LINGUAGEM DE PROGRAMAÇÃO VISUAL: UMA NOVA FORMA DE APRESENTAR A PROGRAMAÇÃO DE COMPUTADORES

https://doi.org/10.5281/zenodo.15558160

TRINDADE, Andrea Garcia, Especialista*

*ETEC de Praia Grande
Pça. 19 de Janeiro, 144, Boqueirão, Praia Grande / SP, CEP: 11700-100
Fone (13) 3591-1303
Prof_andrea@hotmail.com

RESUMO

A iniciação ao desenvolvimento de programação de computadores sempre foi um desafio para professores e alunos. Por meio de novas tecnologias, pessoas que nunca programaram estão conseguindo desenvolver seus aplicativos. Tecnologias baseadas em linguagens de programação visuais, diferentemente das linguagens de programação textuais, os conhecimentos prévios para desenvolvimento são mínimos. Estas linguagens proporcionam desenvolvimento de aplicativos de maneira simples e rápida, procurando despertar o interesse em programação desde o público infantil, alunos de cursos de exatas, com dificuldades em desenvolver raciocínio lógico, até pessoas que não trabalham na área de informática, mas que pretendem desenvolver seus próprios aplicativos.

PALAVRAS-CHAVE: Linguagem de programação visual, programação em blocos, visuais, *Appinventor, Scratch,* iniciação a programação.

ABSTRACT

Initiating the development of computer programming has always been a challenge for teachers and students. Through new technologies people who have never programmed are succeeding in developing their applications. Technologies based on visual programming languages, where unlike the textual programming languages, prior knowledge

for development are minimal. These languages provide application development simple and fast way seeking to reawaken interest in programming from younger audiences, students in courses with exact difficulties in developing logical reasoning even people who do not work in computer science, but who wish to develop their own applications .

KEYWORDS: *visual programming language, block programming, visual, AppInventor, scratch, start programming*

INTRODUÇÃO

A Linguagem de Programação Visual, ou *Visual Programming Language* (VPL), é um novo conceito em aprendizagem de programação de computadores. Com a expansão do uso dos recursos tecnológicos, mais pessoas começarem a programar computadores e várias linguagens de programação foram e estão em desenvolvimento. A VPL se apresenta como uma iniciação ao ato de programar, não substituindo as linguagens de programação como já conhecida nas linguagens textuais, mas como um recurso simples e rápido para o desenvolvimento de programas.

Iniciar a atividade da programação de computadores sempre foi um desafio para professores e alunos. Além dos conceitos de lógica e algoritmos, o programador iniciante precisa também agregar o conhecimento de uma linguagem de programação com suas características próprias e uma série de estruturas e comandos em língua estrangeira. Muitos desistem de aprender a programar devido a esta série de conhecimentos que devem ser mesclados. A dificuldade de adaptação dos alunos em desenvolver raciocínio lógico, quando muitas vezes estão acostumados a decorar conteúdos, unindo-se à falta de motivação destes, gerada pelo despreparo, podem ser citados como motivos para a ocorrência da desmotivação, reprovação e até mesmo a evasão dos cursos.

Segundo Kamiya (2009, *apud* BORGES, 2000, p.1), a disciplina de Lógica de Programação "costuma ter altos índices de evasão e reprovação, sendo um dos gargalos existentes nos cursos de exatas, dificultando, ou até mesmo, impedindo a continuidade dos

alunos no curso". Entre os vários motivos citados como geradores desse problema, está a dificuldade que os alunos encontram para desenvolver programas (algoritmos) necessários para a experimentação (simulação) (TOBAR, 2001).

De acordo com Pereira (2012, p.21): "É necessário que os professores que trabalham com as disciplinas de algoritmos busquem soluções para minimizar o número de reprovações ou abandonos. Uma das formas de sanar esse problema é usar ferramentas ou ambientes facilitadores que provoquem um aprendizado substancial das atividades didáticas".

Na linguagem de programação visual, o usuário desenvolve programas por meio da manipulação de elementos do programa, graficamente, em vez de especificá-los textualmente. A VPL permite a programação com expressões visuais, arranjos espaciais de texto e símbolos gráficos, utilizados tanto como elementos de sintaxe ou notação secundária. Podem ser apresentados baseados na ideia de "caixas e flechas", onde caixas ou outros objetos da tela são tratados como entidades, ligadas por setas, linhas ou arcos que representam as relações, como a *Microsoft Visual Programming Language.* Ou ainda, no formato de quebra-cabeças, encaixando-se os recursos e instruções através de blocos. A linguagem de programação visual não deve ser confundida com ambiente de programação visual, como ocorre, por exemplo, no ambiente do *Microsoft Visual Studio*. Neste, as linguagens oferecidas (*Visual Basic, Visual* C#, etc.) são textuais, não gráficas, por isso, pode ocorrer uma falta de compreensão sobre a conceituação. Nas figuras 1 e 2 são apresentados exemplos de ambiente e linguagem de programação visual.

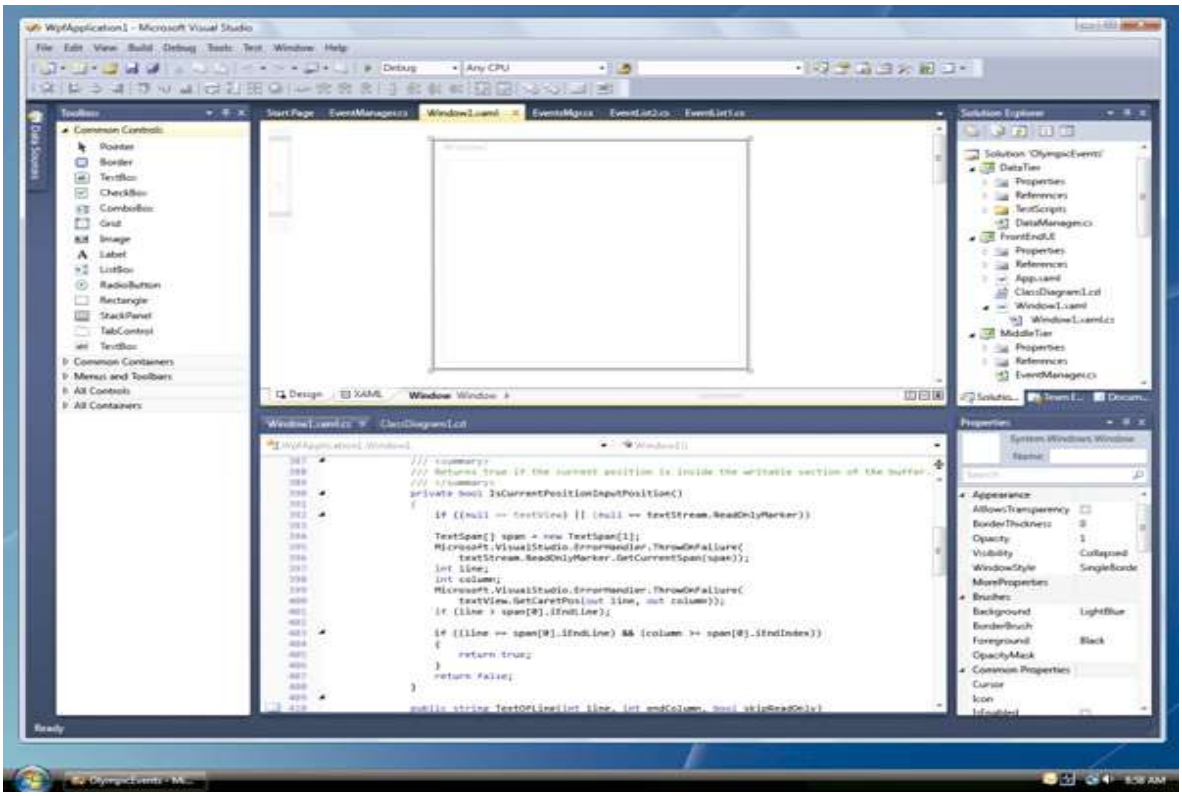

Figura 1 - Ambiente de programação visual – *Microsoft Visual Studio*
Fonte: Microsoft (2014).

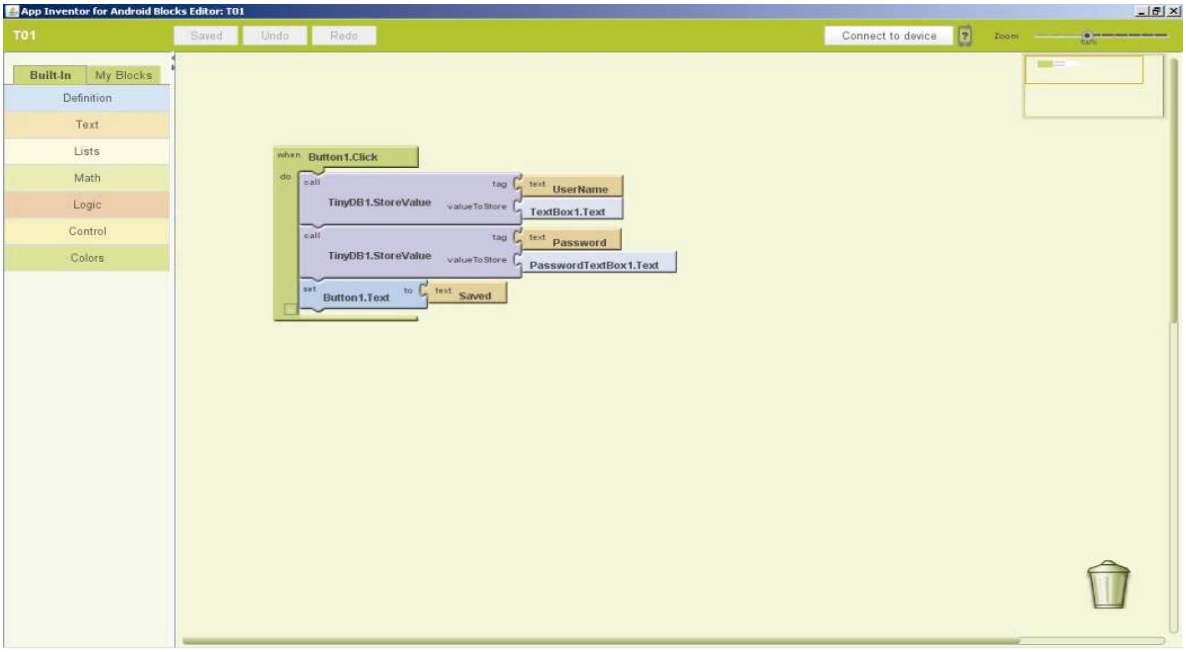

Figura 2 – Linguagem de Programação Visual – do *Appinventor*
Fonte: Microsoft (2014).

1 ALTERNATIVAS PARA INICIANTES EM PROGRAMAÇÃO

Neste artigo são apresentadas três linguagens de programação visual que estão bem popularizadas mundialmente: *Scratch,* desenvolvido e mantido pelo Lifelong Kindergarten Group, no Media Lab do MIT[1]; a *App Inventor*, desenvolvido pelo Google Labs e, atualmente, mantido pelo *Center for Mobile Learning* do MIT; e o projeto Google Blockly, que é uma linguagem de programação visual do Google Labs.zx.

Todas possuem características de programação simples, apesar de fornecerem saídas específicas. Principais características dessas linguagens:

a) *Scratch*: onde as animações, desenvolvimento de jogos são incentivados, pela motivação do público alvo;
b) *App Inventor* (possui a mesma biblioteca do *Scratch*): tem o propósito de desenvolvimento de aplicativos para dispositivos móveis baseados no sistema operacional Android;
c) *Google Blockly:* onde além do desenvolvimento do raciocínio lógico e conhecimentos sobre algoritmos e programação, pode-se converter o conteúdo da programação visual em linguagens como XML, *Javascript* ou *Phyton.*

Nestes, as instruções são apresentadas em blocos, agrupadas por cores de acordo com sua função, onde, para programar, basta o usuário clicar no recurso e arrastar para a área de programação, encaixando-se em outro bloco, formando uma conexão parecida com quebra-cabeças. Os blocos foram desenvolvidos para se encaixar apenas de uma única forma, fazendo sentido sintaticamente, não ocorrendo, desta forma, erros de sintaxe.

Algumas características similares encontradas nestas linguagens de programação visual são:

a) Utilização através de arrastar, soltar e de encaixar, com controles lógicos e de programação;
b) Código de cores para os diferentes elementos – os objetos de texto e de lógica são verdes, outros objetos são azuis, as variáveis são de cor roxa, etc;

1 Massachusetts Institute of Technology (Instituto de Tecnologia de Massachusetts, USA).

c) Objetos de programação podem ser personalizados para diferentes configurações, como o problema do labirinto Blockly;
d) Capacidade de criar procedimentos ou funções (não completamente funcionais nos demos Blockly).

Estes blocos podem ser ajustados de acordo com a necessidade do programa que está sendo desenvolvido, como: mensagens, verificação de condições, criação de variáveis, entre outros itens.

O conceito do produto final também difere do tradicional arquivo executável gravado localmente em máquina para a armazenagem na nuvem, no servidor que fornece a linguagem de programação visual, podendo ser distribuído e acessado por qualquer usuário na internet.

1.1 *SCRATCH*

É uma linguagem de programação visual, desenvolvida por *Lifelong Kindergarten Group* no *Media Lab*, MIT (com financiamento da *National Science Foundation, Intel Foundation, Nokia* e do consórcio de pesquisa do MIT *Media Lab*) desde 2003, coordenada por Mitchel Resnick, inspirada nas linguagens Logo e Squeak, que possibilita a criação de histórias interativas, animações, jogos, música e arte além de permitir o compartilhamento de suas criações na web.

O *Scratch* foi apresentado com o mundo pela primeira vez, em 15 de maio de 2007 e, atualmente, é utilizado em mais de 150 países e disponibilizado em mais de 40 idiomas. Foi projetado para ajudar os jovens (acima de 8 anos), a desenvolverem habilidades de aprendizagem do século 21. Com a criação de projetos utilizando o *Scratch*, os jovens aprendem ideias matemáticas e computacionais importantes, além de ganhar uma compreensão mais profunda do processo de *design.*

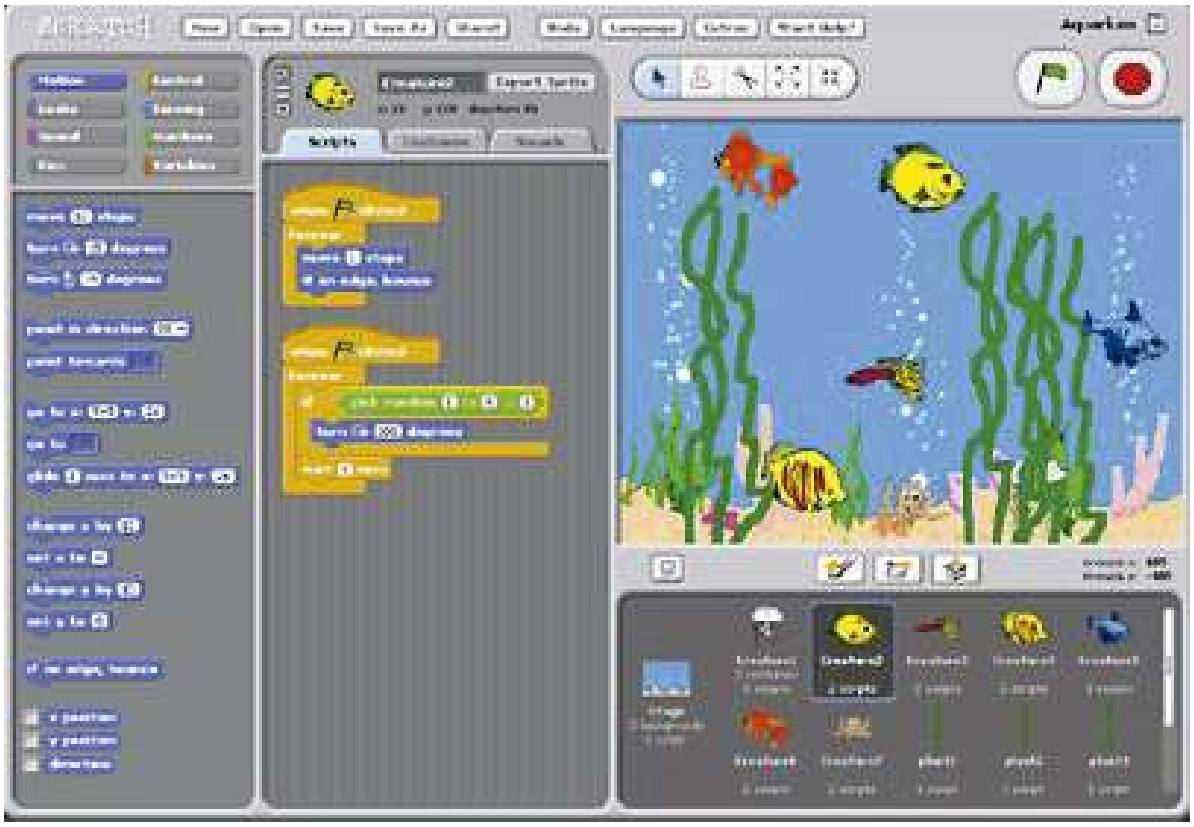

Figura 3 - Ambiente de Desenvolvimento *Scratch*
Fonte: Scratch (2014).

O ambiente de desenvolvimento é dividido em três áreas, como visto na figura 3. A primeira área contém os comandos que serão adicionados ao seu programa. Encontram-se comandos de: controle, movimentos, operações, aparência, sons e outros. Na segunda área fica o desenvolvimento do programa, onde aparecem os blocos de comandos, os trajes de seus *sprites* (os desenhos que aparecem na tela) e os sons que acompanham os *sprites.* E na terceira área encontra-se a tela de animação, chamada de Palco, onde o programa é executado.

Esta linguagem de programação visual pode ser desenvolvida diretamente na nuvem, ou na falta de acesso à internet, através de *download* e instalação local. Já existe versão para a língua portuguesa e inúmeros materiais para aprendizagem.

Como observado na figura 3, as instruções são apresentadas na forma de blocos, onde, separadas por cores, diferenciam os recursos (controles, sons, dados, mensagens, operadores, etc). Existe também uma grande biblioteca de imagens, sons e cenários para o desenvolvimento.

Pode-se ainda salvar o projeto e executá-lo ou ajustá-lo posteriormente (SCRATCH, 2014).

1.2 *APP INVENTOR*

O *App Inventor* é uma plataforma de desenvolvimento de aplicativos Android, desenvolvido para a utilização de pessoas com pouco ou sem conhecimento algum sobre programação. O *App Inventor* é a criação de Hal Abelson, coordenado pelo professor do MIT (enquanto estava no Google). A plataforma é uma consequência de seu trabalho no projeto do *Scratch*, no qual também participou. Para desenvolver aplicativos no *App Inventor*, não é necessário escrever código, basta arrastar e montar as peças como se fosse um quebra-cabeça, criando os seus desenhos e definindo o que eles farão.

O *App Inventor* foi originalmente criado no Google Labs, mas atualmente, pertence ao MIT Labs, que vem empregando programação orientada por eventos para metodologias educacionais. Cria-se, facilmente, qualquer tipo de aplicativo, desde jogos, aplicativos para empresas, aplicativos de desenho ou os que utilizam mapas e GPS. Ele é voltado para o desenvolvimento de aplicativos para sistemas móveis com sistema operacional Android.

Assim como o *Scratch,* o *App Inventor* é executado em ambiente on-line, onde os projetos ficam armazenados nos servidores virtuais (*AppInventor Servers)*.

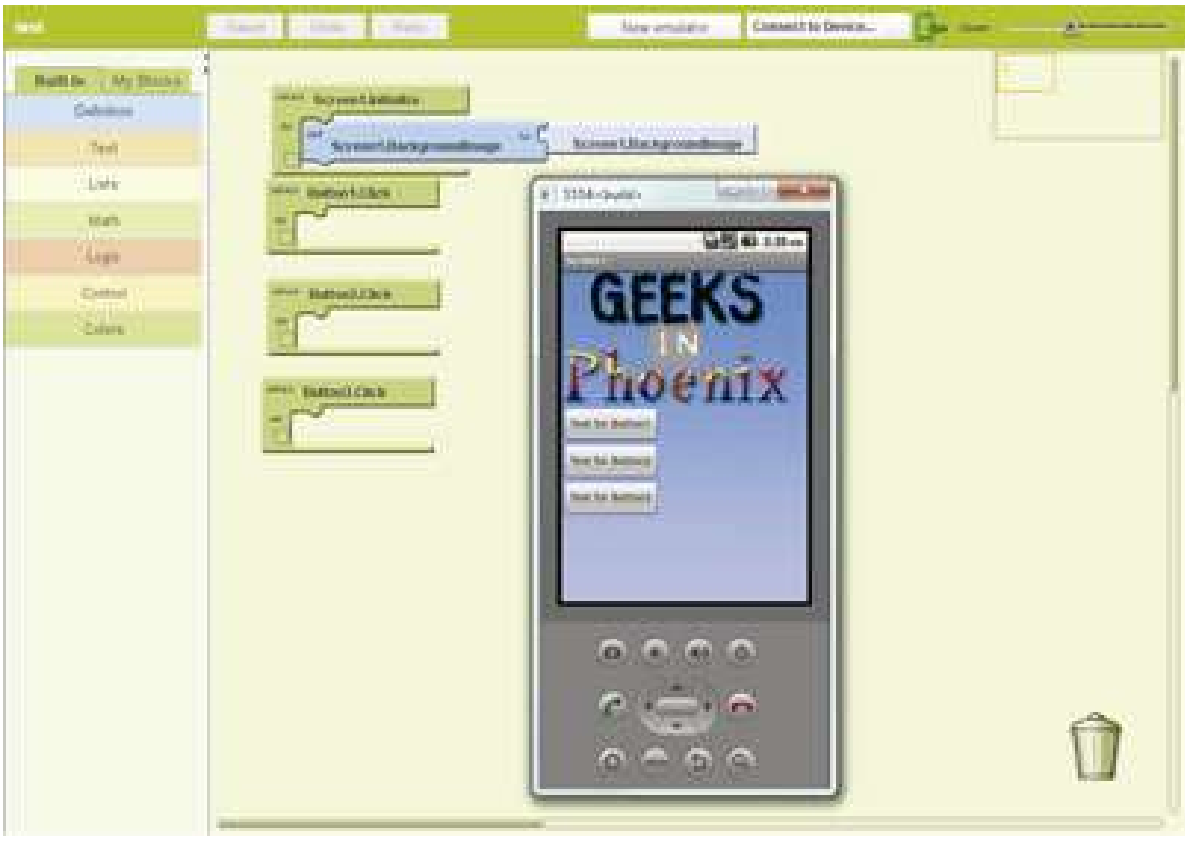

Figura 4 – Ambiente de Desenvolvimento – *Appinventor*
Fonte: Appinventor (2014)

Para sua utilização utiliza-se uma IDE (*Integrated Development Environment*), com Ambiente de Programação Visual, muito semelhante ao da Microsoft Studio em sua composição, mas quando necessita da área de programação, chamada de Editor de Blocos, a linguagem de programação apresenta-se no formato de blocos, com recursos separados por cores, no formato de clicar e arrastar (figura 4).

Possui ainda, na sua visualização de produto final, a possiblidade de ser executado em um emulador, que deve ser instalado no computador local do usuário ou através do próprio dispositivo móvel (celular ou *tablet*), que o usuário possua. Para isto é transferido o programa do ambiente virtual para o aparelho indicado.

Em dezembro de 2013, foi apresentada a versão 2 da linguagem, com novos recursos e estabilidade em sua programação (APPINVENTOR, 2014).

A Universidade de São Francisco (USA) desenvolveu um projeto chamado "*App Inventor to Java*", através da equipe *Democratize*

Computing Lab, coordenado pelo professor David Wolber e que consiste em gerar programas em Java, equivalente ao que foi desenvolvido com o *App Inventor*, em sua IDE, podendo após sua conversão ser editado na IDE Eclipse. O código gerado utiliza a biblioteca *java bridge*, código criado pelo Google, e agora gerenciado pelo time MIT App Inventor. Ainda encontra em versão de testes (APPINVENTOR, 2014).

1.3 GOOGLE BLOCKLY

Esta linguagem de programação visual, baseada em blocos, é um projeto do Google, lançado em junho de 2012. Todo o código do *Google Blockly* é gratuito e *open source* (código aberto), portanto, pode-se manipular os dados da forma que desejar. A tecnologia vem para facilitar a criação de apps (aplicação/aplicativos), evitando que o programador precise decorar comandos e trabalhar com infinitas linhas de código.

Para iniciar sua utilização, existe uma área para um aprendizado gradual, baseado em gamificação, onde por meio da montagem da programação correta, um objetivo é proposto e explicado antes da programação ser iniciada (BLOCKLY, 2014).

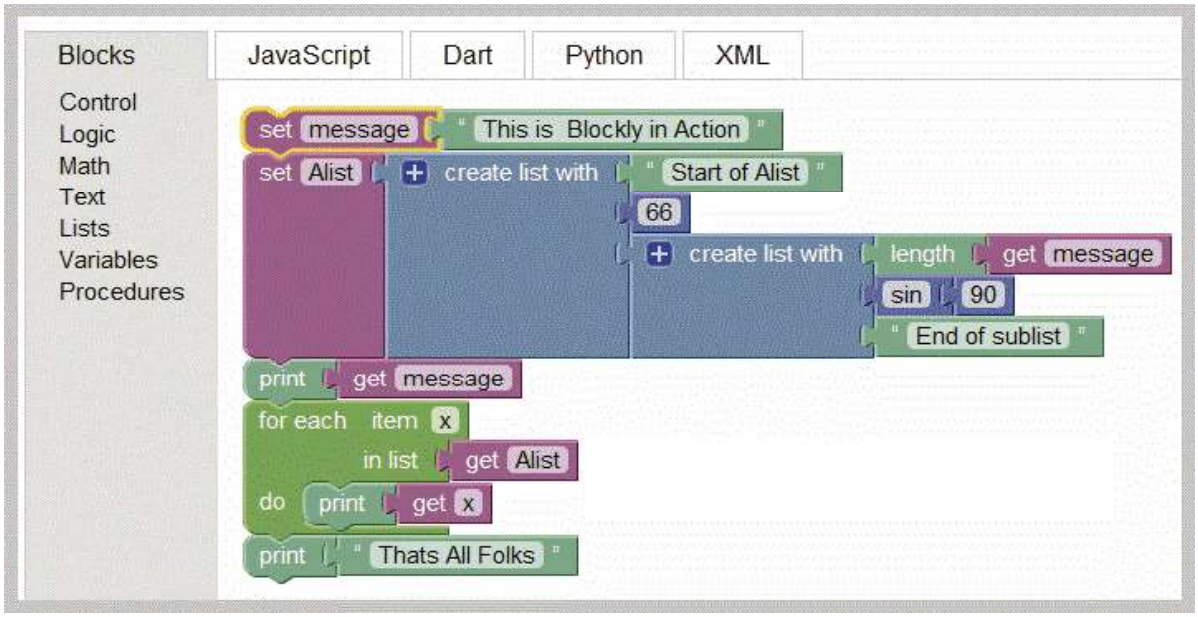

Figura 5 – Ambiente de Desenvolvimento – Blocky

Fonte: Google Blockly (2014).

A versão atual do *Blockly* não possui um ambiente de programação (IDE) como o *Scratch* ou o *AppInventor*, com recursos

como passo a passo de depuração ou utilização de som, cor, sprite e outras extensões de mídia (figura 5).

No entanto, *Blockly* (que é codificado em *Javascript*) produz código utilizável em *JavaScript, Dart, Python* e XML, o que pode ser interessante para conciliar a passagem da aprendizagem da linguagem de programação visual para a linguagem de programação textual. Porém, como está em estágio de desenvolvimento, no momento, ele está adequado para a geração de *scripts* curtos (conjunto de comandos), tendo como próximo estágio atingir a codificação em *JavaScript.*

1.4 OUTRAS INICIATIVAS

Outras iniciativas de desenvolvimento durante o ano de 2013 aconteceram, promovendo o desenvolvimento de códigos, e o principal meio de apresentação para isto foram as linguagens de programação visuais. Como por exemplo, o caso do evento "Hora do Código", promovido pela Code.org[2], em dezembro de 2013 (figura 6). Na ocasião, pessoas sem conhecimento prévio puderam conhecer e serem introduzidas à Ciência da Computação, por meio de material interativo, incluindo tutoriais de Bill Gates e Mark Zuckerberg. A intenção era atingir 10 milhões de estudantes, mas a iniciativa conseguiu alcançar um total de 29.498.992 participantes (CODE, 2014).

2 É uma associação sem fins lucrativos, dirigida por Hadi Partovi, cujo objetivo é divulgar e ensinar programação às pessoas de todas as idades, mantendo parcerias com importantes companhias do setor tecnológico, como Google, Microsoft, Amazon e LinkedIn, por exemplo (CODE, 2014).

Figura 6 – Projeto "Hora do Código" – Code
Fonte: CODE (2014).

No caso do Brasil, a partir de 2014, surgiu a iniciativa do chamado "Ano do Código", criado e mantido por diversas empresas, como GUJ, Alura, Globo.com, Code Miner, Locaweb e Casa do Código, que procura divulgar e incentivar a programação utilizando os recursos semelhantes aos apresentados no projeto "Hora do Código", como a VPL para a iniciação a programação (LOCAWEB, 2014).

Outro exemplo é a faculdade de Ciências da Universidade de Lisboa, que promoveu em julho de 2013, o "O FCUL Rally 2013", que organizou um concurso de programação dirigido aos alunos do Ensino Médio, no qual as ferramentas de desenvolvimento de raciocínio lógico envolviam a utilização da linguagem *Blockly* (CIÊNCIAS ULISBOA, 2014).

Além destas iniciativas, se pode citar outras linguagens visuais que promovem o desenvolvimento da programação, sendo elas:

a) *Alice:* programação em blocos visuais para criar animações em cenários em 3D;
b) *Kodu:* ferramenta da *Microsoft Research* para desenvolvimento de jogos em 3D, sem linhas de código;
c) *Gamesalad* e *Construct 2: d*esenvolvimento de jogos, sem utilização de programação.

CONSIDERAÇÕES FINAIS

As linguagens de programação visuais são ferramentas que auxiliam na apresentação e no aprendizado da programação de computadores, promovendo o interesse crescente e contínuo de usuários comuns a iniciação do desenvolvimento de aplicativos. Uma das frustrações de quem começa a aprender a programação é a falta de resultado imediato, onde estas linguagens conseguem suprimir esta necessidade. O público alvo destas linguagens são usuários iniciantes na programação, onde a aprendizagem da sintaxe e os comandos em língua estrangeira ficam para um segundo momento. Deve-se observar ainda que pela facilidade em sua utilização até o público infantil pode utilizá-la como ferramenta de apoio para as demais disciplinas curriculares, no desenvolvimento de projetos que promovam o raciocínio lógico e matemático. Podem promover o interesse em alunos de cursos de Exatas que encontrem dificuldades no desenvolvimento de raciocínio lógico, auxiliando na redução dos índices de evasão destes cursos, muitas vezes ocorridos pela falta de compreensão do desenvolvimento de lógica e algoritmos.

Estas linguagens não são tão poderosas quanto às linguagens textuais, mas podem ser ajustadas conforme a necessidade e objetivos do programa de forma simples, rápida e lúdica. Com uma ampla aplicabilidade, permitem a criação de jogos, aplicativos para celulares com utilização de recursos sofisticados como aplicação de mapas, utilização de câmera e acesso às páginas de web, além da possibilidade de exportação para linguagens de programação textuais. Podem substituir as linguagens de programação textuais inicialmente, mas não em sua totalidade, as quais possuem um potencial superior de recursos computacionais.

Podem ser utilizadas como ponto de partida para estimular o interesse da programação de computadores pela forma direta e simples em sua utilização. É uma opção viável para a iniciação à programação de computadores.

REFERÊNCIAS

APPINVENTOR. About Us. Disponível em: <beta.appinventor.mit.edu>. Acesso em 26 fev. 2014.

BLOCKLY. *A visual programming editor*. Disponível em: <http://code.google.com/ p/blockly/?redir=1>. Acesso em 28 fev. 2014.

CIÊNCIAS ULISBOA. *FCUL Rally Pro 2013*. Disponível em: www.fc.ul.pt/en/conferencia/fcul-rally-pro-2013. Acesso em 12 jan. 2014.

CODE. Aprenda Programação com a Hora do Código. Disponível em:http://code.org/. Acesso em 12 jan. 2014.

JORDÃO, Fábio. **Google Blockly**: uma linguagem de programação baseada em quebra-cabeças. Disponível em: <http://www.tecmundo.com.br/programacao/24947-google-blockly-uma-linguagem-de-programacao-baseada-em-quebra-cabecas.htm>. Acesso em 2 mar 2014.

KAMIYA, Reginaldo R; BRANDÃO, Leônidas de O. **iVProg - um sistema para introdução à Programação através de um modelo Visual na Internet**. USP, 2009.

LOCAWEB. *Press Releases*. Disponível em: < http://press.locaweb.com.br/page/4?wptouch_ view=normal&wptouch_redirect_nonce=4088b1d581&wptouch_redirect=%2F788%2Flocaweb-participa-do-black-friday-brasil-com-promocoes-ineditas>. Acesso em 15 jan. 2014.

MICROSOFT. Conhecendo o Visual Studio. 2014. Disponível em: <https://msdn.microsoft. com/pt-br/library/windows/apps/dn263225.aspx>. Acesso em 20 fev. 2014.

SCRATCH. Para Educadores. Disponível em:<http://scratch.mit.edu/>. Acesso em 26 fev. 2014.

PEREIRA, Pryscilla de S.;MEDEIROS, Marcos; MENEZES, José

W. M.**Análise do Scratch como Ferramenta de Auxílio ao Ensino de Programação de Computadores**.COBENGE – Congresso Brasileirode Educação em Engenharia, Belém-PA.2012.

TOBAR, C. M. **Uma arquitetura de ambiente colaborativo para o aprendizado de programação.** XII Simpósio Brasileiro de Informática na Educação - SBIE2001.

Tutorial Scratch – Conceitos básicos (versão XO-OLPC). Projeto "XO na escola e fora dela: Uma Proposta Semio-Participativa para Tecnologia, Educação e Sociedade": nº 475105/2010-9, Edital MCT/ CNPq 14/2010 http://www.nied.unicamp.br/xounicamp/. Disponível em <http://portalsme.prefeitura.sp.gov.br/Projetos/ie/Documentos/scratch.pdf>. Acesso em 29 abr 2014.

WOLBER, David *et al*. ***Appinventor, Create your own Android Apps***. O'Really. Canadá, 2011.