Universidade Federal do Rio Grande do Norte – UFRN
Centro de Ensino Superior do Seridó – CERES
Departamento de Computação e Tecnologia – DCT
Bacharelado em Sistemas de Informação – BSI

Desenvolvimento de um Criador de Mapas Multi-Plataforma para Jogos Digitais

Pedro Jonas da Silva Medeiros

Orientador: Prof. Dr. João Paulo de Souza Medeiros

Trabalho de Conclusão de Curso apresentado ao Curso de Bacharelado em Sistemas de Informação como parte dos requisitos para obtenção do título de Bacharel em Sistemas de Informação.

Laboratório de Elementos do Processamento da Informação – LabEPI
Caicó, RN, 20 de julho de 2023

Universidade Federal do Rio Grande do Norte - UFRN
Sistema de Bibliotecas - SISBI
Catalogação de Publicação na Fonte. UFRN - Biblioteca Setorial Profª. Maria Lúcia da Costa Bezerra - -CERES- - Caicó

Medeiros, Pedro Jonas da Silva.
Desenvolvimento de um Criador de Mapas2 Multi-Plataforma para Jogos Digitais / Pedro Jonas da Silva Medeiros. - Caicó, 2023.
44f.: il.

Monografia (Graduação) - Universidade Federal do Rio Grande do Norte, Centro de Ensino Superior do Seridó, Departamento de Computação e Tecnologia, Sistemas de Informação. Caicó, RN, 2023.
Orientação: Prof. Dr. João Paulo de Souza Medeiros.

1. Multi-Plataforma - Monografia. 2. criação de mapas - Monografia. 3. Jogos digitais - Monografia. 4. Jogos 2D - Monografia. 5. Desenvolvimento de Jogos - Monografia. I. Medeiros, João Paulo de Souza. II. Título.

RN/UF/BS-CERES CDU 004.42

Elaborado por Giulianne Monteiro Pereira Marques - CRB-15/714

Resumo

Este trabalho descreve a proposta de desenvolvimento de um criador de mapas bidimensionais. No desenvolvimento da maioria dos jogos, uma das partes mais demoradas é a criação de mapas. Portanto, uma ferramenta produtiva com o intuito de reduzir esse tempo pode ajudar.

O usuário cria mapas maximizando a praticidade, seja para utilizá-lo em seus projetos, para aprender ou ensinar programação de jogos.

Palavras-chave: Criador de Mapas; Jogos 2D; Desenvolvimento de Jogos; Top View; Multi-Plataforma; jogos digitais;

Abstract

This work describes the proposal for the development of a two-dimensional map creator. In the development of most games, one of the most time-consuming parts is map creation. Therefore, a productive tool aimed at reducing this time can be helpful.

The user creates maps, maximizing convenience, whether for use in their projects or for learning and teaching game programming.

Keywords: Map Creator; 2D Games; Game Development;

Sumário

Lista de Figuras iii

Glossário v

1 Introdução 1
1.1 Contextualização e Problema . . . 1
1.2 Objetivos . . . 1
1.2.1 Objetivos Específicos . . . 2
1.3 Delimitação do Estudo . . . 2
1.4 Justificativa . . . 2

2 Fundamentação Teórica 5
2.1 O que é um jogo? . . . 5
2.2 Jogos Digitais . . . 6
2.3 Ferramentas Relacionadas . . . 7
2.3.1 Tiled . . . 7
2.3.2 OGMO Editor . . . 7
2.3.3 Pymapper . . . 7
2.3.4 LDtk . . . 8

3 Desenvolvimento 9
3.1 Proposta de Solução . . . 9
3.1.1 Sistema Proposto . . . 9
3.1.2 Requisitos Funcionais . . . 10
3.1.3 Requisitos Não Funcionais . . . 11
3.2 Protótipo . . . 11

4 Conclusões 17
4.1 Resultados . . . 17
4.2 Trabalhos futuros . . . 18

5 Apêndice 21
5.1 Detalhando exemplo simples da Wiki . . . 21

Referências Bibliográficas 27

Lista de Figuras

3.1 Tela Inicial 11
3.2 Tela Primária 12
3.3 Tela de Configuração. 13
3.4 Chãos Selecionados. 13
3.5 Salvar Construção. 14
3.6 Aba Construções. 14

5.1 Andar de baixo da casa 22
5.2 Andar de cima da casa 22
5.3 Andar de baixo da casa segundo conjunto 23
5.4 Andar de baixo com Propriedades de Evento 24
5.5 Andar de cima da casa segundo conjunto. 25

Glossário

Acrônimos

JSON ..... *JavaScript Object Notation*
XML ..... *Extensible Markup Language*

Simbologia

Definições

Sprite ..... Imagem ou conjunto de imagens utilizados para representação gráfica dentro de um aparelho com interface visual.

Tiles ..... Um retângulo de qualquer tamanho, um bloco. Em alguns jogos, a movimentação dos jogadores é contadas em quadrados, cada quadrado representa um Tile.

TXT ..... Formato simples de arquivo de computadores, utilizado como um bloco de notas qualquer.

Portal ..... Jogo digital onde o jogador deve resolver vários quebra-cabeças para progredir no jogo.

Engine ..... Em português, Motor de jogos, é um sistema totalmente integrado, onde é modelada a I.A, a jogabilidade e movimentação espacial, os personagens e ambientes de um jogo digital, já contando com a parte gráfica.

NPC ..... *Texto*, em português, personagem não Jogável. Seria um personagem em um jogo que não pode ser controlado, como um comerciante, por exemplo.

Capítulo 1

Introdução

"If knowledge can create problems,
it is not through ignorance that we can solve them."
Isaac Asimov

1.1 Contextualização e Problema

Existem diversas ferramentas para criação de jogos atualmente e, dentre elas, várias que não necessitam de conhecimento algum de programação, onde você simplesmente pode arrastar um componente para uma entidade e uma nova funcionalidade de movimento foi adicionada ao jogo. Ferramentas como essas, apesar de esconder o que estão fazendo, impedindo assim o usuário de entender a lógica que está sendo processada ali, ainda podem ser utilizadas no ensinamento de lógica.

Neste cenário, essas ferramentas podem ser importantes aliadas na busca de soluções para o problema de aprendizagem, inclusive no ensino de programação. Segundo Medeiros (2014), "Aprender a programar ensina as pessoas a como pensar por meio do pensamento lógico e criativo".

Nessas ferramentas, é simples escolher um objeto, como, por exemplo, o jogador, e dizer que ao colidir com outro, como, por exemplo, uma moeda, o segundo objeto desaparecer; É fácil ensinar quando já se tem tudo pronto, mas o que exatamente seria uma colisão? O que realmente está acontecendo.

Esse seria um exemplo simples de algo escondido por essas ferramentas, a simples ação de pegar um quadrado e checar se um está dentro de outro está presente, mas não evidente. "Não sei o que fiz, só sei que foi assim".

1.2 Objetivos

Este trabalho propõe a criação de um construtor de mapas para projetos de jogos, seja para aqueles que desejem desenvolver jogos, ou para aqueles que já estão desenvolvendo um e desejam uma forma simples de criar o mapa. Uma forma de salvar/carregar mapas para ser utilizados em outros diversos projetos que podem, ou não né, ser utilizados como referência para novos.

Dessa forma, aqueles que já tem algo preparado podem ter uma ajuda na criação de um mapa para seu jogo, visto que será possível encontrar mapas prontos pela internet,

ou mesmo construções para preencher o seu mundo, além da forma facilitada de criá-lo, o que entrega ao programador uma forma de melhorar os jogos e deixando-os mais bonitos com um mapa mais animado e configurável.

1.2.1 Objetivos Específicos

O Construtor de mapas deve conseguir criar mapas personalizáveis, pedaço a pedaço, como se fosse um quebra cabeça, sem tamanho definido, podendo variar conforme a preferência do usuário.

Também deve conseguir exportar os mapas com seu próprio modelo, simples de ser lido pelos usuários para os mesmos poderem criar plug-ins de importação em outros motores de jogos ou criar um programa completo para importar e começar a criar um jogo.

1.3 Delimitação do Estudo

O projeto será desenvolvido com Java no Eclipse IDE e nele deverá ser possível colocar imagens, obstáculos, paredes, criar eventos de interação, estruturas (cidades, casas), etc. Além de ser possível construir mapas completos dentro dele, será possível criar pequenas ou grandes construções que podem ser reutilizadas em vários outros mapas e compartilhá-las de forma simples. Dessa forma, a comunidade de usuários poderia compartilhar suas ideias de forma simples e interativa, podendo trabalhar em conjunto com canais simples de texto.

Visando desenvolver um construtor de mapas que facilite na programação de jogos sem utilização de qualquer motor de jogos, além de ser possível salvar o mapa criado e poder editá-lo várias vezes, deve ser possível exportá-lo de forma que qualquer linguagem compreenda o que está escrito, como uma exportação em JSON.

A linguagem java foi escolhida devia a familiaridade do autor, onde o mesmo já desenvolveu alguns jogos utilizando a linguagem e possui um certo domínio para realizar a tarefa.

Apesar de ser possível a criação de diversos jogos, até a conclusão do presente trabalho, apenas jogos top-view foram criados e testados.

1.4 Justificativa

Apesar de existirem várias opções para criação de mapas e de jogos, como Game-Maker, Unity, Godot, unreal, dentre outras. Nenhuma dá a liberdade de exportar seu mapa para qualquer programa conseguir usá-la de forma simples e prática. Mesmo existindo a possibilidade de programar dentro de alguns dos criadores de jogos, não há a liberdade de escolha da linguagem qualquer.

Existem diversas razões que uma pessoa possa querer aprender a criar os jogos, uma delas seria um suposto “potencial motivacional” que eles têm frente aos jovens.

É fato que muitos daqueles em idade escolar se interessam por videogames e, por conta desse sentimento, os jogos têm sido visto como forma para engajar estudantes a participarem das atividades escolares regulares, ou mesmo para recuperar alunos desinteressados. (De Paula et al., 2016)

Visando exportar os mapas para qualquer programador importar, dessa forma, o usuário terá uma base simples para a criação de jogos com o foco no mapa, o que facilita até mesmo na vontade de aprender a criar algo, visto que existe um desafio a frente.

Capítulo 2

Fundamentação Teórica

"We can only see a short distance ahead,
but we can see plenty there that needs to be done."
Alan Mathison Turing

2.1 O que é um jogo?

Huizinga (1971) introduz a ideia de que o jogo corresponde a um elemento primitivo, antecedente ao surgimento da cultura, visto que é compartilhado com outros animais. Ele ainda exemplifica com a brincadeira de cães, onde é feito uma arena e eles disputam entre si, respeitando suas próprias regras.

Crawford (1982) evidencia quatro elementos fundamentais de todos os jogos, são eles:

- **Representação**: O jogo fornece uma representação simplificada e subjetiva da realidade tendo um conjunto de regras explícitas, apresentam representações subjetivas originadas e sustentadas pela realidade. Além disso, essa representação fornece um ambiente completo e autossuficiente, pois seus elementos não dependem de nenhuma referência presente no mundo externo ao do jogo.

- **Interação**: O ponto crucial na representação da realidade situa-se na forma como ela se altera e a representação interativa. O espectador deve conseguir sentir essas alterações e saborear suas consequências, sendo, assim, capaz de modificar a realidade apresentada.

- **Conflito**: Surge naturalmente a partir da interação do jogador, está presente em todos os jogos. O jogador busca ativamente atingir algum objetivo e existirão obstáculos que o impede de ser alcançado facilmente. Essa força de oposição se dá de várias formas, como um cronômetro, um inimigo, objetos, dentre outros.

- **Segurança**: Uma vez que o conflito tende a criar um cenário de perigo, dele surge uma situação de risco real. Entretanto, o jogo permite que o jogador submeta-se à experiência psicológica desse perigo sem os danos, possibilitando, assim, desassociar as consequências das ações.

2.2 Jogos Digitais

Sob uma ótica objetiva, de acordo com Battaiola (2000): O jogo eletrônico é composto de três partes: enredo, motor e interface interativa.

- **enredo**: Define o tema, a trama, os objetivos principais e a sequência dos acontecimentos.
- **Motor**: É o mecanismo que controla a reação do ambiente às decisões do jogador, realizando as alterações de estado no ambiente.
- **interface interativa**: Permite a interação entre o jogador e o motor do jogo, fornecendo um caminho de entrada para as ações do jogador e um caminho de saída para as respostas audiovisuais referentes às mudanças do estado causados pelas decisões do jogados..

Dessa forma, fica claro que os jogos digitais fornecem uma nova representação para um jogo, com características e elementos próprios, surge uma questão em relação às vantagens de se utilizar tal abordagem em face dos meios tradicionais. Nesse sentido, é possível retomar os quatro elementos fundamentais dos jogos apontados por Crawford apresentados anteriormente e direcioná-los ao contexto dos jogos digitais:

- **Representação**: Considerando que os jogos digitais consistem numa complexa combinação de recursos, antes apenas áudio e vídeo e hoje em dia evoluindo atingindo todos os sentidos como o paladar ou o tato, fica evidente a grande riqueza dessa forma de representação. Tal riqueza se traduz, de forma geral, no aumento da imersão do jogador, que passa a sentir sensações mais elaboradas, comparáveis, por exemplo, às de um filme, exceto pelo fato de no jogo digital o jogador poder assumir o controle sobre o andamento dos acontecimentos através de interações.
- **Interação**: Como dito anteriormente, a interação tem papel fundamental nos jogos. entretanto, especificamente nos jogos digitais, a interação pode ser realizada de mais formas, podendo ser ainda em tempo real ou não. acima de tudo, as interações são muito bem coordenadas através do programa executável do jogo digital.
- **Conflito**: Em geral, têm-se conflitos em forma de agentes ativos que respondem às ações do jogador, dispondo de algum tipo de mecanismo que lhes forneça uma forma de inteligência. caso esse agente ativo apresente um obstáculo ao jogador, surge um inevitável conflito.
- **Segurança**: O ambiente lúdico criado pelo jogo digital permite uma complexa experimentação das sensações de perigo sem que isso represente qualquer risco ao jogador. Um exemplo dessa segurança seria um jogador capotar seu veículo em um jogo. De fato, quanto mais imersivo o jogo, maior será a sensação do jogador em relação às consequências da manobra, que seria, provavelmente, a completa destruição do veículo. Mas, mesmo o jogador podendo experimentar as emoções envolvidas, em momento algum sua integridade física foi posta em risco, uma vez que as consequências atingiram somente o mundo do jogo e o jogador possui plena consciência desse fato.

2.3 Ferramentas Relacionadas

Dito isso, é possível notar que o mapa é um elemento essencial na grande maioria dos jogos, visto que o mesmo pode atuar como intermediador entre as ações do jogador e as reações do jogo.

Nesta seção, serão apresentadas algumas ferramentas de criação de mapas, e ferramentas de criação de jogos capazes de criar mapas dentro delas, de forma explicar melhor o objetivo deste projeto.

2.3.1 Tiled

O Tiled foi criado por Lindeijer (2021) e por seus contribuidores é um criador de mapas 2D completo utilizando o sistema de Tiles para a criação dos mapas. A criação é completa, indo desde salas detalhadas com objetos, inimigos, luzes até a criação do mundo a longa distância, como casas, fazenda e terrenos. Tudo vai depender do pacote de Sprites que o usuário tiver à disposição.

Por utilizar de um sistema de tiles para a criação, os mapas criados podem ser top View ou plataforma, sendo os top View osa jogos onde temos visão por cima do mapa e o jogador anda em todas as direções e os jogos plataforma onde, geralmente, o jogador se move para a esquerda ou direita.

O criador é extensível a JavaScript, possibilitando que ele tenha uma grande variedade de opções, como suporte a mais tipo de arquivos, escrever suas próprias funções no criador de mapas e até mesmo automatizar sua área de trabalho.

Além da extensibilidade ao JavaScript, o Tiled possui diversos frameworks que dão suporte aos mnapas criados por ele em motores de jogos como o Game Maker, Construct 2, Unity ou em frameworks em linguagens de programação, como Python, Java e até mesmo HTML5.

2.3.2 OGMO Editor

De desenvolvedor Indie para desenvolvedor Indie, criado por Thorson (2012) e seus amigos, o OGMO é uma ferramenta de código aberto de criação de níveis para jogos, com possibilidade de trabalhar com vários projetos simultaneamente.

Similar ao Tiled, o OGMO também trabalha com um sistema de tiles, o que permite que os mapas criados sejam visto de cima ou lateralizados, permitindddo assim a criação de mapas para jogos 2D de estilos difeerentes.

É possível substituir algumas funçòes do projeto por outras novas desenvolvidas em JavaScript, configurando na ferramenta o local das funções externas, o que deeeixa o mapa beem customizável, mas não ao ponto do Tiled. Os códigos são executados através do node.js, no mesmo contexto do editor.

2.3.3 Pymapper

Saindo um pouco do contexto de jogos digitais, temos o Pymapper criado por Pymapper (2009) focado em criação de mapas para role-play, como Dungeons&Dragons.

De forma simples, você possui um conjunto de Sprites e monta seu mapa com eles no formato de Tiles e os exporta no formato de PNG.

2.3.4 LDtk

Deepnight Games é um estúdio de jogos independente formado por uma pessoa, fez parte do desenvolvimento de alguns jogos como Dead Cells e seu criador, Benard (2009), desenvolveu a LDtk - Level Designer Toolkit.

De código aberto e com uma interface amigável, esse criador de mapas baseado em tiles consegue combinar sutilmente a sotisficação de um mapa com a leveza da criação do mesmo. Com capacidade de auto-renderização, backups e até autopreenchimento de alguns locais.

Sendo capaz de exportar em JSON para maior facilidade de importação em outros motores de jogos, capacidade de criar e editar mundos completos e suporte a diversas plataformas e linguagens, como Python, JavaScript, Unity, Game Maker, etc.

Capítulo 3

Desenvolvimento

"Mathematical elegance is not a dispensable luxury
but a factor that decides between success and failure."
Edsger Wybe Dijkstra

3.1 Proposta de Solução

Existem diversos tipos de mapas e diversas formas de criar cada um deles, então seguimos com o desenvolvimento do criador de mapas.

Qual seria a diferença desse criador de mapas para os anteriores apresentados?

Dentre os criadores de mapas para jogos digitais apresentados, todos eles podem exportar em JSON, possuem *frameworks que os conectam com algum motor de jogo ou mesmo a uma linguagem de programação, entretanto nenhuma das apresentadas possui o poder de apresentar um mapa animado diretamente, ou mesmo nas propriedades de cada tile serem separados de forma independente.*

Como assim Propriedades?

Imaginemos um chão com cerâmica em uma casa, onde cada cerâmica tem suas propriedades. Agora imaginamos uma sacola em cada cerâmica, e dentro dessas sacolas, alguns papéis, onde eles seriam as propriedades. Utilizaremos algumas palavras como Solid, onde simbolizamos parede, Speed, onde simbolizamos um acrescento na velocidade, e Damage, onde caso exista, sofra algum ferimento.

Quando uma pessoa anda pelo chão, está andando normalmente. Mas se ele tentar pisar em uma cerâmica que tenha um papel escrito Solid, ele não conseguirá chegar nele, como se fosse uma parede. Já se tivesse um papel com Speed escrito, a pessoa teria um acréscimo de velocidade, assim por diante. O próprio usuário que está criando o mapa poderia definir um valor para essas propriedades e, dessa forma, tornando o mapa mais dinâmico.

3.1.1 Sistema Proposto

Dito isso, surgiu a ideia de desenvolver um criador de mapas, com o intuito dos mapas criados poderem ser exportados para qualquer ferramenta, além de servir como base para aqueles que desejem desenvolver um novo projeto e não saibam por onde começar. Mapas com o poder de serem configurados pedaços separadamente, e de serem completamente animados.

3.1.2 Requisitos Funcionais

Os requisitos funcionais de um sistema descrevem o que ele pode e deve fazer. Eles dependem do tipo de software a ser desenvolvido, de quem são seus possíveis usuários e da abordagem geral adotada pela organização ao escrever os requisitos SOMMERVILLE (2011)

Com base nas necessidades, foram levantados os seguintes requisitos funcionais:

- ***RF01 Construir Mapas:*** *Construir mapas, pedaço a pedaço, permitindo configurar o caminho de um jogador da forma que o usuário desejar.*
- ***RF02 Personalizar Mapas:*** *Permitir adicionar sprites diferentes em cada pedaço, criar um mapa criativo e conectado.*
- ***RF03 Criar Paredes/Obstáculos:*** *Poder impedir o jogador de acessar alguns pedaços do mapa construído.*
- ***RF04 Tiles Independentes****: Permitir que o usuário possa configurar cada tile do mapa de forma independente.*
- ***RF05 Construções:*** *Permitir que o usuário crie e salve construções, além de poder compartilha-lás com outros usuários.*
- ***RF06 Personalizar Bibliotecas:*** *Permitir que o usuário selecione Sprites disponíveis e crie uma biblioteca com eles. Por exemplo: selecionar sprites que lembrem de água e adicioná-los a uma biblioteca mar.*
- ***RF07 Adicionar imagens:*** *Permitir que o usuário possa adicionar a própria lista de imagens para que possa utilizá-los como Sprites caso desejar.*
- ***RF08 Criar Cidades/Casas:*** *Ser possível criar áreas como casas e cidades, não apenas construções que juntas pareçam uma cidade.*
- ***RF09 Exportação em Json:*** *Exportar o mapa criado em formato Json.*
- ***RF10 Exportação em XML:*** *Exportar o mapa criado em formato XML.*
- ***RF11 Salvar:*** *Salvar o mapa em desenvolvimento para finalizá-lo depois.*
- ***RF12 Carregar mapa salvo:*** *carregar o mapa previamente salvo.*
- ***RF13 Altura:*** *Criar escadas ou outra forma onde sejá possível criar a ilusão do jogador estar em uma altura mais elevada.*
- ***RF14 Iterações:*** *Permitir a criação de iterações no mapa, seja através de armadilhas, botões e/ou alavancas.*
- ***RF15 testar:*** *Forma de testar o mapa com personagem de teste.*
- ***RF16 Animação:*** *Permitir o mapa ser animado com mais de um Sprite por tile.*
- ***RF17 Tamanho personalizável:*** *tamanho do mapa personalizável.*
- ***RF18 Compartilhar mapa criado:*** *Compartilhamento de mapa salvo caso o usuário deseje.*

- ***RF19 Compartilhar construção criada:*** *Compartilhamento de construções caso o usuário deseje.*
- ***RF20 Tutorial para utilização:*** *Tutorial de utilização do produto.*

3.1.3 Requisitos Não Funcionais

Já os requisitos não funcionais não estão diretamente relacionados com os serviços específicos oferecidos pelo sistema a seus usuários. Eles podem estar relacionados às propriedades emergentes do sistema, como confiabilidade, tempo de resposta e ocupação de área SOMMERVILLE (2011)

- ***RNF01****: O sistema deve funcionar em Windows.*
- ***RNF02****: O sistema deve funcionar em Linux.*
- ***RNF03****: O sistema deve funcionar em Mac.*
- ***RNF04****: O sistema deve ser rápido, não demorando para exportar mapas criados.*
- ***RNF05****: O sistema deve ser rápido, não demorando para carregar mapas salvos.*

3.2 Protótipo

Nesta seção, são expostas algumas telas do protótipo do sistema para ser possível visualizar e compreender melhor a ideia do sistema proposto. O protótipo projetado é a versão pré-alpha do sistema, visto que o desenvolvedor começou o projeto e já tem algo visível em mãos. Esse protótipo está funcional, ou seja, já existem algumas funções completas para a versão final, necessitando, talvez, de refinamento futuro.

Inicialmente, ao clicar para abrir o programa, o usuário é recebido com uma tela simples para definir a imagem do mundo, como visto abaixo na Figura 3.1.

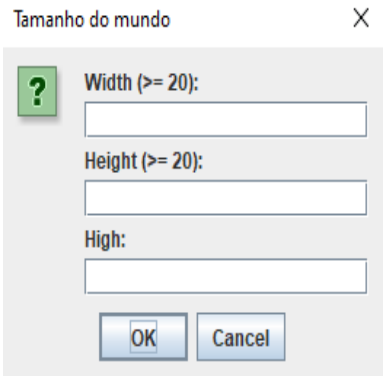

Figura 3.1: Tela Inicial

Aqui, Width seria o comprimento, da esquerda para a direita, Height a largura, de cima para baixo e High a altura, não de cima para baixo, mas para mais perto e mais longe da tela. Imagine um prédio, com comprimento, largua e andares. Caso não seja escolhido o tamanho e pressionado ok, o motor abrirá no tamanho padrão de 20x20x7 definido em

um arquivo de configuração, caso escolha um tamanho, ele abrirá no tamanho informado e mostrará a tela primária:

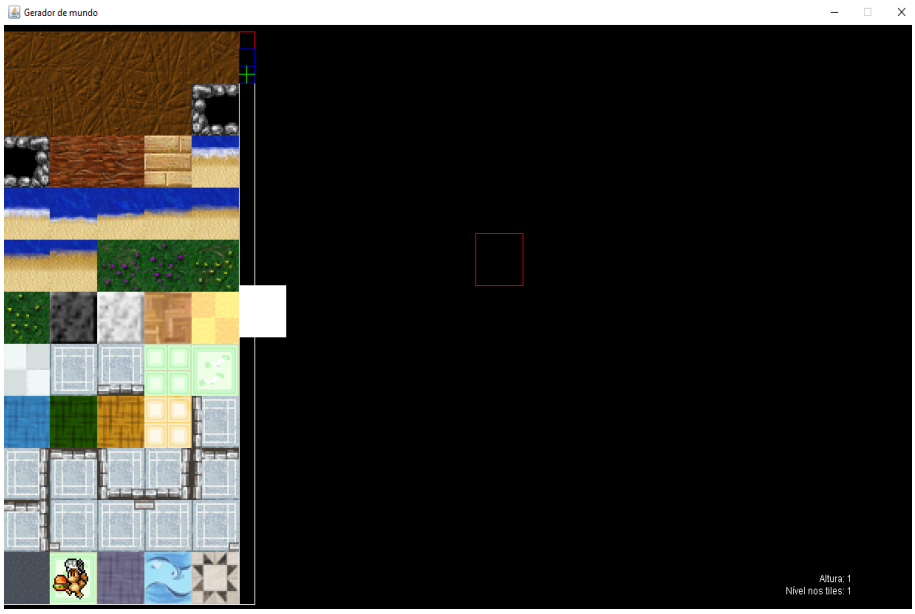

Figura 3.2: Tela Primária

Note que existe uma grande quantidade de sprites a escolha no canto esquerdo, e não existem apenas esses, visto que essa é a primeira de aproximadamente 200 páginas de sprites default colocados no sistema. Atualmente os sprites são separados em n imagens diferentes onde cada imagem contem um número x de imagens do mesmo tamanho. Elas são todas convertidas para o tamanho 32x32 no menu que é exatamente o tamanho de cada Tile do mapa criado nas imagens, entretanto ao selecionar é possível ver o tamanho real da imagem.

A segunda aba, de configuração, demonstrado na Figura 3.3 abaixo, atualmente é possível ajustar a velocidade do jogador, o quadrado branco. Mas também será possível definir paredes, locais de acesso restrito e/ou pós-missões por propriedades que serão definidas no local esquerdo. Notar que na parte de cima existe um menu que desce e sobe caso o mouse se aproxime.

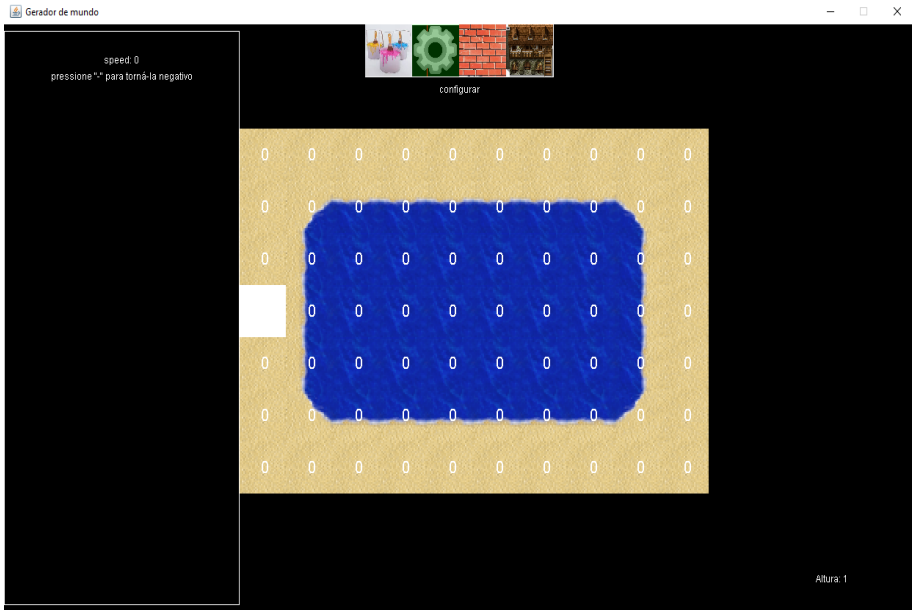

Figura 3.3: Tela de Configuração.

A próxima aba seria a de construções, mas antes de ir para ela precisamos salvar uma construção qualquer primeiro. Ao clicar com o botão direito (em qualquer aba) o chão ficará levemente amarelado, e a direita aparecerá algumas opções, é assim que se seleciona dentro do programa. Para des-selecionar, basta clicar com o botão direito em um chão selecionado ou mesmo clicar na opção limpar selecionados conforme na Figura 4

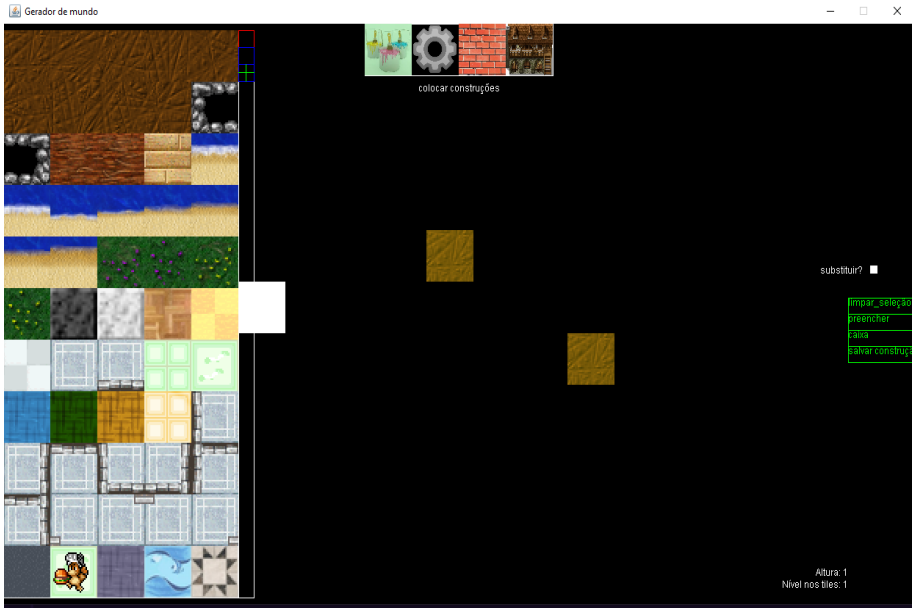

Figura 3.4: Chãos Selecionados.

Ao clicar em Salvar construção, uma caixa aparecerá conforme a Figura 3.5, o usuário deve preencher um nome para esta construção, caso esse nome exista, a tela permanecerá e avisará que já existe uma construção com esse nome, obrigando-o a escrever um nome

novo ou cancelar o salvamento de tal objeto. O usuário também terá a opção de apagar a pasta onde a construção se encontra, não precisando alterar o nome.

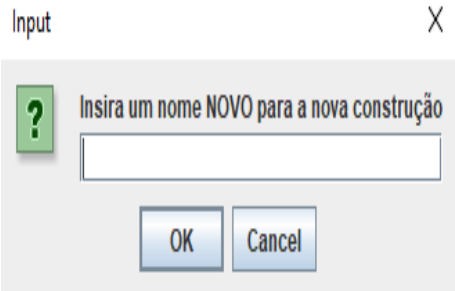

Figura 3.5: Salvar Construção.

Por fim, na aba de Construções mostrará no canto esquerdo um item que ao ser selecionado mostrará uma imagem de exibição gerada ao salvar a imagem, sendo uma miniatura da construção feita, conforme a Figura 3.6. Após isso, basta ir clicando no mapa com a construção selecionada que o ela será colocada no local.

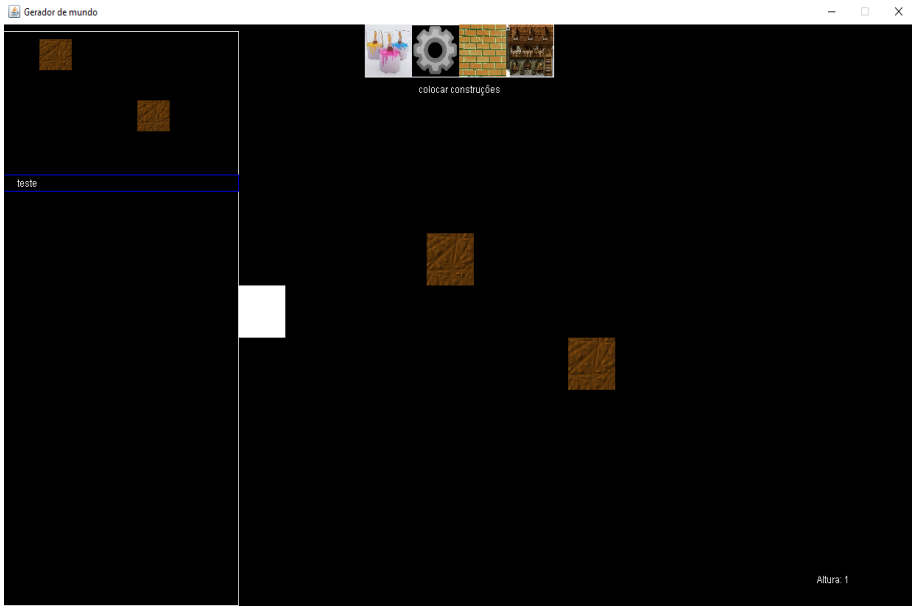

Figura 3.6: Aba Construções.

A próxima aba é a aba de cidades e/ou casas, entretanto a mesma ainda está em desenvolvimento e não funcional. Entretanto, ainda não é o foco do desenvolvedor, visto que o mesmo está focado em bugs e melhorias nas outras três abas, como visto nas tarefas do projeto no github mencionado abaixo.

É possível visualizar e executar Obter o beta para testes através do github do desenvolvedor no seguinte link: https://github.com/pedrojonassm/CriadorMapaTopView

O projeto vem com uma base inicial de imagens que podem ser utilizadas como Sprites para os tiles dos mapas a serem criados, após salvar um mapa é possível localizar um arquivo de texto com o nome do mapa salvo, ao compartilhar o arquivo com outro usuário,

o mesmo poderá carregar o mapa e deitá-lo da forma que desejar. Este padrão segue também com construções, onde a única diferença é que a construção pode ser carregada em qualquer mapa.

Capítulo 4

Conclusões

"If we can really understand the problem,
the answer will come out of it,
because the answer is not separate from the problem."
Jiddu Krishnamurti

Neste trabalho foi desenvolvido uma ferramenta para criação de mapas para jogos 2D, capaz de importar imagens externas para serem posteriormente adicionadas no mapa. Propriedades para disparo de eventos, criação de paredes, falas, dentre outros.

4.1 Resultados

O projeto foi feito e concluído, com possíveis melhorias no futuro que vão ser discutidas na sessão 4.3.

O criador possui alguns sprites já embutidos dentro dele, possibilidade de mover entre um mapa e outro de forma simples, teletransportes embutidos, animações por Tile, mudança de animações, construções que podem ir de um mapa a outro, marcadores para facilitar encontrar os vários sprites que podem ser utilizados para a construção de um mapa dinâmico.

Mas bem, não tem melhor forma de demonstrar os resultados de uma ferramenta que voltar a lista de requisitos já mencionada anteriormente.

- *(✓) **RF01 Construir Mapas:** Mapas criados da forma esperada, pedaço a pedaço.*
- *(✓) **RF02 Personalizar Mapas:** É possível criar cada pedaço distintamente.*
- *(✓) **RF03 Criar Paredes/Obstáculos:** Com as propriedades, é possível fazer de tudo.*
- *(✓) **RF04 Tiles Independentes:** Efetuado em conjunto o RF03.*
- *(✓) **RF05 Construções:** Construções podem ser selecionadas e salvas, então duplicadas quantas vezes desejar.*
- *(✓) **RF06 Personalizar Bibliotecas:** Com as marcações é possível separar os sprites para os Tiles para facilitar encontrá-los posteriormente.*

- *(✓) **RF07 Adicionar imagens:** É possível importar qualquer imagem externa para o mapa para posteriormente colocá-lo.*
- *(X) **RF08 Criar Cidades/Casas:** Apesar de não ter sido feito, é possível chegar no mesmo resultado com a RF03.*
- *(✓) **RF09 Exportação em Json:** O mapa é exportado em Json, em conjunto com imagens utilizadas e lista de propriedades utilizadas.*
- *(X) **RF10 Exportação em XML :** Devido à facilidade de conversão de JSON para XML e vice-versa, não foi realizado.*
- *(✓) **RF11 Salvar:** Essencial, salvo em uma pasta.*
- *(✓) **RF12 Carregar mapa salvo:** Essencial, especialmente com o RF11.*
- *(✓) **RF13 Altura:** É possível mudar a altura e criar montanhas, andares, etc. sem a necessidade de um novo mapa.*
- *(✓) **RF14 Iterações:** Com a possibilidade de Sprites Múltiplos, basta alternar com a bolinha do mouse.*
- *(✓) **RF15 testar:** É possível testar algumas coisas dentro do próprio criador de mapas, mas não tudo que se deseja. Para isso, foi criado uma engine que é possível importar o mapa e tem alguns exemplos na sua wiki, falaremos mais dela na Próxima sessão.*
- *(✓) **RF16 Animação:** Animação podendo escolher mais de 1 sprites dentre os disponíveis.*
- *(✓) **RF17 Tamanho personalizável:** Desde uma casinha até um mundo inteiro (mas recomendo quebrar o mundo em algumas partes).*
- *(✓) **RF18 Compartilhar mapa criado:** Basta compartilhar o arquivo salvo.*
- *(✓) **RF19 Compartilhar construção criada:** Basta compartilhar o arquivo salvo.*
- *(✓) **RF20 Tutorial para utilização:** Essa é uma ótima deixa para o motor criado, então falemos dela.*

4.2 Trabalhos futuros

Apesar do criador estar completo, sempre temos melhorias possíveis em uma ferramenta de software. podemos citar, por exemplo, o requisito RF10, visto que apesar da facilidade, a própria facilidade é um motivo para realizar essa melhoria. Outra possibilidade seria adicionar mais exemplos ao motor criado que pode carregar automaticamente os mapas criados.

Expansão da biblioteca de exemplos: Para melhorar a experiência dos usuários, podemos adicionar mais exemplos ao motor de criação de mapas mostrado. Isso fornecerá aos usuários uma variedade maior de opções e inspiração para criar seus próprios mapas, além de maior facilidade.

Geração automática de mapas com IA: Uma ideia interessante é desenvolver uma funcionalidade que utilize inteligência artificial para gerar mapas aleatórios para jogos. Isso envolveria treinar um modelo de IA capaz de entender os elementos-chave dos jogos e criar mapas que sejam desafiadores e interessantes.

Desenvolvimento de um jogo completo: utilizando-se da ferramenta de criação de mapas, podemos considerar desenvolver um jogo completo. Com uma história envolvente e peculiar que o jogo possa contar, ou mesmo em uma aplicação que ensine sobre poluição, reciclagem ou qualquer outro tópico relevante.

Capítulo 5

Apêndice

Nessa sessão, vamos falar de um motor criado para carregar os mapas exportados e desenvolver um jogo completo. O projeto está no github, com o nome de SimpleMapCreatorLoaders. Nele será colocado alguns exemplos para que seja possível um usuário saber utilizado, com explicações dos exemplos na Wiki do próprio projeto no site.

Ao realizar o download ou o clone do projeto, basta colocar o mapa dentro da pasta Mundos em res que o jogo já poderá ser executado. O player já pode se mover naturalmente pelo mapa e colidir com paredes e escadas de forma similar ao que é no criador de mundo. Agora o usuário deve programar o jogo do jeito que desejar.

O motor vem com uma classe capaz de encontrar o melhor caminho entre um ponto e outro, algo que é utilizado em um dos exemplos criados na Wiki.

5.1 Detalhando exemplo simples da Wiki

Falaremos de um exemplo que está na Wiki do projeto, mais especificamente o primeiro Exemplo, intitulado "Exemplo 1". Por sem apenas um exemplo e, obviamente, o primeiro, ele é simples e pequeno o suficiente para ser demonstrado em alguns parágrafos.

A ideia da história do jogo é simples. Ao iniciar, o jogador vai se encontrar do lado de fora de uma casa e ao entrar nela, a porta fechará trancando-o com 2 NPC's presentes na casa. Para sair, o jogador tem que ir para o andar de cima, pegar a chave e abrir a porta, então estará livre.

Figura 5.1: Andar de baixo da casa

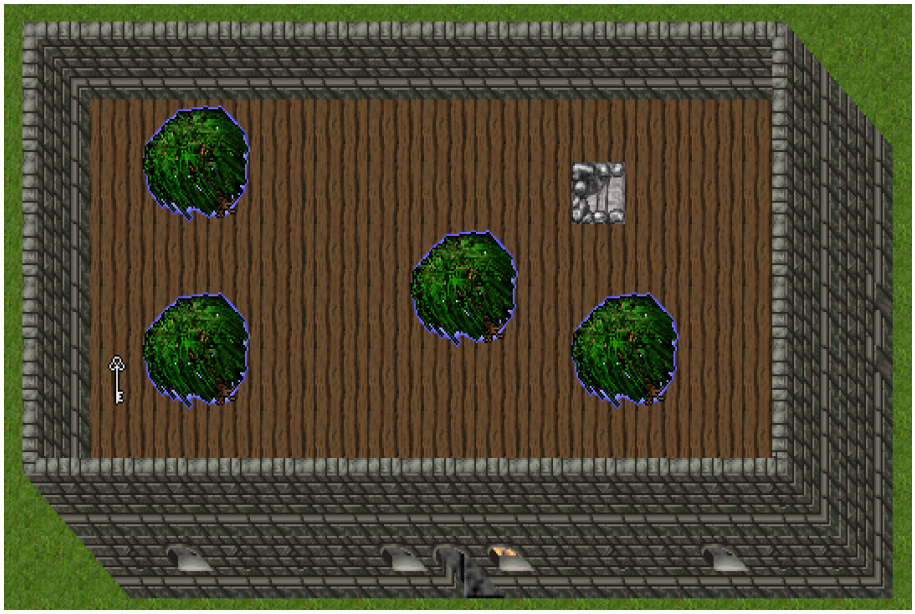

Figura 5.2: Andar de cima da casa

Para spawnar os NPC's vamos precisar utilizar as Propriedades, então nas camas, adicionaremos uma Propriedade NPC com os valores Monika e Sebastião, serão os nomes dos NPC's.

Com 2 NPC's, vamos efetuar o seguinte esquema:

- *Monika não deixa você subir a escada, sempre pedindo para você ficar abaixo com ela e Sebastião.*
- *Monika também não deixa o jogador mexer na alavanca, mas não explicará o motivo.*

- *Sebastião não deixa você mexer na anêmona (item rosa perto da escada), sem explicação.*

- *Os NPCs não impedirão o jogador de fazer algo quando estiverem dormindo.*

Dessa forma, o jogador não vai simplesmente subir, pegar a chave e sair. Ele terá que obrigatoriamente botar os personagens para dormir, mas como fará isso?

Bom, primeiramente precisaremos de alguns desenhos extras no mapa. As camas, a alavanca, o porquinho e a anêmona no andar de baixo vão ter iterações, e precisamos saber quando eles forem ativados/desativados.

Figura 5.3: Andar de baixo da casa segundo conjunto

Com isso, vamos seguir a ordem:

1. ***Jogador interage com Porquinho de Sebastião e Quebra ele**: Sebastião reclama e dormirá para desistressar.*

2. ***Sem Sebastião, jogador deve matar a anêmona**: Monika ficará triste com a morte da anêmona e vai chorar na cama, então cairá no sono.*

3. ***Sem Monika, jogador sobe Escada**: Assim ele sobe a escada, pega a chave da porta e pode sair livremente, fim de jogo.*

E para fazer isso, vamos precisar de uma propriedade nova, vamos chamá-la de Eventos e colocar nos seus respectivos lugares.

Figura 5.4: Andar de baixo com Propriedades de Evento

- ***QuebrarPorco****: Quebrar o Porquinho.*

- ***CallSebastiao****: Anêmona. Se sebastião tiver acordado ele vem pro jogador, caso contrário vai matar ela.*

- ***CallMonika****: Na alavanca. Se ela estiver dormindo vai alterar o sprite dela, caso contrário a Monika virá até o jogador, impedindo-o de mexer nela.*

- ***DestrancarPorta****: Ficará na chave. Destrancará a porta.*

- ***TrancarPorta****: Logo na frente da porta. Quando o jogador passar, fará a porta ser trancada.*

- ***Porta****: Quando o jogador interagir fará a porta abrir se ela estiver destrancada.*

- ***Fujir****: Atrás da porta, ativada quando o jogador tiver com a chave. Faremos o jogador mandar uma lição sobre obedecer a mãe.*

Por fim, vamos colocar um pequeno easterEgg no andar de cima, também para explicar o porquê da Monika não deixar mexer na alavanca, visto que ela não faz nada.

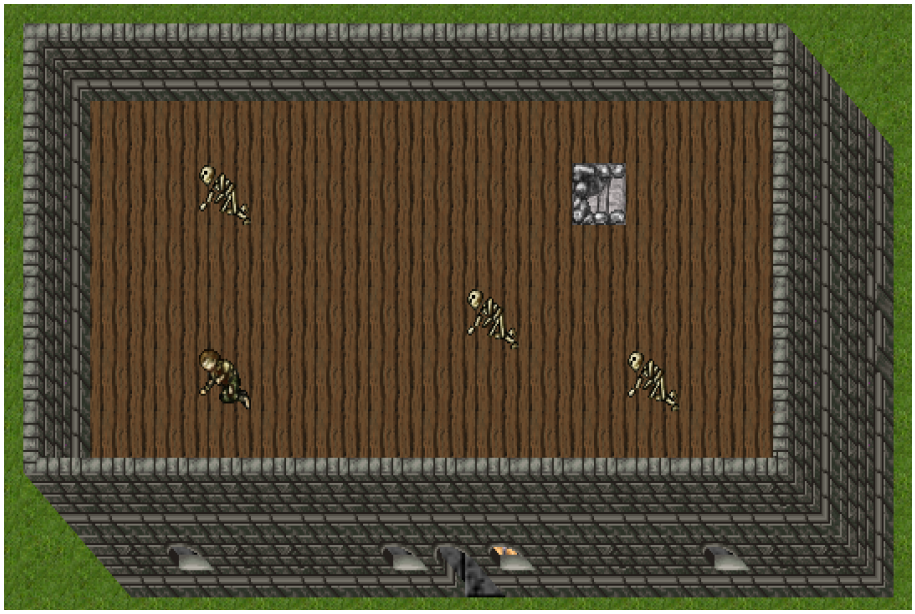

Figura 5.5: Andar de cima da casa segundo conjunto.

No andar de cima, vamos pegar aquelas arvores falsas e trocar por alguns cadáveres, que explica o desespero da Monika sobre a Alavanca.

A programação está toda contida na Wiki, o passo a passo de como foi feito, aqui isso será o suficiente para demonstrar o objetivo do jogo.

Referências Bibliográficas

Battaiola, André L (2000), 'Jogos por computador–histórico, relevância tecnológica e mercadológica, tendências e técnicas de implementação', Anais da XIX Jornada de Atualização em Informática, SBC **2**, *83–122.*
(Citado na página 6)

Benard, Sébastien (2009), 'LDtk'.
***URL:** https://deepnight.net/tools/ldtk-2d-level-editor/*
(Citado na página 8)

Crawford, Chris (1982), 'The art of digital game design', Washington State University, Vancouver .
(Citado na página 5)

De Paula, Bruno Henrique, José Armando Valente e Hermes Renato Hildebrand (2016), 'Criar para aprender: Discutindo o potencial da criação de jogos digitais como estratégia educacional', Tecnologia Educacional **54***(212), 6–18.*
(Citado na página 2)

Huizinga, Johan (1971), Homo ludens: o jogo como elemento da cultura*, Vol. 4, Editora da Universidade de S. Paulo, Editora Perspectiva.*
(Citado na página 5)

Lindeijer, Thorbjørn (2021), 'TiledMap'.
***URL:** https://www.mapeditor.org*
(Citado na página 7)

Medeiros, Tainá Jesus (2014), Um framework para criação de jogos voltados para o ensino de lógica de programação, Dissertação de mestrado, Universidade Federal do Rio Grande do Norte.
(Citado na página 1)

Pymapper (2009), 'Pymapper'.
***URL:** https://pymapper.com*
(Citado na página 7)

SOMMERVILLE (2011), Engenharia de Software*, Pearson Education.*
(Citado nas páginas 10 e 11)

Thorson, Maddy (2012), 'OGMO Editor'.
***URL:** https://ogmo-editor-3.github.io*
(Citado na página 7)