mundo: um jogo educativo sobre desenvolvimento sustentável / Juliano Vieira Rocha.– 2016.

CD-ROM : il.; 4 ¾ pol. (39 f.)

Monografia (Curso Bacharelado em Sistemas de Informação) – Universidade Federal do Piauí, Picos, 2016.

Orientador(A): Prof. Esp. Ismael Leal Holanda

1.Jogo Educativo-Sustentabilidade. 2.Problemas Ambientais-Tecnologia. 3. Desenvolvimento Sustentável-Jogo Eletrônico. I. Título.

CDD 005.3

+MUNDO: UM JOGO EDUCATIVO SOBRE DESENVOLVIMENTO SUSTENTÁVEL

JULIANO VIEIRA ROCHA

Monografia APROVADA como exigência parcial para obtenção do grau de Bacharel em Sistemas de Informação.

Data de Aprovação

Picos – PI, 24 de FEVEREIRO de 2016

Profª. Esp. Ismael de Holanda Leal
Orientador

Prof. Me. Márcio Alves de Macêdo
Membro

Prof. Esp. Pablo de Abreu Vieira
Membro

Dedico este trabalho primeiramente a Deus, pois foi Ele o maior responsável por me fazer chagar até aqui. Também dedico a toda a minha família, em especial minha mãe, Efigênia, meu pai, Everardo, e meu irmão Julielton. Dedico também a todos os meus amigos que contribuíram positivamente de alguma forma nessa longa e árdua caminhada.

AGRADECIMENTOS

Agradeço primeiramente a Deus por tudo que proporcionou na minha vida até aqui. Nos momentos em que desistir parecia ser a única alternativa, Ele esteve comigo e me orientou a fazer as melhores escolhas. Agradeço a Deus também por todas as boas pessoas presentes na minha vida que se propuseram a ajudar quando necessitei.

Agradeço de forma especial aos meus pais pelo apoio, pela força, pelas brigas e por estar comigo desde sempre. Obrigado Dona Efigênia e Senhor Everardo por toda a ajuda. Agradeço ao meu irmão pela parceria e acima de tudo, por sua grande amizade. Agradeço a toda minha grande família, espalhada por todo o Brasil, e que sempre torceram pelo meu sucesso.

Gostaria de Agradecer aos meus grandes amigos de vida, William 1 e 2, Denise, Douglas e Júnior. Obrigado por me ouvir reclamar dos trabalhos, dos tccs e das provas, por falar que tudo vai dar certo, mesmo não sendo possível, e ainda me fazer acreditar nisso, e por todas as vezes que precisei de qualquer coisa e vocês me estenderam a mão. Não teria chegado neste momento sem a ajuda de vocês. Nossa irmandade foi de fundamental importância.

Agradeço aos meus companheiros de UFPI e parceiros Wellington, Walison, Jeferson e José Roberto, que tornaram minhas lutas diárias no decorrer do curso mais fáceis e dividiram comigo grandes momentos. Nossa amizade vai além da Universidade. Agradeço também as minhas parceiras de apartamento e grandes amigas de vida, Káttia e Tamires, por todo o apoio ao longo desses anos, pelo pão de cada dia, pela casa limpa, e acima de tudo, por tornar meus dias menos tediosos.

Agradeço a todos os professores que compartilharam com um pouco de seus conhecimentos e me proporcionaram ser bem mais inteligente que antes. Aos professores Rayner, Frank César, Dennis, Leonardo, ivenilton, Fredson, Juliana, Alcilene, Romuere, Ryan e Laurindo, meu muito obrigado por tudo que me foi ensinado. Agradeço de forma especial a professora e “pessoa” Medyna que sempre se esforça do fundo do coração para que todos seus alunos adquiram a melhor educação possível. Agradeço ao meu orientador de TCC1, Flávio Henrique, que me

ajudou muito para que todas as minhas dúvidas fossem tiradas. E finalmente ao meu orientador de TCC2 e amigo, Ismael Leal, por todo o conhecimento que me foi repassado na criação deste trabalho, e mais do que isso, por toda a ajuda desde o início do curso.

RESUMO

Viver de forma sustentável é essencial para que as novas gerações, além dessa própria, possuam uma melhor qualidade de vida. O mundo sofre com diversos problemas por consequência do homem, e para a solução da grande maioria deles são necessárias novas atitudes. Os jogos educativos são atualmente uma forma bastante interessante e divertida de educar as crianças alertando-as sobre os problemas enfrentados pelo mundo. Vários *games* já foram criados para ajudar na preservação dos recursos do meio ambiente, abordando temas diversos, como o desperdício de energia elétrica, práticas de reciclagem, redução do lixo eletrônico, entre outros. O presente trabalho tem como objetivo demonstrar o desenvolvimento de um jogo educativo no qual visa informar quem joga sobre como utilizar-se de práticas sustentáveis, e melhorar, ou tentar melhorar a sua educação ambiental. O *Unity 3D* é uma importante tecnologia utilizada no desenvolvimento de jogos profissionais. Sua utilização diminui a necessidade de códigos complexos, reduzindo assim o trabalho do desenvolvedor. O trabalho foi desenvolvido utilizando tecnologias como: a modelagem UML; o motor gráfico *Unity 3D*; e a ferramenta para edição de imagem *Photoshop*; diversos testes foram realizados ao longo de seu desenvolvimento para verificar se o *game* está de fato em seu pleno funcionamento e se necessário, realizar a correção de bugs.

Palavras-chave: Jogo Educativo, Sustentabilidade, Problemas Ambientais

LISTA DE FIGURAS

LISTA DE ABREVIATURAS E SIGLAS

CASE	Engenharia de Software Auxiliada por Computador
IA	Inteligência Artificial
IDE	Integrated Development Environment
UML	*Unified Modeling Language*

SUMÁRIO

1 INTRODUÇÃO

O termo consumo está estritamente ligado com desenvolvimento. O mundo é dominado pelo espirito capitalista onde o consumo é o ápice do ideal da sociedade. Para satisfazer o consumo é necessário produzir cada vez mais, obrigando como consequência que o ser-humano retire recursos da natureza a fim de atende-lo, gerando assim um colapso ambiental. Hoje é possível presenciar as consequências em todos os lugares.

A redução da qualidade de vida e a ameaça à sobrevivência humana devido a degradação dos recursos naturais fez com que a questão ambiental ganhasse atenção mundial. Atualmente a relação ser humano-meio ambiente começou a desempenhar uma maior importância. O *marketing* passou a preocupar-se mais com a comunidade. Projetos sustentáveis são cada vez mais comuns. No entanto, ainda falta muito para ser feito. É necessário que todos revejam seus conceitos, mudando o paradigma que a relação de consumo e o desenvolvimento andam juntos.

CAPRA (2006 a,b) diz que:

> "O ser humano contemporâneo passa por uma fase de autoconhecimento e de reflexão do seu papel perante a vida e o planeta, tentando entender os reflexos e os custos do altíssimo desenvolvimento das sociedades; acima de tudo, estamos aprendendo com os erros para, a partir deles, buscar soluções a médio, curto e longo prazo. Uma delas é encontrar a humanidade que nos falta, que nos fará sermos mais completos, mais conscientes e mais voltados para nossa própria espécie".

Por ser um tema que exige mudança de comportamento e conscientização por parte de cada um, leva-se tempo para ser alcançado. O ideal seria se a educação ambiental já estivesse formalizada desde a infância.

Para SIRVINSKAS (2011):

> "A educação ambiental deve estar fundamentada na ética ambiental. Entende-se por ética ambiental o estudo dos juízos de valor da conduta humana em relação ao meio ambiente. É, em outras palavras, a compreensão que o homem tem da necessidade de conservar ou preservar os recursos naturais essenciais à perpetuação de todas as espécies de vida existentes no planeta. Existem diversas alternativas para conscientizar a sociedade desde

> cedo. Uma delas e boa alternativa são os jogos educativos, que por proporcionar a sensação de imersão e ainda por fazer parte da grande maioria das pessoas, torna-se ideal para chamar a atenção das crianças. ”

O mercado de jogos é o que mais cresce a cada ano e está presente na vida da maioria das pessoas, seja ela adulta ou criança. O Brasil é hoje um dos maiores mercados de *game*s do planeta.

ABRAGAMES (2011) afirma que o mercado de *game*s:

> “Em 2009 faturou $57 bilhões contra $29 bilhões do cinema e em 2012 a previsão era de $71 bilhões. Segundo pesquisas realizadas em 2011, estima-se que que hajam 35.000.000 jogadores ativos no Brasil, dos quais, 47% gastam dinheiro em jogos, o que demonstra o potencial do mercado de jogos no Brasil, apesar dos altos impostos cobrados sobre os jogos e os altos índices de pirataria. ”

Diferente da realidade mais antiga, hoje o desenvolvimento de um *game* é muito mais simples. Diversos materiais de apoio foram desenvolvidos, *frameworks*[1], bibliotecas de programação e motores gráficos gratuitos. Uma tarefa realizada por uma equipe de profissionais de diferentes especialidades pode ser feita atualmente por uma única pessoa.

1.1 Objetivos

O principal objetivo deste trabalho é desenvolver um jogo educativo 2D da categoria plataforma, utilizando o motor gráfico *Unity 3D* para informatizar (utilizando-se da diversão que um *game* proporciona) e alertar as crianças sobre os problemas que são causados diariamente ao meio ambiente, para que assim todos saibam o quão importante é cada um fazer a sua parte na busca por uma vida melhor e mais sustentável.

1.2 Organização do Documento

Este documento está estruturado da seguinte maneira:

[1] Em desenvolvimento de software, é uma abstração que une códigos comuns entre vários projetos de software provendo uma funcionalidade genérica.

- Capítulo 2 – Referencial Teórico: Nesse capítulo são apresentadas as tecnologias e ferramentas que foram utilizadas para o desenvolvimento do trabalho.
- Capítulo 3 – Problemática: São mostrados, de maneira objetiva, os problemas encontrados que levaram à necessidade do jogo educativo +Mundo.
- Capítulo 4 – +Mundo: É apresentado o desenvolvimento do jogo, como cada ferramenta foi utilizada, assim como seus requisitos, diagramas e funcionamento.
- Capítulo 5 – Considerações Finais: É apresentada a conclusão do trabalho, bem como trabalhos futuros a serem desenvolvidos para melhoria e ampliação do jogo.

2 REFERENCIAL TEÓRICO

Nessa seção, são apresentadas as Tecnologias e Ferramentas que foram utilizadas para o desenvolvimento deste trabalho.

2.1 Tecnologias

Durante o desenvolvimento do *game*, foram utilizadas várias tecnologias para a obtenção do resultado esperado. Todas elas são descritas abaixo.

2.1.1 Levantamento de Requisitos

Para o desenvolvimento de um bom *software* é necessário primordialmente que seja feito a Análise de Requisitos. Todos os serviços que o sistema vai realizar deve estar presente nela, desde os elementos mais profundos do sistema até os mais próximos do usuário.

Preleciona WAZLAWICK (2004) que:

> "O levantamento preliminar de requisitos tem por objetivo prover uma visão do todo, para poder definir o que é mais importante e depois dividir o todo em partes para especificar os detalhes. Nessa fase, o levantamento é rápido e genérico, sendo feito em extensão e não em profundidade. O analista deve entender a extensão do que o sistema deve fazer, mas sem entrar em detalhes. Somente nos ciclos iterativos os requisitos serão detalhados, especificados e modelados. "

É importante destacar que *software*s educativos tem uma fase de análise diferente em comparação com um *software* comum. Por possuir características educacionais, seu processo de criação torna-se mais complexo.

LACERDA (2007) fala que:

> "No caso dos *software*s educativos a fase de elicitação se mostra ainda mais complexa quando comparada com *software*s convencionais, que não são de domínio tão específico, por envolver *stakeholders* de diferentes áreas de conhecimento (alunos, educadores, programadores, *design*ers, empresários). Devido a essa característica dos *software*s educativos é recomendada a formação de uma equipe multidisciplinar. "

Modelagem UML

Depois de analisar quais as características estarão presentes no desenvolvimento do *software*, documenta-lo é de extrema importância.

Segundo TONSIG (2003):

> “Para conseguir desenvolver um *software* capaz de satisfazer as necessidades de seus usuários, com qualidade, por intermédio de uma arquitetura sólida que aceite modificações, de forma rápida, eficiente, com o mínimo de desperdício e retrabalho, faz-se necessário o emprego de modelagem. ”

A UML (*Unified Modeling Language*) fornece a tecnologia necessária para apoiar a prática de engenharia de *software*, auxiliando na visualização do desenho e comunicação entre os objetos do sistema,

De acordo com a OMG (2006):

> “A UML (*Unified Modeling Language*) é uma linguagem para especificação, documentação, visualização e desenvolvimento de sistemas orientados a objetos. Sintetiza os principais métodos existentes, sendo considerada uma das linguagens mais expressivas para modelagem de sistemas orientados a objetos. Por meio de seus diagramas é possível representar sistemas de *software*s sob diversas perspectivas de visualização. Facilita a comunicação de todas as pessoas envolvidas no processo de desenvolvimento de um sistema - gerentes, coordenadores, analistas, desenvolvedores - por apresentar um vocabulário de fácil entendimento. ”

FURLAN (1998) fala sobre a variedade de diagramas presentes na UML:

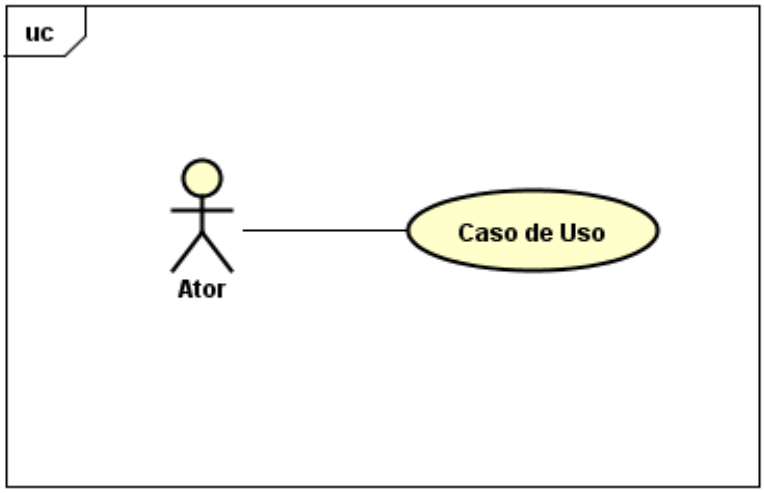
> “O modo para descrever os vários aspectos de modelagem pela UML é através da notação definida pelos seus vários tipos de diagramas. Um diagrama é uma apresentação gráfica de uma coleção de elementos de modelo, frequentemente mostrado como um gráfico conectado de arcos (relacionamentos) e vértices (outros elementos do modelo). ”

Cada tipo de diagrama captura uma perspectiva diferente e um determinado. Na documentação deste trabalho, foram utilizados os diagramas de caso de uso e de classe.

2.1.1.1 **Diagrama de Casos de Uso**

Exibe um conjunto de caso de uso e atores (um tipo especial de classe) e seus relacionamentos. Diagramas de caso de uso (Figura 1) abrangem a visão estática de casos de uso do sistema. Esses diagramas são importantes principalmente para a organização e a modelagem de comportamentos do sistema.

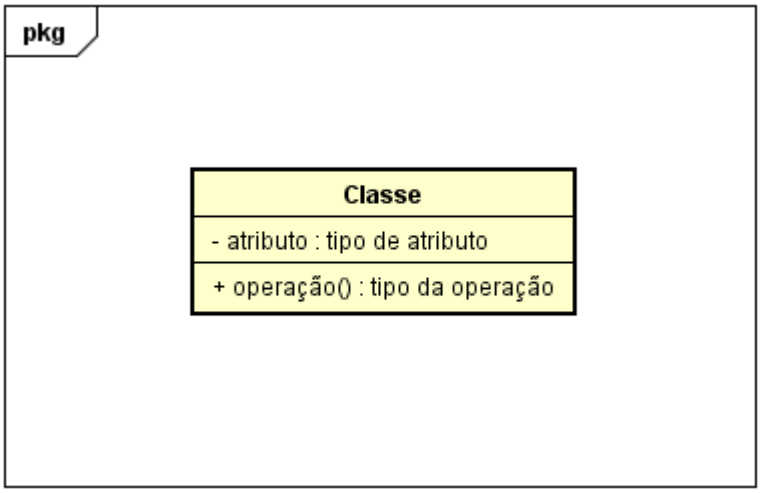
Figura 1 – Exemplo de um Diagrama de Caso de Uso



2.1.1.2 Diagrama de Classes

Exibe conjunto de classes, interfaces e colaborações, bem como seus relacionamentos. Esses diagramas são encontrados com maior frequência em sistemas de modelagem orientados a objeto e abrangem uma visão estática da estrutura do sistema. Os diagramas de classes (Figura 2) incluem classes ativas direcionam a perspectiva do processo estático do sistema

Figura 2 – Exemplo de Diagrama de Classe



2.1.2 Motores Gráficos

Um Motor Gráfico é um conjunto de bibliotecas utilizado para simplificar o desenvolvimento de um jogo ou outras aplicações gráficas, e evitar que a sua criação seja feita do zero. Normalmente utilizam modelagem de imagens 2D e 3D, além de trazer animações e sons padronizados.

OLIVEIRA (2013) fala que:

> "O objetivo de um motor de jogos é agrupar funções fundamentais para o desenvolvimento de jogos, que podem se estender da interação com os periféricos de entrada até a renderização dos cenários e personagens. Assim, várias aplicações podem ser desenvolvidas utilizando como base de código este componente central. Isto certamente reduz o tempo total de produção, à medida que concentra a equipe de trabalho em atividades de mais alto nível. Por mais genéricos que sejam, entretanto, os motores costumam ser projetados tendo em vista uma classe particular de jogos, como 2D ou 3D."

BERNARDO LIMA e RODRIGO MAIA (2006) citam a importância das engines gráficas no processo de desenvolvimento de um jogo:

> "As engines gráficas são essenciais para o desenvolvimento de qualquer jogo, pois elas são o núcleo do jogo, ou seja, a engine que irá determinar os gráficos e a física que irá ser utilizada no desenvolvimento de um jogo. Uma engine não é necessariamente um *software* gráfico, muitas engines são bibliotecas que são utilizadas em conjunto com algum ambiente de desenvolvimento integrado (IDE) ou ambiente para programação. "

2.1.3 Game Design

Toda a interação entre jogo e usuário é realizada pelo *game design*. Os elementos, as regras, o enredo e as dinâmicas de um jogo são aqui definidas. É por meio dela que ocorre na prática, tudo o que o *game* se proporciona a fazer.

ROUSE (2001, p. xix) diz que:

> "O *game design* determina quais escolhas o jogador será capaz de fazer no mundo do jogo e que ramificações estas escolhas terão no restante do jogo. O *game design* determina qual o critério de ganho ou perda o jogo deverá incluir, como o usuário será capaz de controlar o jogo e que informações o jogo comunicará a ele, e isto estabelece o quão difícil o jogo será. Resumidamente, o *game design* determina todos os detalhes de como a jogabilidade funcionará. "

BATES (2004, p. 36) informa um aspecto importante no processo de produção de um jogo:

> “O bom *design* em qualquer campo pode ser distinguido pela simplicidade. Um bom *design*er irá incluir somente aquelas coisas que são necessárias para criar o efeito que ele deseja. Qualquer coisa a mais é supérflua e deprecia o objetivo geral... enquanto o projeto está em desenvolvimento, ideias vão aparecer... como você decidirá o que colocar e o que deixar fora? Uma boa forma de nivelar é avaliar o esboço frente ao conceito. Se isso não o ajudar a alcançar o objetivo básico do jogo, cancele o projeto. Os melhores *game*s não são enormes e esparramados; são justos e objetivos. Eles não distraem o jogador com coisas sem relevância. ”

2.1.4 Linguagem C#

C#, é uma linguagem de programação desenvolvida pela Microsoft baseado em outras linguagens, como *Object Pascal* e *Java*. Possui sintaxe fácil e é atualmente uma das mais utilizadas no mundo. Esses foram os principais motivos para que a linguagem C# fosse escolhida para o desenvolvimento desse projeto.

Segundo a MICROSOFT (2015):

> “A sintaxe do C**#** é altamente expressiva, mas ela também é simples e fácil de aprender. A sintaxe do C# será instantaneamente reconhecida por qualquer pessoa familiarizada com C, C++ ou Java. Os desenvolvedores que sabem qualquer uma dessas linguagens são geralmente capazes de começar a trabalhar de forma produtiva com C# dentro de um tempo muito curto. A sintaxe do C# simplifica muitas das complexidades do C++ e fornece recursos poderosos, como tipos de valor nulo, enumerações, delegações, expressões lambda e acesso direto à memória, que não são encontrados no Java. ”

2.2 Ferramentas

Algumas ferramentas foram utilizadas para o desenvolvimento do *game*. Cada qual responsável por uma parte importante. Todas elas são descritas a seguir.

2.2.1 *Astah Professional*

Astah é uma ferramenta CASE (*Computer-Aided Software Engineering*) de criação de diagramas UML, assim como outros diagramas úteis para a fase de especificação de um sistema.

RIBEIRO (2012) afirma que:

> “O *Astah* é uma ferramenta que visa auxiliar o processo de modelagem de um sistema, é um editor de diagramas UML que incorpora outros recursos de acordo com a distribuição utilizada. É sucessora do JUDE (*Java and UML Developers Environment* –

> Ambiente para Desenvolvedores UML e Java), ferramenta que foi descontinuada em 2010. Assim como o JUDE, esta ferramenta possui versões *CommUnity* e Professional. ”

Desenvolvido em Java, com o *Astah* é possível realizar uma modelagem de dados complexa de maneira simples. Tem seu uso facilitado por possuir um layout intuitivo e trabalha com diversos tipos de diagramas: classes, casos de uso e desenvolvimento.

2.2.2 Unity 3D

Unity 3D é o motor gráfico (*game* engine) multi-plataforma, criado pela *Unity Technologies*, utilizado para o desenvolvimento deste trabalho. É possível escolher entre três linguagens de programação: a *javascript*, responsável por tornar os processos de páginas web mais dinâmicos, a linguagem escolhida para este projeto, o C#, e a sintaxe do *Python*, o *Boo,* que combina a clareza do código com a garantia de desempenho desejado. Assim como todo motor gráfico, o *Unity* é responsável por auxiliar a criação de jogos e aplicativos para diferentes propósitos e plataformas.

2.2.3 Adobe Photoshop

Para o desenvolvimento de toda a parte gráfica do jogo foi utilizado o *Adobe Photoshop*. É atualmente, uma das ferramentas de edição de imagens mais utilizada e conceituada do mundo.

De acordo com NELSON GEROMEL (2005)**:**

> “O *Photoshop* é um programa de edição de imagens que permite alteração cor, retoque e correção de imagens, aplicação de efeitos, inserir textos. Utilizando o *Photoshop*, você produzirá imagens e páginas para a *Web*, ajustando cores e criando efeitos especiais. Em sua versão 6, apresenta novas ferramentas para desenho e edição de formas e imagens. Aprimoramento na utilização das ferramentas possibilitando criar efeitos fantásticos como sombreamento, brilhos, relevos, transparência, deformação, etc. No *Photoshop* a prioridade é a produtividade, as paletas do programa podem ser facilmente agrupadas e escondidas, deixando a área de trabalho desimpedida, facilitando o trabalho do usuário na localização do recurso que necessita. ”

3 PROBLEMÁTICA

Nesta seção será abordado sobre o problema que se pretende resolver com o desenvolvimento do jogo educativo +Mundo.

3.1 Principais problemas ambientais que afetam o planeta.

A maioria dos problemas ambientais que o mundo enfrenta atualmente são causados pelo homem, sendo a sua maioria relacionados ao consumo e produção excessiva de resíduos.

Para BRASIL (2002)

> “O meio ambiente, infelizmente, possui dois contrastes gritantes hoje em dia: o ambiente humano e o natural, no qual, este último está perdendo espaço pelo crescimento populacional e a grande demanda de recursos naturais. O consumismo está degradando os recursos do planeta, esgotando-os e provocando graves e irreversíveis alterações. ”

DIAS (1998) completa que:

> “Nesse processo, a consequência é a gênese de um metabolismo que nas grandes cidades é intenso, em virtude do consumo elevado de matérias-primas, que são abastecidas pelo meio rural, de minerações, de recursos hídricos e a exploração dos recursos biológicos, tendo como reflexo o intenso fluxo de energia gerada pelo calor, eletricidade e combustíveis fósseis e com produção de grande quantidade de resíduos poluentes (fuligem, poeira, gases tóxicos, lixo etc.) que retornam ao ambiente causando desequilíbrios internos e externos.”

Grande parte do consumo humano concentra-se nos grandes centros urbanos. Muitas vezes o seu desenvolvimento não ocorre de maneira planejada, ocasionando problemas de grandeza ambiental.

CÓRDULA (2012) cita que:

> “Os centros urbanos consomem a maior parte dos recursos naturais extraídos do planeta, principalmente nos chamados países desenvolvidos e em desenvolvimento. Com esse tipo de desenvolvimento, estamos condenando nossa própria espécie a um fim trágico, pois o fim deste planeta talvez nunca ocorra, mas o da nossa própria espécie pode estar mais próximo do que os cientistas imaginaram. ”

3.2 Necessidade do +Mundo

Devido à ausência de responsabilidade humana na utilização dos recursos naturais, gerando como consequência a destruição do meio ambiente, fez-se necessária a criação do jogo educativo +Mundo.

Não é necessário realizar estudos muito profundos para perceber a quantidade de problemas ambientais presentes, como a baixa qualidade da água, transformação do clima por conta do efeito estufa e da redução da camada de ozônio, redução da biodiversidade, entre outros.

A defesa do meio ambiente realizada atualmente por vários órgãos importantes, além de uma pequena parcela por parte da população, é irrelevante diante da responsabilidade dessa geração. É importante que cada um, independente se criança ou adulto, assuma sua responsabilidade e passe a desempenhar uma postura mais sustentável.

Com as transformações da sociedade, foi preciso uma mudança no padrão de educação atual. O modelo onde o aluno é agente passivo da aprendizagem foi substituído pela educação dinâmica, proporcionando desafios aos alunos. Educar uma criança é difícil e requer persistência e dedicação. Algo que prende a atenção das crianças como um jogo, pode ser utilizado para acelerar e aperfeiçoar o processo de educar, formar conceitos éticos, de regras, respeito com próximo, além de tornar o aprendizado natural, prazeroso e dinâmico.

O +Mundo foi desenvolvido como uma alternativa para favorecer o aprendizado, buscando de forma desafiadora, motivar as crianças a buscarem por respostas sobre o tema ambiental proposto pelo jogo.

4 +MUNDO

O desenvolvimento do jogo educativo +Mundo dividiu-se em três fases: análise de requisitos, modelagem de dados e a construção do *game*. Neste capítulo serão abordadas todas essas fases.

4.1 Análise de Requisitos

A Análise de Requisitos é o ponto inicial para o desenvolvimento do jogo. É por meio dela que se tem uma maior noção do que se pretende construir. As funcionalidades e os objetivos do sistema ficam mais claros para quem o está desenvolvendo. É importante saber que os *software*s educativos diferem dos comuns, pois envolvem também requisitos do método de aprendizagem e uma análise correta é crucial para que o resultado final seja o esperado.

Nesta etapa foram definidos os usuários presentes no sistema e qual ação ele irá fazer. No diagrama de casos o usuário é representado por um Ator. O jogo possui dois atores: o jogador e o inimigo e as suas principais ações são definidas a seguir:

4.1.1 Jogador

- Informações – Com o avançar da fase algumas informações sobre sustentabilidade são mostardas ao jogador a fim de mantê-lo ciente da realidade atual do meio ambiente.
- Combater inimigos – Ao longo do percurso alguns inimigos aparecem para dificultar a sua passagem, e o jogador deve derrota-los para avançar no jogo. Os jogadores têm uma quantidade inicial de vida[2]. Sempre que o inimigo atingir seu objetivo sua vida é subtraída. Quando chegar a 0 (zero) o personagem (jogador) é destruído.
- Coletar Itens – Toda vez que o jogador eliminar um inimigo ele será contemplado com itens (mudas de plantas). Cada muda deve ser utilizada para fazer o replantio das árvores no decorrer do cenário, gerando pontos.
- Avançar de Fase – O principal objetivo do jogador é avançar de fase. Assim que todos os objetivos de uma fase são cumpridos uma nova fase é aberta.

[2] Indica o nível de vitalidade ou saúde do jogador.

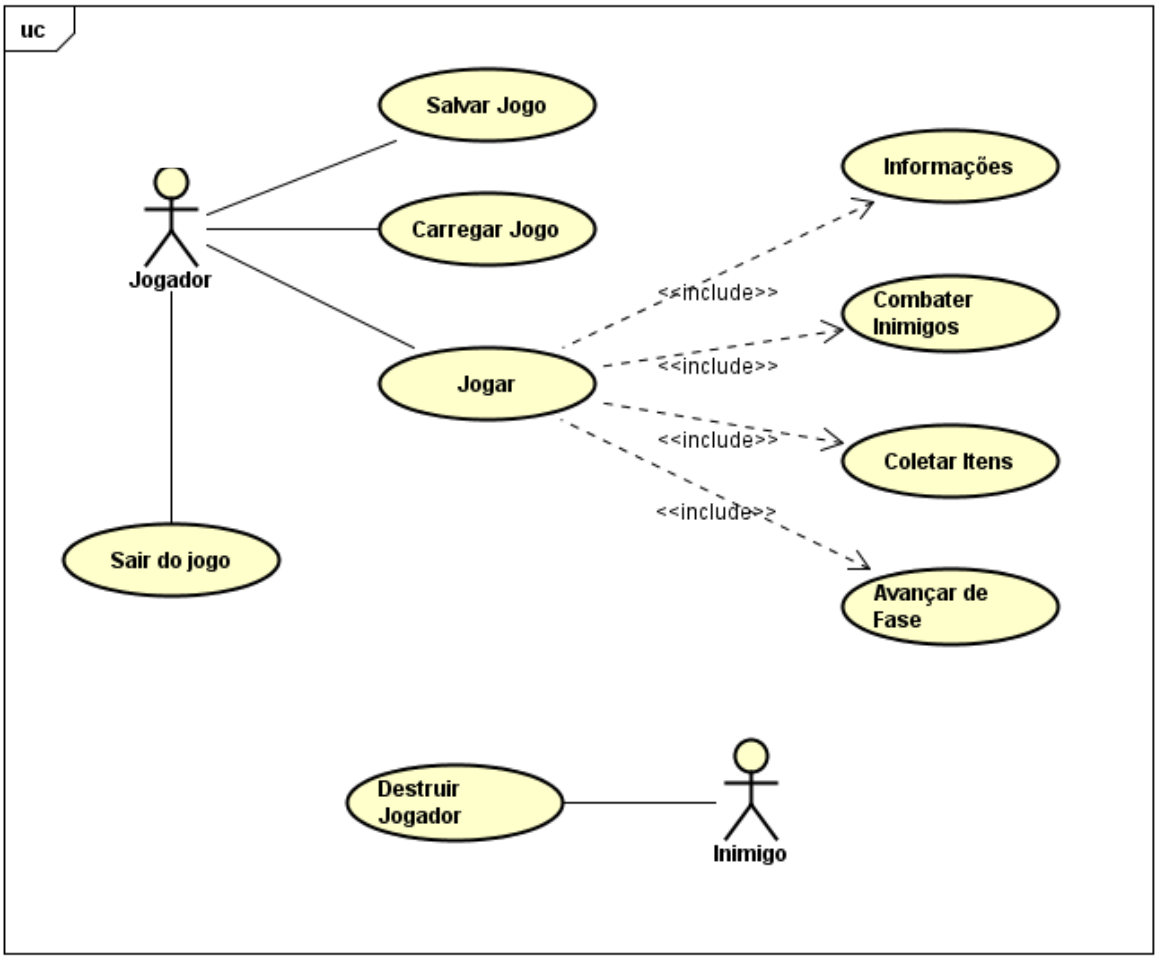
4.1.2 Inimigo

- Destruir o Jogador – A única ação do inimigo é tentar destruir o jogador para impedi-lo de avançar na fase. Cada golpe do inimigo causa dano na “vida” do jogador acabando por “destruí-lo” se realizado em sequência.

4.2 Diagrama de Caso de Uso do +Mundo

O Diagrama de classes (Figura 3) define a função de cada usuário dentro do sistema. Seu aspecto é simples e flexível.

Figura 3 – Diagrama de Caso de Uso do +Mundo

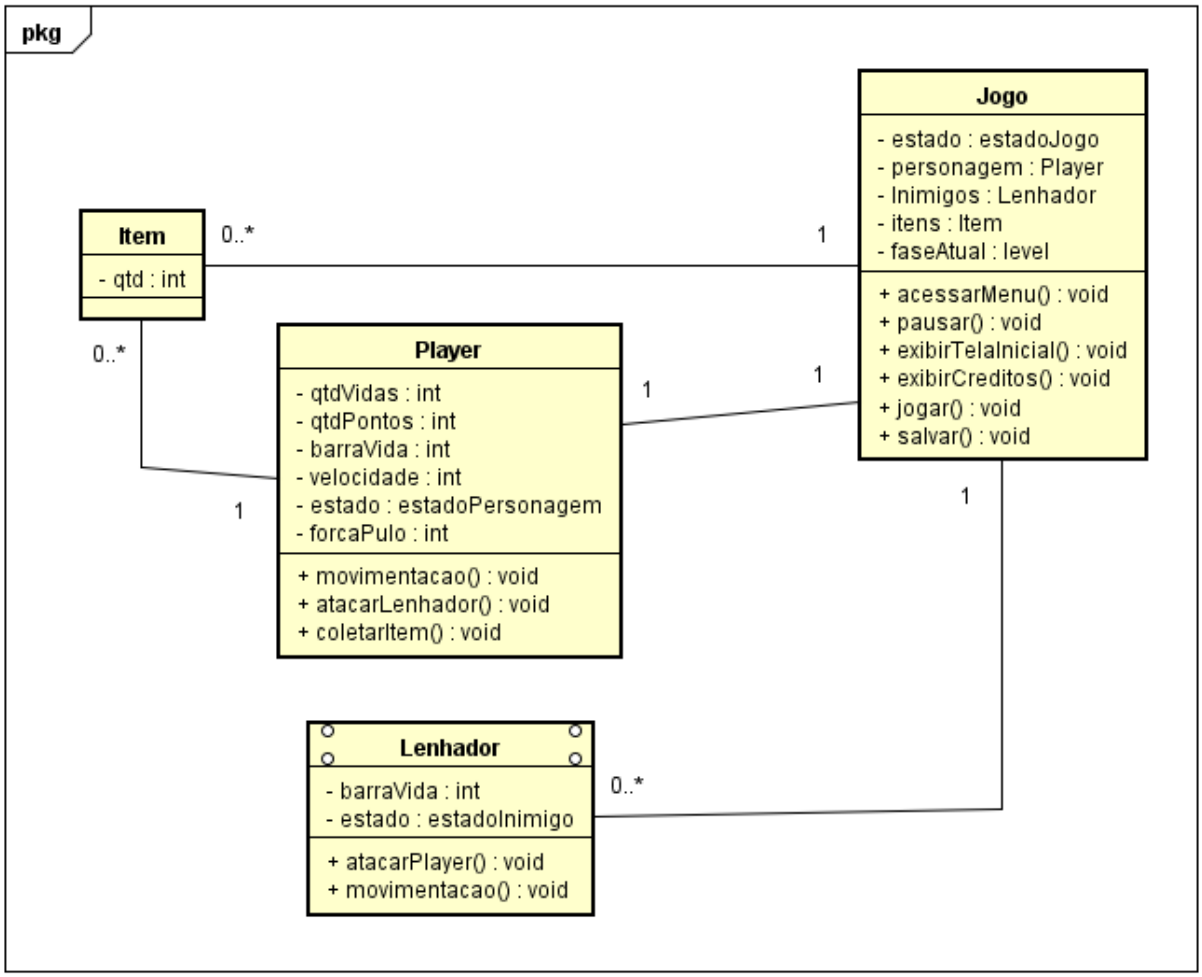


4.3 Diagrama de Classes do +Mundo

O Diagrama de Classe (Figura 4) define a representação da estrutura e relações das classes do sistema. O sistema possui 4 classes ao todo: as classes Player, Inimigo e Item possuem todos os seus respectivos dados (atributos). A

classe jogo é responsável por fazer todas as operações do jogo, como iniciar o jogo, pausar e salvar, além da relação e gerenciamento de todas as demais classes.

Figura 4 – Diagrama de Classes do +Mundo



4.4 Construção do *Game*

Esta seção tem por objetivo relatar detalhadamente o processo de desenvolvimento do jogo.

4.4.1 Interface Gráfica

A Interface Gráfica é responsável pela interação entre o usuário e o jogo por meio de uma representação gráfica. É importante que ela torne o uso do *software* mais fácil para que passe despercebida pelo usuário e assim o foco principal fique para o desenrolar da história do *game*. As posições dos botões, os menus e os itens presentes foram aqui definidas (Figura 5).

Figura 5 – Menu do Jogo

4.4.2 Modelagem do *Game*

Qualquer objeto criado provido de forma geométrica para compor o *game* é chamado de Modelagem do *Game*. É dividida em duas partes: a modelagem estrutural engloba toda a criação dos objetos estáticos, como o cenário, e a modelagem dinâmica os objetos que possuem movimentação. O personagem e os inimigos presentes no *game* são exemplos de modelagem dinâmica.

No cenário estão presentes todos os objetos que tem como finalidade caracterizar o ambiente do jogo. É nele que todos os elementos físicos que definem espaço estão presentes, bem como as suas cores, os estilos e as texturas. Na maioria das vezes o desenvolvimento de um jogo inicia-se pela criação do cenário. O personagem é o ser atuante do *game*. Ele tem como objetivo explorar o cenário.

4.4.3 Funcionalidades do *Unity* Utilizadas no Projeto

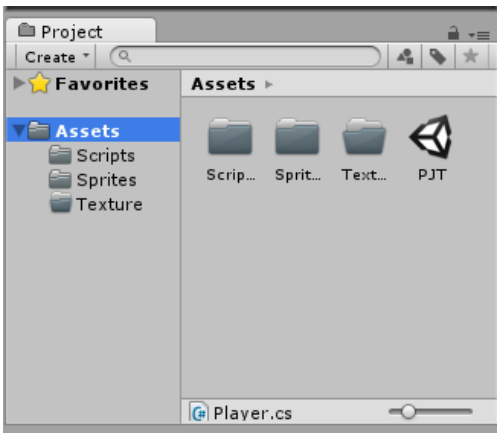
Alguns funcionalidades presentes no *Unity* foram utilizadas na criação e organização do projeto. Todas elas são mostradas a seguir.

4.4.3.1 Project

O *Project* (Figura 6) reune todos os elementos que estão presentes no projeto do jogo, como *scripts*, *sprites* e texturas. É essencial que exista organização na

estrutura de todos esses elementos. Toda base para edição e criação dos elementos do *game* está presente na pasta *Assets.*

Figura 6 – *Project*



4.4.3.2 Scripts

Script é um texto (código) com uma série de instruções que devem ser seguidas. Toda a parte dinâmica do *game*, como movimentação dos personagens, definição de fim de jogo, metas e objetivos foram feitas por meio dos *scripts*. A ferramenta utilizada pelo *Unity* para o desenvolvimento dos *scripts* é a IDE *Mono Develop.*

Toda a movimentação do personagem, as teclas de ação, altura do pulo e a sua velocidade foi definida por meio do script "*Player*" (Figura 7).

Figura 7 – *Script Player* referentes a movimentação

```
22
23  void Movimentacao() {
24
25      animator.SetFloat("movimento", Mathf.Abs (Input.GetAxisRaw ("Horizontal")));
26      estaNoChao = Physics2D.Linecast(transform.position, chaoVerificador.position, 1 << LayerMask.NameToLayer("Piso"));
27      animator.SetBool ("chao", estaNoChao);
28
29      if (Input.GetAxisRaw("Horizontal") > 0 ) {
30          transform.Translate (Vector2.right * velocidade * Time.deltaTime);
31          transform.eulerAngles = new Vector2(0, 0);
32      }
33
34      if (Input.GetAxisRaw("Horizontal") < 0 ) {
35          transform.Translate (Vector2.right * velocidade * Time.deltaTime);
36          transform.eulerAngles = new Vector2(0, 180);
37      }
38
39      if (Input.GetButtonDown("Jump") && estaNoChao) {
40          GetComponent<Rigidbody2D>().AddForce(transform.up * forcaPulo);
41      }
42  }
```

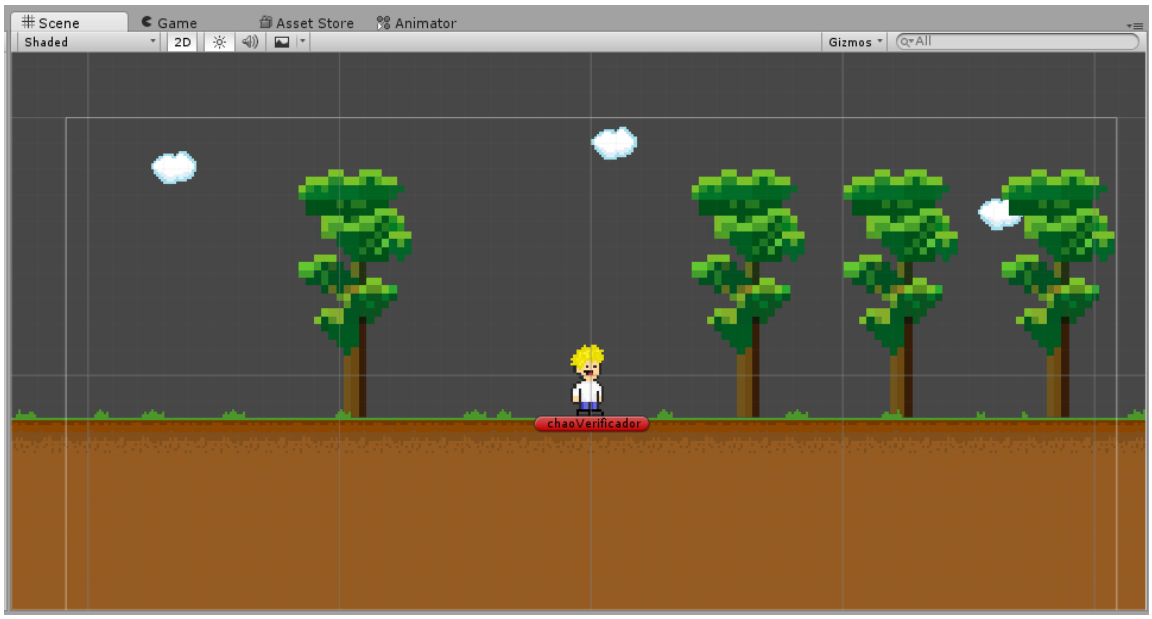
4.4.3.3 Sprites

Sprites são objetos criados com ajuda de alguma ferramenta gráfica para compor o *game* de forma visual, como uma árvore do cenário, o chão ou até mesmo o próprio personagem. Todos os *sprites* criados para o *game* foram desenvolvidos em 32 bits. Ver Figura 8.

4.4.3.4 Scenes

As *Scenes* (Figura 8) são as divisões do jogo. Toda a parte visual do jogo é montado nela. Cada fase (nível) do *game* faz parte de uma scene, assim como todas as telas de início e fim, além do menu principal. Logo após o jogador comcluir a primeira fase, uma nova *Scenes* é carregada.

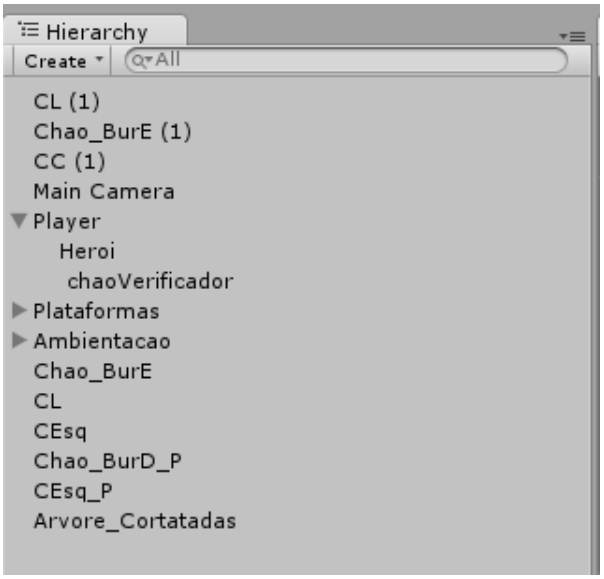
Figura 8 – *Scene* da primeira fase do jogo



4.4.3.5 Game Objects

Os *Game Objects* (Figura 9) são quase todos os objetos presentes em uma scene e são a unidade fundamental dentro delas. Eles podem ser rotacionados, movidos, duplicados, entre outras edições. *Sprites*, a câmera e as luzes são todos *Game Objects*. Cada objeto obdece a uma hierarquia de objetos.

Figura 9 – *Game Objects* presentes no jogo

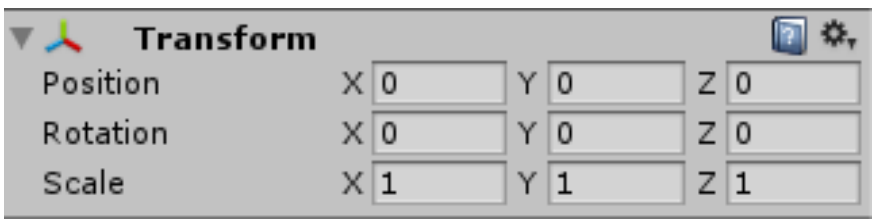


Se uma pasta for modificada todas as suas subpastas recebem a mesma alteração. No *game* por exemplo, a pasta *Player* e as suas subpastas Heroi e chaoVerificador, representados na Figura 9, são organizadas dessa forma por necessitarem das mesma alterações no projeto.

O chaoVerificador tem a função de verificar se o *Player* está ou não sobre o solo, não permitindo assim que o comando pulo seja executado quando ele estiver no ar. Ele é importante para que o personagem não pule infitamente para cima. Este *Game Object* deve acompanhar o *Player* e por esse motivo devem possuir as mesmas configurações.

Todos os objetos são organizados por meio de coordenadas (‘x’, ‘y’ e ‘z’) e podem ser alteradas dentro do *Unity* pela aba *Tranform* (Figura 10), assim como a sua rotação e a escala (tamanho do objeto).

Figura 10 – Aba *Transform*

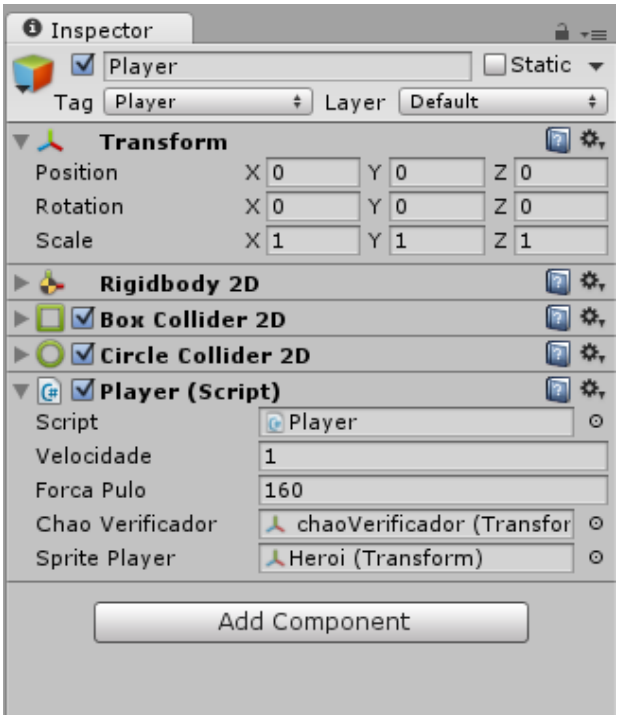


4.4.3.6 Components

Os *Components* (Figura 11) adicionam funcionabilidades e parâmetros a um *game object*. É possivel definir que o objeto adquira caracteríscas físicas, como gravidade, colisores, emissão de luz, texturas, efeitos sonoros, entre outros. Todos os componentes de um objetos estão presentes na aba *Inspector* do *Unity*.

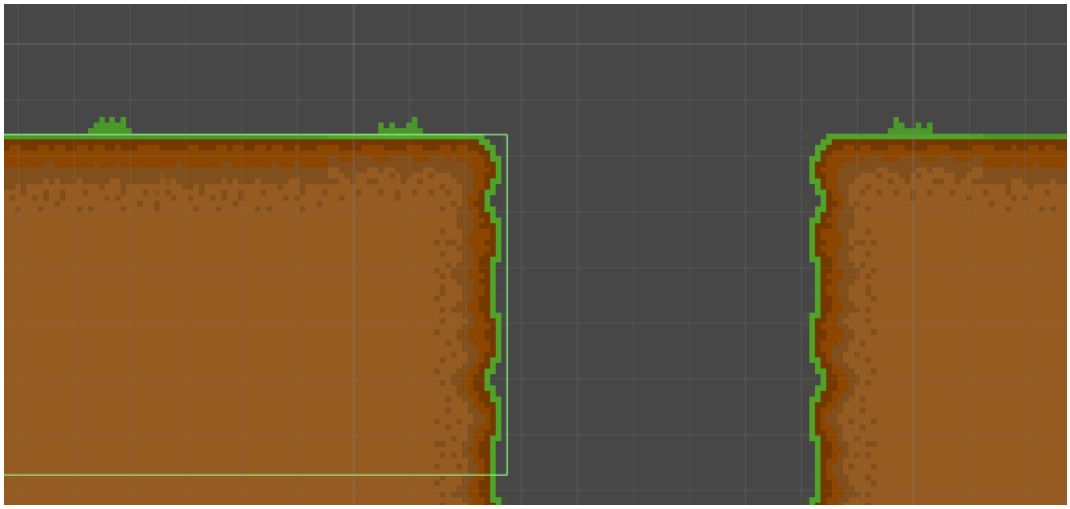
No *Game Object* "*Player*", os componentes responsáveis por todas as caracteriscas físicas presentes no jogador são o *Rigidbody 2D*, o *Box Collider 2D* e o *Circle Collider 2D.* O *Rigidbody 2D* adiciona o efeito da gravidade ao objeto, para que por exemplo, ele caia depois de um pulo. Os colisores *Box Collider 2D e Circle Collider 2D* fazem com que se possa esbarrar em obstáculos, evitando assim que o jogador cruze paredes ou componentes físicos. No objeto *Player* também está presente o código respónsavel pela movimentação do personagem (*Script Player*).

Figura 11 – *Components* de um *Game Object*



Todo objeto que não pode ser atravessado faz-se necessário a utilização de um colisor, como por exemplo o chão, e preenche toda a área que deve permanecer fisica. Ver Figura 12.

Figura 12 – *Box Collider 2D*



No *Player* foram utilizados os dois colisores. Conforme mostra a Figura 13, o componente *Circle Collider 2D*, em formato círcular, foi utilizado para os pés do personagem por possuir menor atrito que o *Box Collider*, evitando-se qualquer que seja a interferencia na movimentação do *Player.*

Figura 13 – Colisores adicionados ao *Player*

4.4.4 Animação

Uma animação (Figura 14) é representada por uma sequência de *Sprites* em alta velocidade e é gerenciada no *Unity* pela aba *Animation* (Figura 15). A velocidade da transição entre um *Sprite* e outro é definida pela alteração do *sample*. Quanto maior seu número mais rápido é a transição, podendo ser feita em *loop* ou não. Um padrão usado nesse projeto foi a velocidade 4 por ser a que mais se adequa a realidade do jogo.

Figura 14 – Animação do *Player*

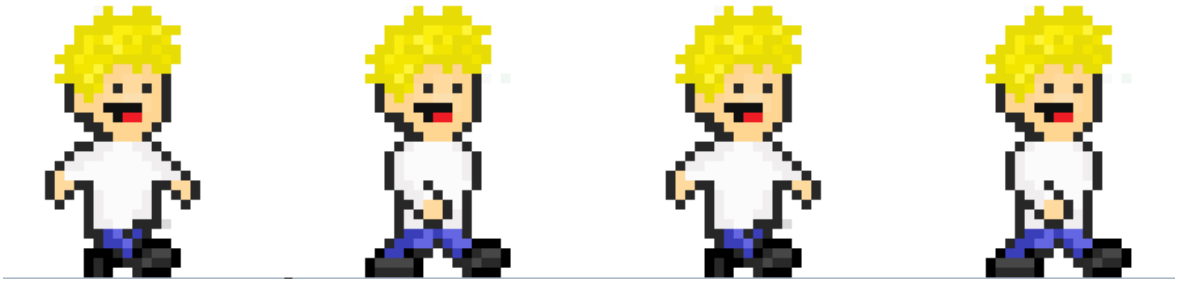
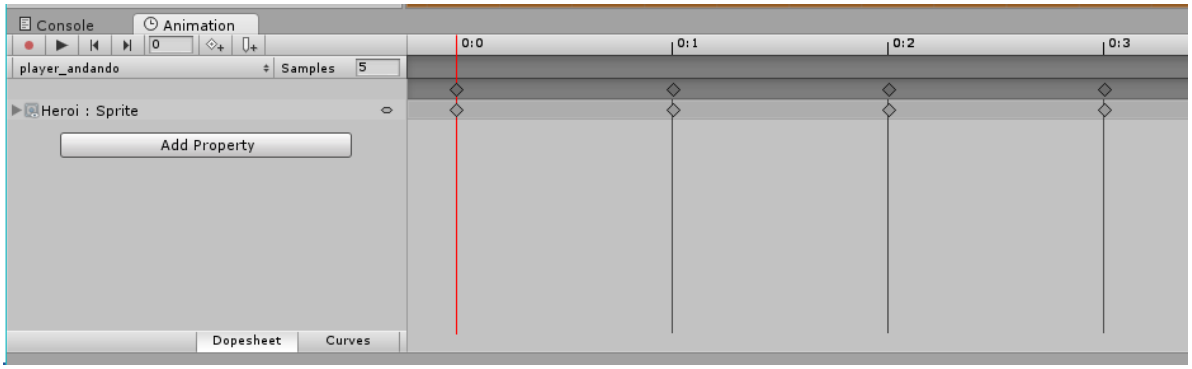
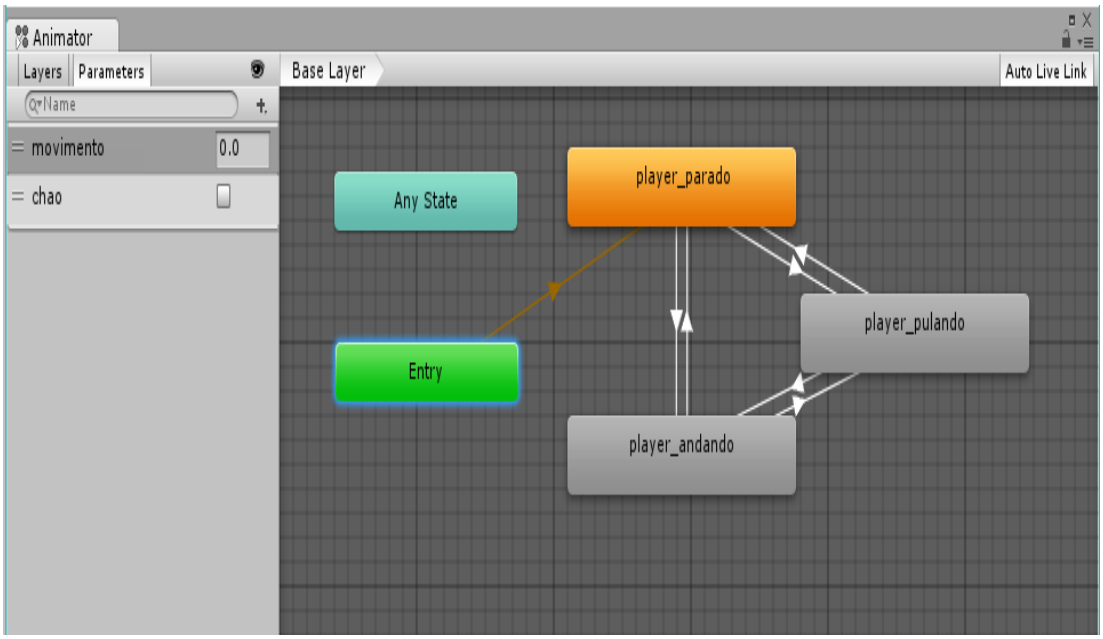


Figura 15 – Aba *Animation*



Quando uma animação deve ser continuidade de outra devido uma mudança de estado, sua configuração é feita na aba *Animator* (Figura 16). Foram criadas três animações. Uma para cada estado: Player_parado, Player_andando e Player_pulando. Quando o personagem fica parado é reproduzida uma animação. Depois que o usuário fizer um comando para o *Player* andar a animação vai ser alterada para condizer com esse estado. Da mesma forma foi feita a animação Player_pulando.

Figura 16 – Aba *Animator*



Animações onde não existem mudanças devido condições, como o efeito da árvore balançando com o vento, são reproduzidas em *loop* constante.

4.4.5 Inteligência artificial

Em um *game*, o elemento inimigo (Figura 17) é utilizado para ampliar o grau de dificuldade para o jogador. Um inimigo deve ser controlado automaticamente e comparta-se de acordo com o seu objetivo no jogo. Esse é o papel realizado pela IA (Inteligência Artificial).

Figura 17 – *Sprite* do inimigo Lenhador

O desenvolvimento de um inimigo é semelhante ao processo de criação do *Player*. A única diferença é a necessidade de programar para que ele execute uma ação automaticamente e não por meio de um comando do jogador feito pelo teclado.

Figura 18 – Sprite do ataque realizado pelo inimigo

No jogo foi criado o inimigo lenhador. Caso o Personagem se aproximar muito ele irá ataca-lo para evitar que avance a fase. A Figura 18 mostra a animação do inimigo atacando. O *Script* Lenhador (Figura 19) é responsável por fazer todo o controle do inimigo, além de definir todos os seus atributos, como a direção que ele deve andar, a distância que ele deve percorrer e a quantidade de vida que ele retira do *Player* a cada golpe.

Figura 19 – *Script* de movimentação do Lenhador

```
39 void Update () {
40
41     if (direcao) {z
42         transform.eulerAngles = new Vector2(0, 0);
43     } else {
44         transform.eulerAngles = new Vector2(0, 180);
45     }
46     transform.Translate(Vector2.right * velocidade * Time.deltaTime);
47
48     tempoNaDirecao += Time.deltaTime;
49     if (tempoNaDirecao >= duracaoDirecao) {
50         tempoNaDirecao = 0;
51         direcao = !direcao;
```

Para o inimigo efetuar seu ataque, o *Player* deve colidir com ele. Cada ataque bem sucedido causa 30 de dano na vida do *Player*: Ver Figura 20.

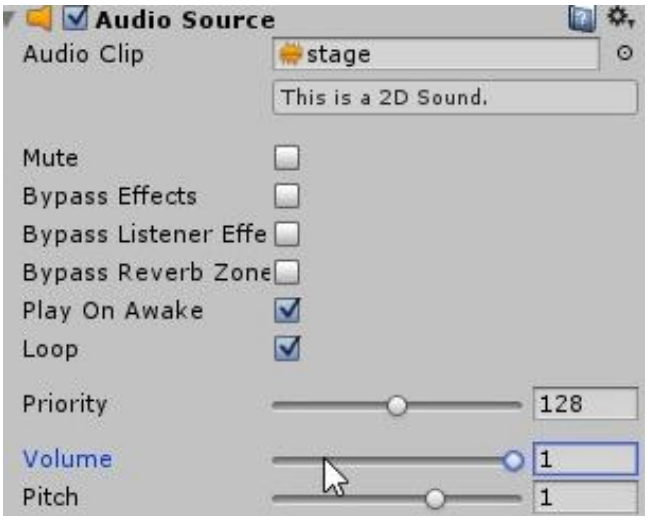
Figura 20 – *Script* de ataque do Lenhador

```
38 void OnCollisionEnter2D(Collision2D colisor) {
39     if (colisor.gameObject.tag == "Player") {
40         animator.SetTrigger("atacou");
41         var player = colisor.gameObject.transform.GetComponent<Player>();
42 
43         player.PerdeVida(30);
44         colisor.gameObject.transform.Translate(-Vector2.right);
45     }
46 }
```

4.4.6 Efeitos Sonoros

O *component* utilizado para o controle sonoro do *game* é o *Audio Source* (Figura 21). Qualquer música pode ser importada para ele e adicionada ao *game*

.Figura 21 – Aba *Audio Source*



Outro *component* importante na criação do efeito sonoro de um jogo, presente no objeto "*Main* Câmera[3]", é o *Audio Listerner*. Ele é o "ouvido'' do *Player* no *game*, sendo responsável por emitir toda fonte de som que está perto do personagem ou emitida por ele próprio.

[3] Toda a visão que o usuário vai ter quando estiver jogando. É a câmera do jogo.

4.4.7 Definições de Final de Jogo

Apenas quando o personagem perder todas as vidas que possui é definido o final de jogo. O *Player* só pode perder uma vida por dois motivos: sua vida esgota por sofrer danos do inimigo ou sofre queda em um dos precipícios presentes pelo caminho.

O código para identificar a morte por queda é simples. Um *Box Collider 2D* é usado para preencher toda a cena do *game* e caso o *Player* caia em um buraco ele vai ultrapassar o colisor, voltando assim para o inicio da fase. Ver Figura 22

Figura 22 – *Script* de delimitação da *Scene*

```
23      void OnTriggerExit2D(Collider2D other){
24          if (other.gameObject.tag == "Player"){
25              Debug.Log ("Player Left Scene");
26          Player.tranform.position = positionStart;
27          }
28      }
29
30  }
```

Para redução da vida por dano inimigo foi criada uma variavel que contem a qunatidade da vida atual do *Player* (100). O dano causado ao personagem é definido no *script* do inimigo. A cada ataque bem sucessedido é retirado a quantia correspondente ao dano causado pelo inimigo. Assim que chegar a 0 (zero) o *Player* retorna ao inicio da fase com uma vida a menos. Conforme Figura 23.

Figura 23 – *Script* da vida do *Player*

```
23      public void PerdeVida(int dano) {
24          vidaAtual -= dano;
25
26          if (vidaAtual <= 0) {
27              Application.LoadLevel(Application.loadedLevel);
28          }
29
30          if ((vidaAtual * 100 / maxVida) < 30) {
31              vida.guiText.color = Color.red;
32          }
33
34          vida.guiText.text = "HP: " + vidaAtual + "/" + maxVida;
35      }
```

5 CONSIDERAÇÕES FINAIS

O meio ambiente atual sofre com o consumismo exagerado humano e pouco é feito por parte da população, principalmente de forma individual, para melhorar a situação. Na maioria das vezes as pessoas não levam uma vida sustentável por não sentirem a necessidade de mudar atitudes para se adequar a ela.

Este projeto visa melhorar a educação ambiental das crianças por meio de um jogo educacional, para assim em um futuro próximo todos tenham uma melhor consciência do certo a se fazer com relação ao nosso meio ambiente. O *game* possui jogabilidade fácil e intuitiva, para que assim as crianças não apresentem dificuldades ao jogá-lo. Todo o ambiente do *game* é criado para tornar o objetivo principal do personagem, que é salvar o meio ambiente, também o objetivo de vida para quem o joga.

Para trabalhos futuros, pretende-se realizar a ampliação do *game*, com a adição de novas fases, novos inimigos e consequentemente, mais desafios para o jogador. Por ser o principal foco do *game*, o papel educativo do jogo será mais bem trabalhado a fim de torna-lo mais conciso e exposto da melhor maneira possível.

Pretende-se realizar também a ampliação do *game* para outras plataformas, para que sua proposta seja repassada para uma maior quantidade de pessoas. Por ser uma tendência cada vez maior a cada ano, por consequência da sua praticidade, a plataforma mobile irá ganhar atenção especial.

6 REFERÊNCIAS

ABRA *GAME*S. **A indústria brasileira de jogos eletrônicos um mapeamento do crescimento do setor nos últimos 4 anos**, 2008.

BRASIL. **Meio ambiente e consumismo.** Inmetro. Disponível em: http://www.saeb.ba.gov.br/vs-arquivos/HtmlEditor/file/compraspublicas/novo/cartilha_meio_ambiente_inmetro.pdf. Brasília, 2002.

CAPRA, Fritjof. **A teia da vida: uma nova compreensão científica dos sistemas vivos**. São Paulo: Cultrix, 2006.

CÓRDULA, Eduardo Beltrão de Lucena et al. Poluição: Eca! In: GUERRA, R. A. T. (org.). **Educação Ambiental: textos de apoio.** João Pessoa: Editora Universitária da UFPB, 1999.

DIAS, Genebaldo F. **Educação Ambiental: princípios e práticas**. 5ª ed. São Paulo: Gaia, 1998.

FURLAN, J. D. **Modelagem de Objetos Através da UML: The Unified Modeling Language**. 1ª ed., Makron Books, 1998

GEROMEL, Nelson. **Apostila de Photoshop**, 2005.

LACERDA R.A. **Proposta de um modelo para análise de requisitos de *software* educativo**; Tese de mestrado, Faculdade de Educação - UnB, 2007.

MICROSOFT. **C# Language Fundamentals em Learning C# 3.0: Master the fundamentals of C# 3.0**, 2015.

OLIVEIRA, E. R. **O Uso de Engines para o Desenvolvimento de Jogos Eletrônicos. Monografia de conclusão de curso apresentada à Universidade Estadual do Sudoeste da Bahia – UESB**. Vitória da Conquista, 2013.

OMG. **Unified Modeling Language: infrastructure**. 2006

RIBEIRO, L. F. **Modelagem de *software* utilizando UML: Análise comparativa entre as ferramentas Astah UML e Umbrello UML Modeller.** Vila Velha. 2012.

ROUSE, Richard. ***Game Design* – Theory & Pratice. Texas: WordwarePublishing**, Inc, 2001.

SIRVINSKAS, Luis Paulo. **Manual de direito ambiental/** 2 ed.rev. atual e ampl, São Paulo: Saraiva, 2003.

TONSIG, Sergio Luiz. **Engenharia de software: analise e projeto de sistema.** ci~encia Moderna, 2003.

WAZLAWICK, R.. **Análise e Projeto de Sistemas de Informação Orientados a Objetos**. Elsevier, 2004.

TERMO DE AUTORIZAÇÃO PARA PUBLICAÇÃO DIGITAL NA BIBLIOTECA "JOSÉ ALBANO DE MACEDO"

Identificação do Tipo de Documento

( ) Tese

( ) Dissertação

(X) Monografia

( ) Artigo

Eu, Juliano Vieira Rocha,

autorizo com base na Lei Federal nº 9.610 de 19 de Fevereiro de 1998 e na Lei nº 10.973 de 02 de dezembro de 2004, a biblioteca da Universidade Federal do Piauí a divulgar, gratuitamente, sem ressarcimento de direitos autorais, o texto integral da publicação + Mundo: um jogo educativo sobre desenvolvimento sustentável

de minha autoria, em formato PDF, para fins de leitura e/ou impressão, pela internet a título de divulgação da produção científica gerada pela Universidade.

Picos-PI 04 de Março de 2016.

Juliano Vieira Rocha

Assinatura