Comparativo entre Game Engines como Etapa Inicial para o Desenvolvimento de um Jogo de Educação Financeira

Carlos Henrique Leitão Cavalcante[1], Maria Luciana Almeida Pereira [1]

[1]Instituto Federal de Educação Ciência e Tecnologia do Ceará (IFCE) Rodovia BR 020, Km 303, s/n - Jubaia, 62700-000 – Canindé – CE – Brasil

{henriqueleitao, lucianaalmeidaaa}@gmail.com

Abstract. *The teaching of financial management is not present in basic education in Brazil or when it is addressed it may not arouse students' interest. Teaching this theme in a playful way through a children's mobile game, it is an interesting alternative. However, to develop this game without great programming knowledge is necessary a game engine that facilitates its implementation. In this way, this work aims to present a comparison between some engines for game development, serving as input for choosing an engine. In the end, Godot presented itself as the best game engine, presenting greater facilities for game development.*

Resumo. *O ensino da gestão financeira não está presente na educação fundamental no Brasil ou quando é abordado pode não despertar o interesse dos alunos. Ensinar de forma lúdica essa temática através de um jogo de celular para crianças, mostra-se uma alternativa interessante. Contudo, para desenvolver este jogo sem grandes conhecimentos de programação é necessário uma game engine que facilita sua implementação. Desta forma, este trabalho tem como objetivo apresentar um comparativo entre algumas engines para desenvolvimento de jogos, servindo de insumos para escolha de uma engine. Ao final, o Godot se apresentou como melhor motor de jogo, apresentando maiores facilidades para desenvolvimento de jogos.*

1. Introdução

Segundo a *Global Financial Literacy Excellence Center*, uma pesquisa realizada em 2014 com 150 mil pessoas em 140 países objetivava medir o grau de alfabetização financeira e obteve como resultado, que a cada três pessoas um é considerado educado financeiramente. Relacionado ao Brasil, a pesquisa mostrou que apenas 35% dos brasileiros presentes podem ser considerados alfabetizados financeiramente. Com a educação financeira a população será capaz de tomar decisões relacionadas a poupança, investimentos, etc.

Segundo Papert (2008), a gamificação corresponde ao uso de mecanismos para motivar a ação de resolver problemas e aumentar a aprendizagem nas mais variadas áreas de conhecimento. Os jogos são um bom exemplo, pois apresentam elementos como regras, competição, recompensa, níveis de dificuldade e etc.. Entretanto, para se desenvolver um jogo sem conhecimentos de programação, é necessário de um *game engine*. Lewis e Jacobson (2002 *apud* Assis *et al*. 2006, p. 4) define uma *game engine* (motor do jogo) ou somente *engine*, como um conjunto de bibliotecas que são criadas com o intuito de facilitar o desenvolvimento de jogos, ou seja, diminui a necessidade de

programação, abstraindo várias etapas do desenvolvimento através de editores e ferramentas gráficas.

Dessa forma, pensar em disseminar os conceitos de educação financeira, para o público infantil, torna-se relevante para se contornar o alto índice de analfabetismo financeiro. Contudo, ensinar conceitos sobre finanças para crianças se torna mais fácil quando se transmite esses ensinamentos de forma lúdica, através de brincadeiras ou da gamificação. Em uma pesquisa realizada com 22 crianças de 8 a 13 anos de idade nas Escola Municipal Francisco Delfino Gomes e Escola Municipal Frei Luciano, na comunidade de Ipueira dos Gomes, localizada na cidade de Canindé - CE, obteve-se como resultado que 91% dos alunos entrevistados consideraram interessante a ideia de se desenvolver um jogo com esta finalidade e os mesmos tem interesse em jogá-lo.

Nesse sentido, o objetivo desta pesquisa é escolher uma *game engine* que será utilizada para o desenvolvimento de um jogo voltado para alfabetização financeira que possibilite a criação do jogo sem conhecimento especializado em programação. Para isto, foi necessário realizar o *benchmarking* (comparativo) entre as *game engines* Unity, Godot e Phaser, apresentando suas principais características e os pontos fortes e fracos de cada uma. Através dela será possível conhecer as características de cada uma das *engines*, fazendo, assim, uma etapa essencial para o sucesso do projeto que é a criação final do jogo educativo em gestão financeira para criança.

2. Material e Métodos

Esta pesquisa se caracteriza como um estudo de caso (Yin, 2013), onde teve o objetivo de realizar um comparativo que visa determinar uma ferramenta para o desenvolvimento de um jogo em 2D. As *engines* escolhidas para serem estudadas foram o Unity, Godot e Phaser. A escolha se deu por possuírem uma boa documentação com exemplos práticos e serem focadas no desenvolvimento de jogos em 2D. Cada um destes motores irá passar por três etapas de avaliação e estão descritas na Figura 1.

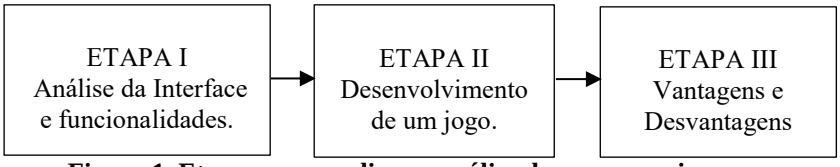

Figura 1. Etapas para realizar a análise das *game engines*

A primeira etapa foi responsável pela análise minuciosa das IDE (*Integrated Development Environment* ou Ambiente de Desenvolvimento Integrado) de cada uma das *engines*, onde foi realizado um estudo e coletadas suas principais características. Na segunda etapa, desenvolveu-se um jogo exemplo, unindo várias funcionalidades de cada *engine*, fornecendo insumos para a comparação realizada na última etapa. A última etapa consistiu em apresentar os pontos fortes e fracos de cada *engine* na visão do autor deste trabalho.

3. Resultados e Discussões

3.1 Análise da Interface e Funcionalidades

A IDE da Unity possui algumas guias de navegação que podem ser reposicionadas de acordo com que o programador achar melhor. Cada aba possui a sua função dentro do

desenvolvimento de jogos em 2D e 3D. Nesta plataforma existe também o conceito de *GameObject* que se caracteriza como um *container* genérico, ou seja, pode tomar a forma de qualquer coisa. Ele possui a *Asset Store* que possibilita realizar o *download* de elementos gráficos, o *Animation* para organizar as animações e o *Animator* para definir o fluxo das animações. A linguagem utilizada para o desenvolvimento de jogos é C# ou JavaScript e possui integrado o editor MonoDevelop. Na figura 2, é apresentado a área de trabalho da *engine* Unity.

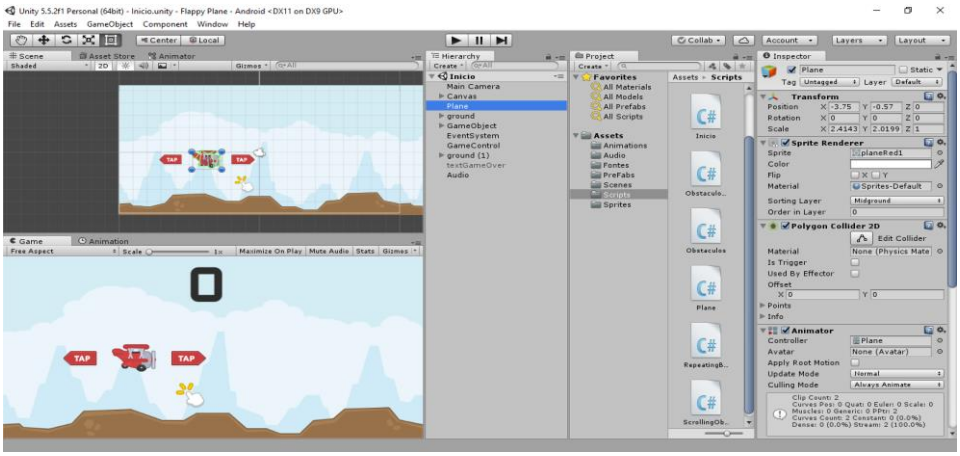

Figura 2. Apresenta a área de trabalho do Unity no desenvolvimento de um jogo

Como apresentado na Figura 2, na guia "*Game",* é possível visualizar como a câmera registra o que está acontecendo no jogo e como o jogo será executado nos dispositivos. Na guia "*Inspector",* é possível encontrar todas as possíveis configurações de objeto presente no jogo. Os objetos são armazenados hierarquicamente dentro da guia "*Hierarchy"*. A aba *Project* representa a pasta do computador no qual o jogo está salvo, uma característica marcante desta plataforma é que é possível apenas jogar os *assets* dentro desta pasta na própria plataforma, sem a necessidade de ir até o local onde está salvo, como é o caso das demais *engines* apresentadas neste projeto.

O motor Godot apresenta uma interface intuitiva com muitas ferramentas para desenvolvimento de jogos em 2D e 3D. Ele trabalha com um sistema de cenas, o onde cada cena é um objeto ou possui vários objetos. É possível utilizar o conceito de herança, através do sistema de nós, no qual cada cena deve possuir um nó que irá identifica-la hierarquicamente. Nessa *engine*, os objetos podem emitir sinais em forma de funções para outros objetos, estes sinais são utilizados, por exemplo, para detectar se um objeto colidiu com outro abjeto. Ela possui um sistema de animação integrado a plataforma onde é possível animar tudo até mesmo funções, baseando-se em quadros. Os jogos são desenvolvidos na linguagem GDScript baseada em Phyton. Na figura 3, apresenta a área de trabalho da Godot.

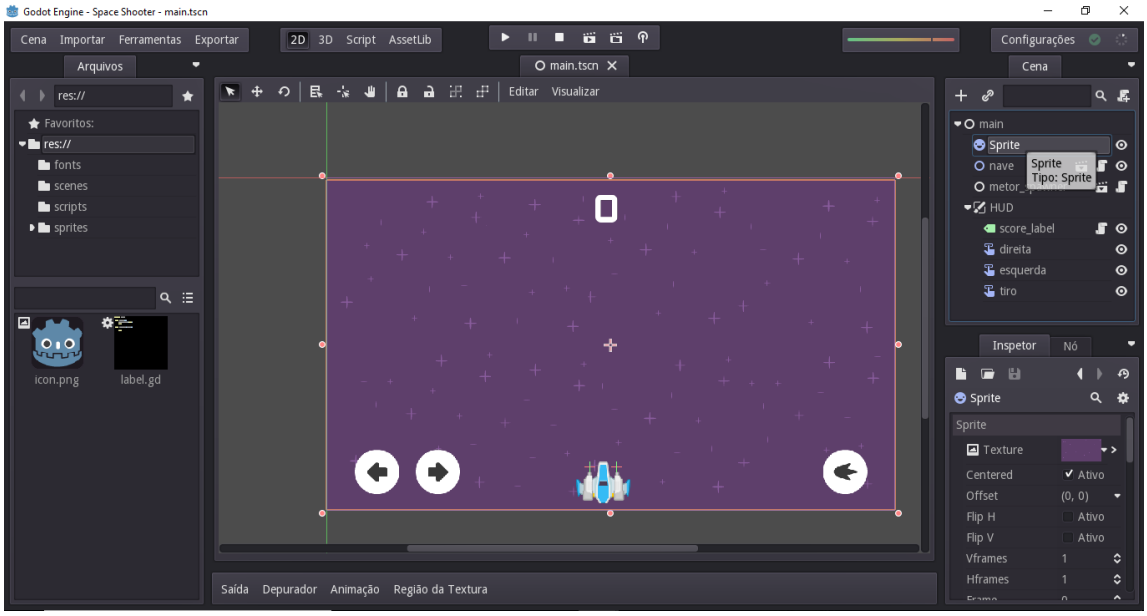

Figura 3. Apresenta a área de trabalho da Godot no desenvolvimento de um jogo

Como apresentado na figura 3, a interface da Godot é semelhante à da Unity, no entanto, existem algumas divergências, por exemplo, não existe uma guia específica para se criar animações, para isto é necessário adicionar um nó responsável por esta função. É possível adicionar os nós na guia "Cena". Contudo, as guias Arquivos, *Inspector* são semelhantes as guias *Project* e *Inspector* do Unity, mas, não é possível adicionar arquivos.

O Phaser é um motor de jogo totalmente voltado para programação em JavaScript ou TypeScript e inicialmente não possuía uma interface que facilitasse a programação. Contudo, a pouco tempo foi criada a IDE Phaser Editor, a qual não foi contemplada nesse estudo devido a limitações de tempo. Neste projeto, foi utilizado somente a biblioteca do *framework* Phaser. Esse *framework* tem como suas principais características o suporte a WebGL e Canvas, possibilitando o desenvolvimento de jogos em 3D e 2D, utilizando os recursos: *preloader*, física, *sprites* (Desenhos do Jogo), grupos e animações de *sprites*, câmera, som, entradas de mouse, teclado, etc.

3.2 Desenvolvimento de um Jogo

Nesta fase do trabalho, foram desenvolvidos três jogos como exemplos, buscando unir os conceitos de física, animação, colisão e o uso de *sprites*.

No Unity, foi desenvolvido um jogo semelhante ao *flappybird* mas em um contexto diferente. O jogo é um avião que tenta desviar dos obstáculos, o objetivo é desviar o máximo possível. Vale ressaltar que os *sprites* dos jogos estão disponíveis no site [OPENGAMEART.ORG] gratuitamente, para ser utilizado em projetos pessoais e comerciais. Como mostra na Figura 4.

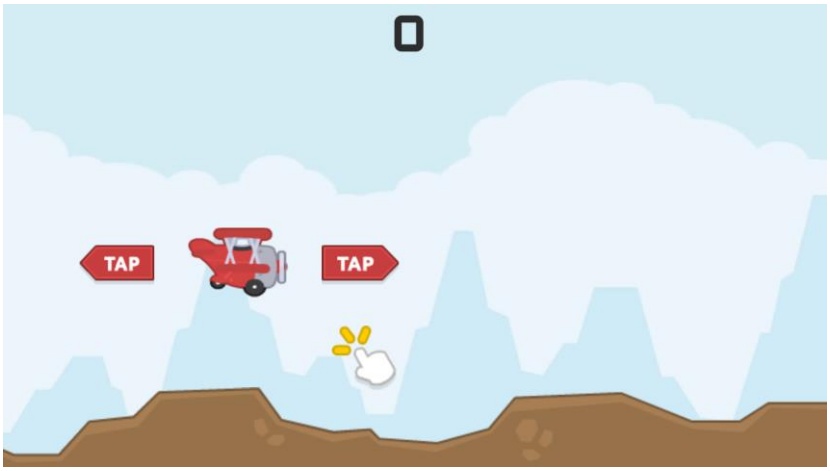

Figura 4. Aparência do jogo desenvolvido no Unity

No Godot, foi desenvolvido um jogo de nave, na qual o objetivo é destruir e desviar dos inimigos. Na Figura 5, é apresentado a aparência deste jogo.

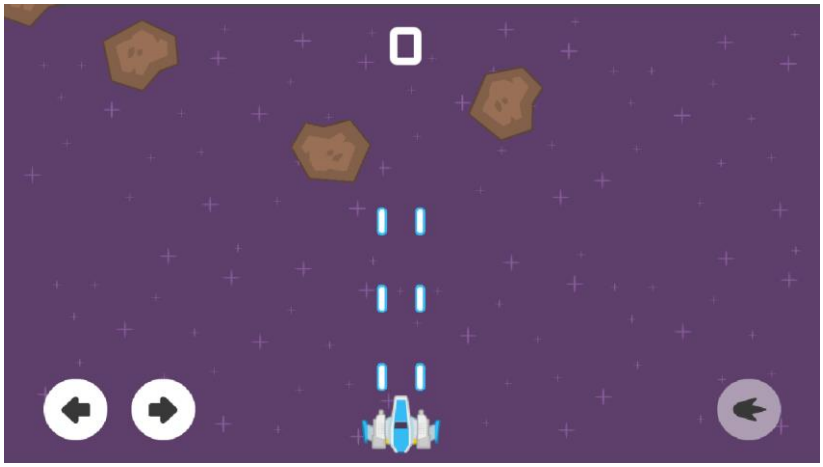

Figura 5. Aparência do jogo desenvolvido na Godot

No Phaser, foi desenvolvido um jogo para coletar cenouras que caem aleatoriamente do céu. A Figura 6, apresenta este exemplo.

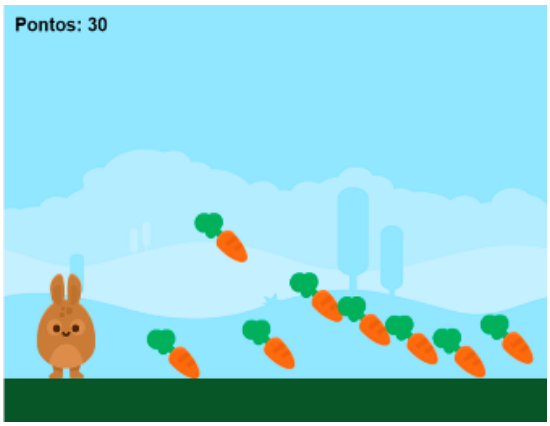

Figura 6. Apresenta a aparência de um jogo desenvolvido utilizando o *framework* Phaser

3.3 Vantagens e Desvantagens

Após conhecer os detalhes sobre cada uma das *engines* é possível identificar os pontos fortes e fracos de cada uma. Na tabela 1, está resumidamente as vantagens e desvantagens de cada *engine*.

Tabela 1. Vantagens e Desvantagens dos motores de jogos

Nome	Vantagens	Desvantagens
Unity	lataforma: Windows e Mac OS X ce completa e muito intuitiva. uma ótima documentação, tutorias e uma comunidade ativa. a gratuita, exporta jogos para Desktop (Linux, Windows e Mac OS X), Android, iOS, tvOS, Tizen, Xbox One, Windows Store, Samsung TV e HTML5.	de ser executada no Linux. ocumentação, tutorias e plataforma estão em inglês. ça gratuita existe limitações, por exemplo, não é possível exportar jogos para PS Vita e PS4. zer *download* completo desta *engine* é necessário uma boa conexão com a internet.
Godot	*ource*, totalmente gratuita. lataforma: Windows, Mac OS X, Linux. vel exportar jogos para HTML5, Android, BlackBerry 10, Linux, Mac OS X, iOS e Windows. necessário ser instalada e possui várias ferramentas integradas. r traduzida para português. ita de pouco conhecimento em programação, por ser muito visual.	é uma ferramenta muito intuitiva é necessário se adaptar a *engine* por serem várias as configurações para que um jogo execute corretamente. uma linguagem própria para *engine*, mas se mostrou muito simples.
Phaser	vel criar jogos multiplataforma que rodam levemente em navegadores de *desktop* e dispositivos móveis. um editor *online* e é possível salvar e compartilhar seu projeto para que ajude outras pessoas. uma ótima documentação e vários tutorias com exemplos práticos.	gos somente para *web*, e para exportar para outras plataformas são necessárias ferramentas auxiliares, como por exemplo, o Cordova. ilizar o Phaser é necessário instalar um servidor *web* e um editor, para criar os jogos. mentação e tutorias são escassos e estão em inglês.

4. Conclusões

Após a realização de todas as etapas em cada uma das *engines*, teve-se maior facilidade para a criação dos jogos testes no Godot por ser bem mais leve, ter a possibilidade de exportar jogos para várias plataformas e sua interface e linguagem de programação são bem amigáveis. Esta análise foi de grande relevância para o primeiro contato e aprendizagem dos conceitos relacionados ao desenvolvimento de jogos, tais como, colisão, física, cenas, etc. Dessa forma, conseguiu-se atingir o objetivo deste trabalho, onde se pretendia determinar a escolha de uma *engine* mais adequada e que facilitasse o desenvolvimento do jogo a ser implementado. Como trabalhos futuros, primeiramente, pode-se sugerir que as *engines* avaliadas sejam apresentadas a outros alunos que não possuem conhecimentos de programação. Assim, eles poderão criar os mesmos jogos de testes desse trabalho e no final apresentar as *engine* que proporcionou maior facilidade. Outro trabalho futuro é a finalização dos requisitos e implementação do jogo de educação financeira que motivou esse artigo.

Referências

Assis, G. A.; Ficheman, I. K.; Corrêa, A. G. D.; Netto, M. L. e Lopes, R. D. (2006). EducaTrans: um Jogo Educativo para o Aprendizado do Trânsito. 10f. Laboratório de Sistemas Integráveis (LSI) – Escola Politécnica da Universidade de São Paulo (USP), São Paulo.

Global Financial Literacy Excellence Center. (2017) "Standard & Poor's Ratings Services Global Financial Literacy Survey", http://gflec.org/initiatives/sp-global-finlit-survey/, Outubro.

Papert, S. A. (2008). Máquina das Crianças: Repensando a Escola na Era da Informática, Artmed, 1th edição.

Yin, R. K. (2003). Applications of case study research, Thousand Oaks: Sage Publications, 2th edição.