

INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DO RIO  
GRANDE DO NORTE

CARLOS EDUARDO ALVES SARMENTO

NÁTALY ENNE DA COSTA GONÇALO

**DESENVOLVIMENTO DE UM JOGO PARA ENSINAR LÓGICA DE  
PROGRAMAÇÃO A CRIANÇAS**

Pau dos Ferros

2017

CARLOS EDUARDO ALVES SARMENTO  
NÁTALY ENNE DA COSTA GONÇALO

**DESENVOLVIMENTO DE UM JOGO PARA ENSINAR LÓGICA DE  
PROGRAMAÇÃO A CRIANÇAS**

Trabalho de conclusão de curso  
apresentado ao Instituto Federal de  
Educação, Ciência e Tecnologia do Rio  
Grande do Norte como exigência parcial  
para obtenção do título de Técnico em  
Informática.

Orientador: Esp. Alan Klinger Sousa Alves

CARLOS EDUARDO ALVES SARMENTO  
NÁTALY ENNE DA COSTA GONÇALO

**DESENVOLVIMENTO DE UM JOGO PARA ENSINAR LÓGICA DE  
PROGRAMAÇÃO A CRIANÇAS**

Trabalho de conclusão de curso  
apresentado ao Instituto Federal de  
Educação, Ciência e Tecnologia do Rio  
Grande do Norte como exigência parcial  
para obtenção do título de Técnico em  
Informática.

Aprovado em \_\_\_\_/\_\_\_\_/\_\_\_\_

**BANCA EXAMINADORA**

---

Prof. Esp. Alan Klinger Sousa Alves – Presidente  
Instituto Federal de Educação, Ciência e Tecnologia do Rio Grande do Norte

---

Prof. Me. Luiz Fernando Virginio da Silva – Examinador  
Instituto Federal de Educação, Ciência e Tecnologia do Rio Grande do Norte

---

Prof. Me. Jeferson Queiroga Pereira – Examinador  
Instituto Federal de Educação, Ciência e Tecnologia do Rio Grande do Norte

A Deus por ter nos agraciado com o dom  
do intelecto e nos guiar a cada escolha,  
que esse trabalho seja para a glória dEle.

## **AGRADECIMENTOS**

Agradecemos primeiramente a Deus, nosso Senhor, por nos apoiar, nos dar refúgio nos momentos difíceis e estar ao nosso lado nos bons momentos. À equipe da Unity Technologies por disponibilizar uma ferramenta tão completa de forma gratuita. À comunidade do Stack Overflow por estar sempre se empenhando em trocar conhecimentos para a resolução de problemas e crescimento de todos.

Também agradecemos ao nosso orientador, Alan Klinger, por oferecer seu tempo para nos ajudar a desenvolver esse trabalho. A todos os canais do YouTube que disponibilizam um conteúdo rico para quem se interessa em aprender mais dessa área, em especial a Paulo, do canal We Make a Game, que introduziu a mim, Carlos Eduardo, o mundo do desenvolvimento de jogos e a Bruno, do canal Games Indie, que trouxe os primeiros conhecimentos das ferramentas aqui utilizadas a mim, Nátaly Enne.

Finalmente, agradecemos aos nossos pais por ter nos provido de uma criação correta, nos passando valores e construindo uma moral em nós que carregaremos por toda a vida e aos nossos amigos por ter nos apoiado durante essa caminhada de quatro anos no curso. Enfim, a todos que ajudaram direta ou indiretamente para a conclusão desse trabalho.

*“I think everybody in this country should learn how to program a computer, should learn a computer language, because it teaches you how to think.”*

*(Steve Jobs)*

## **RESUMO**

A tecnologia já está presente intrinsecamente na vida de todos. Compreender e saber fazer uso dela se torna tão essencial quanto saber ler em um mundo completo de coisas escritas. É por isso que o ensino de programação está se mostrando algo necessário e que deve ser introduzido no currículo básico das escolas. Mas como apresentar esses saberes a quem ainda está no início do seu desenvolvimento cognitivo? Diante dessa problemática, surgiram as linguagens de programação em blocos, que conseguem abstrair conhecimentos bastante específicos e avançados, se prendendo mais à lógica da programação do que à sua sintaxe. Com isso em mente, o presente trabalho tem como objetivo desenvolver um jogo que faça uso dessa forma de programar para introduzir o raciocínio lógico utilizado na criação de códigos para crianças.

Palavras-chave: Jogo. Programação em blocos. Unity.

## **ABSTRACT**

Technology is already present intrinsically in everyone's life. To understand and to know how to make use of it becomes as essential as to know how to read in a world full of written things. That's why the teaching of programming is proving something required and that has to be introduced in the core curriculum of schools. But how to present these knowledges to who is still in the start of its cognitive development? Against this problem, block-based coding languages have emerged, that can abstract very specific and advanced knowledges, holding more on the programming logic that on its syntax. With this in mind, the present work has as objective to develop a game that makes use of this way of programming to introduce the logical reasoning used in the creation of codes to kids.

Keywords: Game. Block-based coding. Unity.



## SUMÁRIO

<b>1 INTRODUÇÃO .....</b>	<b>9</b>
<b>1.1 Objetivos gerais .....</b>	<b>9</b>
<b>1.2 Objetivos específicos .....</b>	<b>10</b>
<b>1.3 Justificativa .....</b>	<b>10</b>
<b>2 FUNDAMENTAÇÃO TEÓRICA .....</b>	<b>12</b>
<b>3 METODOLOGIA.....</b>	<b>13</b>
<b>3.1 Tecnologias utilizadas.....</b>	<b>14</b>
3.1.1 <i>Unity 3D .....</i>	14
3.1.2 <i>C# (C Sharp).....</i>	15
3.1.3 <i>Photofiltre Studio X.....</i>	15
<b>4 DESENVOLVIMENTO .....</b>	<b>16</b>
<b>4.1 Programação em blocos .....</b>	<b>16</b>
4.1.1 <i>Manipulação dos blocos .....</i>	16
4.1.2 <i>Interpretador .....</i>	16
<b>4.2 Montagem dos níveis .....</b>	<b>18</b>
4.2.1 <i>Blocos especiais .....</i>	18
<b>4.3 Menu .....</b>	<b>19</b>
<b>4.4 Robô.....</b>	<b>19</b>
<b>4.5 Resultados .....</b>	<b>20</b>
<b>6 CONSIDERAÇÕES FINAIS .....</b>	<b>23</b>
<b>6.1 Post-mortem.....</b>	<b>23</b>
<b>6.2 Trabalhos futuros .....</b>	<b>24</b>
<b>REFERÊNCIAS .....</b>	<b>25</b>

## 1 INTRODUÇÃO

O mundo está sofrendo um grande avanço tecnológico gradual, no qual tudo ao seu redor vai se adaptando ao seu modelo e mudando a forma de vida de adultos, idosos e crianças. Hoje, os computadores são usados em quase todos os aspectos de nossas vidas (FERNANDO, 2017).

Nessa linha de pensamento, surge uma nova perspectiva na educação: a inclusão da programação na ementa acadêmica das escolas de ensino infantil, fundamental e médio. Segundo especialistas da primeira escola de Programação e Robótica voltada a Crianças, Adolescentes e Jovens do Brasil (SUPERGEEKS, 2016), aprender a programar no século XXI é tão importante quanto aprender a ler e escrever foi no século XX.

O professor Moacir Gadotti (2000) afirma em seu trabalho *Perspectivas Atuais da Educação* que o conhecimento da tecnologia, principalmente a programação, tem participação garantida em qualquer projeção que se faça do futuro. Assim, ela é considerada como base de qualquer sociedade.

Desse modo, surge a junção de dois termos: educação e tecnologia. A informação deixou de ser uma área ou especialidade técnica para se tornar uma dimensão de tudo nesses últimos anos, mudando profundamente a organização da sociedade, principalmente a educacional (GADOTTI, 2000).

É imprescindível que a introdução do ensino de programação nos níveis de educação básica seja discutida nas diferentes áreas do conhecimento de forma a estabelecer relações entre as demais disciplinas (LUIZ, 2016).

Portanto, a ideia desse trabalho é servir como mecanismo que desperte no educando a habilidade de interpretação e resolução de problemas simples e complexos, o raciocínio lógico e matemático, e a capacidade de relacionar causa e consequência a partir do uso da programação para completar cada nível.

### 1.1 Objetivos gerais

O trabalho em questão tem como objetivo primordial a criação de uma ferramenta lúdica que introduza lógica de programação a crianças e que, a partir dessa prática, possibilite o desenvolvimento das habilidades da criança nas demais áreas do conhecimento.

## 1.2 Objetivos específicos

- Desenvolver um jogo eletrônico em três dimensões.
- Apresentar os princípios básicos da programação em bloco para crianças.
- Estimular o educando ao raciocínio lógico e a busca de alternativas para resolver problemas.
- Proporcionar não só aprendizado, mas também divertimento.

## 1.3 Justificativa

Muitos adultos acreditam que o ensino de programação de computador para crianças ajuda apenas para o desenvolvimento de futuros profissionais na área da tecnologia. Porém, especialistas em educação e tecnologia cada vez mais defendem que a programação é um elemento crucial no currículo escolar, simplesmente pelo fato de saber programar é poder “ler o mundo” na Era Digital (FURIA, 2014).

Como já dito, o ensino de programação é importante pois estimula o desenvolvimento do raciocínio lógico, a autonomia, a criatividade e a capacidade de resolução de problemas. Além disso, aprender a programar nos dias de hoje é de grande importância, visto que vivemos em um mundo onde todos estão rodeados de tecnologia e todas as áreas dependem direta ou indiretamente da mesma (LUIZ, 2016).

O ex-presidente norte-americano Barack Obama fez um discurso em dezembro de 2013 para lançar a campanha *Hour of Code*<sup>1</sup> durante a Semana do Ensino da Ciência da Programação, nos Estados Unidos, procurando chamar a atenção dos jovens para a importância do ensino da programação de computadores (SUPERGEEKS, 2016).

A programação não é apenas aplicada na matemática ou na física, mas também na história, geografia, filosofia, na arte e na química (LUIZ, 2016). Ela se encaixa em inúmeras áreas, ajudando no desenvolvimento, aprimoramento e aprendizado do educando.

---

<sup>1</sup> Evento global que mostra que qualquer pessoa pode aprender o básico e ampliar a participação no campo da ciência da computação. Disponível em: <<https://hourofcode.com/pt>>.

Desse modo, a implantação de softwares ou quaisquer ferramentas que despertem, melhorem, reforcem e ajudem no crescimento intelectual da criança é estritamente importante, pois, sendo isso incentivado quando pequenos, terá um efeito demasiado na construção do conhecimento e à medida que crescerem poderão evoluir habilidades mais aprofundadas e uma performance maior e mais produtiva (CASTRO, 2017).

## 2 FUNDAMENTAÇÃO TEÓRICA

A programação em blocos consiste em uma linguagem gráfica que utiliza blocos com funções prontas. Ela tem seus elementos interligados, assemelhando-se a circuitos elétricos.

Os blocos constituem-se de uma ou mais declarações ou definições de identificadores. Eles são fundamentais para a programação estruturada. Podem conter instruções e/ou expressões, que definem um algoritmo executável ou parte dele, no qual são formados a partir de blocos.

A função dos blocos na programação é possibilitar que grupos de instruções sejam tratadas de forma fácil e direta. As ferramentas que trabalham com programação em bloco, auxiliam, de modo prático, na iniciação de crianças ao mundo tecnológico, do desenvolvimento de aplicativos e games para web e mobile (SILOTTO, 2017).

A programação em blocos é bastante recente se comparada a outras formas de programar desenvolvidas para facilitar o aprendizado. Mesmo assim, nos últimos anos se percebe uma grande quantidade de novas ferramentas que usam a programação em blocos para atingir os fins educativos. Essa explosão de novas ferramentas tem como consequência o aumento da capacidade de criação utilizando somente essa forma de programar (WEINTROP; WILENSKY, 2015).

### 3 METODOLOGIA

A princípio o jogo foi baseado em algumas ferramentas para estudo e aprendizado de programação, como:

- CodeCombat – Um jogo de RPG que ensina os fundamentos de programação. No qual, para interagir com os personagens, é necessário inserir comandos em formato de código.
- Blockly Games – Uma série de jogos educativos que ensinam programação. Há níveis que vão ensinando cada fundamento. É baseado em quebra-cabeças, ou seja, para fazer o personagem andar deve-se juntar as peças.
- Scratch – Uma linguagem gráfica de programação. Ele utiliza linguagem de blocos em que é possível criar animações, jogos, simulações, entre outros.
- CodeMonkey – Um jogo online que consiste em escrever linhas de códigos para programar o que o personagem deve fazer e assim chegar ao objetivo do jogo.

Com isso, fizemos um planejamento dividido em etapas, ou seja, períodos que, por sua vez, possuem dois tópicos intitulados “Requisito” e “Objetivo”. O “Requisito” significa a grosso modo o que o jogo vai fazer e o “Objetivo” é o principal feito do desenvolvimento. Veja na figura abaixo:

Figura 1 – Tabela de planejamento produzida no Excel

ETAPA I		
REQUISITO		
O jogador manipula blocos na tela para dar as instruções ao personagem.		
Os blocos são executados na ordem em que foram posicionados.		
OBJETIVO		
Criar a mecânica básica do jogo: programação em blocos.		
FUNCIONALIDADE	RESPONSÁVEL	SITUAÇÃO
Criar a interface básica.	Nátaly	OK
Desenvolver o sistema de manipulação dos blocos.	Nátaly	OK
Desenvolver a mecânica dos blocos.	Eduardo	OK
Criar a arte e animação dos blocos.	Nátaly	OK
Desenvolver o interpretador de blocos.	Eduardo	OK
Criar a resposta visual do interpretador.	Nátaly	OK

Fonte: Elaborada pelos autores

Seguindo a tabela, há três tópicos: “Funcionalidade”, “Responsável” e “Situação”. Respectivamente, significa cada funcionalidade do jogo a ser feita, o responsável pelo desenvolvimento e o estado em que se encontra. Por fim,

elaboramos um cronograma (Figura 2) que consiste na organização de datas para se ter um controle de tempo de desenvolvimento do jogo.

Figura 2 – Cronogramas do planejamento

CRONOGRAMA DE DESENVOLVIMENTO			
ETAPA	INÍCIO	FIM	SITUAÇÃO
I	10/04/2017	07/07/2017	OK
II	10/07/2017	11/08/2017	OK
III	14/08/2017	15/09/2017	OK
IV	18/09/2017	24/10/2017	
CRONOGRAMA GERAL			
ETAPA	INÍCIO	FIM	SITUAÇÃO
Desenvolvimento	10/04/2017	24/10/2017	OK
Teste	11/09/2017	24/11/2017	OK
Escrita	01/10/2017	10/12/2017	OK
Planejamento	20/11/2017	10/12/2017	

Fonte: Elaborada pelos autores

### 3.1 Tecnologias utilizadas

Para a criação da interface e do robô é necessária, primordialmente, a utilização de softwares capazes de modelar e criar objetos tridimensionais e outros para a criação e design das imagens.

Para atender a essas, foram escolhidos a Unity 3D<sup>2</sup>, PhotoFiltre Studio X<sup>3</sup> e utilizamos a linguagem de programação C#<sup>4</sup>.

#### 3.1.1 Unity 3D

A Unity 3D é uma *engine*, ou seja, um motor de jogo genérico. Ela é uma ferramenta de jogos completa que surgiu para facilitar o desenvolvimento de jogos (GASPAROTTO, n.d).

Atualmente, está sendo utilizada tanto por grandes empresas como por desenvolvedores individuais por ser bastante completa e proporcionar a utilização de

<sup>2</sup> Disponível em: <<https://unity3d.com/pt>>.

<sup>3</sup> Disponível em: <<http://www.photofiltre-studio.com/download-en.htm>>.

<sup>4</sup> Disponível em: <<https://docs.microsoft.com/pt-br/dotnet/csharp/quick-starts/>>.

detalhes mais complexos, como física (colisões, gravidade, etc.) e animações (GASPAROTTO, n.d).

Por esse motivo, utilizamos a Unity3D na elaboração do jogo. Ela abre um leque de possibilidades de criação de jogos com jogabilidade melhor e de forma mais rápida, sem se preocupar com a criação de funções gráficas ou opções físicas dos objetos, etc.

Além disso, outros dois motivos pelas quais utilizamos essa ferramenta foi a possibilidade de desenvolvimento para as principais plataformas, como por exemplo o Windows, Android, IOS, Linux, entre outros e, por ser gratuita. Apesar de ter limitações em relação à versão paga, ela é muito completa e proporciona várias ferramentas que possibilitam o desenvolvimento mais completo, até dispensando a programação em alguns casos.

### 3.1.2 C# (C Sharp)

C# é uma linguagem de programação desenvolvida pela Microsoft como parte da plataforma .NET (MARRONE, 2013). Sua sintaxe foi baseada em C++ (linguagem de programação compilada multi-paradigma), mas possui influências de outras linguagens de programação, principalmente Java (linguagem de programação interpretada orientada a objetos).

O motivo pela qual escolhemos utilizar essa linguagem para a programação ao invés das outras disponibilizadas pela Unity3D foi pelo fato de assemelhar-se com Java. Além disso, o C# é uma das linguagens de programação mais completas e utilizadas do mercado (GASPAROTTO, n.d.).

### 3.1.3 PhotoFiltre Studio X

O PhotoFiltre Studio X é um software para o Windows que possui inúmeras ferramentas para a manipulação de imagens. Ele compreende em diversas funções de visualização, edição, ajuste de brilho, contraste e cores de fotos e etc., contando também com opções de inserção de figuras geométricas. Além disso, ele é um software pequeno e leve, o que possibilitou a facilidade no desenvolvimento do gráfico do jogo.



## 4 DESENVOLVIMENTO

### 4.1 Programação em blocos

A criação do jogo começou pelo desenvolvimento da funcionalidade mais básica que ele deveria ter, a programação em blocos. Para isso, o jogo deveria oferecer uma interface em que o usuário pudesse manipular os blocos de instruções, colocando-os na sequência que quisesse executar as ações e o jogo deveria ser capaz de interpretar e executar cada ação na ordem que fosse determinada pelo jogador.

#### 4.1.1 Manipulação dos blocos

As imagens utilizadas na interface foram criadas utilizando o Photofiltre Studio X. Para fazer a manipulação dos blocos (arrastar pela tela, soltar nas filas de execução, mudar de posição, etc.), foram utilizados os diversos *handlers* que a Unity oferece. Esses *handlers* são classes que oferecem funções que são chamadas automaticamente quando certos eventos ocorrem.

Por exemplo, quando o jogador começa a arrastar um bloco específico, a função *OnBeginDrag* é chamada, ela possui um parâmetro que recebe uma classe *PointerEventData*, que conterá as informações sobre o evento que ocorreu, nesse caso, o início do arrasto.







Ao todo, foram utilizados *handlers* para detectar o início, meio e fim de um arrasto. Após o fim do arrasto, o *handler* *OnDrop* também é chamado pela fila de execução. Ela também fará uso deles para detectar quando o mouse entra e sai dela.

#### 4.1.2 Interpretador

Quando o jogador aperta a tecla “R”, o robô deve começar a executar as ações definidas nas filas de execução. Esse processo de “tradução” do que está na interface para ações do robô é dividido entre duas classes.

A primeira é o interpretador propriamente dito, ela irá juntar em uma lista todas as ações que deverão ser executadas e enviar para a segunda classe que participa do processo, a do robô. Na tabela abaixo, pode-se ver a função de cada bloco.

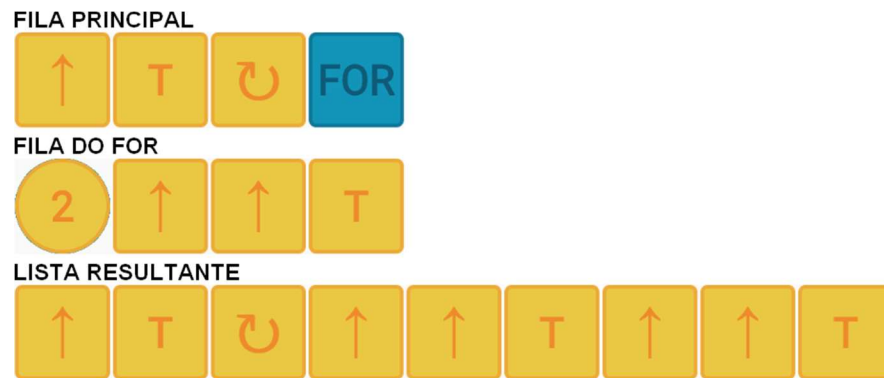
Tabela 1 – Funções dos blocos

BLOCO	FUNÇÃO
	Andar para frente
	Girar em sentido horário
	Girar em sentido anti-horário
	Ativar bloco de teletransporte
	Ativar fila do <i>if</i>
	Ativar fila do <i>for</i>

Fonte: Elaborada pelos autores

O único detalhe é que quando o interpretador encontra um bloco de *for*, ele já irá colocar os blocos da fila específica repetidos quantas vezes determinada pelo jogador. Na figura abaixo é possível observar o funcionamento do interpretador.

Figura 3 – Funcionamento do interpretador



Fonte: Elaborada pelos autores

Ao receber a lista, o robô começará a executar as ações conforme a ordem dada. Nesse lado do processo também há um detalhe: quando o robô encontrar uma instrução de *if* (que não é interpretada anteriormente), ele irá verificar qual a cor do bloco que ele está sobre. Se for algum bloco de cor, ele irá adicionar as ações dos blocos que estiver na fila da respectiva cor ao início da lista de ações e irá executá-las. Se não for, ele não fará nada e passará para a próxima ação.

Um conceito imprescindível para implementar essa funcionalidade é o de *coroutines*. Quando uma função é chamada, ela é executada completamente antes de retornar. Isso gerava um problema quando o robô ia executar as ações, pois todas

elas eram executadas em um único *frame*. Dessa forma, o jogador não tinha um retorno efetivo das consequências que cada bloco tinha no movimento geral do robô.

Para resolver isso, usou-se as *coroutines*, funções que tem a habilidade de parar a execução do código e retomar de onde parou após executar suas linhas de código. Por exemplo, para andar, o robô usa a função Lerp para movimentar aos poucos para frente.

O problema é que todos os pontos intermediários da transição entre a posição inicial e a final adicionados em um único *frame*, destruindo o efeito de movimento. Com o uso de uma *coroutine*, o código de andar será executar ao longo de quantos *frames* forem definidos. Nesse caso, o robô sairá da posição inicial até a final ao longo de um turno, que tem a duração definida em meio segundo.

## 4.2 Montagem dos níveis

Concomitantemente à implementação da programação em blocos no jogo, os níveis foram sendo montados para realizar os testes do que ia sendo feito. Oito tipos de blocos são utilizados para montar os níveis, sendo que só quatro deles estão presentes em todos os níveis. Eles foram criados utilizando os objetos tridimensionais padrão que a Unity dispõe. As texturas dos blocos (Figura 4) foram criadas utilizando o Photofiltre Studio X.

Figura 4 – Texturas dos blocos



Fonte: Elaborada pelos autores

Para fins demonstrativos, o número de níveis foi limitado a dez. Assim, cada nível foi utilizado para trabalhar uma nova forma de utilizar os blocos de programação, que são apresentados um por um em níveis diferentes, e o último foi feito como um teste final.

### 4.2.1 Blocos especiais

Alguns blocos desempenham papéis que vão além de somente compor o nível. O primeiro deles é o bloco de ouro, que é a linha de chegada em todos níveis. Para verificar se o jogador completou o nível, ao final de cada turno o robô executa uma função que verifica se a posição dele é igual a do bloco de ouro.

Outro bloco que tem uma função especial é o de teletransporte, presentes em duas cores: verde e roxo. Para que o robô soubesse para onde ir ao ativar esse bloco, foi criada uma classe que faz a ligação entre o par de blocos de teletransporte, sendo anexada nos dois. Dessa forma, quando o robô encontra uma ação de teletransportar na lista de execução, ele verifica se o bloco abaixo dele é de teletransporte. Se for, ele irá verificar na classe do bloco qual o outro correspondente e vai para ele.

Os últimos dois blocos que possuem uma função especial são os dois de cor, utilizados no *if*. Nesse caso não é preciso de nenhuma classe especial para os blocos, quando o robô vai verificar se o bloco que ele está pisando é de alguma cor, ele só realiza uma busca no nome do bloco e verifica se encontrar “*red*” ou “*green*”, sem necessidade de criar uma classe somente para indicar a cor do bloco.

### 4.3 Menu

O menu está dividido em três partes: tela inicial, instruções e seleção de níveis. A tela inicial contém botões que levam para as outras duas partes e um para sair do jogo. A parte de instruções é constituída por quatro telas que irão ensinar as teclas de controle do jogo, assim como o funcionamento de cada bloco de programação.

Para fazer o menu de seleção de níveis funcionar, uma classe armazena o nível mais avançado que o jogador chegou. Ao abrir a tela desse menu, cada caixa de seleção de nível verifica se o nível que ela representa é menor ou igual ao nível que está salvo na classe citada anteriormente. Se for, ela irá desativar a imagem de cadeado que cobre o número do nível e irá se tornar interativa, funcionando como um botão.

### 4.4 Robô

O modelo do robô foi baixado gratuitamente do BlenSwap<sup>5</sup> sob a licença CC0, ou seja, a pessoa que criou dedicou o trabalho para o domínio público, permitindo copiar, modificar e distribuir sem precisar pedir permissão ou dar créditos.

## 4.5 Resultados

Ao executar o jogo, o jogador irá se deparar com o menu principal, representado abaixo. Nele, encontrará a logomarca com o nome “ProKid” seguida das opções para ir até as instruções, a seleção de níveis ou para sair do jogo.

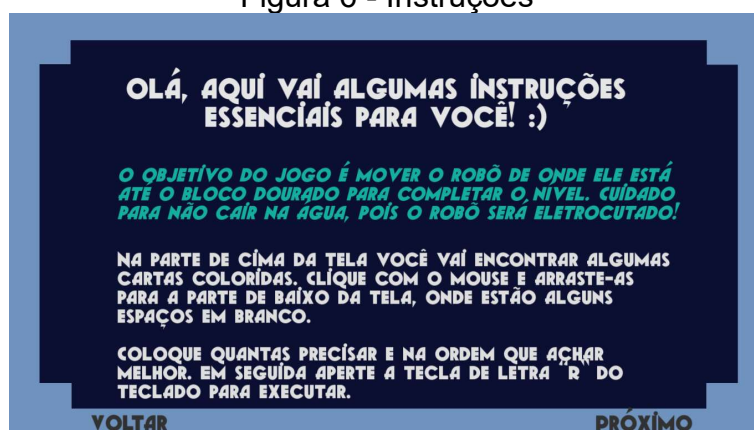
Figura 5 – Tela inicial



Fonte: Elaborada pelos autores

Ao clicar na primeira opção, o jogador será redirecionado para a primeira das quatro telas de instruções (Figura 6), podendo navegar por elas através dos botões na parte inferior.

Figura 6 - Instruções



Fonte: Elaborada pelos autores

<sup>5</sup> Disponível em: <<https://www.blendswap.com/>>.

Se clicar na segunda opção, o jogador será enviado para a tela de seleção de níveis, que pode ser vista na figura abaixo, onde poderá ver quais estão disponíveis para jogar e quais ainda estão bloqueados. Cada vez que o jogador completa um nível, o seguinte será liberado.

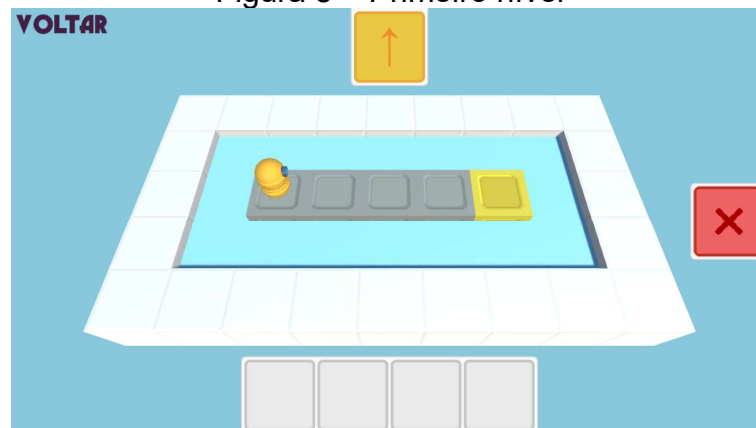
Figura 7 – Seleção de níveis



Fonte: Elaborada pelos autores

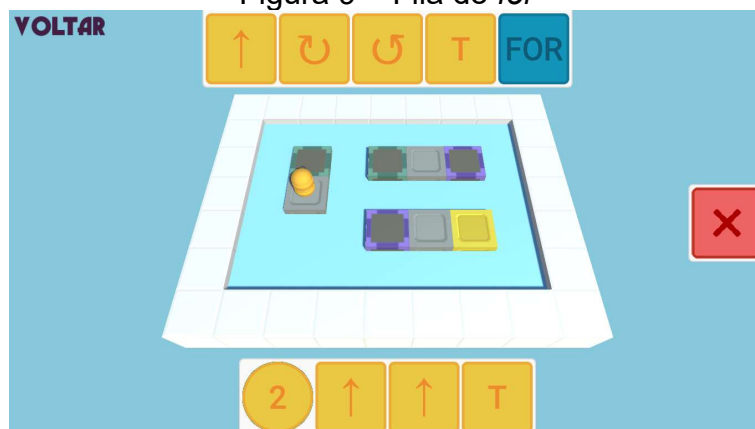
O objetivo de cada nível é levar o robô da posição inicial até o bloco dourado. Para isso, o jogador fará uso da interface de programação em blocos para passar as instruções que o robô deve seguir. Abaixo, na Figura 8, é possível notar o robô em sua posição inicial e o objetivo – o bloco dourado.

Figura 8 – Primeiro nível



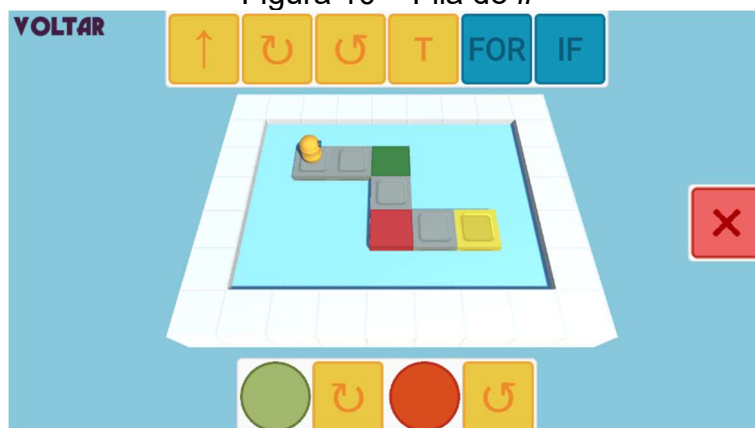
Fonte: Elaborada pelos autores

No nível cinco, o jogador desbloqueia o bloco do *for*, que é utilizado para passar instruções repetidas ao robô. Para isso, uma nova fila de execução também é desbloqueada, podendo ser vista abaixo, na Figura 9.

Figura 9 – Fila do *for*

Fonte: Elaborada pelos autores

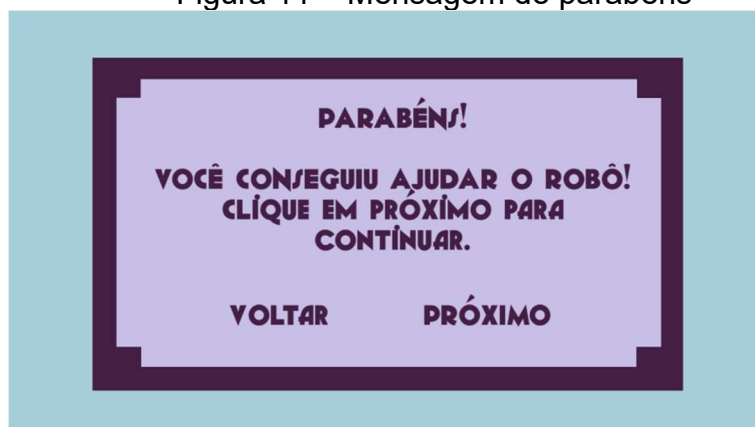
No nível sete (Figura 10), o jogador desbloqueia o último bloco, o do *if*. Semelhantemente ao citado anteriormente, ele também vem acompanhado de uma nova fila de execução, utilizadas para passar as instruções em cada condição.

Figura 10 – Fila do *if*

Fonte: Elaborada pelos autores

Ao completar cada nível, uma mensagem de parabéns é exibida na tela, possuindo também botões para voltar à tela de seleção de níveis ou para ir para o próximo nível diretamente, como mostra a Figura 11.

Figura 11 – Mensagem de parabéns



Fonte: Elaborada pelos autores

## 6 CONSIDERAÇÕES FINAIS

O presente trabalho teve como objetivo principal a criação de um software que auxilie o ensino de lógica de programação para crianças, além de enfatizar a importância do ensino de programação nas escolas. Ainda, o trabalho destaca quais os principais benefícios que o ensino de programação pode trazer para a expansão e aperfeiçoamento das habilidades cognitivas da criança.

É possível observar, também, que programação não é uma prática voltada unicamente para quem deseja trabalhar na área. As próximas gerações irão necessitar dominar o básico para enfrentar os desafios das mais diferentes áreas e também no dia a dia. Desse modo, a programação é de extrema importância para o desenvolvimento pessoal e profissional das crianças (LUIZ, 2016).

Por conseguinte, as crianças sentirão prazer em estudar programação com o software desenvolvido, interagindo com dinamicidade e, além disso, despertando a curiosidade, buscando novas ferramentas para aprendizagem.

### 6.1 Post-mortem

Ao final do desenvolvimento, podemos ponderar o que deu certo e o que deu errado na construção desse jogo. Uma coisa que certamente ajudou muito foi ter planejado tudo que o jogo teria antes de começar o projeto, pois sempre que terminávamos algo, já sabíamos qual a próxima coisa que poderia ser iniciada, sem perder tempo pensando no que fazer.

Após planejar tudo o que seria feito, outra coisa que agilizou o desenvolvimento foi dividir quem seria responsável por fazer cada coisa. Por mais que não tenha sido utilizada nenhuma metodologia ágil específica para realizar isso, só a simples divisão de tarefas trouxe grandes proveitos.

Mesmo com esse planejamento, algumas coisas não deram tão bem. O fato de pensar o que fazer, mas não como fazer, fez com que o jogo demorasse mais para ser terminado. Isso porque uma parte dele, a principal, diga-se de passagem, foi refeita, pois após a finalização de uma primeira versão, veio a ideia de implementar a tarefa de uma forma diferente, o que resolveria vários problemas.



Com isso, boa parte do tempo que tinha sido dedicado no desenvolvimento da primeira versão foi em vão. Não dizemos que todo ele, pois mesmo tendo feito várias coisas, outras ainda puderam ser utilizadas, mas principalmente a noção adquirida ajudou muito na criação da segunda versão.

Mesmo assim, conseguimos cumprir o objetivo inicial. O jogo atende aos requisitos que foram pensados previamente, e o mais importante é que durante o processo de desenvolvimento pudemos adquirir diversos conhecimentos novos, seja na parte dos códigos, como o uso das coroutines, seja no uso da Unity, a função dos handlers, etc.

## **6.2 Trabalhos futuros**

A funcionalidade principal do jogo foi desenvolvida, e ele já está pronto para ser aplicado a crianças como está, mas muitas outras coisas ainda podem ser incluídas para melhorar a experiência delas ao testar. Uma das coisas mais interessantes seria colocar retornos sensoriais. Por se tratar de algo voltado para o público infantil, inserir efeitos visuais e sonoros em momentos como quando o robô morre, se teletransporta ou completa o nível, traria uma satisfação maior ao jogador e tornaria tudo mais atrativo para ele.

Também se pode inserir animações na interface de programação. Os blocos de instruções poderiam reagir às interações como o toque, o arrasto ou ao soltá-los nas filas. Durante a execução do código, o bloco que corresponde à ação que está sendo realizada poderia ganhar algum destaque para que o jogador possa identificar mais facilmente onde eventuais erros estão ocorrendo.

Como se trata de apenas um trabalho demonstrativo, a quantidade de níveis é baixa, o que faz com que a progressão da dificuldade seja bastante acelerada. Caso venha a se realizar testes com crianças em algum trabalho futuro, novos níveis podem ser criados para tornar a curva de aprendizagem mais suave e receptiva.

Por fim, mesmo o jogo contando com uma tela de instruções, pequenos tutoriais entre os níveis poderiam melhor passar as informações ao jogador. Dessa forma, a pessoa teria acesso ao conhecimento para passar de nível somente quando ela fosse ser necessária. Isso se tratando em melhorias para o que já está feito, pois várias mecânicas novas podem ser pensadas e acrescentadas ao jogo para tornar a experiência ainda mais rica para o aprendizado das crianças.

## REFERÊNCIAS

CASTRO, Aline. **A importância de estimular o ensino da lógica, tecnologia e matemática às nossas crianças:** O futuro da nação está nos pequeninos. Disponível em: <<https://trendr.com.br/a-importancia-de-estimular-o-ensino-da-logica-tecnologia-e-matematica-as-nossas-criancas-ce3818b0f678>>. Acesso em: 07 de dezembro de 2017.

FERNANDO, Champika. **Celebrating 50 Years of Kids Coding.** Disponível em: <<https://www.google.com/doodles/celebrating-50-years-of-kids-coding>>. Acesso em: 06 de dezembro de 2017.

FURIA, Fernanda. **Crianças que programam: uma habilidade para toda a vida.** Disponível em: <<http://www.playground-inovacao.com.br/criancas-que-programam-uma-habilidade-para-toda-a-vida/>>. Acesso em: 06 de dezembro de 2017.

GADOTTI, Moacir. **Perspectivas Atuais da Educação.** Disponível em: <<http://www.scielo.br/pdf/spp/v14n2/9782.pdf>>. Acesso em: 06 de dezembro de 2017.

GASPAROTTO, Henrique. **Unity 3D: Introdução ao desenvolvimento de games.** Disponível em: <<https://www.devmedia.com.br/unity-3d-introducao-ao-desenvolvimento-de-games/30653>>. Acesso em: 06 de dezembro de 2017.

LUIZ, Pedro. **A Importância do Ensino da Programação nas Escolas.** Disponível em: <<https://blogcomputacaoeducacao.blogspot.com.br/2016/05/a-importancia-do-ensino-da-programacao.html>>. Acesso em: 07 de dezembro de 2017.

SILOTTO, Reinaldo. **Programação baseada em Blocos pode ser considerada linguagem de programação?** Disponível em: <<https://imasters.com.br/desenvolvimento/programacao-baseada-em-blocos-pode-ser-considerada-linguagem-de-programacao-2/?trace=1519021197&source=single>>. Acesso em: 04 de janeiro de 2018.

SUPERGEEKS. **Especialistas destacam a importância de crianças aprenderem a programar.** Disponível em: <<https://news.supergeeks.com.br/especialistas-destacam-a-importancia-de-criancas-aprenderem-a-programar-b9affac27622>>. Acesso em: 07 de dezembro de 2017.

WEINTROP, David; WILENSKY, Uri. To block or not to block, that is the question: students' perceptions of blocks-based programming. In: INTERNATIONAL CONFERENCE ON INTERACTION DESIGN AND CHILDREN, IDC'15, Medford (Estados Unidos). **Anais...**, p. 199-208, 2015.