MINISTÉRIO DA EDUCAÇÃO
SECRETARIA DE EDUCAÇÃO PROFISSIONAL E TECNOLÓGICA
INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA FARROUPILHA
CAMPUS PANAMBI

FLIPPUNPKIN – JOGO ELETRÔNICO

TRABALHO DE CONCLUSÃO DE CURSO

William George Rocha dos Santos

Panambi, RS, Brasil.
2021

FLIPPUNPKIN – JOGO ELETRÔNICO

por

William George Rocha dos Santos

Monografia apresentada ao Instituto Federal de Educação Ciência e Tecnologia Farroupilha, como requisito parcial para obtenção do título de Tecnólogo em Sistemas para Internet.

Orientador: Profª. Drª Rosana Wagner

Panambi, RS, Brasil

2021

Ministério da Educação
Secretaria de Educação Profissional e Tecnológica
Instituto Federal de Educação Ciência e Tecnologia Farroupilha

A Comissão Examinadora, abaixo assinada,
aprova a Monografia

FLIPPUNPKIN JOGO ELETRÔNICO

elaborada por
William George Rocha dos Santos

como requisito parcial para obtenção do título de
Tecnólogo em Sistemas Para Internet

COMISÃO EXAMINADORA

_____
Profª Drª Rosana Wagner
(Presidente/Orientador)

_____
Rodrigo Luiz Antoniazzi

_____
Thiago da Silva Weingartner

Conceito Final: _____

Panambi, ... de ..... de 2021.

RESUMO

Com o passar dos anos a tecnologia avançou muito, fazendo com que nós seres humanos, conseguise otimizar tarefas simples e complexas por meio de sistemas operacionais, utilizando de técnicas e conceitos de programação para desenvolver esses sistemas. Os sistemas operacionais, na maioria das vezes são voltados para corporações, com o intuito de facilitar e organizar o processo de trabalho, e que hoje estão com a tendência de desenvolvimento para WEB/MOBILE, mas também já algum tempo, como consequência do avanço da tecnologia, houve um grande espaço para o entretenimento, que nesse caso são jogos eletrônicos. Os jogos eletrônicos não são novidades nos dias de hoje, e já existem a algum tempo, já vem lá do século passado, onde se tem os primeiros relatos de jogos eletrônicos desenvolvidos. Mas o tempo passou e a tecnologia avançou, e com isso o desenvolvimento de jogos ficou mais acessível e popular, pois as possibilidades de desenvolver um jogo eletrônico ficou muito maior e fácil, comparado com os primeiros jogos eletrônico desenvolvidos, onde tinha pouco recurso comparado com os dias de hoje. Com base nessas informações, o objetivo desse trabalho é desenvolver um jogo eletrônico para entretenimento utilizando as linguagens HTML, CSS e javascript, no qual tem como temática o universo do *halloween*, onde terá na sua mecânica de jogo, a tarefa de conduzir o seu personagem principal, para que o mesmo passe por obstáculos, sem cair ao chão e sem sofrer colisões com os obstáculos, assim somando pontos, nos quais são classificados com medalhas de ouro, prata e bronze e papel.

Palavras-chave: Jogos Eletrônicos, Entretenimento, Desenvolvimento de software

ABSTRACT

Over the years, technology has advanced a lot, making us human beings manage to optimize simple and complex tasks through operating systems, using programming techniques and concepts to develop these systems. Operating systems, most of the time, are aimed at corporations, in order to facilitate and organize the work process, and which today are with the development trend for WEB/MOBILE, but also for some time, as a result of the advancement of technology, there was a large space for entertainment, which in this case are electronic games. Electronic games are nothing new these days, and have existed for some time, as far back as the last century, where the first reports of developed electronic games are available. But time passed and technology advanced, and with it the development of games became more accessible and popular, as the possibilities of developing an electronic game became much bigger and easier, compared to the first developed electronic games, where it had little resource compared to nowadays. Based on this information, the objective of this work is to develop an electronic game for entertainment using the languages HTML, CSS and javascript, which has as its theme the universe of halloween, which will have in its game mechanics the task of conducting your character main, so that it passes through obstacles, without falling to the ground and without suffering collisions with obstacles, thus adding points, in which they are classified with gold, silver, bronze and paper medals.

Key words: *Electronic Games, Entertainment, Development of software*

Sumário

1 INTRODUÇÃO

Falar em jogos eletrônicos hoje se tornou comum, ao ponto de ser tão popular quanto jogos tradicionais coletivos ou individuais que existem no mundo todo, como futebol, basquete assim como outros. Mas nem sempre foi assim, isso por que logo no início, quando surgiram os primeiros relatos de jogos eletrônicos, esse assunto era pouco comentado, pois não se tinha muitos recursos de desenvolvimento para esse tipo de *software* na época, assim como consumidores desses jogos os populares usuários.

Depois de muitos estudos, foi constatado por historiadores onde entraram em um consenso que em 1958, foi criado um dos primeiros jogos eletrônicos, o tênis para dois (*Tennis for Two*), que segundo Amorin(2006) foi criado pelo físico *Willy A. Higinbothan*, que por sua vez criou um jogo eletrônico relacionado ao esporte, no qual simulava uma partida de tênis. O jogo foi criado para ser exposto no Laboratório Nacional de Brookhaven, onde Higinbothan descobriu que o modelo 30 do computador analógico, que era usado para pesquisar governamentais dentro da instituição, poderia fazer a simulação da trajetória com a resistência do vento. Com isso teve a ajuda de um técnico Robert V. Dvarak, onde em poucas semanas desenvolveram o jogo assim podendo expor o mesmo na exposição, onde conseguiam visualizar o jogo em um osciloscópio de jogar com dois controles.

Nesse jogo a bola é rebatida em uma linha horizontal na parte inferior da tela de um osciloscópio, existe também uma linha vertical no centro que apresenta a rede. Há duas caixas como um potenciômetro e um botão para que o jogador controle o jogo. Os potenciômetros afetam o ângulo, a bola da bola e o botão rebate a bola de volta para outro lado da tela. Caso o jogador erre o ângulo, a bola cai da rede. Existe um botão de reset que permite o jogador reinicie o jogo fazendo com que a bola reapareça do outro lado da tela. Nesse jogo não há placar e a tela é o cinescópio de fósforo verde monocromático de um osciloscópio (AMORIN, 2006).

Durante o ano de 1993 foi feito mais um lançamento e um avanço nas tecnologias de desenvolvimento de jogos eletrônicos, com o lançamento do jogo *Virtua Fighter,* pela empresa *SEGA*, que utilizou animação por objetos tridimensionais, onde foi levado em consideração a massa e aceleração destes em tempo real de processamento (BATISTA, 2007). Com o lançamento desta tecnologia, outros jogos começaram a utilizar esses recursos gráficos de gerenciamento geométrico, assim a tecnologia 3D foi adotada de forma unânime na produção de jogos (BATISTA, 2007).

Com o passar dos anos a tecnologia evoluiu, e com isso nos apresentou uma grande opção de formas de desenvolvimento de softwares, com conceitos e

tecnologias que proporcionam uma forma de desenvolvimento de softwares muito mais moderna e rápida. Essa evolução implica diretamente nos jogos eletrônicos, que tiveram um respaldo e um crescimento expressivo nos últimos anos, isso devido a evolução da tecnologia como um todo, onde apresentou um crescimento de consumidores desses jogos, o que cada vez cresce mais e mais, fazendo com que os jogos eletrônicos fiquem mais populares ao ponto de ter um espaço muito grande no mundo do desenvolvimento de software, voltado para os jogos eletrônicos.

Com base nessas informações, o estudo apresentado nesse trabalho terá como objetivo o desenvolvimento de um jogo eletrônico, um protótipo(*Demo*), o mesmo será desenvolvido utilizando técnicas e conceitos de desenvolvimento de softwares tradicionais e modernos, onde tem como linguagem de programação HTML, CSS, e javascript. O jogo apresentara uma temática e senário com base no *halloween,* onde o objetivo e juntar mais ponto, que são convertidos em medalhas, outo, prata e bronze, que são adquiridos conforme a evolução do percurso, onde será testada as habilidades do jogador, pois o mesmo deve conduzir o seu avatar no percurso sem cair no cão ou colidir com os obstáculos.

2 OBJETIVOS

Nas fundamentais etapas para o desenvolvimento deste projeto estão o objetivo geral e objetivos específicos, os quais estão descritos a seguir.

2.1 Objetivo Geral

Desenvolver um jogo eletrônico utilizando para entretenimento, aplicando técnicas de desenvolvimento de software que se aplicam tanto para jogos eletrônicos quanto para softwares de corporações assim como outros, e utilizando como base as linguagens HTML, CSS e javascrit.

2.2 Objetivo Específico

a) Levantar os requisitos funcionais e não funcionais do jogo eletrônico usando técnicas de análise de sistema, definindo estrutura de senário, movimentos e imagens que serão utilizadas para o desenvolvimento do jogo, e organizar as informações coletadas, para que as mesmas sejam aplicadas durante o desenvolvi emento do jogo eletrônico de forma organizada.

b) Desenvolver o protótipo(*Demo*) do jogo eletrônico com base nos requisitos levantados(Funcionais e Não Funcionais), aplicando as técnicas de desenvolvimento tradicionais e modernas de acordo com a necessidade e complexidade dos problemas apresentados para a construção do jogo, e tornar o jogo funcional, apresentando um jogo com imagens e movimentos bem definidos possibilitando o entretenimento do usuário durante a partida.

3 JUSTIFICATIVA

Com base nos conhecimentos estudados e adquiridos no curso de Tecnologia de Sistemas para Internet, decidi desenvolver um software para o entretenimento, ou seja, um jogo eletrônico.

A ideia partiu do ponto onde é sair da zona de conforme ariscar fazer algo diferente do que estamos acostumados no dia a dia, onde criei e desenvolvi aplicações corporativas como controle de estoque, aplicativos de entregas entre outros.

Nesse sentido foi verificado a possibilidade de desenvolver um jogo eletrônico onde o mesmo tem como premissa o entretenimento, e no qual não tivesse um escopo muito grande e como isso tivesse o desenvolvimento rápido e assertivo.

Com isso a ideia tem como base explorar esse conhecimento adquirido durante o curso, aplicando os conceitos tradicionais e modernos da programação, tendo como base linguagem de marcação, HMTL e CSS, e para interagir com essa linguagem de marcação, fazer a dinâmica das informações o javascript, onde será aplicado técnicas tradicionais e populares, que são usadas em diversas formas de desenvolvimento de software, ou seja, que não são usadas só para o desenvolvimento de jogos mas que podem ser aplicadas em vários tipos de software que utilizam desses conceitos.

4 REFERENCIAL TEÓRICO

Diante disso, nesse capítulo estão descritas considerações sobre as ferramentas necessárias para o desenvolvimento do jogo eletrônico.

4.1 Requisitos do Software

No desenvolvimento de *software*, o principal processo, é compreender os a necessidades do cliente para que seja atingido o objetivo, tornando assim um dos pontos de partida quando iniciado um projeto de software, onde podem ser aplicadas em qualquer modelo de metodologia (DAVIS, 1994).

Esse processo é responsável por buscar as informações e necessidades do cliente, assim classificando como levantamento de requisitos. O mesmo está presente no processo de desenvolvimento como ponto de partida, ou seja, é o primeiro ciclo do desenvolvimento de software, e que por sua vez defini as funcionalidades e o escopo do projeto (DEVMEDIA, 2009).

Em um levantamento de requisitos e extremamente necessário a troca de informações entre o analista e o cliente. No projeto que é a construção de um jogo o usuário final é o cliente, nesse caso o analista devera compreender e tentar explorar o lado do usuário, onde deve ter um olhar crítico sobre o software, ter em mente o cenário que o software vai ser applicado, para que seja evitado problemas durante o incremento dos requisitos (LOPES et al.,2004).

4.2 HTML

O HTML é uma linguagem de marcação, que apresenta hipertextos que representam conjuntos de elementos, que são denominados hiperlinks, esses elementos podem ser imagens, vídeos, palavras, documentos, assim como outros. Segundo Silva (2014, p. 26) HTML(*Hypertext Markup Language*) são informações inseridas em documento web, tendo em sua principal característica a interligação de outros documentos web.

4.3 CSS

De acordo com a organização W3C(2016), é definido que o CSS (*Cascading Style Sheet*) é uma linguagem que apresenta uma forma de descrever o conteúdo de páginas web, onde é possível organizar e visualizar as informações de uma página em

diversos tamanhos de tela, que quando somadas ao HTML apresenta uma forma de manutenção de páginas mais acessíveis.

4.4 JavaScript(JS)

No início quando o javascript apareceu por volta de 1995, a ideia era manipular as autenticações das entradas de informações que estavam sendo esquecidas por linguagem que atuavam do lado do servidor como *Perl*. Com isso era necessário fazer uma chamada para o servidor, para verificar se a informação necessária estaria em branco ou valor nulo. Assim O *Netscape* trouce a possibilidade de alterar essa questão com a introdução ao javascript, que possibilitou fazer validações mesmo que básicas no cliente no qual foi um recurso muito empolgante no momento pois a atual situação como navegação lenta que os modens apresentavam, tornava a as chamadas ao servidor algo complexo e demorado exigindo paciência (ZAKAS, 2012).

Em concordância com Borba(2006), o javascript, está entre as linguagens mais usadas no planeta e dentro do mundo da internet. Pois a mesma possibilita codificar essas informações que estão em campo de formulários, cores, efeitos visuais, temas, e muitos outros.

4.5 Sprite

Segundo (FERNANDES, 2009), um *sprite* é um objeto bi ou tridimensional, onde o mesmo se move na tela sem deixar rastros ou traços. Com isso é possível fazer movimento com o personagem ou cenário dando a impressão de movimentação, onde esse sprite deve conter várias imagens que serão os movimento, ou seja, cada imagem vai conter um movimento, assim fazendo com que o *sprite* se mova livremente e também possa interagir com outros *sprite*.

Essa é uma técnica bastante aplicada em jogos 2d, onde é carregada a imagem com os movimentos na memória e então é feito o greb, ou seja, é um clip de uma parte da imagem em tempo real (FERNANDES, 2009).

4.6 Jogos Eletrônicos

Para o desenvolvimento de jogo eletrônico é preciso definir seus princípios. Segundo Schuytema (2008, p447 apud Lucchese e Ribeiro, 2009, p.8) um jogo eletrônico nada mais e que uma atividade voltada para diversão, que apresentam movimento e decisões que entregam um resultado final. Esses movimentos e decisões são controlados contendo um limite, apresentando um conjunto de regras que dentro da ideia de jogos eletrônicos, são comandados pelo software do computador

No desenvolvimento de um jogo eletrônico conseguimos encontrar vários caminhos para organizar a estrutura do mesmo. Segundo Battaiola (2000, p. 83–122, apud Lucchese e Ribeiro, 2009, p.8) um jogo eletrônico é formado por três partes: enredo, motor e interface interativa. O tema define a história a trama do jogo, o objetivo do jogo e como ira se desenvolver os acontecimentos durante o jogo, o motor é o que controla ações e decisões do jogador, e a interface que é a interatividade do motor com o jogador, que é um caminho de entrada para os movimentos do jogador, e a saída com resultados visuais que apresenta à mudança dos processo.

4.7 Métodos Ágeis

Com o passar dos anos, o desenvolvimento de software passou por muitas evoluções e uma delas está nos processos de desenvolvimento, ou seja, tornar o processo de desenvolvimento mais rápidos. Com isso foi introduzido no ambiente de desenvolvimento de software assim como nos jogos a metodologia ágil. Para Keith (2010, p. 22), um dos pontos benéficos do desenvolvimento ágil nos jogo eletrônicos é a confecção de pequenas etapas, que vão apresentando o resultado em partes para o cliente final.

A metodologia ágil tem dua origem com o sistema Toyota de produção popularmente conhecido por STP, no qual tem seu início no que conhecemos hoje por filosofia de *Lean Thinking*, ou seja, Pensamento Enxuto. Essa filosofia apresenta uma forma de melhoria contínua na produção de um modo geral, e com ela apresentam ferramentas que possam assegurar a qualidade do projeto de modo ágil, ou seja, são caracteristicas que possibilitam, acompanhar, gerenciar e definir ações dentro da produção.

O desenvolvimento de jogos digitais na atual realidade do Brasil, apresentam aos profissionais da área algumas limitações, como orçamentária, qualificação específica e adequada, e o próprio tempo, que muitas vezes é um grande vilão. Com isso é apresentando de forma extremamente importante a utilização dos conceitos e técnicas dos métodos ágeis, para gerir o tempo e a qualidade da produção do jogo.

5 METODOLOGIA

Esse trabalho tem como objetivo aplicar os estudos e conhecimento adquiridos no curso de Sistemas para Internet, no desenvolvimento de um software, nesse caso foi escolhido desenvolver um jogo eletrônico, utilizando HTML, CSS *e* javascript, onde o mesmo será desenvolvido utilizando conceitos e metodologias ágeis, no qual vai ter seu ciclo de desenvolvimento dividido em etapas.

No primeiro momento, será feito a história do jogo, contando um pouco sobre o jogo e suas características, ainda nessa parte será feito o levantamento de requisitos, onde será verificada a mecânica do jogo, os movimentos que o personagem fará, o tema, cores e imagens que serão aplicadas.

No segundo momento vai ser trabalhado o tema do jogo, onde será feito buscas por imagens para o jogo, onde poderão ser adquiridas imagens gratuitas ou até mesmo criar alguma imagem para cenário, assim como o áudio para o jogo. Logo em seguida serão feitos os levantamentos das ferramentas que serão utilizadas para o desenvolvimento do jogo, como editor de texto, navegador, entre outros.

E por fim será feito o desenvolvimento do jogo, aqui será utilizada a metodologia ágil, onde será dividido o desenvolvimento em pequenas partes, assim tendo um desenvolvimento mais rápido e assertivo, podendo assim identificar possíveis erros ou melhorias em tempo de desenvolvimento sem causar impactos na entrega do trabalho.

6 DESENVOLVIMENTO DO PROJETO

6.1 História

Nos jogos eletrônicos é apresentada a história, conhecido por alguns também por enredo que apresenta para o usuário ou jogador, uma ideia geral sobre o jogo. No caso deste projeto que tem como premissa o desenvolvimento de um jogo eletrônico, temos a história, que é bem simples pois se trata de um jogo demo, ou seja, uma demostração do jogo.

O jogo conta a história de um pássaro que tem como objetivo atravessar por um ambiente caótico e assustador de sua floresta, enfrentando obstáculos durante um trajeto onde os mesmos mudam conforme o decorrer do jogo, para alcançar o objetivo de ganhar a medalha dourada.

6.2 Requisitos

Em uma análise inicial foram observados os requisitos funcionais e não funcionais do jogo, nos quais são de suma importância para o desenvolvimento do jogo, onde é possível aperfeiçoar o jogo de uma forma que se obtenha uma melhor desempenho, assim como tornar futuras atualizações e melhorias do jogo mais fáceis e organizadas.

6.2.1 **Requisitos Funcionais**

Nos requisitos funcionais são apresentados:

Controle do Jogo

1. **Inicializar Jogo**
    1. **Prioridade:** Essencial
    2. **Descrição:** A aplicação quando iniciada já apresenta a configuração zerada(pontuação(s*core*)) e mostra tela de início do jogo apresentando mensagem para que o usuário inicie o jogo.
2. **Continuar o Jogo**
    1. **Prioridade:** Desejável
    2. **Descrição:** Após final de jogo o usuário é redirecionado na tela inicial do jogo podendo assim reiniciar a partida. O jogo é reiniciado voltando com a configuração zerada(pontuação(*score*)) e do ponto inicial da partida.

3. **Fim de Jogo (*Game Over*)**
    1. **Prioridade:** Essencial
    2. **Descrição:** Apos colidir com algum obstáculo ou cair ao chão, é apresentada tela de fim de jogo (*game over*) com as informações de pontuação(*score*) a classificação em forma de medalha(branca, bronze, prata e dourada) e ainda a melhor pontuação alcançada.

Mostrar Informações na Tela

1. **Mostra Pontuação(*score*)**
    1. **Prioridade:** Essencial
    2. **Descrição:** Aplicação mostra a pontuação em tempo real conforme evolução dentro do jogo.

Movimentos

1. **Movimentos do Personagem**
    1. **Prioridade:** Essencial
    2. **Descrição:** O usuário tem o controle do personagem através do click do mouse, o que faz com que o personagem movimente-se para cima assim evitando que o mesmo colida com os obstáculos e com o chão.
2. **Som de Colisões**
    1. **Prioridade:** Essencial
    2. **Descrição:** Quando o personagem sofre uma colisão ou uma queda ao chão, e representado com um som esse impacto.

6.2.1 **Requisitos Não Funcionais**

1. **Tempo de Resposta**
    1. **Prioridade:** Importante
    2. **Descrição:** O usuário não deve esperar mais que 2(dois) segundo, após o início do jogo para movimentar o personagem pois isso pode apresentar fim de jogo.
2. **Interface Intuitiva**
    1. **Prioridade:** Importante

2. **Descrição:** Possibilitar para o usuário uma interação com o jogo de forma rápida e clara.

3. **Pouco Processamento**

    1. **Prioridade:** Essencial
    2. **Descrição:** Aplicação que não exige muito processamento nem memória do computador.

6.3 Mecânica do Jogo

No desenvolvimento deste jogo eletrônico, é apresentada uma mecânica de movimentos e interação bem simples e intuitiva, apresentado um cenário com rolagem lateral, conhecido por *side-scroller,* que tem na sua característica uma visão do ângulo da câmera lateral, que à medida que o personagem vai movimentando para frente a tela de rolagem acompanha. Podemos visualizar na ilustração da imagem abaixo

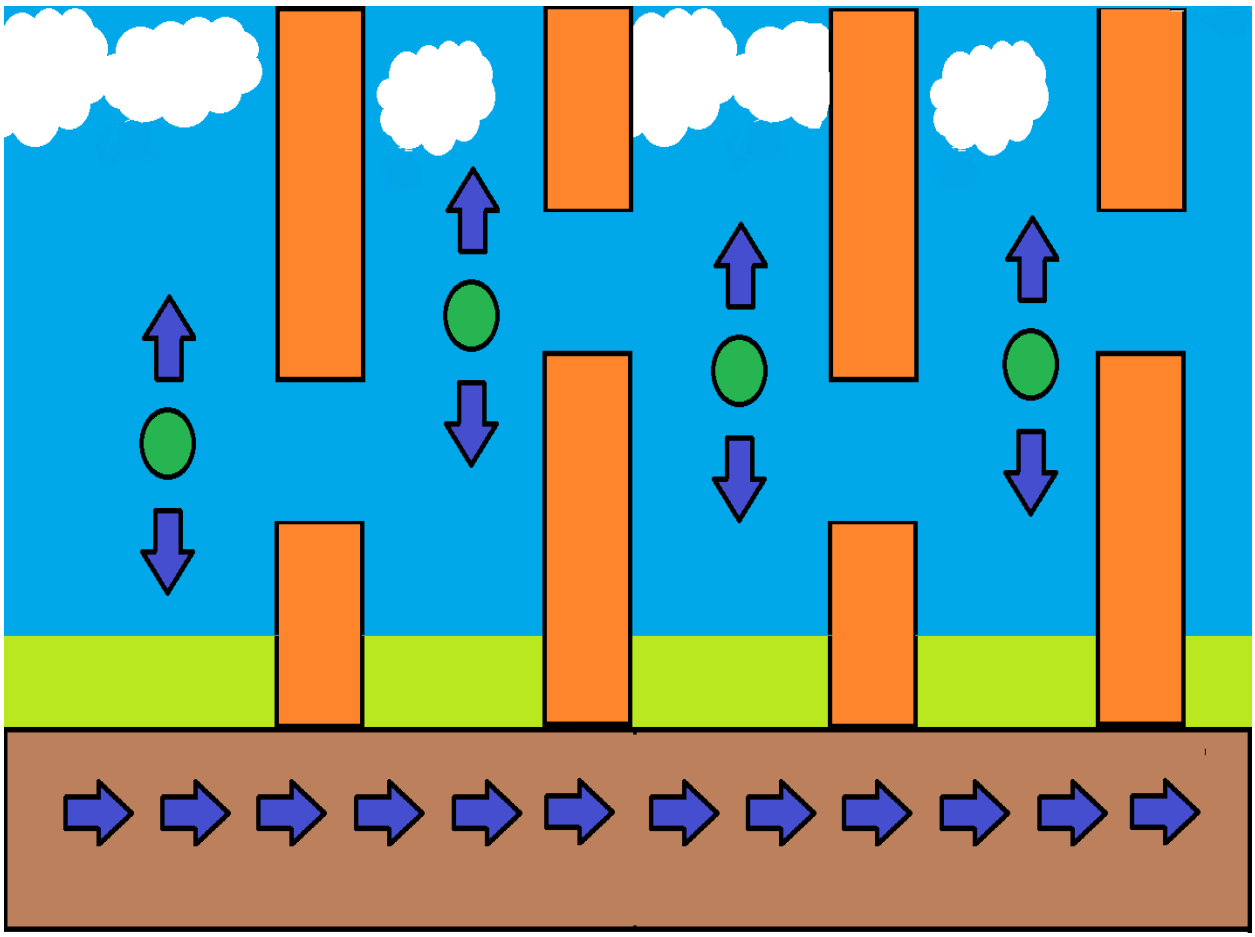

Figura 1: Representação da mecânica de movimentação do cenário do jogo e personagem

No nosso caso aqui o cenário tem a rolagem lateral automática dando uma característica um pouco mais complexa para o jogo, assim tornando para o jogador um jogo mais emocionante.

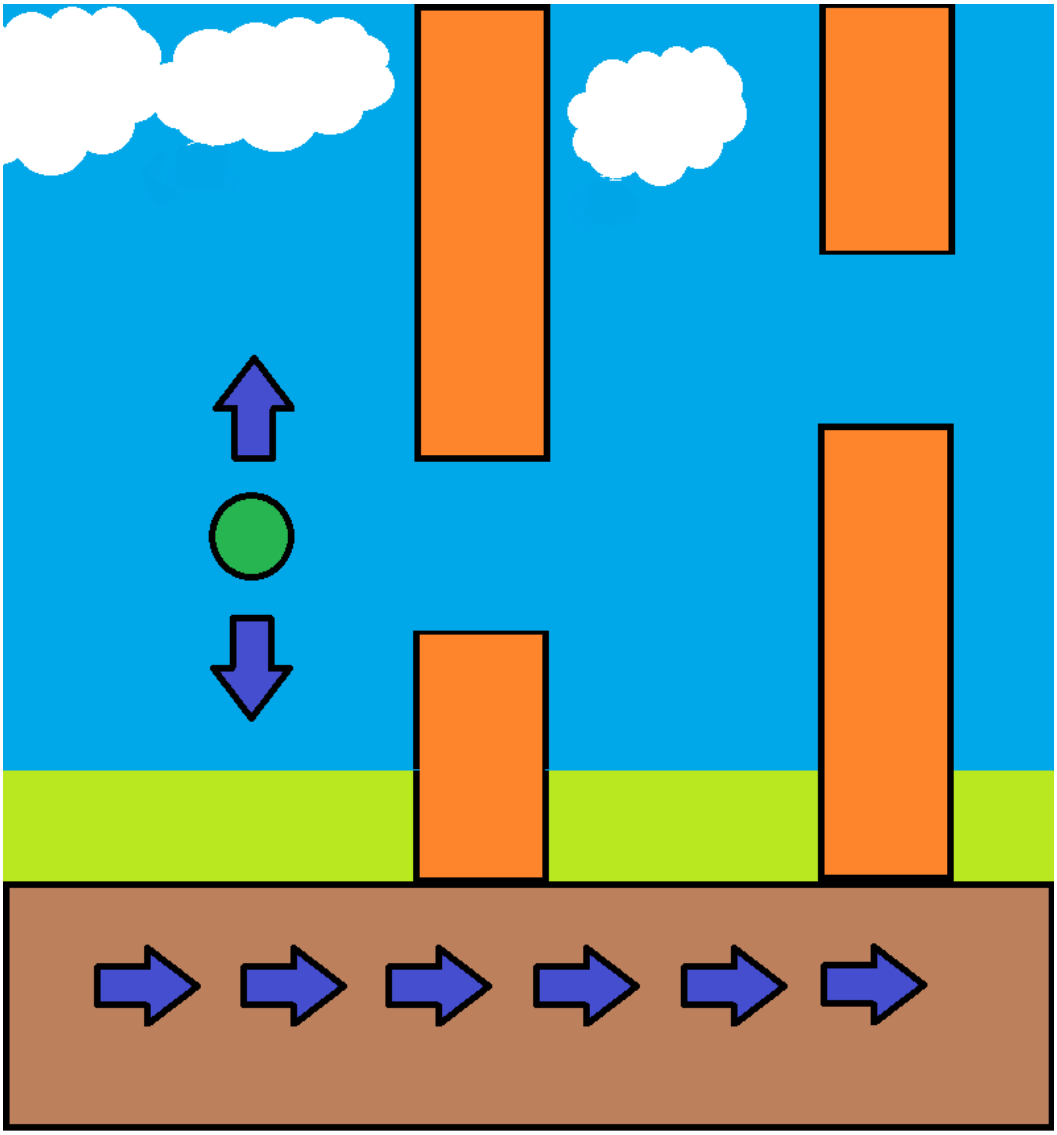

Figura 2: Representação da mecânica do movimento do personagem

Ainda temos o movimento do personagem representado pelo círculo na cor verde que vai ser executado quando houver um click no mouse, esse por sua vez vai ser o comando mais importante pois vai fazer com que o jogador possa conduzir o personagem do jogo durante o percurso sem esbarar nos obstáculos.

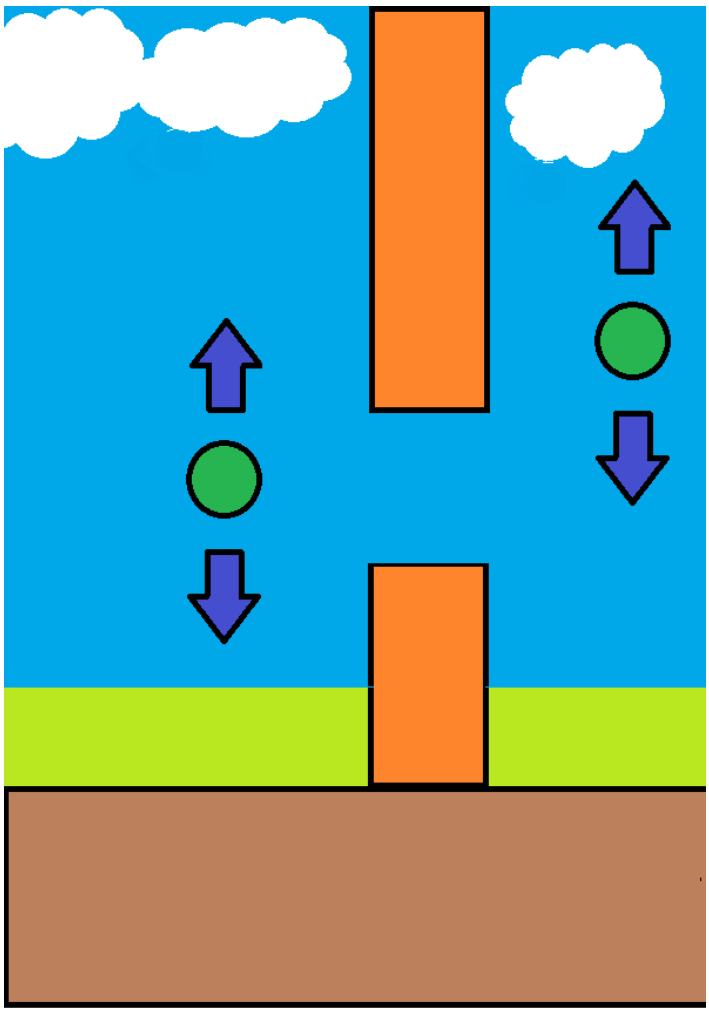

Figura 3: Apresenta os movimento do personagem vai ter dentro do jogo

6.4 Imagens

Com base na característica de desenvolvimento do jogo em 2d, usamos a técnica para carregamento de imagens com *sprite*, para isso foi montado um arquivo com todas as imagens para que a partir do mesmo, seja feito a inclusão das imagens no jogo conforme a necessidade.

Para montar esse arquivo com essas imagens, foram buscadas algumas imagens gratuitas na internet voltadas para jogos eletrônicos e com a característica utilizada em jogos 2d. Ainda foi utilizado o software de edição de imagens Gimp, que tem por sua característica ser opem *source*, ou seja, é de livre acesso onde não é preciso adquirir licença através de meios financeiros, e sem dúvida um software que tem todas as ferramentas para a construção do arquivo com as imagens.

Logo no início da escolha das imagens, verificou-se a necessidade de conseguir imagens que tivessem disponíveis na internet gratuitamente, para que o processo ficasse um pouco mais ágil, porém surgiram dificuldades em meio a procura. A dificuldade aconteceu logo no primeiro momento quando estive procurando por um avatar(personagem), pois o personagem que gostaria de usar não foi possível encontrar, então comecei a procurar por maneiras de criar um personagem.

Com isso por meio de pesquisar por um tempo encontrei a aplicação online PISKEL, onde seria possível criar o personagem como eu queria. A aplicação é totalmente gratuita e dispõem de bons recursos para a criação do personagem e imagens para jogos.

Na criação do personagem, desenvolvi o mesmo com base na ideia do tema que é o *halloween*, então como já tinha em mente, criei uma abóbora com assas, no qual é bem pressente no meio do *halloween*, assim dando uma ênfase maior ao tema. Para a criação do personagem no PISKEL, tive que buscar algumas informações de como seria o personagem pois o mesmo teria que obedecer alguns critérios, como por exemplo a sombra, curvas, olhos entre outros detalhes da imagem, já que a mesma tem a característica de ser em 2d. Depois de algum tempo entre erros e acertos a imagem do avatar ficou assim como mostra a Figura 4.

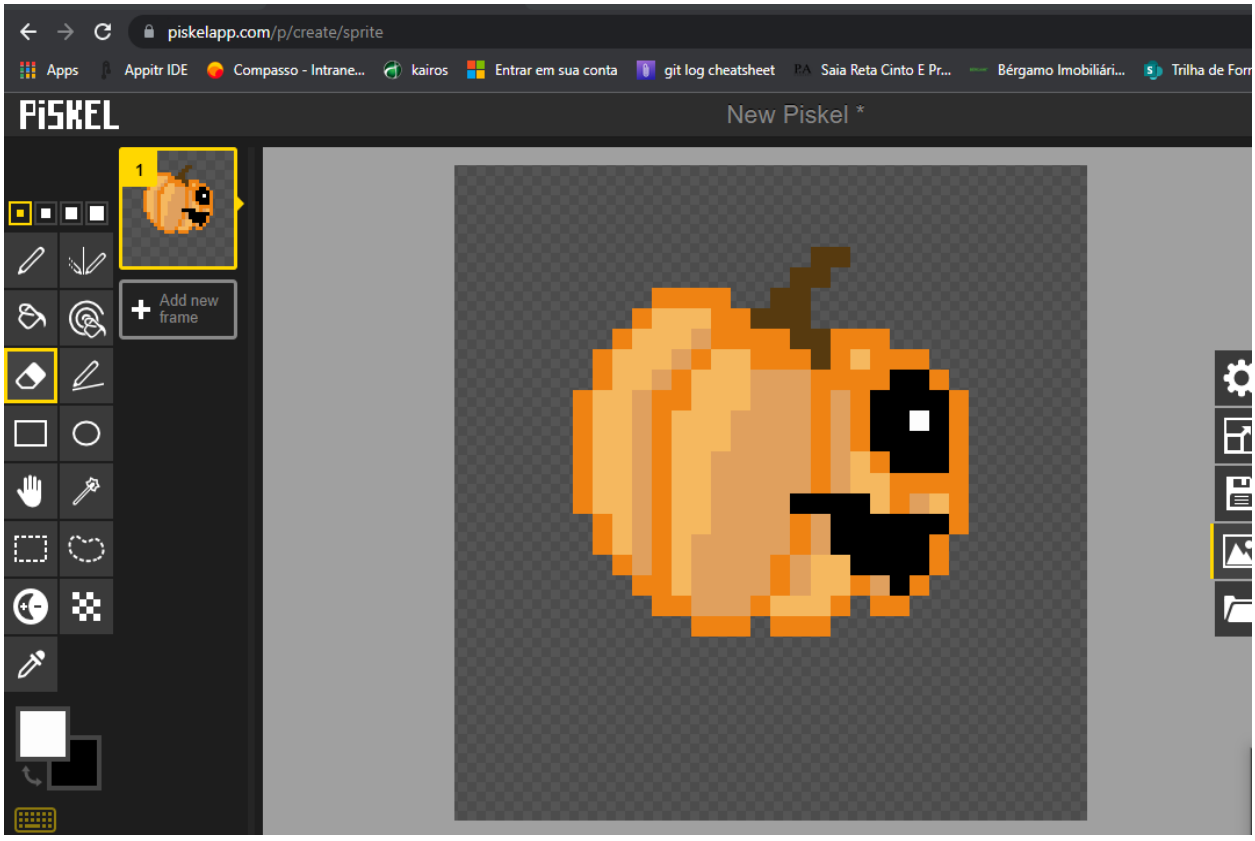

Figura 4: Criação do personagem

Após a criação do personagem, foi necessário fazer ainda alguns ajuste na imagem, e os mesmos foram feitos no Gimp. O ajuste do personagem, foi o que mais trabalhoso, pois o mesmo tem a característica de se movimentar e com isso incluiu-se as assas para o personagem, e o mais importante criar o movimento das assas, com isso foi usado três imagens do avatar para simular o movimento, ficando conforme Figura 5 logo na sequência.

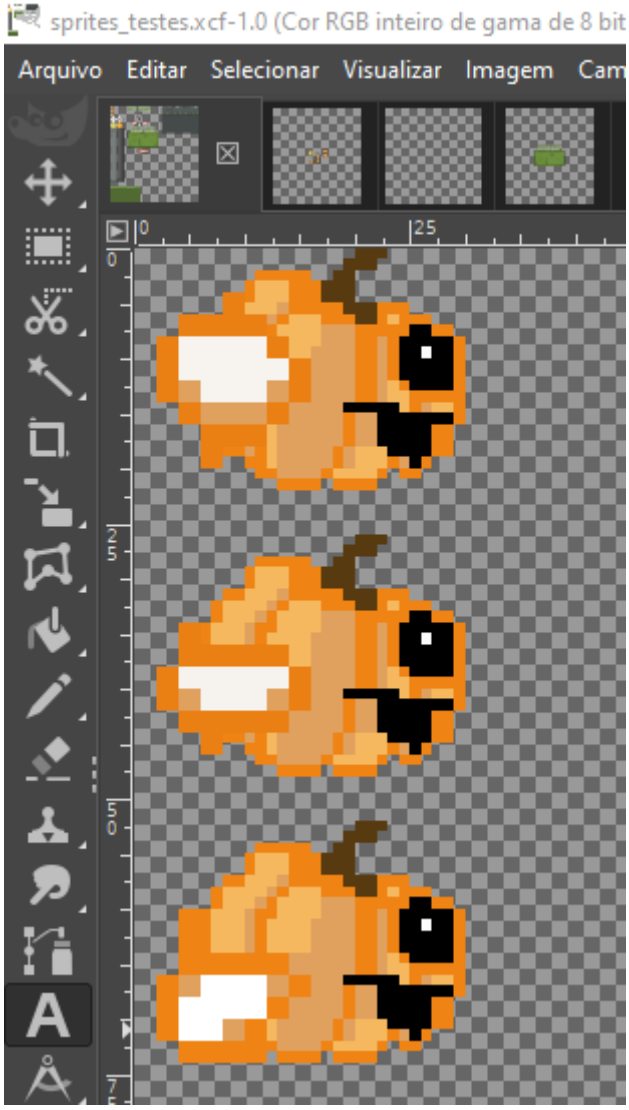

Figura 5: Representação dos movimentos do personagem por imagens

Podemos notar que para cara imagem tem um formato diferente na asa, isso é fundamental para que durante o jogo, o avatar se movimente de uma forma mais natural possível. Ainda as assas foram criadas dentro Gimp mesmo, obedecendo o critério usado no PIXKEL.

Avançando para outras imagens foi preciso ter um chão para o jogo, assim como o avatar foram feitas tentativas de buscadas imagens gratuitas na internet, mas tive dificuldades de conseguir, com isso optou-se por criar a mesma. A imagem do chão criei no Gimp, com as medidas de altura 226, e largura 772, colocando um detalhe de faixas para dar a impressão de movimento assim como no avatar, porém aqui em uma imagem só, onde a mesmo vai seguir continuamente, ou seja, a imagem vai ser repetida durante o jogo dando a impressão do movimento, então com uma imagem já resolve a questão. Podemos conferir na Figura 6.

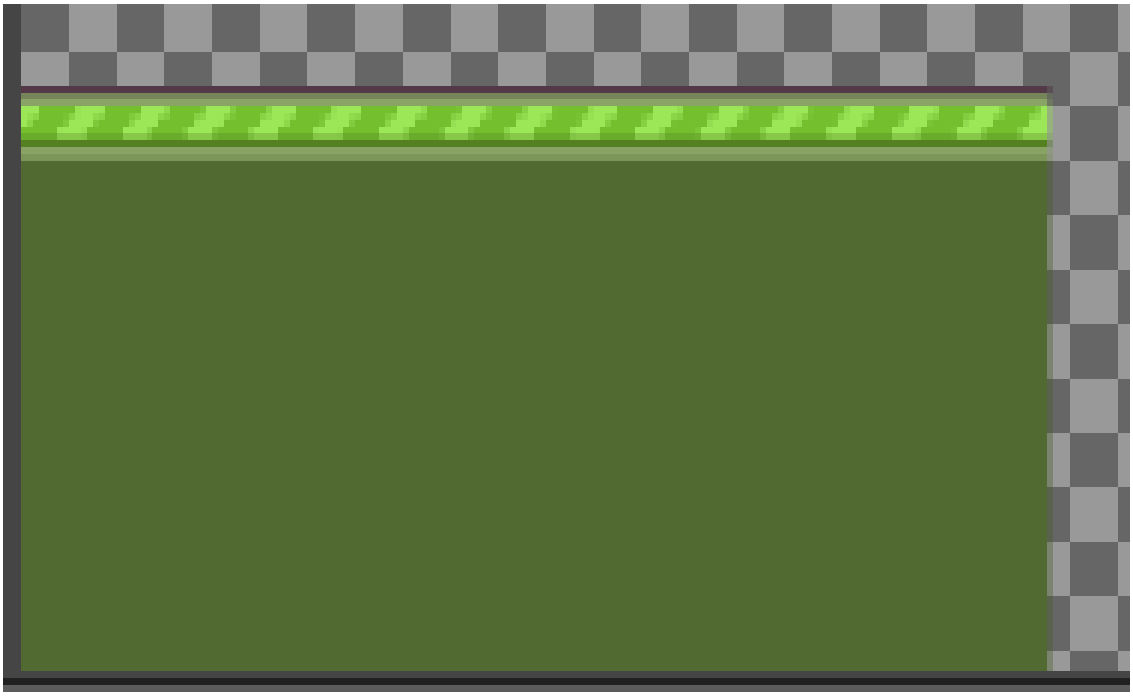

Figura 6: Representação do chão do jogo

Ainda no senário, precisei de uma imagem de fundo, onde representa-se um contexto de floresta e cidade, com isso foi feita a criação da imagem assim como a do chão criei no Gimp, ficando com a altura de 202 e largura de 666. Coloquei cores escuras para combinar com o tema do *halloween*, assim como tentei dar uma destaque para as os prédios colocando cores representando luzes.

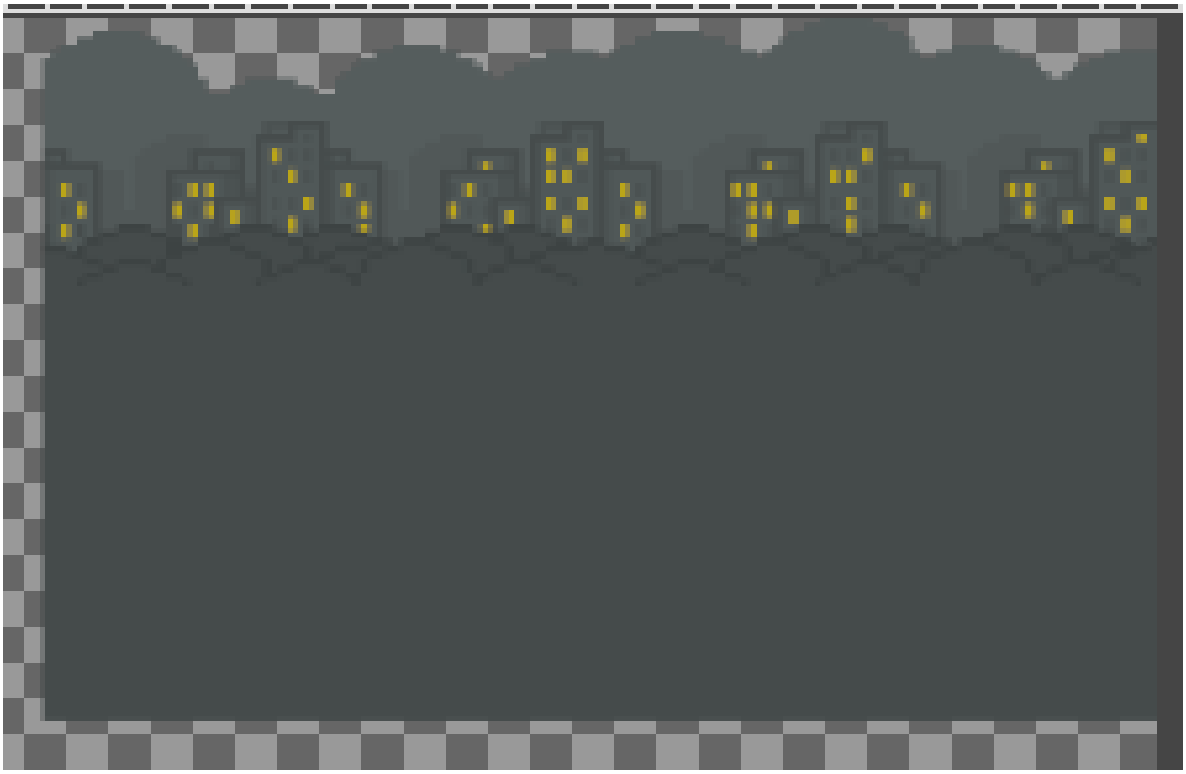

Figura 7: Fundo do jogo representado com elementos de cidade e árvore

Na sequência das imagens, foi necessário ter imagens de obstáculos, então foram criados canos, semelhantes os que são usados em jogos, como no Super Mário, entre outros então criei a mesma no Gimp. Foram criados dois canos com cores diferentes um com uma textura mais escura, já o outro tem uma textura um pouco mais clara para dar um tom diferenciado dentro do jogo.

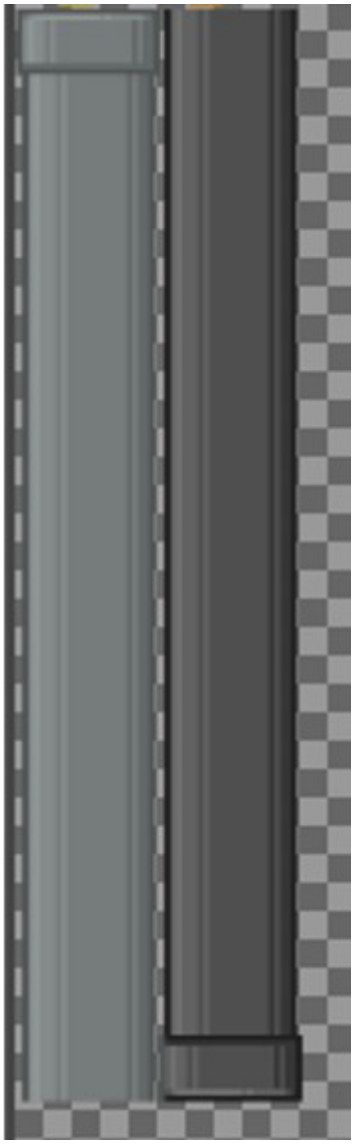

Figura 8: Obstáculos do jogo representados por canos

Também foram criadas imagens para as medalhas, as mesmas foram criadas no Gimp, nas cores, branca, bronze, prata e ouro. As medalhas contam com a imagem do personagem em cada uma delas, como podemos observar na Figura 9 o seu resultado.

Figura 9: Medalhes do jogo

Por fim foram criadas imagens para o início do jogo e para o final do jogo. A imagem inicial contém a descrição *Get Ready* que quer dizer prepare-se, ainda contém a imagem do avatar, e a imagem de uma mão pequena indicando que deve ser clicado. Já na imagem de fim de jogo ou *game over*, temos a literal game over, e um quadro com as informações, as mesmas apenas as literais referentes a medalha(*medal*), Pontuação(*score*) e Melhor Pontuação(*best*), contendo os espaços para as respectivas informações para cada uma, e ainda abaixo do quadro tem o botão s*tart*, que quando selecionado volta a tela inicial do jogo.

Depois de criada todas as respectivas imagens foram incluídas as mesmas em um único arquivo (*sprites*.png) contendo todas as imagens. Isso é uma característica

para que sejam organizadas as imagens que serão usadas no jogo em um arquivo. Esse arquivo ficou conforme Figura 10.

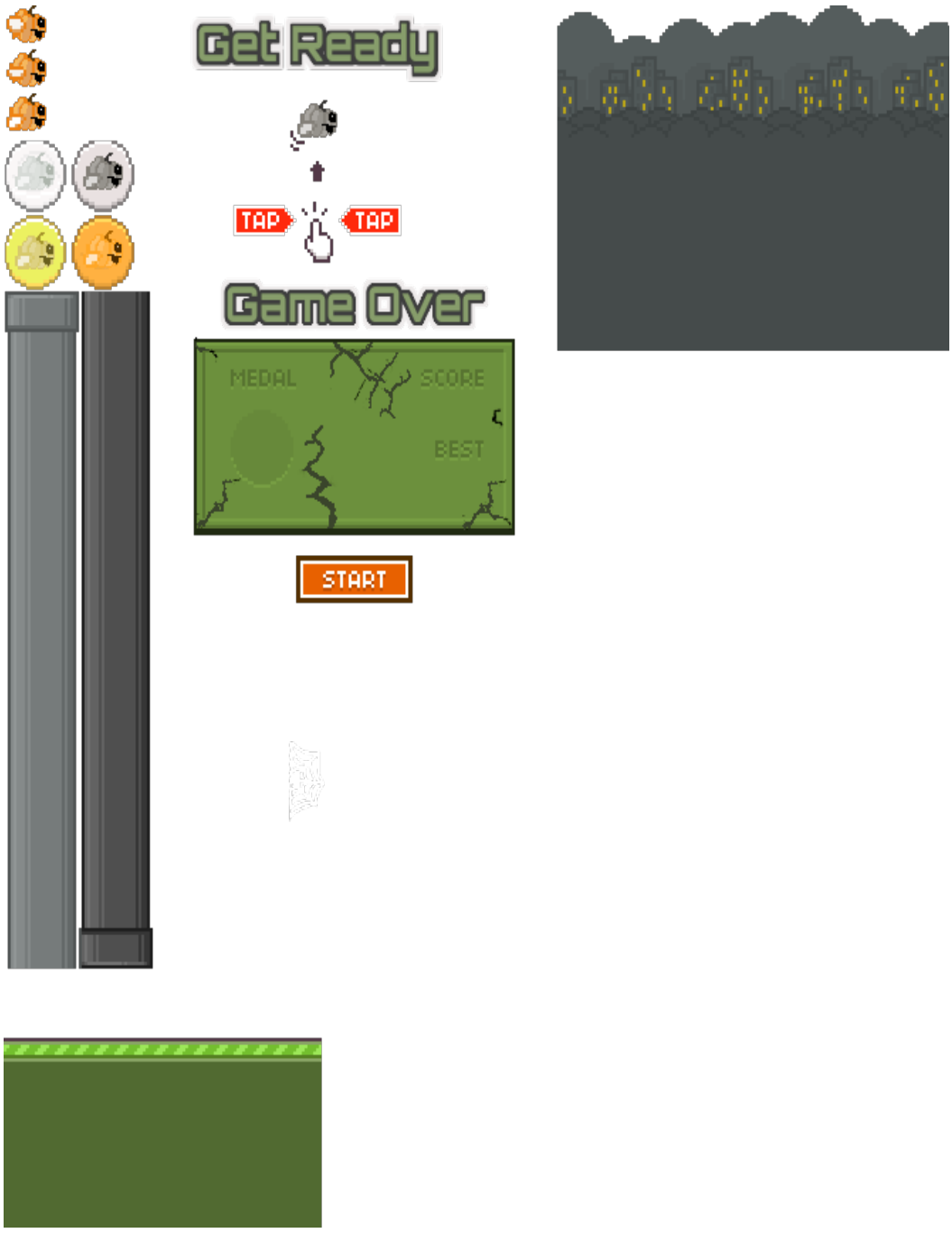

Figura 10: Sprite do jogo, contem todas as imagens que serão usadas no jogo

6.5 Análise de Sistemas

Na análise do sistema temos por base as informações coletadas nos requisitos(funcionais e não funcionais) levantados e apresentados anteriormente, assim como apresentar uma melhor forma e maneira de implementação da aplicação, no desenvolvimento do jogo, para que o mesmo atinga o objetivo desejado.

Logo no início da análise foi levado em consideração as ferramentas que serão utilizadas para o desenvolvimento do jogo, onde foi escolhido o editor de código Visual Estúdio Code, essa escolha aconteceu com base nas características do editor no qual eu tenho um bom conhecimento, ou seja, já tenho domínio na ferramente e ainda pelo fato que o editor proporciona um desenvolvimento mais rápido, aumentando a produtividade.

Já na parte de linguagem foi escolhido utilizar o HTML, CSS e javascript, pois hoje em dia com a evolução das linguagens já é possível desenvolver jogos com essas linguagens, e também por ser usada em muitos lugares, ou seja, não só no desenvolvimento de jogos mas em várias frentes de desenvolvimento.

A partir da escolha das ferramentas e linguagem, foi feito uma análise sobre maneiras de como desenvolver o jogo levando em conta a ideia e suas características assim como os requisitos.

Primeiramente para que tivesse uma melhor organização e entendimento do processo de desenvolvimento classifiquei as etapas para que tivesse mais claro cada etapa que usei para desenvolver o jogo.

1. **Aplicar Personagem e Senário do jogo.**
    1. O senário deverá seguir o modelo de layout para senário.
        1. Incluir fundo e chão.
            1. Senário deverá movimentar automaticamente.
        2. Incluir personagem.
            1. Incluir movimento do personagem.
                1. O mesmo terá movimento através do click do mouse.
        3. Tamanho da tela.
            1. A tela deverá ter 320X480.
2. **Aplicar Tela Inicial.**
    1. A tela inicial devera seguir o modelo do layout.
        1. Devera apresentar a escrita de início de jogo.
3. **Aplicar obstáculos.**

1. Obstáculos
    1. Deverá seguir o modelo de layout para obstáculos.
    2. Os obstáculos deverão ser apresentados de forma aleatória, não possuindo uma uniformidade, ou seja, deverá variar o tamanho obedecendo apenas a distância entre um e outro que devera ser a mesma.
    3. Os obstáculos devem ter uma distância do obstáculo de cima para o de baixo, onde seja possível passar com o personagem essa distância deverá ser igual para todos. Não permitindo passar por outro lugar que não seja esse espaço determinado para passagem.
        1. Caso tente passar por dentro do obstáculo será encerado o jogo, ou seja, fim de jogo (*game over*).
            1. O mesmo acontecera se colidir como chão do senário
4. **Aplicar tela de Fim de Jogo(*Game Over*)**
    1. A tela de fim de jogo devera seguir o modelo de layout para essa tela.
        1. Essa tela deverá ser apresentada sempre quando houver a colisão do personagem com algum obstáculo assim como ou colidir com o chão do senário.
        2. Deverá apresentar uma medalha conforme pontuação
            1. Branca;
            2. Bronze;
            3. Prata;
            4. Ouro;
        3. Deverá apresentar a pontuação(*score*).
        4. Deverá apresentar a melhor pontuação de todas.
5. **Aplicar pontuação em tempo real**
    1. Pontuação Conforme Evolução.
        1. Devera ser apresentada pontuação em tempo real conforme evolução durante o jogo.
            1. A mesma deverá ser apresentada na tela de fim de jogo.

6.6 Desenvolvimento

Para o início do desenvolvimento crie uma pasta, no qual vai conter os arquivos com as extensões HTML, e js, assim como os arquivos e pastas com informações de imagens e sons.

Já no início do projeto criei o arquivo HTML, onde o mesmo possui uma estrutura básica de página HTML, e no mesmo inclui a tag do canvas que por sua vez é responsável por receber o desenho, ou seja, todo o jogo vai ser desenhado dentro dessa tag, assim como podemos observar na Figura 11.

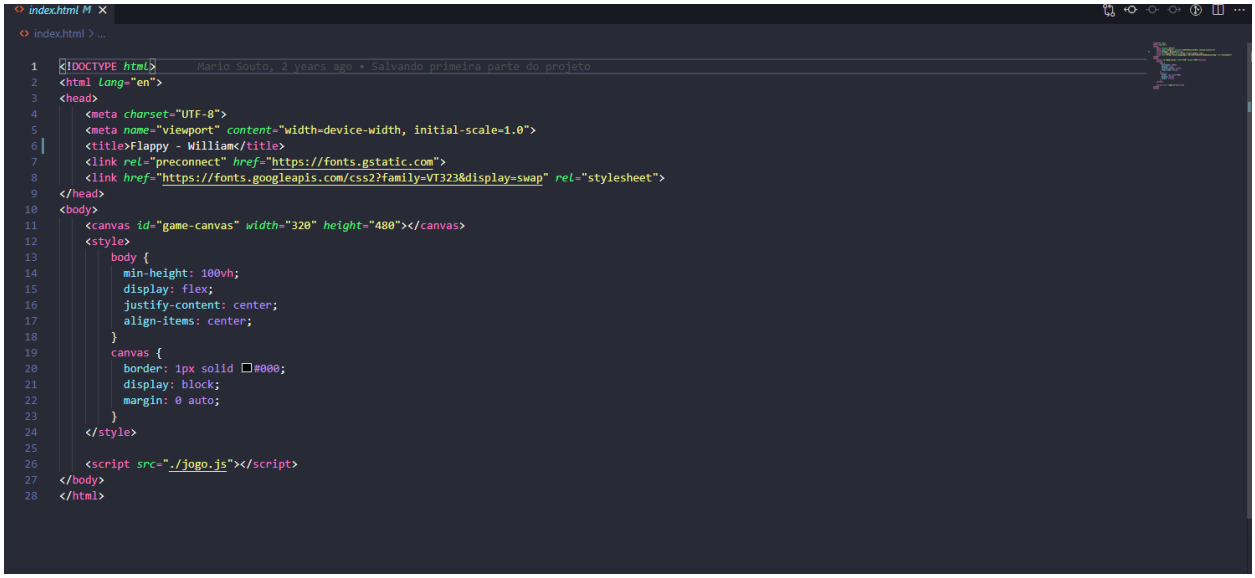

Figura 11: Estrutura HTML do jogo

Ainda no HTML inclui um CSS, que contém uma borda para diferenciar a área do jogo, e também criou-se um arquivo “jogo.js”, onde o mesmo está importado no HTML. Por fim dentro da pasta do jogo tem o arquivo referente as imagens que é o sprite, que contém todas as imagens que serão utilizadas no jogo.

6.6.1 Senário e Movimentos

Para início, no arquivo jogo.js foi incluído uma variável que recebe uma propriedade “new image()”, que cria via javascript uma imagem dentro da memória do computador e onde associo uma url para ela, assim podendo carregar nossas sprites. Logo após eu selecionei nossa tag do canvas que está no arquivo index.html, e com isso eu determinei que o jogo será dois 2d.

Com a configuração básica já feita iniciei a parte de incluir o personagem na tela, isso foi feito usada a função drawImagem do canvas, passando o mínimo ou o máximo de parâmetros, aqui no meu caso utilizei o máximo, pois como nos temos um arquivo que contem todas as imagens, vou ter que buscar dentro do arquivo, qual imagem que eu vou querer, qual tamanho vou querer desenhar na tela e pontos de posicionamento dentro da tela de desenho, ou seja, do canvas.

No desenvolvimento foi utilizada como base a documentação do MDM Web Docs, na a utilização das propriedades do canvas, pois precisou pegar o tamanho da imagem e o espaço que a mesma vai utilizar na tela.

Syntax

```
void ctx.drawImage(image, dx, dy);
void ctx.drawImage(image, dx, dy, dWidth, dHeight);
void ctx.drawImage(image, sx, sy, sWidth, sHeight, dx, dy, dWidth, dHeight);
```

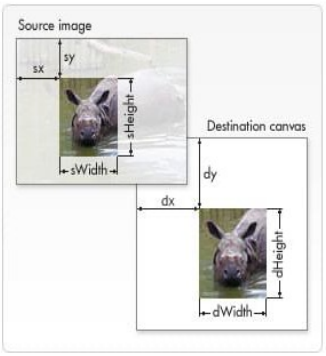

Figura 12: Representa as informações da documentação do canvas para selecionar as imagens

Para buscar as informações referentes ao tamanho da imagem e o espaço que ela vai ocupar na tela, utilizei as ferramentas do Gimp, como podemos ver na Figura 13.

Figura 13: Apresenta as informações de tamanho da imagem dentro do arquivo, conforme seleção

Com isso podemos determinar qual imagem vamos pegar dentro do arquivo de imagem que contém todas as imagens, assim qual o tamanho ela vai ter e assim também o lugar que vai ser apresentada.

Logo após precisou ser incluída uma função "*loop*" para que a nossa imagem seja mostrada na tela, isso acontece por que dentro de um jogo ele não desenha as

coisas uma vez só e sim várias vezes, onde nada mais é do que o FPS, ou *Frames Per Second*, que são os vários quadros da tela a cada segundo, no caso aqui um padrão é Figura 12: Representa as informações da documentação do canvas para selecionar as imagens Figura 13: Apresenta as informações de tamanho da imagem dentro do arquivo, conforme seleção 60 quadros por segundos, podendo até ser mais, mas aqui no nosso trabalho está sendo usado 60. E dentro da função *loop* inclui outra função, que por sua vez vai ser a responsável por criar os quadros.

```
function loop() {

  telaAtiva.desenha();
  telaAtiva.atualiza();
  // som_Trilha.play();//William

  frames = frames + 1;
  requestAnimationFrame(loop);
}
```

Figura 14: Apresenta a função loop(infinito), na sua estrutura dentro do código fonte

Na sequência, já com a imagem do personagem aparecendo na tela, o mesmo ainda não tem vida, ou seja, não tem movimentos, nesse caso inclui uma estrutura para que o mesmo possa se movimentar. Para fazer levamos em consideração que teríamos que fazer para cada elemento de sprite várias funções, o que poderia acarretar em uma desorganização do código fonte.

Diante dessa questão foi necessário criar uma estrutura onde para cada elemento que vamos precisar(avatar, chão, canos, etc), foi criado um objeto, onde dentro desse objeto, vamos receber as informações do tamanho do sprite, assim como

comentado anteriormente e ainda, o seu movimento e as possíveis atualização, ficando da seguinte forma.

```
// [Chao]
function criaChao() {
  const chao = {
    spriteX: 0,
    spriteY: 610,
    largura: 224,
    altura: 112,
    x: 0,
    y: canvas.height - 112,
    atualiza() {
      const movimentoDoChao = 1;
      const repeteEm = chao.largura / 2;
      const movimentacao = chao.x - movimentoDoChao;

      // console.log('[chao.x]', chao.x);
      // console.log('[repeteEm]',repeteEm);
      // console.log('[movimentacao]', movimentacao % repeteEm);

      chao.x = movimentacao % repeteEm;
    },
    desenha() {
      contexto.drawImage(
        sprites,
        chao.spriteX, chao.spriteY,
        chao.largura, chao.altura,
        chao.x, chao.y,
        chao.largura, chao.altura,
      );

      contexto.drawImage(
        sprites,
        chao.spriteX, chao.spriteY,
        chao.largura, chao.altura,
        (chao.x + chao.largura), chao.y,
        chao.largura, chao.altura,
      );
    },
  };
  return chao;
}
```

Figura 15: Apresenta a estrutura que foi usada para criação do jogo no javascript

Seguindo com a utilização da estrutura que organizamos para criar as funções, implementamos o chão e o plano de fundo no jogo. Para essas duas partes foi preciso duplicar a exibição das imagens, ou seja, mostrar duas vezes as duas, com isso dando

a impressão de uma imagem maior e continua. Ainda para finalizar a parte do cenário, colocamos uma cor no fundo no céu com uma cor mais escura para combinar com o contexto, ficando conforme Figura 16.

Figura 16: Apresenta elementos do senário do jogo, como fundo, personagem, chão e céu

Com o cenário pronto, vamos partir para a parte de movimentação do avatar, para isso seguindo a questão de organização, dentro do nosso objeto que se refere ao

personagem colocamos, mais um atributo do mesmo tipo do desenho, mas agora para atualizar. Isso vai fazer com que o personagem tenha a sua queda dentro do espaço da tela.

Para complementar essa queda foi criada um novo atributo que se chama velocidade que por sua vez vai ir aumentando, e com isso coloquei mais um atributo chamado gravidade, que vai servir para puxar e aumentar a velocidade, para o avatar cair. Com isso ficou conforme Figura 17.

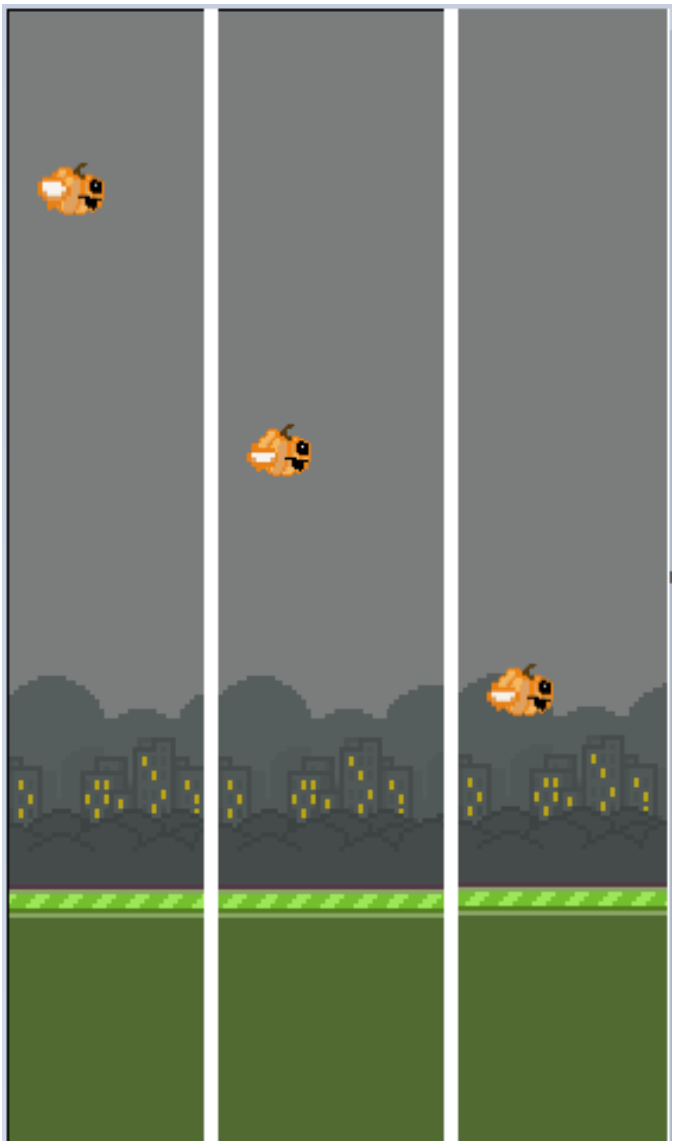

Figura 17: Apresenta os movimentos de queda do personagem

6.6.2 Posicionamento de tela e Movimentos

Já com o cenário desenvolvido, iniciou-se a parte de movimentos, então ajustei de início o movimento do avatar, para que o mesmo tenha seu movimento, ou seja, que ao cair selecionando na tela com o click o avatar pule, ou seja, vá para cima para que durante o jogo consigamos conduzir o avatar. Para fazer essa implementação basicamente pegamos a velocidade e subtrair com o valor do pulo que nos vamos dar.

A nossa velocidade sempre está sendo incrementada com isso se nos diminuir com o valor do pulo que terá um valor inicial vamos ter o pulo. Podemos ver na Figura 18 o resultado.

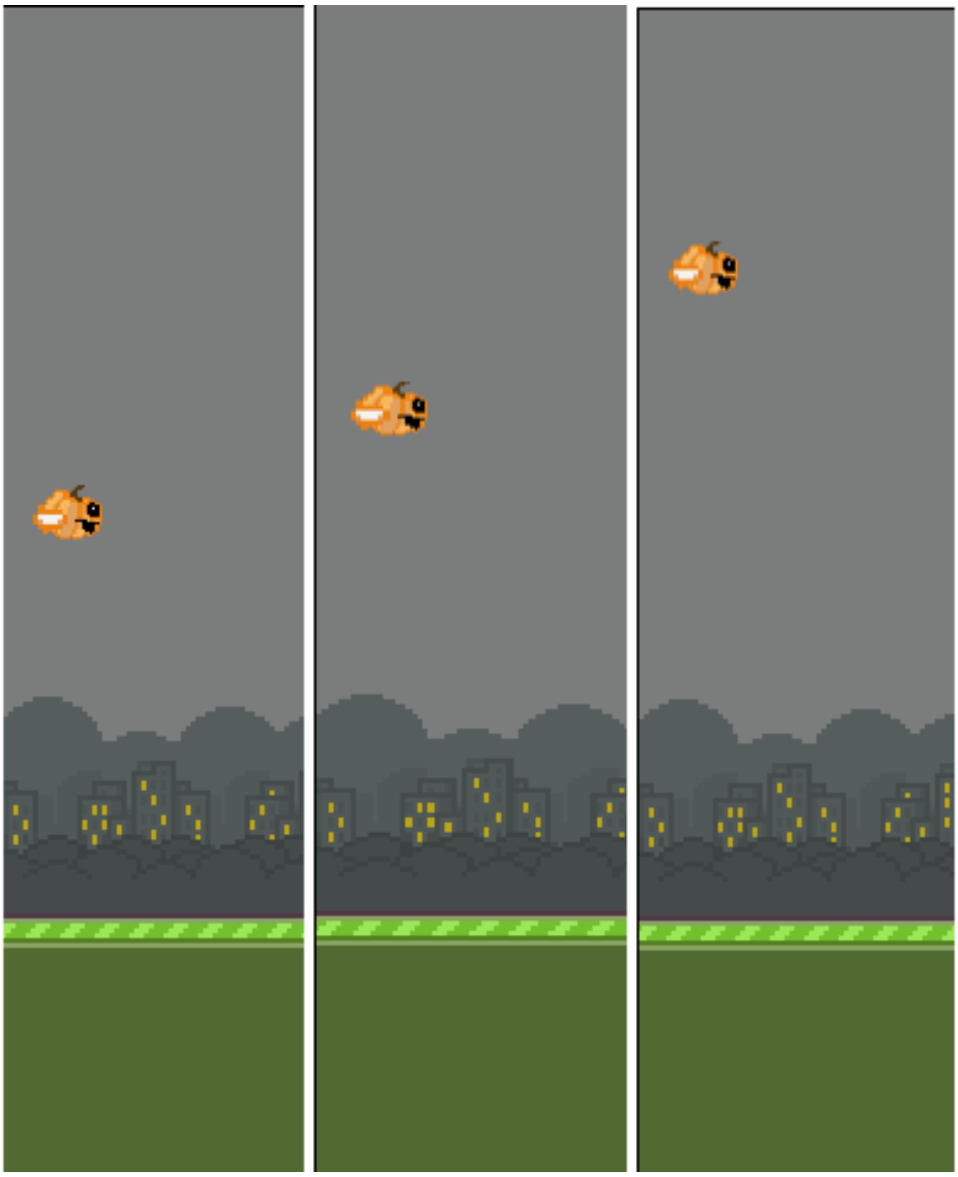

Figura 18: Apresenta o movimento do pulo do personagem

Agora com o movimento do avatar funcionando, iniciei a parte de colisões, para isso seguimos o mesmo padrão de organização, e criei uma nova “função fazColisao()”, que vai receber as informações do avatar e do chão. Ainda para que seja feito a colisão certa, temo que fazer a subtração da posição do avatar menos a altura do mesmo for maior ou igual ao chão, nos mostra que eles se colidiram.

Ainda o personagem quando sofrear a queda e o usuário fazer o click deverá voltar para o início da partida, e com isso foi feito a limpeza das variáveis para que fiquem zeras, e assim o jogo e o avatar volte para o seu ponto inicial.

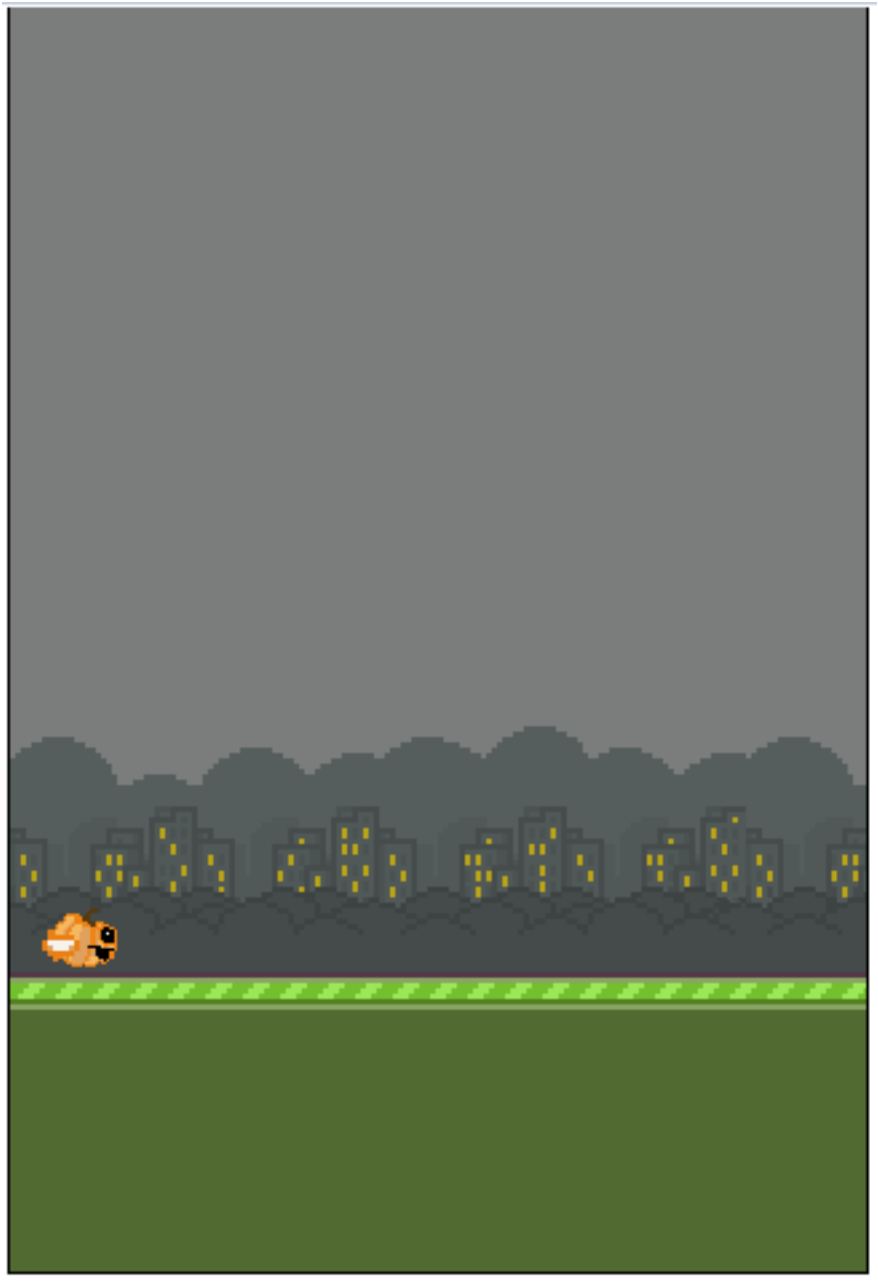

Figura 19: Apresenta a colisão do personagem com o chão do jogo

Com o avatar em movimento, ou seja, caindo e pulando, vamos para a parte de movimento do chão, pois precisamos que o mesmo se movimento conforme evolução dentro do jogo. Aqui conforme todo o projeto usamos a mesma organização, então

Coloquei uma nova função criaChao, para que possamos fazer com que a mesma seja atualizada e assim apresentada no decorrer do jogo. Para Faze o movimento do chão, usamos a função atualiza, para que a mesma possa atualizar e apresentar o chão continuamente, para isso foi feito um cálculo da largura do chão dividido por 2, e ainda foi pego o resto da divisão, usando o operador lógico %, que vai servir para sabermos em que momento da imagem vamos repetir a mesma, podemos ver na Figura 20 como ficou o resultado

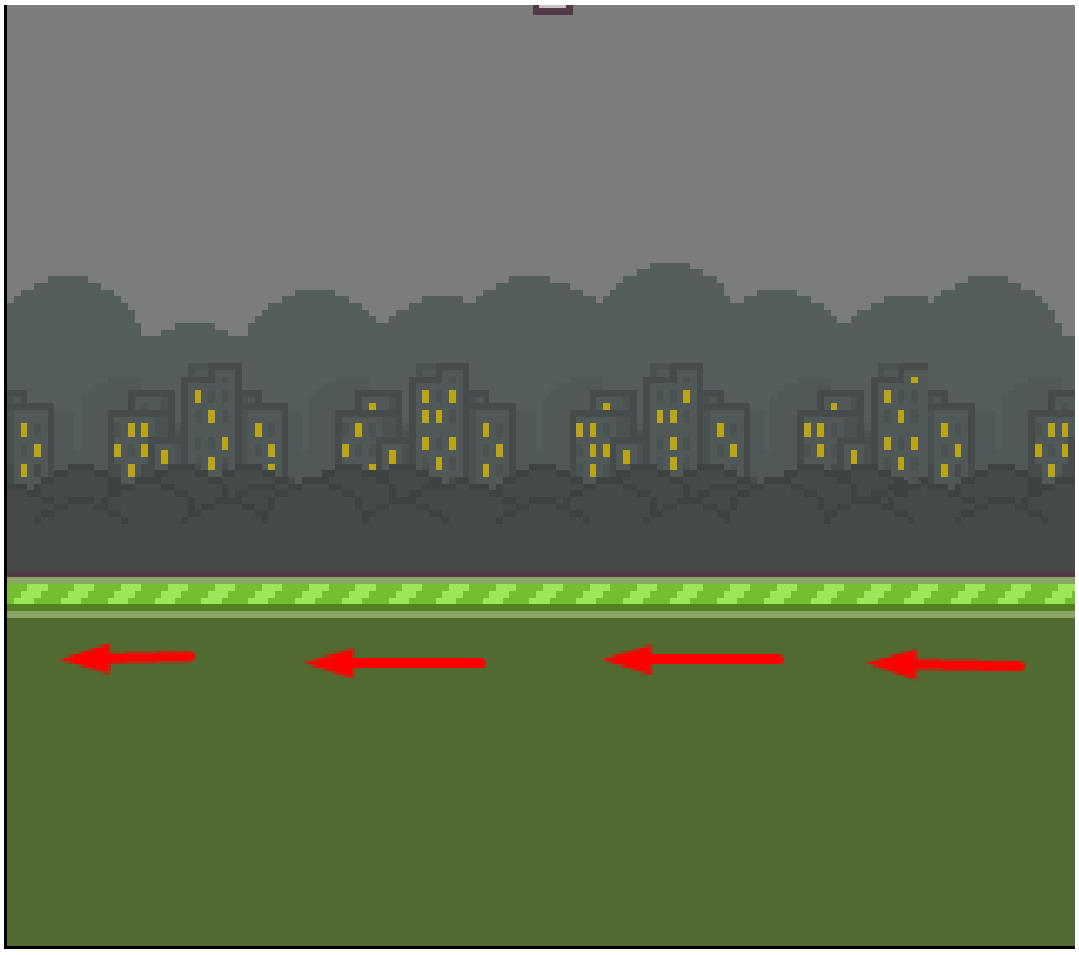

Figura 20: Apresenta o movimento do chão do jogo

Para finalizar os movimentos, foi feito o movimento do avatar, ou seja, o movimento do bater das assas, que indica e representa que o avatar está voando. Aqui

já existe o avatar, ou seja, já incluimos o mesmo anteriormente, porém o movimento vai envolver três imagens do sprite. Para fazer esse movimento foi criado um movimento para cada imagem, onde foi preciso fazer uma desestruturação do objeto para que possamos adicionar todos os movimento em um objeto só e assim fazer o movimento.

Ainda foi preciso fazer o movimento contínuo do bater das assas, ou seja, dentro dos quadros do jogo, para que o avatar movimente continuamente precisei cria uma função para atualizar o *frame*, com base em como o jogo está renderizado. Para fazer essa atualização do *frame*, precisos buscar o *frame*, como até aqui não usei, tive que criar uma variável, que vai conter a informação do *frame* com valor 0 no início da aplicação, e para cada *loop* completo quando atualizamos o jogo vamos somar, a variável *frame* com seu valor mais 1.

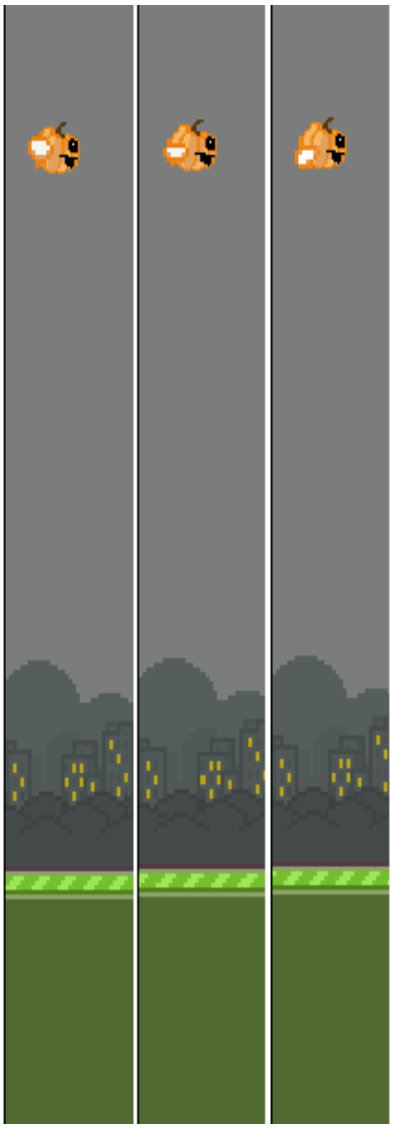

Figura 21: Apresenta o movimento de bater asas do personagem

6.6.3 Tela Inicial do Jogo

Para a tela Inicial usamos a mesma estrutura que organizamos para criar o avatar, assim como outros, para a tela do início. Ficando da seguinte forma.

Dentro de uma organização, resolvi criar um padrão para as telas, onde criei um objeto para colocar as telas, ou seja, todas as telas início e fim, entre outras que o jogo pode ter, nesse caso vão ser colocadas aqui dentro. Isso vai servir para que fique organizado mas também para que tenha uma estrutura para que sejam apresentadas nos momentos corretos durante o jogo.

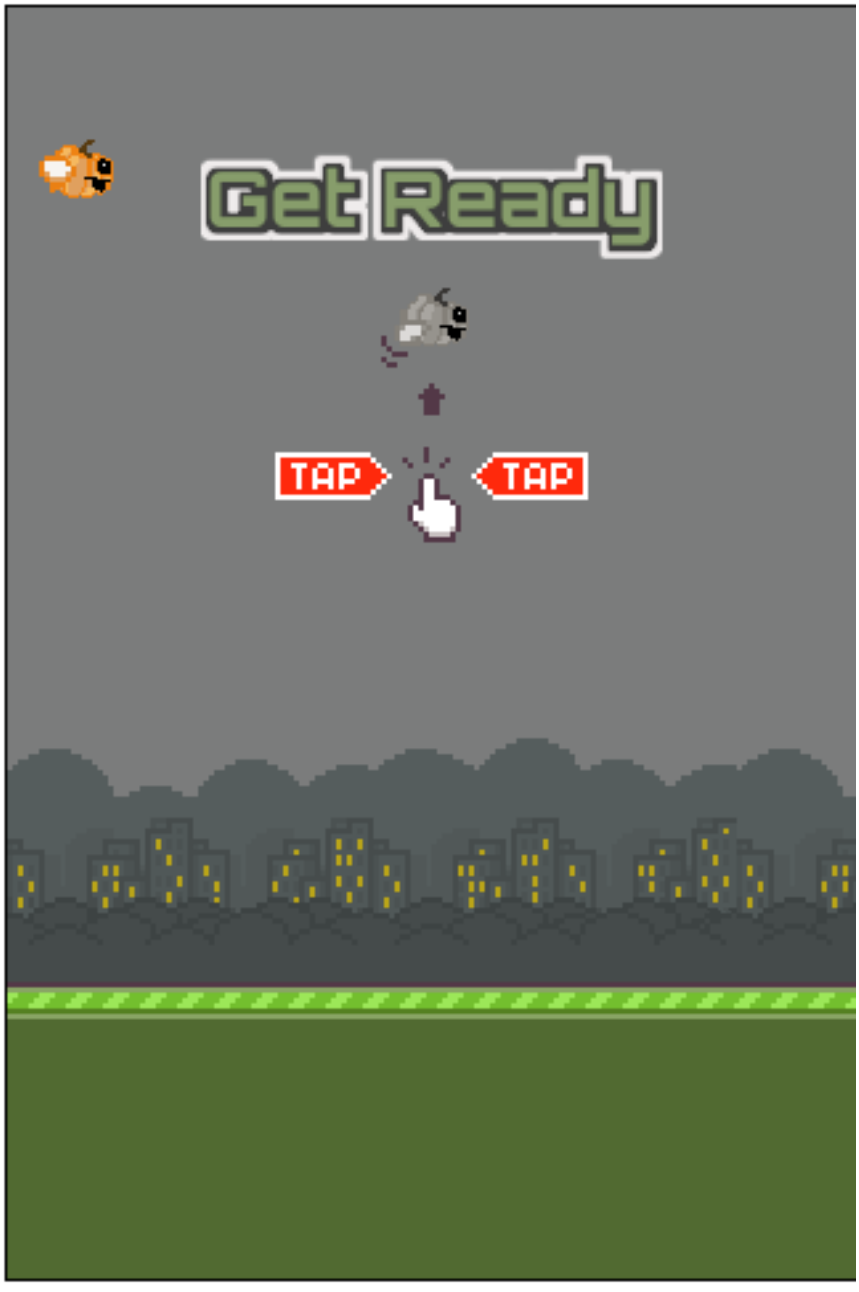

Figura 22: Apresenta tela inicial do jogo

Ainda nessa parte foi preciso fazer a ação, ou seja, quando o usuário selecionar na tela, assim o jogo tem quer ser iniciado apresentando a tela do início apenas quando reiniciar o jogo.

Figura 23: Apresenta a tela inicial do jogo apos o inicio do mesmo

Para fazer o click, foi criada uma função onde a mesma verifica se foi feito um click na área do jogo, usando a propriedade *window.addEventListener* passando click. Ainda foi usado uma condição verificando quando o click é apenas para o início do jogo, para que não tenho conflito com o restante do jogo, pois o click vai servir para movimentar o avatar.

6.6.4 Obstáculos no Cenário

Para incluir os obstáculos na tela usamos a mesma característica de buscar a imagem no sprite, com auxílio do Gimp, igual usamos para as demais, imagens. Também seguindo a organização do jogo coloquei uma nova função que vai fazer o desenho dos canos assim como a distribuição dos mesmos durante o jogo. Para fazer essa dinâmica de aparecer os canos aleatoriamente, fizemos com que os mesmos sejam apresentados em duplas, um na parte de cima da tela e outro na parte de baixo da tela, e com isso duplicar o cano.

Já com a função criada para os dois canos, onde até então os canos estão apenas sendo apresentados na tela, ou seja, só tem um par de canos sem a lógica de criá-los aleatoriamente e sem o espaço entre os dois. Na sequência, criei o espaçamento entre os dois canos, e aqui desenvolvi uma lógica para que os canos tenham um tamanho aleatório durante o percurso do jogo, onde criei uma variável que vai receber um valor de entrada e esse valor vai ser somado com o lugar que os canos vão aparecer na tela, fazendo com que os canos tenham seu tamanho alterado, ou seja, os canos possuem um espaçamento fixo entre eles, de cima para baixo, e aqui eles vão subir e descer conforme o valor da soma da variável com o lugar da imagem na tela, podemos verificar na Figura 24 o resultado.

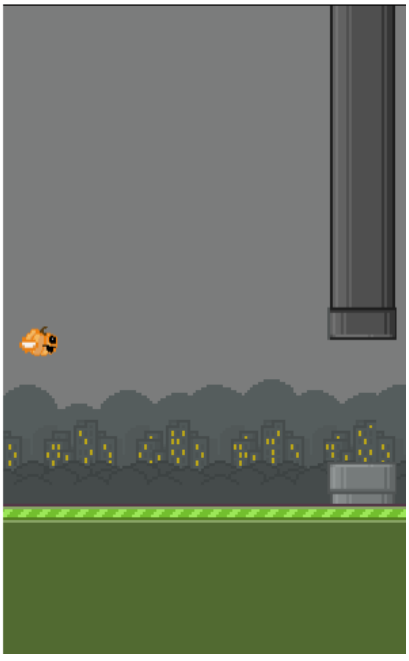

Figura 24: Apresenta os obstáculos do jogo, que são os canos

Com a lógica dos canos criadas, foi a hora de criar vários canos e mostrar os mesmos na tela, conforme decorrer do jogo. Para isso dentro da função atualiza, coloquei uma lista de canos que serão apresentados na tela, onde peguei o resultado da divisão do *frame* atual por 100 que vai dar um resultado igual a 0, pois a cada 100 *frames* que passar eu vou criar o par de canos com tamanho diferente obedecendo o espaçamento interno. Podemos ver o resultado na Figura 25.

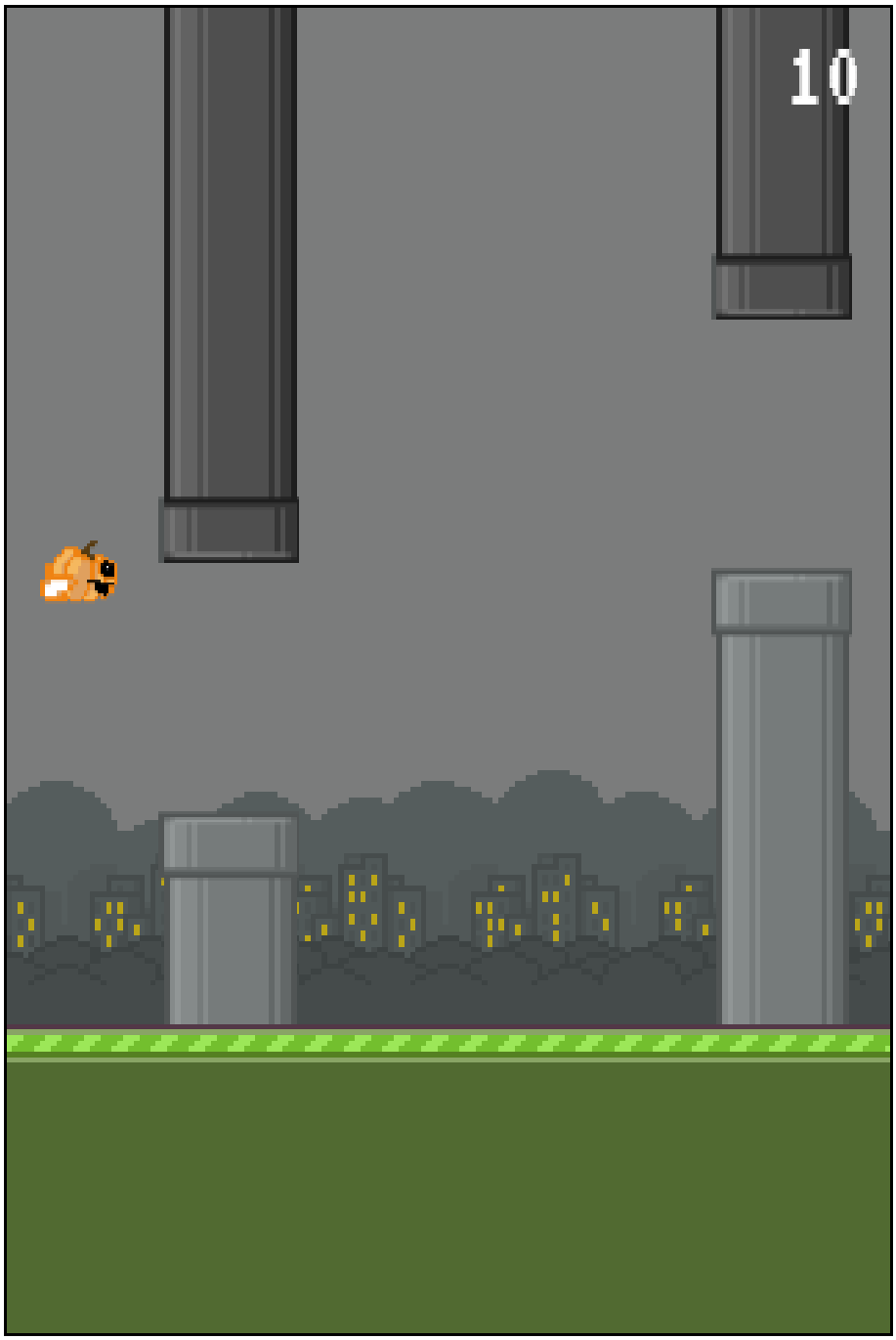

Figura 25: Representa a dinâmica que apresenta automaticamente os canos na tela.

Logo na sequência, desenvolvi o movimento dos canos, onde os canos vão aparecer um ao lado do outro conforme o movimento da tela. Ainda por questões de performance, incluir um ajuste para limpar as informações dos canos que são criadas infinitamente, assim não sobrecarregando a memória do jogo.

Com o movimento dos canos já feito, foi a hora de fazer a parte de colisão do avatar com os canos. Nesse ponto para cada cano, ou dupla de canos, foi criada uma verificação, e dento da verificação está sendo chamada uma função que verifica se tem colisão. Na função está sendo retornado um *booleano* se é verdadeiro(*true*) ou falso(false), para isso estou verificando a posição do cano na tela e se o avatar está igual ou maior a ele. Com isso temos a colisão, podemos verifica na Figura 26.

Figura 26: Apresenta a colisão do personagem com os canos

6.6.5 Fim de Jogo (*Game Over*)

Na tela final do jogo foi utilizada a mesma prática usada para buscar a imagem no sprite, utilizado o Gimp, assim como as demais imagens durante o desenvolvimento. Essa imagem vai ser apresenta quando o avatar colidir com alguma coisa, sendo disparado o fim do jogo(*game over*), e com isso fazendo com que o jogo seja reiniciado, zerando as pontuações. Podemos observar na sequência na Figura 27 o resultado.

Figura 27: Apresenta o fim do jogo

6.6.6 Pontuação e Medalhas

Na sequência foi implementado um s*core*, ou seja, a pontuação, essa pontuação vem dos *frames*, assim como foi feito em outros casos, ele busca dos *frames* um valor conforme atualização dos quadros, assim conseguimos recuperar esse valor e utilizar como pontuação. Dessa forma conforme vai passando pelos Figura

27: Apresenta o fim do jogo obstáculos vai aumentando a quantidade de *frames* e assim mostramos isso com a pontuação do jogo. Vejamos na Figura 28 o resultado.

Figura 28: Apresenta a pontuação do jogo

A pontuação(s*core*) está sendo mostrada em tempo real no canto superior direito, e também ao final do jogo no campo *score*. Logo após fazer a pontuação, desenvolveu-se a parte das medalhas, que são classificadas, como, branca, bronze, prata e ouro. As medalhas seguem um padrão que Figura 28: Apresenta a pontuação

do jogo tem como principalmente informação a pontuação que desenvolvemos anteriormente. Nesse caso são mostradas as medalhas conforme evolução com base no *score*, onde o usuário atingindo menor ou igual a 5, apresenta a medalhinha branca, menor igual a 25 apresenta a medalha de bronze, menor igual a 50 apresenta medalha de prata, e por fim, ficando menor igual a 100 apresenta a medalha de ouro. Como é uma versão teste(demo) foram colocados fixos os valores, a ideia e aperfeiçoar para que tenha um ranking conforme os jogadores jogam e vão evoluindo, mas por hora foi desenvolvido assim. Ainda, para a busca da imagem no sprite foi feito a mesma busca feita no avatar, buscando o tamanho que vai usar na tela e o tamanho do elemento. Podemos verificar na Figura 29 o resultado final.

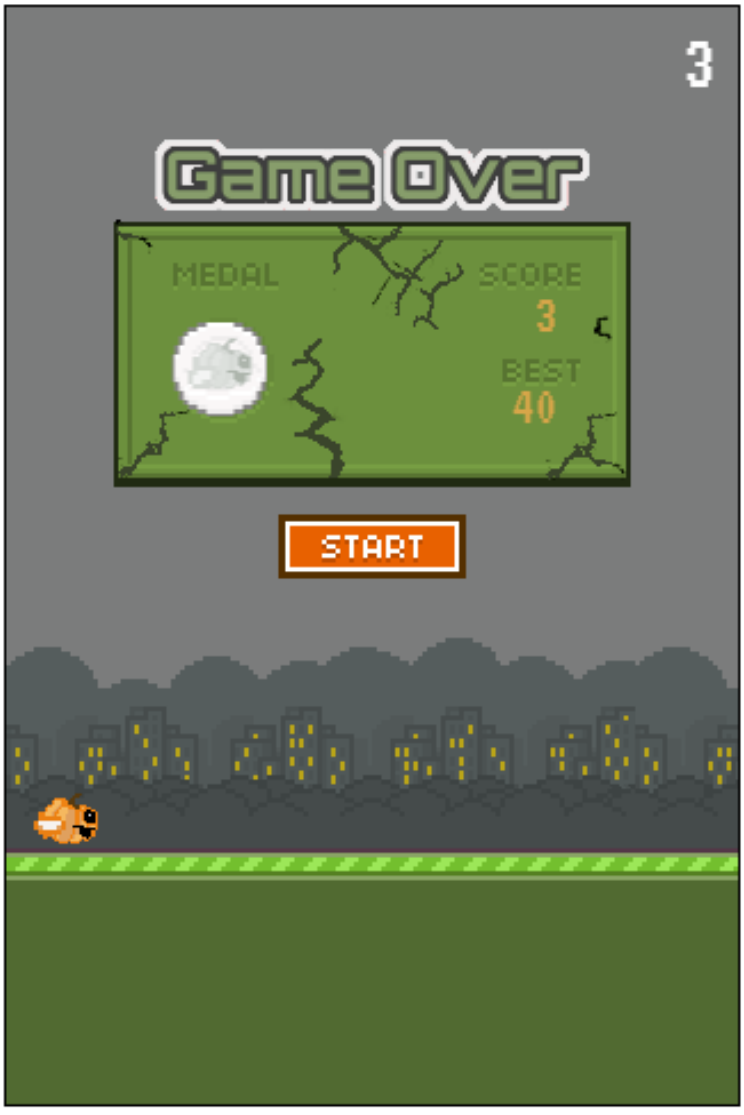

Figura 29: Apresenta a tela de final do jogo com a medalha e as pontuações

Algo que foi um bônus foi a inclusão de som no jogo, da mesma forma foi feito uma busca na internet em sites com sons para jogo, que tem sua característica livre, então foi pego uma música que tem uma característica de suspense, para trilha que é tocada durante o jogo, e um som de impacto, que é para a colisão do personagem, que simula essa colisão.

7 Conclusão

Apesar de não ser um tema novo, e que já estamos habituados a ouvir falar e desenvolver sobre jogos eletrônicos, é sempre um desafio propor um trabalho relacionado a esse tema. Aqui nesse trabalho, procurei estudar e aplicar conceitos do dia a dia de programação, utilizando HTML CSS e javascript, para que eu pudesse construir um jogo eletrônico pois, consegui mesclar um pouco da experiência de desenvolvedor e estudando do curso de sistemas para internet, para desenvolver o jogo da melhor forma possível.

O jogo por sua vez é simples pois se trata de um demo, apresentando senário com imagens e corres neutras, claro que dentro da ideia do tema que é o *halloween*, mas que possibilita ao usuário a melhor forma de entretenimento possível sem ter muita preocupação em saber jogar ou ter que procura instruções, pois o mesmo é bem intuitivo e prático de jogar.

Com tudo o desafio foi agregador, aprendi conceitos novos, apliquei conceitos que já dominava e aproveitei para dar aquela jogadinha durante os testes, pois ninguém é de ferro. Estou feliz com o resultado, ainda podendo no futuro dar continuidade no projeto, pois essa é apenas uma versão demo, assim podendo ser melhorado no futuro, caso eu queira, mas dentro da proposta de desenvolver um jogo usando HTML, CSS e javascript, saio contente com o que foi desenvolvido, o jogo ficou funcional e prático, atendendo aos objetivos de um jogo eletrônico para o entretenimento.

7 BIOGRAFIA

AMORIN, A. **A origem dos jogos eletrônicos**. USP, 2006.

BATISTA, Mônica de Lourdes et al. **Um estudo sobre a história dos jogos eletrônicos**. Revista Eletrônica da Faculdade Metodista, 2007, p. 0377. Disponível em:<http://re.granbery.edu.br/artigos/MjQ4.pdf > Acesso em 08 novembro 2018.

BATISTA, Mônica de Lourdes et al. **Um estudo sobre a influência dos jogos eletrônicos** sobre os usuários. Revista Eletrônica da Faculdade Metodista, 2008, p. 0377. Disponível em:<http://re.granbery.edu.br/artigos/MTM4.pdf> Acesso em 08 novembro 2018.

BORBA, F.E. **AJAX: guia de programação**. São Paulo: Erica, 2006.

DAVIS, Alan M.; HSIA, Pei. Giving voice to requirements engineering, IEEE Software,v. 11, n. 2, 1994.

DEVMEDIA. **Artigo Engenharia de Software 10 – Documento de Requisitos.** 2009. Disponível em: <https://www.devmedia.com.br/artigo-engenharia-de-software 10-documento-de-requisitos/11909> Acesso em: 29/06/2021.

DOCTORZEROTH. **SPRITE – GAME CONCEITO.** Disponível em <https://doctorzeroth.wordpress.com/2009/12/25/sprite-game-conceito/> Acesso em: 10/06/2021

KEITH, Clinton. **Agile Game Development With Scrum**. New Jersey: Pearson Educational, 2010.

LOPES, Leandro T. et al. (2003). **Uma proposta para o processo de requisitos em ambientes de desenvolvimento distribuído de software,** In: WER 2003 – VI Workshop em Engenharia de Requisitos, São Paulo, Brazil.

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

SILVA, M.S. **HTML 5 A Linguagem Que Revolucionou a Web**. 2. ed. São Paulo:Novatec, 2014.

W3C. **HTML e CSS**. Disponível em: <https://www.w3.org/standards/webdesign/htmlcss> Acesso em: 13/04/2020.

ZAKAS, Nicholas. **Professional JavaScript for Web Developers** - 3° edição. Janeiro, 2012.