UNIVERSIDADE DO VALE DO TAQUARI - UNIVATES
SISTEMAS DE INFORMAÇÃO

HELPBLOCK: UMA FERRAMENTA WEB BASEADA NA BIBLIOTECA BLOCKLY PARA APOIO AO ENSINO DE ALGORITMOS

Eduardo Rodrigues Gomes

Lajeado, novembro de 2017

Eduardo Rodrigues Gomes

HELPBLOCK: UMA FERRAMENTA WEB BASEADA NA BIBLIOTECA BLOCKLY PARA APOIO AO ENSINO DE ALGORITMOS

Monografia apresentada na disciplina Trabalho de Conclusão de Curso Etapa II, na linha de formação em Sistemas de Informação, da Universidade do Vale do Taquari - UNIVATES, como parte da exigência para a obtenção do título de Bacharel em Sistemas de Informação.

Orientadora: Me. Maria Claudete S. Wildner

Lajeado, novembro de 2017

RESUMO

Com o avanço da tecnologia novas profissões foram surgindo, principalmente na área da computação. Neste contexto, a necessidade de novos profissionais para ocuparem essas vagas é algo que vem crescendo consideravelmente. O grande problema está na elevada taxa de desistências e reprovações nas disciplinas iniciais destes cursos, o que se deve à falta de preparo dos alunos nas séries anteriores. O raciocínio lógico e técnicas de programação geralmente não são assuntos trabalhados no ensino médio, o que faz com que ao se depararem pela primeira vez com códigos e linguagens complexas, os alunos acabem se assustando e optando por trancar a faculdade ou até mesmo trocar de curso. Pensando em diminuir essa taxa de desistências foram desenvolvidas técnicas de programação visual, que focam em exercitar o raciocínio lógico do aluno, deixando a parte de códigos e sintaxe para um segundo momento. Esta monografia teve como propósito o desenvolvimento de uma ferramenta web que utiliza a biblioteca blockly para possibilitar aos alunos a resolução de exercícios e criação de algoritmos através de blocos lógicos conectáveis, o nome da ferramenta criada é HelpBlock. A mesma foi testada em uma turma de algoritmos, onde foi possível constatar um aumento no interesse dos alunos em relação a exercícios propostos, assim como uma boa aceitação na forma alternativa de aprender programação.

Palavras-Chave: Blockly. Ensino de algoritmos. Programação visual. Ferramenta web

ABSTRACT

With the advancement of technology new professions have been emerging, especially in the area of computing. In this context, the need for new professionals to fill these vacancies is something that has been growing considerably. The main problem is the high dropout rates and disapprovals in the initial subjects of these courses, due to the lack of preparation of the students in the previous grades. Logical reasoning and programming techniques are generally not subjects in high school, which means that when faced with complex codes and languages for the first time students become scared and choose to lock college or even change course . Thinking about decreasing this dropout rate, visual programming techniques were developed, which focus on exercising the student's logical reasoning, leaving the code part and syntax for a second moment. This monograph aimed to develop a web tool that uses the blockly library to enable students to solve exercises and create algorithms through connectable logical blocks, the name of the tool created is HelpBlock. It was tested in a class of algorithms, where it was possible to observe an increase in students' interest in relation to proposed exercises, as well as a good acceptance in the alternative way of learning programming.

Keywords: Blockly. Teaching algorithms. Visual programming. Web tool

LISTA DE FIGURAS

LISTA DE QUADROS

LISTA DE SIGLAS E ABREVEATURAS

CSS	*Cascading Styling Sheets*
HTML	*HyperText Markup Language*
BD	Banco de dados
RF	Requisitos Funcionais
RNF	Requisitos Não Funcionais
JSP	*Java Server Pages*

SUMÁRIO

1 – INTRODUÇÃO

Os cursos de computação tem como conteúdo inicial o ensino de algoritmos que é realizado na disciplina denominada de Algoritmos e Programação. Porém a mesma tende a intimidar os alunos, causando muitas vezes a desistência dos mesmos.

Segundo Giraffa e Mora (2013), a evasão nas disciplinas dos níveis iniciais dos cursos de computação não é um fato recente. Esta situação já vem sendo estudada por vários autores nacionais e internacionais, e causa preocupação tanto para docentes quanto para a sociedade em geral.

Segundo Ipesen (2013) diante da percepção das dificuldades existentes, as formas de ensinar algoritmos se tornaram alvo de vários estudos que tem como objetivo diminuir as adversidades dos alunos. Assim, ao se deparar com disciplinas iniciais de programação acabam sofrendo um choque de realidade, que em certos casos acaba resultando na desistência do aluno, algumas vezes por não ser o que esperavam e outras por ter muita dificuldade na compreensão do assunto apresentado.

Essa dificuldade em compreender o assunto geralmente é resultante da falta de preparação que o aluno recebeu nos anos anteriores, assim o raciocínio lógico somado com os diferentes códigos presentes nas linguagens de programação acaba se tornando um agravante na decisão de desistência do aluno. A disciplina de

algoritmos e programação mesmo sendo uma disciplina básica e introdutória para os cursos de computação, acaba causando pânico nos alunos. Isso por exigir que o aluno resolva problemas utilizando raciocínio lógico e linguagens de programação. Assim o aluno além de ter que se preocupar em desenvolver sua capacidade lógica ainda precisa se preocupar em aprender e entender códigos que na maioria das vezes nunca teve nenhum tipo de contato. Todos esses fatores acabam sobrecarregando o aluno e dificultando a compreensão dos algoritmos apresentados.

Algoritmos e programação é sem dúvida a base da área da computação, sendo assim, mesmo o aluno não gostando de programar é importante que ele ao menos consiga entender processos e linhas de código. Por ser uma disciplina inicial de curso, geralmente é formada por turmas de muitos alunos, o que acaba dificultando o trabalho do professor que por sua vez, não consegue dar a devida atenção para cada um dos alunos. Assim, alguns alunos mais retraídos deixam de perguntar e não tem suas dúvidas esclarecidas, ao decorrer do curso acabam por ter um rendimento muito inferior ao restante dos colegas que tiveram mais auxílio ou que tem uma maior facilidade em entender os algoritmos.

Almeida et al. (2002) cita que existe uma falta de interesse dos alunos. Motil e Epstein (2000) afirmam que a grande maioria das linguagens utilizadas em cadeiras iniciais da computação é muito complexa e que é mais adequada a locais de trabalho, também afirmam que mesmo os melhores alunos podem ficar frustrados com os problemas de sintaxe que podem ocorrer, e assim acabarem desistindo da área da computação e irem para outra área como: matemática, negócios ou qualquer outra escolha.

Jekins (2002) cita algumas causas do insucesso em disciplinas de programação, como o baixo nível de abstração, falta de competência de resolução de problemas, métodos de ensino nada adequados para o estilo de aprendizagem dos alunos e ainda que as linguagens de programação utilizam sintaxes mais adequadas a profissionais e não iniciantes.

Considerando que várias pesquisas apontam um número elevado de desistências e reprovações em disciplinas de programação, principalmente na

disciplina de algoritmos, foram desenvolvidas algumas ferramentas e linguagens como Scratch, Portugol, Visualg, Blockly que tem como objetivo facilitar a compreensão e aprendizagem dos alunos que são expostos a programação pela primeira vez, facilitando o aprendizado na parte de lógica de programação.

Sendo assim, este trabalho tem como tema o desenvolvimento de um ambiente web que utilizará a biblioteca blockly para permitir que os alunos resolvam exercícios utilizando uma plataforma de programação visual. O trabalho será dividido em 3 etapas, sendo elas o levantamento de conhecimento, o desenvolvimento da ferramenta proposta, e os testes da ferramenta. Quanto aos testes, serão realizados em uma turma de algoritmos da Universidade Univates, através de um pré-teste, uma intervenção pedagógica e um pós-teste.

1.1 - Tema

Desenvolvimento de uma ferramenta que utilize programação visual para auxiliar alunos e professores na disciplina de algoritmos e programação.

1.1.1 - Delimitação Do Tema

Este trabalho abordará o desenvolvimento de uma ferramenta, que terá como objetivo diminuir a alta taxa de evasão nos cursos de computação, especialmente na cadeira de algoritmos e programação. Para isso será utilizado o conceito de programação visual, através da utilização da biblioteca visual Google Blockly, que é baseada em blocos.

1.2 - Objetivo Geral

Disponibilizar uma ferramenta que utilize programação gráfica para facilitar e auxiliar o ensino de algoritmos nas turmas iniciais dos cursos de computação.

1.2.1 - Objetivos Específicos

- Estudar ferramentas já existentes
- Construir um ambiente web utilizando HTML5, CSS3, JavaScript e a biblioteca blockly;
- Elaborar exemplos para apresentar a ferramenta.
- Aplicar a ferramenta com alunos da disciplina de algoritmos

1.3 - Justificativa

Pesquisas apontam que o mercado atual sofre uma deficiência de profissionais na área da computação, devido à alta taxa de desistência nos cursos de informática. Um dos motivos apontados é o fraco ensino das matérias de exatas e raciocínio lógico no ensino básico, faltando para a maioria dos alunos competências para a resolução de problemas.

A disciplina de algoritmos e programação é a que recebe maiores apontamentos por causar a desistência de alunos dos cursos de T.I. (Tecnologia da Informação). O aluno tende a se assustar quando exposto a uma tela branca com um cursor piscando esperando que linhas de código sejam escritas para resolver exercícios propostos pelos professores, sendo que muitas vezes não sabe nem o que é um código.

Visando melhorar essa primeira experiência do aluno com a programação, este trabalho objetiva criar uma ferramenta onde o orientador/professor vai ter liberdade de adicionar exercícios e o aluno poderá resolver trabalhando apenas com blocos conectáveis que possuem uma linha de código predefinida, não tendo que se preocupar com o código e sim com o raciocínio lógico.

2 – REFERENCIAL TEÓRICO

Neste capítulo será apresentado o conceito de ensino de algoritmos, e também formas alternativas que foram desenvolvidas com o objetivo de auxiliar na aprendizagem dos alunos.

2.1 – Ensino de algoritmos

Raciocínio lógico, lógica, lógica de programação e linguagem de programação são alguns conceitos básicos necessários para o ensino de algoritmos. Lógica é a arte de pensar corretamente e que ensina a colocar em ordem o pensamento (FORBELLONE, 1993). Raciocínio lógico é a capacidade de analisar dados e compreender o que há de essencial e de geral e formar uma relação entre eles (PORTAL, 2004).

Lógica de programação é a técnica de ordenar pensamentos para alcançar determinado objetivo. Pode ser representada por inúmeras linguagens de programação, como por exemplo, C++, Java, Javascript, entre outras. Porém essas linguagens envolvem muitos detalhes computacionais, então para melhor representar o raciocínio da lógica de programação são utilizados algoritmos (GUIMARÃES, 2004).

Linguagem de programação é um conjunto de símbolos finitos que são utilizados para escrever programas de computador. Um programa utiliza informações de entrada para definir o que o computador deve fazer para obter uma determinada saída (GUIMARÃES, 2004).

A disciplina de Algoritmos é a base dos cursos de graduação na área de computação, e é considerada imprescindível e obrigatória. Definições de algoritmos podem variar dependendo do autor, porém a ideia principal se mantém, que é receber entradas, processá-las e produzir saídas. Para CORMEN (2002), algoritmo é uma sequência de passos computacionais bem definidos que geram uma saída através do processamento de uma entrada.

Da mesma forma que alguns alunos passam por dificuldades para aprender, os professores acabam tendo dificuldades de ensinar o conteúdo. Para (NOBRE, 2002), algumas dificuldades pelas quais os professores do ensino de programação passam são:

- Reconhecer as habilidades inatas de seus alunos;
- Propor técnicas de soluções de problemas;
- Aproveitar a capacidade que o aluno tem em abstrair, tanto na escolha de estruturas de dados a serem utilizadas quanto na busca de possíveis soluções.
- Despertar o sentimento de cooperação e colaboração entre os alunos.

A dificuldade de compreensão e aplicação da lógica de programação pela qual os alunos passam é uma das maiores causas de reprovação e desistência no decorrer do curso (MARTINS, 2003). E quando essa dificuldade não é solucionada a tempo, acaba resultando em prejuízos enormes aos discentes e docentes da instituição, tornando difícil e lenta a formação acadêmica do aluno e, muitas vezes, incompatível com a dificuldade das atividades práticas que lhe serão exigidas durante o curso.

Segundo (TALL, 2002), o computador oferece o algoritmo, mas é o aluno que oferece o raciocínio e que precisa saber como utilizar o algoritmo para encontrar a melhor solução possível. Os alunos tendem a pensar que o mais importante é chegar no resultado certo, mas na verdade o importante é chegar no melhor resultado.

O desenvolvimento de algoritmos requer habilidades de raciocínio lógico, matemática e interpretação. Assim quando o aluno se depara com dificuldade em

qualquer um desses pontos e o professor não consegue identificar isso, o aluno acaba optando por utilizar algoritmos prontos que foram feitos por colegas ou apresentados pelo próprio professor, e assim deixando de lado o desenvolvimento de seu próprio algoritmo, o que acaba resultando em algoritmos estáticos, fechados e sem significado algum ao aluno (MUNIZ, 2001).

A diferença de experiências, habilidades e origens dos alunos acabam gerando resultados distintos mesmo em classes submetidas às mesmas condições de ensino (PIMENTEL, 2003), isso mostra que é necessário buscar técnicas variadas que permitam melhores resultados de aprendizagem.

Considerando a enorme importância que a disciplina de algoritmos tem nos cursos de informática, foram desenvolvidas ferramentas que auxiliam no ensino das disciplinas de programação. Nos subcapítulos seguintes serão apresentadas algumas das ferramentas existentes.

2.2 Ferramentas alternativas de auxílio a programação

Noschang et. al (2014) cita que o ensino da programação inicial dos cursos deve focar no desenvolvimento da lógica de programação dos alunos e a capacidade de resolver problemas, deixando a aprendizagem de uma linguagem de programação para um segundo momento. Também argumenta que as IDEs (Integral Development Environment) profissionais disponíveis (Netbans, Eclipse, entre outras.) estão longe de ser a melhor opção, pois na sua maioria apresentam suas opções e mensagens no idioma inglês, o que acaba sendo um obstáculo a mais na jornada de aprendizado dos estudantes.

Outro problema dos IDEs profissionais é a quantidade de opções de configuração que possuem, isso pode acabar assustando ainda mais os iniciantes, juntamente com mensagens de erro que são apresentadas no idioma inglês e na maioria das vezes acaba não fazendo nenhum sentido para o aluno. Isso acaba sendo um problema pois toda mensagem de erro gerado pelo IDE deveria ser interpretada e utilizada para ajudar que o aluno, porém dessa forma acaba tendo o resultado oposto do esperado.

Uma das maneiras encontradas para melhorar o desempenho dos alunos nas disciplinas de programação foi o desenvolvimento de ferramentas para auxiliar na programação. Essas ferramentas tem como objetivo apresentar ambientes de boa usabilidade aos aluno, proporcionando uma melhor experiência de cada um deles com a programação. Onde cito: Portugol, VisualG3, Scratch e Blockly.

2.2.1 Portugol

Também conhecido por português estruturado, o Portugol nada mais é que um pseudocódigo escrito em português. É considerado uma linguagem de programação introdutória por trazer comandos em português e também utilizar operadores aritméticos, relacionais e outros. O fato de ser em português ajuda a resolver os problemas com a língua inglesa que possa estar prejudicando o aprendizado do aluno. Isso pois, mesmo sabendo a importância da língua inglesa, é importante que o aluno consiga focar em aprender um conteúdo de cada vez, assim o aprendizado de novas línguas pode ser deixado para um outro momento (Noschang et. al 2014).

No quadro 1 é apresentada uma pequena comparação entre alguns comandos presentes na linguagem Java que foram traduzidos e se encontram no Portugol.

Quadro 1 – Comparação entre comandos em Portugol e Java.

Comando	**Portugol (Visualg)**	**Java**
Variável tipo inteiro	I: inteiro	short i; Int i; long i; byte i;
Variável tipo real	d: real	float d; double d;
Variável tipo lógico	b: logico	boolean b;
Variável tipo texto	s: caractere	char s; String s;
Variável tipo vetor	v: vetor[linha1..linhaN] de tipo	Tipo[] v = new tipo[linhas];
Variável tipo matriz	m: vetor[linha1..linhaN, coluna1..colunaN] de tipo	Tipo[][] m = new tipo [linhas][colunas];
Comando se-então-senão	se condição então ...	If (condição) { ...

	senão
	...
	Fimse
		} else {
		....
		}

Fonte: Do autor, 2017

Além dos comandos citados no quadro anterior, também existem outros comandos como por exemplo os de input e output (entrada e saída):

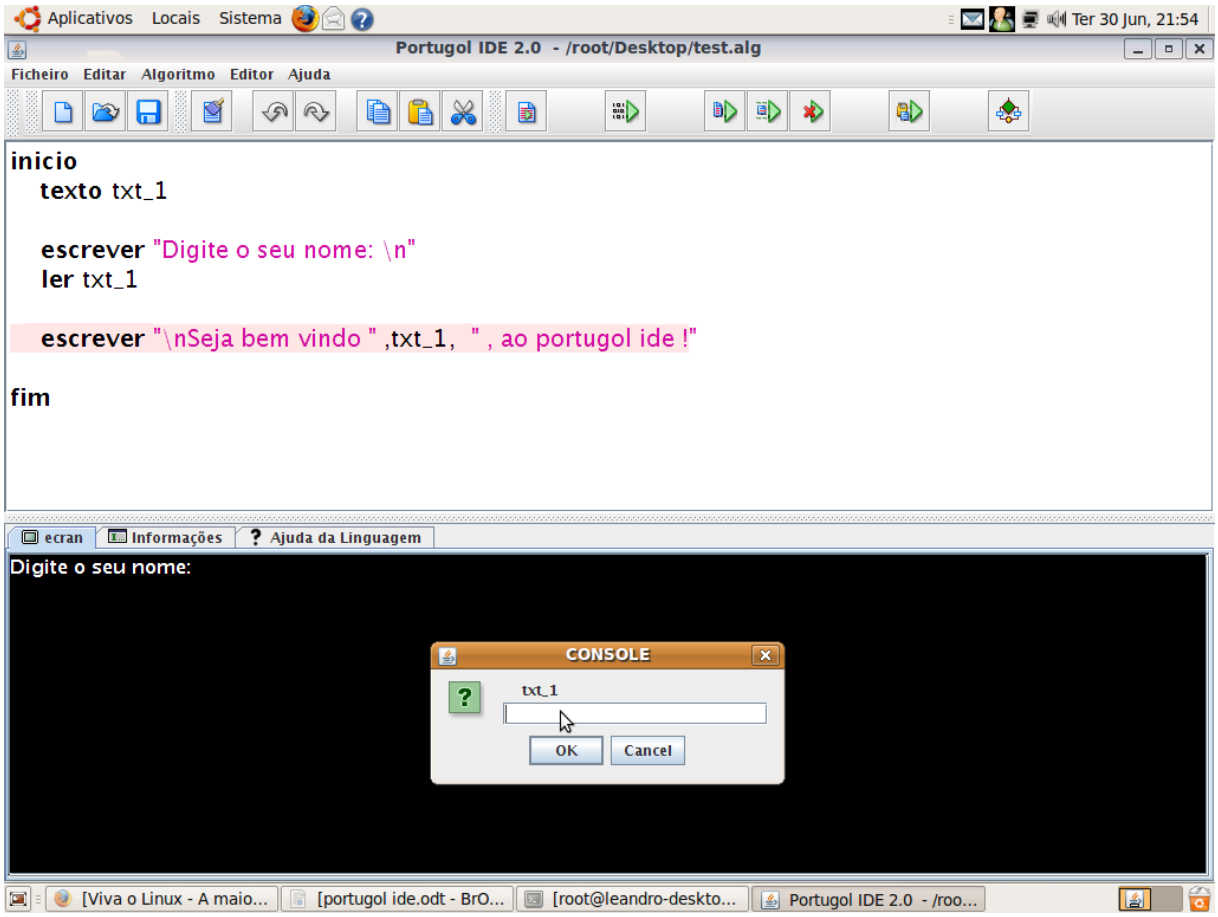
- LER – Comando de entrada que permite a leitura de variáveis de entrada.

- ESCREVER – Comando de saída que exibe uma informação na tela do monitor.

- IMPRIMIR- Comando de saída que envia uma informação para a impressora.

A figura 1 mostra um algoritmo desenvolvido na ferramenta PortugolIDE, que tem como objetivo solicitar ao usuário que digite seu nome. Primeiro é usado o comando “inicio” para identificar o começo do algoritmo, e também é criado uma variável do tipo texto chamada de txt_1, em seguida o comando “escrever” que serve para imprimir uma mensagem na tela, no caso “Digite o seu nome”. Após isso é utilizado o comando “ler” txt_1, este servira para solicitar que o usuário digite seu nome. Depois de obter o nome do usuário é utilizado mais uma vez o comando “escrever” seguido de uma mensagem de “Boas Vindas” e o nome digitado pelo usuário que ficou armazenado na variável txt_1. E ao final o comando “fim” que indica o final do algoritmo.

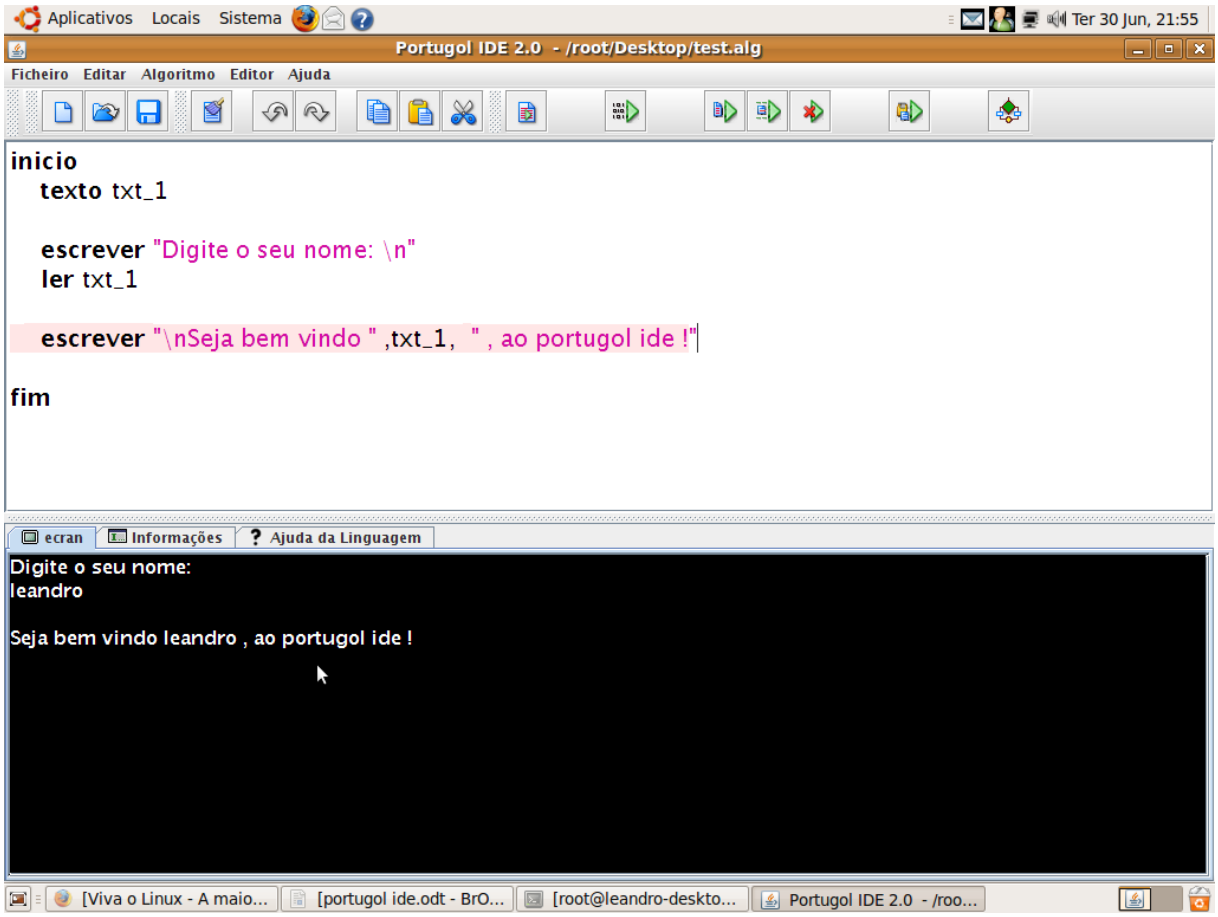
Figura 1 – Exemplo tela 1 criado com a ferramenta Portugol IDE.



Fonte: Teixeira (2009), artigo Portugol IDE

Já na figura 2 podemos ver que assim que o usuário informou seu nome o algoritmo imprimiu na tela a frase que havia sido programada anteriormente, completando com o nome digitado.

Figura 2 – Exemplo tela 2 criado com a ferramenta Portugol IDE.



Fonte: Teixeira (2009), artigo Portugol IDE

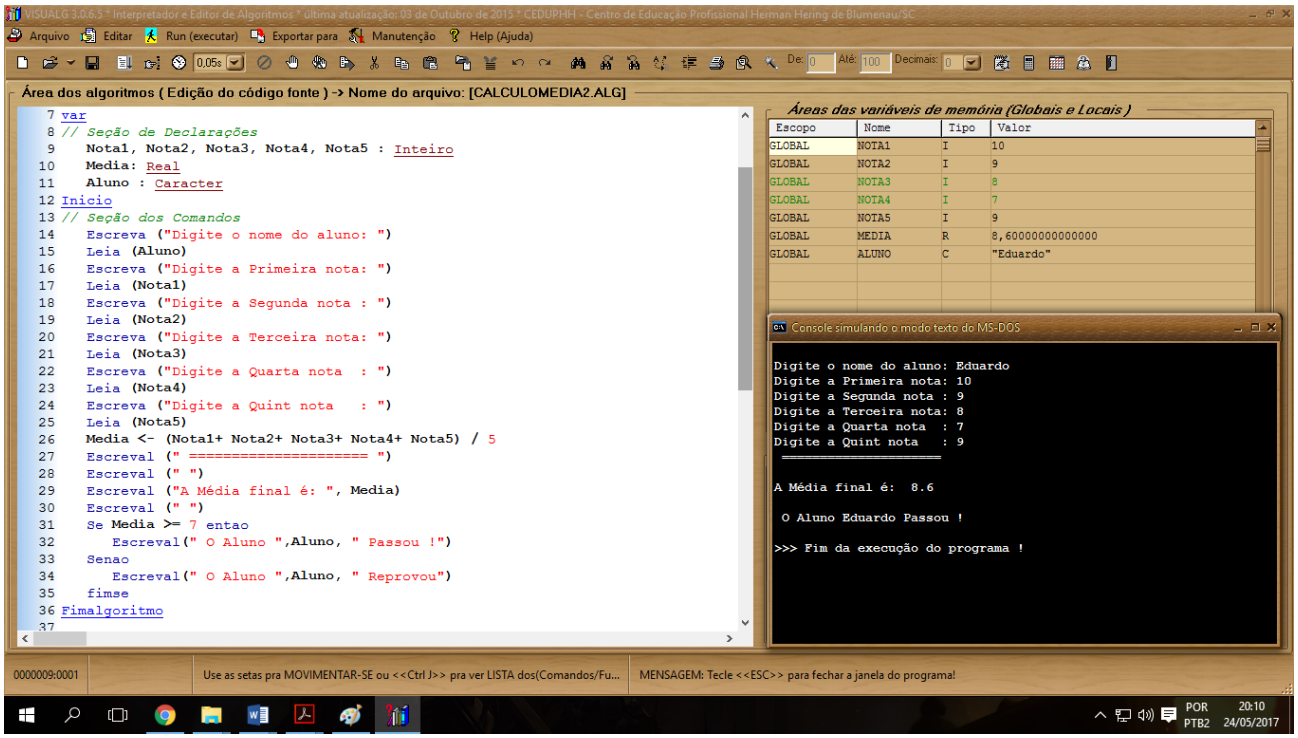
2.2.2 Visualg3

O Visualg é um programa que permite aos alunos trabalhar com pseudocódigos. Ele permite criar, editar, interpretar e também executar algoritmos em português estruturado (Souza, 2009).

Com o tempo o programa foi sendo melhorado até chegar na versão Visualg3, que possui recursos como simulação da tela do MS-DOS, visualização de variáveis, “breakpoints”, exporta algoritmo para um código similar em Pascal, e outras características que ajudam na aprendizagem da programação.

Na figura 3 é possível ver a tela principal do programa executando um código que recebe 5 notas do aluno, calcula a média das notas e informa se o aluno foi ou não aprovado.

Figura 3 – Tela principal do Virtualg demonstrando a execução de um pseudocódigo



Fonte: Do autor, desenvolvido a partir da tela principal do Visualg3, 2017

2.2.3 Scratch

Ambiente produzido pelo Lifelong Kindergarten Group do Massachussets Institute of Technology/MIT Media Lab. O Scratch está disponível para download desde Maio de 2007 no site www.scratch.mit.edu.

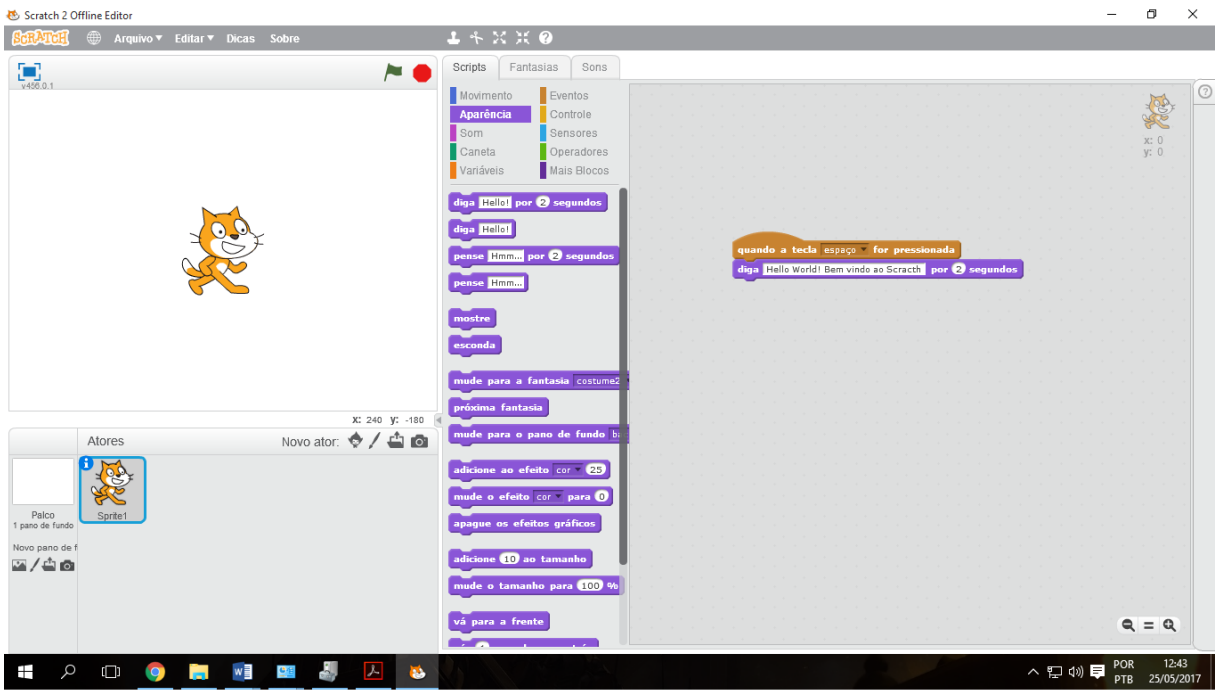
Segundo Alencar et. al (2014), o Scratch é uma linguagem gráfica de programação desenvolvida com o objetivo de ser de fácil manuseio e bastante intuitiva. Por possuir uma IDE, pode ser considerada uma linguagem de programação visual.

O software permite trabalhar com conceitos como: criação de interfaces, números randômicos, sequência, iteração, condições variáveis, sincronia, execução paralela, interação em tempo real, lógica booleana, tratamento de evento entre outros (Ventorini, 2014).

No Scratch é possível trabalhar com imagens, sons, fotos, criar desenhos, alterar aparências, dar movimento aos objetos, fazer com que haja interação entre as coisas. Tudo é feito através da conexão entre blocos que possuem uma lógica de programação. Isso pois é direcionado a crianças do ensino fundamental e busca ser o mais intuitivo possível.

Na Figura 4 é possível ver um exemplo simples onde foi arrastado um bloco de “eventos” que faz com que ao ser pressionada a tecla “espaço”, o programa rode o comando abaixo que consiste em falar a frase “Hello World! Bem vindo ao Scratch” durante 2 segundos.

Figura 4 – Tela inicial do Scratch em execução



Fonte: Do autor, desenvolvido a partir da tela principal do *Scratch*, 2017

2.2.4 Blockly

O blockly é um ambiente de programação visual que funciona na web e tem código aberto, possibilitando o uso em ferramentas próprias de cada desenvolvedor. A biblioteca blockly serve para adicionar um editor de código visual em aplicações web e android. O editor Blockly utiliza blocos gráficos conectáveis para representar os conceitos de código como repetição, condição, loops entre outros. Permitindo assim

que os usuários usem de princípios de programação sem ter de se preocupar com erros de sintaxe (Blockly API, 2017).

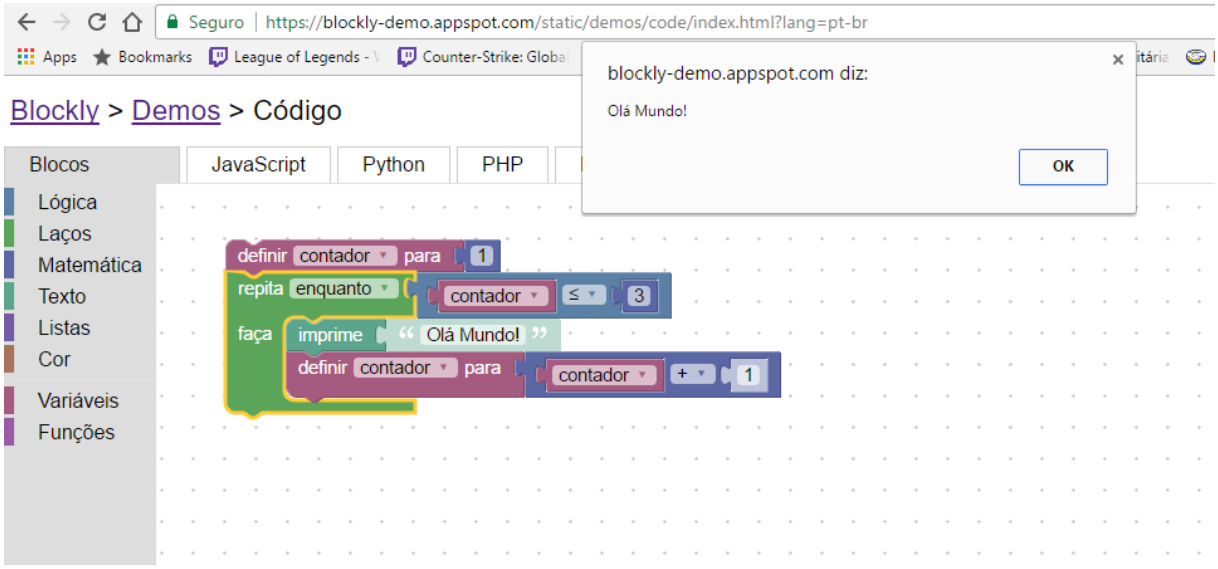
Atualmente o Blockly pode exportar blocos para várias linguagens, incluindo:

- JavaScript
- Python
- PHP
- Lua
- Dardo

O Bockly possui código aberto, permitindo que seja utilizado em aplicações próprias dos usuários, também permite que os programas criados em linguagem visual sejam exportados para linguagem de programação comuns tornando a transição para programação baseada em texto algo fácil. É uma biblioteca de Java Script puro, sem dependências do lado do servidor, altamente personalizável, compatível com todos os principais navegadores: Chrome, Firefox, Safari, Opera e IE (Blockly Google Developers, 2017).

Na figura 5 é apresentado um exemplo simples de um algoritmo feito no blockly, que consiste na definição de uma variável com o nome de “contador” que é inicializada com o valor de 1. Em seguida é adicionado um laço de repetição que irá repetir até que a variável “contador” chegue no valor 3. Em cada repetição o algoritmo irá emitir um alerta com a frase “Olá Mundo!”, e irá incrementar o “contador” em mais 1. Totalizando assim 3 mensagens de alerta na tela.

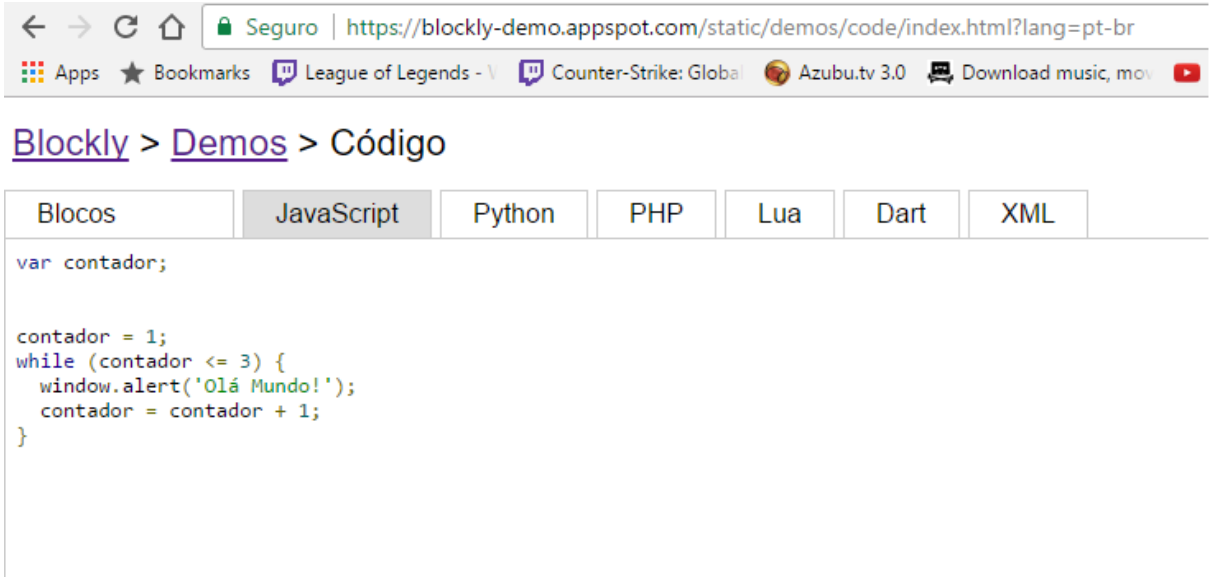
Figura 5 – Algoritmo criado utilizando o blockly



Fonte: Do autor, 2017

Assim como foi citado anteriormente, após criar um algoritmo ou programa utilizando o Blockly, é possível converter o mesmo em código. Assim na figura 6 é demonstrado o algoritmo na versão Java Script em texto.

Figura 6 – Código em Java Script do algoritmo da figura 5



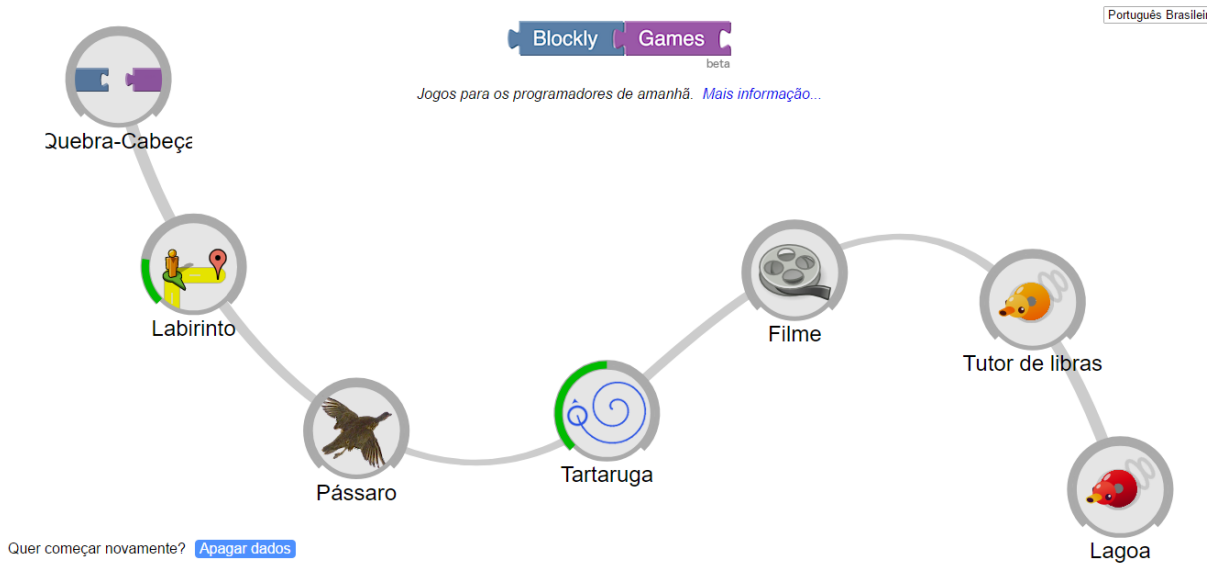
Fonte: Do autor, 2017

Além disso o blockly também possui uma página na web contendo alguns jogos voltados a crianças e iniciantes na área de programação, que tem como objetivo esclarecer

alguns conceitos da ferramenta, como formas de encaixe das peças, loops, condicionais e equações matemáticas.

A tela inicial do Blockly Games pode ser vista na figura 7. Nesta tela é possível ver todos os jogos que são oferecidos pelo Blockly, sendo eles: quebra-cabeça, labirinto, pássaro, tartaruga, filmes, tutor de libras e lagoa.

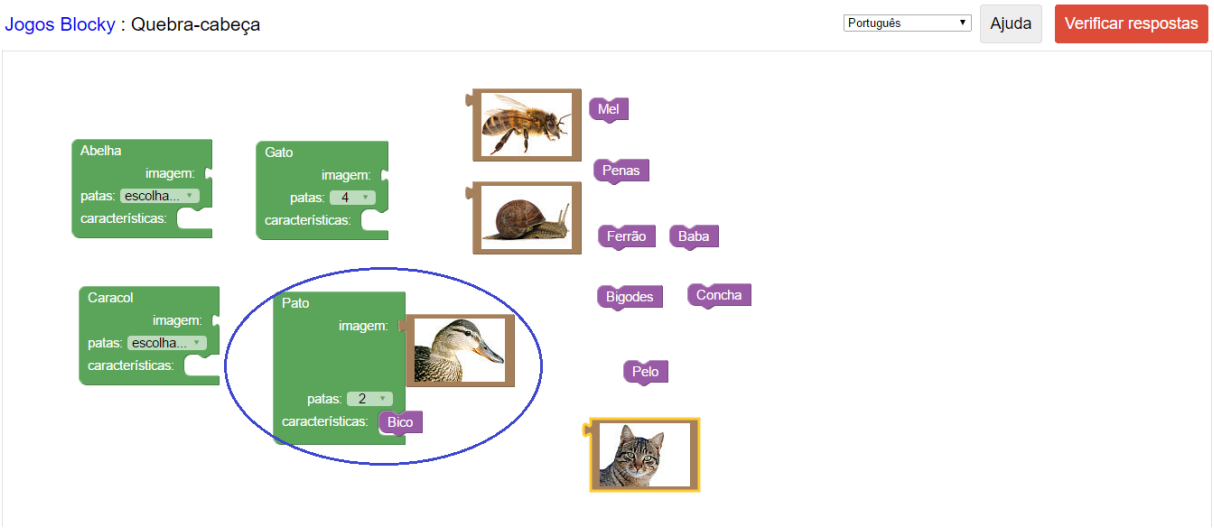
Figura 7 – Tela inicial do site de jogos do Blockly



Fonte: Blockly Games, 2017.

O quebra-cabeça consiste em juntar os blocos conforme as características de cada animal, considerando sua imagem, número de pernas e uma listagem de particularidades de cada um. Isso pode ser visto na figura 8, onde está circulado um dos exemplos prontos.

Figura 8 – Jogo quebra-cabeça do Blockly Games



Fonte: Do autor, 2017

O labirinto tem como objetivo utilizar os blocos e conceitos de loops e condicionais para levar o boneco até a marcação final do labirinto. Conforme você vai passando de nível, a dificuldade vai aumentando.

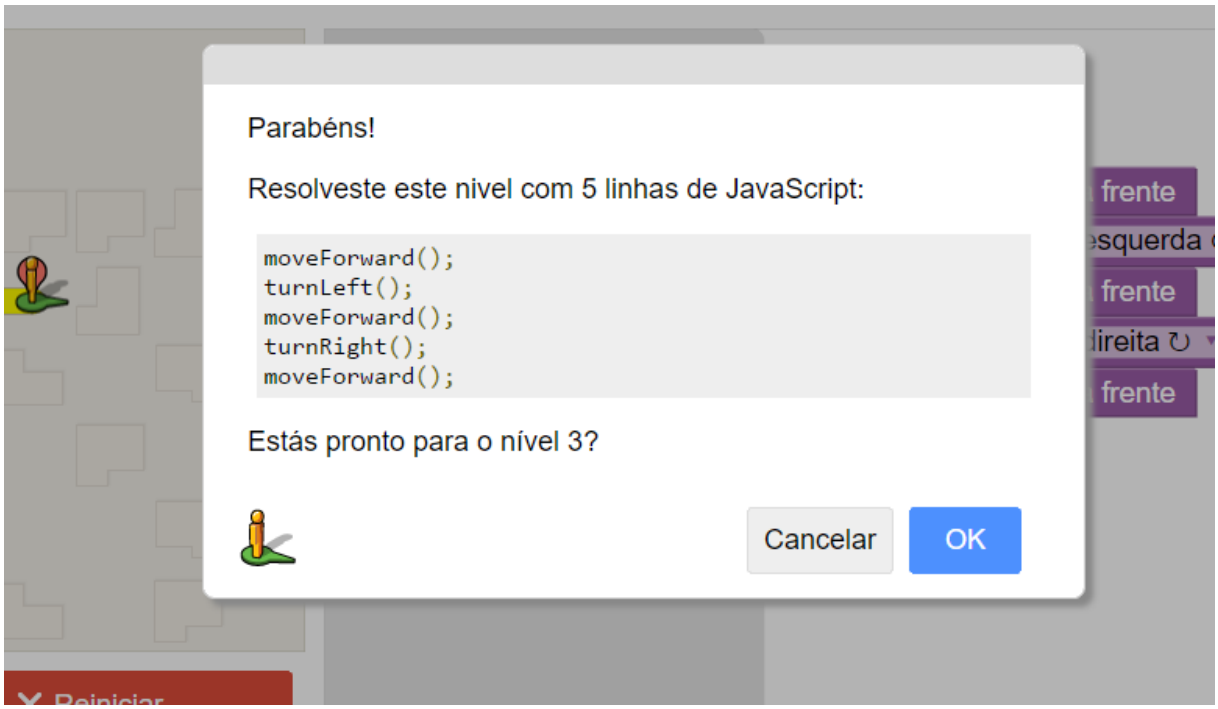
Na figura 9 é possível ver um dos níveis com a solução ao lado. Após completar o nível é apresentado a versão em código para o usuário.

Figura 9 - Jogo labirinto do Blockly Games

Fonte: Do autor. 2017

Na figura 10 é possível ver o código que é apresentado ao usuário após ter completado o nível.

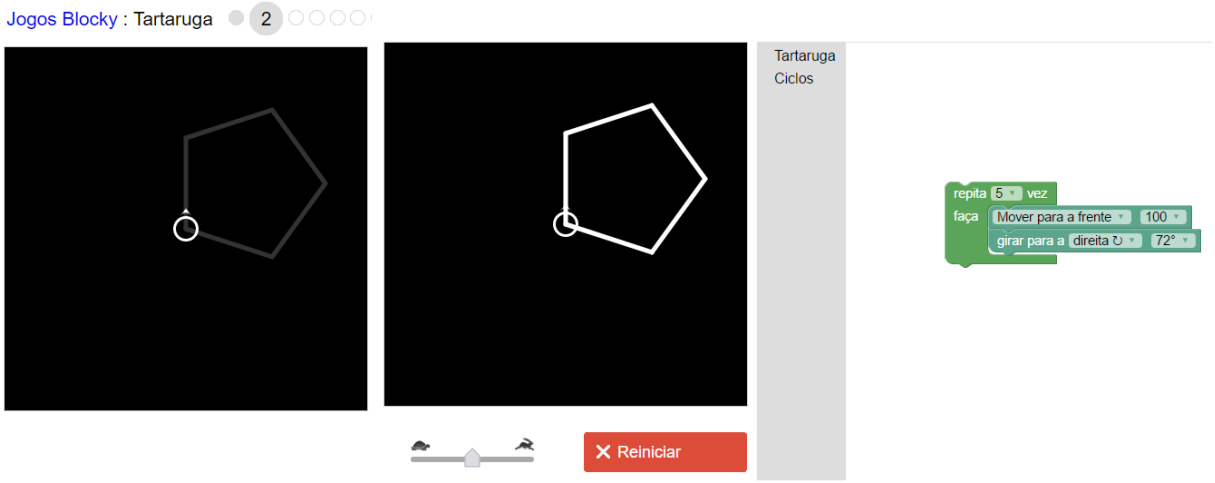
Figura 10 – Código em Java Script da solução encontrada.



Fonte: Do autor, 2017

Todos os níveis apresentam o código após serem resolvidos e antes de passar para o próximo nível. Na figura 11 pode ser visto o jogo da tartaruga, que consiste em usar laços de repetição junto com ângulos para desenhar diferentes formas apresentadas.

Figura 11 – Jogo tartaruga do Blockly Games



Fonte: Do autor, 2017

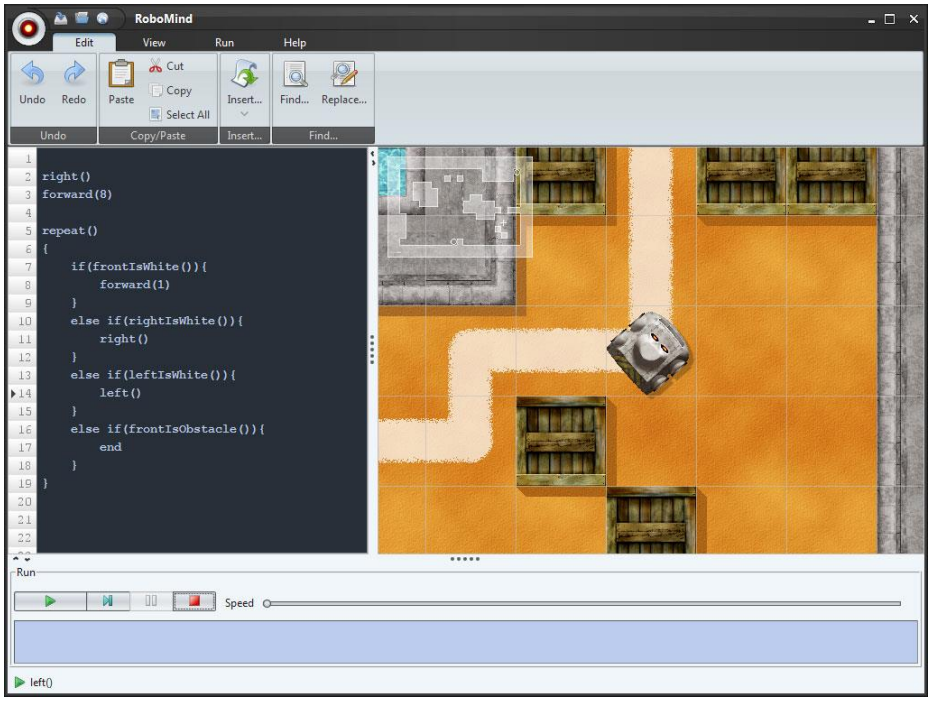
2.3 Trabalhos Relacionados

Nesta parte do trabalho serão apresentados alguns projetos já criados com o objetivo de auxiliar e melhorar o aprendizado de programação para estudantes que estejam iniciando na área de computação e que possuem alguma característica em comum com a ferramenta proposta.

2.3.1 RoboMind

RoboMind é um software que foi criado com o objetivo de auxiliar a educação em tecnologia. Possui sua própria linguagem de script e consiste em programar um robô simulado. Assim ao estarem programando os alunos aprendem lógica, informática e robótica. A linguagem desenvolvida recebeu o nome de ‘ROBO’. É uma linguagem resumida que possui um pequeno número de regras e não depende de um conhecimento prévio, possibilitando o início imediato dos alunos na ferramenta. Apesar de ser uma linguagem resumida, ‘ROBO’ permite que sejam desenvolvidos vários programas muito interessantes (Robo Mind, 2017).

Figura 12 – Tela do RoboMind



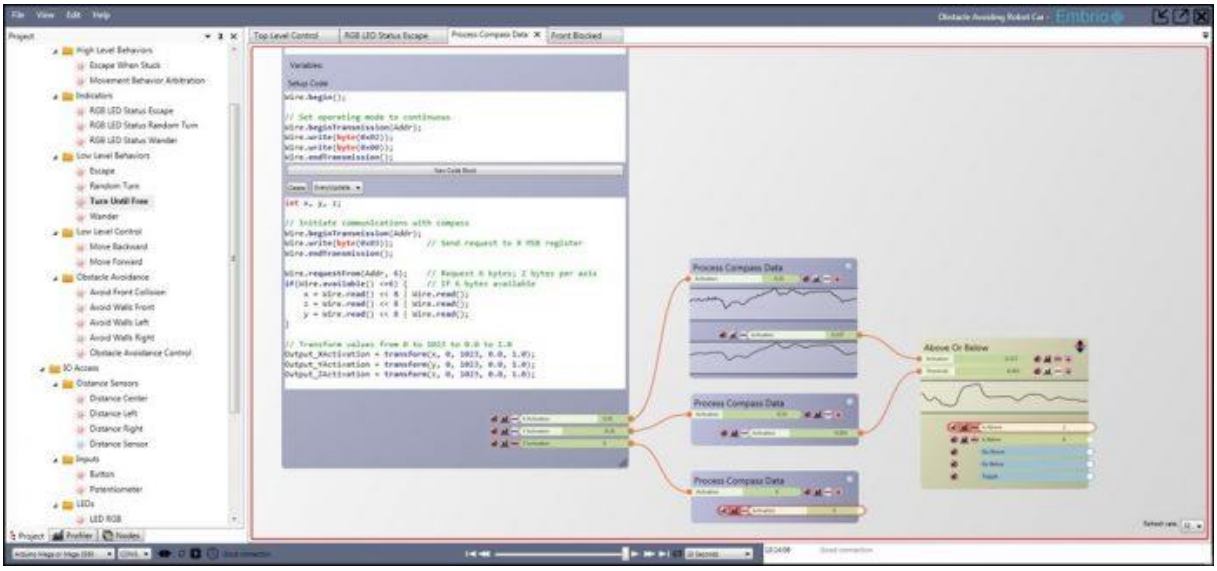
Fonte: RoboMind, 2017

2.3.2 Embrio

Embrio é um ambiente de programação Visual que auxilia na programação de plataformas Arduino. Ele trabalha com blocos conectáveis, porem também permite que o código seja escrito diretamente caso o usuário julgue a codificação seja uma melhor opção que os blocos em algum momento. Também vale ressaltar que ele permite a criação de blocos próprios (Souza, 2015).

A programação tem base em blocos chamados “nodes” e uma arquitetura paralela chamada de “agente based”. O Embrio permite que a interação com a placa Arduino seja em tempo real, possibilitando testes de programa antes de realizar a compilação e o upload da aplicação final (Souza, 2015).

Figura 13 – Tela do Embrio



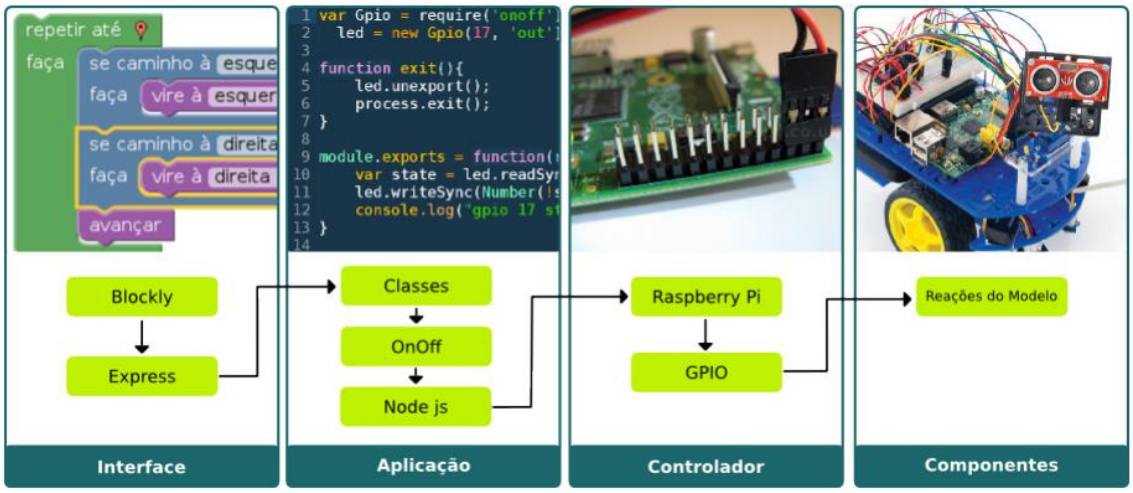
Fonte: Souza, 2015

2.3.3 Raspiblocos

Trata-se de um ambiente de programação visual capaz de traduzir instruções representadas por blocos encaixáveis, que serve para enviar comandos para um modelo robótico. A interface do ambiente foi implementada utilizando a biblioteca Blockly e o modelo robótico com o mini-computador Raspberry Pi. Os componentes do modelo, conectados ao minicomputador são: dois motores com rodas, um sensor de proximidade, um led e um botão (Heinen et. al, 2015).

A arquitetura do Raspiblocos foi dividida em quatro camadas que são: interface, aplicação, controle e componentes. Na Figura 17 é possível ver essa divisão assim como os elementos das camadas e a interação entre eles. Na sequência é descrito mais detalhadamente cada uma das camadas.

Figura 14- Arquitetura e interação entre os componentes da ferramenta



Fonte: Heinen, 2015

2.3.3.1 Camada de Interface

A interface da aplicação foi desenvolvida levando em conta que deveria ser simples e intuitiva para que usuários de diferentes níveis de experiência conseguisse utilizar. Assim foi criada uma interface que pode ser acessada através do próprio navegador web, conectando-se à rede *wi-fi* oferecida pelo minicomputador. A biblioteca Blockly foi utilizada, fazendo com que as instruções disponíveis no ambiente sejam apresentadas através de blocos conectáveis. Os programas gerados pela conexão desses blocos são executados pelo minicomputador, gerando reações no modelo robótico (Heinen et. al, 2015).

A Figura 18 demonstra um programa construído utilizando a interface da ferramenta. O programa em questão tem como objetivo fazer o led do micro robô piscar 10 vezes. As conexões destacadas relacionam-se com: (1) instruções anteriores e subsequentes: (2) blocos que representam valores ou o retorno de uma função: e (3) instruções executadas dentro do escopo do bloco atual.

Figura 15 - Exemplo de programa construído utilizando a interface da ferramenta

Fonte: Heinen, 2015

2.3.3.2 Camada de Aplicação

Ao acessar a aplicação *web*, é apresentado ao usuário a interface de programação visual com blocos contendo as instruções de execução para o modelo robótico. Após construir um programa utilizando estas instruções e executá-lo, é enviado à aplicação mensagens contendo o componente eletrônico e a ação que deve ser realizada. As classes da aplicação, representando cada um desses componentes, oferecem métodos que encapsulam o código responsável por realizar suas respectivas ações. Tais métodos são acionados pela aplicação de acordo com as mensagens recebidas da interface, gerando reações nos componentes eletrônicos do modelo robótico.

2.3.3.3 Camada de Controle

A camada de controle converte as instruções recebidas em corrente elétrica e, depois, em reações no modelo robótico.

Os métodos das classes da Camada de Aplicação utilizam a biblioteca *OnOff* para controlar o estado das portas GPIO do minicomputador. Ao alterar o estado

dessas portas, o minicomputador envia correntes elétricas aos componentes nelas conectados.

2.3.3.4 Camada de Componentes

A camada de componentes é composta por motores de corrente direta, sensores de proximidade, botões e leds. Este conjunto permite que o modelo robótico consiga se movimentar em uma área desviando de obstáculos.

Este capítulo abordou os principais conceitos relacionados ao ensino de algoritmos e ferramentas alternativas de auxiliam na hora de aprender a programar, como o Visualg3, Scratch Blockly, e também apresentou alguns trabalhos relacionados, onde cito o Raspiblocos. No capítulo seguinte será apresentado a metodologia utilizada nesta pesquisa.

3 – METODOLOGIA

Este capítulo apresenta a metodologia utilizada na presente pesquisa, será apresentada em subtítulos que irão descrever a caracterização da pesquisa assim como a organização e o desenvolvimento da ferramenta HelpBlock.

3.1 Caracterização da Pesquisa

O projeto apresentado é de natureza quantitativa e qualitativa. Segundo D'Angelo (2016), a pesquisa quantitativa consiste em quantificar os dados para obter respostas para um questionamento de um problema de pesquisa. Essa quantificação pode ser realizada tanto através de dados levantados por questionários, como também através da análise de resultados. Este tipo de pesquisa geralmente é usado quando se quer confirmar algum tipo de hipótese. D'Angelo (2016) também fala sobre a pesquisa qualitativa, que diferente da quantitativa, se preocupa mais com as considerações particulares dos entrevistados e os dá uma maior liberdade, normalmente é utilizada para validar teste de produtos.

Também foi levado em consideração a aprendizagem significativa, que segundo Ausubel (1963, p. 58) é o aprendizado de uma nova informação de forma substantiva e não arbitrária. As características básicas da aprendizagem significativa são a não-arbitrariedade e a substantividade.

Não-arbitrariedade significa que um material significativo se relacionado de maneira não arbitrária com um conhecimento já existente no indivíduo. O conhecimento prévio é importante para que haja a compreensão e fixação de novas

informações que de certa forma “se ancoram” em conhecimentos relevantes já existentes na estrutura cognitiva da pessoa. Ausubel (1963).

Ainda segundo Ausubel (1963), a substantividade significa que o que o indivíduo aprende é a substância do novo conhecimento, das novas ideias, e não as palavras precisas usadas para expressá-las.

Então levando em conta a ideia de aprendizagem significativa, a realização do levantamento de resultados foi feito através da aplicação de um pré-teste contendo 2 exercícios de programação, seguido de uma intervenção pedagógica onde foi apresentada a ferramenta HelpBlock, desenvolvida nesta pesquisa. Após isso então foi aplicado um pós-teste, que assim como o pré-teste consistiu em 2 exercícios de programação. O pré-teste teve como objetivo medir o conhecimento prévio que os alunos tinham sobre o assunto, e o pós-teste serviu para medir se a ferramenta HelpBlock facilitou a resolução dos exercícios e a compreensão da forma como o código funciona. Por fim foi solicitado que os alunos respondessem um questionário de satisfação.

A turma na qual foi realizado a intervenção foi a de algoritmos e programação, e contava com 23 alunos. No dia da intervenção os alunos estavam na terceira aula sobre matrizes e assim já tinham conhecimento sobre laços de repetição, laços condicionais, entradas e saídas. A atividade teve duração de duas horas e meia.

3.2 Organização da Pesquisa

A organização deste trabalho foi dividida em sete partes, sendo elas pesquisa, proposta, desenvolvimento, aplicação do pré-teste, intervenção pedagógica, aplicação do pós-teste e participação no questionário de satisfação. A pesquisa foi realizada através de consultas em artigos, livros, sites e apresentada em tópicos no decorrer do trabalho. As próximas partes são detalhadas nos tópicos seguintes

3.3 Desenvolvimento do HelpBlock

Nesta seção é apresentado o desenvolvimento da ferramenta web HelpBlock.

O HelpBlock foi desenvolvido com o intuito de ser um ambiente intuitivo e de fácil compreensão por parte dos usuários. Contendo um sistema de cadastros de exercícios classificados por tipo e dificuldade e disponibilizando um local de programação visual todo feito em blocos conectáveis, isso através da utilização da biblioteca blockly.

3.3.1 – Lista de requisitos

Segundo Mendes (2017), requisito funcional é um requisito que especifica uma função que o sistema ou componente deve ser capaz de realizar. São requisitos que pegam as funcionalidades de acordo com o ponto de vista do usuário.

Ainda segundo Mendes (2017), requisito não funcional é aquele que descreve como e não quais funções o sistema irá realizar.

No quadro 2 são exibidas as prioridades dos requisitos, que definem qual a importância de cada um dentro do projeto.

Quadro 2 - Prioridade dos Requisitos

Prioridade	**Descrição**
Obrigatório/Essencial	Deve ser feito para que o sistema atenda as principais funções do projeto.
Importante	Não implicam no funcionamento do sistema, mas são interessantes de se desenvolver caso haja tempo.
Desejável	Não são necessários, mas são melhorias possíveis.

Fonte: Do autor, 2017

Já no quadro 3 são apresentados a lista de requisitos funcionais mais importantes que foram levados em consideração na hora do desenvolvimento da ferramenta HelpBlock.

Quadro 3 - Lista de Requisitos Funcionais

RF001 – Gerenciar Usuários **Descrição:** O sistema deve permitir o cadastro, alteração e exclusão de usuários. **Prioridade:** Essencial **Pré-condições**: Estar logado no sistema e ter permissões suficientes para realizar as mudanças.
RF002 – Gerenciar Categorias de Usuários **Descrição:** O sistema deve permitir o gerenciamento de categorias relacionadas aos usuários (Professor, Administração e Aluno). **Prioridade:** Essencial **Pré-condições:** É necessário estar logado no sistema com uma conta de administrador.
RF003 – Gerenciar permissões por categoria **Descrição:** O sistema deve permitir que os administradores consigam gerenciar quais tipos de permissões de acesso cada categoria possui. **Prioridade:** Essencial **Pré-condições:** É necessário estar logado no sistema com uma conta de administrador.
RF004 – Gerenciar exercícios **Descrição:** O sistema deve permitir adicionar, editar e excluir exercícios. Assim como possibilitar o compartilhamento dos exercícios entre professores. **Prioridade:** Essencial **Pré-condições:** Estar logado no sistema e ter permissões suficientes para realizar as mudanças.
RF005 – Gerenciar envio e recebimento de soluções

Descrição: O sistema deve permitir que as soluções encontradas para os exercícios possam ser enviadas pelos alunos aos professores. E também que os professores consigam manter um controle sobre esses envios. **Prioridade:** Essencial **Pré-condições:** Estar logado no sistema e ter permissões suficientes para realizar as mudanças.

Fonte: Do Autor, 2017

O quadro 4 traz a lista de requisitos não funcionais que foram levados em consideração no momento do desenvolvimento da ferramenta em questão. Estes requisitos são aqueles mais técnicos, que tem maior importância para os desenvolvedores.

Quadro 4 - Lista de Requisitos Não Funcionais

RNF001 – Utilizar banco de dados PostgreSQL **Descrição:** O sistema deverá utilizar o banco de dados PostgreSQL para armazenar os dados cadastrados. **Prioridade:** Essencial
RNF002 – Utilizar a linguagem JavaScript **Descrição:** O sistema deverá utilizar a linguagem JavaScript para codificação. **Prioridade:** Essencial
RNF003 – Prever velocidade de resposta de 3 segundos. **Descrição:** O sistema deverá prever velocidade de resposta de no máximo 3 segundos em suas operações **Prioridade:** Desejável
RNF004 – Controlar concorrência

Descrição: O sistema deverá controlar a concorrência, não permitindo que dois usuários tentem alterar o mesmo registro ao mesmo tempo, emitindo assim uma mensagem de alerta. **Prioridade:** Essencial

Fonte: Do Autor. 2017

3.3.2 Tecnologias Utilizadas

Para o desenvolvimento do HelpBlock foram utilizadas as seguintes ferramentas: HTML5, CSS3, JavaScript, Servlets e JSP. O HTML5 foi escolhido por permitir uma navegação rápida e simples, melhorando a performance e usabilidade da ferramenta, o CSS3 por oferecer controle total da apresentação e manutenção da ferramenta a partir de um arquivo, e JavaScript por funcionar muito bem com HTML. Quanto ao Servlet e as páginas JSP, foram escolhidos por permitir um melhor controle na hora de trabalhar com as informações dos usuários, e também por permitir a utilização da linguagem JAVA junto com HTML. Todas estas tecnologias são descritas a seguir.

3.3.2.1 JavaScript

Segundo Fernandes et. al (2017), as páginas web eram estáticas antigamente, próximo do início dos anos 90. Os usuários não tinham nenhum tipo de interação com as mesmas, podendo apenas ver o que elas haviam sido projetadas para mostrar.

Ainda segundo Fernandes et. al (2017), naquela época foi lançada uma nova versão do Netscape Navigator que era um popular web browser desenvolvido pela Netscape Communications Corp. Essa nova versão do Netscape Navigator continha integrada à linguagem Java, desenvolvida pela Sun Microsystems. Isso revolucionou a web, pois a partir disso os programadores já conseguiam escrever pequenos aplicativos denominados *applets*, que eram executados de acordo com o

comportamento do usuário. Porém a linguagem Java era um tanto complexa, o que acabava permitindo que apenas desenvolvedores experientes conseguissem usar. Isso fez com que a Netscape buscasse uma linguagem mais simples e fácil de aprender. Nasceu então a linguagem Livescript, parecida com C, C++ e de Java, Livescript, criada por Brendan Eich. Com o passar do tempo a Netscape Communications fechou parceria com a Sun e assim a linguagem Livescript mudou de nome para JavaScript.

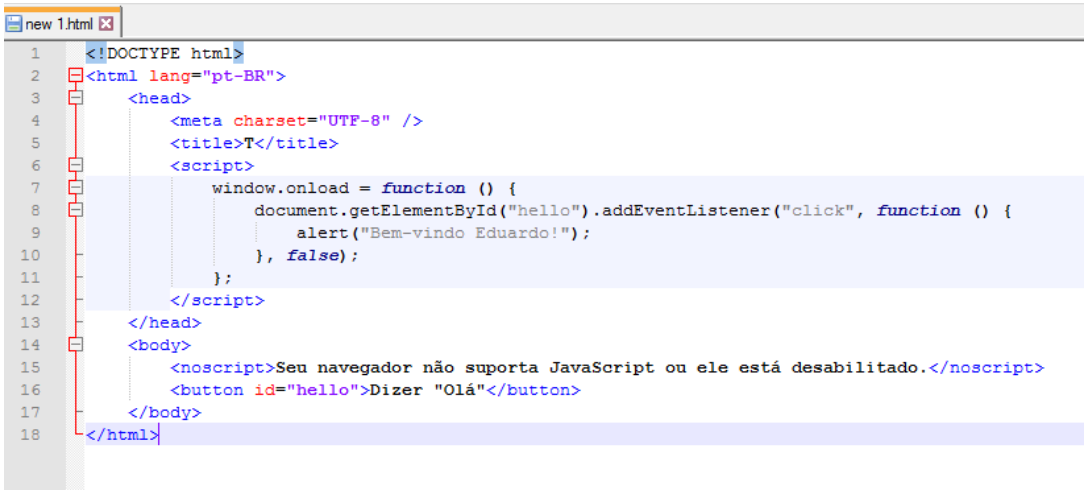
Algo importante a se ressaltar é que JavaScript não é o mesmo que Java, as pessoas costumam confundir achando que não existe diferença entre ambas as linguagens, o que não é verdade.

De acordo com Schimid (2017), uma das definições mais importantes da linguagem Javascript, é o fato de ela ser cliente *side*, que significa que ela é executada no lado do cliente, ou seja, não no servidor e sim no computador do usuário. JavaScript consegue interagir com vários elementos de uma página HTML, tanto quanto trabalhar com variáveis, gerar resultados ou até alterar aparência de elementos.

Schimid (2017) ainda afirma que é uma linguagem interpretada pelo *browser*, ou navegador, que o cliente está usando. Isso pode gerar alguns problemas pois existem browsers que não interpretam JavaScript.

Na figura 12 é apresentado um código escrito em JavaScript dentro de uma estrutura HTML. Esse código faz com que, ao clicar no botão escrito “Dizer Olá”, traz um alerta com a mensagem “Bem-vindo Eduardo!”.

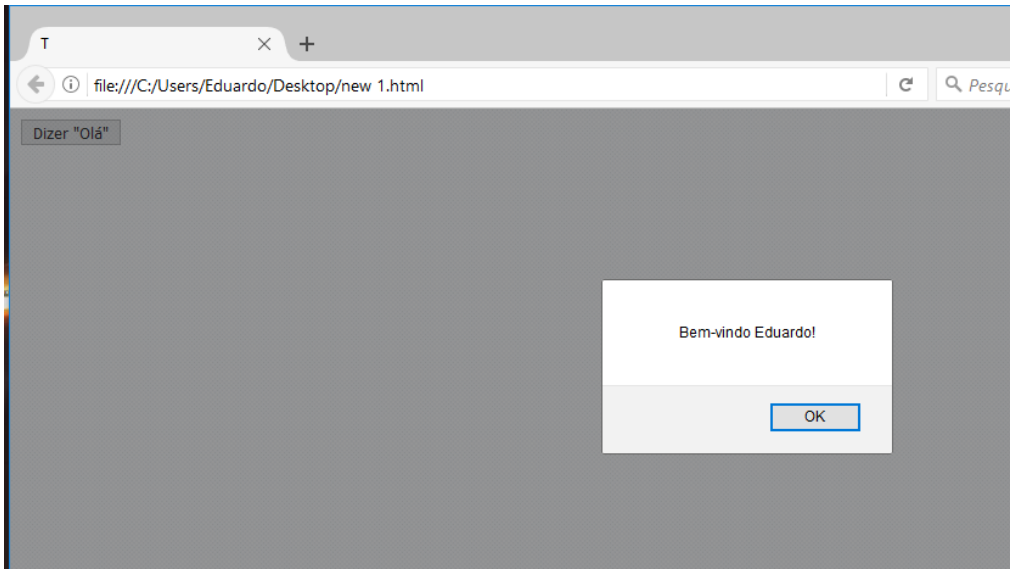
Figura 16 – Exemplo de código JavaScript dentro de uma estrutura HTML



Fonte: Do autor, 2017

Na figura 13 é possível ver o resultado da execução do código JavaScript mencionado anteriormente.

Figura 17 – Resultado obtido com o código do exemplo da figura 12



Fonte: Do Autor, 2017

3.3.2.2 HTML 5

HTML é abreviação de *Hypertext Markup Language* – Linguagem de Marcação de Hypertexto. O que significa que é uma linguagem que serve para publicar conteúdo na web, como por exemplo imagens, vídeos, texto e etc.

Ferreira et. al (2017) afirma que Hipertexto são conjuntos de elementos ou nós ligados por conexão. O que significa que juntos formam uma rede de informação, mas que é diferente de um livro, por exemplo, que tem suas informações ligadas sequencialmente.

Segundo Alvarez (2004) o HTML se criou com objetivos de divulgação, porém não contava com o grande avanço da web e como se tornaria importante e tão utilizado. Assim acabou sendo criado sem um planejamento adequado, que levantasse todas as possibilidades de uso possíveis. Logo foram necessária mudanças e aprimoramentos, que passaram por algumas versões até chegar no HTML5.

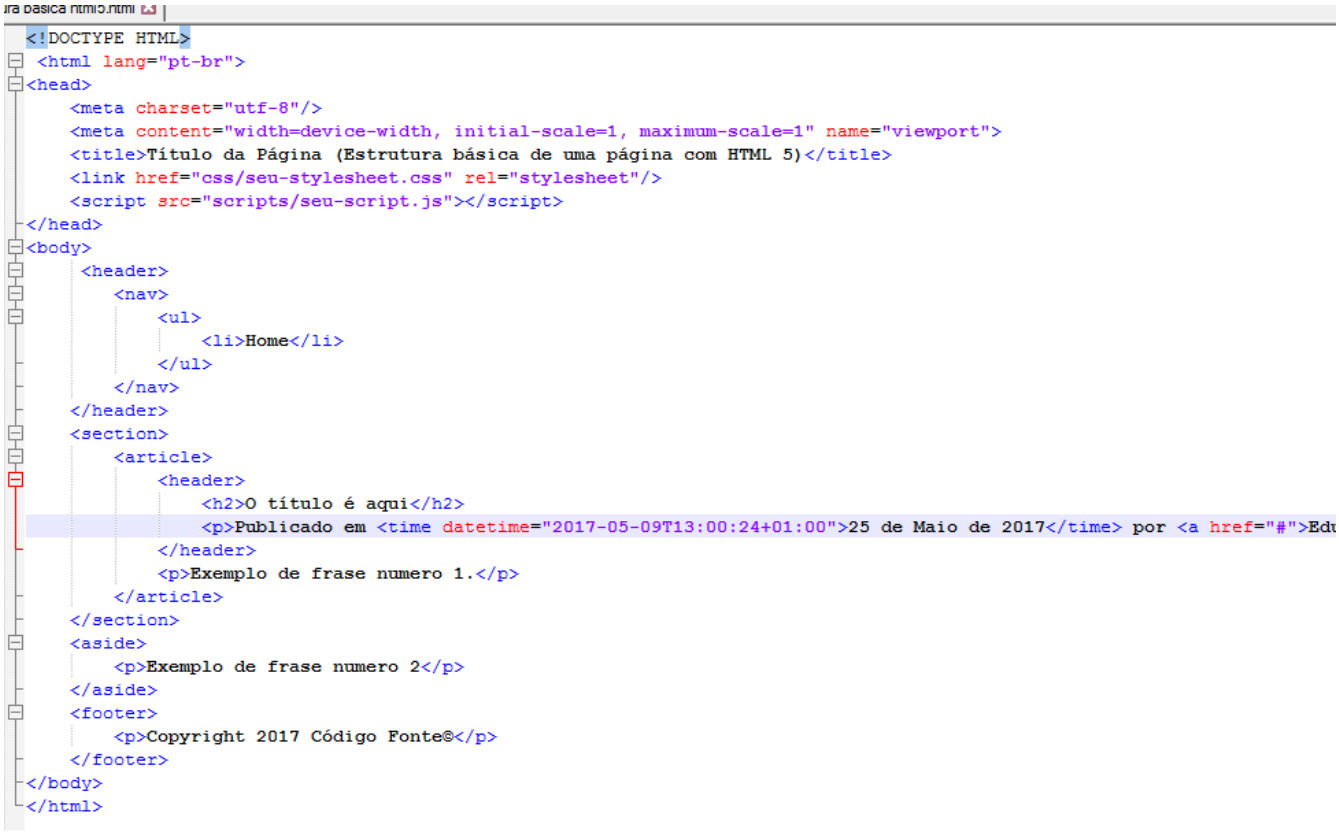
Politi (2012) cita que o HTML5 ampliou o papel da linguagem, sendo mais preparada perante as necessidades dos profissionais que trabalham com criação de sites, aplicativos entre outras coisas.

Segundo Junior (2013), o HTML5 tem como objetivo principal facilitar o trabalho com todos os elementos, melhorando assim a parte de desenvolvimento e deixando ela mais simples e otimizada.

Outra vantagem é que esta versão oferece bastante suporte a utilização de CSS e JavaScript. E ainda não necessita de instalação de plug-ins externos para realizar tarefas como rodar vídeos, ou realizar algum tipo de renderização 3D.

Na figura 14 é possível ver uma estrutura básica de uma página com HTML5.

Figura 18 – Estrutura básica de um HTML5



```
<!DOCTYPE HTML>
<html lang="pt-br">
<head>
    <meta charset="utf-8"/>
    <meta content="width=device-width, initial-scale=1, maximum-scale=1" name="viewport">
    <title>Título da Página (Estrutura básica de uma página com HTML 5)</title>
    <link href="css/seu-stylesheet.css" rel="stylesheet"/>
    <script src="scripts/seu-script.js"></script>
</head>
<body>
    <header>
        <nav>
            <ul>
                <li>Home</li>
            </ul>
        </nav>
    </header>
    <section>
        <article>
            <header>
                <h2>O título é aqui</h2>
                <p>Publicado em <time datetime="2017-05-09T13:00:24+01:00">25 de Maio de 2017</time> por <a href="#">Edu
            </header>
            <p>Exemplo de frase numero 1.</p>
        </article>
    </section>
    <aside>
        <p>Exemplo de frase numero 2</p>
    </aside>
    <footer>
        <p>Copyright 2017 Código Fonte©</p>
    </footer>
</body>
</html>
```

Fonte: Do autor, 2017

3.3.2.3 CSS3

O CSS é responsável por formatar a informação que é entregue pelo HTML, e preparar para que ela seja consumida da melhor maneira possível.

Com as atualizações do CSS3, os pontos que podem ser controlados aumentos sendo alguns deles, bordas arredondadas, sombras em texto e elementos, manipulação de opacidade, controle de rotação, controle de perspectiva, animação entre outros. (Curso W3C, 2017).

3.3.2.4 Servlets

Além dessas também foi utilizado a tecnologia de Servlets, que são classes Java, desenvolvidas de acordo com uma estrutura bem definida que quando instaladas e configuradas em um Servidor que implemente um Servlet Container, podem tratar requisições recebidas de clientes Web, como por exemplo os Browsers (LEONARD).

Ao receber uma requisição, um Servlet pode capturar os parâmetros desta requisição, efetuar qualquer processamento inerente a uma classe Java, e devolver uma página HTML. Resumindo, o objetivo é receber chamadas HTTP, processá-las e devolver uma resposta ao cliente (LEONARD).

3.3.2.5 JSP

Também foi utilizado JSP (Java Server Pages), que consiste em uma linguagem de script gratuita, criada pela SUN. É possível escrever HTML com códigos JSP embutidos, onde o HTML é estático e o JSP fica como responsável pelo dinamismo. Nesse tipo de página a codificação é escrita em Java, inserida entre as tag's <% e %>. Como o JSP necessita de um servidor para funcionar, foi utilizado o servidor Tomcat. O Apache Tomcat foi desenvolvido pela Apache Software Foundation, e nada mais é que um servlet container de código aberto, ou seja, uma aplicação que interpreta e processa servlets e JSP (GOMES, 2012).

3.3.3 Descrição do HelpBlock

Inicialmente é necessário criar um cadastro para começar a utilizar o HelpBlock, sendo que diferentes comandos são liberados de acordo com a categoria a qual cada usuário pertence, podendo ser professor ou aluno.

Na figura 19 é possível ver a tela inicial da ferramenta, contendo as opções de Entrar ou Cadastro, após informar o email e a senha cadastrados no sistema é possível seguir para a tela seguinte.

Figura 19 - Tela inicial do HelpBlock

Fonte: Do autor, 2017.

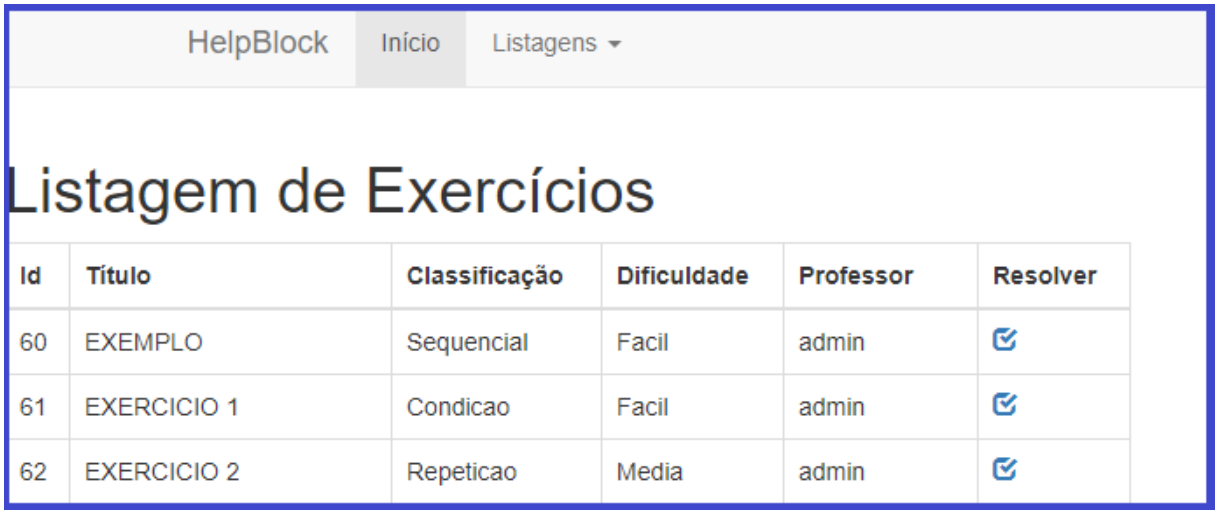
Já na figura 20 é apresentado o menu referente ao login realizado anteriormente, que no caso é o de um aluno. Assim são disponibilizadas apenas as opções de listagem de exercícios cadastrados por professores e das respostas que o aluno já entregou.

Figura 20 - Tela de menu do aluno

Fonte: Do autor, 2017.

Assim que o aluno clica na opção Exercícios na listagem ele é encaminhado para a página que contém os exercícios cadastrados, nesta página como podemos ver na figura 21, existe um botão chamado Resolver no final da linha de cada exercício, sendo assim o aluno deve escolher o exercício que deseja resolver e clicar no botão para ser encaminhado para a página seguinte.

Figura 21 - Listagem de exercícios



Fonte: Do autor, 2017.

Após escolher o exercício e clicar em resolver, o aluno é encaminhado para a página de resolução de exercícios, que é mostrada nas figuras 22 e 23. Na figura 22 é apresentada a parte superior da página, que contém a descrição do exercício, os blocos de programação, divididos nos seguintes grupos:

- **Logic** – Contém os blocos lógicos de condição como por exemplo o if else, número maior ou menor, verdadeiro ou falso.
- **Loops** – Contém os blocos de repetição como por exemplo o while e o for.
- **Math** – Contém os blocos de matemática, que são blocos que trazem o cálculo da raiz quadrada por exemplo ou até blocos que verificam que determinado número é par ou ímpar, restos de divisão entre outros.
- **Text** – Contém os blocos de texto como por exemplo o imprime, converte para maiúscula, bloco criador de texto, e também um bloco de entrada que solicita ao usuário alguma informação.
- **Variables** – Neste grupo é possível criar as variáveis, assim como utilizar blocos específicos para definir valores a cada variável criada.

E ao lado direito a área de desenvolvimento, que é onde os alunos terão de arrastar os blocos para criar os algoritmos.

Figura 22 - Tela de resolução de exercícios, parte de cima

Fonte: Do autor, 2017.

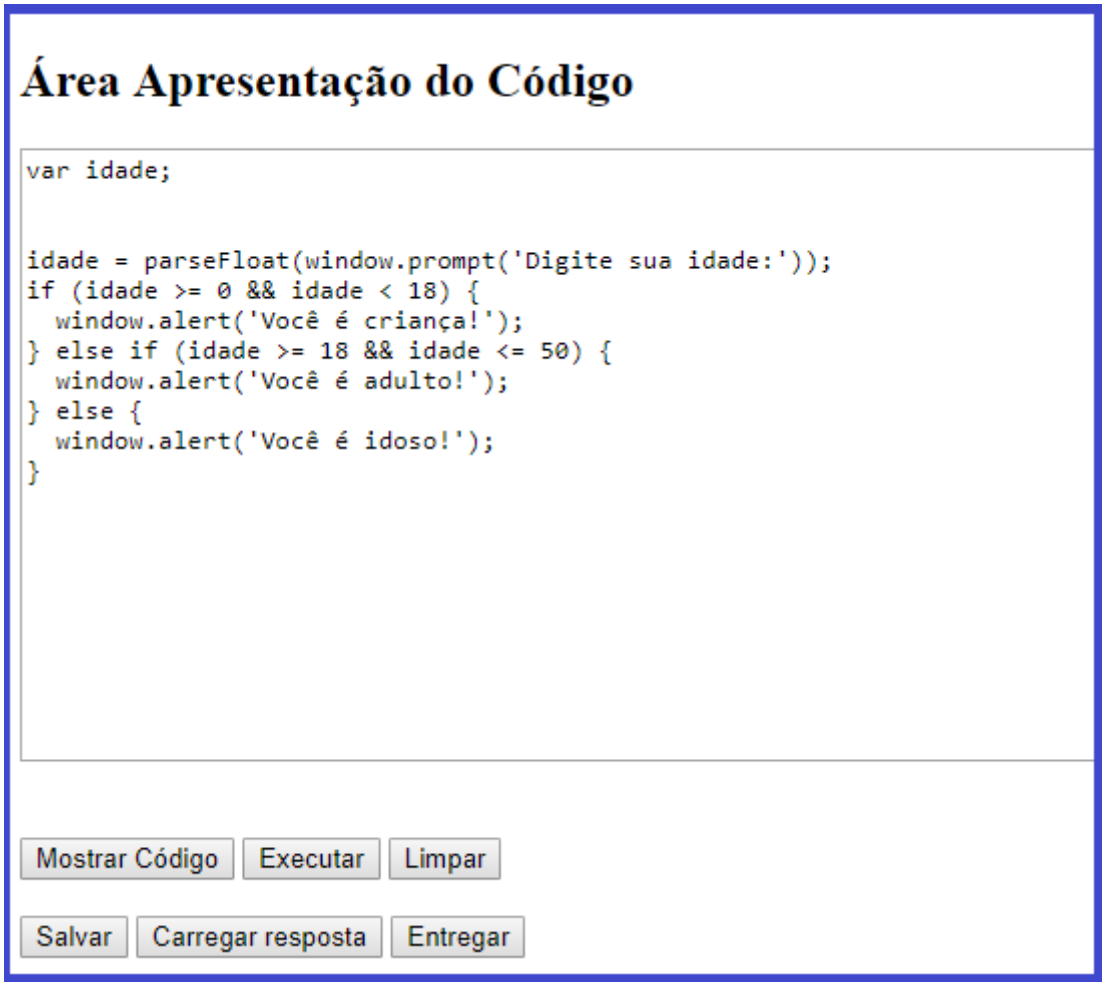
Já a figura 23 traz a parte inferior da tela de resolução, onde existe um local de apresentação de código já com um algoritmo em formato de código, assim como os botões mostrar código, executar, limpar, salvar, carregar resposta e entregar, que realizam as seguintes funções:

- **Mostrar código** – Converte em javascript os blocos que estão na área de desenvolvimento e apresenta na área de apresentação de código.
- **Limpar** – Limpa a área de apresentação do código.
- **Executar** – Pega os blocos que estão na área de desenvolvimento e executa o algoritmo formado.
- **Salvar** – Pega todos os blocos que estão na área de desenvolvimento, converte em texto e salva no banco, porém é importante salientar que cada aluno pode salvar apenas uma resposta por exercício, caso exista uma resposta já salva de determinado usuário para determinado exercício a próxima vez que o

usuário clicar em salvar essas respostas irão se sobrepor mantendo apenas o último registro salvo.

- **Carregar resposta** – Busca o que está salvo no banco e monta na área de desenvolvimento.
- **Entregar** – Envia para o professor os blocos que estão na área de desenvolvimento, juntamente com um comentário falando sobre como foi a resolução do exercício e se teve alguma dificuldade durante. Cada aluno pode entregar apenas uma resposta por exercício.

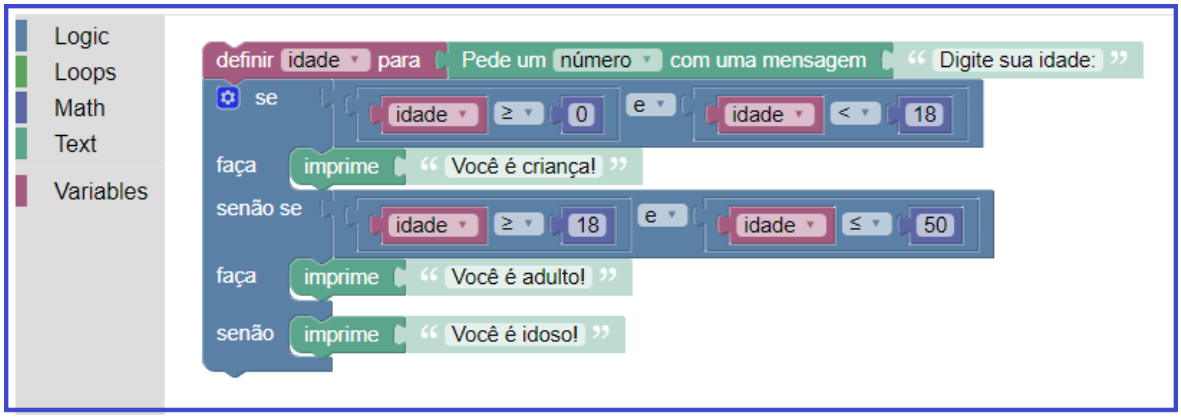
Figura 23 - Tela de resolução de exercícios, parte de baixo



Fonte: Do autor, 2017.

Na figura 24 é possível ver o mesmo algoritmo que é apresentado na figura 23, porém em formato de blocos.

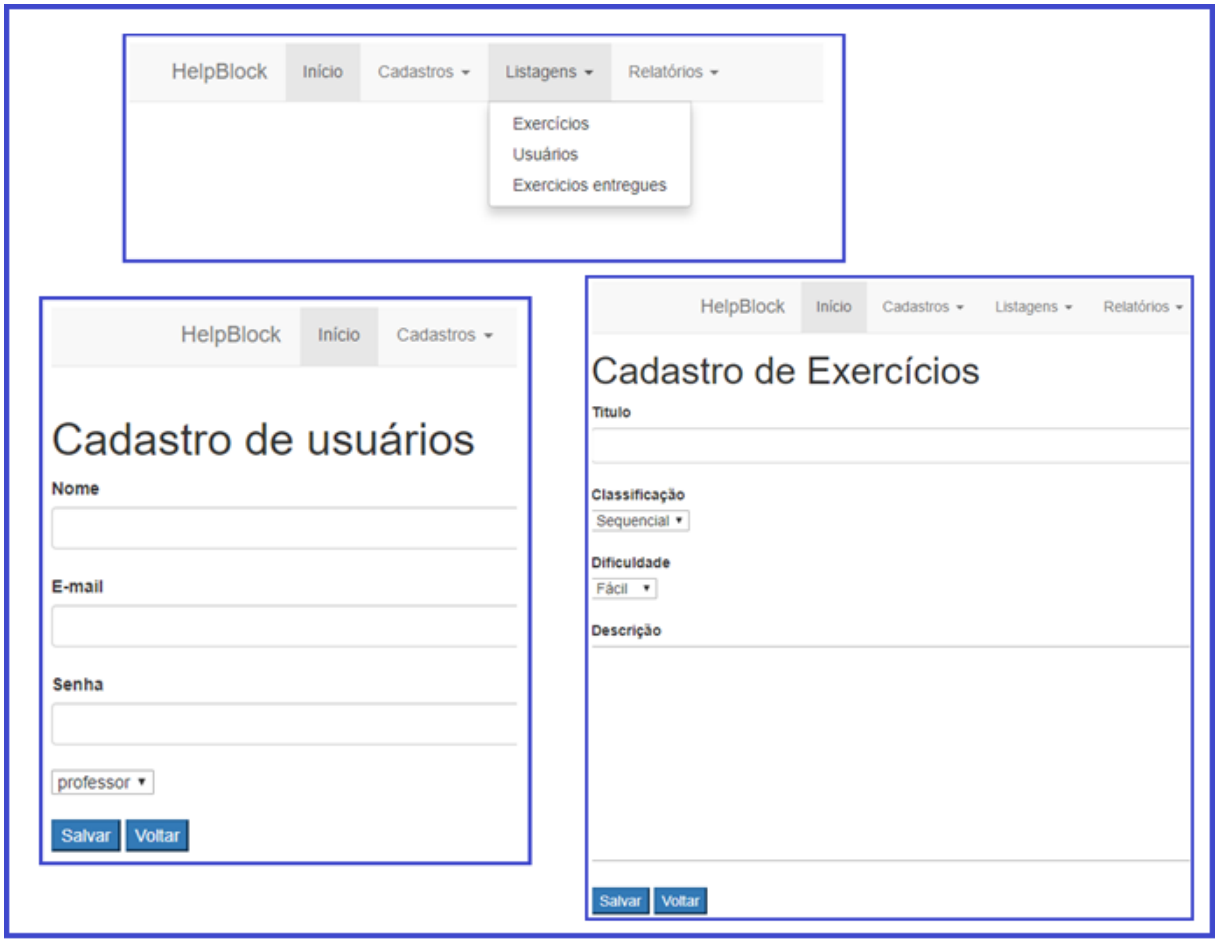
Figura 24 - Código apresentado na figura 23 em formato de blocos



Fonte: Do autor, 2017.

Caso o usuário seja um professor, ele tem acesso a alguns recursos extras, na figura 25 apresenta-se alguns destes recursos, no qual existem as opções de cadastrar exercícios, alunos e professores, uma listagem de exercícios, semelhante a que é apresentada na figura 21, porém com a possibilidade de editar exercícios já existentes, uma listagem de usuários cadastrados e também uma listagem dos exercícios entregues por alunos.

Figura 25 - Recursos disponíveis para professores



Fonte: Do autor, 2017.

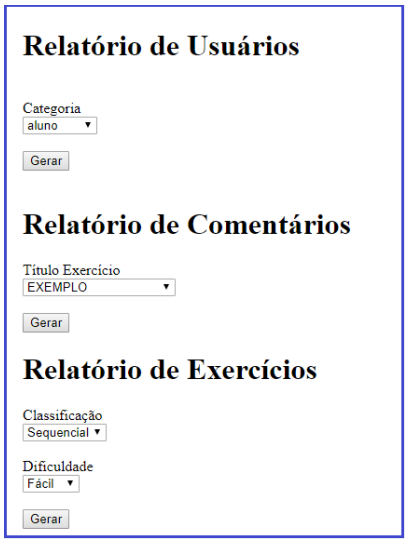
Na figura 26 é apresentada a listagem de entregas contendo o id da entrega, o id do exercício, título, professor que cadastrou o exercício, aluno que realizou a entrega, comentário deixado pelo aluno e o feedback deixado pelo professor após a correção. É possível clicar no botão “mostrar blocos” para ver a resposta em blocos.

Figura 26 - Listagem de entregas

Fonte: Do autor, 2017.

Também é possível gerar relatórios de usuários passando como parâmetro a categoria, comentários deixados pelos alunos passando como parâmetro o título do exercício do qual se deseja verificar os comentários e também um relatório de exercícios cadastrados na ferramenta passando como parâmetros a classificação e a dificuldade do exercício. Na figura 27 é possível ver a tela de geração de relatórios com os combos contendo os parâmetros para gerar os relatórios.

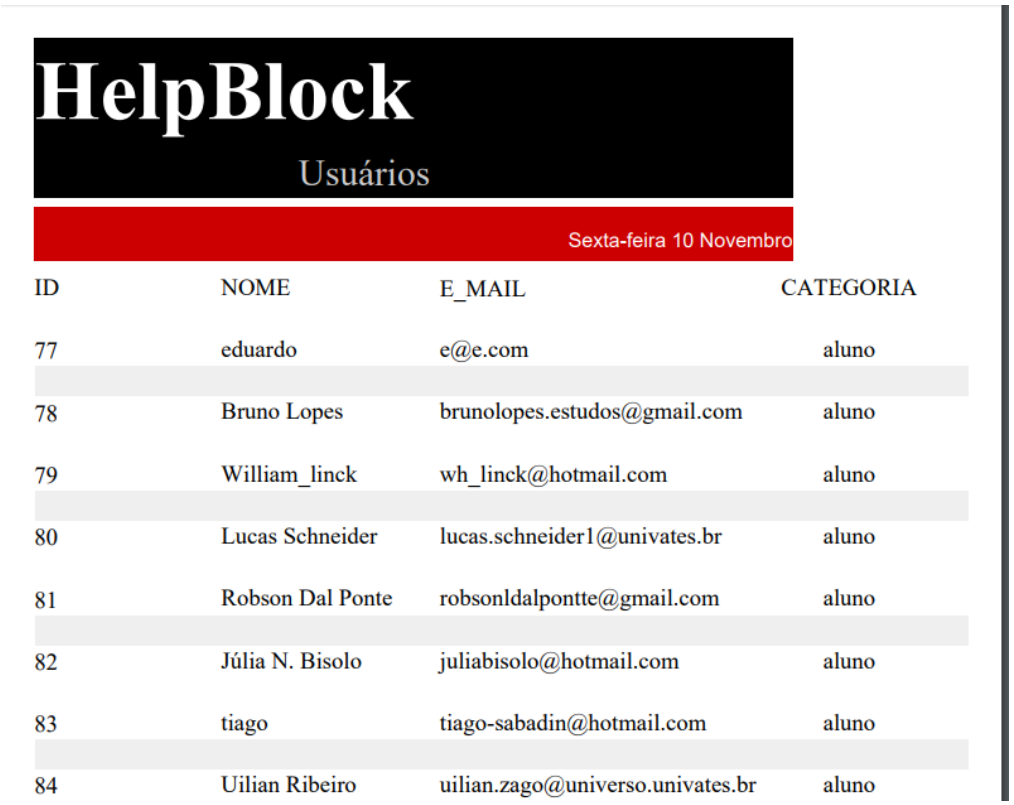
Figura 27 - Tela de geração de relatórios com parâmetros



Fonte: Do autor, 2017

Na figura 28 é mostrado um exemplo de um relatório de usuários, onde foi passado a categoria de aluno como parâmetro, trazendo assim apenas a lista dos alunos cadastrados no sistema.

Figura 28 - Relatório de usuários da categoria aluno



HelpBlock

Usuários

Sexta-feira 10 Novembro

ID	NOME	E_MAIL	CATEGORIA
77	eduardo	e@e.com	aluno
78	Bruno Lopes	brunolopes.estudos@gmail.com	aluno
79	William_linck	wh_linck@hotmail.com	aluno
80	Lucas Schneider	lucas.schneider1@univates.br	aluno
81	Robson Dal Ponte	robsonldalpontte@gmail.com	aluno
82	Júlia N. Bisolo	juliabisolo@hotmail.com	aluno
83	tiago	tiago-sabadin@hotmail.com	aluno
84	Uilian Ribeiro	uilian.zago@universo.univates.br	aluno

Fonte: Do autor, 2017

3.3.4 Manual de utilização do HelpBlock

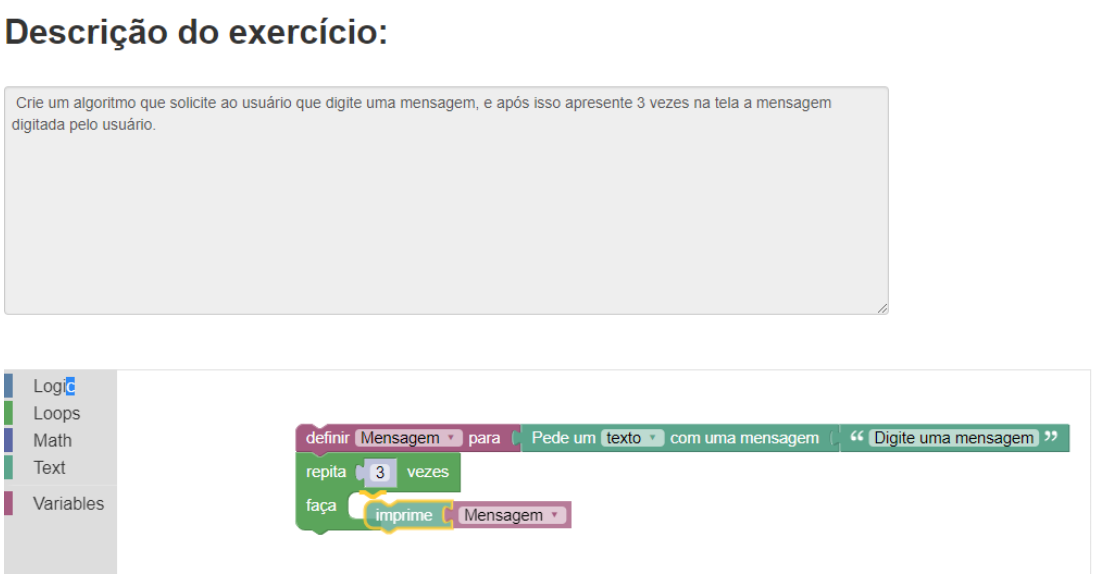
Primeiramente deve-se clickar no grupo de blocos desejado, fazendo com que apareçam os blocos contidos no grupo, assim como pode ser visto na figura 29.

Figura 29 - Passo número 1

Fonte: Do autor, 2017.

Assim que os blocos aparecerem basta escolher o bloco desejado, clickar, segurar e arrastar para a área de desenvolvimento e ir conectando um bloco ao outro. É possível ver quais blocos podem ser conectados ao aproximá-los um do outro, isso fará com que apareça um contorno de outra cor no bloco, assim como pode ser visto na figura 30.

Figura 30 - Passo número 2



Fonte: Do autor, 2017.

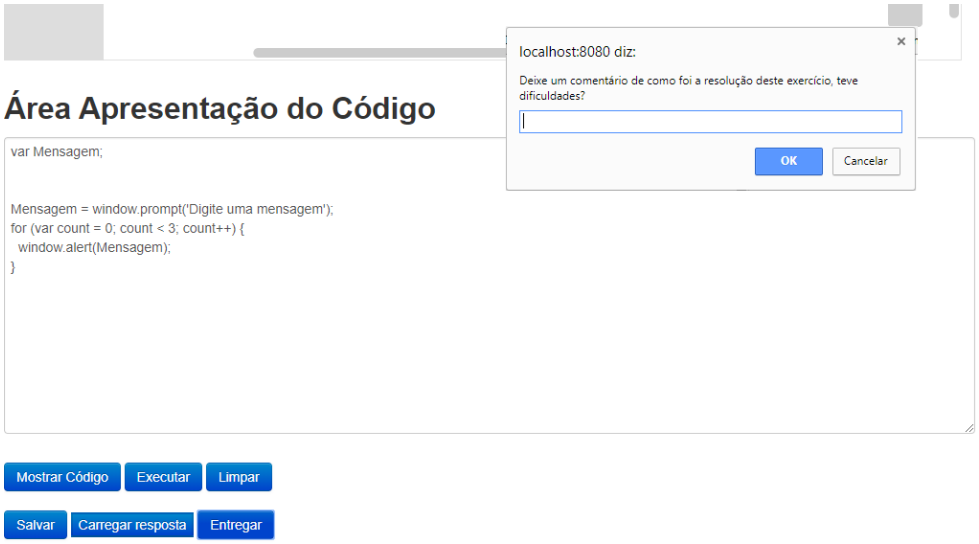
Após montar o algoritmo, basta clicar em mostrar código para que o algoritmo seja apresentado em JavaScript na área de apresentação do código, ou em executar para que o algoritmo seja executado. Caso deseje salvar os blocos que estão na área de desenvolvimento basta clicar em salvar, e quando quiser carrega-los novamente só é necessário clicar em carregar resposta, assim como é apresentado na figura 31.

Figura 31 - Passo número 3

Fonte: Do autor, 2017.

Assim que o exercício for finalizado, deve-se clicar em Entregar para enviar a resposta para a lista de respostas dos professores, nesse momento é necessário escrever um comentário que irá juntamente com a resposta, assim como pode ser visto na figura 32.

Figura 32 - Passo número 4



Fonte: Do autor, 2017.

Após realizar o desenvolvimento da ferramenta, foram realizados os testes de validação da mesma, estes testes assim como os resultados obtidos serão melhor descritos no capítulo seguinte.

4 – Análise dos Resultados

Os testes da ferramenta foram realizados em três momentos, sendo eles o pré-teste, a intervenção pedagógica e o pós-teste. Primeiramente foi apresentada a proposta da pesquisa e feito alguns comentários sobre os motivos de ter sido escolhida a programação visual para tentar melhorar o ensino nas cadeiras iniciais dos cursos de informática.

4.1 – Pré-teste

Em seguida foi solicitado que os alunos resolvessem 2 exercícios (Apêndice A) utilizando a ferramenta com a qual estavam trabalhando já durante as aulas, que no caso era o Dr. Java. No Quadro 5 apresenta-se o exercício 1.

Quadro 5 - Exercício 1 do pré-teste

Crie um algoritmo que recebe a nota de um aluno e informe se este aluno está aprovado, reprovado ou em recuperação. Considerando que para ser aprovado o aluno tenha de tirar 8 ou mais, para realizar a recuperação o aluno tenha que ter nota entre 6 e 7, e se o aluno tiver nota 5 ou menor está reprovado.

Fonte: Do autor, 2017.

O exercício número 1 a grande maioria dos alunos conseguiu realizar sem nenhum tipo de problema, apenas um ou dois alunos tiveram algum erro de sintaxe. No quadro 6 é apresentado o enunciado do exercício 2 do pré-teste.

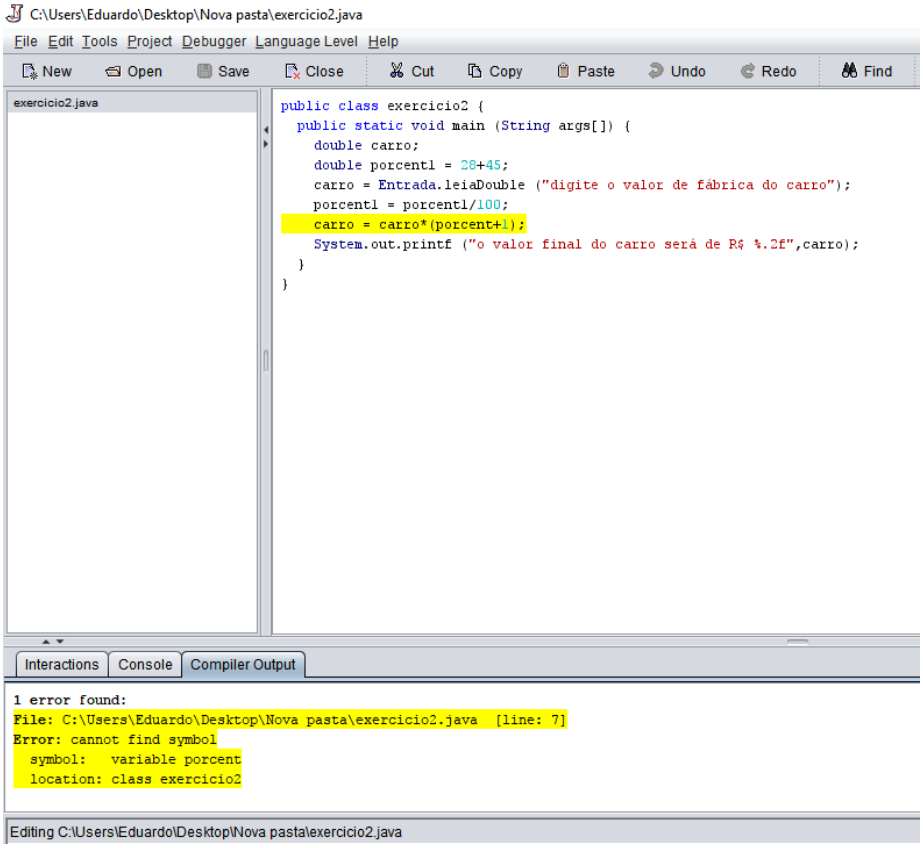
Quadro 6 - Exercício 2 do pré-teste

O custo ao consumidor de um carro novo é a soma do custo de fábrica com a percentagem do distribuidor e dos impostos (aplicados ao custo de fábrica). Supondo que a percentagem do distribuidor seja de 28% e os impostos de 45%, escrever um algoritmo que leia o custo de fábrica de um carro e escreva o custo ao consumidor.

Fonte: Do autor, 2017

No exercício número 2 (Quadro 6) houveram algumas confusões na hora de interpretar o enunciado, provavelmente por envolver porcentagem, assim alguns alunos calcularam cada uma das porcentagens separadas sobre o valor inicial e outros calcularam primeiro uma das porcentagens sobre o valor inicial e a segunda porcentagem calcularam sobre o valor gerado pelo primeiro cálculo. Mas a maioria foi bem e conseguiu resolver o problema, contudo ainda apareceram alguns erros clássicos de sintaxe assim como é mostrado na figura 29.

Figura 33 - Erro clássico



Fonte: Do autor, 2017

4.2 – Utilização do HelpBlock

Assim que a primeira etapa foi finalizada, seguiu-se para a segunda parte do experimento que foi a utilização da ferramenta HelpBlock. Nessa parte então foi apresentada a ferramenta HelpBlock aos alunos e feito a resolução de um exercício de exemplo, conforme figura 30, para demonstrar como que funcionavam alguns blocos principais e como que eles deveriam ser encaixados, e também a parte de salvar e carregar suas respostas e por fim como deveriam finalizar e entregar cada um dos dois exercícios propostos.

Figura 34 - Exercício utilizado para trabalhar com a ferramenta HelpBlock

Fonte: Do autor, 2017.

4.3 – Pós-teste

Após terminar as explicações e demonstrações relacionadas à ferramenta seguiu-se para a terceira parte, onde foi aplicado o pós-teste solicitando que resolvessem mais dois exercícios semelhantes aos dois exercícios do pré-teste. O primeiro exercício pode ser visto no quadro 7.

Quadro 7 - Exercício 1 do pós-teste

Crie um algoritmo que recebe a idade de uma pessoa e informe se ela é criança, adulto ou idoso. Considerando que para ser criança ela deve ter menos de 18 anos, para ser adulto deve ter de 18 a 50 anos e para ser idoso deve ter mais de 50 anos.

Fonte: Do autor, 2017

E o quadro 8 mostra o enunciado do exercício número 2 que foi proposto aos alunos no pós-teste.

Quadro 8 - Exercício 2 do pós-teste

Joaquim tem 1,50cm e cresce 2 centímetros por ano, enquanto Maicon tem 1,10m e cresce 3 centímetros por ano. Crie um algoritmo que calcule e imprima quantos anos serão necessários para que Maicon seja maior que Joaquim.

Fonte: Do autor, 2017

Em ambos os exercícios os alunos se saíram bem, alguns problemas que surgiram eram geralmente em relação à falta de experiência com a ferramenta, devido ao pouco tempo que tiveram de contato com a mesma. Durante a resolução dos exercícios utilizando o HelpBlock, alguns alunos pediram ajuda, pois tinham dúvidas de qual bloco deveriam utilizar para realizar determinadas tarefas, mas assim que mostrado e explicado uma vez, o aluno já conseguia dar sequência sem problemas. Outro ponto importante que foi possível perceber é o de que por utilizar blocos, a parte de tirar dúvidas dos alunos ficou muito mais fácil, pois era possível mostrar onde cada parte do algoritmo deveria se encaixar para funcionar e o motivo de ter que ser daquela forma.

Ao terminar cada exercício foi solicitado que eles entregassem utilizando o recurso da ferramenta, pois ao clicar no botão entregar fazia com que fosse aberto uma janela solicitando um comentário que seria enviado junto com a resposta, esta solicitação perguntava ao aluno como foi a experiência de resolver um problema utilizando o HelpBlock e se teve algum tipo de dificuldade, alguns dos comentários podem ser vistos no quadro 9:

Quadro 9 - Comentários dos alunos ao entregar os exercícios

Resolução fácil e prática.
Dificuldades iniciais de criar um else if, com a prática peguei o jeito.

Maiores dificuldades com a parte text, basicamente uma melhor separação visual entre as opções que permitem uma abertura/abrangimento das que finalizam um "conjunto" seria interessante.
O programa ficou muito bom, só tive um pouco de dificuldade para encontrar os comandos que eu queria utilizar.
No primeiro momento sim, mas depois de alguns instantes tornou-se lúdica e prática
Os recursos oferecidos (entradas, processamento e saídas) foram satisfatórios e de boa usabilidade para a resolução do problema.
Apenas no encerramento dos laços, mas descobri que as chaves eram automáticas
Gostei dessa plataforma, é bem intuitiva, fica fácil de aprender pela montagem dos blocos.
Blocos de print um pouco confusos

Fonte: Do autor, 2017.

Levando em consideração os comentários enviados pelos alunos é possível perceber que houve dificuldade apenas no início, pois era algo novo e diferente para os alunos que até o momento haviam utilizado apenas Dr. Java e codificação em modo texto, mas conforme iam descobrindo as funcionalidades da ferramenta, tudo ia ficando mais fácil e rápido.

4.4 – Questionário de satisfação

Após finalizar o pré-teste, intervenção com a utilização do HelpBlock e pós-teste, pediu-se para que os alunos respondessem a um questionário de satisfação (Apêndice C) que tinha como objetivo obter a opinião dos alunos em relação a prática de programação visual e também sobre a experiência que tiveram com o HelpBlock.

No quadro 10 está à pergunta número 1 do questionário, que questiona os alunos se o uso da ferramenta com programação visual ajudou na hora de resolver os exercícios. Ao avaliar as respostas obtidas foi possível concluir que a maioria dos

alunos concorda que utilizar uma ferramenta com programação visual facilita e agiliza o processo de desenvolvimento, porém também notasse alguma dificuldade justamente pela falta de prática com a ferramenta.

Quadro 10 - Pergunta número 1 do questionário de satisfação

01. O uso da ferramenta web com programação visual facilitou a resolução dos exercícios? Por que?
Sim em partes será necessário ter mais pratica
Se uma certa forma agregou valor no conhecimento sem dúvidas a facilidade vem com a pratica de mexer com o programa
Sim.
Sim, pois é muito intuitiva, e de fácil compreensão.
Já tenho o conhecimento básico, mas acredito que com algum aprimoramento visual teria sido uma ótima ferramenta de aprendizado.
Parcialmente, pois sem a noção básica da funcionalidade dos blocos fica um pouco confuso o uso para a resolução dos exercícios.
Sim, porque é possível entender melhor o que está acontecendo.
Inicialmente não, pois ainda não temos prática com a ferramenta
Sim, porque não precisa digitar a codificação completa
Sim, pois não precisa digitar e assim consegue manter o pensamento.
Não, pois eu não estava familiarizado com o programa.
Sim, apesar de não conhecer todas as ferramentas a tempo.
Sim, achei bem legal e mais prático.
No primeiro exercício sim. Nos segundo não tanto, as funções matemáticas ficaram confusas
Mais ou menos. É complicado de localizar algumas coisas básicas como entrada e outras que acabam confundindo. Mas em sua maioria facilitou sim
Facilitou demais, pois simplificou o entendimento.
Sim, facilidade em entender os exercícios.

Sim, pois trata apenas lógica, deixando as formalidades por conta do próprio programa.
S
Sim, facilitou muito a criação dos códigos.

Fonte: Do autor, 2017.

O quadro 11 traz a pergunta de número 2, na qual é questionado se o aluno teve alguma dificuldade em resolver os exercícios na qual a ferramenta conseguiu ajudar a superar. Aparentemente os alunos não apresentavam grandes dificuldades na hora de resolver os exercícios, até por que os exercícios propostos foram simples com o intuito de permitir que todos fossem resolvidos, mas é interessante ressaltar que alguns conseguiram entender melhor os algoritmos.

Quadro 11 – Pergunta número 2 do questionário de satisfação

02. Você tinha alguma dificuldade em resolver os exercícios que a ferram ajudou você a superar?
Não (4)
Não tinha
Nem uma dificuldade a ferramenta facilitou um monte
Sim, muito fácil de entendimento, mais muito prático.
Não, os desafios eram simples.
Não.
Sim
Sim, a memorização dos códigos
Não tinha muita dificuldade nos exercícios que foram apresentados, porém em algoritmos maiores e complexos provavelmente a plataforma seria uma melhor opção para criar os mesmos.
Sim, ajuda a entender melhor

Seria interessante, conforme as informações vão sendo processadas, os blocos terem uma cor diferente ou algo que o difere no momento da execução.
Não tinha dificuldade, mas acabou acelerando a resolução do problema
Não especificamente, pois durante as aulas nos foi dada todas as “ferramentas” necessárias para resolver um exercício em programação, como já sabíamos qual usar o programa simplificou.
Sim.
O exercício 2 ficou confuso nas operações matemáticas
S
Não havia mais nenhuma dificuldade após muito estudo, porém a ferramenta simplifica muito o trabalho.

Fonte: Do autor, 2017.

O quadro 12 contém a pergunta número 3 do questionário, que pede ao aluno se ele acha que a utilização de blocos lógicos ajuda na hora de programar. Nessa questão praticamente todos concordam que ajuda e facilita, principalmente para quem está começando a aprender programação.

Quadro 12 - Pergunta número 3 do questionário de satisfação

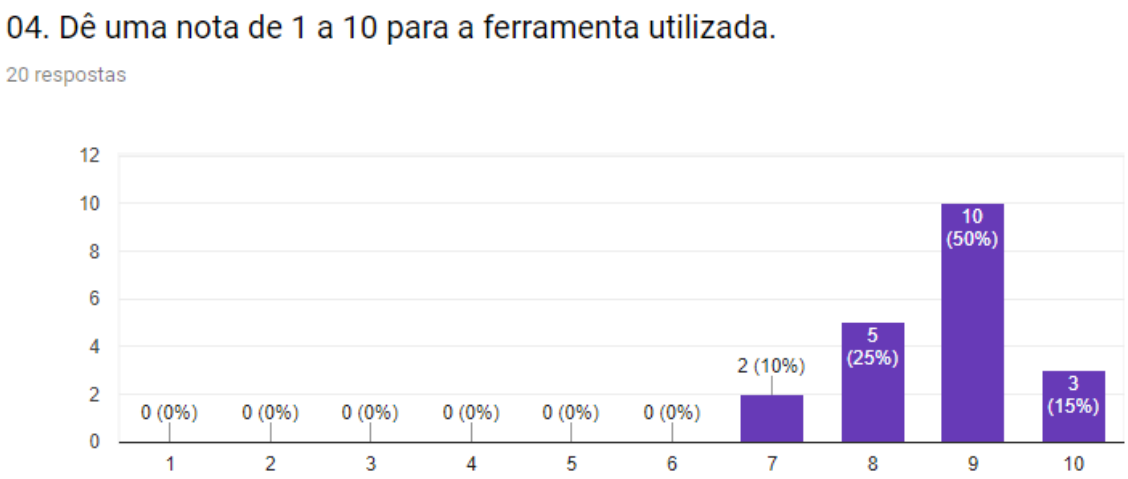
03. Você acha que a utilização de blocos de programação ajuda na hora de programar?
Sim (2)
Sim pois deixa tudo mais pratico
Com certeza blocos prontos facilitam bastante como por exemplo o ladder é bloco e é muito fácil para programar
Muito porque auxilia no melhor entendimento na hora da montagem.
Sim, principalmente, ao iniciar uma cadeira de programação, pois os blocos lhe mostram como os comandos se interligam, a sequência de execução.
Precisaria testar com exercícios mais complexos para ter certezas, como usamos pouco ainda tinha dificuldades de visualização para montar os blocos.

Ajuda parcialmente, mesmo utilizando os blocos para uma resolução mais rápida do problema. Se perde um pouco do conceito de programação, de criar os laços identificando o no próprio código fonte.
Sim, principalmente para quem não conhece muito sobre programação.
Ajuda para pessoas começando a programar
Com certeza
Sim, eles permitem que o pensamento fique mais ágil.
Acredito que na hora de startar do zero sim, mas como já estamos familiarizado com o drjava e utilizamos o programa pela primeira vez, foi mais complicado.
Sim, pois o uso de blocos facilita a visualização como se fosse um diagrama.
Sim, mas é necessário um tempo de adaptação
Dependendo da pergunta...pode tanto ajudar quanto atrapalhar (isso porque é recente para mim e não estou acostumado com a ferramenta). Mas ajuda muito na visualização e resolução dos exercícios.
Sim, pois diminui tempo de programação e diminui o código
Sim, bastante.
Sim e muito, pois você não precisa ficar analisando o código e sua formatação, apenas se a lógica está coerente.
S

Fonte: Do autor, 2017

E por último o gráfico 1 mostra a questão número 4, onde é solicitado ao aluno que deixe uma nota para a ferramenta utilizada, no caso o HelpBlock. Dois alunos deram nota 7, cinco deram nota 8, dez deram nota 9 e três deram nota 10. Com isso foi possível concluir que a ferramenta foi bem aceita pelos alunos e que despertou interesse na maioria.

Gráfico 1 – Pergunta número 4 do questionário de satisfação



Fonte: Do autor, 2017.

6 – CONSIDERAÇÕES FINAIS

A biblioteca blockly é realmente muito interessante de se trabalhar, com ela foi possível disponibilizar recursos bastante satisfatórios aos alunos. A quantidade de blocos com variadas funções é muito útil na hora de criar um algoritmo, também é importante ressaltar a facilidade de encontrar ajuda no site da API, existe muito suporte aos desenvolvedores que resolvem adicionar a API a seus projetos. Outro fator que enriqueceu a ferramenta desenvolvida foi a possibilidade de salvar os algoritmos criados com blocos e carrega-los em outro momento, assim como enviar a resposta para o professor.

Além disso, a possibilidade de converter os blocos em código é muito útil, pois permite que o aluno compare sua solução em blocos com a solução em linhas de código. Porém, seria interessante se fosse possível converter para a linguagem JAVA também, além das que já são possíveis.

Utilizar ferramentas inovadoras que possibilitam oferecer novas maneiras de ensino aos alunos é algo muito interessante, a programação visual ajuda a despertar o interesse dos alunos na hora de aprender, e facilita o entendimento dos algoritmos.

Durante a realização dos testes foi possível perceber que os blocos despertaram interesse e curiosidade nos alunos, ao analisar os resultados percebeu-se que alguns alunos além de resolver os 2 exercícios propostos, ainda resolveram o exercício de exemplo que estava cadastrado na ferramenta, mesmo sem ser solicitado, isso mostra que houve interesse em explorar a nova ferramenta de programação.

Os comentários deixados pelos alunos e as respostas do questionário de satisfação deixaram claro que programar através de blocos conectáveis, ajuda na hora de compreender os algoritmos e agiliza o processo de programação. Houve um pouco de dificuldade no início por não estarem familiarizados com a ferramenta, pois só tiveram a chance de trabalhar com ela durante uma noite, mas se a mesma fosse utilizada durante duas ou três aulas tudo se tornaria mais fácil. Dessa forma, acredita-se que é uma maneira interessante de se ensinar no início do conteúdo de algoritmos, pois o aluno tem de se preocupar apenas com a lógica que está sendo aplicando em cada algoritmo e não precisa perder tempo com a sintaxe. Mesmo utilizando blocos para programar, a ferramenta ainda possibilita que os alunos gerem o código formado por seus blocos, então ainda existe um contato com as linguagens de programação, o que é importante para um segundo momento, onde se introduziria a programação com as linguagens de alto nível, como Java, Javascript, PHP entre outras.

Ao concluir este trabalho confirma-se a hipótese de que oferecer novas maneiras de ensino aos alunos é muito bom e ajuda a despertar o interesse e a curiosidade nos alunos, fazendo com que eles mesmos busquem o conhecimento. Para que haja aprendizagem não basta apenas que os professores se empenhem em trazer conteúdo, mas sim, que exista um comprometimento dos alunos em aprender, e isso só ocorre quando há interesse do aluno no conteúdo proposto. A ferramenta desenvolvida nesta pesquisa ainda pode ser continuada futuramente, sendo integrada em um sistema maior, disponibilizando esta maneira alternativa de programar com blocos aos alunos.

REFERÊNCIAS

ALENCAR, Gersica A.; FREITAS Ana K; PESSOA, Maércio dos S.; MARTINS Danielle J. S. **Utilizando o SCRATCH nas aulas de Lógica de Programação do Proeja: Um relato de experiência.** TISE 2014. Disponível em: <http://www.tise.cl/volumen10/TISE2014/tise2014_submission_110.pdf> Acesso em: 25 mai. 2017

ALMEIDA, Eliana S.; COSTA, Evandro de B.; BRAGA, Juliana D.; SILVA, Klebson dos S., PAES, R. B. e ALMEIDA, Antanasio M. **AMBAP: Um Ambiente de Apoio ao Aprendizado de Programação. In X Workshop sobre Educação em Computação**, Florianópolis. Anais do WEI 2002/ SBC2002.

ALVAREZ, Miguel Angel. **O que é HTML**. 14 abr. 2004 Disponível em: <http://www.criarweb.com/artigos/7.php> Acesso em: 25 mai 2017.

Apostila de Portugol Disponível em: <http://www.netsoft.inf.br/aulas/1_SIN_Algoritmos_Programacao/Apostila_2_Portugol.pdf> Acesso em: 24 mai 2017.

AUSUBEL, D.P. (1963). ***The psychology of meaningful verbal learning***. New York, Grune and Stratton.

Blockly API. Disponível em: <https://developers.google.com/blockly/guides/overview> Acesso em: 25 mai 2017.

CSS Curso W3C Escritório Brasil Disponível em: <http://www.w3c.br/pub/Cursos/CursoCSS3/css-web.pdf> Acesso em: 25 mai 2017.

D'ANGELO, Pedro **Pesquisa quantitativa e pesquisa qualitativa: qual a diferença?**. 27 jan. 2016. Disponível em: <http://blog.opinionbox.com/pesquisa-quantitativa-e-pesquisa-qualitativa-qual-a-diferenca/> Acesso em: 26 mai 2017.

FERNANDES, Patrícia Gomes; BASTOS, Indiara Maria. **Introdução sobre as linguagens abordadas na página. Linguagem PHP. Linguagem JavaScript.** Disponível em: <http://www.inf.ufg.br/~eduardo/lp/alunos/php/PHPeJAVASCRIPT.html#JavaScript> Acesso em: 25 mai 2017.

Ferreira. E.; Eis. D. **HTML 5 Curso W3C Escritório Brasil**. Disponível em:<http://www.w3c.br/pub/Cursos/CursoHTML5/html5-web.pdf>. Acesso em: 25 mai 2017.

GOMES, Fabio **Introdução ao Java Server Pages – JSP**. 18 fevereiro 2012 Disponível em:< https://www.devmedia.com.br/introducao-ao-java-server-pages-jsp/25602> Acesso em: 11 nov 2017.

HEINEN, Eduarth. **RASPIBLOCOS: Ambiente de Programação Didático Baseado em Raspberry Pi e Blockly**. 1º Semestre de 2015. Disponível em: <http://tcc.tsi.gp.utfpr.edu.br/attachments/approvals/26/GP_COINT_2015_2_09_PROPOSTA.pdf?1455717641> Acesso em: 26 mai 2017.

JENKINS, T. (2002). ***On the difficulty of learning to program. In Proceedings of 3rd Annual LTSN_ICS Conference*** (Loughborough University, United Kingdom, August 27-29, 2002). ***The Higher Education Academy***, p.53-5

LEONARD, Fabricio **Fundamentos de Servlets**. 20 março 2013 Disponpivel em:< https://www.devmedia.com.br/fundamentos-de-servlets/3573> Acesso em: 08 nov 2017.

JUNIOR, Djalma Gonçalves Costa; MAGALHÃES, Willian Barbosa. **HTML5 E WEB SEMÂNTICA, A WEB COM SIGNIFICADO**. 24 jul. 2013. Disponível em: <http://web.unipar.br/~seinpar/2013/artigos/Djalma%20Goncalves%20Costa%20Junior.pdf> Acesso em: 25 mai 2017.

JÚNIOR, Rogério Paulo Marcon; BONIATI, Bruno Batista. **LogicBlocks: Uma Ferramenta para o Ensino de Lógica de Programação**. 01 nov. 2015. Disponível em: <http://eati.info/eati/2015/assets/anais/Longos/L7.pdf> Acesso em: 26 mai 2017.

LogicBlocks. Disponível em: <http://inf.fw.iffarroupilha.edu.br/logicblocks/index2.php> Acesso em: 26 mai 2017.

MENDES, Antonio. **Artigo Engenharia de Software 3 - Requisitos Não Funcionais**. Disponível em: <http://www.devmedia.com.br/artigo-engenharia-de-software-3-requisitos-nao-funcionais/9525> Acesso em: 26 mai 2017.

Motil, J. and Epstein, D. (2000). **JJ: a Language Designed for Beginners (Less Is More)**. Disponível em:<http://www.csun.edu/~jmotil/BeginLanguageJr.pdf > Acesso em 23 mai 2017.

NICOLDI, Antonio Carlos. **Manual do Visualg 3.0**. 22 abr. 2017. Disponível em: <https://manual.visualg3.com.br/doku.php?id=manual> Acesso em 24 mai 2017.

NOSCHANG, Luiz F. ; PELZ, Fillipi; JESUS, Elieser A. ; RAABE, André L. A. **Portugol Studio: Uma IDE para Iniciantes em Programação.** XXXIV Congresso da Sociedade Brasileira de Computação - CSBC 2014. Disponível em: <http://www.lbd.dcc.ufmg.br/colecoes/wei/2014/001.pdf > Acesso em 24 mai 2017.

POLITI, Cassio. **O que é HTML5? Por que se dá tanta importância a ele?**. 30 out. 2012 Disponível em<http://www.tracto.com.br/blog/o-que-e-html5/> Acesso em: 25 mai 2017.

RoboMind. Disponível em:
<https://translate.googleusercontent.com/translate_c?depth=1&hl=pt-BR&prev=search&rurl=translate.google.com.br&sl=en&sp=nmt4&u=http://robomind.net/en/index.html&usg=ALkJrhjPnUTe-XYzVFKurf--t1iiAdb24g> Acesso em: 26 mai 2017.

SCHIMID, Jaison. **Introdução ao Javascript.** Disponível em:
<http://www.devmedia.com.br/introducao-ao-javascript/25548> Acesso em: 25 mai 2017.

Scratch. Disponível em:
https://scratch.mit.edu/ Acesso em 23 mai 2017.

SOUZA,Cláudio Morgado. **VisuAlg - Ferramenta de Apoio ao Ensino de Programação**. 21 set. 2009. Disponivel em:
<http://editorauss.uss.br/index.php/TECCEN/article/view/234/182> Acesso em 24 mai 2017.

SOUZA, Fábio. **Embrio: Uma ferramenta de programação visual para Arduino**. 09 abr. 2015. Disponível em: <https://www.embarcados.com.br/embrio-uma-ferramenta-de-programacao-visual-para-arduino/> Acesso em: 26 mai 2017.

Tabela portugol-vs-java Disponível em:
<https://docente.ifrn.edu.br/joaoqueiroz/disciplinas/programacao-estruturada-e-orientada-a-objetos/tabela-portugol-vs-java> Acesso em 24 mai 2017.

TEIXEIRA, Leandro Bruno. **Portugol IDE.** 04 jul. 2009. Disponível em:
<https://www.vivaolinux.com.br/artigo/Portugol-IDE> Acesso em 24 mai 2017.

VENTORINI, André Eduardo; FIOREZE, Leandra Anversa. **O Software Scratch: Uma contribuição para o ensino e a aprendizagem da matemática**. 08 ago. 2014. Disponível em: <http://w3.ufsm.br/ceem/eiemat/Anais/arquivos/ed_4/MC/MC_Venturine_Andre.pdf> Acesso em 25 mai 2017.

APÊNDICES

APÊNDICE A – PRÉ-TESTE

Pré teste

Exercícios de algoritmos do pré-teste:

Exercício 1:

Crie um algoritmo que recebe a nota de um aluno e informe se este aluno está aprovado, reprovado ou em recuperação. Considerando que para ser aprovado o aluno tenha de tirar 8 ou mais, para realizar a recuperação o aluno tenha que ter nota entre 6 e 7, e se o aluno tiver nota 5 ou menor está reprovado.

Exercício 2:

O custo ao consumidor de um carro novo é a soma do custo de fábrica com a percentagem do distribuidor e dos impostos (aplicados ao custo de fábrica). Supondo que a percentagem do distribuidor seja de 28% e os impostos de 45%, escrever um algoritmo que leia o custo de fábrica de um carro e escreva o custo ao consumidor.

APÊNDICE B – PÓS-TESTE

Pós teste

Exercícios de algoritmos do pós-teste:

Exercício 1:

Crie um algoritmo que recebe a idade de uma pessoa e informe se ela é criança, adulto ou idoso. Considerando que para ser criança ela deve ter menos de 18 anos, para ser adulto deve ter de 18 a 50 anos e para ser idoso deve ter mais de 50 anos.

Exercício 2:

Joaquim tem 1,50cm e cresce 2 centímetros por ano, enquanto Maicon tem 1,10m e cresce 3 centímetros por ano. Crie um algoritmo que calcule e imprima quantos anos serão necessários para que Maicon seja maior que Joaquim.

APÊNDICE C – QUESTIONÁRIO DE SATISFAÇÃO

01. O uso da ferramenta web com programação visual facilitou a resolução dos exercícios? Por que?

_____
_____

02. Quais dificuldades que você tinha em resolver os exercícios que a ferramenta ajudou você a superar?

_____
_____

03. Você acha que a utilização de blocos de programação ajuda na hora de programar?

_____
_____

04. Dê uma nota de 1 a 10 para a ferramenta utilizada.

_____
_____