UNIVERSIDADE FEDERAL DO ESPÍRITO SANTO
CENTRO TECNOLÓGICO
PROGRAMA DE PÓS-GRADUAÇÃO EM INFORMÁTICA

AMBIENTE DE PROGRAMAÇÃO PARA CRIANÇAS DAS SÉRIES INICIAIS – NEWPROG+

CLEZIEL FRANZONI DA COSTA

VITÓRIA
2017

CLEZIEL FRANZONI DA COSTA

AMBIENTE DE PROGRAMAÇÃO PARA CRIANÇAS DAS SÉRIES INICIAIS – NEWPROG+

Dissertação apresentada como requisito parcial à obtenção do grau de Mestre em Informática. Programa de Pós-Graduação em Informática.
Universidade Federal do Espírito Santo.
Orientador: Prof. Dr. Orivaldo de Lira Tavares.

VITÓRIA
2017

CLEZIEL FRANZONI DA COSTA

AMBIENTE DE PROGRAMAÇÃO PARA CRIANÇAS DAS SÉRIES INICIAIS – NEWPROG+

Dissertação apresentada ao Programa de Pós-Graduação em Informática do Centro Tecnológico da Universidade Federal do Espírito Santo, como requisito parcial para obtenção do Grau de Mestre em Informática.

Aprovada em 11 de agosto de 2017.

COMISSÃO EXAMINADORA

_____
Prof. Dr. Orivaldo de Lira Tavares
Universidade Federal do Espírito Santo
Orientador

_____
Prof. Dr. Crediné Silva de Menezes
Universidade Federal do Espírito Santo
Examinador interno

_____
Prof. Dr. Eloi Luiz Favero
Universidade Federal do Pará
Examinador externo

Dedico aos meus pais, Helena e Benedito, sem os quais não seria possível chegar até aqui, e, com muita fé em Deus e dedicação, sempre me incentivaram a alcançar meus objetivos, a minha noiva Carla, pelo apoio e compreensão de sempre e a toda minha família, que sempre me incentivou a lutar e a vencer.

AGRADECIMENTOS

Agradeço a Deus que, através de minha família, concedeu-me toda força necessária para conseguir conquistar mais esta vitória, e por me conceder sabedoria sempre que necessário;

Aos meus amigos que suportaram minha ausência durante esse período;

Aos meus professores, principalmente ao Prof. Orivaldo de Lira Tavares, pelo direcionamento na conquista do saber e pelo incentivo em criar e gerar oportunidades.

“Todo mundo neste país deveria aprender a programar um computador, porque ele nos ensina a pensar. ”

Steve Jobs

RESUMO

Este trabalho apresenta o NewProg+, um ambiente digital para crianças das séries iniciais aprenderem os conceitos básicos da programação de computadores. O NewProg+ dispõe de um editor de atividades para ser usado pelos educadores, uma Linguagem Visual de Programação (LVP) e um avaliador de programas para serem usados pelos aprendizes. O NewProg+ é uma atualização do ambiente NewProg e se diferencia do mesmo, principalmente pela inclusão das estruturas de repetição, testes de condição e subprogramas em sua linguagem visual de programação. Aprender programação de computadores exercita diversas habilidades nas crianças, como o raciocínio lógico e a capacidade de resolver problemas, e nos testes com a LVP apresentada, foi possível comprovar que os alunos foram capazes de programar as suas soluções no NewProg+.

Palavra-chave: *Newprog+, aprendizagem de programação, programação para crianças, linguagem visual de programação.*

ABSTRACT

This work presents NewProg+, a digital environment for children in the early grades to learn computer programming. NewProg+ has an activity editor for use by educators and a Visual Programming Language (LVP) and program evaluator for use by apprentices. NewProg+ is an update of the NewProg environment and differs from it, especially by including repetition structures, condition tests and subprograms in its visual programming language. Learning computer programming exercises a variety of skills in children, such as logical thinking and problem solving skills, and in the tests with the LVP presented, it was possible to prove that the students were able to program their solutions in the NewProg+.

Keyword: *Newprog+, programming learning, programming for children, visual programming language.*

LISTA DE FIGURAS

LISTA DE QUADROS

LISTA DE SIGLAS

CLR – *Common Language Runtime*

LP – Linguagem de Programação

LVP – Linguagem Visual de Programação

WF - *Windows Workflow Foundation*

WPF – *Windows Presentation Foundation*

SUMÁRIO

1 INTRODUÇÃO

As habilidades cognitivas das crianças influenciam em seu cotidiano e principalmente em sua vida escolar. Com esta proposta, de acordo com Moraes (1997), o Laboratório de Estudos Cognitivos do Instituto de Psicologia (LEC/UFRGS) desenvolveu, no final da década de 70, uma pesquisa, prioritariamente, com crianças das escolas públicas que apresentavam dificuldades na aprendizagem da leitura, escrita e cálculos. O objetivo desse estudo foi compreender o raciocínio lógico-matemático dessas crianças e quais as possíveis intervenções que poderiam promover a aprendizagem autônoma destes indivíduos. Com essa pesquisa, pôde-se comprovar, desde então, que havia uma considerável potencialidade na utilização do computador para o desenvolvimento cognitivo destas crianças.

Considerando as dificuldades encontradas no processo tradicional de aprendizagem de programação, apontadas por Oliveira *et* al. (2014), como a limitação do vocabulário de instruções das linguagens de programação e a dificuldade em utilizar estas linguagens, a abordagem encontrada aqui apresenta a ideia de atualização e aprimoramento para o NewProg, o NewProg+, um software baseado em uma linguagem visual de programação para a aprendizagem de programação de computadores para crianças com a faixa etária entre 5 e 8 anos. Esta linguagem visual permite que os alunos interajam com a ferramenta somente utilizando o mouse para a edição dos programas, com a finalidade de construir algoritmos para a resolução de diversos problemas.

Os desafios apresentados ao estudante pelo ambiente do NewProg+ se assemelham a interface de um jogo, portanto o aprendiz, ao desenvolver a atividade proposta, se depara com um ambiente similar a um *game*. De acordo com Fessakis e Lappas (2013), as crianças são atraídas e motivadas cada vez mais ao usar jogos de computador, por isso é importante explorar a viabilidade de desenvolver o potencial criativo das crianças, principalmente das mais novas, através do uso de jogos de computador.

O NewProg proporciona uma sensação de entretenimento, enquanto pode favorecer o desenvolvimento de habilidades cognitivas, importantes em diversas situações do cotidiano do aprendiz, tais como o planejamento e a resolução de problemas. O

NewProg, de acordo com Torezani (2013), é um software desenvolvido com o objetivo de auxiliar as crianças na aprendizagem dos conceitos de programação de computadores. O NewProg+ mantém as funcionalidades do NewProg e se diferencia com a atualização de suas funcionalidades, oferendo ao usuário, além das sequências básicas de instruções, testes condicionais e estruturas de repetição.

1.1 Definição do Problema

A aprendizagem de programação de computadores poderia estar inserida no contexto de aprendizado dos alunos desde as séries inicias, visto que, segundo Castro, Júnior e Menezes (2002), este processo de aprendizagem envolve o desenvolvimento de diversas habilidades cognitivas, principalmente a capacidade de abstração.

Ao analisar os dados apresentados pela folha de São Paulo, em 2012 existiu uma evasão de 28% dos alunos dos cursos de matemática e de ciências da computação. Dentre os vários motivos que provocam a evasão, estão as dificuldades dos alunos com o desenvolvimento da lógica de programação e das diversas sintaxes presentes nas diferentes linguagens de programação (RIBEIRO, BRANDÃO E BRANDÃO 2012).

No trabalho apresentado por Oliveira *et al.* (2014), os autores afirmam que é difícil para a maioria dos ingressantes dos cursos das áreas aprenderem certos conteúdos referentes à Ciência da Computação, destacando-se as disciplinas que envolvem algoritmos, lógicas de programação e cálculo, visto que em diversas vezes essas disciplinas propõem um “nova forma de pensar” e requerem diversas habilidades, que provavelmente não foram desenvolvidas no ensino regular, o que dificulta e traz resultados diferentes do esperado para o aprendiz. Assim, é necessário o desenvolvimento dessas habilidades cognitivas, desde as séries iniciais.

1.2 Justificativa

Os métodos tradicionais de aprendizagem de programação são considerados cansativos e de difícil assimilação pelos alunos de curso superior e da área de informática ou de engenharia. A proposta de desenvolver a habilidade de programar

computadores em cada criança das séries iniciais visa seu desenvolvimento cognitivo e também a construção de uma cultura, permitindo a ela um primeiro contato com esta área, de forma a prepará-la melhor para seu ingresso nos cursos técnicos, de tecnologias e engenharia.

A proposta do NewProg+ é inserir no processo de aprendizagem uma ferramenta capaz de possibilitar ao aluno uma linguagem visual, capaz de desenvolver seu raciocínio lógico, que é uma das maiores dificuldades de quem inicia o processo de aprendizagem de programação de computadores.

Existem algumas propostas de ambientes de aprendizagem de programação de computadores para crianças, porém poucas com uma linguagem visual de programação e que permita ao professor acompanhar as atividades desenvolvidas pelos aprendizes. Com base neste ponto de vista, há a necessidade de uma linguagem visual de programação para auxiliar as crianças a aprenderem a programar, de modo a exercitarem as habilidades cognitivas decorrentes dessa atividade e que este processo de aprendizagem permita o acompanhamento pelo educador responsável.

1.3 Objetivos Gerais

Estender o ambiente computacional de aprendizagem programação – NewProg, para permitir atividades de programação que precisem testar condições e especificar uma quantidade certa de repetições de blocos de comandos. Esse novo ambiente – NewProg+, deve continuar oferecendo todas as características do NewProg: uma linguagem visual de programação; um simulador da execução de cada programa; um editor de atividades, a ser usado pelo professor, e um gravador das ações de cada estudante no ambiente.

1.4 Objetivos Específicos

Os objetivos específicos desta dissertação contemplam:

- Estender a linguagem de programação visual do NewProg para permitir o uso de testes condicionais e a repetição de blocos de comandos;
- Auxiliar os educadores no processo de acompanhamento da aprendizagem de programação de computadores por seus alunos;
- Oferecer aos educadores uma ferramenta que permita a construção de atividades para o aprendizado de programação de computadores;
- Possibilitar acompanhamento individualizado das atividades feitas pelos aprendizes, a fim de verificar a coerência do processo utilizado pela criança.

1.5 Questões de Investigação

Durante o desenvolvimento desta pesquisa, foram levantadas algumas Questões de Investigação (QI), ou seja, questões que seriam analisadas com a finalidade de buscar uma determinada solução dentro do produto desta pesquisa. Dentre estas questões, destacam-se:

QI1 - Quais os requisitos de um ambiente digital para crianças das séries iniciais aprenderem programação de computadores?

QI2 - Como projetar um ambiente digital para as crianças das séries iniciais aprenderem programação de computadores?

QI3 - Como avaliar a usabilidade desse ambiente?

1.6 Metodologia

Para o desenvolvimento do ambiente computacional proposto, este projeto de pesquisa foi elaborado de acordo com os seguintes passos:

O primeiro passo consiste em conhecer o que realmente é importante para ser incorporado ao ambiente, de forma que seja alcançado o objetivo final de propor um ambiente de programação para crianças, projetado com uma linguagem visual e com ícones e símbolos que são facilmente compreendidos pelas crianças das séries iniciais. Para isto, foi realizada uma revisão bibliográfica, com a finalidade de identificar os requisitos essenciais ao sistema proposto e aprimorar a sua versão anterior, com

o objetivo de promover a aprendizagem de programação de computadores de forma descontraída e atrativa.

O segundo passo foi construir a ferramenta proposta, que se trata de um sistema desenvolvido para a web.

Antes de iniciar a terceira etapa, foi preciso estabelecer parceria com pelo menos uma escola com o perfil adequado de alunos, com idades entre 5 a 8 anos, para iniciar os testes de implementação e validação do software. Com a escola selecionada, a terceira etapa foi implantar e validar o sistema, gerando relatórios para validar sua usabilidade e seus resultados. Essa validação foi realizada com os alunos da escola e com seus professores, que são os criadores das atividades no ambiente.

2 APRENDIZAGEM DE PROGRAMAÇÃO DE COMPUTADORES

Neste capítulo serão apresentados alguns fatores que implicam na aprendizagem da programação de computadores e ainda que justificam o desenvolvimento desta habilidade nas crianças, o que será fundamental, principalmente para aquelas que irão ingressar na área de ciências da computação.

O presente capítulo traz ainda um estudo sobre outros trabalhos que possuem como objetivo a aprendizagem de programação de computadores para crianças e explicita a forma que estes propõem o cumprimento de suas propostas com exemplos e imagens de suas interfaces de uso.

2.1 Por que ensinar programação para crianças?

O *National Research Council*, em 1999, definiu uma ideia chamada de Fluência em Tecnologia da Informação, ou ainda Fluência Digital. Esta ideia enfatiza a capacidade que os alunos possuem para aprenderem, de forma independente, uma nova tecnologia, à medida em que a Tecnologia da Informação (TI) evolui naturalmente ao longo do tempo. Um ponto desejável na Fluência em TI é a aprendizagem de programação de computadores. Aprender a programar um computador envolve a decomposição de um problema em uma sequência de etapas, identificando precisamente os passos para que este problema possa ser resolvido de forma eficiente. A programação de computadores abrange uma avançada forma de pensar, a abstração para desenvolver os algoritmos, habilidade que não pode ser adquirida apenas com a matemática básica (KIND, 2015, p. 3).

Segundo o mesmo autor, geralmente, as escolas de ensino fundamental não ensinam programação de computadores, o que, de acordo com este, tira a oportunidade destes estudantes de adquirirem significativas habilidades que auxiliam na construção do pensamento.

Para Fessakis, Gouli e Mavroudi (2013), as escolas de ensino fundamental não ensinam programação para seus alunos principalmente pelas avançadas sintaxes que a maioria das Linguagens de Programação (LP) possuem. No entanto, já existem

diversos projetos que visam facilitar este processo, como os ambientes de programação derivados da Linguagem Logo.

Com o desenvolvimento destas novas LPs, pesquisadores descobriram que estudantes de 5 (cinco) e 6 (seis) anos de idade são capazes de aprender a maioria dos conceitos básicos de programação, desde que haja o adequado acompanhamento por um professor responsável ou outro adulto que saiba instruí-lo (KIND, 2015).

Segundo Baytak e Land (2011), além de ajudar no desenvolvimento cognitivo dos alunos, a programação de computadores também os beneficia emocionalmente. De acordo com os autores, a inclusão de programação nas matrizes curriculares destes alunos permite que estes exerçam uma maior expressão pessoal, manifestando-se mais livremente, através dos algoritmos, suas ideias, pensamentos e sentimentos em relação ao conteúdo que eles estão estudando.

Para Fessakis, Gouli e Mavroudi (2013), com as LPs apropriadas, os alunos são capazes de compreender os conceitos básicos da programação de computadores. Em um estudo de caso desenvolvido, estes mesmos autores criaram um ambiente para programação de computadores, com uma linguagem simples e, segundo eles, apropriadas para as crianças com faixa etária entre 5 e 8 anos de idade. Neste estudo de caso, de acordo com os autores, os alunos envolvidos na pesquisa tiveram a oportunidade de desenvolver habilidades pertinentes ao ensino da matemática, visto que o ambiente envolvia a utilização de números, ângulos e comparação entre valores. Neste mesmo estudo, os alunos tiveram a oportunidade de aprender os conceitos básicos da programação de computadores, como execução de comandos, instruções sequenciais de comandos, testes de erros e depuração de um programa.

Kind (2015) afirma que, além de ser propiciamente motivador e de despertar interesse nas crianças, a programação de computadores é um ótimo meio com o qual os alunos podem desenvolver e praticar importantes habilidades necessárias para a resolução de problemas de seu dia a dia escolar.

2.2 A Programação e o Desenvolvimento Cognitivo

De acordo com Oliveira *et* al*.* (2014), os conhecimentos referentes às ciências da computação estão restritos aos alunos dos cursos superiores e técnicos das áreas afins. O autor ainda ressalta que o ideal seria que cada cidadão compreendesse pelo menos os princípios básicos dessa área. Em consequência disso, estudos apresentados pelo mesmo autor mostram uma previsão de uma grande demanda por profissionais das áreas de tecnologia até o ano de 2018, devido à escassez de profissionais qualificados.

Conforme já mencionado neste trabalho, Castro*,* Júnior e Menezes (2002) afirmam que o processo de aprendizagem de programação está relacionado com o desenvolvimento de diversas habilidades cognitivas, incluindo a capacidade de raciocínio lógico e de abstração.

Wangenheim, Nunes e Santos (2014), afirmam que as crianças deveriam aprender computação desde o ensino fundamental, mas se faz necessário a aprendizagem da proficiência digital, que se diferencia com a aprendizagem e a aplicação produtiva das novas tecnologias.

Tavares, Menezes e Nevado (2012) citam que o aluno reinicia suas tentativas de construir um programa de computador após tentativas anteriores infrutíferas, buscando uma solução correta para resolver o problema. O intuito do aluno é descobrir e corrigir seus erros em um processo de tentativa e erro, até desenvolver uma solução correta. Segundo os autores, esse processo de tentativa e erro pode ser mais prolongado ou mais curto, dependendo da experiência anterior de cada aluno na resolução do problema. Os autores ainda relatam que o processo de correção dos erros é fundamental para o sucesso dos alunos. Neste processo é bem-vinda a existência de mecanismos de autorregulação, para apontar formas de aperfeiçoar a solução construída. Além disso, afirmam que a construção de uma solução correta não significa que o aprendiz já dominou o processo de construção de soluções para um dado problema. A aprendizagem sobre esse processo somente se consolida quando o aprendiz desenvolve outras soluções diferentes para o mesmo problema e reflete sobre o processo de construção. Quando o aprendiz tem oportunidade de comparar suas soluções com as dos seus colegas, a consolidação da aprendizagem pode ser acelerada.

Para Wangenheim, Nunes e Santos (2014), a programação de computadores é fundamental para a computação, tornando-se necessário o desenvolvimento da competência para criar programas. Os autores ainda afirmam que com o aprendizado de programação se estimula a aprendizagem do pensamento computacional, o que envolve um conjunto de conceitos computacionais, como a abstração, recursão e iteração, entre outros.

2.3 A linguagem LOGO e uma nova proposta para a aprendizagem de programação de computadores

A linguagem LOGO é uma Linguagem de programação desenvolvida por Seymour Papert no ano de 1985. Esta linguagem apresenta diversas características planejadas para implementar uma metodologia de ensino baseada na utilização do computador. Esta teoria constitui-se num ambiente aberto que permite ao aprendiz construir soluções próprias, refletir sobre elas e reelaborá-las (SILVA e MORO, 1998).

Papert criou uma abordagem denominada construcionista. Esta abordagem diz que o aprendiz é capaz de construir o seu próprio conhecimento utilizando um computador (PAPERT, 1996).

Baseando-se nesta abordagem construicionista, Lataille (1990, p. 111) diz que a Linguagem Logo segue a ideia de levar a criança a trabalhar com o erro, em auxiliar o aluno no momento que o erro acontece e a fazer com que este aluno perca o medo de cometê-lo, perca o medo de ser punido ou humilhado por ter cometido o erro. Essa ideia de erro é instrumentada na teoria de Piaget, onde o erro é integrante do processo de construção do conhecimento.

2.4 Trabalhos Correlatos

Existem diversos outros trabalhos similares ao NewProg+ e com o mesmo objetivo, que é a aprendizagem dos conceitos iniciais de programação de computadores, porém todos eles possuem suas particularidades, mesmo a maioria sendo derivada da linguagem LOGO ou baseada na estrutura de blocos apresentada pelo Scratch.

Mesmo com várias semelhanças, o NewProg+ se diferencia com a Linguagem Visual de Programação apresentada, visto que esta linguagem não é encontrada nestes trabalhos que são semelhantes ao ambiente aqui proposto.

2.4.1 O code.org e uma nova visão

Criado em 2013 pelo Irariano Hadi Partovi, um investidor de grandes empresas como Facebook e Dropbox, juntamente com o apoio de seu irmão Ali, o code.org mantém um sítio eletrônico (Figura 1) com o objetivo de promover a aprendizagem dos passos iniciais na programação de computadores para crianças, jovens e adultos. Em 2011, a ideia de Hadi era apenas criar um vídeo com grandes nomes da informática para incentivar o aprendizado de programação. Ainda em 2013, o vídeo possuía mais de 11 milhões de visualizações. O primeiro a gravar uma mensagem para o vídeo seria Steves Jobs, que faleceu antes das gravações começarem. O code.org obteve o apoio inicial de 60 pessoas, entre elas estão Bill Gates (Microsoft), Mark Zuckenberg (Facebook), Jack Dorsey (Twitter), além do cantor Will.i.am e do político Michel Bloomberg. A iniciativa contou com o investimento das empresas Google, Microsoft, LinkedIn e Amazon. No final de 2013, o code.org lançou a “Hora do código 2013”, uma iniciativa onde qualquer pessoa poderia dedicar uma hora de seu tempo para aprender a programar (GERALDES, 2014).

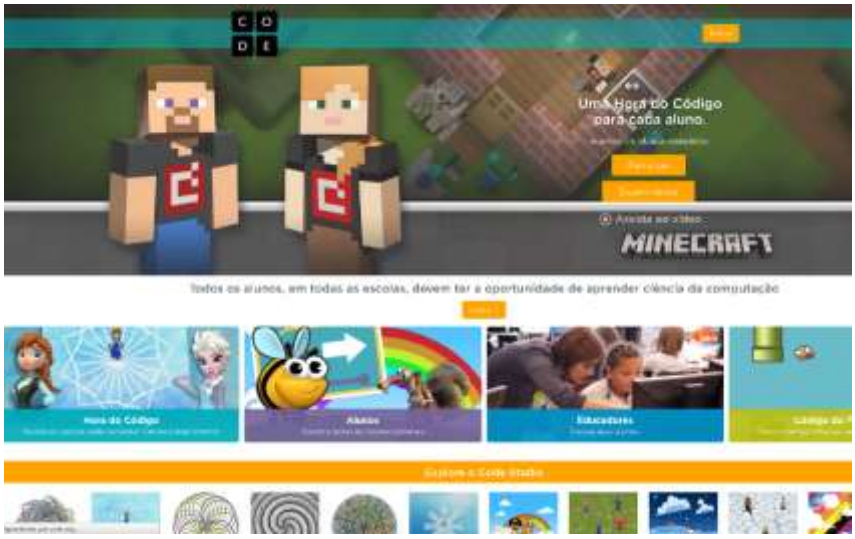

Figura 1 - Sítio eletrônico code.org

2.4.2 Scratch

O Scratch é uma linguagem de programação de blocos. Permite a construção de soluções por meio de blocos de comandos, visto que estes blocos são visuais, proporcionando um contanto com muitas informações visuais, representadas por imagens e ainda com a possibilidade da utilização de sons. As atividades são desenvolvidas a partir de blocos que se encaixam e são divididos em 8 categorias: Movimento, Aparência, Som, Caneta, Sensores, Controle, Operadores e Variáveis (OLIVEIRA *et* al., 2014). A Figura 2 ilustra a página principal do site do scratch.

O *Scratch* foi desenvolvido no *Massachusetts Institute of Technology* – MIT. Esta linguagem foi inspirada nos princípios construtivistas da Linguagem Logo. Seu principal objetivo é auxiliar a aprendizagem de programação de computadores de forma lúdica e muito criativa, podendo ser usada por crianças e pessoas que não possuem nenhum conhecimento de programação (OLIVEIRA *et* al., 2014).

Figura 2 - Página inicial do site do Scratch

O ambiente Scratch permite a criação de animações, histórias interativas e ainda permite a construção de jogos. Para as animações, é possível utilizar tanto os personagens presentes no próprio ambiente, quanto qualquer outra imagem que

queira utilizar. Com tantas possibilidades, os alunos são estimulados à criatividade e a imaginação, visto que não são tratados apenas como usuário do software, mas como gerados de conteúdo. A Figura 3 exibe uma atividade sendo construída no scratch, em sua versão online.

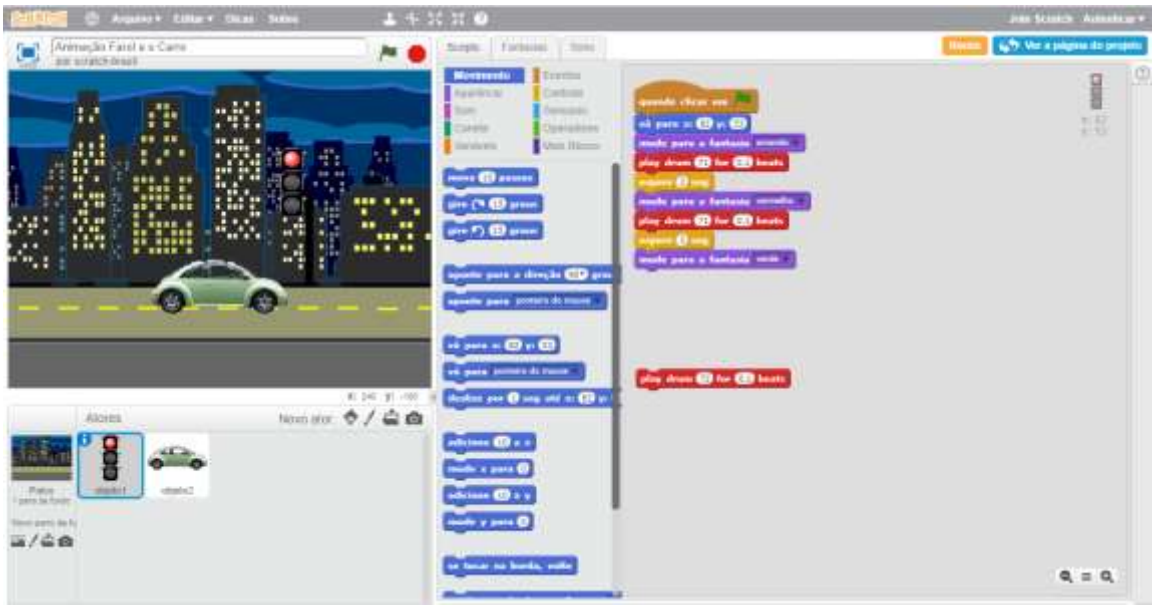

Figura 3 - Exemplo de atividade no scratch

2.4.3 CodeCombat

Mais uma iniciativa que objetiva a aprendizagem da programação de computadores é o CodeCombat, um jogo com o intuito de proporcionar uma divertida forma para a aprendizagem da programação de computadores. Durante o jogo, os jogadores poderão escolher entre as linguagens de programação Phyton, JavaScript, Lua e CoffeeScript para cumprirem seus objetivos, conforme mostra a Figura 4.

De acordo com os autores, o CodeCombat pode ser utilizado até mesmo por crianças, a partir dos 6 anos de idade e já conta com diversas traduções em diferentes idiomas. O jogador pode, ainda, selecionar um personagem para ser seu Avatar durante o jogo. Ao cumprir determinada etapas e atingir pontuações específicas, ele pode selecionar novos avatares, que serão disponibilizados aos poucos, e assim consegue novos

recursos para aumentar ainda mais suas habilidades em programação (CODECOMBAT, 2016).

O CodeCombat conta ainda com ambientes diferenciados para alunos e para professores, onde um professor pode propor as tarefas que seus alunos poderão realizar, obtendo, em seguida, uma estatística completa do rendimento deste aluno.

Figura 4 - Selecionando a linguagem de programação desejada

A Figura 5 apresenta o painel do professor no codecombat. Neste painel o professor possui acesso a todos alunos presentes em sua sala. Aqui também é possível visualizar e acompanhar o rendimento dos alunos, com a opção de visualizar, através de gráficos comparativos, e comparar o rendimento de um determinado aluno em

relação à turma. Para exibição deste gráfico comparativo, basta clicar sobre o nome do aluno na página administrativa do professor, ilustrada na Figura 5.

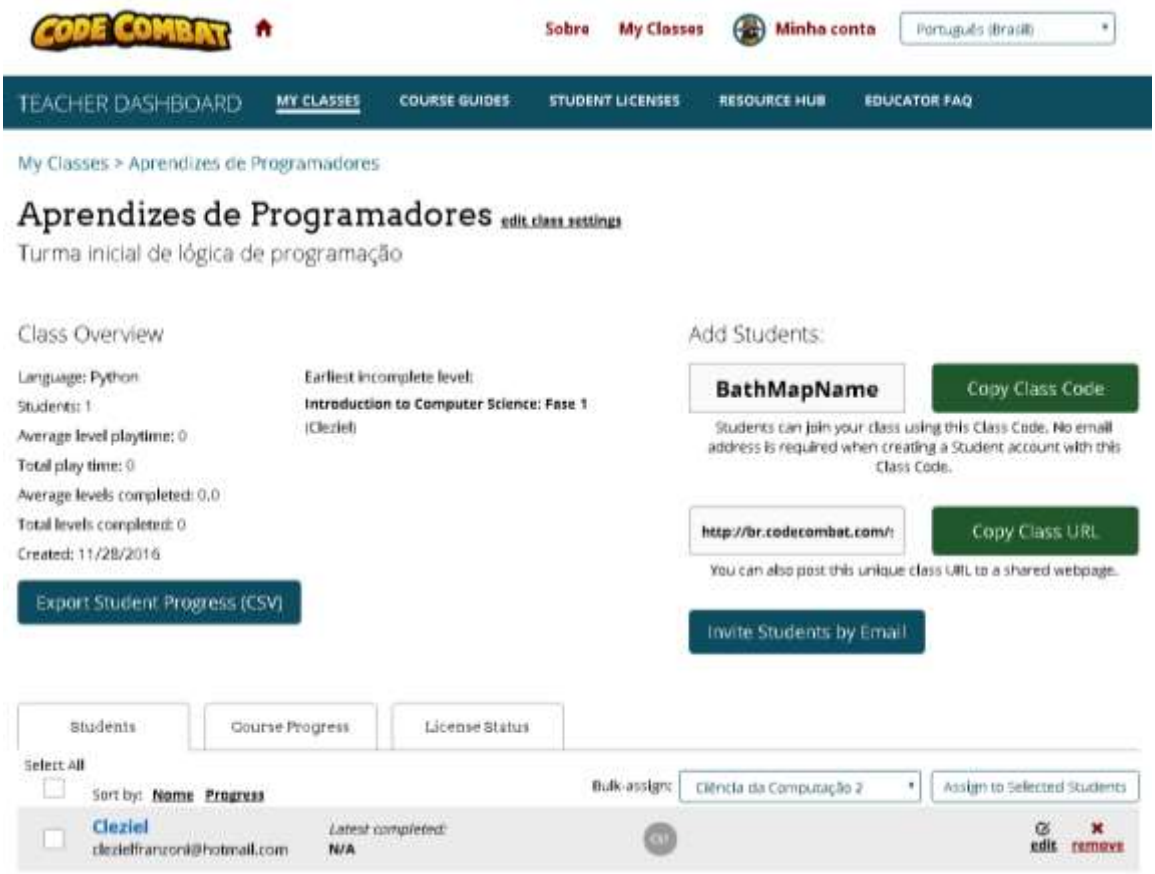

Figura 5 - Painel do Professor no CodeCombat

A Figura 6 exibe o ambiente do aluno no CodeCombat. Neste espaço o aluno terá acesso a todas as suas turmas, visto que ele pode fazer parte de quantas turmas forem necessárias, e assim selecionar uma para entrar na sala de aula e realizar as atividades preparadas para ele por seu professor responsável.

Um aluno pode ingressar em uma turma de duas possíveis formas: por um código disponibilizado pelo professor responsável, ou ainda por um *link*, também disponibilizado pelo professor. Quando o ingresso na turma se dá por meio do *link* compartilhado, o aluno precisa apenas realizar seu cadastro no ambiente.

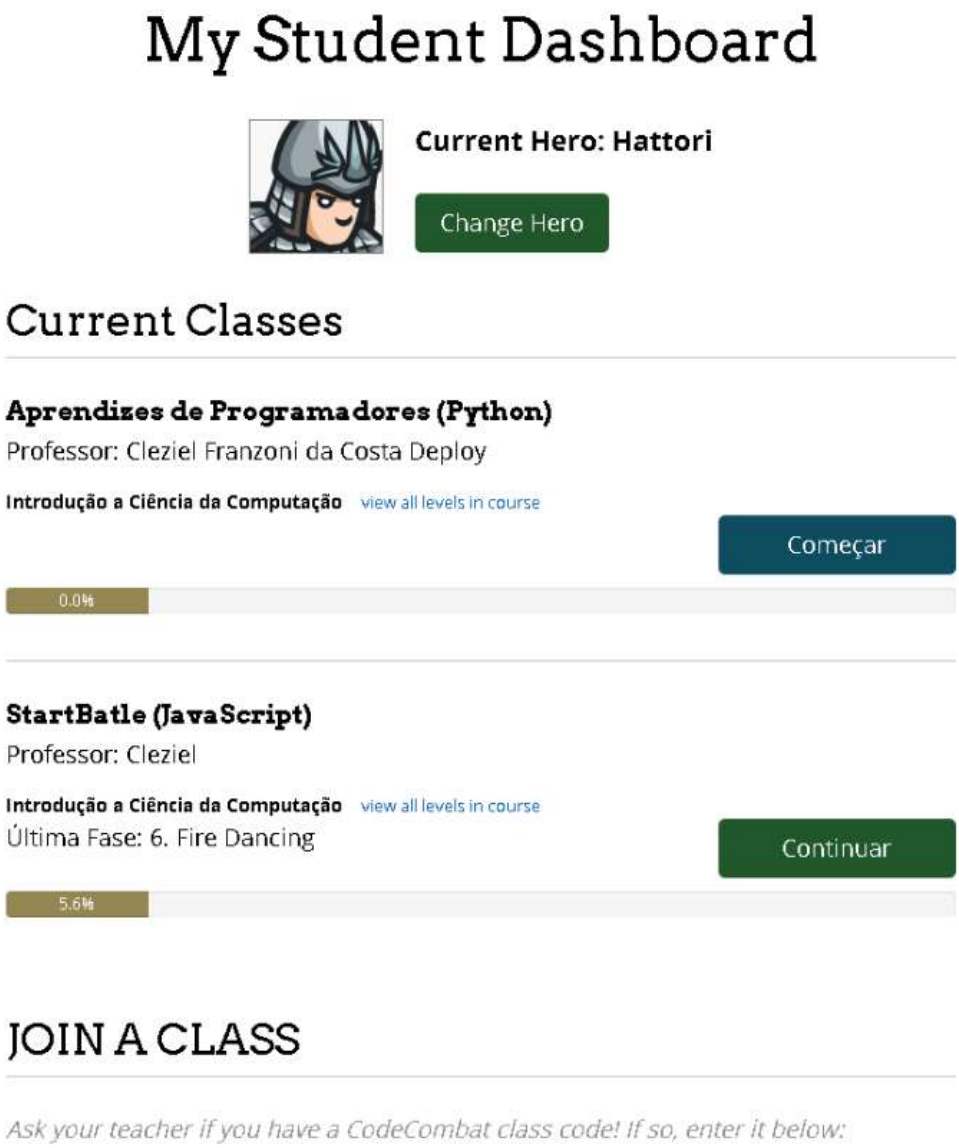

Figura 6 - Ambiente do Aluno participante de uma classe

A Figura 7 mostra o ambiente do aluno, já em modo de execução de uma possível solução para sua atividade atual. Neste espaço, após digitar o código para solucionar os problemas apresentados, o aluno pode executar a sua solução e ver seu código “criando vida”, por meio de um avatar que executa todas as instruções que foram digitadas em forma de códigos.

Figura 7 - Ambiente do Jogo em execução

2.4.4 Code Hunt

O Code Hunt é uma iniciativa desenvolvida pela Microsoft com o intuito em incentivar o aprendizado da programação de computadores. O Code Hunt é um jogo, onde o jogador, que assume o papel do “Caçador de Código”, deve descobrir e caçar os fragmentos perdidos do código que deseja completar. O jogador ganha pontos para cada código completado e ainda consegue pontos bônus pelas “soluções elegantes”. No Code Hunt, o jogador pode escolher se deseja utilizar a linguagem de programação Java ou a linguagem C# (MICROSOFT RESEARCH, 2016).

A aprendizagem no Code Hunt acontece durante todo o jogo, onde os jogadores aprendem sobre operadores aritméticos, estruturas condicionais, estruturas de repetição, algoritmos de pesquisa e outros mais.

Além de contribuir para a aprendizagem de uma nova linguagem de programação, o CodeHunt permite àqueles que já programam a oportunidade de aperfeiçoar suas habilidades de programação com uma forma simples e bem divertida.

A Figura 8 ilustra o painel do usuário no codehunt. O usuário, após escolher a sua linguagem de programação, é desafiado a resolver os problemas disponíveis e assim “caçar” o código correto para uma melhor solução.

Figura 8 – Codehunt

Se o “jogador” conseguir capturar o código com uma solução correta, o ambiente lhe atribui um *score*, onde o total de pontos está relacionado a diversos aspectos, como a quantidade de linhas digitadas e a “elegância” do código digitado. A Figura 9 mostra a tela de *score,* que é exibida sempre que o código for “capturado”.

Figura 9 - Código capturado no codehunt

3 REQUISITOS PARA O AMBIENTE COMPUTACIONAL

Este capítulo aborda a origem do NewProg+ e os requisitos desejáveis para que o ambiente criado fosse capaz de promover o aprendizado da programação de computadores para seu público alvo, que são as crianças com faixa etária entre cinco e oito anos de idade.

O NewProg+ é uma extensão e atualização do NewProg, assim possui os mesmos requisitos que o mesmo para o desenvolvimento de seu ambiente computacional. Este capítulo possui o objetivo de realizar uma conexão entre toda fundamentação abordada, tanto para o NewProg quanto para o NewProg+, e os objetivos traçados para esta linguagem visual de programação. De acordo com Torezani (2014), além dos requisitos teóricos, ainda foram considerados os levantamentos feitos com professores, por pesquisa etnográfica.

3.1 A Origem

Há continuamente um crescimento no quantitativo de pesquisas que enfatizam o uso da programação de computadores por crianças. Entretanto, os ambientes existentes, como *Scratch*, *Toon Talk* e *Squeak Etoys*, não permitem um acompanhamento detalhado da evolução da aprendizagem dos alunos, permitindo apenas a criação e execução das atividades. (TOREZANI, 2014, p. 48).

O autor ainda afirma que, com as ferramentas citadas anteriormente, os educadores não podem visualizar a evolução das atividades realizadas pelos alunos, o que os impossibilita de acompanharem e verem quais são as principais dificuldades que os aprendizes enfrentam.

Segundo o mesmo autor, é essencial a uma educação que atenda as exigências deste século, a utilização de recursos que estimulem a criatividade dos estudantes e permita, ainda, o acompanhamento detalhado pelos educadores.

Fessakis, Gouli e Mavroudi (2013) defendem que a aprendizagem de programação permite o desenvolvimento de conhecimentos em diversas áreas. Seu trabalho apresenta um estudo de caso, onde são levantados certos conceitos computacionais

que podem ser utilizados para crianças com faixa etária entre 5 e 8 anos de idade aprenderem a programar. Sua pesquisa utilizou dois ambientes, o *Ladybug Maze* (Figura 10) e o *Ladybug Leaf* (Figura 11), ambos disponíveis gratuitamente na *National Library of Virtual Manipulatives* (NLVM) através o sítio , da UTAH *State University*, nos EUA. A partir da leitura deste trabalho, surgiu a ideia de desenvolvimento do NewProg e, consequentemente, a sua atualização, o NewProg+.

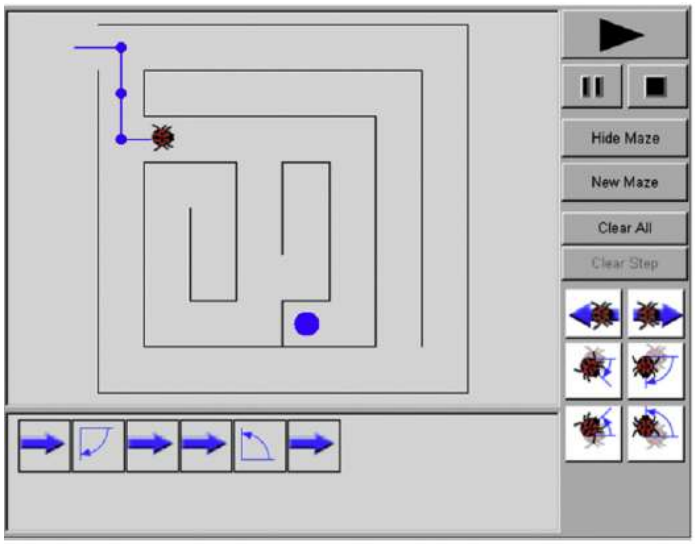

Figura 10 – Ladybug Maze

Fonte: Fessakis, Gouli e Mavroudi (2013, p. 90)

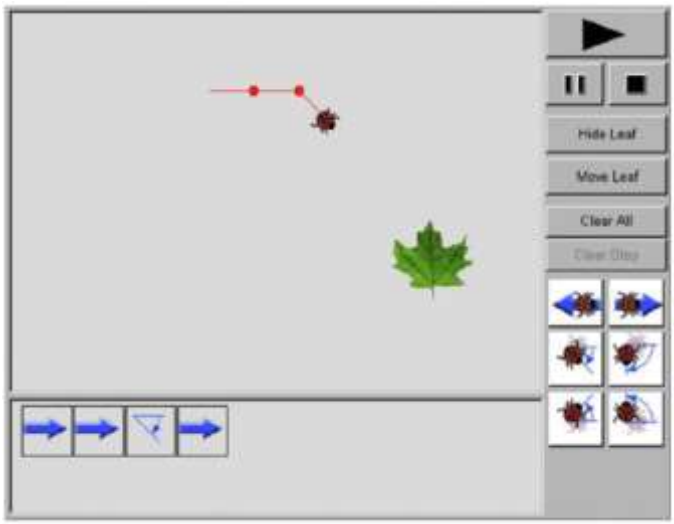

Figura 11 - Ladybug Leaf

Fonte: Fessakis, Gouli e Mavroudi (2013, p. 90)

3.2 Requisitos do ambiente

O ambiente precisa ser simples, fácil de utilizar, intuitivo, mas ao mesmo tempo motivador e desafiador. De acordo com Law, Lee e Yu (2010), os desafios, juntamente com a clareza e objetividade do que os alunos realmente deverão fazer, são aspectos que os motivariam a realizarem as atividades.

Os requisitos desejáveis para o NewProg+ são os mesmos do NewProg, dessa forma, são apresentados, logo abaixo, os requisitos funcionais do sistema, assim como uma descrição resumida sobre eles.

- Simplicidade – este ambiente deve ser simples, com uma interface de utilização clara e objetiva, objetivando facilitar a interação dos usuários com o mesmo;
- Escolha do Avatar – o usuário precisa escolher um avatar que mais lhe agrade, entre as opções disponíveis, com a finalidade de motiva-lo um pouco mais a fazer as atividades;
- Tela Limpa – é necessário que o ambiente possua apenas as informações relevantes ao usuário, sem recursos e informações desnecessárias;
- Agrupamento das Atividades – todas as atividades deverão ser agrupadas, de acordo com seu contexto, visando facilitar aos usuários na busca pelas mesmas;
- Acompanhamento das Atividades – necessita conter os recursos necessários para permitir que os educadores acompanhem individualmente os seus alunos;
- Cadastramentos Diversos – o ambiente proverá ferramentas para cadastro e manutenção de diferentes perfis de usuários;
- Executor de Atividades – o executor de atividades precisa dispor de mecanismo necessário para a inclusão e exclusão de comandos para a realização das atividades;
- Ajuda - é necessário que o ambiente disponha de um recurso de “ajuda”, adequado à faixa etário do usuário;
- Ambientação – de forma a facilitar o aprendizado, o ambiente possuirá ferramentas que permitam que as crianças interajam com o *software* para se ambientarem na utilização do mesmo;

- Editor de atividades – o ambiente deve conter um editor que permita que os educadores criem e editem as atividades desejadas, tornando assim o ambiente adaptável e rico em atividades;
- Simulador – com a finalidade de testar as atividades antes de serem disponibilizadas aos alunos, o ambiente deverá prover recursos que permitam as simulações e testes destas pelos educadores.

Além dos requisitos funcionais citados, que são comuns ao NewProg e ao NewProg+, existem alguns requisitos que são exclusivos do NewProg+. São eles:

- Estruturas de Repetição – o ambiente deverá dispor de recursos necessários para a interpretação e execução das estruturas de repetição presentes na linguagem visual e utilizada pelos alunos;
- Testes Condicionais – assim como nas estruturas de repetição, o ambiente compreenderá os testes condicionais presentes na linguagem visual, como uma verificação se uma porta está aberta ou não;
- Alteração das imagens do ambiente – para suprir as demandas específicas de determinados educadores, assim como planejamentos pedagógicos específicos de cada educador, o ambiente permitirá a alteração das imagens presentes, podendo assim um educador incluir imagens que ilustram um cenário específico de determinada disciplina ou conteúdo;
- Gerenciamento de avatares – é desejável que o administrador possa inserir ou remover opções de avatares no ambiente, podendo assim incluir avatares diferentes de acordo com as faixar etárias trabalhadas.

3.3 Interface do ambiente

A interface do ambiente deve ser agradável e atrativa, de forma a obter a maior aceitação possível de seu público alvo, que são crianças com faixa etária entre cinco e oito anos de idade.

Para Torezani (2014, p. 49), a interface cuida da ligação do usuário com a máquina, sendo incumbida de facilitar no processo de comunicação. Ainda de acordo com o autor, um software educativo tem que se apresentar com muita qualidade, de modo a facilitar o processo de aprendizagem. Isso permite que o aluno atinja o objetivo

proposto com mais facilidade, rapidez, sem medo e sem ansiedade, conseguindo aumentar o seu desempenho.

Quadro 1 - Itens considerados para a criação da interface do NewProg e do NewProg+

Item de Avaliação	Descrição	Ambiente Proposto
Facilidade aprendizado	O primeiro contato com a interface precisa ser tranquilo e sem problemas. É recomendável que na função de ajuda do software apareça um tutorial, para assim permitir que o aluno experimente diferentes estratégias de solução associadas ao seu problema.	A ajuda do ambiente apresenta uma linguagem simples e adequada à faixa etária do aluno que o utiliza. Seus ícones e botões possuem aspectos visuais bastante atrativos, do ponto de vista da criança.
Facilidade de uso	Após o aprendizado da interface, o nível de dificuldade exigido para manter o seu uso não deve ser alto. O software permiti a independência de uso e ação, sem assistência humana. O aluno é capaz de lembrar com facilidade o manuseio da interface, mesmo após algum tempo (uma semana, por exemplo) sem ter tido contato com o ambiente.	O ambiente mantém o mesmo padrão da interface (*layout*) para todos os recursos, de modo a não ser necessária uma nova ambientação.
Ortogonalidade de Conceitos	O software oferece uma linguagem sintaticamente homogênea e comportamento semelhante em situações semelhantes, solicitando do usuário ações similares para tarefas similares. Cada ação, comando ou objeto, recebe um nome que seja consistente durante todo o diálogo entre o usuário e o software.	A linguagem visual usada nas atividades pedagógicas pode ser definida pelo professor (linguagem da sala de aula), o que facilita na obtenção da consistência desejada.

Tolerância a erros de operação do usuário	O sistema, em hipótese alguma, pode se “enrolar” caso o usuário cometa algum erro na operação da interface. Na repetição do erro, a situação deve ser reapresentada da maneira mais adequada possível, ou então uma ajuda pode aparecer para auxiliar o aluno a completar ou reiniciar a tarefa interrompida pelo erro.	As mensagens são sempre de incentivo ao “repensar”, com dicas que levam a descobrir e corrigir o erro, por meio da reflexão e da análise cognitiva.
Organização estrutural	As categorias e regras de classificação devem ser usadas de maneira adequada, para o aluno perceber como estão relacionadas. As sequências de telas devem são organizadas de forma análoga às tarefas semelhantes já conhecidas.	Ao manter o mesmo padrão da interface para todos os recursos, o aprendiz facilmente percebe a relação entre os conceitos e relações apresentados para ele nas telas do ambiente.
Feedback	O *feedback* aqui tratado não é o de conteúdo, mas o reflexo do software perante a ação do usuário. O *feedback* associado à interface orienta o usuário quanto a sua participação na tarefa, e informa o efeito que sua ação teve sobre o sistema, mostrando todas as consequências possíveis da mesma. Exibi, ainda, o novo estado do sistema bem como a nova localização do usuário.	O ambiente trabalha com diversos *feedbacks*, permitindo ao usuário acompanhar cada passo de sua solução e identificar os possíveis erros.
***Layout* da tela**	A tela não pode ser densa nem vazia, e também precisa manter relações lógicas e funcionais entre todos os seus itens. As informações importantes necessitam ficar na tela e com facilidade de acesso, mas sem a tornar demasiadamente congestionada. As cores apresentadas, em geral, devem aparecer em número reduzido, produzindo assim um efeito agradável. Se mal utilizadas, podem até provocar sentimentos agressivos ou de apatia, e pode chamar atenção para algo não muito importante.	O visual do ambiente foi trabalhado junto com a pesquisa etnográfica, de maneira a atender da melhor forma seus usuários. No decorrer da pesquisa, onde o ambiente foi posto em utilização, foram feitas várias correções visuais, indicadas pelos próprios utilizadores, mantendo um ambiente limpo e informativo, com cores alegres e motivadoras.

Ícones	Precisam ser claros e objetivos, sugerindo exatamente a sua funcionalidade, sem complicações. Precisam ser bem visualizados e fáceis de serem selecionados com o movimento do cursor.	Todos os ícones foram desenhados para serem facilmente entendidos.
Memorização	Todas as informações da interface que necessitam ser memorizadas, precisam ser reduzidas, para que o aprendiz desempenhe adequadamente as tarefas atuais e subsequentes, que são objetivos de fato do software.	Foi reduzida ao máximo a necessidade de memorização, foram trabalhadas, juntamente com os ícones, maneiras de ajudar neste quesito.
Histórico	O software disponibiliza para o educador, ou mesmo para o aluno, um histórico de todas as suas ações no decorrer do desenvolvimento da atividade. Armazena também, de forma permanente (em disco), o registro de utilização de cada usuário, juntamente com o seu histórico de acesso. Se for usado em rede, o professor, numa estação, pode verificar o andamento de todos os alunos através dos históricos individuais.	O histórico é um dos principais recursos do ambiente. É com ele que o educador acompanha o desenvolvimento dos seus aprendizes, podendo com isto traçar estratégias para estimular o aprendizado dos seus alunos.
Vocabulário e regionalidade	Não pode apresentar erros gramaticais, mesmo que estes já sejam pertencentes à linguagem cotidiana, pois se o software é educativo precisa usar a linguagem formal. O vocabulário usado no software deve ser adequado ao nível do aluno que irá operá-lo. As gírias, os termos e os costumes regionais apresentados no software devem ser adequados ao local onde a escola se situa. Se possível o software deve usar um vocabulário neutro em relação à regionalidade.	O vocabulário utilizado na concepção do ambiente foi neutro, usando uma linguagem adequada ao seu público alvo, sem uso de gírias ou termos e costumes regionais.

4 ANÁLISE E PROJETO DO AMBIENTE NEWPROG+

O NewProg+ seguir a base da modelagem do NewProg, porém a análise de seu ambiente foi refeita e esta nova análise contou também com uma proposta de arquitetura com o máximo de escalabilidade possível, permitindo aos educadores total liberdade para criarem as atividades para seus alunos com o máximo possível de sua imaginação e diretamente relacionado a seu planejamento para promover a aprendizagem em seus alunos.

Este capítulo apresenta parte da modelagem do NewProg+, apresentando o seu diagrama físico do seu banco de dados e um diagrama de classes, gerado por uma das tecnologias que foram utilizadas em seu projeto, o *Entity Framework*.

4.1 Modelagem

A modelagem do NewProg+ foi projetada visando a escalabilidade do ambiente, permitindo que, com pequenas alterações, o software cresça da forma que for necessário e se adapte aos educadores que o estão utilizando. Como o NewProg+ será um sistema para a web, qualquer atualização do sistema estará disponível para qualquer usuário que desejar o utilizar.

4.1.1 Diagrama de Classes

Na Figura 12, apresentada mais a baixo, temos um diagrama de classes do NewProg+. Este diagrama foi gerado automaticamente pelo Entity Framework, tecnologia utilizada para a manipulação de dados na implementação do sistema juntamente com o software Visual Studio. O diagrama em questão conta com 11 classes. Com uma descrição resumida de cada uma delas, estas classes são:

- usuario – representa um usuário do sistema e suas principais características;
- perfilusuario – determina se o usuário é um aluno ou professor, ou ainda um outro perfil que possa surgir;
- historicoacesso – um log completo da utilização do sistema para cada usuário cadastrado;

- atividade – as atividades criadas e cadastradas para serem realizadas pelos alunos;
- categoriaatividade – a qual categoria a atividade pertence. Por exemplo: Lógica Matemática, Estudo da Língua portuguesa, Estruturas de Repetição, Testes Condicionais e qualquer uma outra que for criada no sistema;
- nivelatividade – delimitar as atividades de acordo com a faixa etária do envolvido e ainda de acordo com as atividades concluídas no NewProg+;
- atividade_aluno – Armazena um histórico de todas as atividades realizadas por cada aluno no NewProg+, incluindo quando houverem mais de uma tentativa para uma mesma atividade;
- tabuleiro – armazena o “cenário” do jogo, onde o atributo “planta” representa a solução de um aluno que se relaciona com este tabuleiro e a classe tabuleiro_imagensTabuleiro representa o tabuleiro na forma em que foi criado por um professor. Nesta classe o atributo usuarioId representa qual usuário criou este tabuleiro, que neste caso será um usuário de perfil Professor;
- imagenstabuleiro – representa o caminho, *link*, para todas as possíveis imagens que possam ser utilizadas para a criação de um tabuleiro;
- tipoImagem – se a imagem cadastrada é um item, parede, porta, armadilha, dentre outros.

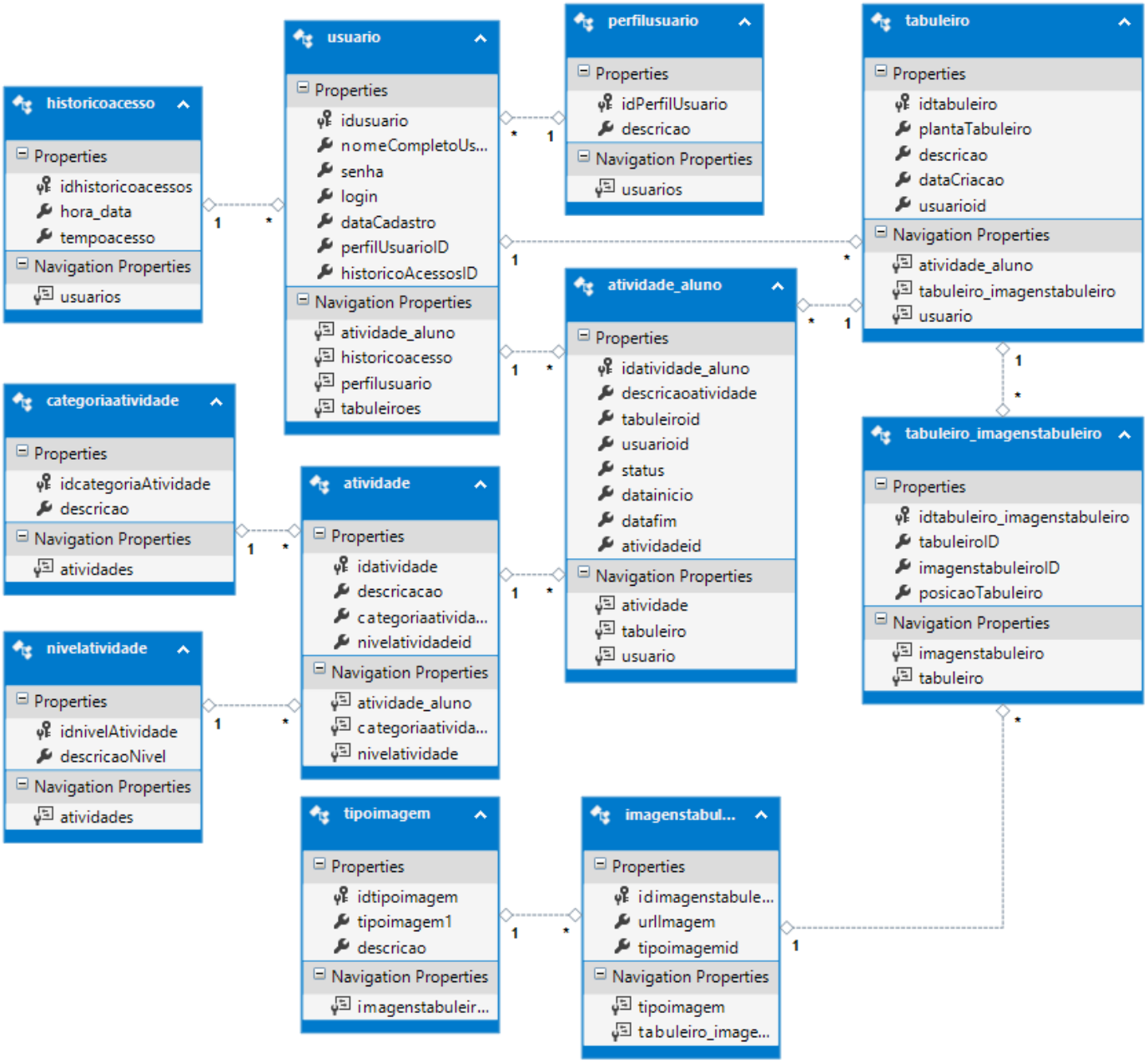

Figura 12 – Diagrama de Classes gerado automaticamente pelo Entity Framework

4.1.2 Diagrama Entidade Relacionamento (DER)

A Figura 13 representa o Banco de Dados do NewProg+, com este modelo de entidades e relacionamentos. Novamente, o NewProg+ foi desenvolvido visando a escalabilidade do software, além de permitir que os usuários (Professores) explorem

toda a sua imaginação e possam compartilhar o máximo de conhecimento possível com seus alunos sem serem limitados pela ferramenta.

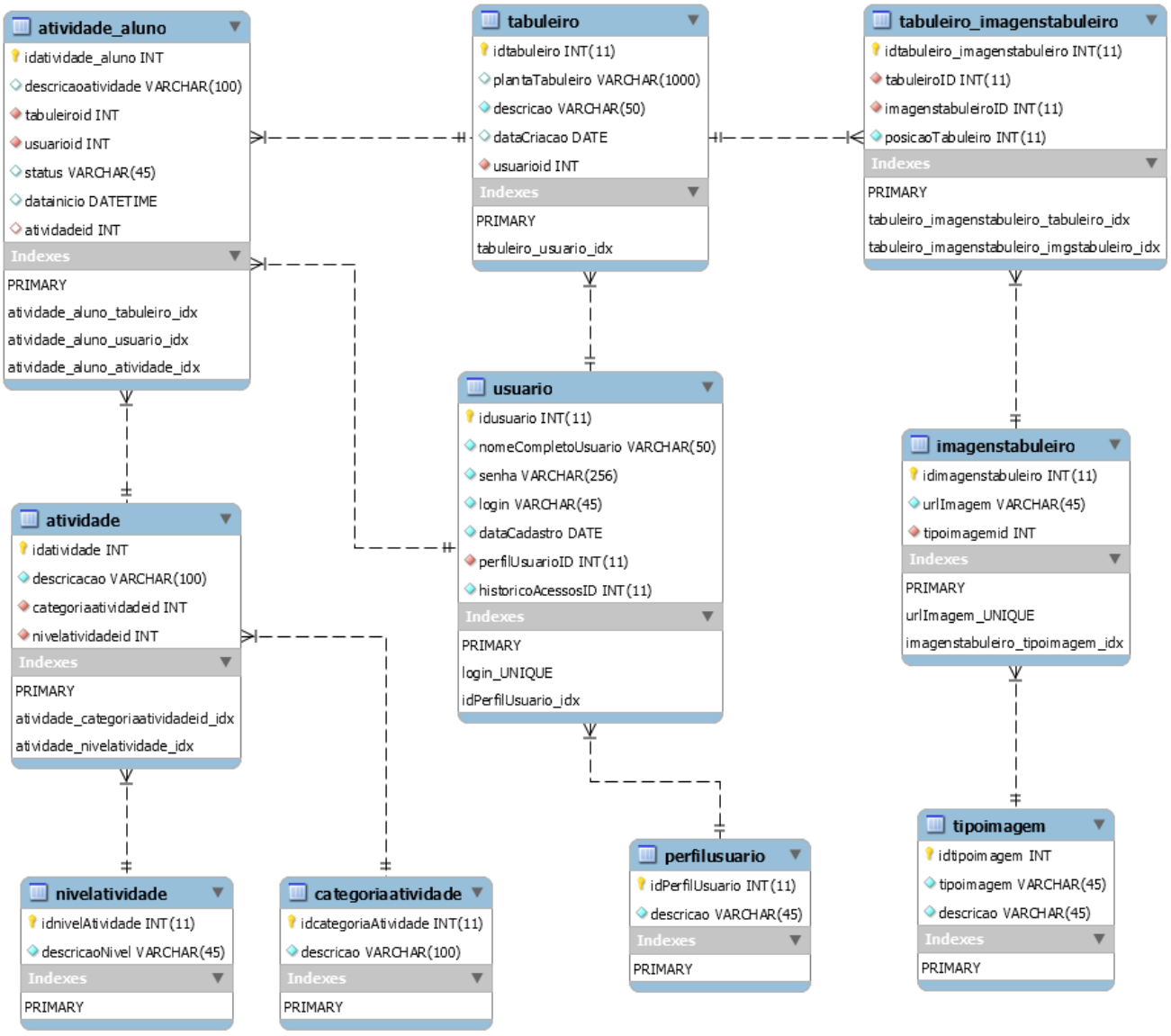

Figura 13 – Diagrama Entidade Relacionamento

5 O NEWPROG E O NEWPROG+

Este capítulo aborda os ambientes do NewProg e do NewProg+ e explicita as suas principais diferenças. Entre estas diferenças, são destacadas as principais funcionalidades que foram adicionadas no NewProg+, que são os testes condicionais e as estruturas de repetição, além de outras atualizações na linguagem visual de programação.

O NewProg é um ambiente que foi desenvolvido com o objetivo de auxiliar as crianças com faixa etária entre cinco e oito anos de idade na aprendizagem dos conceitos de programação de computadores. Este ambiente, segundo Torezani (2013), proporciona uma sensação de entretenimento, enquanto favorece o desenvolvimento de habilidades cognitivas, importantes em diversas situações do cotidiano do aprendiz, tais como o planejamento e a resolução de problemas. A Figura 14 ilustra uma atividade a ser desenvolvida por um aluno no ambiente do NewProg.

Figura 14 – Atividade desenvolvida no NewProg

De forma geral, o NewProg+ se diferencia do NewProg principalmente pela inserção das estruturas de repetição e de testes condicionais na linguagem visual de programação, além de um novo sistema de gerenciamento e edição de atividades.

Assim como o NewProg, o NewProg+ disponibiliza ao aluno diversos testes e desafios com problemas a serem resolvidos por eles. A Figura 15, ilustra a interface de interação com o aluno no NewProg+. Nessa interface, além do cenário representado, existe um avatar que simula a execução das instruções do programa elaborado pelo aprendiz. Esse avatar é escolhido pelo próprio aprendiz, dentre as opções para ele apresentadas.

Figura 15 – Interface de interação com o aluno no NewProg+

5.1 Uma Linguagem Visual de Programação

De acordo com Dahl *et* al. (1972), uma linguagem de programação estruturada implementa e permiti a utilização das seguintes características:

- Sequências de Instruções;
- Testes Condicionais;
- Estruturas de Repetições.

Para Guezzi e Jazayeri (1997), a programação modular engloba o desenvolvimento das rotinas de programação através de módulos, sendo estes interligados através de uma interface comum.

Considerando o público alvo do NewProg, crianças na faixa etária entre 5 e 8 anos, foi necessário pensar em uma Linguagem de Programação que permitisse um maior envolvimento dessas crianças com o ambiente. Para tal, foi desenvolvida uma Linguagem Visual de Programação (LVP), em acordo com os conceitos apresentados por Dahl *et* al. (1972) e Guezzi e Jazayeri (1997), onde com simples cliques em figuras e imagens específicas, essas crianças possam usar esta linguagem, construindo sequências de passos e instruções para a resolução dos problemas propostos.

Na LVP, usada pelas crianças para resolverem os problemas propostos nas atividades, apresentada no NewProg, é possível apenas a definição de sequências de instruções. Já no NewProg+, houve uma atualização e continuidade dessa LVP. Agora a LVP, além de permitir a utilização das Sequências de Instruções, como já ocorria no NewProg, permite também a utilização dos testes condicionais e das Repetições de Instruções (Estruturas de Repetição). A LVP do NewProg+ ainda permite o uso de blocos de comandos, externos ao corpo do programa e chamados de subprogramas, de modo que possam ser referenciados várias vezes no programa.

5.1.1 Um programa considerado correto na LVP do NewProg+

A LVP do Newprog+ permite a organização estruturada da sequência de instruções necessária para que o avatar apresentado no ambiente conclua o objetivo proposto. Assim, é possível que o avatar se locomova até o ponto final desejado, como exemplo,

o último baú do tesouro disponível em uma atividade. Porém, simplesmente chegar a este baú do tesouro não torna a solução do aprendiz correta. É possível que o avatar vá até este baú e em seguida fique batendo a cabeça na parede ou até mesmo caia em uma armadilha presente no cenário. Assim, uma solução será considerada correta nesta LVP de acordo com os seguintes requisitos:

- A última instrução atribuída deverá levar o avatar *exatamente* para o último ponto que determinará o fim da atividade (nenhum passo a mais);
- O avatar não deverá colidir com nenhuma armadilha presente na atividade;
- O avatar deverá capturar todos os itens necessários para cumprir os desafios existentes no ambiente, quando estes estiverem presentes.

É importante destacar que quando a codificação do aprendiz gerar mais instruções que o necessário isto não resultará em uma codificação errada, mas sim em uma codificação que poderá ser melhorada. O mesmo vale para a utilização de itens desnecessários, desde que não falte o item no momento que este será realmente necessário.

5.2 Testes Condicionais

A LVP do NewProg+ foi enriquecida com os testes condicionais, dessa forma é possível que o aprendiz programe o avatar para tomar decisões a partir de determinadas condições. No exemplo de codificação presente na Figura 15, existem alguns obstáculos no cenário que impedem a passagem do avatar. Estes obstáculos são uma porta, que poderá ser aberta, e um bloco de madeira, que poderá ser quebrado, além de algumas armadilhas. Os testes condicionais presentes neste cenário, foram elaborados por um professor responsável por um grupo de alunos. O teste em questão se baseia na comparação de itens e pode ser associado ao “se..então” (“if”) das linguagens de programação tradicionais.

Conforme visualizado na Figura 16, nesta atividade existem três testes, dois referentes a duas portas fechadas, sendo que uma leva à uma armadilha, e outro com um bloco de madeira. Os itens, do menu de comandos do aprendiz (Figura 16), na frente da porta e do bloco, que são, respectivamente, uma chave e um taco, foram

inseridos pelo aprendiz. O educador cria a atividade e diz que haverá uma porta que deverá ser aberta e um bloco que deverá ser quebrado, a utilização dos itens necessários é feita pelo aprendiz.

Podemos analisar os testes da imagem em questão da seguinte forma: “SE HOUVER UMA PORTA FECHADA NA FRENTE DO AVATAR (faz-se necessário direcioná-lo à porta com o comando “*girar*”), ENTÃO, ABRA A PORTA COM O ITEM CHAVE”, e para o segundo teste, “SE HOUVER UM BLOCO DE MADEIRA NA FRENTE DO AVATAR, ENTÃO UTILIZE O TACO PARA QUEBRA-LO”.

Figura 16 - Testes Condicionais e Comando Girar

O comando “Girar”, visualizado também na Figura 16, é indispensável para a utilização dos testes condicionais na LVP do NewProg+. Este comando serve para girar em 90º o avatar, tanto para a direita, quanto para a esquerda. O comando “Girar” se faz necessário, visto que poderão haver mais de um teste para uma mesma condição e em lados opostos a localização atual do avatar. Ou seja, a partir da localização atual do avatar, poderá haver, por exemplo, uma porta à sua direita e ao mesmo tempo uma segunda porta à sua esquerda, assim, sem o comando para girar o avatar, este não saberia qual das duas portas deveria ser aberta. Com o comando girar, o aprendiz deverá direcionar o avatar para a direção exata de onde está o teste condicional ao qual ele irá propor uma resposta. Na “Sequência de Ações” apresentada na Figura 15, podemos visualizar a utilização do comando girar antes do item chave. Neste exemplo, o avatar irá girar 90º à sua direita, sendo direcionado a uma porta antes de utilizar o item chave.

5.3 Subprogramas

Segundo Sebesta (2009), os subprogramas estão entre os conceitos mais importantes de uma Linguagem de Programação, eles são os blocos de construção fundamentais dos programas. O autor ainda afirma que os subprogramas possuem as seguintes características: um único ponto de entrada; a unidade de programa “chamadora” fica suspensa durante a execução do subprograma chamado, implicando a existência de apenas um subprograma em execução em um mesmo momento no tempo; o controle sempre retorna para o chamador quando a execução do subprograma finaliza.

De acordo com Sebesta (2009), existem duas categorias de subprogramas: procedimentos e funções, e ambas podem ser utilizadas como abordagens que visam estender as linguagens de programação. Os procedimentos são considerados coleções de sentenças que definem computações parametrizadas, sendo estas realizadas por sentenças de chamadas únicas. A LVP do Newprog+ apresenta a implementação dos procedimentos, porém, sem passagem de parâmetros, mas permitindo que o aluno utilize esta abstração para construir os seus algoritmos. Os procedimentos na LVP do NewProg+ são intitulados de “Blocos de comandos”, conforme pode ser observado na Figura 17.

Figura 17 – Menu para utilização de blocos de comandos

Nesta versão da LVP do Newprog+, é possível inserir no máximo 12 comandos em um mesmo bloco de comando. Na Figura 17, por exemplo, existe um bloco B1 que executará, quando acionado, 10 comandos. Caso o aluno necessite utilizar mais de 12 comandos, ele poderá utilizar mais de um bloco de comando.

Na construção da atividade, no painel administrativo, o professor responsável poderá disponibilizar quantos blocos de comandos julgar necessário para a realização da atividade pelo aluno, deixando assim disponível outros blocos (B2, B3, B4, etc.), conforme podemos visualizar no exemplo da Figura 18.

Figura 18 – Adicionar Blocos de Comandos: Painel do Professor

Para utilizar mais instruções em blocos de comandos (mais de 12), basta utilizar mais de um bloco, fazendo chamadas internas de acordo com o necessário. Assim, teríamos no bloco B1, por exemplo, uma chamada para o bloco B2, onde como resultado serão executadas todas as instruções contidas em B1 e em seguida em B2, de acordo com a ordem de chamada (Figura 19).

Figura 19 – Utilização de mais de um bloco de comando

5.4 Estruturas de Repetição

A LVP do NewProg+ permite a utilização de estruturas de repetição. Somente será possível utilizar essas estruturas se o professor responsável pela atividade habilitar tal recurso da linguagem, para a atividade em questão, no painel administrativo. Cabe ao professor responsável o total planejamento e construção da atividade, desta forma, o próprio determinará os recursos necessários para a realização da mesma.

Após habilitado pelo professor, o menu de estruturas de repetição estará presente no ambiente de programação do aluno. A Figura 20 ilustra o menu para utilizar esta estrutura no ambiente do aluno. Ao clicar no botão repita, o aluno ativa a estrutura de repetição em seu programa.

Figura 20 - Menu com a estrutura de repetição e menu de Blocos de Comandos

Ainda na Figura 20 podemos visualizar o menu de blocos de comandos, onde, nesta versão da LVP, o aluno será limitado a 12 comandos, conforme já mencionado no tópico 5.3. É possível inserir mais de um bloco de comandos para uma mesma atividade: B1, B2, B3, etc., e quantos outros forem necessários, de acordo com o julgamento do professor responsável pela atividade. O quantitativo de blocos será configurado no momento da criação da atividade no painel do professor.

Após ativar o comando de repetição pelo menu “Repetir”, será necessário que o aluno programe a quantidade de repetições desejadas na LVP. Será apresentado na

sequência de ações do aluno a função “REPITA”, Figura 21, onde este aluno deverá editar o quantitativo de repetições desejadas e qual o bloco de comandos deverá ser repetido. O comando de repetição apresentado na LVP do NewProg+ pode ser comparado a estrutura “para..faça” (*for*) das linguagens de programação tradicionais.

Figura 21 - Comando de repetições na sequência de ações

Após a configuração da estrutura de repetição apresentada acima, Figura 21, podemos fazer a seguinte leitura: “*Repita duas vezes todo o conteúdo que está no bloco B1*”.

Frisando que quando há a necessidade de inserir mais de 12 comandos para serem repetidos, a LVP permite a utilização de mais blocos de comandos, assim poderá ser configurado um segundo bloco, B2, que será invocado dentro do primeiro, B1, assim teríamos no B1 uma chamada para B2, estendendo o quantitativo de comandos de 12 para 23, visto que um dos comandos de B1 será ocupado pela chamada a B2, conforme foi exemplificado pela Figura 19 no tópico 5.3. Para uma nova versão desta LVP, o quantitativo de instruções a serem repetidos será ampliado, permitindo ao aprendiz decidir o número máximo de comandos que queira inserir nesta estrutura.

5.5 Comparativo das funcionalidades do NewProg com o NewProg+

O NewProg+ veio para complementar o ambiente do NewProg, contribuindo principalmente na inclusão de recursos em sua linguagem visual de programação.

O Quadro 2 abaixo, demonstra um comparativo entre as principais funcionalidades dos dois ambientes. Neste comparativo podemos ver que o NewProg+ possui a maioria das funcionalidades existentes no NewProg, com exceção das opções de ajuda para mostrar o menor caminho e o menu que exibe o número de ajudas utilizadas.

Quadro 2 – Comparativo entre as principais funcionalidades do ambiente do NewProg e do NewProg+

RECURSO	NEWPROG	NEWPROG+
Editor de Atividades	✓	✓
Gerenciamento de Imagens para o tabuleiro	X	✓
Gerenciamento de Usuários	✓	✓
Histórico de Acessos	✓	✓
Gerenciamento de Atividades	✓	✓
Histórico das atividades realizadas	✓	✓
Seleção de Avatares	✓	✓
Gerenciamento de Avatares	X	✓
Menu com comandos de ações	✓	✓
Opção para ajuda mostrando o menor caminho	✓	X

No quadro anterior, foram destacadas somente as principais funcionalidades dos dois ambientes. Outras funcionalidades consideradas como básicas, por exemplo *login*, telas principais, entre outras, não foram comparadas neste quadro.

As principais contribuições do NewProg+ estão em seu LVP. Esta LVP traz novos recursos e permite que o professor responsável explore ainda mais o ambiente para que seus alunos aprendam os conceitos básicos da programação de computadores. No Quadro 3 podemos comparar a funcionalidades presentes na LVP dos dois ambientes. Neste comparativo podemos analisar que o NewProg+ não possui os

comandos para andar 3 vezes e andar 4 vezes, visto que isto poderá ser realizado com a combinação de outros comandos encontrados na nova LVP. Por outro lado, o NewProg+ inclui outros comandos fundamentais para a aprendizagem dos conceitos iniciais da programação de computadores.

Quadro 3 – Comparativo entre a Linguagem Visual de Programação do NewProg com a do NewProg+

Comando	**NewProg**	**NewProg+**
Andar 1 vez	✓	✓
Andar 2 vezes	✓	✓
Andar 3 vezes	✓	X
Andar 4 vezes	✓	X
Girar	X	✓
Estruturas de Repetição	X	✓
Estruturas de Decisão (testes condicionais)	X	✓
Subprogramas (blocos de comandos)	X	✓

6 PROTÓTIPO E IMPLEMENTAÇÃO DO NEWPROG+

O NewProg+ mantém a mesma ideologia de implementação do NewProg, um ambiente com funcionalidades básicas e objetivas que possam ser operadas por crianças com a faixa etária entre 5 e 8 anos. O ambiente também visa o gerenciamento e a administração de suas funcionalidades por profissionais e educadores que trabalham com estas crianças.

Para implementar este ambiente, foram utilizadas várias tecnologias. Neste capítulo serão destacadas as principais, dentre estas, e as suas principais características.

Vale destacar que algumas dessas tecnologias são de empresas privadas, porém todas podem ser utilizadas gratuitamente com finalidades educativas e sem fins lucrativos.

6.1 Tecnologias utilizadas no NewProg+

Para o desenvolvimento do ambiente do NewProg+, foram selecionadas algumas tecnologias que, até então, fazem-se presentes no mercado, com frequentes atualizações e referências de uso, para, assim, facilitar o desenvolvimento e a manutenção do software desenvolvido.

6.1.1 O Visual Studio e o .Net Framework

O .Net Framework é um componente integral do *Windows* que suporta a construção e execução de aplicações *desktop* e *web services*. A principal ideia dos *web services* é a possibilidade da comunicação e troca de dados entre as aplicações, de forma simples e transparente, independente do Sistema Operacional (SO) ou da Linguagem de Programação que esteja sendo utilizada. (LIMA, 2002, p. 03)

O *.Net Framework* fornece um ambiente de execução gerenciado, desenvolvimento e implantação simplificados e suporte para uma ampla variedade de linguagens de programação. Possui dois componentes principais, são eles: o *Common Language Runtime* (CLR), que gerencia a memória, a execução de código, e outros serviços do

sistema, e o .*Net Framework Class Library*, que é uma coleção de tipos reutilizáveis que você pode usar para desenvolver seus aplicativos. O .*Net Framework* também inclui tecnologias como ADO.Net, Asp.Net, *Windows Presentation Foundation* (WPF) e *Windows Workflow Foundation* – WF (GROSS, 2008).

6.1.2 Linguagem de Programação C#

A Linguagem de Programação C# (pronuncia-se C *Sharp*) faz parte do conjunto de ferramentas oferecidas pelo *.Net Framework.* O C# surgiu como uma linguagem simples, fortemente tipada, robusta e orientada à objetos. Além de ser altamente escalável, o C# possui a finalidade de permitir que uma mesma aplicação possa ser executada em diversos dispositivos de *hardware.* A linguagem C# também tem como objetivo permitir o desenvolvimento de qualquer tipo de aplicação, sejam elas *Web Services,* aplicação do tipo *Desktop, Mobile, handhelp,* aplicações para a *internet* etc (LIMA, 2002, p. 04).

6.1.3 Asp.NET

ASP.NET é uma plataforma da Microsoft para o desenvolvimento de aplicativos para a *web*. Usando o ASP.NET, pode-se criar lojas de comércio eletrônico, sites, *blogs* e praticamente qualquer outra coisa que se possa encontrar na Internet. Com o ASP.NET pode-se criar aplicativos da *Web* em grande escala, usando nada além de código e uma ferramenta de design, como o *Visual Studio* (MATTHEW, 2010, p. 29).

O custo de toda essa inovação, de acordo com o mesmo autor, é a curva de aprendizado. Para dominar o ASP.NET, precisa-se aprender a usar uma ferramenta de design avançada (*Visual Studio*), um conjunto de ferramentas de objetos (o .NET *Framework*) e uma linguagem de programação orientada a objetos (como C#).

6.1.4 HTML5 e CSS3

De acordo com Castro (2012, p. 26), o HTML5 é uma evolução natural das versões anteriores do HTML e se esforça para refletir as necessidades dos sites atuais e

futuros. Ele herda a grande maioria dos recursos de seus predecessores (HTML). Isso também significa que uma grande parte do HTML5 funciona em navegadores antigos e novos, visto que este foi projetado para ser compatível com versões anteriores.

O autor ainda afirma que o HTML5 adiciona diversos novos recursos. Muitos são diretos, como elementos adicionais (artigo, seção, figura e muitos mais) que são usados para descrever o conteúdo. Outros são complexos e ajudam na criação de aplicativos Web poderosos. É preciso ter uma compreensão sólida sobre criação de páginas para a *Web* antes de se aprofundar nos recursos mais complexos que o HTML5 oferece. O HTML5 também apresenta a reprodução nativa de áudio e vídeo para suas páginas, o que torna desnecessário a utilização de *plug-ins* e complementos para este fim.

O mesmo autor diz que assim como o HTML5 possui relação com versões anteriores do HTML, o CSS3 é uma atualização natural das versões do CSS que o precederam. O CSS3 é mais poderoso do que suas versões anteriores e apresenta inúmeros efeitos visuais, como sombras, sombras de texto, cantos arredondados e gradientes.

6.1.5 Linguagem Javascript

O JavaScript é a linguagem de programação da Web, utilizado pela ampla maioria dos sites e navegadores modernos. O JavaScript é utilizado para programação em diversos tipos de dispositivos, como computadores de mesa, consoles de jogos, *tablets* e smartphones. Todos estes dispositivos incluem interpretadores JavaScript, o que torna a linguagem de programação mais onipresente da história. A linguagem JavaScript faz parte da tríade de tecnologias que todos os desenvolvedores Web deveriam conhecer: HTML, que apresentará o conteúdo das páginas Web; CSS, para formatar a apresentação dessas páginas; e o JavaScript, para programar o comportamento delas (FLANAGAN, 2014, p. 18).

6.1.6 Entity Framework

O *Entity Framework* é um mapeador de objeto-relacional (ORM), que reduz a diferença de impedância entre o mundo orientado a objetos do .Net Framework e o mundo dos

bancos de dados relacionais. Ele permite aos desenvolvedores interagir principalmente com o modelo conceitual de uma aplicação, utilizando técnicas familiares orientadas a objetos. Em *Entity Framework* se pode trabalhar com dados na forma de objetos de domínio específico e propriedades, tais como clientes e endereços de clientes, sem ter que se preocupar com as tabelas do banco de dados subjacentes e colunas onde estão armazenados. Os desenvolvedores podem emitir as operações de acesso aos dados em relação ao modelo conceitual, e traduzir as operações de *Entity Framework* em ações de Banco de Dados relacional (MILLER, 2016).

O mesmo autor ainda diz que o *Entity Framework* oferece suporte a dois cenários: Pode-se inferir um modelo conceitual com base nos tipos de dados e configurações adicionais que se definir. Os metadados de mapeamento são gerados durante o tempo de execução com base em uma combinação de tipos como se foi definido em seu domínio e informações adicionais de configuração que foram fornecidos no código. O *Entity Framework* gera o banco de dados conforme necessário com base nos metadados ou se pode mapear a camada de objeto para um banco de dados existente e o modelo conceitual correspondente ao mapeamento é deduzida.

6.1.7 Sistema Gerenciador de Banco de Dados Mysql

Segundo Date (2003, p. 6), os sistemas de banco de dados são *softwares* que realizam manutenção de registros, ou seja, possuem a função de armazenar informações para que os usuários as atualizem ou recuperem-nas quando necessário. Tais informações podem ser qualquer coisa com algum significado ao indivíduo em questão, ou a uma organização, a que o sistema servirá. Moura (2011, p. 53) completa dizendo que “[...] bancos de dados e sistemas de bancos de dados são componentes essenciais da vida na sociedade moderna”.

Os sistemas de bancos de dados estão disponíveis em diversos tipos de dispositivos que podem variar de tamanho, como computadores de mãos ou até mesmo grandes servidores, sendo que seus recursos são limitados pela potência de seus hardwares. Geralmente os sistemas de pequeno porte tendem a ser monousuário, ou seja, somente um usuário pode acessar o banco de dados em um determinado momento e

por sua vez os de grande porte podem ser multiusuários, onde muitos usuários podem acessar o banco de dados ao mesmo tempo.

Os dados em um banco de dados são referidos como persistentes, com esse termo pode-se dizer que uma vez os dados aceitos pelo Sistema de Gerenciamento de Banco de Dados (SGBD), eles só serão removidos do banco de dados caso exista uma requisição explícita do SGBD, assim se define a persistência. Portanto, define-se banco de dados como uma coleção persistente usadas por uma aplicação qualquer (DATE, 2003).

De acordo com Neves e Ruas (2005), o MySQL é um SGBD relacional, que suporta Linguagem de Consulta Estruturada (SQL), *open source* e também é o SGBD mais utilizado no mundo, com aproximadamente 5 milhões de usuários ativos. O MySQL foi desenvolvido e disponibilizado pela empresa MySQL AB *Limited Company* que atualmente vende um conjunto de softwares relacionados a essa tecnologia. Pode-se encontrar diversos clientes importantes como: Alcatel, AOL, Dow Jones, Nasa, Susuki, Google, entre outros.

Segundo o Manual de referência do MySQL 4.1 (2012), o servidor de banco de dados MySQL é extremamente rápido, confiável e fácil de usar além de contar com um conjunto de recursos práticos enviados por usuários.

O servidor de banco de dados MySQL foi desenvolvido para lidar originalmente com banco de dados muito grandes de maneira rápida, usado em ambientes de alta demanda oferecendo um rico e proveitoso conjunto de funções. A conectividade, velocidade, segurança faz com que o MySQL seja facilmente adaptado para acessar banco de dados na internet.

6.2 Página inicial (*home*) e página de *login*

Assim como no NewProg, o NewProg+ possui uma página inicial (*home*) diferenciada para cada nível de usuário atual, sendo uma para os usuários administradores, outra para usuários com funções pedagógicas e outra para os aprendizes. O que diferencia uma tela da outra são as funcionalidades e recursos disponíveis.

A primeira exibição da tela inicial é genérica para usuários anônimos, ou seja, que ainda não estão “logados” no sistema. Nesta exibição é mostrado algumas informações sobre o NewProg+, a descrição resumida das últimas atividades que foram criadas no ambiente e os campos para o usuário realizar o seu *login* no sistema, sendo necessário informar apenas o seu nome de usuário e a sua senha, conforme podemos visualizar na Figura 22.

Figura 22 - Tela inicial e tela de login

Para o nível administrador, é exibido um controle geral do ambiente, sendo atribuído a este usuário privilégios para cadastrar e gerenciar outros usuários e estruturação do

ambiente em si. O usuário administrador apenas não tem acesso ao histórico de atividades criado pelos aprendizes.

Os usuários com funções pedagógicas – professores e outros educadores, possuem privilégios para editar e gerenciar os recursos que estão relacionados com as atividades que serão realizadas pelos alunos, assim como os tabuleiros, imagens e histórico das atividades desenvolvidas por todos aprendizes que estão sobre a sua responsabilidade.

A Figura 23 mostra a tela inicial dos usuários com o perfil de aprendiz, que são os alunos em si. Tais usuários possuem apenas privilégios para visualizar e realizar as atividades disponíveis para ele. Este nível de usuário pode acessar e editar suas próprias soluções das atividades, visto que um aprendiz poderá realizar atividades criadas por mais de um professor.

Figura 23 - Tela inicial para Aprendizes

Um código criado por um aprendiz pode ser editado a qualquer momento, porém será registrado em seu histórico de atividades tudo o que foi desenvolvido, assim o educador responsável por esta atividade poderá consultar, a qualquer momento, tudo o que o aluno desenvolveu, podendo assim acompanhar seu rendimento e evolução.

A qualquer momento, em sua página principal, o aprendiz poderá alterar o seu avatar e utilizar o personagem que preferir, dentre as opções disponíveis.

Quando o aprendiz clica no botão “Alterar Avatar”, presente na Figura 23, o ambiente o direcionará para a página de seleção de avatares (Figura 24), onde bastará um clique no botão “Escolher” para selecionar um novo avatar.

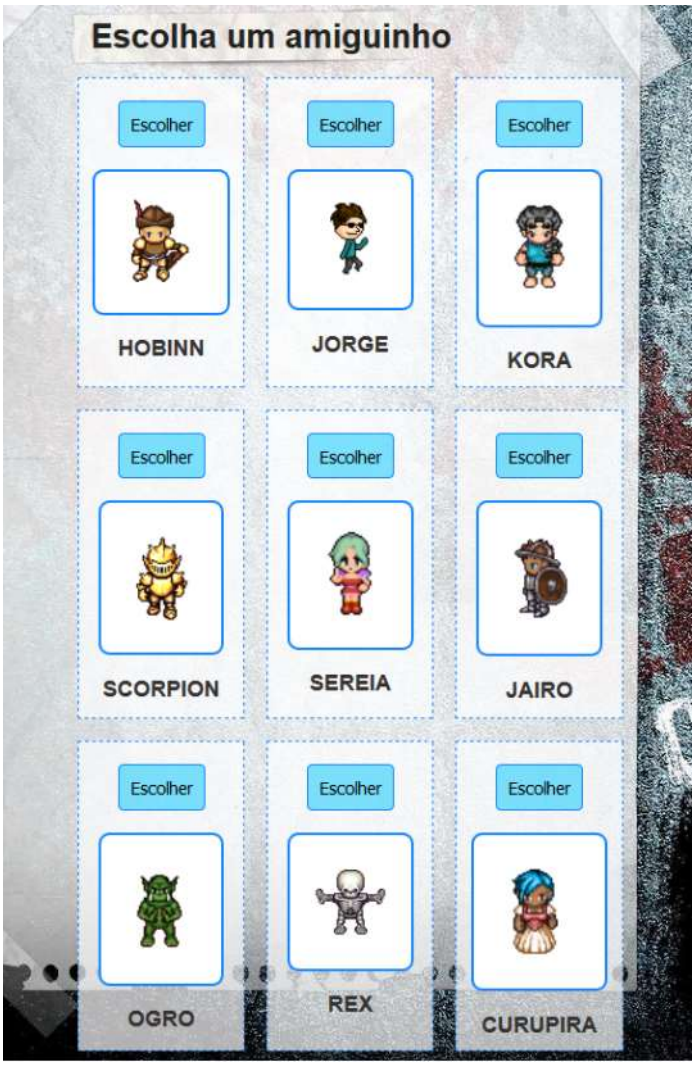

Figura 24 - Seleção de Avatar

6.3 Gerenciamento de usuários

A página de gerenciamento de usuários este disponível para os administradores do software, que podem gerenciar qualquer usuário e para os usuários com o perfil pedagógico, que podem gerenciar apenas os usuários com o perfil de aprendiz. Aqui é possível cadastrar, alterar, listar e excluir usuários. A Figura 25 ilustra a tela de gerenciamento de usuários para um usuário com o perfil de administrador e com o filtro habilitado para exibir apenas usuários com o perfil de professor.

Figura 25 - Usuários Cadastrados

6.3.1 Histórico de acessos

Aos usuários com privilégios de gerenciamento de usuários, também é concedido a permissão para acompanhar todo histórico de acesso destes usuários ao sistema. Aqui é exibido todos os acessos de um determinado usuário ao software e o total de tempo que esteve conectado (logado). Este histórico permite a um educador saber quanto tempo seus aprendizes estão dedicando para a realização das atividades por eles realizadas, conforme mostrado na Figura 26.

Figura 26 - Tela de histórico de acessos de um aprendiz ao sistema

Para visualizar uma solução criada pelo aprendiz, dentre as listadas na Figura 26, basta clicar sobre o ícone lupa, no canto direito da atividade. Após clicar no botão de visualização, o usuário será direcionado para a página de detalhes, para assim analisar a resposta do aluno, segundo aquele histórico da atividade clicado. A página de detalhes pode ser visualizada na Figura 27.

Figura 27 - Visualização da atividade realizada pelo aluno

6.4 Gerenciar perfil de usuário

Disponível apenas para usuários administradores, nesta tela é possível cadastrar e gerenciar os perfis disponíveis. Para o NewProg+, foram necessários apenas três perfis de usuários: administrador, professor e aprendiz, porém, para uma atualização ou nova versão do ambiente, poderão surgir novos tipos de usuários.

6.5 Gerenciamento de atividades

Os usuários administradores e com funções pedagógicas possuem permissões para acessar estas funcionalidades, porém esta tela é destinada aos usuários com funções pedagógicas, que em nosso caso será o usuário Professor. Aqui é possível acessar e gerenciar tudo que estiver relacionado com as atividades que serão realizadas pelos aprendizes. Cada professor terá permissão de acesso apenas às atividades desenvolvidas por ele.

6.5.1 Gerenciar nível da atividade

Permite cadastrar e gerencias os possíveis níveis de atividades. Para a versão do NewProg+ que foi testada, foram criados os níveis Iniciante, Intermediário e Avançado, de forma a separar as atividades de acordo com estes níveis. Os aprendizes subirão de nível de acordo com as atividades concluídas ou de acordo com seu real nível de conhecimento na linguagem visual de programação do NewProg+. O professor pode direcionar os aprendizes para níveis diferenciados, se assim julgar necessário. A ideia em separar os alunos em níveis implica principalmente nos relatórios de aprendizado e estatísticas que podem ser gerados pelos educadores.

6.5.2 Gerenciar categoria da atividade

Com a opção de cadastrar e gerenciar categorias para as atividades existentes, o NewProg+ fica bastante flexível quanto à sua finalidade. Assim, o ambiente poderá ser utilizado tanto para aprendizagem de estruturas de repetição quanto para estruturas de decisão. Desta forma futuramente será possível também cadastrar categorias para aprender programação com conceitos de matemática, português e qualquer outra área desenvolvida pelo professor responsável.

6.6 Gerenciar tabuleiro

O tabuleiro é onde o aprendiz passará a maior parte do tempo desenvolvendo as suas atividades. Este é o espaço onde ele programa, codifica, a sua solução para conseguir resolver as atividades idealizadas pelo professor.

O aluno interage diretamente com o tabuleiro, por meio de um avatar, por ele escolhido. O tabuleiro será editado por um professor, de acordo com as opções para ele disponíveis, como podemos visualizar na Figura 28.

Os professores não são limitados à um tabuleiro estático e cingido, visto que o tabuleiro é editável, assim como as imagens que o compõe e as suas respectivas

finalidades. Assim, em um tabuleiro teremos imagens que serão apenas obstáculos, como paredes e outras que poderão ser armadilhas, portas, itens, entre outros.

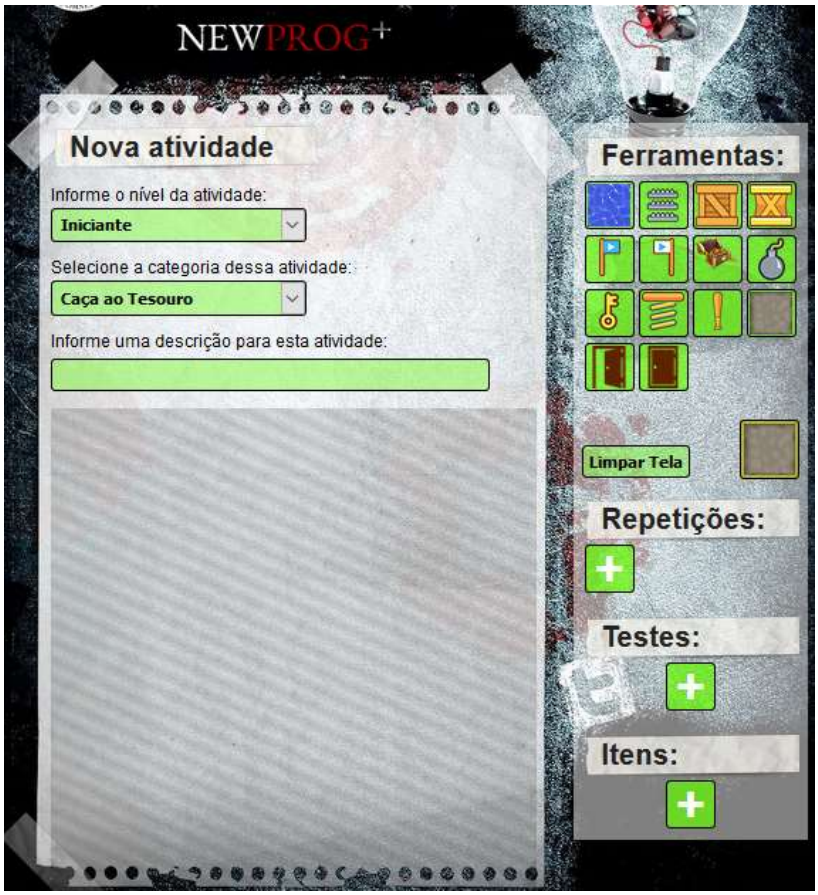

Figura 28 - Cadastrar uma atividade

Mais abaixo, no tópico 6.10, será abordado um pouco mais sobre a criação de uma atividade com a descrição das funcionalidades presentes em um tabuleiro.

6.6.1 Gerenciar imagens do tabuleiro

Conforme mencionado anteriormente, é possível gerenciar as imagens que compõe o tabuleiro. Assim o professor terá um ambiente dinâmico, que poderá ser moldado à sua realidade, para assim desenvolver as atividades que julgar necessário para conseguir desenvolver o aprendizado em seus alunos.

Para gerenciar estas imagens, o professor deverá fazer o upload das imagens que forem necessárias para a sua finalidade. É possível também editar as imagens já existentes, com a seguinte limitação: nenhum usuário professor poderá editar uma imagem criada por um outro usuário, mas poderá visualizar e utilizar estas imagens.

A programação atual do ambiente foi desenvolvida para tratar as imagens de acordo com os tipos delas, fazendo assim uma distinção entre um simples obstáculo, como uma parede e uma porta ou armadilha.

6.6.2 Gerenciar tipos de imagem

Aqui é possível cadastrar e editar os tipos de imagens existentes no NewProg+. Este recurso está disponível apenas para usuários administradores, visto que o ambiente atual foi programado de acordo com os tipos de imagens pré cadastrados, portanto qualquer alteração indevida poderá acarretar em problemas para a atual versão do NewProg+.

Este gerenciamento foi criado com o intuito de permitir a escalabilidade do NewProg+, permitindo que com poucas alterações, o ambiente possa permitir a utilização de diversos outros tipos de imagens.

Nesta versão do NewProg+, os seguintes tipos de imagens foram considerados:

Tipo de Imagem	Descrição
Água	Imagens que representam a água em si. Podem ser utilizadas como obstáculos, onde os avateres utilizados pelos aprendizes não serão capes de passar andando. Um avatar não se afoga ao ter contato com a água, ele simplesmente não a atravessará caminhando. É possível pular sobre pequenos trechos de água, utilizando o item “saltar”. Para uma futura versão do NewProg+, poderá ser incluído a opção de utilização de veículos aquáticos.

Armadilha	São armadilhas em si e podem definir o fracasso de um aluno em uma atividade. Qualquer imagem poderá ser uma armadilha se um professor a definir assim, por isso é importante treina-lo antes da utilização do recurso de gerenciamento de imagens. Também é possível saltar sobre pequenas armadilhas.
Bloco	Os blocos são como as paredes, ou seja, obstáculos que não poderão ser atravessados pelos avatares. A diferença entre um bloco e uma parede no NewProg+ é que os blocos podem ser explodidos e as paredes não. Não é possível saltar sobre os blocos, tampouco sobre as paredes.
Parede	São obstáculos que impedem a passagem de um avatar. Não poderá ser explodido.
Bomba	Itens que podem explodir blocos e outros itens.
Item	Representam item localizados no tabuleiro e servem para compor o inventário de um avatar. Na versão atual do NewProg+ estes itens não geram nenhuma pontuação, mas a ideia é que esta funcionalidade seja implementada em uma versão posterior.
Porta	São portas em geral, ou seja, passagens que podem liberar ou bloquear o acesso de um avatar a determinado espaço do tabuleiro. Existe dois tipos de porta, aberta e fechada. É possível abrir ou fechar uma porta com o item chave.

FimGame	Determina o fim do jogo. Este tipo de imagem também se apresenta em duas possibilidades: Game Over ou Sucesso, ou seja, pode-se utilizar uma imagem do tipo FimGame para determinar o ponto final, objetivo, de um determinado tabuleiro, indicando que quando o avatar alcançar este ponto, terá concluído seu objetivo, ou poderá ser utilizado para determinar que o avatar percorreu um caminho incorreto e não alcançou seu objetivo final em uma determinada atividade.

Quadro 4 - Tipos de Imagens

6.7 Gerenciar atividades

Espaço criado para o professor visualizar e editar as atividades criadas. Aqui o usuário professor terá permissão de gerenciar apenas as atividades que ele criou. Neste gerenciamento ele pode cadastrar uma nova atividade, que possui apenas os campos Descrição, Categoria e Nível da Atividade, alterar estas informações e ainda excluir atividades, deste que estas não possuam alunos conectados.

6.8 Relatório de alunos cadastrados

Exibe todos alunos cadastrados no ambiente. Este relatório poderá ser ordenado ou filtrado de acordo com as Atividades, o Nível da Atividade, Categoria da Atividade. Este relatório possui a finalidade de exibir os alunos cadastrados no ambiente.

6.9 Atividades realizadas pelos aprendizes

Mostra todos os alunos e as atividades que eles realizaram no ambiente. Nesta tala é possível visualizar o percentual das atividades concluídas pelos alunos, para cada professor. Os relatórios de atividades realizadas são para controle do professor que passou a atividade para os alunos e somente este terá acesso a estes dados.

Aqui o professor poderá visualizar o histórico das soluções que os alunos criaram e analisar o seu real aprendizado, podendo readaptar alguma atividade ou criar outras atividades complementares de acordo com a necessidade de seus alunos.

É muito importante que os professores acompanhem frequentemente o desenvolvimento das atividades, com o propósito de interferir quando necessário e também incentivar os seus alunos a realizarem as atividades.

6.10 Editor de atividades

Espaço onde é possível criar e gerenciar as atividades. Aqui o professor poderá utilizar, além de sua imaginação, de acordo com seus objetivos, todas as imagens disponíveis, incluindo as que ele mesmo inseriu no ambiente, para criar as suas atividades.

No editor de atividades, Figura 29, o professor seleciona o nível, a categoria e informa uma descrição resumida sobre a atividade que será criada. Logo em seguida o professor inicia o processo de construção do tabuleiro.

Figura 29 - Editor de atividades

Para edição do tabuleiro, o professor conta com as ferramentas disponíveis. Estas ferramentas são imagens cadastradas. Para saber qual o tipo de imagem, basta colocar o cursor do mouse sobre esta imagem, assim como ilustrado na Figura 30, onde o cursor estava sobre uma imagem do tipo “Armadilha”.

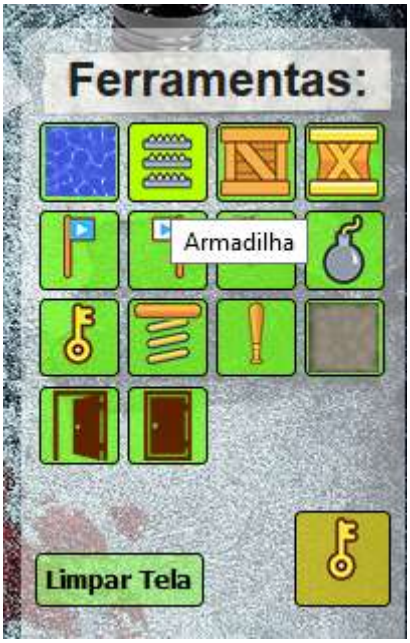

Figura 30 - Visualizando o tipo de imagem

Para incluir uma imagem, basta clicar sobre esta e em seguida clicar no tabuleiro, exatamente no espaço onde se deseja que esta imagem seja inserida. A última imagem clicada é exibida logo abaixo das demais imagens contidas no menu de Ferramentas, conforme podemos visualizar na Figura 30, onde a última imagem clicada foi um item chave, portanto, ao clicar em algum ponto do tabuleiro, um item chave será inserido.

O editor de atividades ainda conta com os menus de Repetições, Testes e Itens.

O menu de Repetições, Figura 31, representa as estruturas de Repetições que serão utilizadas pelos aprendizes. Cabe ao professor disponibilizar, ou não, as estruturas de repetição no ambiente do aprendiz. Nesta versão do NewProg+, poderão ser incluídos no máximo cinco estruturas em uma mesma atividade.

Figura 31 - Menu de Repetições

Após configurado pelo professor, durante a construção de uma atividade, as estruturas de repetições criadas serão apresentadas para os aprendizes, conforme ilustrado na Figura 32.

Figura 32 - Estruturas de Repetição, modo Aprendiz

A Figura 33 ilustra o menu de Testes. Este menu serve para definir os testes condicionais que deverão ser realizados pelos aprendizes. No exemplo da Figura 33, o professor criou o teste porta fechada. Neste caso, o que ele, o professor, quer que o aprendiz utilize para abrir uma porta? Aqui ele definiu uma resposta, que somente será válida caso o aluno utilize esta mesma resposta.

Figura 33 - Menu de Testes

Como resposta, o professor deverá incluir um item, que no exemplo citado, será um item chave. Automaticamente, quando o professor inclui um item neste menu, este

mesmo item já aparecerá para o aprendiz, nesta atividade. O aprendiz deverá encontrar este item para poder passar no teste condicional.

A Figura 35 mostra o menu de testes e o inventário de itens, no modo aluno. Neste exemplo, ao lado da figura da porta fechada, aparecerá para o aluno um botão com um ponto de “?”, assim o próprio aprendiz deverá clicar neste botão e selecionar o item que deseja utilizar quando encontrar esta figura (da porta). Ou seja, “SE A PORTA ESTIVER FECHADA, ENTÃO UTILIZE UMA CHAVE”. Porém para o aluno utilizar um item chave, ele deverá localizar e obter esta chave no ambiente. Na criação da atividade, o professor deverá incluir, pelo menos uma chave para cada porta fechada que ele queira que seja aberta por seus aprendizes.

O menu Itens, ainda na Figura 34, possui zero itens chave. Esta chave com o quantitativo zero é configurada pelo professor, durante a edição da atividade. Durante a execução do programa, cada vez que o avatar passar sobre uma chave, o quantitativo de chaves será incrementado. Da mesma forma, toda vez que o avatar utilizar uma chave, este quantitativo será decrementado.

Figura 34 - Menus de Teste Condicional e Inventário de Itens

O menu de Itens, Figura 35, é uma representação de um inventário. Este inventário é configurado pelo professor, de acordo com a atividade projetada. Neste exemplo, como era necessário o item chave para abrir a porta, o educador inseriu este item e o quantificou com o valor zero, ou seja, ao iniciar a atividade, o aluno não possui nenhum item chave. Mas é possível definir um outro valor, a critério do professor.

Figura 35 - Menu de Itens

Nesta versão do NewProg+, os itens do inventário deverão ser inseridos manualmente pelos professores. Para uma próxima atualização, pretende-se que este inventário seja criado automaticamente, de acordo com o tabuleiro que está sendo criado, mas não tirando a possibilidade da edição manual, pelo professor responsável.

6.11 Realizar atividades

Essa funcionalidade está presente no Módulo Aprendiz. É aqui que o aluno acessa e desenvolve a atividade que seu professor preparou para ele.

Ao selecionar uma atividade para ser realizada, o aluno se depara com o tabuleiro, que foi criado por um professor responsável, contendo um menu de ações, com todas as opções disponíveis para realizar a atividade proposta.

Figura 36 - Atividade desenvolvida

6.11.1 Funcionalidades presentes no Módulo Aprendiz

A interface do ambiente, na versão apresentada para os alunos, possui as seguintes características:

- Andar 1 vez: faz o avatar avançar uma posição (ou um quadro), no sentido da seta indicadora (para cima, para baixo, esquerda ou direita).

Figura 37 - Andar uma vez

- Andar 2 vezes: similar ao “Andar 1 vez”, porém nesta opção o avatar avança duas posições (ou quadros) no sentido das setas.

Figura 38 - Andar duas vezes

- Girar: gira o avatar 90º à sua esquerda ou 90º à sua direita. Este comando é essencial para auxiliar nas estruturas de decisão. Serve para o avatar se direcionar para o objeto ao qual deverá tomar alguma decisão. Quando a posição frontal do avatar não coincidir com o objeto a ser analisado, este deverá ser direcionado ao mesmo. No exemplo da Figura 36, o avatar deverá abrir uma porta que está à esquerda de sua localização frontal, porém poderia haver uma porta, por exemplo, tanto à esquerda, quanto à direita do avatar, ao mesmo tempo e assim, com o comando para gira-lo, ele saberá exatamente qual a porta deverá ser aberta.

Ainda no exemplo das portas, o avatar não consegue abrir uma porta que não esteja em frente a ele.

Figura 39 – Comando Girar

- **Repetir**: ao selecionar o botão de repetição (REPITA) – Figura 40, e inseri-lo na Sequência de Ações (Figura 42), todas as instruções encontradas dentro do bloco (B1), ou outro bloco selecionado, se repetirão a quantidade de vezes configurada (Figura 42).

Figura 40 – Menu para ativar a estrutura de repetição

- Blocos de Comandos: Para repetir as instruções contidas em um mesmo bloco de comandos, basta utilizar o símbolo que ativa este bloco (ex.: B1). É possível incluir vários blocos de comandos (B2, B3, etc), de acordo com a necessidade da atividade proposta pelo professor, conforme já discutido no tópico 5.3, onde trata-se dos subprogramas.

Figura 41 - Menu de blocos de comandos no painel do aluno

Na Figura 42, podemos ver um exemplo de utilização de uma estrutura de repetição e de um bloco de comandos. Neste exemplo, a estrutura de repetição invoca o bloco B1 duas vezes. Assim são executados todos os comandos existentes em B1Figura 41.

Figura 42 - Utilização de uma estrutura de repetição

- **Itens**: representa um inventário contendo todos os itens que o avatar possui. O aluno utiliza esses itens para cumprir seus objetivos finais, como abrir uma porta, se ela estiver fechada, utilizar um taco para quebrar um bloco de madeira, entre outras possibilidades. Se o programador (aprendiz) guardar mais itens do que ele vai precisar na execução (avaliação) do programa, não haverá erro no código, mas sim um excesso de itens em seu inventário. Este excesso de itens representa mais código que o necessário para a realização de uma determinada atividade, assim, ao término da atividade, o professor responsável pode apresentar ao aluno uma sugestão de otimização do código desenvolvido.

Para capturar um item no ambiente, basta que o aluno direcione o seu avatar para caminhar sobre este item, assim, ao passar pelo item, o avatar o captura e ele é imediatamente inserido em seu inventário.

Figura 43 - Itens capturados pelo avatar (inventário)

Para utilizar o item desejado, basta acioná-lo no momento certo, de acordo com a sequência de passos construída, como utilizar o item chave no momento que desejar abrir uma porta.

- Testes: representam os testes condicionais que o avatar realizará. Neste menu aparem os testes preparados pelo professor. Aqui o aluno diz o que seu avatar fará se encontrar certo item ou obstáculo. De acordo com o nível selecionado para a atividade, o teste estará presente na interface do aluno, apenas aguardando a sua iteração, ou estará oculto, com a finalidade do aluno inseri-lo quando julgar necessário. A Figura 44 demonstra um exemplo de um teste que se inicializou com a atividade, aguardando apenas a iteração do aluno. Neste exemplo, o teste foi para verificar se a porta está fechada. Ou seja, se a porta estiver fechada, use uma chave. Porém, para utilizar o item chave, seu avatar deverá captura-lo e tê-lo armazenado em seu inventário de itens.

Figura 44 - Testes condicionais

7 EXEMPLOS DE ATIVIDADES E DE USO DO NEWPROG+

Neste capítulo serão apresentados alguns exemplos de uso das atividades criadas e gerenciadas pelo NewProg+. Estas atividades foram criadas para apresentar algumas funcionalidades existentes no ambiente, porém, o ambiente poderá ser editado e adaptado de acordo com a necessidade específica de cada professor, de acordo com os recursos disponíveis, como gerenciamento de imagens do tabuleiro.

A Figura 45 ilustra uma atividade com o nome “Capture a Bandeira Azul”, onde o objetivo final do aprendiz é chegar à bandeira azul. Para isto, o aprendiz deverá passar por alguns obstáculos existentes no cenário, como armadilhas, água e um bloco de madeira bem resistente. O cenário possui uma porta, porém esta porta leva à uma armadilha, portanto o caminho escolhido foi saltar, pular sobre a armadilha que se encontra mais abaixo, no tabuleiro.

Figura 45 – Atividade Capture a Bandeira Azul

É importante observar que, para esta atividade, o professor não habilitou o menu de repetições, considerando que este não é necessário para a realização da mesma.

A atividade em questão foi desenvolvida para possibilitar que o aprendiz conduza o seu avatar a utilizar os itens disponíveis no tabuleiro, porém, ao ser criada a atividade, o professor já disponibilizou alguns recursos para auxiliarem o aprendiz, como a possibilidade de utilizar apenas três saltos, conforme visualizado na Figura 46.

Figura 46 – Itens da atividade Capture a Bandeira

Ainda sobre esta atividade, é importante destacar a utilidade da funcionalidade “Girar”, lembrando que é preciso girar o avatar direciona-lo exatamente de frente para o objeto onde será utilizado o item em questão. Como exemplo, antes de saltar sobre os espinhos, o avatar deverá estar direcionado para estes.

Também podemos observar um bloco de madeira no tabuleiro desta atividade. É necessário remover este bloco antes de passar por ele e para isto o avatar deverá utilizar o item “Taco”, que deverá ser adquirido durante a execução da atividade.

Na Figura 47, temos a Sequência de Ações da atividade Capture a Bandeira Azul, analisando esta sequência de ações, podemos definir a programação do aprendiz. Destacando alguns pontos dessa programação temos: o primeiro comando “girar à esquerda” – direciona o avatar para uma armadilha de espinhos presente no cenário e logo após este comando, existe o comando “*jump*”, que fará o avatar saltar sobre estes espinhos, passa a ocupar o próximo quadro existente após o item armadilha de espinhos; o segundo comando “*jump*” será utilizado para o avatar realizar um salto sobre uma parte menor da água encontrada no cenário. Este salto move o avatar

imediatamente para o próximo bloco após a água; em seguida temos dois comandos para girar à esquerda, um após o outro – o objetivo destes comandos é direcionar o avatar para realizar um novo salto, onde este saíra do canto do cenário, onde foi apenas para adquirir um item do tipo “taco”; em seguida o avatar se locomove em direção ao outro taco existente no cenário para após ir em direção ao bloco de madeira; o próximo comando em destaque são os dois “tacos”, um após o outro – estes tacos estão sendo utilizados para remover o bloco de madeira existente no cenário, porém com apenas um golpe de taco, não será possível remover o bloco em questão, havendo assim a necessidade de utilizar dois golpes, representado na sequência de ações por dois tacos, ou seja, cada taco adquirido representa um golpe que poderá ser utilizado pelo avatar.

Figura 47 – Sequência de Ações da Atividade Capture a Bandeira Azul

A Figura 48 ilustra a atividade “Capture todo o tesouro”. O objetivo desta atividade é capturar todo o ouro disponível no ambiente. Para tal, o aprendiz deverá passar por todos os obstáculos encontrados, como diversas armadilhas de espinhos, que deverão ser evitadas, com exceção de uma que será saltada. A água também representa um obstáculo para o avatar, visto que este deverá capturar os itens “chave” e “taco”, que se encontram cercados por água. O avatar é capaz de saltar sobre um bloco de água, desde que o professor habilite a opção salto. Para esta atividade, também foram liberados, pelo professor, três saltos para serem utilizados pelo avatar. Desta forma, o aluno deverá decidir sabiamente onde irá utilizar os saltos disponíveis.

Esta é uma atividade de nível avançado para o ambiente, onde os alunos que já estão utilizando o ambiente por mais tempo são mais indicados para realizarem a mesma.

É necessário observar a importância do comando girar, visto que este direciona o avatar à um obstáculo, que no NewProg+ representa um teste a ser verificado. Por exemplo: se encontrar uma armadilha, utilize um salto para passar sobre ela; se encontrar uma porta fechada, utilize o item chave para abri-la; e se encontrar um bloco de madeira, utilize um item taco para remove-lo. Com a atual programação do ambiente, não é necessário utilizar um teste condicional para saltar sobre um bloco de água, mas este teste poderá ser criado pelo professor caso este julgue necessário.

Figura 48 – Atividade Capture Todo o Tesouro

Se considerarmos cada comando existente na sequência de ações desta atividade (Figura 49) como uma função simples, podemos dizer que a programação apresentada para esta atividade é representada por uma linha de código.

Figura 49 – Sequência de Ações da atividade Capture Todo o Tesouro

Transcrevendo esta sequência de ações para um algoritmo com chamadas de função, teremos o seguinte resultado:

1. AndarUmBlocoParaBaixo()
2. AndarDoisBlocosParaDireita()
3. AndarDoisBlocosParaBaixo()
4. AndarDoisBlocosParaBaixo()
5. AndarDoisBlocosParaDireita()
6. AndarDoisBlocosParaDireita()
7. GirarParaDireita()
8. SaltarUmBloco()
9. GirarParaEsquerda()
10. SaltarUmBloco()
11. GirarParaEsquerda()
12. SaltarUmBloco()
13. AndarDoisBlocosParaEsquerda()
14. AndarUmBlocoParaEsquerda()
15. GirarParaDireita()

16. UtilizarTaco()
17. AndarDoisBlocosParaCima()
18. AndarDoisBlocosParaDireita()
19. GirarParaEsquerda()
20. UtilizarChaveParaAbrirPorta()
21. AndarDoisBlocosParaCima()
22. AndarUmBlocoParaDireita()
23. AndarDoisBlocosParaEsquerda()
24. AndarUmBlocoParaEsquerda()

Assim, temos o total de vinte e quatro chamadas de funções, onde cada uma é responsável pela ação definida pelo significado do nome dela.

Mais abaixo, na Figura 50, temos mais uma atividade, com o título “Capture a Bandeira Branca”. Esta atividade foi criada com o intuito de treinar o aprendiz para reaproveitar códigos, utilizando os blocos de comandos, representados aqui pelo bloco B1.

Figura 50 – Atividade Capture a Bandeira Branca

Para esta atividade o professor disponibilizou apenas um bloco de comandos, B1, onde é possível reutilizar alguns comandos, conforme o aprendiz julgar necessário.

Analisando a atividade em questão, é possível observar que o labirinto criado no ambiente foi construído para seguir um certo padrão. Este padrão poderá ser abstraído pelo aprendiz, onde o mesmo deverá capaz se analisar o padrão e propor um reuso de algum código já criado. Esta atividade precisa de um conhecimento básico de programação na LVP do NewProg+.

Figura 51 – Sequência de ações da atividade Capture a Bandeira Branca

A sequência de ações do aprendiz para esta atividade, ficou bastante reduzida, com apenas cinco comandos (Figura 51), visto que o aprendiz utilizou um bloco de comandos B1, representado pela Figura 52.

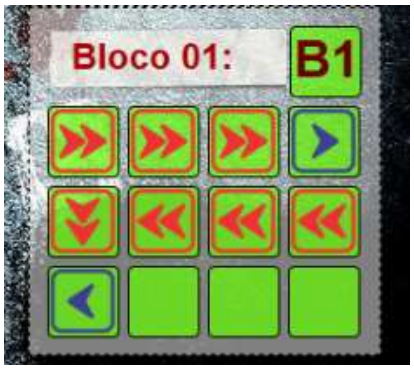

Figura 52 – Bloco de comandos da atividade Capture a Bandeira Branca

Na Figura 52, podemos observar também que o aprendiz utilizou alguns comandos combinados para locomover o avatar pelo labirinto. Desta forma, todas as instruções existentes no bloco, se repetirão todas as vezes que o bloco for acionado. Destacando que uma atividade será finalizada quando chegar ao seu objetivo final, que neste exemplo será encontrar a Bandeira Branca, ou quando a codificação do aprendiz apresentar algum erro, como exemplo, o avatar colidir contra uma armadilha de espinhos. Desta forma, a solução dada pelo aprendiz, será considerada correta desde que leve o avatar para cumprir seu objetivo final, desviando das armadilhas e superando os obstáculos encontrados.

8 TESTES E AVALIAÇÃO DOS RESULTADOS

Como a principal diferença do NewProg+ em relação ao NewProg é a inclusão das estruturas de repetição e dos testes de condição, os experimentos foram realizados com intuito de testar, principalmente, essas duas funcionalidades da ferramenta.

O presenta capítulo apresenta os testes de aceitação da nova versão do ambiente, o NewProg+, e quais os resultados foram obtidos após a realização dos testes em questão.

O NewProg+ foi testado durante o primeiro semestre de 2017, com trinta e três alunos, com a faixa etária de sete e oito anos de idade, e dois professores, em uma escola de ensino fundamental. Com o Editor de Atividades, os professores criaram uma atividade com a proposta de analisar a capacidade de seus alunos para a resolução de problemas que demandavam soluções com repetição de blocos de comandos.

O desafio de cada estudante, em uma das atividades usada nos experimentos, era percorrer todo o labirinto, abrindo as portas existentes e ainda capturando todo o tesouro disponível sem cair nas armadilhas (Figura 53). Quando o estudante desenvolvia uma solução, ele era novamente desafiado a tentar construir uma nova solução menor (um programa com menos comandos).

Para esta atividade, cada aluno precisa cumprir os seguintes desafios:

- Capturar todos os tesouros (condição que indica o fim do desafio, concretizando o sucesso da atividade em questão);
- Abrir todas as portas existentes:
    - Para cada porta é necessário um item chave;
- Não colidir com nenhuma armadilha (condição que define o término do desafio, acarretando insucesso);
- Executar o mínimo de passos possíveis, reaproveitando soluções desenvolvidas.

Na área administrativa, o professor cria e edita a atividade (Figura 53). Com o Editor de Atividades o professor edita sua atividade exatamente como desejar. No painel de edição existe o menu “Testes”, onde o professor adiciona os testes que devem ser

realizados, com as respectivas condições de aceitação. Na atividade da Figura 54, o teste configurado prevê verificar se a porta está fechada. O professor pode configurar um item que determine a condição do teste como verdadeira, quando estiver em posse do avatar. Assim, no exemplo citado, se a porta está fechada, mas o avatar possui uma chave da porta, então a condição do teste torna-se satisfeita (verdadeira).

Figura 53 – Painel para edição de atividade

A Figura 54 apresenta a mesma atividade da Figura 53 sendo desenvolvida pelo aluno. Aqui o aluno pode editar seu algoritmo. O aprendiz define uma sequência de ações a serem realizadas pelo avatar para solucionar a atividade proposta. Pode-se observar que existe o menu “Testes”, assim como no modo de edição, ilustrado na Figura 53. Nesse menu, o aluno deve adicionar os testes necessários para a realização da atividade. No exemplo da Figura 54, o aluno precisa adicionar um teste para porta fechada, definindo que “se encontrar uma porta fechada, precisa utilizar

uma chave para abri-la”. Quando o aluno define que precisa utilizar uma chave para abrir a porta, automaticamente foi inserido um item “chave” = 0 em seu inventário. Assim, somente é possível abrir uma porta se o avatar possuir o item chave, que precisa ser capturada no cenário.

Na Figura 54 é possível observar o menu “Bloco 01”, onde existe a possibilidade de incluir uma quantidade limitada de comandos. Esse menu é representado pelo símbolo **B1**. Quando esse símbolo é inserido na “Sequência de Ações”, juntamente com o comando “REPITA”, todas as instruções existentes no menu “Bloco 01” são executadas a quantidade de vezes programadas no comando REPITA. No exemplo da Figura 54, a estrutura criada no menu Repetir é executada duas vezes.

Figura 54 – Estrutura de Repetição e bloco de comando (Menu do aluno)

Cada atividade é criada e gerenciada pelo professor responsável.

No experimento do ambiente, antes da realização da atividade, os professores explicaram e demonstraram aos alunos as funcionalidades do ambiente, assim como os objetivos da atividade. Durante o experimento, foi permitido a cada aluno realizar até cinco tentativas de programação de uma solução.

Após a aplicação da atividade, obteve-se os resultados representados no Quadro 5:

- 1ª amostragem: 100% dos alunos envolvidos realizaram a atividade, porém apenas 67% deles conseguiram concluir a atividade com sucesso nesta primeira tentativa;
- 2ª amostragem (Aqui não foram incluídos os alunos que finalizaram a atividade na primeira amostragem, procedimento que ocorre também nas demais amostragens): Após uma segunda explicação sobre o objetivo da atividade, 82% obtiveram sucesso ao completar a atividade;
- 3ª amostragem: podemos constatar que os alunos que chegaram a esta amostragem ainda não estavam entendendo o objetivo da atividade. Após o esclarecimento das dúvidas, 50% deles conseguiram concluir a atividade;

  4ª e 5ª amostragem: 3% destes alunos chegaram a esta amostragem, mostrando-se desmotivados e ainda sem entender o que era para ser realizado.

Quadro 5 - resultados após a realização da atividade

	1ª Amostragem	**2ª Amostragem**	**3ª Amostragem**	**4ª Amostragem**	**5ª Amostragem**
Alunos	100%	33%	6%	3%	3%
Sucesso	67%	82%	50%	0%	0%
Erros	33%	18%	50%	100%	100%

Os alunos que não obtiveram êxito nas duas primeiras amostragens avaliaram o ambiente do NewProg+ como fácil de ser utilizado, porém alegaram que faltou um pouco de compreensão sobre o real objetivo da atividade. Logo que entenderam o objetivo, em sua maioria, facilmente concluíram a atividade proposta.

De uma forma geral, os alunos e professores envolvidos acharam o ambiente divertido e fácil de ser utilizado a ainda afirmaram que voltariam a utilizar o NewProg+.

9 AVALIAÇÃO DE APRENDIZAGEM

Após a conclusão dos testes e de analisar os resultados obtidos, este capítulo apresenta uma avaliação da aprendizagem dos alunos que utilizaram o NewProg+.

Vale ressaltar que a participação das crianças nas avaliações realizadas nesta pesquisa foi aprovada pelos seus respectivos responsáveis e os nomes e as imagens destas crianças foram preservados.

Como mencionado no capítulo anterior, o ambiente teve uma boa aceitação pelos alunos que o utilizaram, assim como pelos educadores que o testaram. Podemos avaliar a aprendizagem dos estudantes de acordo com suas propostas de soluções para as atividades criadas para eles. Como vimos nos testes de avaliação do ambiente, foram obtidos bons resultados, onde apenas 3% dos 33 alunos envolvidos não conseguiram realizar a sua atividade, mas ainda assim conseguiram propor uma solução, mesmo que não satisfatória.

Da mesma forma que em sua versão anterior, o NewProg+ busca auxiliar as crianças no processo de aprendizagem de programação de computadores. Assim como no estudo elaborado por Torezani (2014), nesta pesquisa também observamos indicadores que comprovam o desenvolvimento da lógica de programação – primeiro passo para a aprendizagem de programação de computadores – nestes alunos. Esses indicadores são: 1. Correção de erros – após a identificação de problemas e, em alguns casos, de acordo com o processo de tentativa e erro; 2. Busca por soluções alternativas para a resolução de um mesmo problema, onde os estudantes, não satisfeitos com uma solução atual, refaziam a atividade, propondo uma nova solução; 3. Autoria na criação das soluções visto não criavam suas próprias soluções e 4. Reflexões sobre as soluções propostas.

Pôde-se observar também uma atenção especial para solucionar as repetições propostas, como a reutilização de um bloco de comandos para resolver problemas diferentes e ainda a associação de objetos às soluções de outros problemas, como nos casos onde os passos deveriam levar o avatar a capturar alguns itens existentes no cenário e abrir algumas portas.

Com a análise das amostragens, vimos um aumento significativo dos alunos que conseguiram construir uma solução para o problema em questão. Em alguns casos, durante uma mesma amostragem, esses alunos reconstruíam a solução anterior, pois julgavam ter encontrado uma melhor solução.

Com esta análise, podemos afirmar que o NewProg+ é capaz de auxiliar as crianças no desenvolvimento de habilidades de programação e de resolução de problemas. O NewProg+ possibilita, ainda, o planejamento de atividades que envolvam o reaproveitamento de soluções para mais de um problema.

10 CONSIDERAÇÕES FINAIS E TRABALHOS FUTUROS

Este trabalho apresentou o NewProg+, uma atualização e continuação do *software* NewProg. O NewProg+ é um *software* educacional para auxiliar as crianças, na faixa etária de cinco a oito anos de idade, durante o processo de aprendizagem dos conceitos básicos da programação de computadores.

O NewProg+ possui um ambiente fácil de ser utilizado, intuitivo e limpo, que proporciona ao aluno, de uma forma bem dinâmica e divertida, recursos necessários para promover a aprendizagem dos conceitos iniciais da programação de computadores.

O ambiente desenvolvido possui uma Linguagem Visual de Programação, com o objetivo de facilitar o processo de aprendizagem dos alunos, desta forma, a interação destes aprendizes com o ambiente é por meio de cliques, o que simplifica ainda mais a sua utilização.

Os testes de usabilidade do ambiente foram realizados em uma escola municipal, onde os professores utilizaram a ferramenta para criar as atividades e gerenciá-las e seus alunos realizaram as atividades criadas, com a finalidade de resolver os desafios propostos, de forma descontraída e participativa, com total acompanhamento do professor responsável.

A análise dos resultados obtidos nos testes realizados com o ambiente mostrou que o NewProg+ permite aos envolvidos o desenvolvimento de habilidades de programação e de resolução de problemas. O *software* também possibilita a criação de soluções utilizando estruturas de repetição e testes condicionais.

O NewProg+ permite ainda um acompanhamento pelo professor responsável, que, além de criar e propor as atividades desenvolvidas pelos alunos, é capaz de observar o histórico da sessão de cada estudante, para analisar o percurso de aprendizagem desses alunos.

Continuaremos usando e avaliando o ambiente NewProg+ em escolas de ensino básico, de modo a fazermos um estudo mais detalhado sobre os benefícios do

desenvolvimento das habilidades de programação e os impactos delas no cotidiano escolar dessas crianças.

Como trabalho futuro, é desejável que o ambiente possua um avatar de ajuda, para apresentar ao aprendiz uma sugestão de melhor caminho, ou apenas de caminho sugerido para cumprir os objetivos da atividade. Essa ajuda é fundamental para os alunos que não conseguem completar a atividade. Considerando ainda que ao visualizar uma solução, o aprendiz pode se inspirar para construir outra solução.

REFERÊNCIAS BIBLIOGRÁFICAS

BAYTAK, A.; LAND, S. M. ***An investigation of the artifacts and process of constructing computers games about environmental science in a fifth grade classroom****. Educational Technology Research and Development*, 59, 765-782, 2011.

CASTRO, E; HYSLOP, B. ***HTML5 and CSS3***. *Seventh Edition*. Berkeley: Peachpit Press, 2012. 550 p.

CASTRO, T.H.C., JÚNIOR, A.N.D.C., MENEZES, C.S. **Representando Padrões em um Ambiente de Apoio à Aprendizagem de Programação**. *Brazilian Symposium on Computers in Education* (Simpósio Brasileiro de Informática na Educação-SBIE). 2002. p. 558-561.

CODECOMBAT. **Code Combat**, 2016. Disponível em: <http://br.codecombat.com/about>. Acesso em: 12 ago. 2016.

DAHL, O.; DIJKSTRA, E. W.; HOARE, C. A. R. ***Structured programming****. Academic Press Ltd*., 1972.

DATE, J.C. **Introdução a sistemas de Banco de Dados**. Tradução de Daniel Vieira Rio de janeiro: Editora Campos, 2003.

FESSAKIS, G.; GOULI, E.; MAVROUDI, E. ***Problem solving by 5–6 years old kindergarten children in a computer programming environment****: A case study. Computers & Education,* v. 63, p. 87-97, 2013.

FESSAKIS, G; LAPPAS, D. ***Cultivating Preschoolers Creativity Using Guided Interaction With Problem Solving Computer Games. In****: European Conference on Games Based Learning. Academic Conferences International Limited,* 2013. p. 763.

FLANAGAN, D. **JavaScript**: O Guia Definitivo, 6. ed. Bookman, 2014. 1017 p.

GERALDES, W. B. ***PROGRAMMING IS GOOD FOR CHILDREN? A CRITICAL VIEW ABOUT TEACHING PROGRAMMING IN SCHOOLS***. Texto Livre: Linguagem e Tecnologia, v. 7, n. 2, 2014. ISSN 1983-3652.

GROSS, C. ***Beginning* C# 2008**: *From Novice to Professional. Second Edtion*. *New York:* Apress, 2008. 550 p.

GUEZZI, C.; JAZAYERI, M. ***Programming Language Concepts***. 3ª ed. *New York: John Wiley & Sons*, 1997. p. 7. 427p.

KIND, A. ***Computing attitudes****: Will teaching 2nd grade students computer science improve their self-efficacy and attitude and eliminate gender gaps. Rising Tide*, v. 8, 2015.

LATAILLE, Y. **Ensaio sobre a utilização de computadores na educação**. São Paulo: Iglu, 1990.

LAW, K.M.; LEE, V.C.; YU, Y.T. ***Learning motivation in e-learning facilitated computer programming courses***. *Computers & Education*, v. 55, n. 1, p. 218-228, 2010.

LIMA, E. **C# e .Net - Guia do Desenvolvedor**. Rio de Janeiro: Campus, 2002. 358 p.

MATTHEW, M. ***Beginning* ASP.NET 4 in C# 2010**. *New York*: Apress, 2010. 1016 p.

MICROSOFT RESEARCH. **Code Hunt**, 10 jun. 2016. Disponivel em: <https://www.codehunt.com/about.aspx>. Acesso em: 09 mai. 2016.

MILLER, R. ***ENTITY Framework***. Disponível em: <http://msdn.microsoft.com/en-us/library/gg696172(v=VS.103).aspx>. Acesso em: 12 abr. 2016.

MOURA, E. **Banco de Dados**. IFES, 2011, p. 56.

NEVES, P.M.C.; RUAS, R.P.F. **O guia Prático do MySQL**. Lisboa: Coleção Tecnologias, 2005.

OLIVEIRA, M.L.S *et* al. Ensino de lógica de programação no ensino fundamental utilizando o Scratch: um relato de experiência. In: **XXXIV Congresso da SBC-XXII Workshop de Ensino de Computação, Brasília**. 2014.

PAPERT, S. ***Looking at Technology Through School-Colored Spectacles***. MIT Media Lab, 1996.

RIBEIRO, R.D.S.; BRANDÃO, L.; BRANDÃO, A. 2012. **Uma visão do cenário Nacional do Ensino de Algoritmos e Programação**: uma proposta baseada no

Paradigma de Programação Visual. *In: Brazilian Symposium on Computers in Education* (Simpósio Brasileiro de Informática na Educação-SBIE). 2012.

SEBESTA, R. W. **Conceitos de linguagens de programação**. Bookman Editora, 2009.

SILVA, P.V.B.; MORO, M.L.F. ***The interaction of street teenagers with Logo language****. Psychology: Research and Review*. Porto Alegre, vol.11 n.1, 1998.

TAVARES, O. L.; MENEZES, C.S.; NEVADO, R.A.: ***Pedagogical architectures to support the process of teaching and learning of computer programming****: In FIE2012-Frontiers in education conference,* 2012.

TOREZANI, C. **NEWPROG - Um ambiente online para crianças aprenderem programação de computadores**. 2014. 110 f. Dissertação (Mestrado em Informática) – Programa de Pós-Graduação em Informática, Universidade Federal do Espírito Santo. Vitória, Espírito Santo. 2014.

TOREZANI, C.; CHAGAS, L. B. C.; TAVARES, O. D. L. **NewProg - um ambiente para crianças aprenderem programação de computadores**. Anais do Congresso Brasileiro de Informática na Educação (Wie/Cbie), Campinas - SP, 23 Nov. 2013. 140-149. Disponível em: <http://www.br-ie.org/pub/index.php/wie/article/view/2637>.

WANGENHEIM, C. G. V.; NUNES, V. R.; SANTOS, G. D. **Ensino de Computação com SCRATCH no Ensino Fundamental – Um Estudo de Caso. Revista Brasileira de Informática na Educação**, 22(03), p.115. Disponível em: <http://www.br-ie.org/pub/index.php/rbie/article/view/2885>.