Universidade de Brasília
Instituto de Ciências Exatas
Departamento de Ciência da Computação

CODEX - Objeto de Aprendizagem para Ensino de Laços de Repetição

Breno D. Teixeira
Bruno A. Gonçalves

Monografia apresentada como requisito parcial
para conclusão do Curso de Computação — Licenciatura

Orientadora
Prof.a Dr.a Letícia Lopes Leite

Coorientadora
Prof.a Dr.a Márcia Cristina Moraes

Brasília
2017

Universidade de Brasília
Instituto de Ciências Exatas
Departamento de Ciência da Computação

CODEX - Objeto de Aprendizagem para Ensino de Laços de Repetição

Breno D. Teixeira
Bruno A. Gonçalves

Monografia apresentada como requisito parcial
para conclusão do Curso de Computação — Licenciatura

Prof.a Dr.a Letícia Lopes Leite (Orientadora)
CIC/UnB

Prof. Dr. Wilson Henrique Veneziano
CIC/UnB

Prof. ~~M.~~ Cauê Zaghetto
CIC/UnB

Prof. ~~M.~~ Pedro Antônio Dourado Rezende
Coordenador do Curso de Computação — Licenciatura

Brasília, 20 de ~~junho~~ de 2017

Dedicatória

Breno Domingues Teixeira
Bruno Amorim Gonçalves

Agradecimentos

Breno Domingues Teixeira
Bruno Amorim Gonçalves

Resumo

Este trabalho apresenta as etapas de construção do CODEX, um objeto de aprendizagem que permite ao usuário aprender um importante conceito de linguagem de programação, a saber, o *loop*. O objeto de aprendizagem foi projetado e desenvolvido de maneira a proporcionar um processo de aprendizagem de estruturas de repetição de forma lúdica, interativa e abstrata, sem que os estudantes necessitem de conhecimentos prévios de programação.

Palavras-chave: objetos de aprendizagem, estruturas de repetição, ensino de programação

Abstract

This paper presents the steps taken to build CODEX, a learning object that allows the users to learn an important programming concept, loop. The learning object was projected and developed in a way to allow the learning process of loops in a easy, fun, interactive and abstract way, without the need for previous programming knowledge by the students.

Keywords: learning objects, loop, programming

Sumário

Lista de Figuras

Lista de Tabelas

Capítulo 1

Introdução

Com o avanço das tecnologias nas mais diversas áreas do conhecimento, a informática se revela a cada dia como uma peça fundamental para os processos educativos de aprendizagem. Dessa forma, há uma maior preocupação no que se refere aos benefícios de se ensinar programação desde a educação infantil para proporcionar o desenvolvimento de competências e habilidades da área. A consequência desta proposta é uma maior integração entre a computação e as diversas áreas do conhecimento [1, 2, 3]. Apesar dessa proposta, no Brasil, o ensino de programação ainda está em fase embrionária, ficando esta temática mais centrada nos cursos superiores e técnicos [4].

Os cursos superiores e técnicos, no entanto, enfrentam dificuldades com o baixo rendimento dos alunos em disciplinas ~~relacionadas à lógica e aos algoritmos da programação~~. ~~Acreditamos~~ que, caso os alunos tivessem o primeiro contato com a linguagem de programação nos anos iniciais da Educação Básica, essa dificuldade de abstração e interpretação poderia ser atenuada [5, 6, 7].

Considerando a importância ~~deste~~ tema, este trabalho pretende contribuir com a área de pesquisa relacionada ao ensino de programação. Para isso, foi desenvolvido um Objeto de Aprendizagem (OA) chamado CODEX que permite ao usuário aprender um ~~conceito importante~~ para a linguagem de programação – laço de repetição.

1.1 Problema

Os cursos na área de tecnologia enfrentam grandes desafios relacionados ao ensino de programação. Um desses desafios é a capacidade dos estudantes em assimilar conceitos fundamentais que servem como alicerce da programação. Diante disso, torna-se necessário o desenvolvimento de propostas que apoiem a compreensão dos conceitos relacionados a esta temática, dentre eles, o de laço de repetição.

1.2 Justificativa

O estudo realizado pelos autores deste trabalho aponta que existe uma grande dificuldade dos novos alunos dos cursos da área de Computação na Universidade de Brasília, pois a disciplina que apresenta os conceitos básicos da programação, variável, condicional, vetor, laço de repetição, entre outros, tem uma taxa de reprovação perto dos 50%. Diante deste cenário, é necessário ~~se~~ buscar alternativas que possam contribuir para a melhoria dos processos de ensino e de aprendizagem de programação. Neste sentido, ~~esse~~ trabalho se apresenta como uma destas ~~alternativas.~~

1.3 Objetivo

O objetivo geral desta monografia é desenvolver um objeto de aprendizagem lúdico e interativo ~~para auxiliar professores, estudantes e pessoas interessadas em aprender o conceito de laço de repetição,~~ sem a necessidade de conhecimentos prévios sobre o assunto.

1.4 Objetivos Específicos

Para atingir o objetivo geral, foram definidos os seguintes objetivos específicos:

- Desenvolver ~~o~~ objeto de aprendizagem como um *software* que possa ser evoluído e adaptado.
- Criar um ambiente gráfico que contribua para motivar o estudante no entendimento do conceito de laço de repetição.
- Garantir ao professor um nível de personalização da ferramenta a fim de adequá-la às suas necessidades de uso.
- Validar, testar e ajustar as atividades desenvolvidas do *software*.
- ~~Disponibilizar o *software* de forma gratuita para *download*.~~

1.5 Organização do Trabalho

O presente trabalho está organizado da seguinte forma:

O Capítulo 2 apresenta o ensino de programação, suas fragilidades e alternativas possíveis.

No Capítulo 3 é apresentado o conceito ~~e~~ aplicação ~~de~~ Objetos de Aprendizagem ~~(OAs). Ainda neste capítulo, é apresentado o~~ conceito de repositórios digitais e a sua funcionalidade.

O Capítulo 4 detalha o processo de desenvolvimento do objeto de aprendizagem CODEX. Aborda-se a estrutura do sistema e as tecnologias utilizadas para sua confecção.

Para finalizar, ~~as considerações finais, as percepções~~ do trabalho desenvolvido e as sugestões para continuidade deste projeto ~~estão contidas no Capítulo 5~~.

Capítulo 2

Ensino de Programação

~~Na atualidade muito se fala sobre as potencialidades das tecnologias para os avanços sociais.~~ Percebe-se que há um crescente interesse do público em geral pelas tecnologias, principalmente por aquelas que facilitam o dia a dia de quem as consomem. Como consequência, a cada dia aparecem mais pessoas interessadas pela temática e pela possibilidade de desenvolver novos recursos tecnológicos. Portanto, faz-se necessário pensar em uma formação na área computacional capaz de atender a essas demandas.

Nos cursos que se destinam à formação de profissionais da área de computação, cada vez mais se debate sobre as potencialidades, os avanços e as dificuldades relacionadas à temática. Uma grande dificuldade encontra-se no primeiro contato do estudante com a linguagem computacional ~~que~~ geralmente é feito em disciplinas ~~que estão presentes nos~~ cursos de computação. Essa iniciação se apresenta como um desafio tanto para o professor quanto para o estudante [8, 9, 10]. De acordo com Souza et al. [10], os estudantes têm dificuldades em entender determinados conceitos de programação, tais como ponteiros, recursão, declaração de variáveis, dentre outros e que alguns estudantes, apesar de entenderem os conceitos de programação, têm dificuldades em aplicá-los durante a construção de programas. É fundamental que os professores apresentem essa disciplina utilizando-se de uma prática pedagógica que facilite ao estudante a compreensão dos conceitos e a construção do conhecimento, pois geralmente os estudantes apresentam dificuldades para assimilarem tais abstrações.

De acordo com Júnior [8] e Souza et al. [10], o ensino e a aprendizagem de programação são consideradas tarefas complexas e, como consequência, os cursos de programação frequentemente têm altas taxas de reprovação e desistência. Na tentativa de demonstrar as dificuldades da aprendizagem de programação, foram levantados dados referentes à quantidade de estudantes inscritos, aprovados e reprovados na disciplina de Algoritmos e Programação de Computadores na Universidade de Brasília (UnB). Essa disciplina apresenta os princípios fundamentais da programação, sendo o primeiro e principal contato do

estudante com a temática. Esses dados são referentes aos semestres de 2015/2, 2016/1, 2016/2 e 2017/0 (curso de verão) (Tabela 2.1 e Tabela 2.2). Nesse período foram matriculados 933 estudantes, dos quais somente 515 concluíram com êxito. Os outros 418, representando 45%, por algum motivo, foram reprovados, abandonaram ou trancaram. ~~Esse~~ percentual serve como importante ponto de reflexão sobre a área, sendo que pode indicar uma necessidade de se repensar as metodologias e as práticas pedagógicas utilizadas para o ensino da programação.

Tabela 2.1: Aprovados e Reprovados

	2015/2	2016/1	2016/2	2017/0	TOTAL
SR	46	57	38	0	141
II	40	54	22	0	116
MI	33	29	12	11	85
MM	61	46	49	5	161
MS	65	66	74	10	215
SS	48	47	33	11	139
TR	9	8	9	19	45
TJ	0	0	31	0	31
TOTAL	302	307	268	56	933

Fonte: ~~Teixeira e Gonçalves, 2017~~

Tabela 2.2: Legenda das Menções

Menção	Descrição	Equivalência Numérica	Situação
SR	Sem Rendimento	0 (zero) ou acima de 25% de faltas	Reprovado ou Abandono
II	Inferior	0,1 a 1,9	Reprovado
MI	Médio Inferior	2,0 a 4,9	Reprovado
MM	Médio	5,0 a 6,9	Aprovado
MS	Médio Superior	7,0 a 8,9	Aprovado
SS	Superior	9,0 a 10,00	Aprovado
TR	Trancamento Parcial de Matrícula	-	Trancado
TJ	Trancamento Justificado	-	Trancado

Fonte: ~~Teixeira e Gonçalves, 2017~~

O debate sobre o ensino da programação é amplo e envolve pesquisas que englobam desde a Educação Básica até a Educação Superior [11, 12]. Júnior [8] aponta que uma das metas dos cursos de computação está definida em torno da capacidade do estudante apresentar soluções para diversas classes de problemas encontradas no cotidiano das pessoas, das organizações e de muitos outros elementos. Dessa forma, em caráter introdutório, o autor salienta a necessidade de se fornecer aos estudantes as bases necessárias para o desenvolvimento da lógica de programação e representar o raciocínio envolvido por meio de

algoritmos nexos e corretos. Atualmente, ainda são poucas as instituições que se preocupam em trabalhar as noções de programação na Educação Básica e adotar em seu projeto pedagógico essa temática. Porém, estudos afirmam que o raciocínio lógico essencial à programação apresenta reflexos positivos no desenvolvimento de estudantes [9].

A partir ~~desse pensamento~~, nos anos sessenta, com a ideia de apoiar o desenvolvimento do raciocínio lógico, foi criada a linguagem LOGO[1] [13]. Ela foi desenvolvida por Seymour Papert, um educador matemático do MIT - Massachusetts Institute of Technology, Estados Unidos, e adaptada para o português em 1982, na Unicamp, pelo Núcleo de Informática Aplicada à Educação (NIED). A linguagem LOGO foi desenvolvida para ser usada por crianças e traz embutida uma filosofia de educação não diretiva, baseada na educação piagetiana, onde a criança aprende explorando o seu ambiente. Dessa forma, percebe-se que é uma linguagem de fácil aprendizagem e que permite pessoas alfabetizadas e de qualquer idade a programarem a partir do primeiro contato com a linguagem. Independentemente de ser acessível às crianças, LOGO não é uma linguagem focada apenas no público infantil. Ela permite que pessoas aprendam explorando, investigando e descobrindo por si mesmas, isso a torna uma ferramenta muito eficaz no processo de aprendizagem.

LOGO é uma linguagem interpretada, ou seja, cada linha é lida pelo interpretador que a executa. Esse processo é de execução lenta, mas tem a vantagem de não exigir a compilação completa para cada mudança feita em seu código e também oferece algo diferente de outras linguagens: a tartaruga gráfica. A linguagem é composta de um conjunto de simples comandos para manipular a tartaruga que, ao se movimentar pela tela, de acordo com os comandos, deixa um rastro, dessa forma, o rastro deixado pela tartaruga dá um retorno imediato (*feedback*) ao usuário. Esse retorno é o que torna essa linguagem divertida e mais fácil de aprender.

2.1 Dificuldades no ensino de programação

De acordo com Souza [14], historicamente o ensino de programação tem sido considerado de difícil entendimento para os estudantes pelos seguintes motivos: falta de preparo dos estudantes, ausência de uma didática adequada e de ferramentas computacionais que ajudem os atores (professores e estudantes) a superarem os problemas que se apresentam no processo de ensino e aprendizagem.

~~Souza~~ [14] afirma que, geralmente, os cursos de programação iniciam-se com um pseudocódigo, uma linguagem simples e natural que não exige conhecimento de nenhuma sintaxe de linguagem de programação. Um ponto positivo dessa abordagem tradicional

[1] http://projetologo.webs.com/texto1.html

é a possibilidade de se usar apenas lápis e papel para se codificar um algoritmo, sem a necessidade de computadores.

O Portugol, nome usual dessa forma de pseudocódigo, permite que a pessoa foque inicialmente na solução de um problema proposto, mesmo sem conhecimento prévio de uma linguagem de programação. Porém, ao mesmo tempo apresenta conceitos básicos como declaração de variáveis, linhas de código, palavras-chave, comentários, dentre outros, que posteriormente serão transpostos com mais facilidade para ambientes reais de desenvolvimento.

Um aspecto negativo é que, para a maioria dos estudantes, esta abordagem é muito abstrata, pois não conseguem associar os comandos que escrevem com as execuções e respostas do programa. Ainda, existe o fato de que o estudante enfrenta o obstáculo de entender o problema e criar uma solução ou compreender uma solução apresentada por um colega ou pelo professor, isto é, a lógica de programação em si. E, muitas vezes, o percurso formativo o impede de interpretar os problemas matemáticos e compreender as abstrações.

Considerando as pesquisas sobre ensino de programação relacionadas à Educação Superior, alguns trabalhos, como o de Júnior et al. [5], questionam as metodologias e as práticas pedagógicas adotadas para familiarizar os futuros programadores. Os autores realizaram uma pesquisa na Faculdade de Educação Tecnológica do Estado do Rio de Janeiro - Campus Paracambi e mapearam, por meio de dados coletados, os pontos críticos que geraram os maiores problemas de aprendizagem para que medidas corretivas pudessem ser direcionadas com o objetivo de melhorar o aprendizado da disciplina Algoritmos e Programação de Computadores. Por meio de um questionário aplicado ao final do semestre, eles levantaram dados sobre as dificuldades para entendimento dos problemas e identificação do conteúdo em que apresentaram as maiores dificuldades. No quesito das dificuldades para entendimento dos problemas, foram elencados quatro fatores determinantes e seus respectivos percentuais:

- raciocínio lógico (31%)
- capacidade de abstração (28%)
- leitura e interpretação de textos (24%)
- conhecimentos matemáticos (17%)

No que diz respeito aos conteúdos em que apresentaram maiores dificuldades, foram elencados nove tópicos:

- matrizes (34%)
- vetores (23%)

- estruturas de repetição (13%)
- estruturas de decisão/condicional (8%)
- atribuição, operandos/operadores e expressões (7%)
- estrutura sequencial (6%)
- conceitos de algoritmos e sua utilidade (4%)
- variáveis, constantes e tipos de dados (4%)
- entrada, processamento e saída (1%)

Nesta mesma perspectiva, o trabalho de Souza et al. [10] fez um mapeamento sistêmico envolvendo 519 artigos e teve o objetivo de identificar os principais problemas e dificuldades no ensino e na aprendizagem de programação, assim como as possíveis soluções que vêm sendo propostas a fim de minimizá-los. Na primeira parte os artigos foram classificados considerando os problemas e dificuldades que eles abordam e foram divididos em sete categorias:

- Aprendizagem de Conceitos de Programação (39%): Dificuldades relacionadas à aprendizagem de conceitos de programação, tais como recursão, ponteiros, estruturas de repetição, classes e objetos.
- Aplicação de Conceitos de Programação (24%): Dificuldades em utilizar os conceitos de programação aprendidos durante a construção de programas.
- Motivação (22%): Dificuldades relativas à falta de interesse e/ou desânimo dos estudantes em realizar a atividade de programação.
- Compreensão de Programas (11%): Dificuldades em ler e entender programas.
- Fatoração e Refatoração de Programas (2%): Dificuldades em dividir o programa em módulos, funções, classes, etc.
- Dificuldades Relacionadas aos Professores (2%): Dificuldades dos professores em ensinar os conceitos de programação, desenvolver materiais e exercícios de apoio, bem como avaliar os trabalhos de programação.
- Outros (4%).

Na segunda etapa, os artigos foram classificados considerando as soluções propostas. Estas foram divididas em treze categorias:

- Visualização (21%): Utilização de animações para demonstrar as sequências de ações dos programas e algoritmos para superar as dificuldades encontradas.

- *Serious Games*[2] (15%): Utilização e desenvolvimento de jogos para o ensino e a aprendizagem de programação.
- Ambientes de Desenvolvimento Pedagógico (13%): Utilização e desenvolvimento de ambientes que proveem funcionalidades para construção e execução de programas, mas visando o ensino e a aprendizagem de programação.
- Colaboração (11%): Estratégias de ensino e de aprendizagem em que os estudantes possam aprender uns com os outros.
- *Scaffolding*[3] (6%): Estratégias de ensino em que o professor vai adaptando a tarefa de acordo com o nível de habilidade dos estudantes, realimentando-a por contínuo *feedback* durante a progressão da tarefa.
- Notações (5%): Consiste em ensinar os conceitos de programação por meio de notações e linguagens mais familiares aos aprendizes, tais como as linguagens naturais e as notações musicais.
- Reflexão (5%): Estratégias de ensino que envolvem a aprendizagem por meio da reflexão de experiências anteriores.
- Feedback (3%): Envolve a utilização de feedback mais significativo aos estudantes com respeito à qualidade dos seus programas.
- Instrução Ancorada (2%): Estratégia de ensino em que os estudantes devem resolver problemas considerando um material instrucional previamente fornecido.
- Interatividade (2%): Estratégias que promovem uma maior interação entre os estudantes e dos estudantes com o professor.
- Problemas Reais (2%): Consiste em prover atividades em que os estudantes irão construir programas que podem ser utilizados para alguma finalidade ao contrário dos *toy programs* que após as atividades costumam ser descartados.
- Representações Semânticas (2%): Estratégias que visam fornecer aos estudantes representações dos códigos dos programas em uma linguagem mais natural.
- Outros (13%): Soluções menos citadas. Incluem: aprendizagem baseada em problemas, estimular os estudantes a aprender desenvolvendo programas ou modificando

[2] *Serious games* são jogos eletrônicos que têm como principal objetivo treinar pessoas – vendedores de loja, operários, médicos – por meio de um ambiente virtual que imita a realidade e faz com que os jogadores pratiquem atividades para aprender. [15]

[3] *Scaffolding* é um termo em inglês da engenharia civil e que no âmbito de aprendizagem é um termo que designa o estágio inicial para um indivíduo que está sendo submetido a uma experiência pela primeira vez. Desta forma, é dado a ele todo o suporte para que o mesmo possa atingir as metas de aprendizagem.

programas existentes, aprendizagem por meio de um ambiente de rede, aprendizagem por tentativa e erro, avaliação em pares ou semanais, e-learning, sessões adicionais de ajuda e fornecimento de uma melhor visibilidade do progresso de avaliação.

O trabalho de Souza et al. [10] identificou alguns problemas no ensino e na aprendizagem, assim como as melhores soluções para amenizá-los. O trabalho também correlaciona as melhores soluções para cada tipo de problema citado, dessa forma, contribuindo para o aperfeiçoamento da docência na área.

2.2 Alternativas educacionais

A partir das pesquisas apresentadas, percebe-se que a abstração inicial da programação, no que se refere a conceitos de programação e raciocínio lógico, é um determinante no desenvolvimento do aprendizado e o raciocínio requisitado muitas vezes foge à realidade dos estudantes.

Como apresentado por Souza et al. [10], o trabalho enumera algumas alternativas para solucionar os problemas oriundos do ensino de programação. Dentre eles, destacaremos os *Serious Games* que utilizam jogos eletrônicos com o principal objetivo de treinar pessoas por meio de um ambiente virtual que imita a realidade. Além de treinar pessoas para diversos fins, eles também podem ser utilizados para fins educacionais. Os jogos eletrônicos, *Serious Games*, quando utilizados para a educação, fazem parte dos objetos de aprendizagem ~~(OA)~~. Objetos de Aprendizagem são considerados recursos digitais (jogos, vídeos, animações, etc) que têm finalidade educativa e podem ser utilizados para complementar e aperfeiçoar os estudos.

Capítulo 3

Objetos de Aprendizagem

As mudanças e o avanço tecnológico mundial nas áreas de informação e de comunicação estão refletindo na área educacional. O aumento do uso da internet para fins educativos tem se tornado mais frequente nos últimos anos e vem criando uma demanda por recursos digitais educativos, também conhecidos como Objetos de Aprendizagem ~~(OA)~~.

3.1 Conceito

De acordo com Moraes et al. [16], não existe uma definição exata sobre o que é OA. De uma forma simplificada, Objeto de Aprendizagem é um recurso digital que tem a intencionalidade educativa e pode ser utilizado para complementar e aperfeiçoar os estudos, a fim de atingir a aprendizagem desejada, pois amplia as possibilidades de acesso ao conhecimento.

De acordo com Sabbatini [17], os OAs se distinguem dos demais recursos didáticos por meio das seguintes características:

1. reutilização: com a possibilidade de uso em diferentes contextos educativos, proporcionando eficiência econômica em sua preparação e desenvolvimento;
2. simultaneidade: com a possibilidade de ser utilizado por inúmeras pessoas ao mesmo tempo;
3. portabilidade: com disponibilidade de utilização por meio de diferentes plataformas técnicas;
4. granularidade: da ideia de grão, algo tão pequeno e básico, como por exemplo, uma foto, mas que possa conter ou estar contido em outros objetos, com a perspectiva de combiná-los;
5. autossuficiência: no sentido de não depender de outros objetos para fazer sentido;

6. descritos por metadados: são informações que descrevem outros recursos de informação; o termo significa “dados sobre dados”, ou seja, informação que qualifica outra informação.

~~As características descritas são o que distinguem um recurso digital de um objeto de aprendizagem.~~ Além disso, a intencionalidade educacional dada ao material é outro fator determinante para os OAs, ou seja, eles são recursos digitais com fins educacionais e podem assumir qualquer formato ou mídia, desde imagens e arquivos de texto, até animações e simulações.

Segundo Wiley [18], objeto de aprendizagem ~~(OA)~~ é “qualquer recurso digital que possa ser reutilizado para o suporte ao ensino”. Para Koper [19], citado por Sabbatini [17], um objeto de aprendizagem é definido como qualquer recurso digital, reprodutível e referenciável, utilizado em atividades de aprendizagem ou de apoio à aprendizagem, disponível para que outras pessoas o utilizem. Assim, percebe-se que o recurso precisa ser digital, reutilizável e com fins educacionais, pois sem a intenção pedagógica o objeto de aprendizagem torna-se apenas um recurso digital qualquer.

Para melhor compreensão sobre o assunto, algumas metáforas foram abordadas por pesquisadores do tema. Com base na perspectiva da granularidade dos objetos de aprendizagem, surgiu a ideia de que poderiam ser agrupados de qualquer forma. Segundo Tavares [20], a primeira metáfora utilizada para descrever OA, comparava-o com as peças de lego, a partir das seguintes premissas:

1. as peças de lego podem se combinar com qualquer outra peça;
2. as peças de lego podem ser montadas de qualquer forma;
3. as peças de lego são tão simples e fáceis de manusear que até uma criança é capaz de uní-las.

Porém, Wiley [18] apontou algumas falhas na proposta de Tavares e buscou uma analogia mais apropriada e apresentou a metáfora do átomo. Ele entende que esta metáfora capta melhor as definições e as características, pois:

1. nem todo átomo pode se combinar com outro átomo;
2. os átomos apenas podem se agrupar em determinadas estruturas se sua estrutura interna permitir;
3. é preciso treinamento para ordenar e agrupar os átomos.

Essas metáforas auxiliam no entendimento de que para um recurso digital ser de fato um OA, não basta simplesmente ser criado. Por exemplo, caso fosse criado um objeto de

aprendizagem sobre física quântica para estudantes da pré-escola, este OA não conseguiria alcançar o aprendizado, visto que os estudantes ainda estão sendo alfabetizados. Diante disso, vale ressaltar o que Togni [21] aponta em seus estudos: um projeto de construção de OA exige organização, é preciso delimitar o público-alvo, o conteúdo que será estudado por meio do OA, a forma como será utilizado (se presencial, à distância ou em sala de aula) e o que se pretende que o estudante aprenda a partir do seu uso.

3.2 Aplicações

Com o avanço tecnológico, a troca de informações ocorre de forma acelerada e desenfreada ocasionando um acúmulo de conteúdo aleatório e desorientado. Os objetos de aprendizagem são capazes de sintetizar e esquematizar um conteúdo específico e têm o objetivo de potencializar e dinamizar um determinado conhecimento, tornando-o cada vez mais acessível. As novas gerações de estudantes já estão inseridas neste novo contexto histórico e buscam pela web diferentes formas de obter novos conhecimentos e aprimorar os que já possuem, portanto, cada vez mais, percebe-se a necessidade de novas tecnologias na área educacional.

Na área pedagógica, a utilização de objetos de aprendizagem justifica-se pela lacuna individual que cada estudante apresenta durante o seu desenvolvimento educacional e pelas necessidades de formação inicial e continuada. Ao desenvolver um OA, um educador deve ter clareza que pode proporcionar aos estudantes formas diversificadas de acesso ao conhecimento, como também pode proporcionar o aprimoramento e a complementação do aprendizado.

Percebe-se que a utilização de OAs vem se tornando frequente no processo de aprendizagem de pessoas com necessidades especiais. Os objetos de aprendizagem se tornam ferramentas potentes no ensino para este público, pois além de motivarem e facilitarem o aprendizado, colaboram também para sua inclusão digital [22].

3.3 Repositórios Digitais

De acordo com o Instituto Brasileiro de Informação em Ciência e Tecnologia [1], os Repositórios Digitais (RDs) são coleções de informações digitais, bases de dados online, que reúnem de maneira organizada a produção científica de uma instituição ou área temática e podem ser construídos de diferentes formas e com diferentes propósitos e armazenam arquivos de diversos formatos. Os RDs podem ser divididos de duas formas: institucionais ou temáticos. Os repositórios institucionais lidam com a produção científica de uma

[1] http://www.ibict.br

determinada instituição enquanto que os repositórios temáticos, com a produção científica de uma área específica, sem limites institucionais.

Existem vários repositórios de objetos de aprendizagem com diferentes formatos de mídias, por exemplo, vídeos, imagens, áudios, experimentos, *softwares*. Devido à abrangência, à diversidade e à confiança, escolhemos apresentar os seguintes repositórios: Banco Internacional de Objetos Educacionais (BIOE), Rede Interativa Virtual de Educação (RIVED) e Multimedia Educational Resource for Learning and Online Teaching (MERLOT).

3.3.1 Banco Internacional de Objetos Educacionais - BIOE

O Banco Internacional de Objetos Educacionais[2] [23] é o repositório criado em 2008 pelo Ministério da Educação em parceria com o Ministério da Ciência e Tecnologia, Rede Latinoamericana de Portais Educacionais - RELPE, Organização dos Estados Ibero-americanos - OEI e outros.

O BIOE tem o propósito de manter e compartilhar recursos educacionais digitais de livre acesso e em diferentes formatos - como áudio, vídeo, animação, simulação, *software* educacional, mapa, imagem. Estes objetos são considerados relevantes e adequados à realidade da comunidade educacional local, respeitando-se as diferenças de língua e culturas regionais. Esse repositório está integrado ao Portal do Professor, também do Ministério da Educação, além disso, o repositório conta com recursos e colaboração de diferentes países e línguas, o que enriquece ainda mais o banco de dados.

O BIOE organiza seus objetos em três áreas de busca: recursos, nível e área do conhecimento.

1. **Recursos**

   O repositório apresenta oito tipos de recursos diferentes. São eles:

   - Animação/Simulação - 5929
   - Vídeo - 4376
   - Imagem - 3631
   - Áudio - 3081
   - Experimento Prático - 1768
   - *Software* Educacional - 794
   - Hipertexto - 242
   - Mapa - 21

[2] http://objetoseducacionais2.mec.gov.br

2. **Níveis**

   O repositório apresenta seis tipos de níveis educacionais diferentes. São eles:

   - Ensino Médio - 10289
   - Educação Superior - 9206
   - Ensino Fundamental - 5068
   - Educação Infantil - 851
   - Educação Profissional - 523
   - Modalidades de Ensino - 327

3. **Áreas do Conhecimento**

   O repositório apresenta vinte tipos de áreas do conhecimento diferentes. São eles:

   - Matemática - 4574
   - Física - 3266
   - Química - 2093
   - Biologia - 1590
   - Língua Estrangeira - 1438
   - Língua Portuguesa - 1420
   - Meio Ambiente - 1118
   - Ciências Naturais - 853
   - Agronomia - 784
   - Letras - 743
   - Educação - 628
   - Microbiologia - 608
   - História - 554
   - Biologia Geral - 490
   - Geografia - 486
   - Natureza e Sociedade - 438
   - Literatura - 399
   - Astronomia - 332
   - Informação e Comunicação - 316
   - Botânica - 304

O site também fornece outros dados referentes ao uso de seu repositório. Ele fornece o quantitativo de downloads e visualizações para os anos de 2009, 2010 e 2011 e também mostra o quantitativo de downloads e visualizações por país. No quesito downloads, o Brasil lidera o ranking com mais de 650.000 downloads, o que representa aproximadamente 73% do total, logo em seguida tem os Estados Unidos com 19%. Embora o site não indique o ano referente a esses downloads, é possível inferir que é o total realizado durante o período dos três anos citados.

3.3.2 Rede Interativa Virtual de Educação - RIVED

O RIVED[3] [24] é um programa da Secretaria de Educação a Distância - SEED que tem por objetivo produzir conteúdos pedagógicos digitais na forma de objetos de aprendizagem. Ele tem como meta melhorar a aprendizagem das disciplinas da educação básica e a formação cidadã do estudante. Além de promover a produção e publicar na web os conteúdos digitais para acesso gratuito, o RIVED realiza capacitações sobre propostas metodológicas para produzir e utilizar os objetos de aprendizagem nas instituições de ensino superior e na rede pública de ensino.

A equipe do RIVED, na SEED, foi responsável, até 2003, pela produção de 120 objetos de Biologia, Química, Física e Matemática para o Ensino Médio. Em 2004 a SEED transferiu o processo de produção de objetos de aprendizagem para as universidades cuja ação recebeu o nome de Fábrica Virtual. Com a expansão do RIVED para as universidades, previu-se também a produção de conteúdos nas outras áreas de conhecimento e para o ensino fundamental, para o profissionalizante e para o atendimento às necessidades especiais.

O RIVED oferece um sistema de busca que está organizado em duas áreas: nível e área do conhecimento.

1. **Níveis**

   O repositório apresenta quatro tipos níveis educacionais diferentes. São eles:

   - Ensino Fundamental
   - Ensino Médio
   - Educação Profissional
   - Educação Superior

2. **Áreas do Conhecimento**

   O repositório apresenta doze tipos de áreas do conhecimento diferentes. São eles:

[3] http://rived.mec.gov.br

- Artes
- Biologia
- Biologia – Ensino Superior
- Ciências
- Engenharia – Ensino Superior
- Filosofia
- Física
- Geografia
- História
- Matemática
- Português
- Química

Além da atividade/conteúdo, o RIVED também disponibiliza:

- Guia do Professor - com sugestões de uso do conteúdo para o professor
- Download - permite baixá-los e arquivá-los para uso posterior
- Visualização - permite a execução diretamente da internet
- Detalhes - traz as informações técnicas e pedagógicas sobre o conteúdo
- Comentar - permite opinar, criticar e sugerir sobre cada atividade vista. Aqui as experiências de todos os usuários serão compartilhadas

3.3.3 Multimedia Educational Resource for Learning and Online Teaching - MERLOT

O MERLOT[4] [25] é um projeto que teve início em 1997 e foi desenvolvido pelo Centro Universitário do Estado da Califórnia para Ensino Descentralizado (CSU-CDL).

De acordo com o sítio, em 1998 a Organização Estadual de Ensino Superior/Centro de Qualidade e Produtividade Americano (SHEEO/APQC) fez um estudo sobre o desenvolvimento do corpo docente e da tecnologia instrucional e selecionou o CSU-CDL como um dos seis melhores centros de práticas na América do Norte. A partir disso, outras instituições de ensino superior ficaram interessadas em colaborar com o MERLOT.

O sistema da Universidade da Georgia, do Centro de Ensino Superior do Estado de Oklahoma, da Universidade da Carolina do Norte e da CSU-CDL criaram um consórcio

[4]https://www.merlot.org

informal e alcançaram quase cem campus servindo mais de 900.000 estudantes e mais de 47.000 professores. Esse consórcio dos quatro sistemas estaduais era coordenado pela SHEEO.

Em 1999, os quatro sistemas viram a importância de expandir as coleções do MERLOT e, para isso, cada sistema contribuiu com US$ 20.000 em dinheiro para desenvolver o *software* MERLOT e mais de US$ 30.000 para avançar o projeto de colaboração. A CSU manteve a liderança e as responsabilidades para o funcionamento e a melhoria dos processos e ferramentas.

Em janeiro de 2000, os quatro sistemas selecionaram e patrocinaram 48 professores das disciplinas de Biologia, Física, Administração e Formação de Professores (doze professores de cada um dos quatro sistemas) para desenvolverem normas de avaliação e processos de revisão para o material de ensino-aprendizagem online.

Em abril de 2000, outros sistemas e instituições de ensino superior foram convidados a participar do MERLOT. Em julho de 2000, vinte e três sistemas e instituições de ensino superior tornaram-se parceiros institucionais do MERLOT. Cada parceiro institucional contribuiu com US$ 25.000 e forneceram o apoio de oito professores e um diretor de projeto, em tempo parcial, para coordenar as atividades. A CSU continuou com sua liderança e responsabilidade para a operação e a melhoria dos processos e ferramentas. O MERLOT oferece um sistema de busca que está organizado em sete áreas: materiais, membros, exercícios de aprendizagem, comentários, coleções, ePortfólios e revisão paritária.

1. **Materiais:** dentro da pesquisa de Materiais é possível refinar a busca de quatro formas: pelas disciplinas (artes, educação, matemática e outras), pelo tipo de material (animação, estudo de caso, simulação, testes), pelo suporte móvel (iOS, android, windows mobile e blackberry) e por outros filtros (comentários de membros, classificação dos usuários, revisões paritárias).

2. **Membros:** dentro da pesquisa de Membros é possível refinar a busca de quatro formas: pelas disciplinas (artes, educação, matemática e outras), pelos tipos de membros (professores, servidores, estudantes, administradores), pelas categorias de filiação (eduação, corporação, governo, sem fins lucrativos) e por outros filtros (comentários de membros, classificação dos usuários, revisões paritárias).

3. **Exercícios de Aprendizagem:** dentro da pesquisa de Exercícios de Aprendizagem é possível refinar a busca apenas pelas disciplinas (artes, educação, matemática e outras).

4. **Comentários:** dentro da pesquisa de Comentários, é possível refinar a busca de duas formas: pelas disciplinas (artes, educação, matemática e outras) e pelos tipos de membros (professores, servidores, estudantes, administradores).

5. **Coleções:** dentro da pesquisa de Coleções é possível refinar a busca apenas pelas disciplinas (artes, educação, matemática e outras).
6. **ePortfólio:** dentro da pesquisa de ePortfólio é possível refinar a busca apenas pelas disciplinas (artes, educação, matemática e outras).
7. **Revisão Paritária:** dentro da pesquisa de Revisão Paritária é possível refinar a busca apenas pelas áreas das revisões (biologia, química, engenharia, matemática e outras).

Dentre cada uma dessas pesquisas é possível fazer uma pesquisa avançada na qual é possível filtrar pelo título, descrição, tópicos e outros atributos referentes ao OA.

Capítulo 4

CODEX

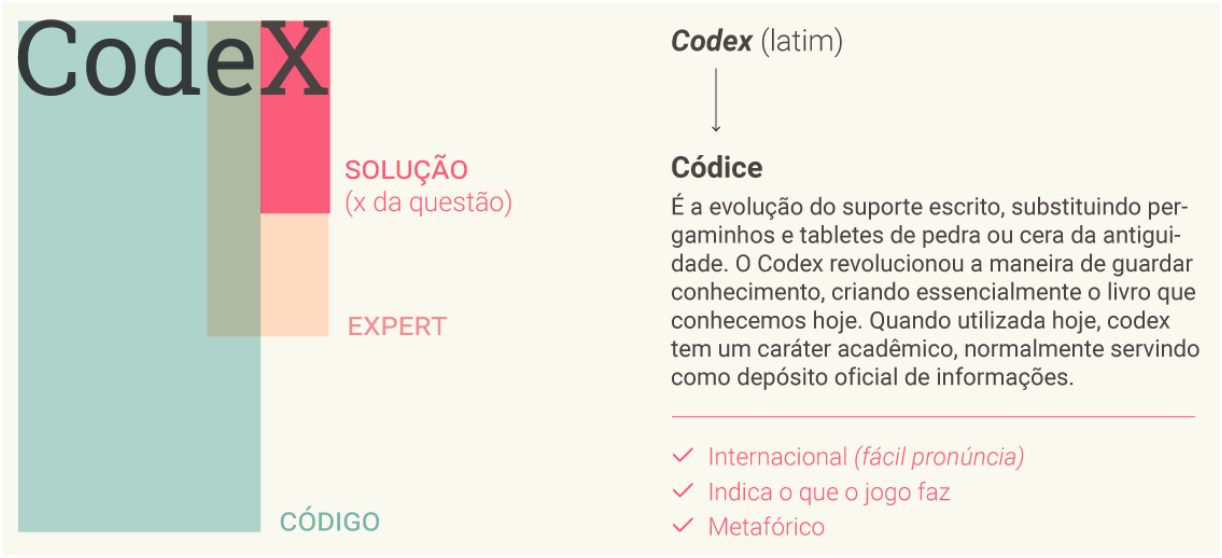
Este capítulo apresentará informações específicas do Objeto de Aprendizagem CODEX que foi desenvolvido neste projeto como uma ferramenta de apoio disponível a qualquer pessoa, de qualquer idade, que tenha interesse e vontade de iniciar estudos na área de programação. Este *software* visa oferecer novas oportunidades de aprendizagem sobre o tema, tanto de forma complementar, quanto de forma inicial, ensinando os conceitos de laços de repetição de forma lúdica e diferenciada.

4.1 Desenvolvimento do *software*

O CODEX é um OA com a finalidade de auxiliar professores, estudantes e curiosos oferecendo um aplicativo gamificado que apoia o ensino e a aprendizagem de laços de repetição. Este processo ocorre de forma lúdica por meio de um jogo que se utiliza de comandos básicos de orientação para trabalhar os conceitos relacionados aos comandos de *loops* de programação.

Durante a criação do CODEX, percebemos a necessidade e o diferencial que um *layout* próprio faria ao OA. O *design* do OA é um recurso capaz de conduzir e facilitar a interação do usuário com o aplicativo. A comunicação de forma eficiente, eficaz e efetiva se torna imprescindível para o êxito de um OA, pois se não conseguir transmitir suas informações de forma clara e precisa, não conseguirá atingir todo seu público.

Figura 4.1: Referência para a definição do CODEX



Fonte: Behr e Belus, 2017

Para a realização desta etapa do *layout*, foi feita uma parceria com estudantes do Departamento de Design Industrial, Max Behr e Luísa Belus. Após se interarem do assunto e compreenderem a temática principal, conseguiram elaborar e desenvolver uma ideia simples e elegante para que o OA atingisse seus objetivos. A equipe também elaborou uma dinâmica de grupo, um *brainstorm*, para nos ajudar a definir o nome do aplicativo (Figura 4.1).

4.2 Tecnologias Utilizadas

Nesta seção serão apresentadas as tecnologias de *software* utilizadas na confecção do objeto de aprendizagem produzido.

4.2.1 React

React[1] é um *framework* usado pelas maiores empresas de *front-end* do mercado (Netflix, Facebook, Instagram), ou *client-side*, e permite o controle do fluxo de renderização e da possibilidade de modularização. Este *framework* é uma biblioteca declarativa, eficiente e flexível do JavaScript e é utilizada para construir interfaces de usuários [26]. Ele também permite que desenvolvedores criem aplicações na internet que utilizam dados que podem sofrer alterações com o tempo sem a necessidade de recarregar a página. Seu objetivo é ser rápido, simples e escalonável.

[1] https://facebook.github.io/react/

O React foi criado por Jordan Walk, um engenheiro de *software* do Facebook. Ele teve influências do XHP (que auxilia a interpretação XML - Extensible Markup Language), uma extensão da linguagem PHP (Hypertext Preprocessor). O XHP converte XML (eXtensible Markup Language - formato para a criação de documentos com dados organizados de forma hierárquica), e, consequentemente, HTML, para blocos de código em PHP com expressões válidas. Além de tornar o código mais fácil de ler, o resultado é uma notação enxuta, que diminui a taxa de erro e auxilia os programadores a manter uma visão geral mais organizada do código [27].

O React foi lançado em 2011 no Facebook e, em 2012, no Instagram.com, sendo atualmente mantido pelo Facebook, Instagram e uma comunidade de desenvolvedores e corporações. De acordo com o Libscore[2], serviço de análise do JavaScript que varre a internet em busca de sites que usam alguma de suas bibliotecas, o React está atualmente sendo utilizado em páginas da internet como Netflix, Imgur, Buffer, Airbnb, SeatGeek, HelloSign, Walmart entre outros [28].

4.2.2 JavaScript

A linguagem JavaScript foi criada por Brendan Eich na Netscape. Inicialmente ela foi implementada como parte dos navegadores web para que scripts pudessem ser executados do lado do cliente (*client-side*) e interagissem com o usuário sem a necessidade de passar pelo servidor. Dessa forma, podendo controlar o navegador e alterar o conteúdo do documento exibido [29].

Esta linguagem de programação tem se transformado em uma das mais populares da web, porém não foi bem recebida no início. Com o surgimento do Ajax, o JavaScript ficou em foco e recebeu mais atenção profissional e o resultado foi a proliferação de *frameworks* e bibliotecas, práticas de programação melhoradas e o aumento no uso do JavaScript fora do ambiente de navegadores, bem como o uso de plataformas de JavaScript server-side [30].

A estruturação em camadas de uma página web pode ser dividida em três camadas: a camada de informação, a cargo do HTML; a camada de apresentação, a cargo do CSS e, por último, a camada de comportamento, a cargo do JavaScript. Sendo assim, o JavaScript é uma linguagem de programação utilizada para controlar o HTML e o CSS.

[2]https://libscore.com/

4.3 Arquitetura e modelagem do CODEX

Flux[3] é uma arquitetura de aplicativos que foi criada pela equipe do Facebook e é usada para criar aplicativos da web do lado do cliente. Ela complementa os componentes de exibição da biblioteca do React e utiliza um fluxo de dados unidirecional.

As aplicações Flux têm três partes principais: *dispatcher*, *stores* e *views* (componentes do React).

4.3.1 *Dispatcher*

O *dispatcher* é o hub central que gerencia todo o fluxo de dados em um aplicativo Flux. É um mecanismo simples que distribui as *actions* para as *stores*. Cada *store* registra e fornece um retorno de chamada. Quando uma *action* é criada, todas as *stores* no aplicativo recebem a *action* por meio das devoluções de chamada no registro.

4.3.2 *Stores*

As *stores* contêm o estado de aplicação e lógica. Elas gerenciam o estado de vários objetos e também o estado de aplicação para um determinado domínio dentro do aplicativo. A *store* registra a si mesmo por meio do *dispatcher* e disponibiliza isso com um retorno de chamada. Este retorno de chamada recebe uma *action* como parâmetro. Após os processamentos e as atualizações das *stores*, elas transmitem um evento sinalizando que seu estado foi alterado para que as *views* possam se atualizar a partir das novas informações.

4.3.3 *Views*

React fornece o tipo de visualizações que são agregáveis e livremente renderizáveis para a camada de exibição. Existe uma *view* especial chamada de *controller-view*. Ela recebe informações transmitidas pelas *stores* que são essenciais à ela. A partir dos dados e informações recebidos, a *controller-view* transmite-os para as outras *views* que estão vinculadas à ela.

4.3.4 *Actions*

As *actions* são cargas úteis de informações que enviam dados da sua aplicação para sua *store*. Eles são a única fonte de informações para a *store*. O *dispatcher* expõe um método que nos permite desencadear um despacho nas *stores* e incluir uma carga útil de dados,

[3] https://facebook.github.io/flux/ (acesso em: 02/09/17)

que chamamos de *actions*. A criação da *action* pode ser encapsulada em um método auxiliar semântico que envia a *action* ao *dispatcher*.

4.4 Ferramentas Similares

Ao pesquisarmos por objetos educacionais em formato de jogo disponíveis na internet e loja de aplicativos com a temática de laços de repetição, encontramos alguns jogos que desenvolvem ideias similares ao proposto pelo CODEX. Optamos dar ênfase ao jogo Lightbot [4], pois tem uma versão gratuita e nos possibilitou compreender mais sobre o que queríamos desenvolver.

O Lightbot (Figura 4.2) foi desenvolvido por Danny Yaroslavski em 2008. Ele é um jogo educacional que permite o aprendizado de princípios lógicos de programação, enfatizando laços de repetição sem precisar escrever nenhum código.

Figura 4.2: Lightbot

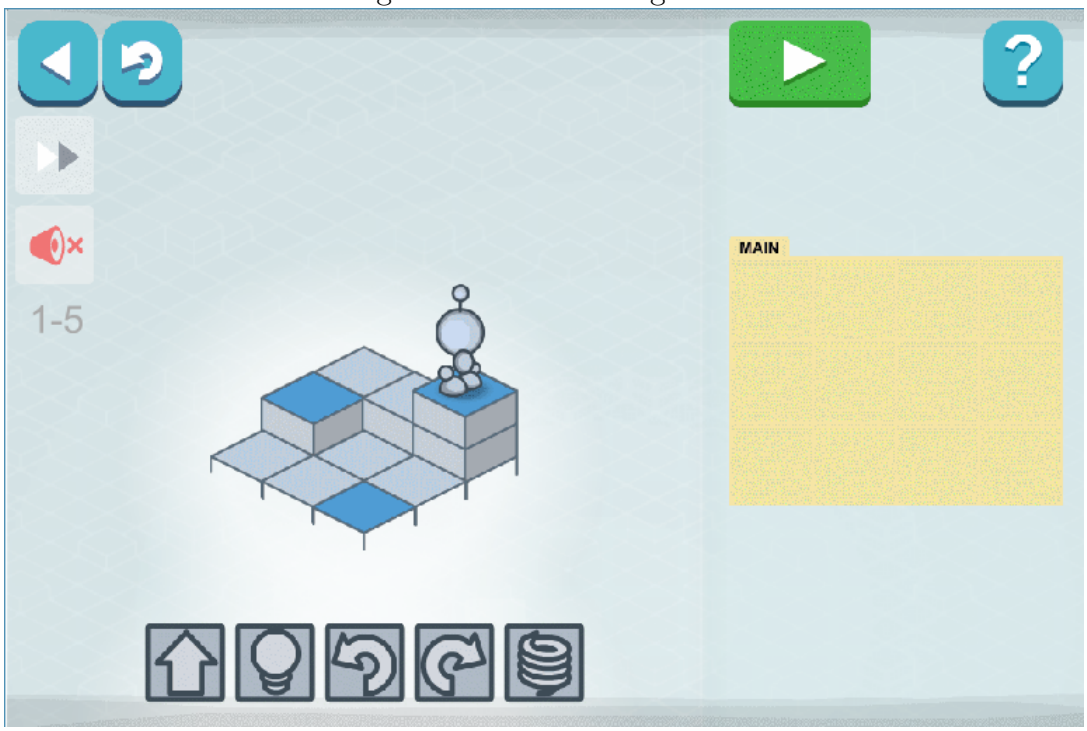
Fonte: Lightbot

O Lightbot consiste em comandar um robô por meio de um tabuleiro resolvendo desafios ao longo dele. É um OA que envolve um mecanismo lógico de programação para cumprir os objetivos. Os usuários devem comandar um robô utilizando comandos de movimento, sequenciais, condicionais e *loops* sem escrever nenhuma linha de código.

[4] https://lightbot.com/flash.html (acesso em: 15/04/17)

Os desafios consistem em que o robô acenda algumas luzes pelo caminho do tabuleiro. O tabuleiro e a complexidade de solução vão aumentando de acordo com o progresso do usuário no jogo.

Figura 4.3: Tela do Lightbot



Fonte: Lightbot

O jogo permite os comandos de avançar, acender a lâmpada, virar no sentido anti-horário, virar no sentido horário e pular. Com esses comandos é possível realizar todos os movimentos para a conclusão dos níveis.

4.5 CODEX

O OA CODEX foi desenvolvido com o intuito de auxiliar a compreensão e funcionamento das estruturas de repetição. O OA incentiva a utilização de *loops* para resolver o desafio proposto em cada nível sem a necessidade de nenhum conhecimento prévio de computação.

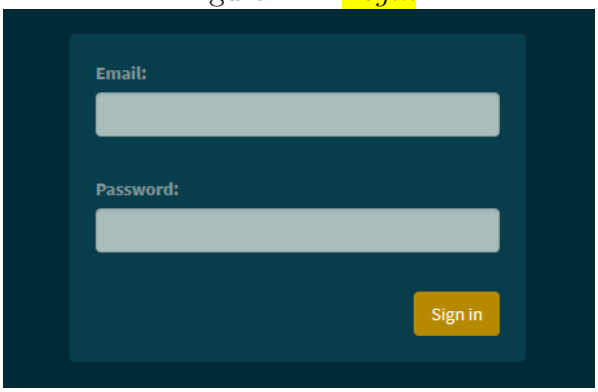
O CODEX consiste em fazer uma bola percorrer o caminho pré-selecionado do círculo rosa até o círculo verde do tabuleiro. Para isso, utiliza-se comandos de movimentação para cima, para baixo, para a esquerda e para a direita e o comando de laço de repetição. Para concluir o objetivo, deve-se selecionar os comandos que a bola deverá realizar para percorrer o caminho desejado.

O OA armazena os dados dos usuários de forma a captar o quantitativo de instruções utilizadas e o número de tentativas em cada nível. Será possível ver quantas vezes cada nível teve soluções corretas e erradas. Estes dados estarão visíveis a todos os usuários.

4.5.1 Funcionamento do CODEX

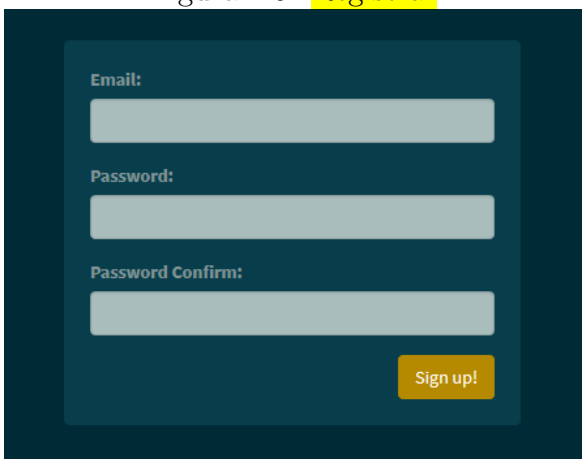
Ao iniciar o OA, no canto superior direito, o usuário terá a opção de fazer o *login* (Figura 4.4) ou se registrar (Figura 4.5) no OA.

Figura 4.4: *Login*



Fonte: Teixeira e Gonçalves, 2017

Figura 4.5: Registrar



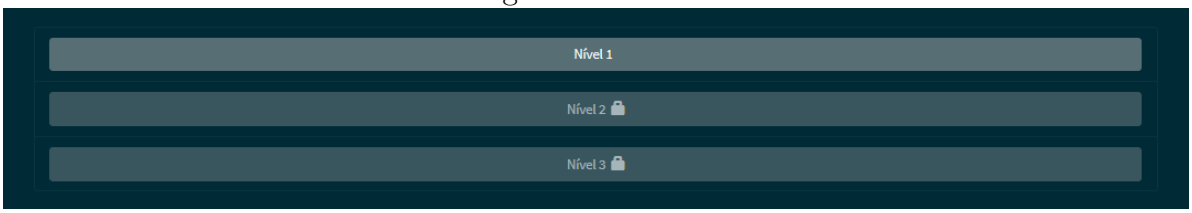
Fonte: Teixeira e Gonçalves, 2017

O registro no OA servirá para manter a evolução do usuário registrando, para cada nível, quantas soluções corretas e quantas soluções erradas foram elaboradas.

Após o *login*, o usuário será redirecionado para a tela que apresenta os níveis do jogo (Figura 4.6) e será capaz de visualizar todos os níveis disponíveis, mas somente poderá ter acesso aos níveis que já concluiu com o êxito e o último nível não completado. Caso seja o primeiro acesso, terá apenas o nível 1 desbloqueado.

Os níveis foram divididos a partir da complexidade de solução de cada um. Dessa forma, o usuário irá construindo e consolidando o raciocínio para poder chegar ao próximo nível. Inicialmente, o jogo conta com três níveis de dificuldade.

Figura 4.6: Níveis

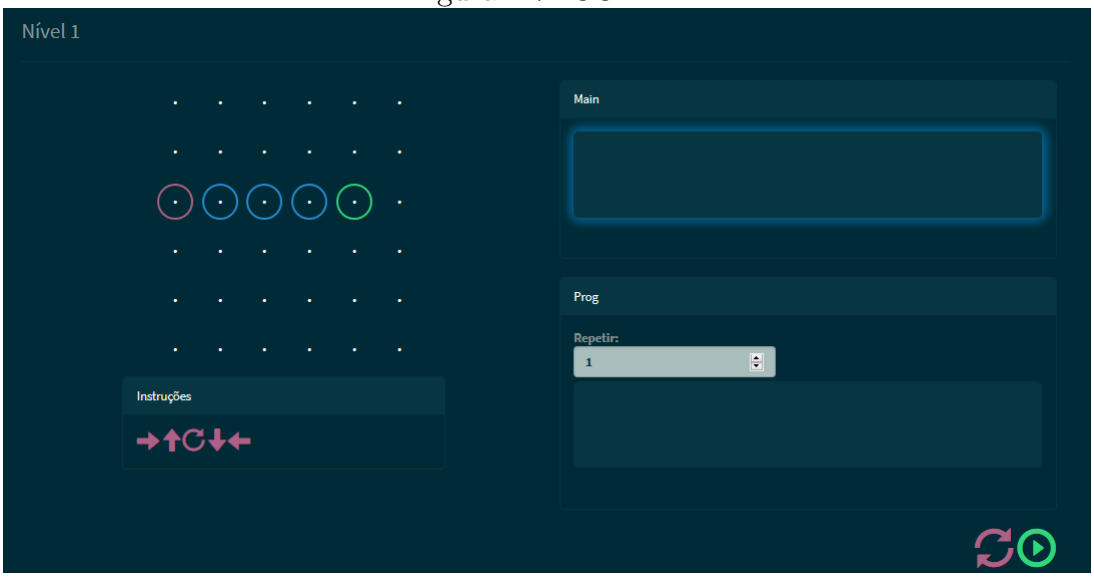


Fonte: Teixeira e Gonçalves, 2017

O símbolo do cadeado representa que o usuário ainda não conseguiu desbloquear um determinado nível e, para isso, precisará concluir o nível anterior para desbloqueá-lo.

Ao entrar em um nível, o usuário terá a tela dividida em duas colunas (Figura 4.7).

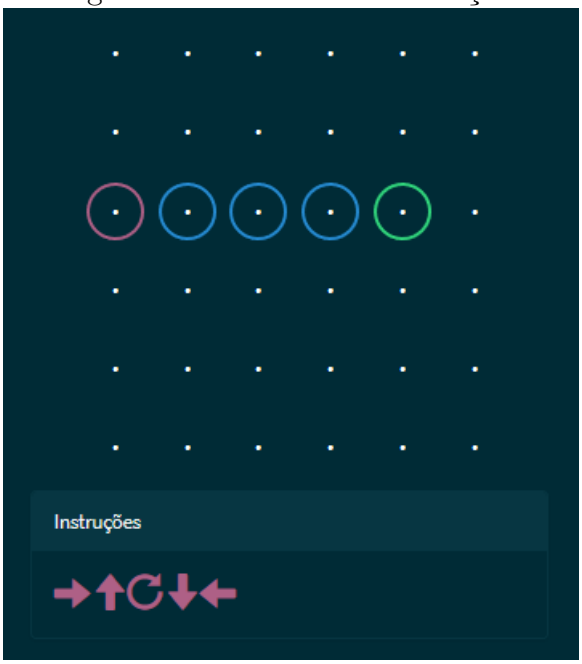
Figura 4.7: CODEX



Fonte: Teixeira e Gonçalves, 2017

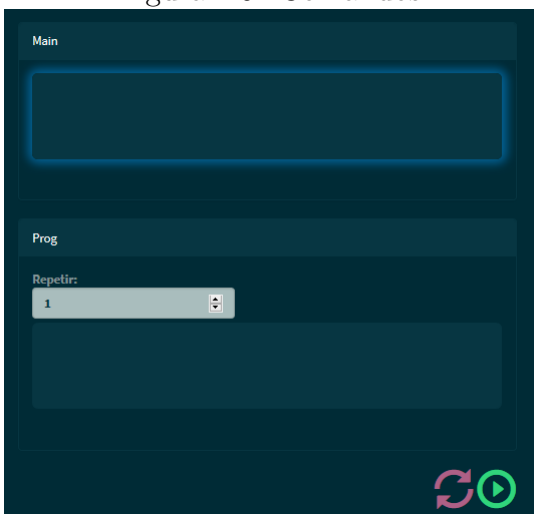
A coluna da esquerda terá o tabuleiro do CODEX e os comandos de movimentação disponíveis abaixo do tabuleiro (Figura 4.8) e, na coluna da direita, o usuário encontrará o *box* de comandos (*Main*), o *box* de *loops* (*Prog.*), o mostrador de quantas vezes o laço de repetição deverá ser repetido, o botão de limpar a tela e o botão de executar (Figura 4.9).

Figura 4.8: Tabuleiro e Instruções



Fonte: Teixeira e Gonçalves, 2017

Figura 4.9: Comandos



Fonte: Teixeira e Gonçalves, 2017

As instruções do CODEX são comandos de simples compreensão e utilização. São eles:


esquerda


cima

baixo

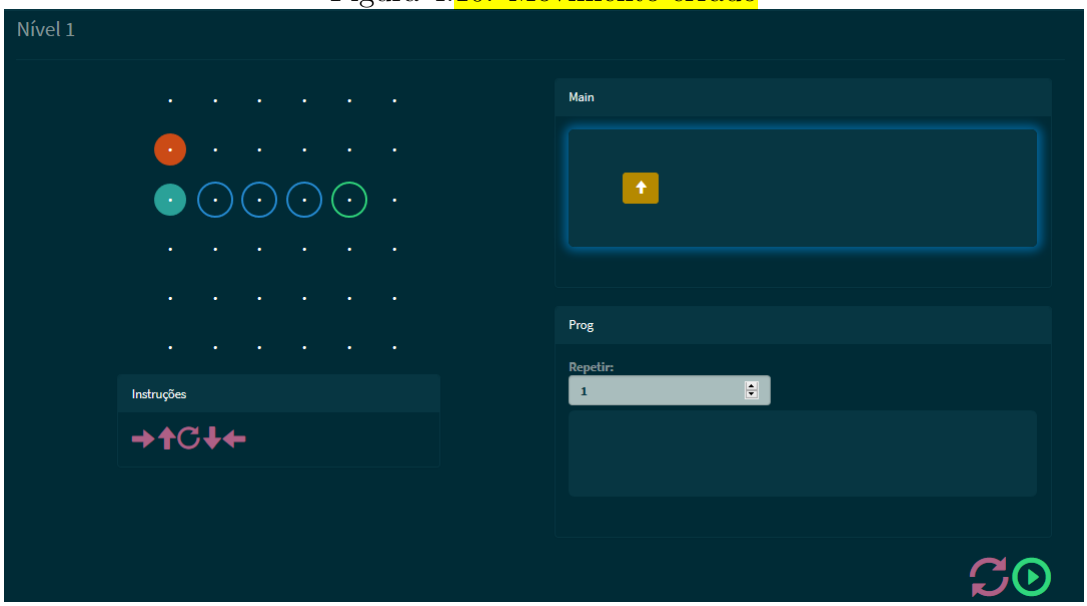

direita


laço de repetição

Ao finalizar um nível com sucesso, o aplicativo identifica que o caminho foi o correto e retorna uma mensagem parabenizando o usuário.

Ao escolher comandos que não resultem na solução do desafio, a bola percorrerá um caminho diferente do esperado e mostrará um rastro que indicará o erro (Figura 4.10).

Figura 4.10: Movimento errado



Fonte: Teixeira e Gonçalves, 2017

Após o erro, o usuário terá que pressionar o botão para limpar a tela e tentar encontrar a solução correta.

4.6 Diferenças entre o CODEX e o Lightbot

O CODEX tem um tabuleiro com uma visão vertical e, dessa forma, possibilita comandos baseados em orientações a partir de um ponto central, como se a locomoção no tabuleiro

fosse orientada pela rosa dos ventos. Esses comandos foram escolhidos porque facilitam a compreensão do usuário e a visualização do tabuleiro. O Lightbot possui um tabuleiro em perspectiva e necessita que o usuário compreenda o posicionamento geoespacial do robô, o que dificulta a dinâmica do jogo. Os comandos do Lightbot requerem que o usuário projete-se na posição para entender qual movimento deve ser utilizado.

Outro diferencial é que o CODEX armazena os dados dos usuários em todos os níveis. Este armazenamento será dividido em quantidades de acertos e erros por nível e possibilitará interpretar as maiores dificuldades dos usuários. No âmbito educativo, este *feedback* permitirá ao professor reconhecer algumas dificuldades encontradas pelos alunos e, assim, trabalhar de forma a saná-las.

Capítulo 5

Considerações Finais

5.1 Conclusões

A temática deste trabalho foi construir um objeto de aprendizagem educacional com a finalidade de ajudar e auxiliar o processo de ensino e aprendizagem de laços de repetição, tanto para o professor quanto para o aluno.

As evoluções tecnológicas e suas influências estão cada dia mais presentes no nosso cotidiano e modificando os processos em nossas vidas. O meio educacional, como vários outros meios, sofreu modificações resultantes dessa era digital.

O uso das tecnologias é capaz de proporcionar novas perspectivas, e, ao mesmo tempo, analisar qual o melhor caminho a ser adotado no processo educacional. Dessa forma, permite-se um ambiente mais propício ao processo de aprendizagem.

O aplicativo foi pensado e elaborado com base nas dificuldades vivenciadas por estudantes de computação. Foi dada a ideia de ser uma ferramenta auxiliar, seja em classe ou extraclasse, para que o aluno possa revisar, compreender e fixar o conhecimento previamente estudado, ou até mesmo, iniciar o processo de aprendizagem de forma lúdica e informal.

Dado que o *software* trabalha com a ideia abstrata de laços de repetição e não utiliza nenhuma linguagem de programação, o estudante é capaz de abstrair e compreender o conceito principal do *loop*. Dessa forma, será capaz de utilizar o conhecimento adquirido em qualquer situação e linguagem que venha a encontrar futuramente.

5.2 Trabalhos Futuros

Com o intuito de aprimorar e aumentar a contribuição do *software* projetado, foi considerado que ainda é possível acrescentar novos recursos e ferramentas com foco no de-

senvolvimento dos alunos. Para isso, como trabalhos futuros, são sugeridas as seguintes implementações:

- módulo de acesso ao professor para que ele tenha a autonomia de criar novas fases e desafios para seus alunos;
- implementação de *learning analytics*;
- autenticação via rede social que permitirá o aluno compartilhar seu desenvolvimento e experiência com o CODEX;

Referências

[1] *Teaching computational thinking is the first step to bridging stem skills gap.* https://edtechmagazine.com/k12/article/2016/11/teaching-computational-thinking-first-step-bridging-stem-skills-gap, Acesso em: 07/09/17. 1

[2] *The case for improving u.s. computer science education.* http://www2.itif.org/2016-computer-science-education.pdf, Acesso em: 07/09/17. 1

[3] *Computing at school in the uk.* https://www.microsoft.com/en-us/research/wp-content/uploads/2016/07/ComputingAtSchoolCACM.pdf, Acesso em: 07/09/17. 1

[4] *Pensamento computacional no ensino de computação em escolas: um relato de experiência de estágio em licenciatura em computação em escolas públicas.* http://ceur-ws.org/Vol-1667/CtrlE_2016_AC_paper_55.pdf, Acesso em: 07/09/17. 1

[5] Vieira, Carlos Eduardo Costa, José Augusto Teixeira de Lima Junior e Priscila de Paula Vieira: *Dificuldades no processo de aprendizagem de algoritmos: uma análise dos resultados na disciplina de al1 do curso de sistemas de informação da faeterj–campus paracambi.* Cadernos UniFOA, (n.27), 2015. 1, 7

[6] *Problems and weaknesses in the teaching and learning of programming: a mapping review.* http://www.br-ie.org/pub/index.php/rbie/article/view/3317, Acesso em: 07/09/17. 1

[7] *Evasão na disciplina de algoritmo e programação: um estudo a partir dos fatores intervenientes na perspectiva do aluno.* http://www.alfaguia.org/www-alfa/images/ponencias/clabesIII/LT_1/ponencia_completa_136.pdf, Acesso em: 07/09/17. 1

[8] Júnior, JCRP e Clevi Elena Rapkiewicz: *O processo de ensino-aprendizagem de fundamentos de programação: uma visão crítica da pesquisa no brasil.* Em *Anais do XII Workshop sobre Educação em Computação (SBC)*, 2004. 4, 5

[9] Raabe, André Luís Alice e JMC da Silva: *Um ambiente para atendimento as dificuldades de aprendizagem de algoritmos.* Em *XIII Workshop de Educação em Computação (WEI'2005). São Leopoldo, RS, Brasil*, 2005. 4, 6

[10] Souza, Draylson Micael, Marisa Helena da Silva Batista e Ellen Francine Barbosa: *Problemas e dificuldades no ensino e na aprendizagem de programação: Um mapeamento sistemático.* Revista Brasileira de Informática na Educação, v.24(n.1), 2016. 4, 8, 10

[11] Júnior, JCRP, Clevi Elena Rapkiewicz, Carla Delgado e José Antonio Moreira Xexeo: *Ensino de algoritmos e programação: uma experiência no nível médio.* Em *XIII Workshop de Educação em Computação (WEI'2005). São Leopoldo, RS, Brasil*, 2005. 5

[12] Martins, Ricartty, Ronaldo Reis e Anna Beatriz Marques: *Inserção da programação no ensino fundamental uma análise do jogo labirinto clássico da code. org através de um modelo de avaliação de jogos educacionais.* Em *Anais do Workshop de Informática na Escola*, volume 22, 2016. 5

[13] California State University: *Projeto logo - LOGO.* `http://projetologo.webs.com/texto1.html`, Acesso em: 11/04/17. 6

[14] Souza, Cláudio Morgado de: *VisuAlg - ferramenta de apoio ao ensino de programação.* Revista Eletrônica TECCEN, v.2(n.2), 2016. 6

[15] Oniria: *Como funciona o treinamento com serious games?* `https://oniria.com.br/como-funciona-o-treinamento-com-serious-games/`, Acesso em: 01/05/17. 9

[16] Moraes, Márcia, Letícia Leite e Paulo Wagner: *Objetos de aprendizagem: a tecnologia apoiando o processo de ensino e de aprendizagem.* Mundo Jovem, 2012. 11

[17] Sabbatini, Marcelo: *Reflexões criticas sobre o conceito de objeto de aprendizagem aplicado ao ensino de ciências e matematica.* Em Teia | Revista de Educação Matemática e Tecnológica Iberoamericana, 2013, ISSN 2177-9309. `https://periodicos.ufpe.br/revistas/emteia/article/view/2189`, Acesso em: 11/04/17. 11, 12

[18] Wiley, David A: *The Instructional Use of Learning Objects: Online Version.* 2000. `http://reusability.org/read/chapters/wiley.doc`, Acesso em: 11 de abril de 2017. 12

[19] Koper, Rob: *Combining re-usable learning resources to pedagogical purposeful units of learning in: Littlejohn, a. and buckingham shum, s.(eds.) reusing online resources (special issue) journal of interactive media in education, issn: 1365-893x*, 2003. 12

[20] Tavares, Sandra Cristina Samico de Pinho: *Desenvolvimento de um learning object para o ensino-aprendizagem da língua inglesa: regra de formação do present simple.* Tese de Doutoramento, 2006. 12

[21] Togni, Ana Cecília: *Construindo objetos de aprendizagem.* `http://paginapessoal.utfpr.edu.br/kalinke/novas-tecnologias/grupos-de-pesquisa/grupos-de-pesquisa/pdf/construindo_objetos.pdf`, Acesso em: 11 de abril de 2017. 13

[22] Balbino, Raquel Ribeiro et al: *Jogos educativos como objetos de aprendizagem para pessoas com necessidades especiais*. Renote, v.7(n.3):p. 209–220, 2009. http://www.seer.ufrgs.br/renote/article/view/13591, Acesso em: 11/04/17. 13

[23] Ministério da Educação: *Banco internacional de objetos educacionais - BIOE*. http://objetoseducacionais2.mec.gov.br/staticspages?t=0, Acesso em: 11/04/17. 14

[24] Ministério da Educação: *Rede internacional virtual de educação - RIVED*. http://rived.mec.gov.br/site_objeto_lis.php, Acesso em: 11 de abril de 2017. 16

[25] California State University: *Multimedia educational resource for learning and on-line teaching - MERLOT*. http://info.merlot.org/merlothelp/index.htm#who_we_are.htm, Acesso em: 11/04/17. 17

[26] *Tutorial: Intro to react*. https://facebook.github.io/react/tutorial/tutorial.html#what-is-react, Acesso em: 04/06/17. 21

[27] *Introduction to react*. https://pt.scribd.com/document/356326308/React, Acesso em: 22/08/17. 22

[28] *React (javascript library)*. https://en.wikipedia.org/wiki/React_(JavaScript_library), Acesso em: 04/06/17. 22

[29] *O que é javascript?* http://tableless.github.io/iniciantes/manual/js/, Acesso em: 04/06/17. 22

[30] *Javascript*. https://pt.wikipedia.org/wiki/JavaScript, Acesso em: 04/06/17. 22