UNIVERSIDADE ESTADUAL PAULISTA – UNESP
FACULDADE DE CIÊNCIAS E TECNOLOGIA

O USO DO CELULAR COMO FERRAMENTA DE ENSINO EM FÍSICA

EDUARDO MORENI LOPES

Presidente Prudente – SP
2015

EDUARDO MORENI LOPES

O USO DO CELULAR COMO FERRAMENTA DE ENSINO EM FÍSICA

Trabalho de Conclusão de Curso apresentado ao Departamento de Física, Química e Biologia da Faculdade de Ciências e Tecnologia de Presidente Prudente da UNESP, para obtenção do título de Licenciado em Física, sob a orientação do Professor Doutor Angel Fidel Vilche Pena

Presidente Prudente – SP
2015

EDUARDO MORENI LOPES

O USO DO CELULAR COMO FERRAMENTA DE ENSINO EM FÍSICA

MONOGRAFIA REFERENTE À OBTENÇÃO DO TÍTULO DE LICENCIADO EM FÍSICA

_____
Prof. Dr. Angel Fidel Vilche Pena
Departamento de Física, Química e Biologia – FCT/UNESP

_____
Profª. Drª. Agda Eunice de Souza
Departamento de Física, Química e Biologia – FCT/UNESP

_____
Prof. Dr. Moacir Pereira de Souza Filho
Departamento de Física, Química e Biologia – FCT/UNESP

Presidente Prudente – SP
2015

AGRADECIMENTOS

Agradeço primeiramente aos meus pais pelo incentivo e ajuda nos momentos mais difíceis.

Ao meu irmão pelo apoio e por me ajudar em certas decisões.

A Rose pelos momentos felizes juntos nessa vida louca.

Agradeço aos grandes professores da Física, Química e Matemática pelos conhecimentos adquiridos e por serem uma fonte de inspiração.

Aos amigos de Paraguaçu pelo companheirismo e os bons momentos de alegria juntos.

Resumo

A informática já faz parte do cotidiano de todas as pessoas. Na educação começou com as Teleaulas, ensino a distância e até videoconferências. Nos projetos de informatização das escolas se propõe que cada aluno tenha na sua carteira, um notebook que vai fazer parte da sala de aulas, podendo servir como ferramenta de apoio ao professor da disciplina. Com uma velocidade muito maior, a tecnologia de comunicação via celular cresceu rapidamente passando do simples ligar a receber mensagens há 10 anos atrás, e agora para os Tablets, Iphones, Androides e outros, com aplicativos e capacidades iguais aos de um notebook. Os avanços nos aplicativos de comunicação via celular, como SMS, *WhatsApp* e outros, tornou-se um problema na própria escola, pela presença constante destes na sala de aulas, interferindo no andamento da disciplina. Numa versão simplificada de "senão pode vencer, una-se ao inimigo", estamos propondo utilizar o celular como ferramenta de apoio ao Professor de Física da escola através de aplicativos direcionados a temas que podem ser trabalhados em sala de aulas. A vantagem deste trabalho é o duplo investimento do aluno para atingir o resultado, por uma parte vai conhecer ferramentas de programação que sempre estiveram a seu alcance e ele próprio aplicá-las para resolver algum problema da disciplina de Física. Inicialmente preparamos dois pequenos programas para o estudo do movimento uniforme e movimento acelerado, que serão expostos com mais detalhes.

Palavras-chave: Educação. Professor de Física. Aplicativos de Celular. Ferramentas de Programação.

Abstract

The computer is already part of everyday life for all people. In education began with teleclasses, distance learning and even video conferencing. School computerization projects is proposed that each student has in his wallet, a notebook that will be part of the classroom, serving as a support tool for the subject teacher. With a much higher speed, the cell via communication technology has grown rapidly moving from simple to connect to receive messages from 10 years ago, and now for Tablets, iPhones, Androids and other applications and with capacities equal to those of a notebook. Advances in communication through mobile applications such as SMS, WhatsApp and others, has become a problem at school, the constant presence of these in the classroom, which interferes with the progress of the discipline. In a simplified version of "if you can not win, join the enemy", we are proposing to use the phone as a tool to support the school's professor of physics through applications to the issues that can be worked in the classroom . The advantage of this study is twice the student investment to achieve the result, on the one hand will know programming tools that have always been within our grasp and apply them to solve a problem of the discipline of physics. Initially prepared two small programs for the study of uniform motion, accelerated motion, which will be treated in more detail.

Keywords: Education. Physics teacher. Mobile applications. Programming tools.

SUMÁRIO

1. Introdução

Nos dias de hoje, vemos que o avanço tecnológico dos computadores e celulares estão cada vez mais acelerados. A cada ano surgem novos celulares, novos computadores, e assim também aparecem novos estudos sobre a utilização destes equipamentos na Educação.

Existem diversos programas educacionais das Secretarias Estaduais e Municipais da Educação e do Ministério da Educação (MEC), que propõem levar internet e computadores para as escolas públicas brasileiras, entretanto, não é apenas introduzir estes equipamentos que a educação irá melhorar (LOPES et al., 2009). A questão é que na maioria dos casos, o professor não tem uma formação adequada e tempo suficiente para se inserir nesse processo de informatização da escola, e acontece que o professor não sabe lidar com essas novas tecnologias ou muito menos fazer um bom uso delas em suas aulas (LOPES et al., 2009 *apud* UNESCO, 2008).

Ainda existem muitas visões distorcidas sobre uso da Informática na Educação. Segundo Diorgenes Grzesiuk, existem argumentos para os céticos e os otimistas. Para os céticos os argumentos mais comuns é a falta de materiais básicos para a escola, então como pensar em adquirir computadores. Outro argumento diz respeito a desumanização da educação gerada pela máquina, que diz que a máquina irá substituir o professor. Agora, para os otimistas os argumentos são levados pelo modismo, se o computador é um meio didático, então deve-se adquirir um computador. Se o computador já faz parte da vida dos alunos, é preciso ensinar os alunos a lidarem com essa tecnologia (GRZESIUK, 2008).

Através da implementação dessas novas tecnologias na educação, os estudantes devem ser alimentados pelo querer saber, por querer buscar conhecimentos para si, resolver problemas diversos e aprender independentemente. A utilização do computador na educação, com o uso de diversos *softwares* educacionais em diferentes áreas, comprova que esta ferramenta pode ser muito útil para o processo de ensino-aprendizado (GRZESIUK, 2008).

2. Justificativa

Há algum tempo, a importância dos professores no processo de ensino-aprendizagem dos alunos e dos seus trabalhos em salas de aulas vem sendo prejudicados por um pequeno aparelho: o celular. A grande maioria dos estudantes, principalmente nas cidades grandes, não conseguem desconectarem dos celulares e tablets um minuto sequer. E isso não é diferente nas escolas públicas e particulares em Presidente Prudente.

O professor tem enfrentado uma batalha difícil em sala de aula, que é de conseguir atrair a atenção dos alunos em suas aulas e despertar um maior interesse pela aula. Para os estudiosos e professores, o uso dos celulares em salas de aula é polêmico, pelo fato que muitos acham um absurdo os estudantes usaram durante a aula, e outros já começaram a usar o aparelho sem que comprometa o aprendizado em aula. (PEREIRA, 2014)

Muitos estudiosos afirmam que não se deve dispensar o uso dessas novas tecnologias, pelo contrário, utilizá-las cada vez mais nas escolas, mas como meios que auxiliam os professores e não para substituí-los. Para a UNESCO, a instituição estimula o acolhimento dessas tecnologias nas disciplinas, pois pode "permitir a aprendizagem a qualquer hora, em qualquer lugar", "minimizar a interrupção em aulas de conflito e desastre" e "criar uma ponte entre a educação formal e a não formal". (ALVIM, 2014)

A cada dia que passa surgem novos aplicativos para celular. Muitos deles podem ser encontrados na loja do *Google Play* via *website* e muitos são gratuitos. Se os aplicativos fossem manipulados apropriadamente por professores e alunos, poderiam ser de grande utilidade no processo ensino-aprendizado. Alguns exemplos destes aplicativos são:

- ✓ Aplicativo que mostra a tabela periódica e as características de cada elemento químico, que poderia ser muito útil em uma aula de Química. Conhecido como *Periodic Table*.
- ✓ Também existe um aplicativo que pode construir diversos gráficos e resolver diversas equações em Matemática, muito útil para uma aula de Matemática. Conhecido como *Mathematics.*
- ✓ Um aplicativo que mostra as equações de Físicas que poderia facilitar o aluno quando estivessem fazendo uma tarefa de casa, por exemplo. Conhecido como Fórmulas de

Física *Free* 1.3.

Fica evidente que o uso do celular é multidisciplinar, o aluno pode ter todos esses aplicativos em seu celular e então usá-los a qualquer hora e em qualquer lugar. O celular traz mais pontos positivos do que negativos. Basta o professor saber gerenciar suas aulas e propor atividades aos alunos usando estes pequenos aparelhos.

Mas ainda existem maus usos destes pequenos aparelhos. Muitos professores reclamam que os alunos usam os celulares apenas para se comunicarem entre si, através do *WhatsApp*, e das redes sociais como *Facebook* e *Twitter*. E ainda usam o celular através destas ferramentas para passarem "cola" uns para os outros. Por isso que é preciso que o professor esteja preparado para essas situações. Uma ideia bem inovadora é criar nesses aplicativos de conversas, como WhatsApp e outros, grupos de estudos para debaterem os conteúdos vistos em sala de aula, é uma ideia inovadora e muitos dizem que dão resultados (PEREIRA, 2014).

"Nessa linha, o professor vai usar o inimigo (o celular) como um aliado para atrair a atenção dos alunos e obter êxito em seus objetivos e atingir resultados satisfatórios" (PEREIRA, 2014, p.1).

3. Objetivos

3.1. Objetivos Gerais

- Conhecer algumas técnicas de programação, que poderão ser utilizadas em Sala de Aulas.

- Desenvolver um aplicativo (simulador) para celular relacionado com os movimentos uniforme ou acelerado, conceitos que foram estudados nas aulas de Física.

3.2. Objetivos Específicos

Para atingir o objetivo principal, alguns objetivos específicos são requeridos, são eles:

- Desenvolver alguns conceitos básicos de informática que serão necessários para a ferramenta de construção do aplicativo no celular.

- Mostrar aos alunos, que eles podem, sem muito esforço, construir um aplicativo (simulador) para celular.

- Ensinar aos alunos a importância da utilização dessas novas tecnologias para o uso na Educação.

4. Metodologia Desenvolvida

Utilizaremos neste trabalho uma ferramenta que nos permitirá construir um aplicativo para celular, e que esse aplicativo será um simulador de Física. Para isso vamos utilizar o *MIT App Inventor* que é uma ferramenta de programação baseada em blocos, que permite, que todos, mesmos os novatos possam iniciar na programação e criar aplicativos funcionais para dispositivos Android. A ferramenta foi desenvolvida pelo *Google*, e agora é mantida pelo Instituto Tecnológico de Massachusetts (MIT).

A escolha desta ferramenta foi simplesmente pelo fato, de que não é necessário fazer nenhuma instalação no computador, basta ter acesso a internet e uma conta de *e-mail* no *Google* (gmail). Mas irá existir um grande desafio, a ferramenta está apenas disponível em Inglês, mas isso servirá de apoio para que os alunos comecem a estudar uma língua estrangeira.

Vamos separar as nossas atividades. Primeiro vamos aprender acessar a ferramenta e visualizar seus componentes e acessórios para a construção dos aplicativos propostos. Em seguida veremos como fazer um simulador baseado no movimento uniforme, e logo depois veremos um exemplo de simulador para o movimento acelerado. As figuras que serão postas estarão com setas vermelhas e os nomes de cada item importante em vermelho para uma melhor visualização.

4.1. Acessando e Conhecendo a Ferramenta: *MIT App Inventor*

Em primeiro lugar, temos que acessar o site, o endereço eletrônico é dado por: http://appinventor.mit.edu/explore/, e então entraremos no site, como indicado na **Figura 1**.

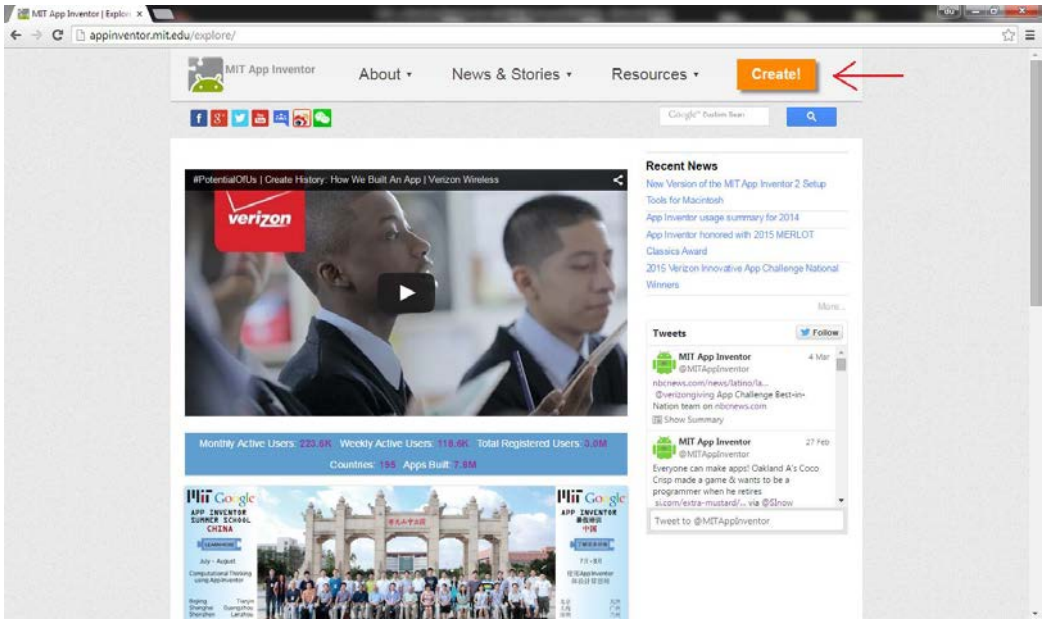

Figura 1 - Site de acesso do *MIT App Inventor*.

Clicaremos no botão *Create,* que significa criar. E então aparecerá outro site, como mostra a **Figura 2**. Aqui devemos colocar nosso e-mail e senha. Por isso é necessário ter um cadastro de e-mail no *Google*.

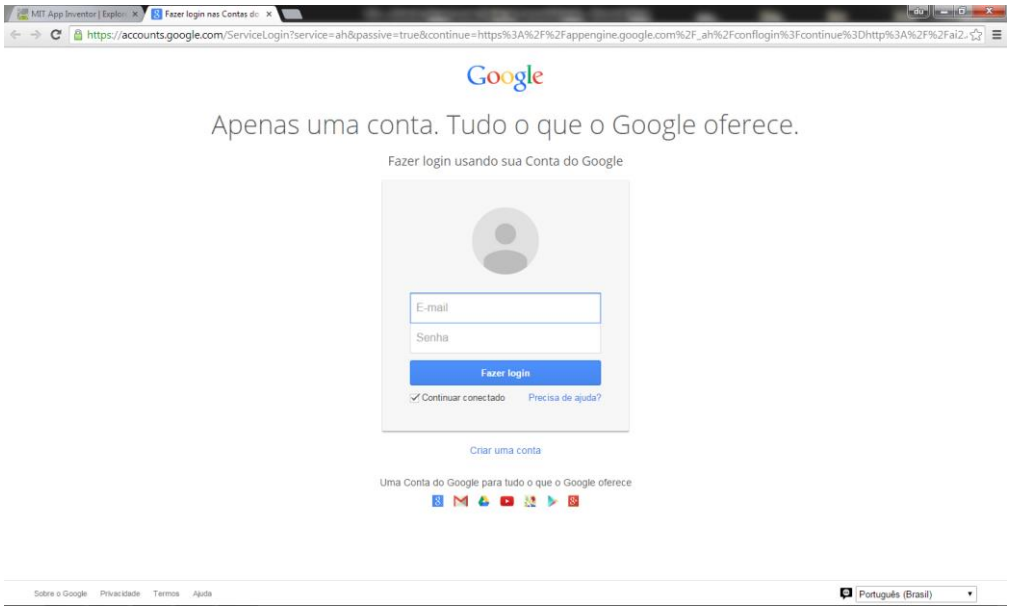

Figura 2 - Site para efetuarmos *login* no *MIT App Inventor.*

Pondo o e-mail e senha, efetuaremos a conexão. E então aparecerá, como está indicado na **Figura 3**.

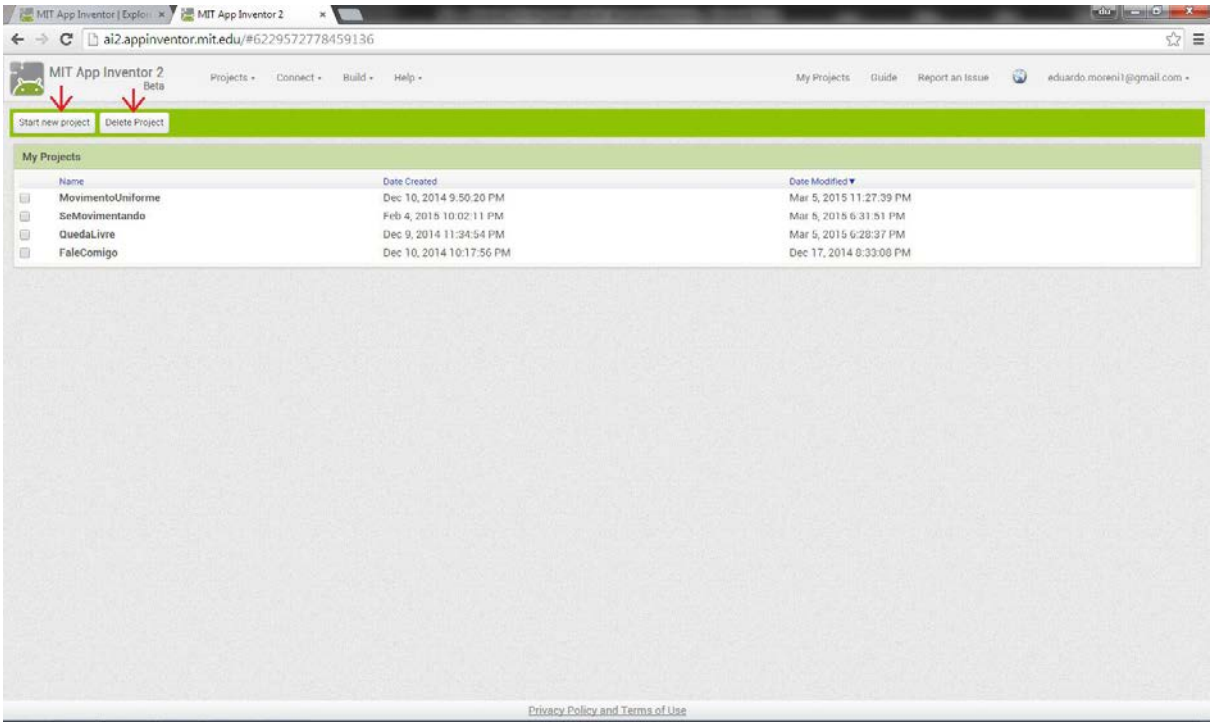

Figura 3 - Acesso ao *MIT App Inventor*.

Agora podemos ver os meus projetos feitos. Mas, no caso de alguém que está iniciando, não irá aparecer nenhum projeto. Então para iniciar um projeto, basta clicar em *Start new project* que significa começar um novo projeto. Também é possível deletar projetos, basta selecioná-los e clicar no botão *Delete.*

Quando começarmos um novo projeto irá pedir um nome para designar tal projeto, basta por um nome e clicar em *Ok* que, em seguida, abrirá esta página: segue a **Figura 4**.

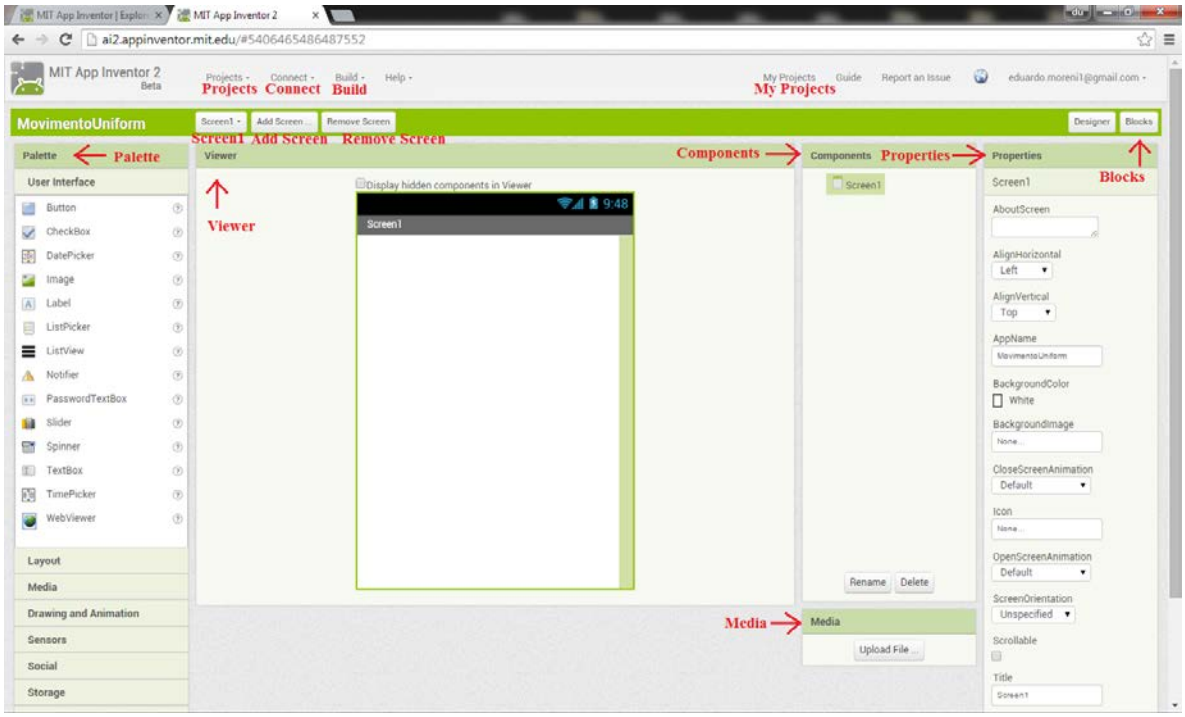

Figura 4 - Página de *interface* e *designer* de um projeto (aplicativo).

A partir da **Figura 4** vamos fazer uma análise geral das ferramentas presentes nesta página. Perceba que usamos setas e os nomes em vermelhos para ser mais fácil de identificar cada item citado.

Em *Viewer*, que significa espectador, no centro da tela temos um modelo de celular onde irá mostrar tudo o que está sendo construído como: fundo de tela, botões inseridos, alinhamento de textos, ícones, e muito mais. Basta arrastarmos os componentes que estão em *Palette* para irmos modelando o aplicativo aos poucos.

Em *Palette*, que significa paleta (acessório de pintor), no canto esquerdo da imagem é onde está localizado tudo o que será necessário para montar o aplicativo, desde botões à sensores de tempo.

No canto direito, em *Properties* que traduzindo significa Propriedades, como o próprio nome já diz, são as Propriedades de todos os componentes que serão inseridos no modelo de celular. Através das propriedades é possível alinhar textos, alterar cor de fundo, inserir um papel de parede, criar um ícone para o aplicativo, dentre outras funções.

Em *Components* é possível verificar como estão dispostos os componentes inseridos em *Viewer* e também renomeá-los.

Em *Media*, abaixo de *Components*, é possível inserirmos diversas imagens retiradas do computador para serem postas no aplicativo.

Um pouco acima de *Viewer*, temos os botões *Screen1*, *Add Screen* e *Remove Screen* que significam respectivamente Tela1, Adicionar Tela e Remover Tela. Quando clicamos em adicionar tela, estamos criando mais uma tela para o aplicativo e então é possível acessar essas telas a qualquer momento, e também é possível removê-las quando quisermos.

No topo da página, em *Projects* podemos acessar nossos projetos, começar um novo projeto, salvá-lo, importar um projeto de aplicativo do computador, exportá-lo para o computador, e outras funções.

Em cima também, clicando em *Connect* é possível criarmos uma rápida simulação do aplicativo, servindo como base para vermos se há algum problema.

Logo do lado, em *Build* que significa construir, é possível construirmos o nosso aplicativo. Clicando logo para a primeira opção *App (provide QR code for .apk)* irá aparecer mais tarde um código, num estilo códigos de barras e através do próprio celular usaremos um aplicativo do MIT que escaneia o código, para instalarmos o nosso aplicativo construído em nosso celular.

Os outros botões restantes não são tão importantes. Mas existe um ainda que não mencionamos, é o botão *Blocks*, que significa blocos, fica logo abaixo do e-mail no canto extremo direito. É nele onde vamos escrever toda a linha de programação, ou seja, é o cérebro do aplicativo.

Ao clicar no botão *Blocks,* uma nova página irá se abrir, ilustrada na **Figura 5**. É aqui onde vamos montar toda a linha de programação. No lado esquerdo em *Blocks* existem infinidades de processos que podemos criar. Ao clicarmos em algum dos botões *Control, Logic, Math, Text...* Abrirá uma janela ao lado, e assim podemos selecionar qual bloco vamos ligar uns aos outros ou como vamos relacionar os componentes postos no celular com tais lógicas de programação. Para construirmos a programação, basta arrastarmos os blocos ou componentes até o centro onde se localiza em *Viewer*. Se precisarmos remover algum bloco basta arrastá-lo até a lixeira que fica no canto inferior direito.

Importante notar, logo abaixo temos um botão *Show Warnings*, que significa mostrar avisos, ele nos diz se existe algum erro existente na programação. Quando clicamos em *Designer* voltaremos para a página de modelamento da estrutura do nosso aplicativo, como vimos na **Figura 4**.

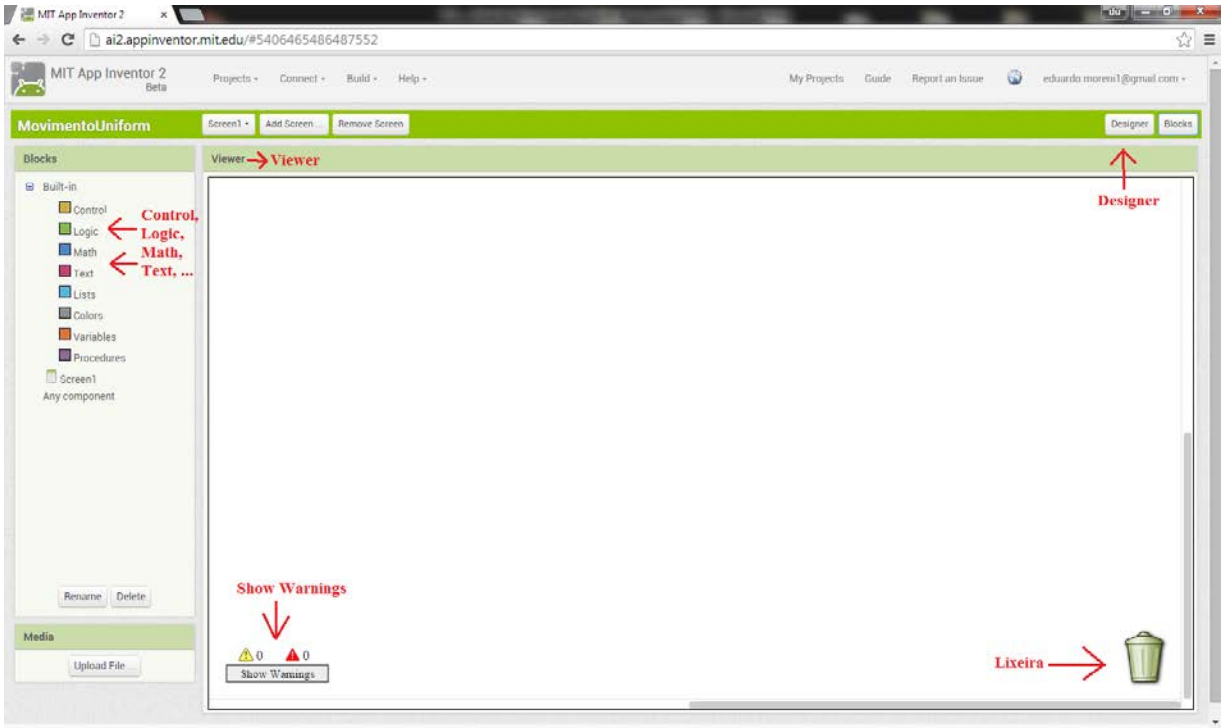

Figura 5 – Página *Blocks* onde fazemos a programação do aplicativo (simulador).

Agora vamos para a nossa primeira tarefa. Desenvolver um aplicativo (simulador) de Física para celular, que simule o Movimento Uniforme.

4.2. Desenvolvendo um simulador de Física para celular: Movimento Uniforme

Agora, iremos construir um simulador (aplicativo para celular) de Física, que simule um fenômeno do Movimento Uniforme. A ideia proposta é muito simples, vamos fazer uma pequena disputa de dois carrinhos de corrida, um dos carrinhos terá o dobro da velocidade do outro. Lembrando que essas velocidades não se alteram, pois estamos pensando em criar um Movimento Uniforme, ou seja, velocidades constantes.

O primeiro passo é acessar o *site* e criar um novo projeto, como vimos anteriormente. Basta por um nome ao projeto, que será o nome do simulador (aplicativo) no celular.

Será exposto aqui um simulador já feito para ser mais fácil de trabalhar, e que possa ser um guia de exemplo. Lembrando, que na criação podemos inserir qualquer tipo de figura, fundos de tela, e outras coisas. A modelagem fica a critério de cada um.

A **Figura 6** ilustra como é a tela inicial do nosso aplicativo (simulador).

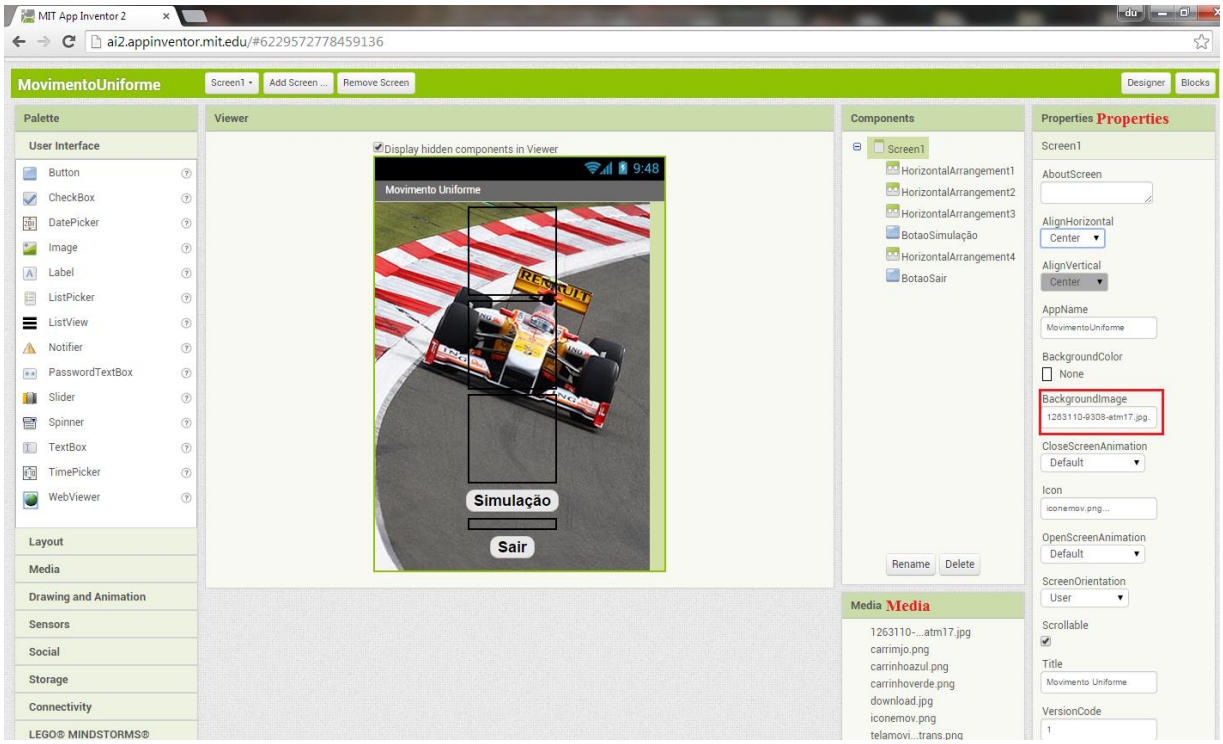

Figura 6 - Tela inicial do aplicativo vista pelo *site*.

Já podemos notar que, na tela principal vamos ter que inserir dois botões e quatro arranjos de tabelas. O fundo foi tirado de uma imagem de busca através da *internet*, ou seja, a escolha do fundo é livre. Lembrando que, para inserir um fundo, é preciso ter a foto salva no computador, inserir em *Media* como tínhamos visto e ir em *BackgroundImage* que significa imagem de fundo marcado na **Figura 6**, que está em *Properties* à direita e então inserir a imagem desejada.

Vamos separar em etapas a construção do simulador:

Etapa 1: Como vemos há um alinhamento central nos botões. Para isso, vamos em *AlignHorizontal* em *Properties* e selecionamos *Center* marcado em vermelho pela **Figura 7**. Pronto estará tudo centralizado, botões e textos.

Etapa 2: Clicando em *Layout* em *Palette*, como podemos ver na **Figura 7,** vamos clicar e arrastar um *HorizontalArrangement*, que significa arranjo horizontal, até o centro. Vamos inserir três destes arranjos, lembrando que não vamos alterar nenhuma propriedade deles. Vamos deixar exatamente deste tamanho e centralizado na tela.

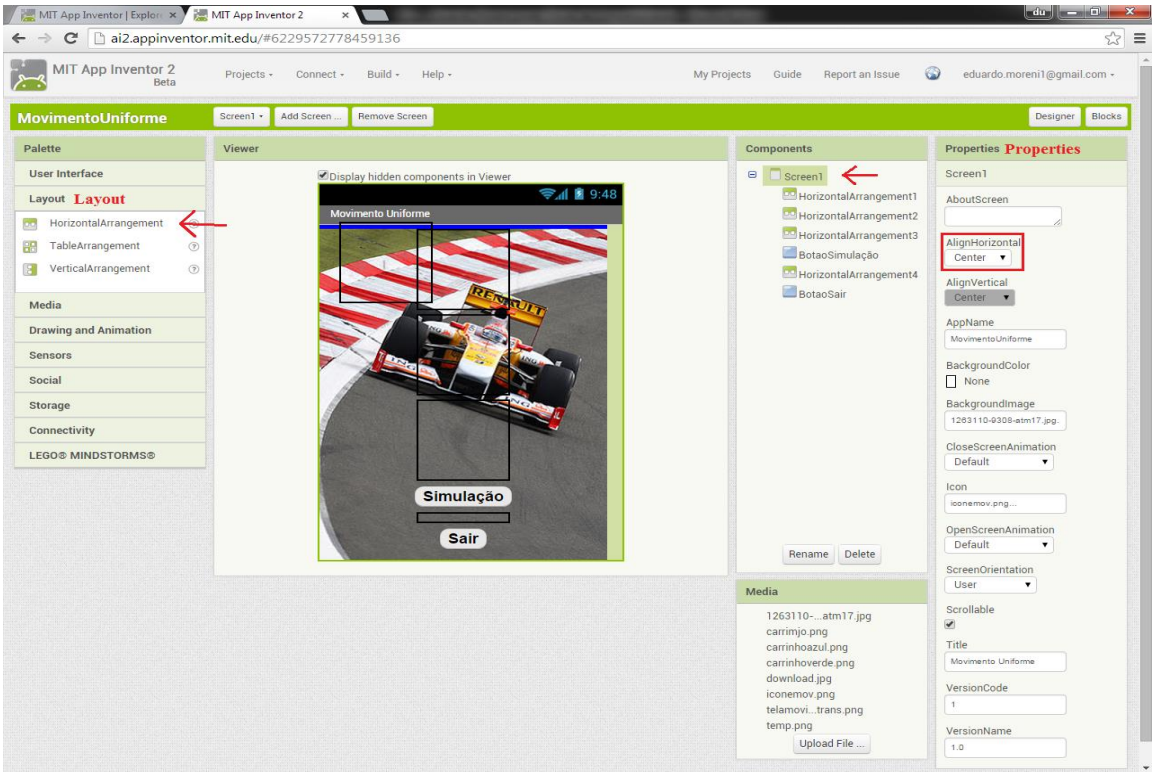

Figura 7 - Clicando em *Layout* vamos arrastar três *HorizontalArrangement* até o centro.

Etapa 3: Vamos inserir o primeiro botão. Basta clicarmos em *User interface* em *Palette,* como podemos ver na **Figura 8**, e clicar em *Button* e arrastá-lo até onde estaria o botão Simulação. O *Button* aparecerá em *Components* e clicando em *Rename* será possível alterar o nome, alteraremos para BotaoSimulação marcado em vermelho. Em *Properties*, na **Figura 8,** em *Text* podemos alterar como vai ser o texto no botão em nosso aplicativo. Vamos escrever Simulação, ou seja, a ideia é que quando clicarmos nesse botão abrirá uma nova tela, que será a tela da simulação do evento. Mas, ainda não o programamos para isso, logo veremos como fazer. Podemos deixar os botões de diversas formas, em forma retangular ou arredondados, para mudar bastar ir em *Shape,* o utilizado aqui foi o *rounded* que seria o arredondado. Pode-se alterar também a fonte do texto, por negrito, a cor, e outras funções.

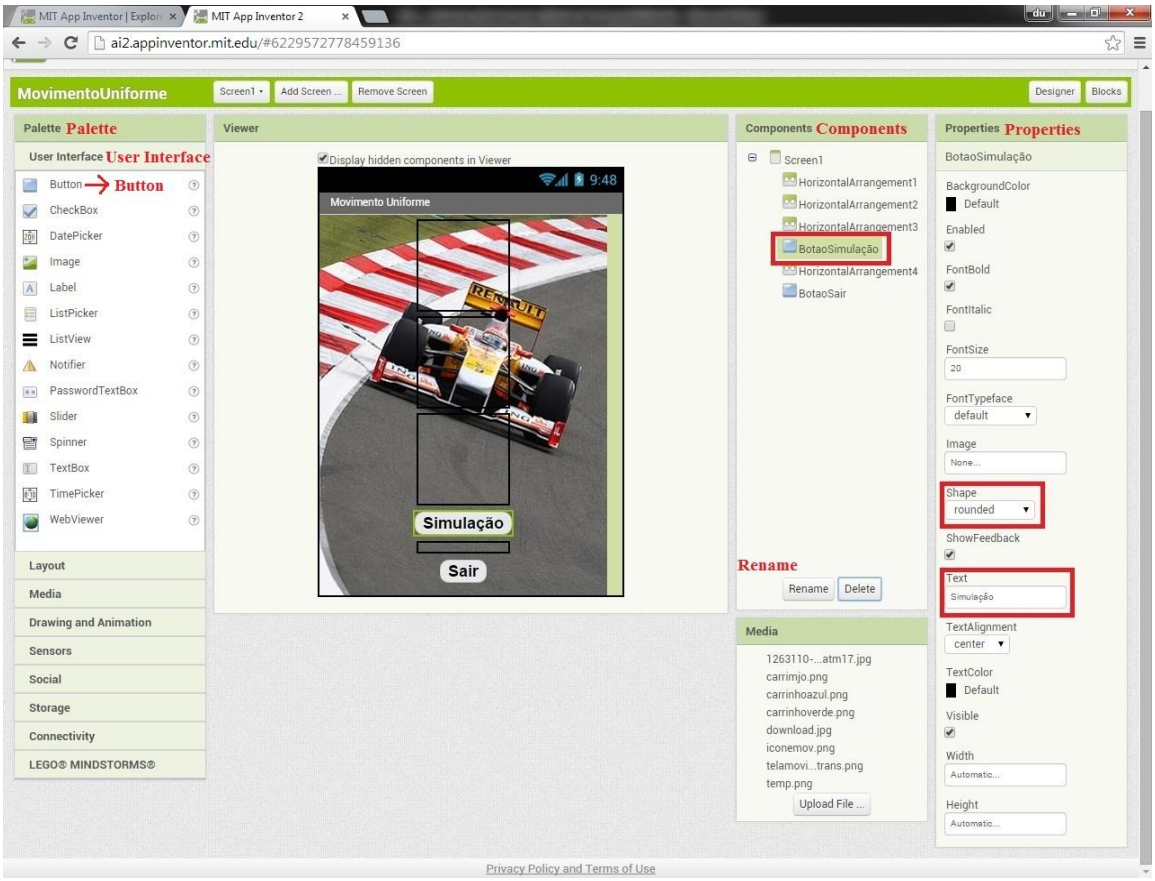

Figura 8 – Inserindo o primeiro botão Simulação no aplicativo.

Etapa 4: Vamos inserir um novo arranjo parecido com o que fizemos na Etapa 2. Lá em *Layout* selecionamos o *HorizontalArrangement*, e arrastamos abaixo do botão Simulação. Em *Properties* alteramos o valor de *Height*, que seria a altura deste arranjo horizontal, para 10 *pixels,* como vemos na **Figura 9.** Estamos fazendo isso, apenas para que os botões Simulação e Sair não fiquem tão próximo um do outro.

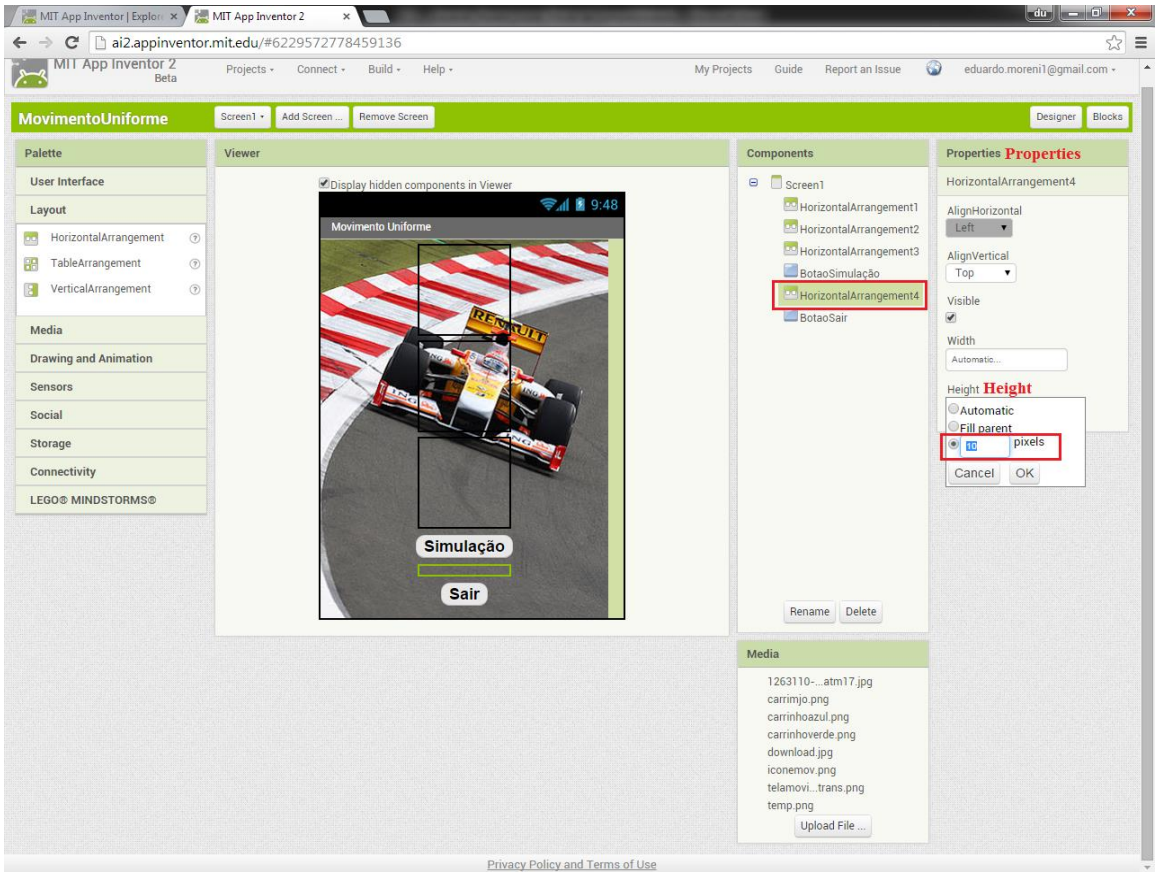

Figura 9 - Alterando a altura entre os botões.

Etapa 5: Por fim, vamos inserir o último botão. Basta fazermos como foi feito na Etapa 3. Colocando o botão logo abaixo do último arranjo e então renomeá-lo em *Components* e por o nome do botão de Sair. A ideia é que quando clicarmos neste botão iremos finalizar o aplicativo, mas ainda precisamos programá-los.

Etapa 6: Aqui iremos criar mais uma tela para o nosso aplicativo. Clicamos em *Add screen* acima de *Viewer*, confira **Figura 9**, e colocamos o nome da tela de "Simulacao" desse jeito mesmo, sem acentuação. Pronto, mais uma tela foi criada. Depois vamos trabalhar nessa nova tela, onde ocorrerá a simulação do fenômeno.

Etapa 7: Iremos fazer a programação para os dois botões inseridos no aplicativo. Clicando em *Blocks* como vimos anteriormente, iremos acessar a área de programação do aplicativo. A **Figura 10** ilustra como deve ficar a programação. O primeiro bloco nos diz: *when* BotaoSimulação .*Click do open another screen screenName* "Simulacao", que significa em termos de programação: quando BotaoSimulação for clicado abrirá uma nova tela e essa tela se chama "Simulacao". Ou seja, iremos programar para que quando clicarmos no botão chamado Simulação abrirá uma outra tela, onde ocorrerá a simulação.

O segundo bloco está escrito: *when* BotaoSair *.Click do close application*, que significa: quando BotaoSair for clicado fechar o aplicativo. Como podemos ver, programamos para que quando clicarmos no botão Sair do aplicativo, ele irá finalizar o mesmo.

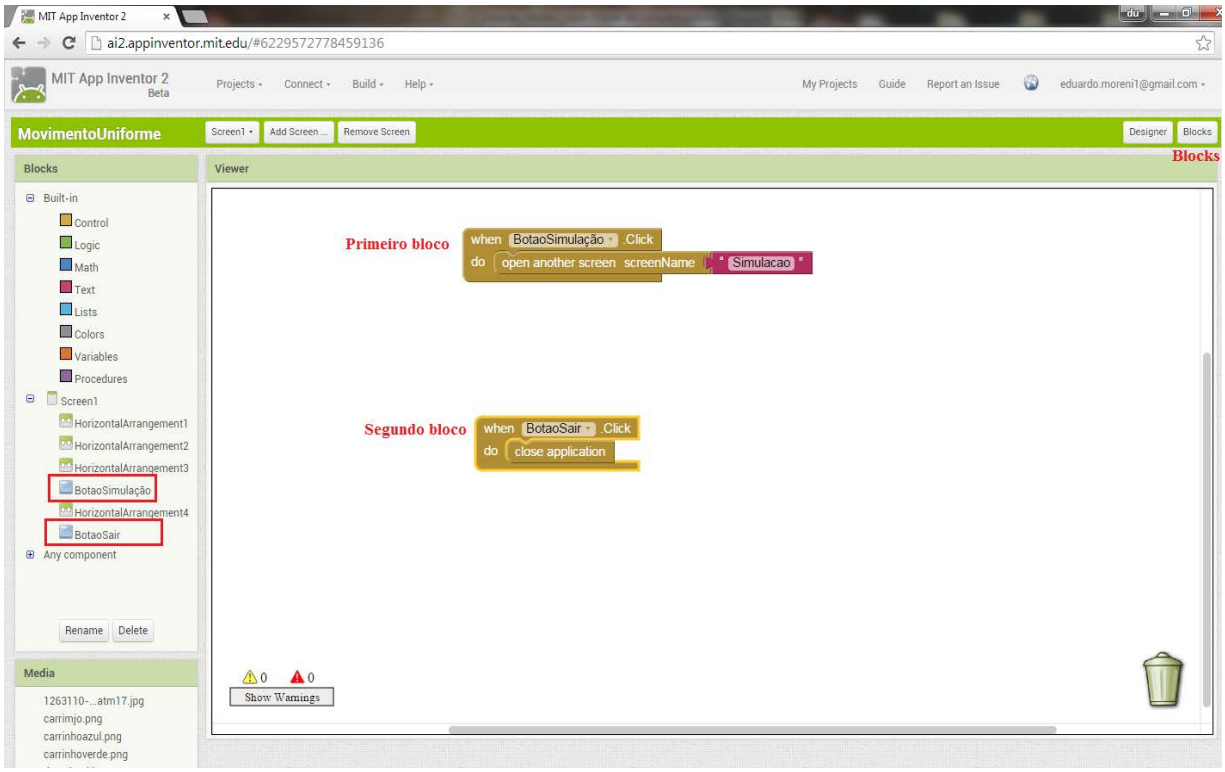

Figura 10 - Programação dos botões inseridos.

Para criarmos estes dois blocos de programação devemos seguir a **Figura 11**. Em Blocks clicamos primeiro em BotaoSimulação que está indicado em vermelho. Abrirá uma janela com os possíveis blocos que podemos usar. Vamos arrastar o primeiro bloco, indicado em vermelho, até o centro como está na figura. Iremos fazer o mesmo para o BotaoSair indicado em vermelho e arrastá-lo para o centro.

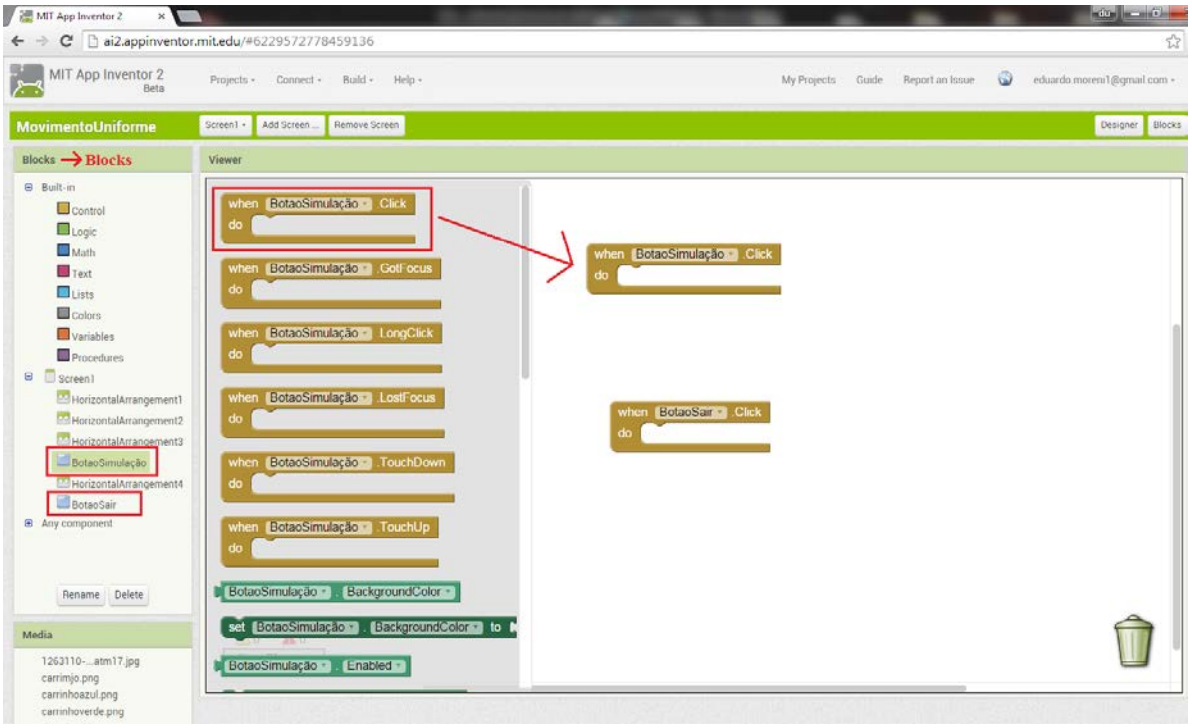

Figura 11 - Criando os blocos de programação.

Temos que completar a programação, ou seja, quando clicarmos nos botões queremos que eles executam um comando. Para isso, clicamos em *Control* e selecionamos mais dois blocos que vamos ligar. Veja **Figura 12**. Como vimos no começo desta etapa, nós queremos que quando clicarmos no botão de simulação ele irá abrir uma nova tela e essa tela se chama "Simulacao" e quando clicarmos no botão sair ele irá fechar o aplicativo. Estes são os blocos que fazem essa programação, podemos encontrá-los em *Control* e para fazermos a sua conexões basta arrastarmos até onde está indicado pela seta na **Figura 12**.

Mas ainda não estaremos completando totalmente a programação, precisamos inserir mais um bloco no BotaoSimulação pois precisamos identificar qual tela irá abrir quando clicarmos no botão. Podemos acompanhar o processo na **Figura 13**. Clicamos em *Text* e arrastamos o primeiro bloco até encaixarmos onde está indicado pela seta vermelha. Escrevemos nesta caixa de texto, apenas Simulacao sem espaços ou acentuação, senão poderão ocorrer problemas futuros na programação.

A programação ficou de acordo como abordamos no começo desta etapa. Vamos agora construir a tela de simulação do aplicativo.

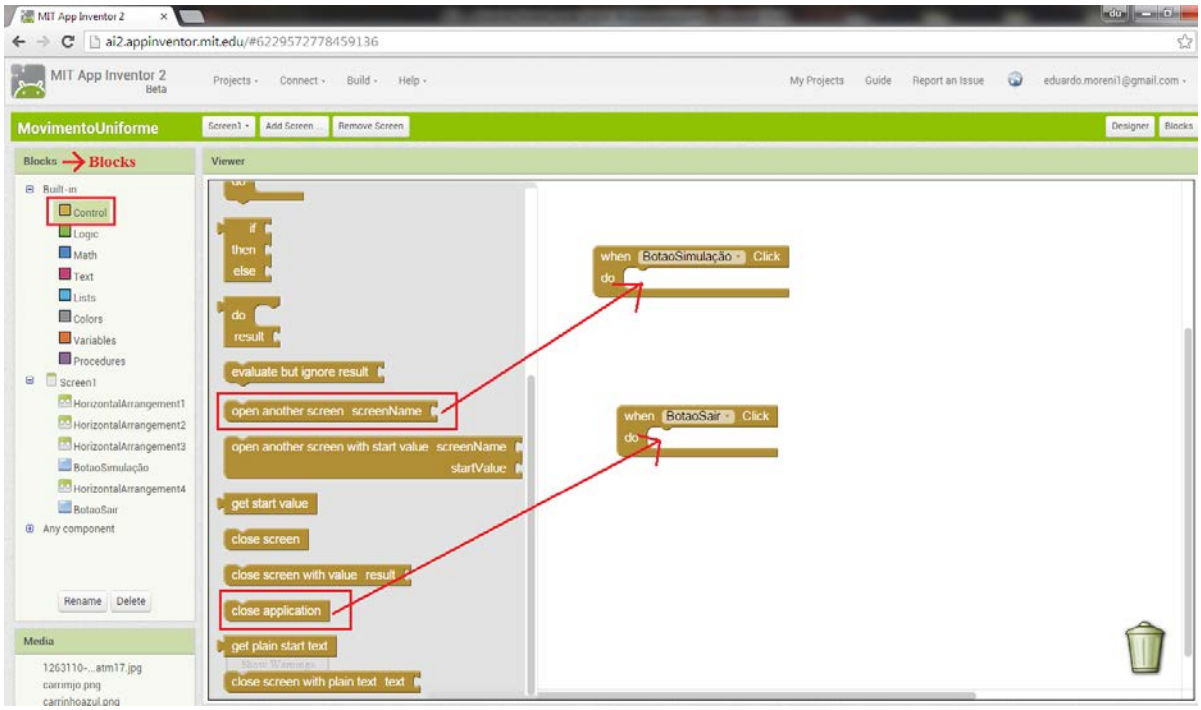

Figura 12 - Clicando em *Control* aparecerá os blocos que vamos utilizar para a programação.

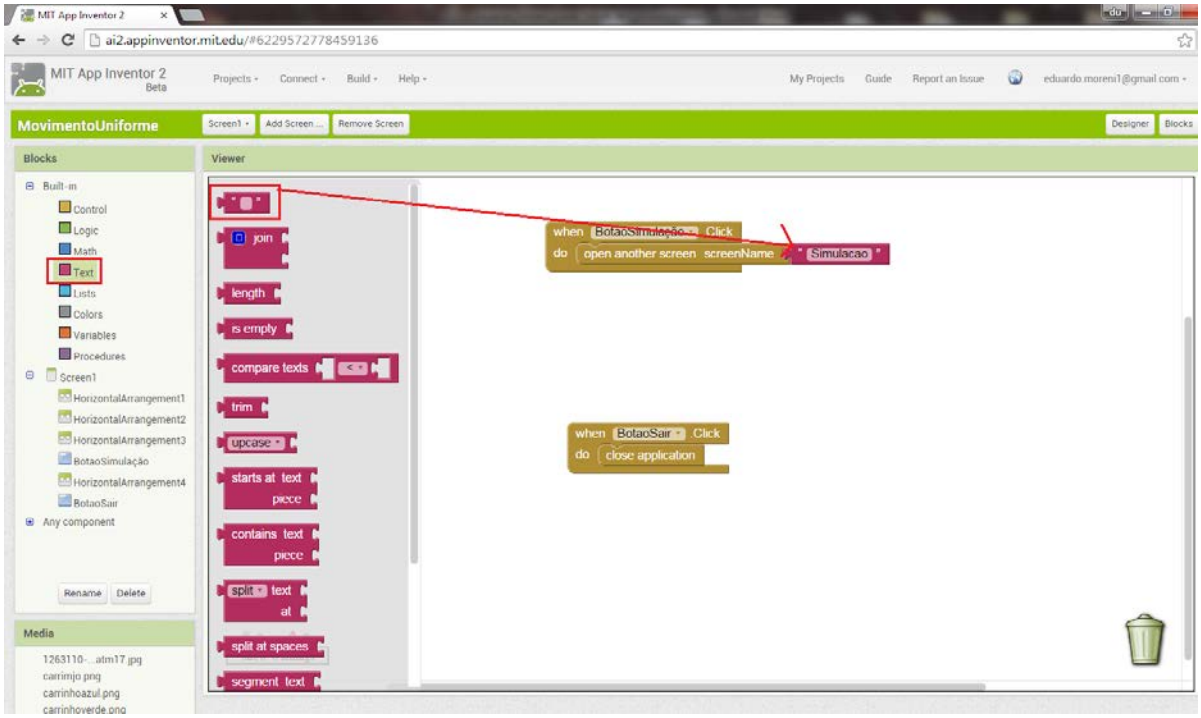

Figura 13 – Clicando em *Text* e arranjando o bloco certo podemos completar a programação da primeira tela.

Etapa 8: Vamos começar agora a trabalhar com a parte da simulação.Voltaremos para o *designer* do aplicativo. Primeiro clicaremos em *Screen1* e selecionamos a outra tela que criamos "Simulacao". A **Figura 14** mostra o modelo da tela "Simulacao" feito neste trabalho.

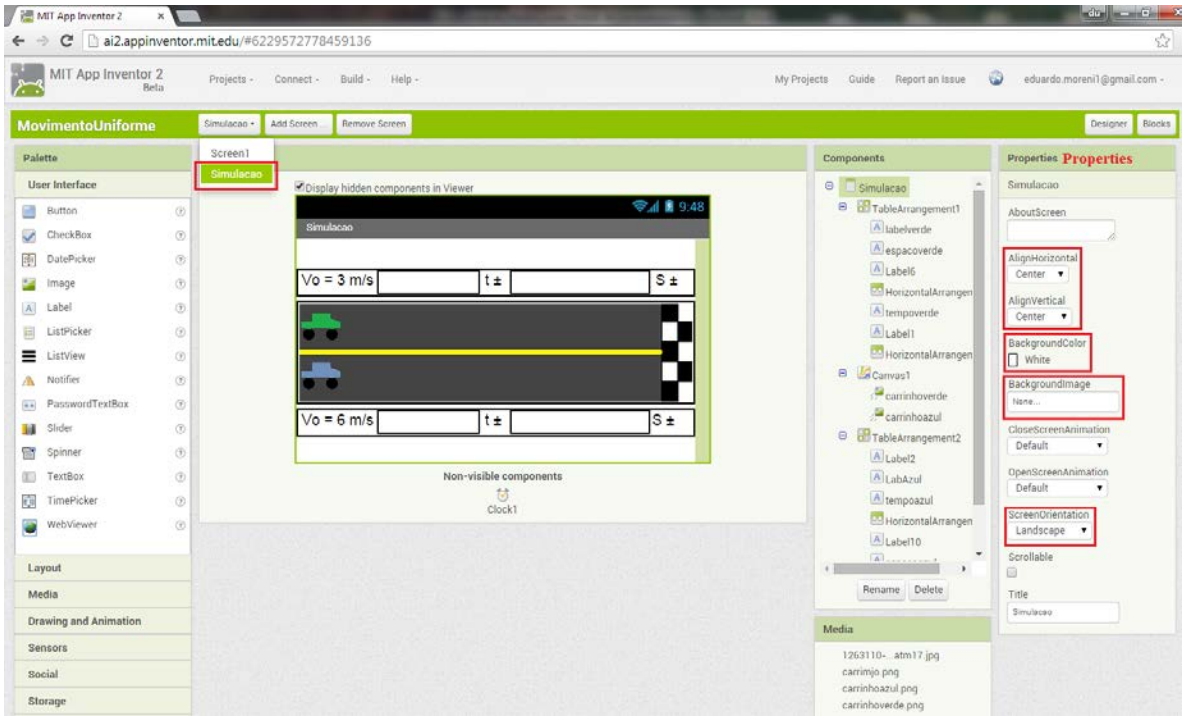

Figura 14 - Tela de simulação do aplicativo.

Primeiramente, precisamos inserir algumas propriedades desta nova tela "Simulacao". Podemos ver apartir da **Figura 14**, que em *Properties* os itens marcados em vermelho iremos alterá-los. Em *AlignHorizontal* e *AlignVertical*, que significam respectivamente alinhamento horizontal e alinhamento vertical, vamos selecionar *Center,* ou seja centralizados. Em *BackgroundColor*, que significa cor de fundo, deixaremos na cor branca. Em *BackgroundImage*, que significa imagem de fundo, não colocaremos nenhuma imagem. E por último, em *ScreenOrientation*, que significa orientação da tela, selecionaremos *Landscape* que significa paisagem, ou seja, a orientação da tela será em modo paisagem.

Etapa 9: Nesta etapa, vamos fazer a construção da primeira janela de notificação onde vão ficar os dados da velocidade do carrinho verde, o tempo e o espaço percorrido. Podemos ver na figura anterior, as notificações vão ficar acima e abaixo da pista, sendo uma para o carrinho verde e outra pro azul. Ou seja, o processo de construção das notificações serve para ambos.

Estando na página de *Designer*, vamos em *Layout* e selecionamos o *TableArrangement* e o arrastamos até o centro, como pode ser visto na **Figura 15**. Será necessário alterarmos algumas propriedades deste arranjo de tabelas. As propriedades podem ser acessadas à direita, onde estão marcadas em vermelho lá em *Properties*. Em *Columns* e *Rows,* que significam Colunas e Linhas, inserimos os valores de dez colunas e uma linha. Em *Width* e *Height,* que significam Largura e Altura respectivamente, colocaremos o valor da Largura em "*Fill*

parent...” e da Altura em trinta pixels. Estamos fazendo isso para que possamos ajustar o arranjo de tabelas perfeitamente no modelo que estamos trabalhando. O tamanho do arranjo de tabela tem que estar de acordo com o mostrado na **Figura 15**.

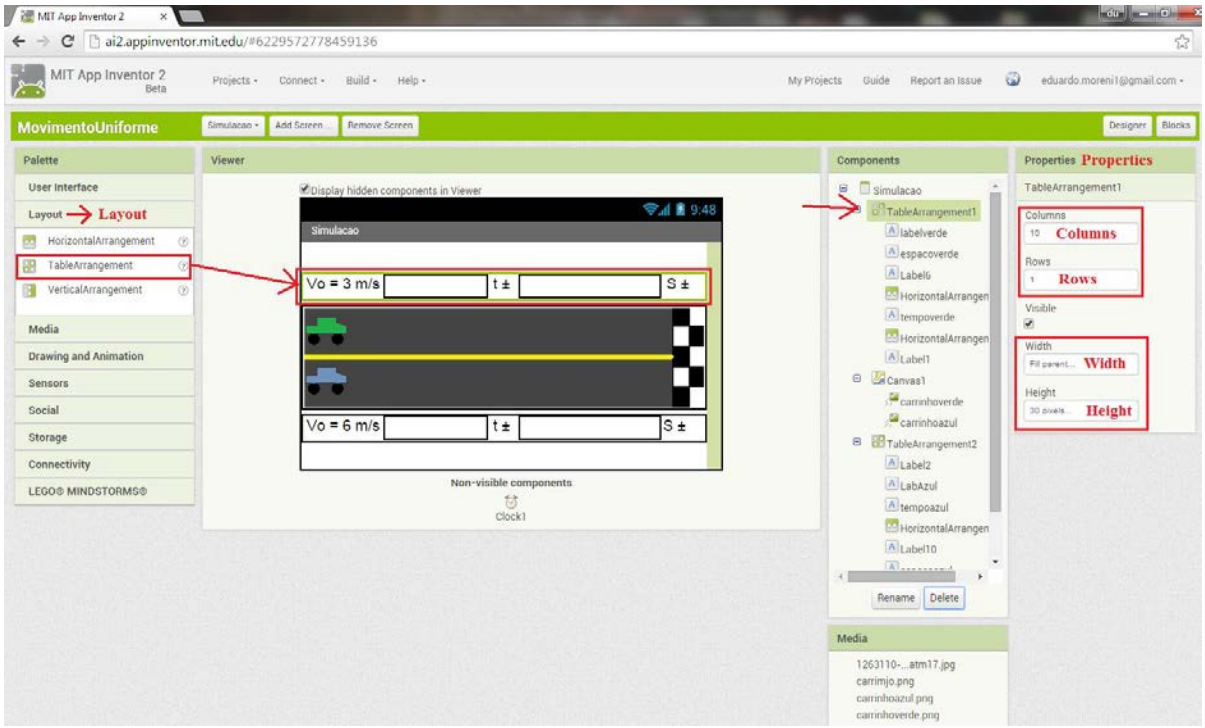

Figura 15 - Criando a primeira tabela de notificação.

Etapa 10: Feito a etapa anterior, vamos agora inserir os *Labels*, que significa Etiqueta, nas tabelas de arranjo. Eles são usados para inserirmos algum pedaço de texto ou programá-los para aparecer certas informações importantes. São essas etiquetas que vão mostrar a velocidade dos carrinhos, o tempo e a distância percorrida. Como podemos ver, na **Figura 15**, as velocidade do carrinho verde e azul são mostradas com seus respectivos valores. Lembrando que vamos construir como está indicado na figura.

Para inserirmos vamos em *User interface*, e selecionamos os *Labels* e arrastamos até o arranjo de tabela como está indicado na **Figura 16**.

Vamos inserir cinco etiquetas (*Label*), como podemos ver na figura estão enumerados em cima do arranjo de tabela em *Viewer*. Em *Properties*, como já sabemos, podemos alterar algumas propriedades, que no caso dos *Labels*(etiquetas) são: tamanho do texto, inserir texto, alinhamento do texto, cor do texto e o tamanho. Na primeira etiqueta indicada na figura, vamos inserir o valor da velocidade dada ao carrinho, por exemplo, pro carrinho verde ficou: “V_0 = 3 m/s” que nos diz que a velocidade é de três metros por segundo. Na segunda vamos

deixar apenas o texto: “t ±”, se notarmos bem na **Figura 16**, logo ao lado haverá uma terceira etiqueta (*Label*) que vamos deixar sem texto algum e em *Components* vamos renomear o nome desta etiqueta para “tempoverde”. Estamos fazendo isso para facilitar na hora da programação desta etiqueta que, na hora da simulação, vai nos mostrar o progresso do tempo em segundos. Na quarta etiqueta como podemos ver na figura, vamos deixar o texto como “S ±”, similarmente às etiquetas anteriores, a quinta etiqueta, logo ao lado, ficará com o texto em branco, e iremos renomeá-la para “espacoverde”. Esta última etiqueta vai nos mostrar os valores das distâncias quando os carrinhos percorrerem a pista. Entenderemos melhor quando iniciarmos a programação.

Lembrando que, para a construção das etiquetas para o carrinho azul, seguimos o mesmo raciocínio.

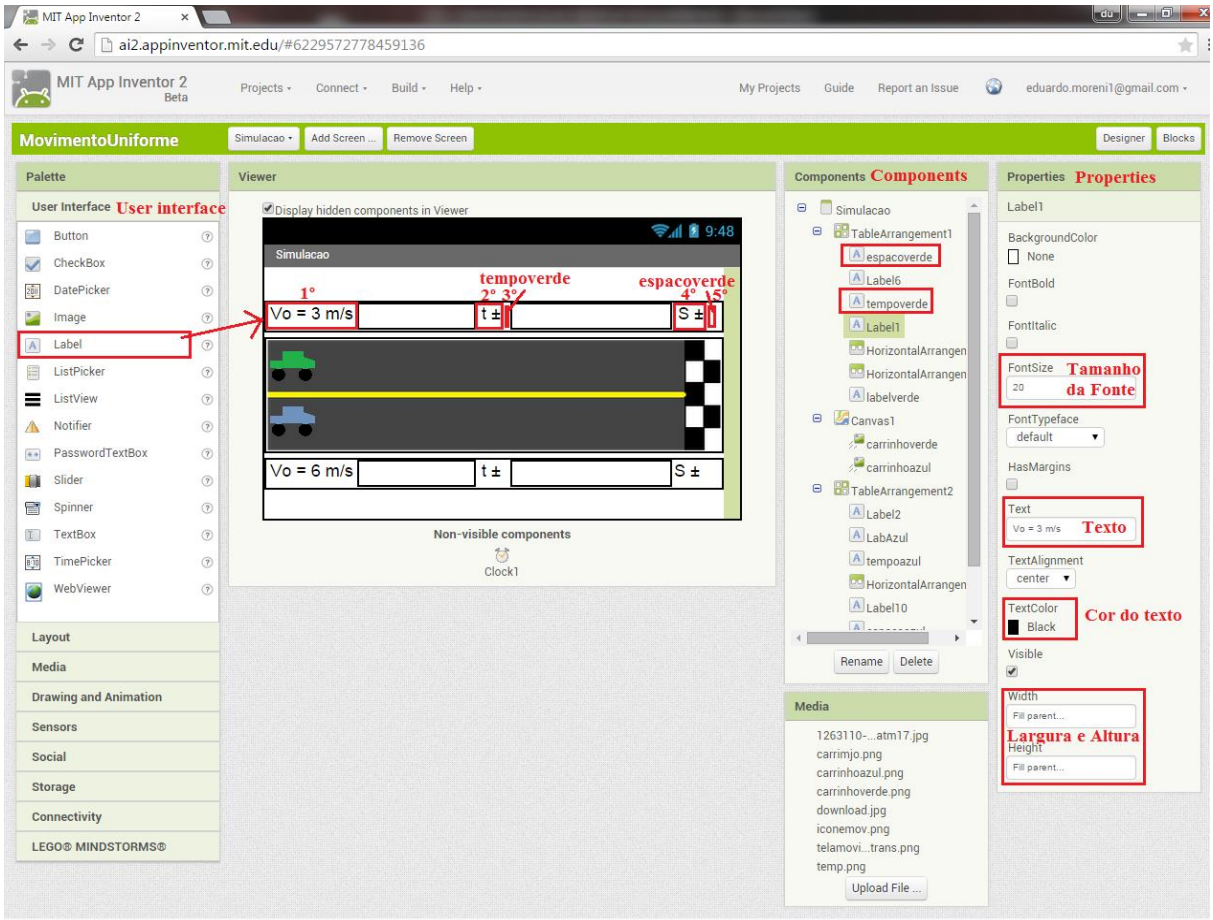

Figura 16 - Inserindo os Labels para obtermos as notificações.

Ainda precisamos ajustar os *Labels* nas tabelas de arranjo. Para isso vamos em *Layout* e arrastamos quatro *HorizontalArrangement* (arranjo horizontal) como podemos ver na **Figura 17**. Ajustamos suas propriedades de tal forma, para obtermos uma separação entre as

etiquetas. Pela **Figura 17**, podemos ver que há uma enumeração dos arranjos horizontais. Todos os quatros arranjos vão ter alinhamentos horizontais e verticais centralizados, como podemos vermos em *Properties*. O primeiro e terceiro arranjo terá largura igual a 120 *pixels* e altura igual a 25 *pixels*, já o segundo e quarto arranjo terá largura igual a 165 *pixels* e altura igual a 25 *pixels*. Os valores inseridos podem ser outros, isso vai depender do modelamento que se quer.

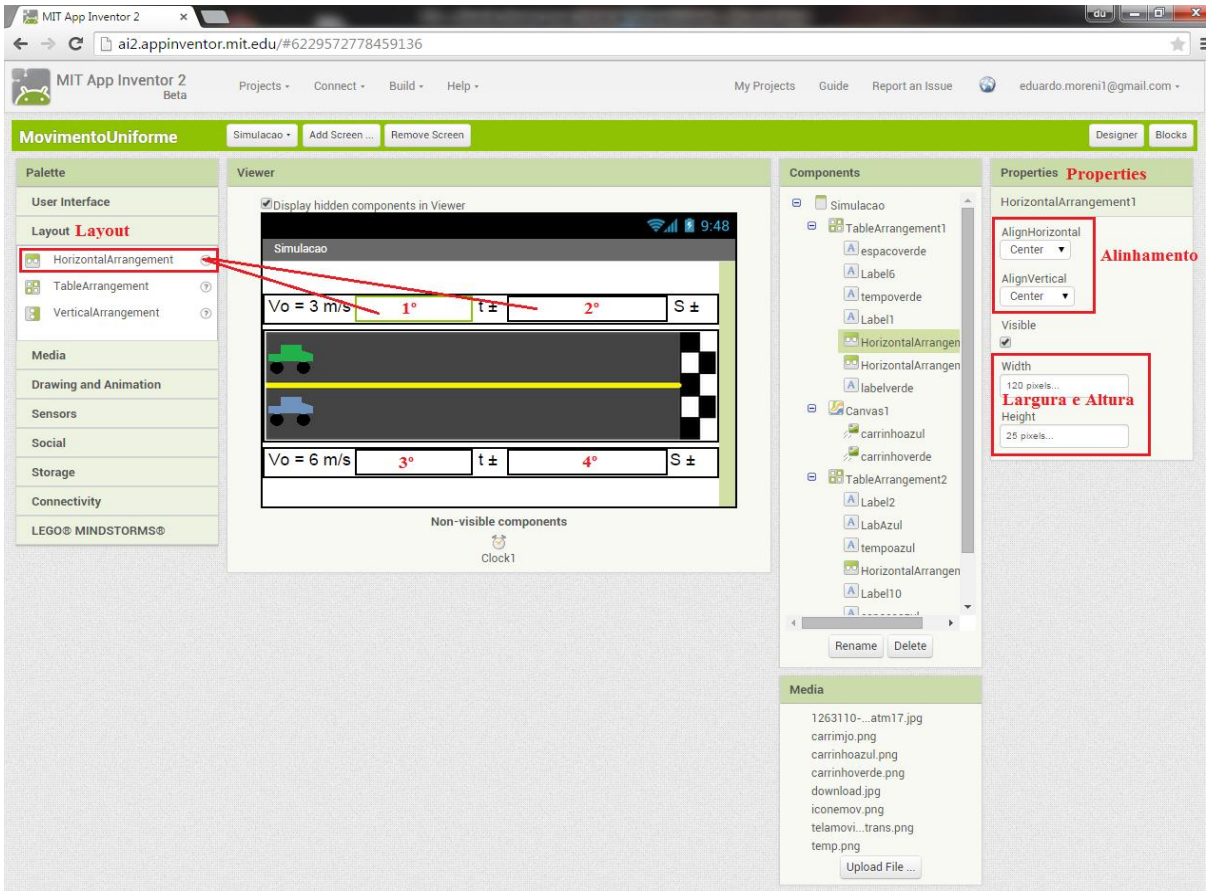

Figura 17 - Inserirmos os arranjos horizontais para separarmos os *Labels*.

Etapa 11: Vamos fazer a animação agora, para isso vamos continuar na página de *Designer*. Vamos clicar em *Drawing and Animation*, é neste lugar que consiste todas as ferramentas para a animação. O que iremos fazer é construir uma pista de corrida e dois carrinhos que vão se movimentar com velocidades específicas.

A **Figura 18** ilustra quais ferramentas usaremos. Quando clicarmos em *Drawing and Animation* iremos clicar e arrastar um *Canvas* até o centro, como pode ser visto na imagem. *Canvas*, que significa Lona, é usado para criar animações de imagens bidimensionais. A pista de corrida utilizada neste simulador foi feita no *Paint* que é um *software* de desenhos gráficos.

Para inserirmos uma imagem, como, por exemplo, a pista de corrida, temos que passar o arquivo da imagem salvo no computador para *Media* clicando em "*Upload File...*" e então em *Properties* do *Canvas* selecionamos a imagem em *BackgroundImage*, que seria a imagem de fundo do *Canvas*, ou seja, estamos querendo que o fundo seja uma pista de corrida, como podemos ver na **Figura 18**. Ainda é preciso fazer alguns ajustes na largura e na altura de onde acontecerá a simulação, que é justamente no *Canvas*.

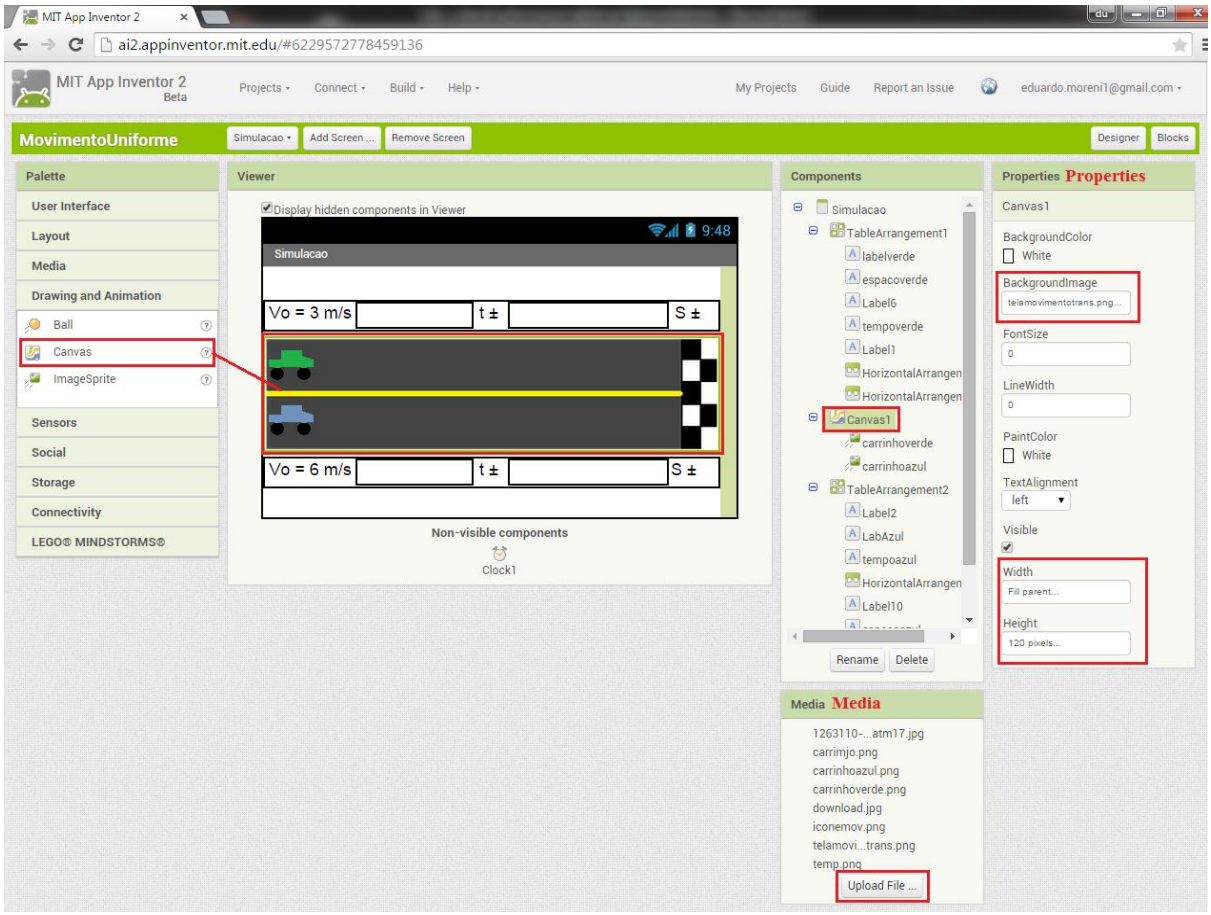

Figura 18 - Construção da animação.

Ainda em *Properties* lá em *Width* e *Height*, que significa Largura e Altura respectivamente, alteraremos o valor para "*Fill parent...*" da Largura, ou seja, o *Canvas* terá um tamanho proporcional em toda extensão (horizontalmente) do celular, e o valor da altura (vertical) será de 120 pixels. Lembrando que os valores podem ser outros, isso vai depender de como se quer criar ou modelar as imagens de simulação no celular.

Vamos inserir agora os dois carrinhos. Considerando que temos uma pista igual ao da **Figura 18**. Continuaremos na página de *Designer*, clicando em *Drawing and Animation* vamos arrastar dois *ImageSprite* onde estão localizados os dois carrinhos, o verde e o azul, como

podemos ver na **Figura 19**. O *ImageSprite* só pode ser posto quando existir um *Canvas* e ele nos permitirá a criação dos carrinhos.

Os carrinhos aqui foram feitos no *Paint* também, mas pode ser postos qualquer carrinho, qualquer outra figura que queira ilustar um evento de corrida.

As propriedades do *ImageSprite* ou melhor do carrinho verde pode ser vista à direita da **Figura 19**. Nas propriedades, podemos alterar o intervalo de medida, inserir uma respectiva foto de aparência, alterar o valor da velocidade (horizontal) do carrinho, posicioná-lo em alguma coordenada espacial específica e mudar a largura e altura do carrinho.

Apenas para seguir este modelo, os valores utilizados para o carrinho verde nas propriedades foram: intervalo igual a cem, velocidade igual a três e os tamanhos para ambos os carrinhos ficaram em automático. Já para o carrinho azul as propriedades são as mesmas, alteramos apenas a velocidade, ou seja, ele estará com o dobro da velocidade que é igual a seis.

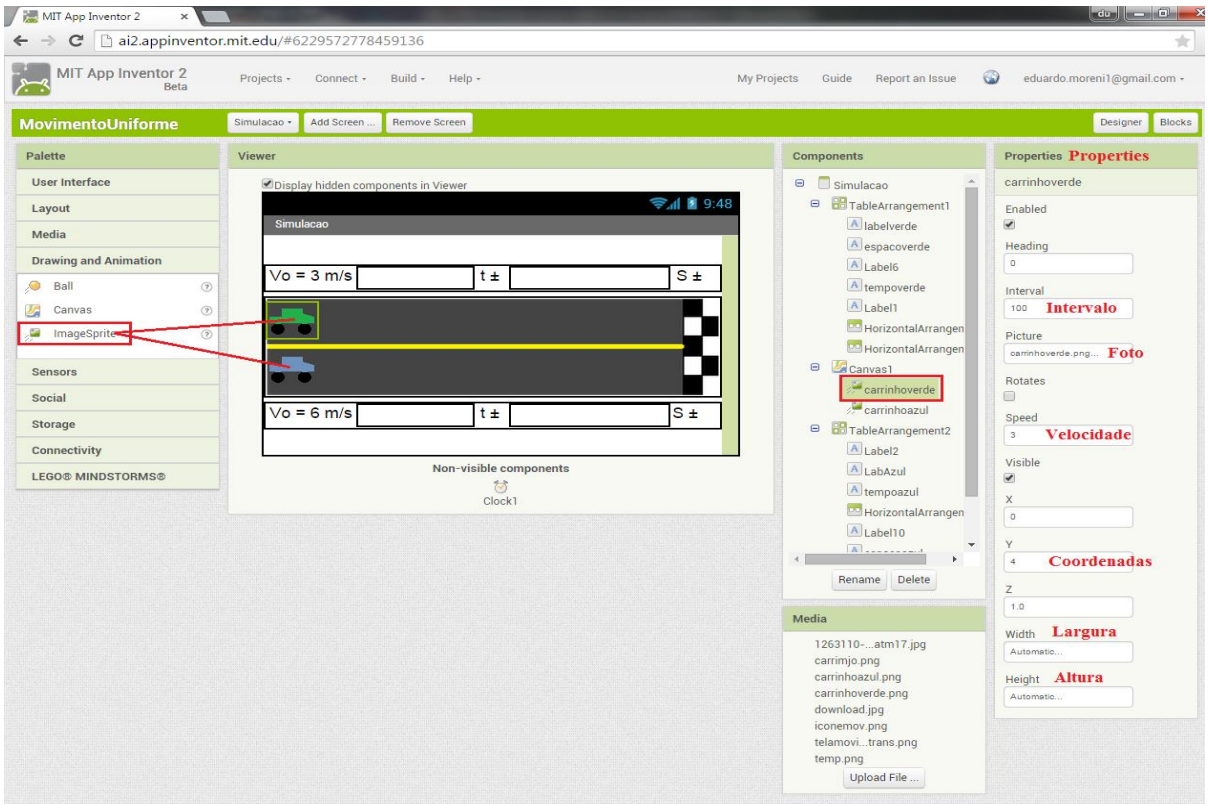

Figura 19 - Inserindo os carrinhos (*ImageSprite*) na pista de corrida.

Como vimos na etapa anterior, a ideia é que quando os carrinhos se moverem vamos saber precisamente sua posição e quem irá chegar no fim da pista primeiro. Mas ainda precisamos programar para que estes dados apareçam em que cada *Label* posta na etapa anterior. Então, vamos programar.

Etapa 12: Antes de começarmos a programar, precisamos inserir um sensor de tempo no simulador. Para fazermos isso continuaremos na página de *Designer* e clicamos em *Sensors*, que significa sensores, e arrastamos o *Clock* até o centro como está indicado na **Figura 20**.

Esse sensor de tempo é um componente não visível do celular e nos dará informações sobre o tempo, como se fosse um cronômetro.

As propriedades do sensor de tempo podem ser vistas na **Figura 20** em *Properties*, na qual deixaremos o intervalo de medida igual a mil.

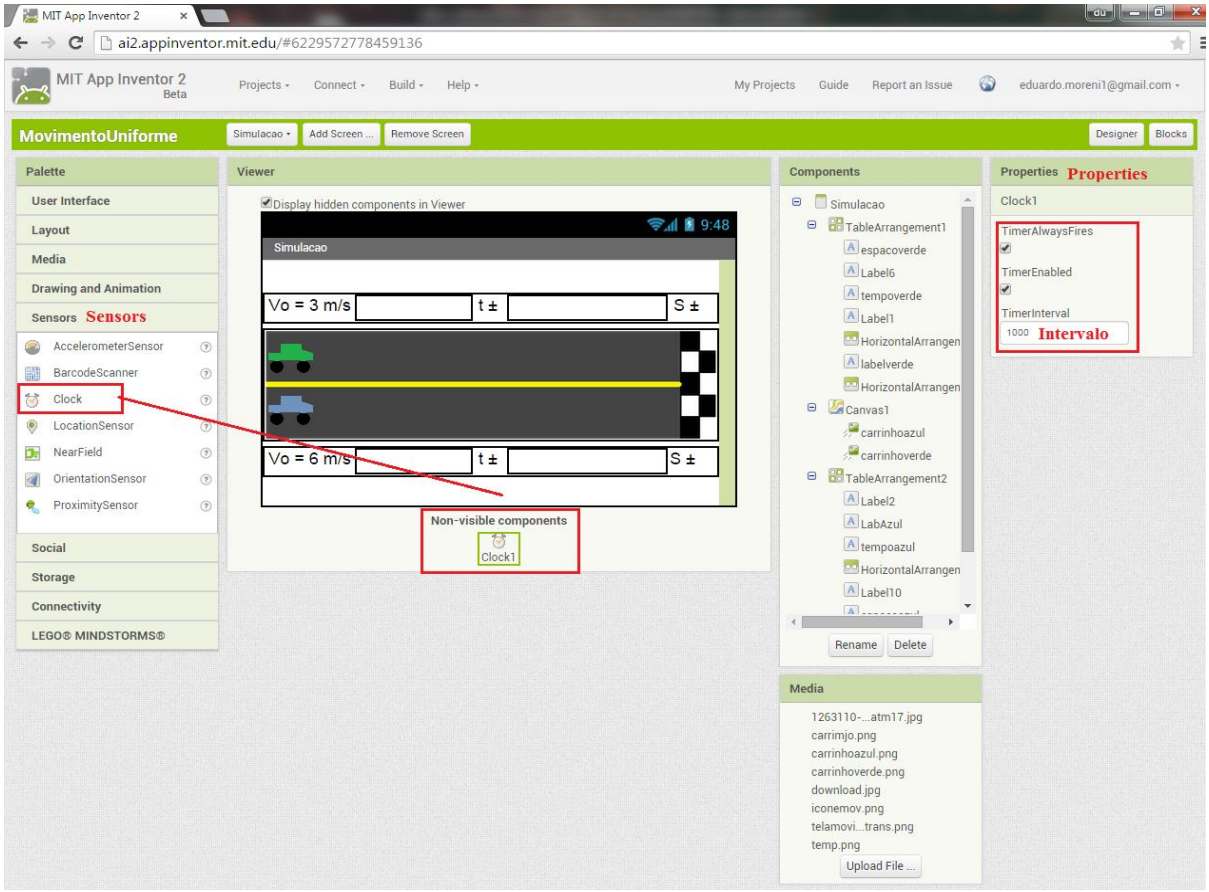

Figura 20 - Inserindo o sensor de tempo no simulador.

Feito tudo isso, vamos entrar na página de programação, clicando em *Blocks* logo acima de *Properties*. Iremos montar a programação para obtermos os valores do tempo decorrido e do espaço que os carrinhos percorreram a cada segundo passado. Queremos que na hora da simulação, os valores apareçam nos *Labels* (etiquetas) de tempo e espaço como presumimos na etapa dez.

A **Figura 21** mostra como ficou a programação feita neste trabalho. Não será exposto aqui em detalhes a criação da programação, faremos uma pequena análise de cada bloco de

programação, pois deixaremos como desafio para que os alunos ou professores montem a própria programação.

Para entendermos melhor foram enumerados os principais blocos de programação e marcados em vermelho como podemos ver na **Figura 21**.

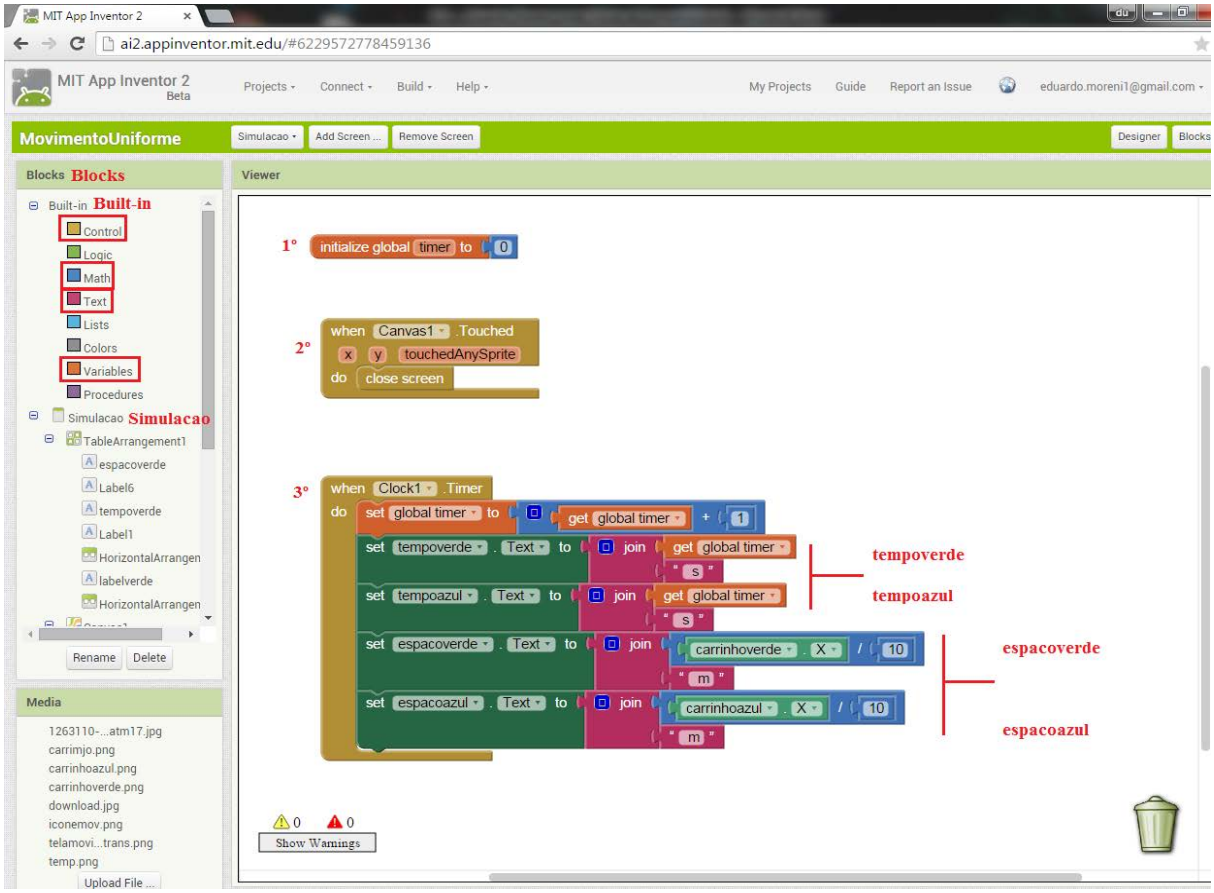

Figura 21 - Página de programação do simulador.

O primeiro bloco nos diz: "*initialize global timer to* zero", estamos criando uma variável global chamada "*timer*", que traduzindo significa cronômetro, e damos o valor inicial de zero. Para criarmos uma variável global, clicamos em *Variables,* que significa variáveis, em *Built-in* e aparecerá uma janela com os blocos para a criação de variáveis. Vamos selecionar o mesmo bloco e arrastar até o centro, alterar o nome da variável para *timer* e ir em *Math* e selecionar o primeiro bloco que corresponde ao valor igual a zero.

Se notarmos, existe um padrão nas cores nos blocos de programação, por exemplo, este primeiro bloco que construímos uma parte dele é alaranjado (para a variável global) e azul (para o valor zero), se olharmos bem há uma indicação em *Variables* alaranjada (um pequeno quadrado) e em *Math* azul escuro, ou seja, onde tudo indica que é ali que encontraremos os

mesmos blocos. Isto facilita muito a reprodução e o entendimento de uma programação que é copiada de algum trabalho por exemplo.

No segundo bloco, está escrito: "*when Canvas1 Touched do close screen*", que traduzindo significa: quando "*Canvas1*" for tocado fechar a tela. Ou seja, quando tocarmos na pista de corrida, lembrando que na pista fica o "Canvas1", a tela irá fechar e voltaremos para a tela inicial do aplicativo (simulador). Para criarmos este bloco de programação, basta seguirmos a **Figura 22**. Em *Blocks* clicaremos em *Canvas1* marcado em vermelho na imagem, e encontraremos o bloco parecido e então arrastaremos até o indicado na **Figura 22**. Para completar a programação, vamos em *Control* e encontraremos o bloco "*close screen"* e então encaixaremos com o bloco "*when Canvas1 Touched do*". Pronto, terminamos o segundo bloco. Vamos analisar o terceiro e último bloco de programação.

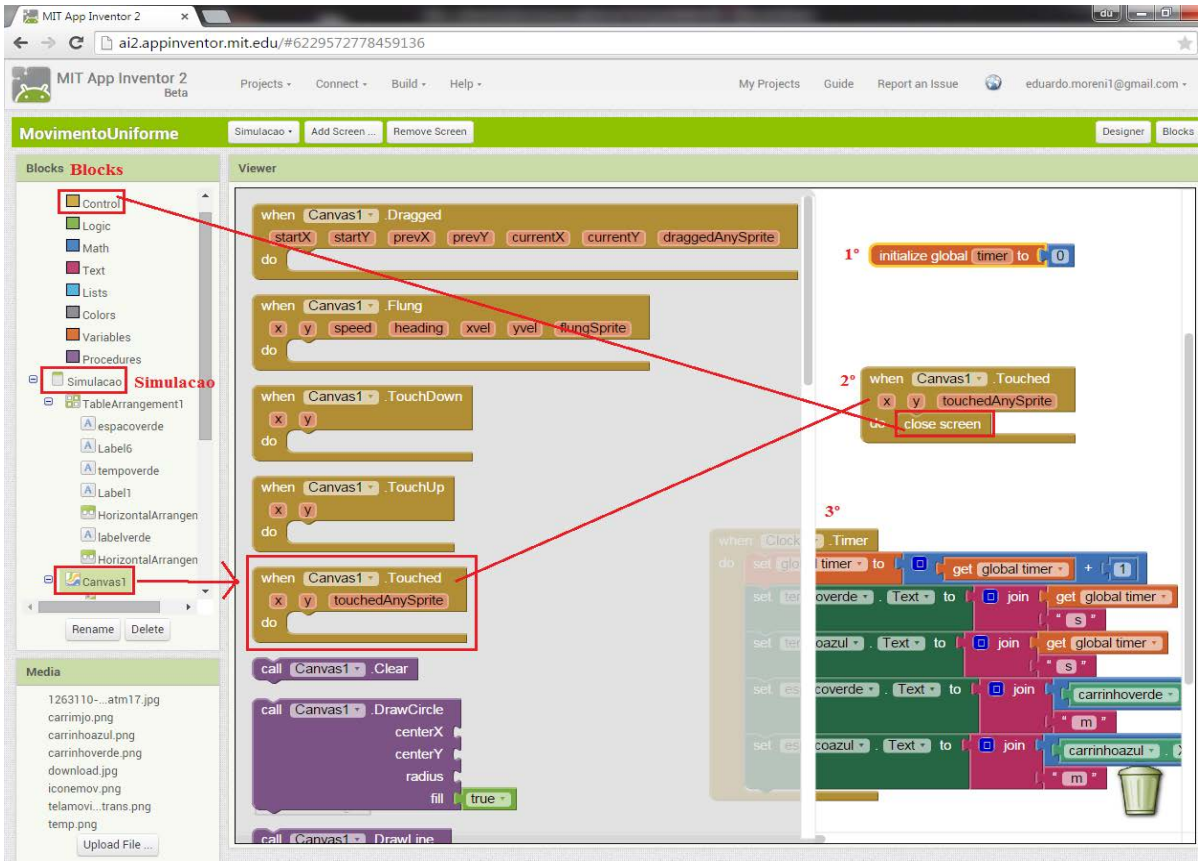

Figura 22 - Criando o segundo bloco da programação.

O terceiro bloco e último bloco, é o mais importante, é dele que vamos obter os valores dos tempos decorridos durante a corrida e do espaço percorrido pelos carrinhos. A **Figura 21** ilustra melhor o terceiro bloco. Como podemos ver na imagem, o bloco principal é o que está escrito: "*when Clock1 .Timer do*", que significa que quando inserirmos o "*Clock1*", ou seja, o

sensor de tempo ele funcionará como um cronômetro. Mais ainda, existirão cinco blocos que vamos inserir dentro do bloco principal. O primeiro deles está escrito: "*set global timer to get global timer* + 1", é aqui que criamos o nosso cronômetro. Ou seja, da variável que criamos "*timer*" estamos ajustando que o valor global do cronômetro para somar sempre mais um com o valor global do cronômetro. Um pouco complexo de entender, mas é assim o funcionamento, no final veremos o resultado de tudo isso.

Para a construção deste primeiro bloco, basta irmos em *Variables* (quadradinho alaranjado) e depois em *Math* (quadradrinho azul escuro) e inserir os blocos parecidos com estes.

Os outros quatros blocos restantes, são blocos de programação dos *Labels* (etiquetas), é neles que irão aparecer às informações textuais na tela de simulação, como o valor o tempo decorrido e o valor do espaço percorrido. O primeiro destes blocos nos diz: "*set* tempoverde. *Text to join get global timer* ' s'", traduzindo essa programação, ela nos mostra que no *Label "*tempoverde" aparecerá uma informação em forma de texto do valor global do cronômetro em segundos. Isso é igualmente para o "tempoazul", do próximo bloco. Fizemos essas duas programações para que apareça nos seus respectivos *Labels* (etiquetas) o valor do tempo decorrido na hora da simulação. Para criarmos um destes blocos, temos que procurar os *Labels* que ficam no canto esquerdo da **Figura 21**, e então inserirmos os mesmo blocos parecidos, depois inserir um bloco "*join*" que podemos encontrar em *Text* e o último do "*global timer*" em *Variables* ambos podemos encontrar em *Built-in*.

Os dois últimos blocos inseridos no bloco principal são também blocos de programação dos *Labels*, mas são relacionados ao espaço quando os carrinhos forem caminhar. Faremos a análise de um deles, pois como podemos ver, eles são um pouco análogos. Um dos blocos nos diz: "*set* espacoverde. *Text to join* carrinhoverde.X / 10 " m"', esta programação nos diz que no *Label* "espacoverde" teremos uma informação em forma de texto do valor da distância que o carrinho verde vai percorrer na direção do eixo X (horizontal) dadas em metros. Para construirmos este bloco de programação devemos acessar primeiro o *Label* "espacoverde" selecionar o bloco parecido com o construído aqui, em segundo lugar em *Built-in* vamos em *Text* e selecionamos um bloco "*join*", em terceiro acessamos o "carrinhoverde" que podemos encontrar em *Blocks* nos componentes da "Simulacao" e selecionarmos este bloco que indica "carrinhoverde . X". Ainda assim, o bloco "carrinhoverde . X" está dentro de um bloco de uma operação matemática de divisão por dez. Fazemos isso como uma correção do valor do espaço, já que o valor da distância é dada em *pixels* obteríamos valores muito altos para o espaço e então dividimos por dez para ajustarmos o resultado.

Feito tudo isso, o último bloco restante está relacionado com o *Label* “espacoazul” e o *ImageSprite* “carrinhoazul”, mas a programação é a mesma que a anterior, basta criarmos os mesmos blocos e apenas trocar “espacoverde” por “espacoazul” e “carrinhoverde” por “carrinhoazul”.

Feito todas essas etapas, e a mesma ideia de programação e construção do simulador, podemos agora construir o aplicativo e instalarmos no celular. Mas isso ficará para a etapa seguinte.

Etapa 13: Feito toda a programação e posto todos os componentes no celular, podemos agora criar nosso aplicativo e nos usufruir dele. Mas antes de tudo, precisamos acessar a loja virtual *Google Play* através do próprio celular, e então instalar um aplicativo chamado: MIT AI2 *Companion*, este aplicativo como veremos, nos ajudará a scanear o código QR que é similar a um código de barras, e então conseguiremos instalar o nosso aplicativo (simulador) em nosso celular. A **Figura 23** ilustra o aplicativo no qual vamos instalar em nosso celular.

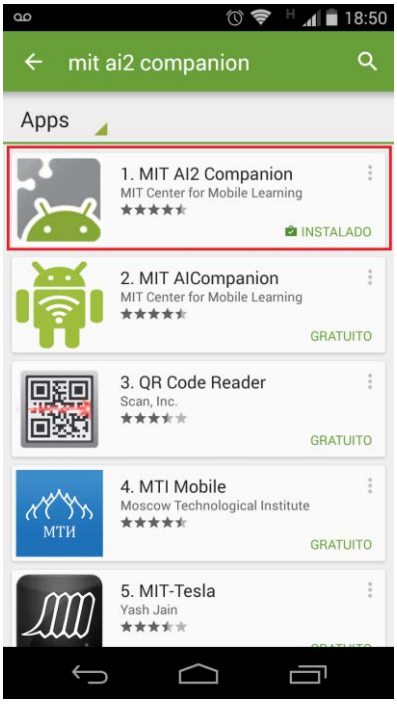

Figura 23 - MIT AI2 *Companion* na loja virtual *Google Play*.

Depois de instalado o MIT AI2 *Companion* no celular, vamos acessar o aplicativo e aparecerá como está indicado na **Figura 24**. Clicaremos em *scan* QR *code*, que está indicado em vermelho, e então abrirá uma tela com a câmera fotográfica e assim aproximaremos a câmera perto do código de barras. Lembrando que o celular precisa estar conectado na internet via *Wi-Fi* ou internet móvel para podermos instalar o simulador (aplicativo) no celular.

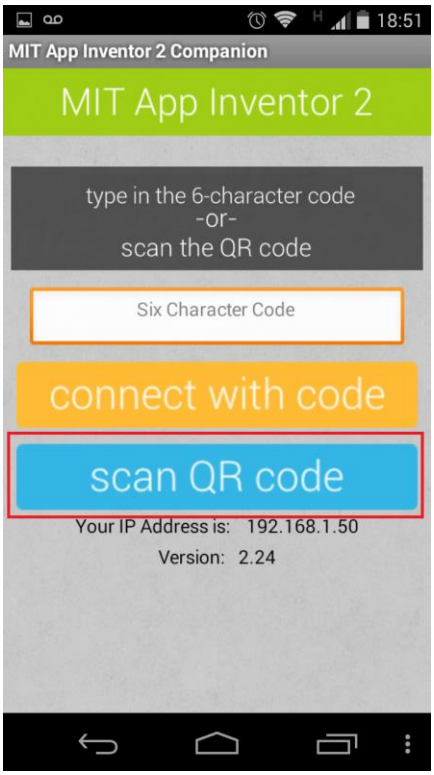

Figura 24 - Acesso ao aplicativo MIT AI2 *Companion*, clicaremos em *Scan* QR *code* para escanear o código de barras que aparecerá na tela.

Estando na página de *Designer* do nosso aplicativo (simulador), vamos clicar em *Build* e logo depois em App (*provide* QR *code for* .apk) como está indicado na **Figura 25**.

Aparecerá uma pequena janela de progresso, e quando terminado aparecerá o código QR e então com o celular utilizaremos o aplicativo MIT AI2 *Companion* para podermos scanear este código. A **Figura 26** ilustra como devemos fazer para scanear o código QR. Feito isso, quando scaneado aguardaremos alguns segundos e então surgirá em uma outra tela, parecido com a **Figura 27**, que é aqui que vamos instalar o nosso aplicativo (simulador) no celular. Basta clicarmos em Instalar (em vermelho) e aguardar alguns instantes. Caso a tela de instalação não apareça, verifique se o celular está com conexão com a Internet, *Wi-fi* ou internet móvel, e tente novamente.

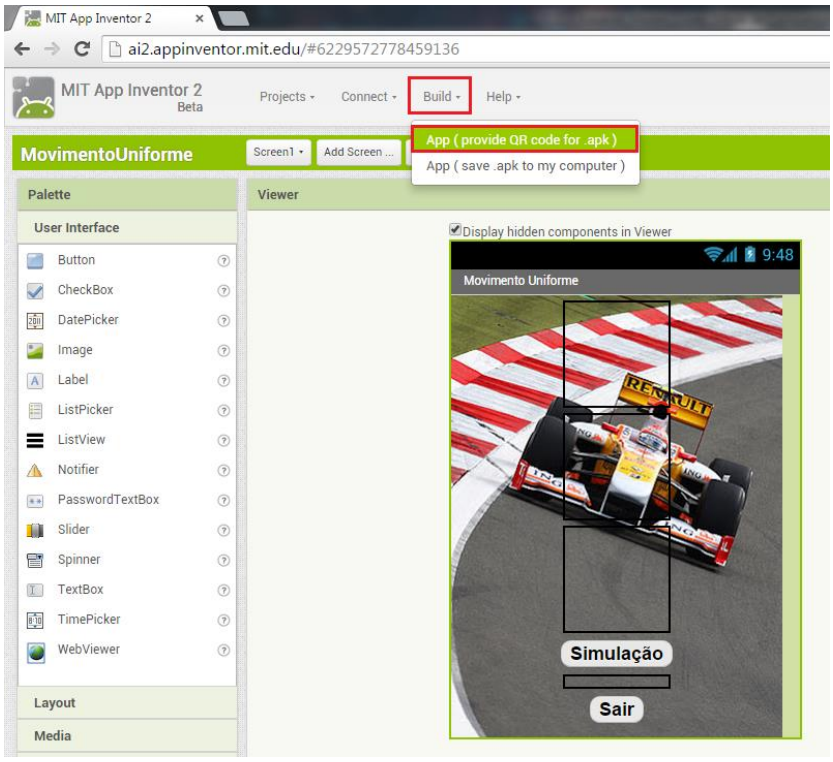

Figura 25 - Clicamos em *Build* para construirmos o nosso aplicativo (simulador) para celular.

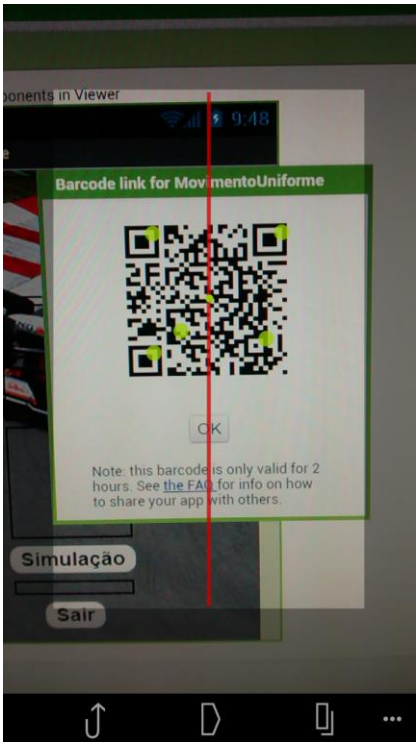

Figura 26 - Quando o código (QR) aparecer clicaremos no botão *scan* QR *code* no aplicativo MIT AI2 *Companion* e scanearemos o código que surgir.

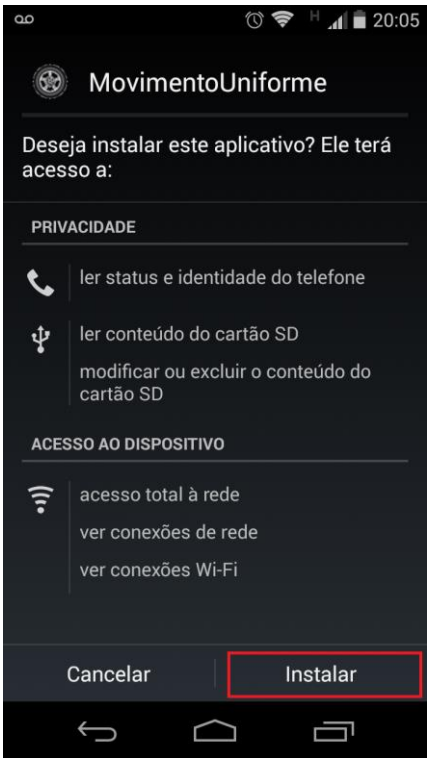

Figura 27 – Tela de instalação do nosso simulador.

Pronto, depois de ter seguido todas essas etapas temos o simulador (aplicativo) instalado em nosso celular.

Agora iremos iniciar o segundo trabalho proposto, desenvolver um simulador para o movimento acelerado.

4.3. Desenvolvendo um simulador de Física para celular: Movimento Acelerado.

Vamos agora desenvolver outro possível simulador de Física através da ferramenta do MIT. Não iremos aprofundar como fizemos no simulador anterior, apenas será mostrado como ficou o *designer* da tela principal e como ficou a linha de programação. Até porque a linha de programação acabou ficando complexa e a ideia é não querer complicar algo que já é desconhecido pelos alunos. Apenas veremos a diferença de complexidade entre o primeiro simulador que já criamos, e deste, que vamos mostrar em pequenos detalhes.

O simulador feito se chama “SeMovimentando”. A ideia foi criar um simulador de Física, que pudesse contar os passos dados quando uma pessoa for caminhar com o celular. Haverá

apenas uma tela principal onde estará toda a informação da velocidade naquele momento, da distância percorrida, em metros, e, em passos e informações como altitude, longitude, latitude e intervalo de medida. A **Figura 28** mostra como é a tela principal deste simulador através da página de *Designer* do MIT App *Inventor*. Como podemos ver, existirão três botões que são eles: Sair, que finaliza o aplicativo, o Minimizar, que esconderá os dados do GPS, e o botão Resetar, que reseta as informações do velocímetro e das distâncias.

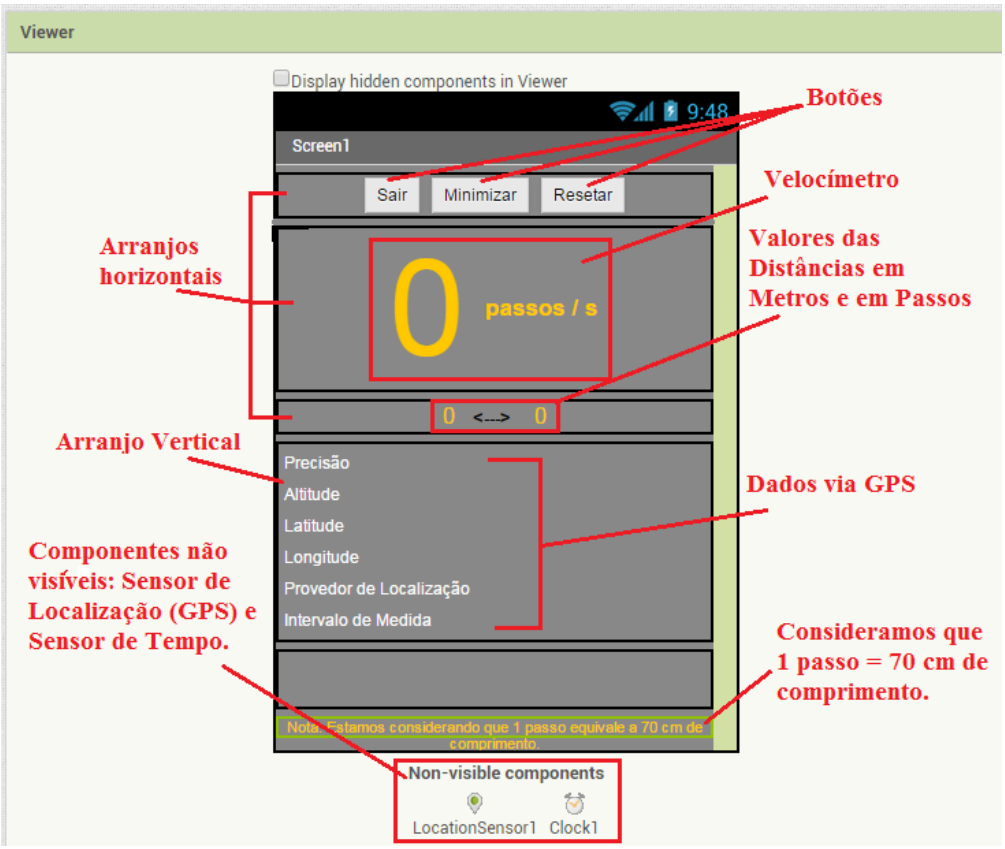

Figura 28 - Tela principal do simulador "SeMovimentando".

Logo abaixo dos botões temos o velocímetro, dado em passos por segundo, e mais em baixo teremos os valores das distâncias, dada em metros, e a outra, em passos. Notamos que aqui foi considerado que um passo equivale a setenta centímetros de comprimento. E como veremos, temos que corrigir este fator na programação para obtermos os valores corretos.

Todas essas informações textuais como o velocímetro, as distâncias e os dados do GPS são obtidas através dos *Labels* (etiquetas), como vimos no simulador anterior, mas neste simulador haverá uma programação enorme ligada aos *Labels*. Ainda como podemos ver na **Figura 28** existirão os arranjos horizontais e verticais, que como vimos anteriormente, servem para modelarmos o *design* do nosso simulador e rearranjar a posição dos textos e botões

inseridos no aplicativo. Ainda do simulador, existirão dois componentes não visíveis que são eles: sensor de localização e o sensor de tempo. Ambos serão fundamentais para a obtenção dos resultados, sendo o sensor de localização o mais importante, pois é dele que será possível obter o valor da distância através da programação de uma equação muito utilizada nos sistemas de navegação (GPS), e em conjunto com o sensor de tempo, será possível obtermos o valor da velocidade naquele momento.

Não iremos nos aprofundar, como fizemos anteriormente, apenas será mostrado como ficou a programação através de imagens com pequenas descrições.

Agora iremos ver como ficou a página da linha de programação deste simulador. A **Figura 29** mostra as variáveis globais que foram criadas. Foram estabelecidas algumas importantes relações, como o valor da constante π, o raio da Terra, e os valores das latitudes e longitudes em unidades de graus e radianos dentre outras, pois logo mais veremos que essas variáveis vão ser fundamentais para uma importante equação na qual foi programada.

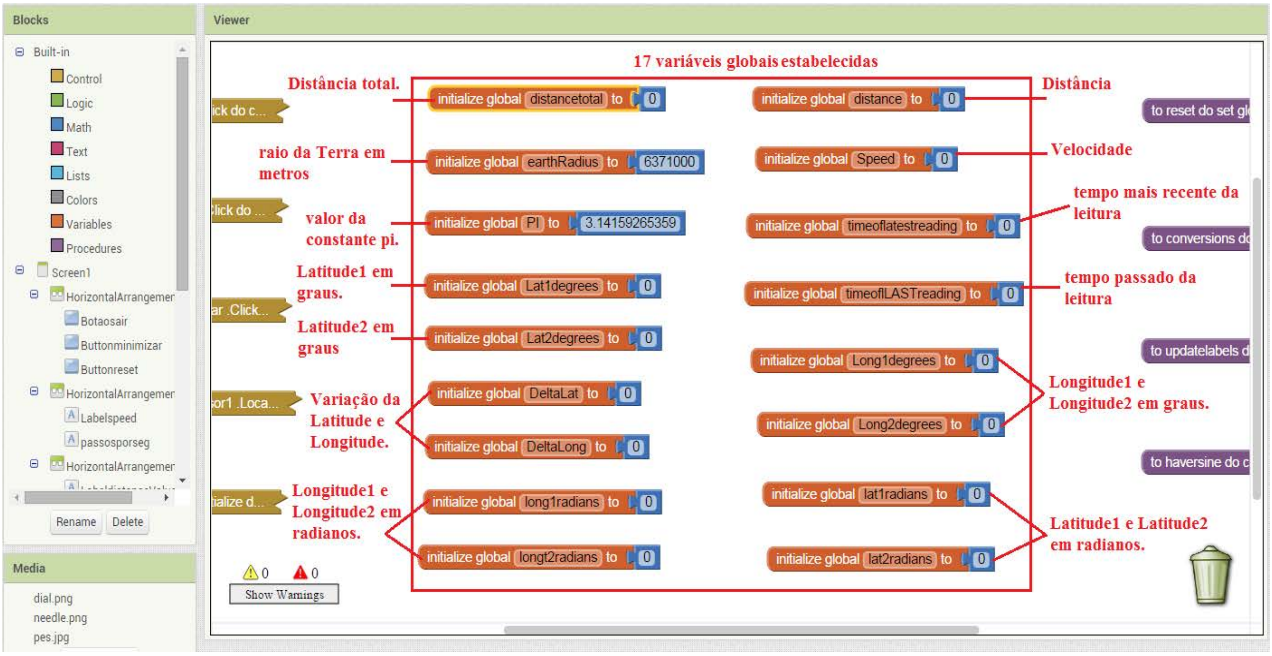

Figura 29 - Página de programação do simulador "SeMovimentando". Foram criadas dezessete variáveis globais nas quais serão utilizadas em outras programações.

Em seguida veremos como ficou a programação dos três botões inseridos na tela inicial do simulador, são eles: Sair, Minimizar e Resetar. A programação pode ser vista pela **Figura 30**. Podemos notar que há pequenas informações escritas em vermelho na imagem, foi feito isso para facilitar o entendimento de cada bloco de programação e suas funcionalidades.

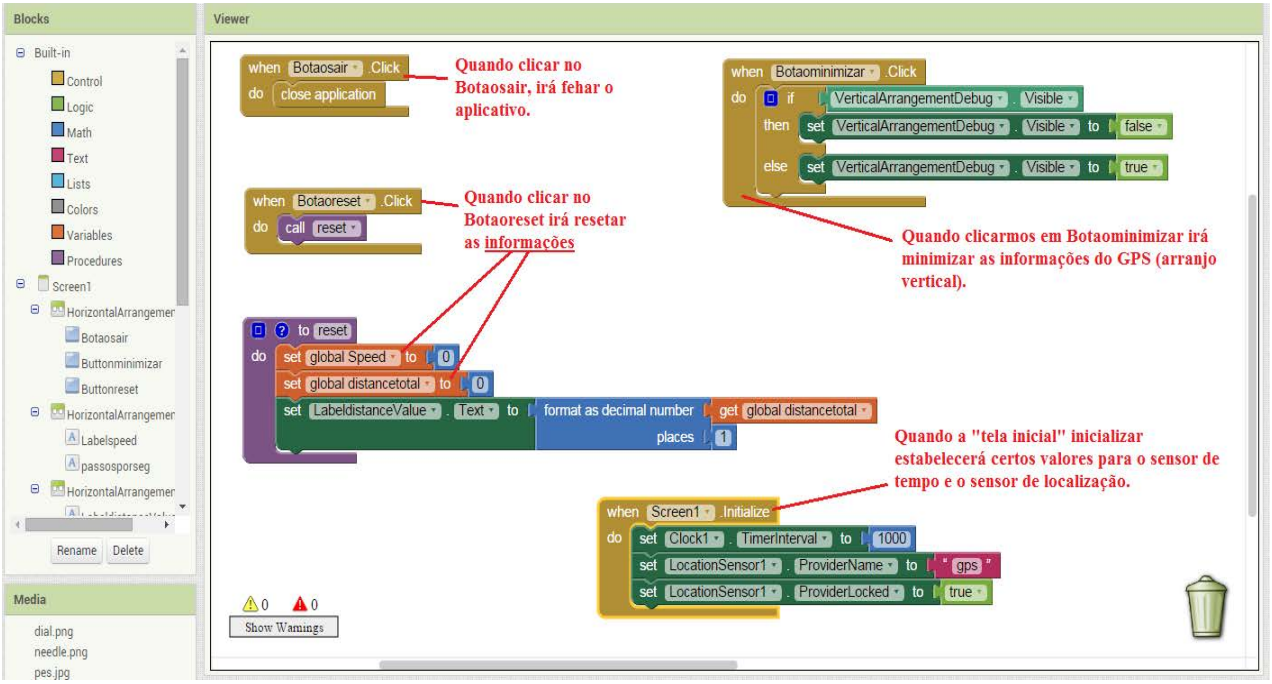

Figura 30 - Programação dos três botões inseridos na tela principal e programação da inicialização da tela principal (*Screen*1).

Ainda teremos muitas programações, a **Figura 31** ilustra um processo de conversão de unidades das Latitudes e Longitudes, como podemos ver pela programação.

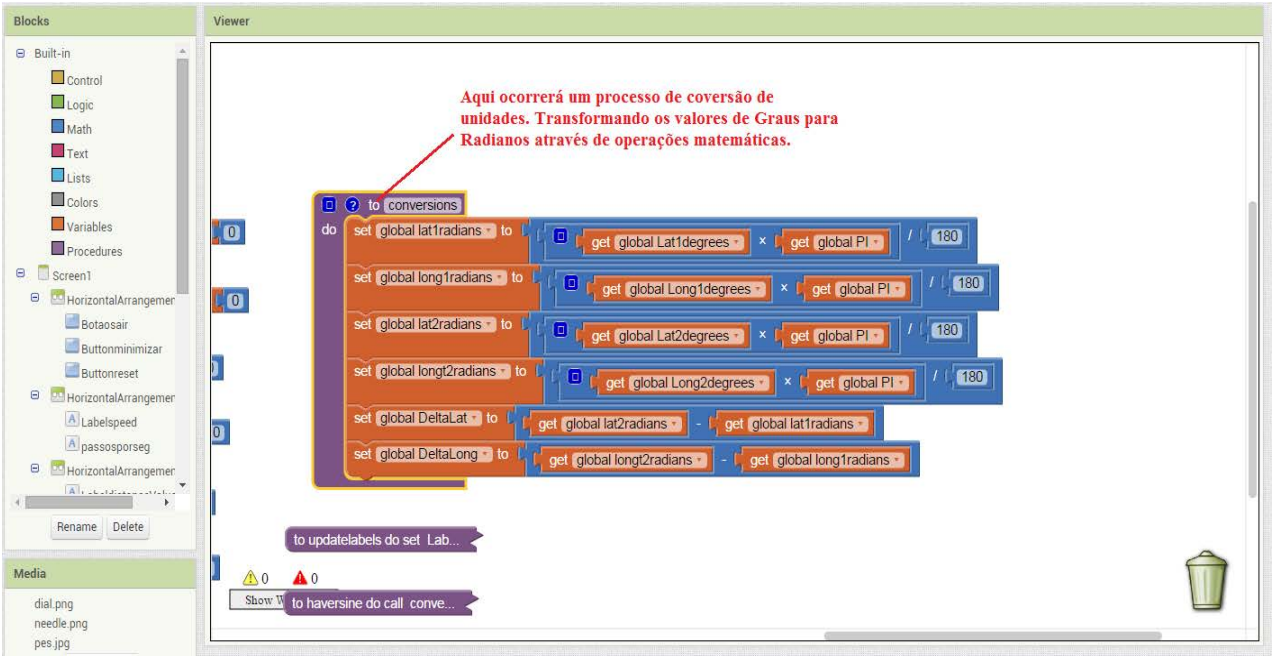

Figura 31 - Processo de conversão de unidades das latitudes e longitudes.

A **Figura 32** nos mostra a programação de uma equação utilizada para o desenvolvimento do simulador. Foi utilizada na programação a equação de Haversine. Ela é muito utilizada nos

sistemas de navegação via GPS, e é muito útil no cálculo de distâncias quando há variações dos valores de latitude e longitude. Como podemos ver, algumas das variáveis globais da **Figura 29** vão estar presentes nesta equação. Podemos notar que a equação é bem grande e não cabe na imagem, então foi indicada uma seta vermelha que mostra o restante da equação.

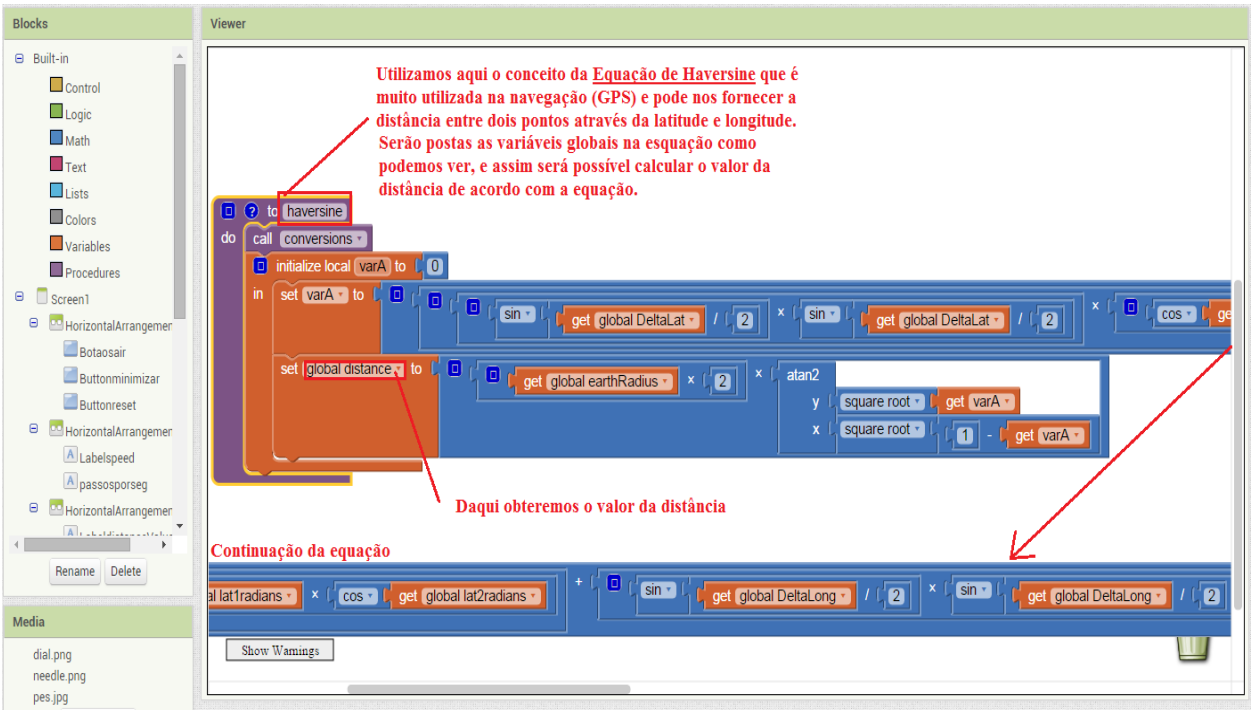

Figura 32 - Programação da equação de Haversine que é muito utilizada para a navegação (GPS).

Ainda foi feito a programação do sensor de localização como podemos visualizar na **Figura 33**. Quando o sensor detectar uma variação da latitude e longitude toda aquela programação vai ser executada, claro que é tudo muito rapidamente, mais o principal é que vamos obter o valor da velocidade. Essa velocidade vai depender da variação da distância pelo intervalo de tempo medido como pode ser visto na imagem. Lembrando que o valor da distância surge através da equação de Haversine.

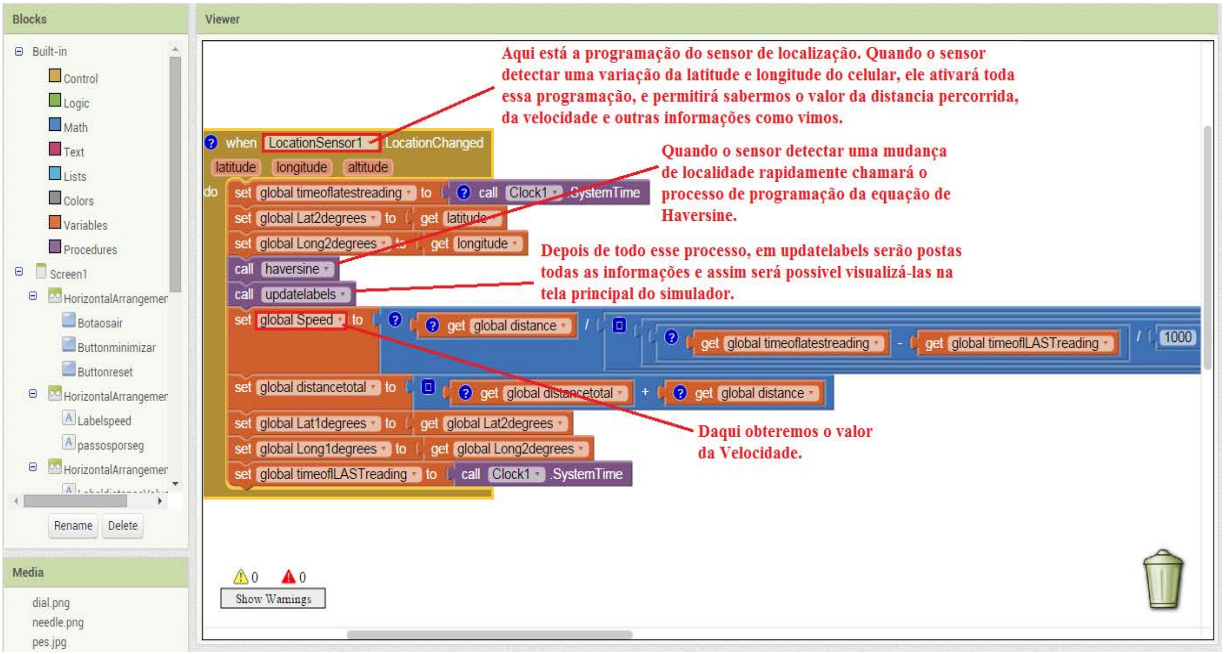

Figura 33 - Programação do sensor de localização.

E por fim, temos a última programação, como podemos ver na **Figura 34**. É aqui que ocorrerá toda a programação textual dos *Labels* (etiquetas), ou seja, as informações como Precisão, Altitude, Latitude, Longitude, Velocidade, Distância e outras, aparecerão na tela principal. Vale a pena notar que só aqui que foi posto o fator de conversão da velocidade de metros por segundo para passos por segundo, como está indicado na imagem. E o mesmo fazemos para a distância, basta multiplicarmos o último bloco "*global distancetotal*" pelo mesmo fator, pois consideramos que um passo equivale a setenta centímetros de comprimento.

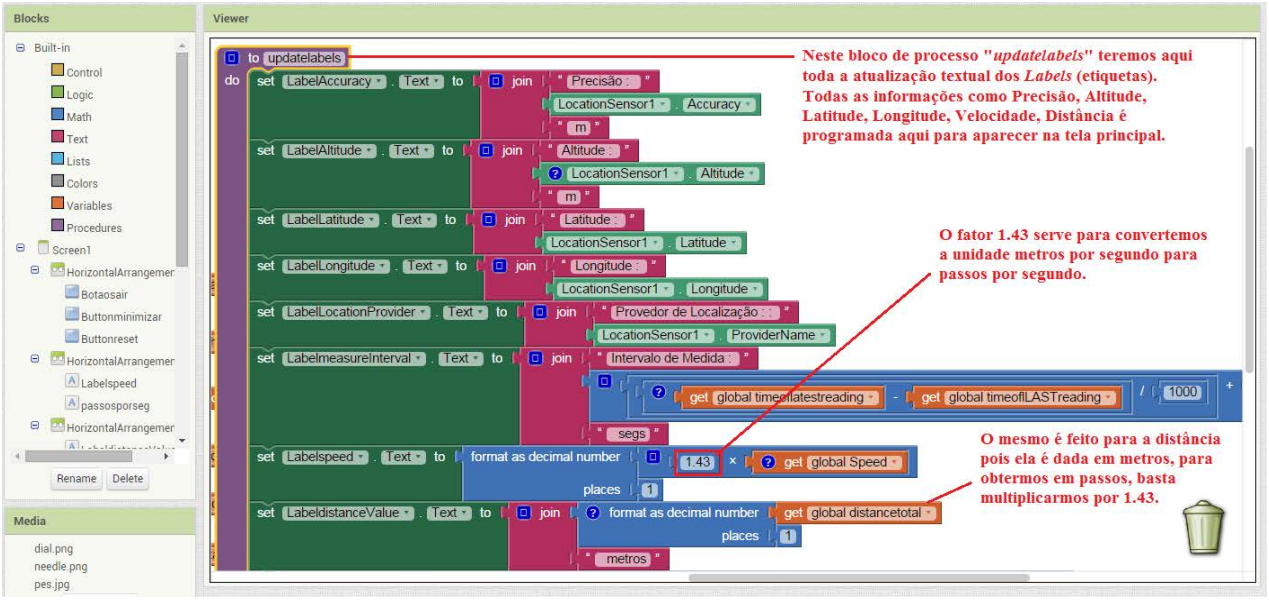

Figura 34 - Programação dos *Labels* (etiquetas).

Como podemos ver, a programação deste simulador é muito mais complexa do que no outro simulador. A ideia aqui foi apenas querer mostrar esta diferença de complexidade na programação e também mostrar os conceitos de programação ligados com a Física.

5. Plano de Aula

Aqui será exposto um Plano de Aula como proposta de uso para uma aula de Física no Ensino Médio.

- **Dados de Identificação**

  Disciplina: Física

  Série: Primeira, Segunda ou Terceira série do Ensino Médio

  Período: 03 (três) ou 04 (quatro) aulas com duração de 50 (cinqüenta) minutos cada, ou ainda, propor um trabalho bimestral.

- **Tema:** Construir um simulador de Física para celular utilizando os conceitos do Movimento Uniforme (ou outros conceitos da Física).

- **Objetivos**

  Identificar o modo de pensar dos alunos frente ao uso dos celulares na educação.

  Construir juntos com os alunos os principais conceitos de programação para a elaboração do simulador. (em sala de aula)

  Expor roteiros e dar todo o ferramental que será necessário para os alunos desenvolverem o simulador por conta própria. (em um trabalho bimestral)

  Despertar o interesse dos alunos pela Física, através do uso de novas tecnologias na Educação.

- **Procedimentos de Ensino**

  Utilização de recursos áudios-visuais.

  Utilização de computadores. (salas de informática na escola)

  Exposição oral de conceitos

  Debate em sala de aula

  Uso de roteiros (para um trabalho bimestral)

- **Recursos Didáticos**

  Data Show, Lousa, Roteiros, Vídeos tutoriais do YouTube, Computadores, Celular (smartphones), livro de Física.

- **Avaliação**

  Como principal instrumento de avaliação será considerado o envolvimento do aluno com o trabalho proposto. E também, como será feita a assimilação do simulador com o conceito da Física, sendo um ponto crucial desta avaliação.

6. Resultados

Depois de instalado os simuladores no celular, mostraremos os resultados aqui nesta seção. Foi utilizado para a análise de dados o modelo de celular MOTO G da marca Motorola. Para começar vamos analisar o primeiro simulador construído, o simulador do Movimento Uniforme. A **Figura 35** ilustra os simuladores instalados no celular, como podemos ver o simulador do Movimento Uniforme é aquele que tem um ícone em forma de pneu. Clicando nele iniciaremos o aplicativo.

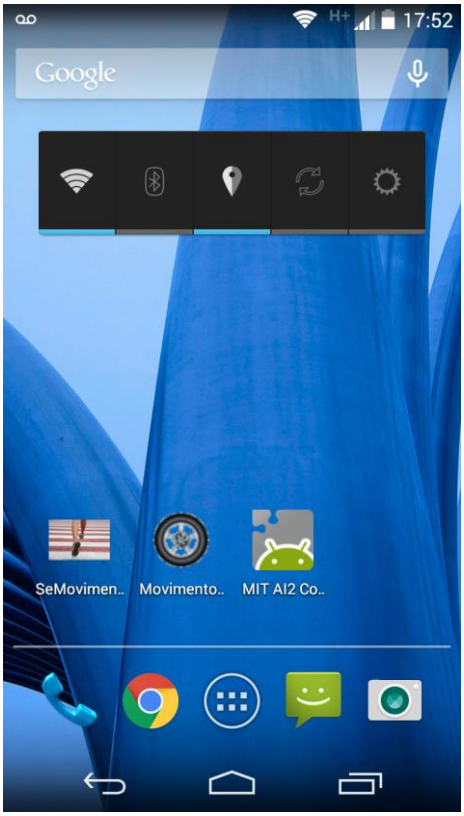

Figura 35 - Simuladores instalados no Celular.

A **Figura 36** mostra a tela inicial do aplicativo. Como tínhamos proposto, a tela inicial teria dois botões. Quando clicarmos no botão Simulação abrirá uma segunda tela, e é nessa tela que programamos toda a simulação dos carrinhos na pista de corrida. E quando clicarmos no botão Sair, o aplicativo fecha e volta pra tela inicial do Celular.

Figura 36 - Tela inicial do simulador: Movimento Uniforme.

Logo quando clicarmos no botão Simulação a corrida irá iniciar, e assim podemos acompanhar qual o carrinho vai chegar mais rápido. A **Figura 37** ilustra uma foto tirada no momento da simulação.

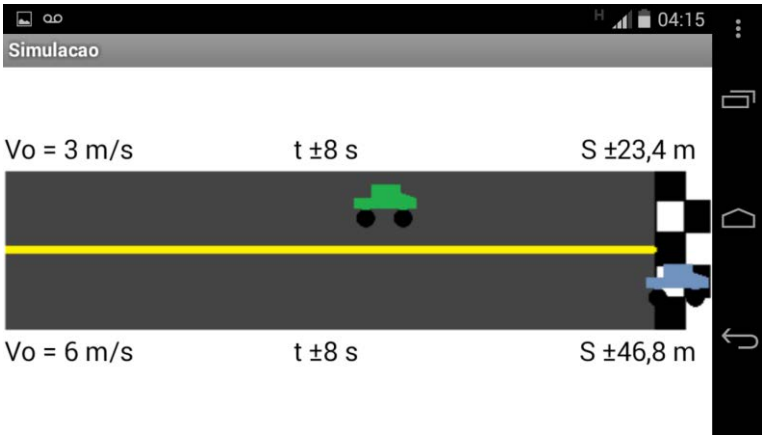

Figura 37 - Uma foto tirada de um momento da simulação.

Como tínhamos proposto, um carrinho teria o dobro da velocidade do outro carrinho. O carrinho azul tem o dobro da velocidade do carrinho verde, como eles estão em Movimento Uniforme, ou seja, aceleração nula e velocidade constante, o carrinho azul vai chegar primeiro no fim da pista, pois sua velocidade é maior e precisará de menos tempo para percorrer o mesmo trajeto do outro carrinho. Como podemos constatar pela **Figura 37**, o carrinho azul

chega ao fim da pista com 8 segundos, enquanto o carrinho verde ainda está na metade da pista.

Existe ainda um pequeno detalhe a se notar, se o carrinho azul termina o trajeto em 8 segundos com velocidade de 6 m/s, então o espaço percorrido seria de 48 metros, mas está marcado 46,8 m na imagem. Como a foto foi tirada em um certo instante de tempo não foi possível mostrar o valor correto, pois ainda não tinha atualizado o espaço percorrido ao mesmo tempo em que a foto foi tirada.

A ideia da Física envolvida neste simulador é muito simples. Queríamos apenas verificar qual dos carrinhos iria chegar ao final da pista primeiro. Como vimos à programação deste simulador não foi tão complicada e pode ser facilmente entendida pelos alunos. Existem muitas outras possibilidades de criação de simuladores utilizando a mesma ferramenta que foi utilizada neste trabalho e também sobre o mesmo tema. Os alunos poderiam, por exemplo, construir um simulador da equação do movimento uniforme, onde facilmente eles colocariam os valores da velocidade, tempo e espaço e poderiam obter as respostar para os exercícios.

Iremos abordar o funcionamento do segundo simulador que é baseado no movimento acelerado. A ideia para este simulador foi em criar um contador de passos, ou seja, quando caminharmos com o simulador ativo no celular ele nos dará informações como a distância percorrida em metros e em passos, e a velocidade dada em passos por segundo naquele instante. Lembrando que consideramos o valor de 1 passo equivalente a 70 cm de comprimento, entretanto, nem sempre 1 passo de uma pessoa equivale exatamente esta medida, pode ser que seja maior ou menor, na média este valor está em 60 a 70 centímetros.

A **Figura 38** mostra uma foto tirada do simulador em uso num instante qualquer. A foto foi tirada durante uma pequena caminhada. Podemos notar que naquele instante temos o valor da velocidade em passos por segundo, o valor da distância percorrida em metros e também em passos, e abaixo todos os dados obtidos via GPS. A precisão mostra o quão vai ser preciso o valor das medidas, ou seja, se o valor da precisão for grande indica que a precisão do celular com os satélites é bem baixa, e se o valor da precisão for baixo significa uma ótima precisão do celular com os satélites. Como esta na imagem, o valor da precisão é de 4 metros, este valor indica que o celular esta no centro de uma circunferência de raio 4 m. É um valor bem preciso, enquanto que se o celular estiver num ambiente fechado, em uma sala, por exemplo, o valor da precisão pode subir até um raio de 40 m, por isso que é melhor utilizá-lo em ambientes mais abertos pois a precisão será sempre maior. Os outros dados, como Altitude, Latitude e Longitude são apenas dados de onde está localizado o celular, mas são de suma

importância principalmente a latitude e longitude. Vimos que a programação deste simulador é bem complexa e trabalhosa. Para calcularmos o valor da distância temos que utilizar a equação de Haversine e com ela precisaremos dos dados da latitude e longitude para podermos depois calcular o valor da velocidade. O simulador é praticamente um pequeno GPS, através dos dados de localização é possível fazer certa programação e assim podemos obter as variáveis desejadas.

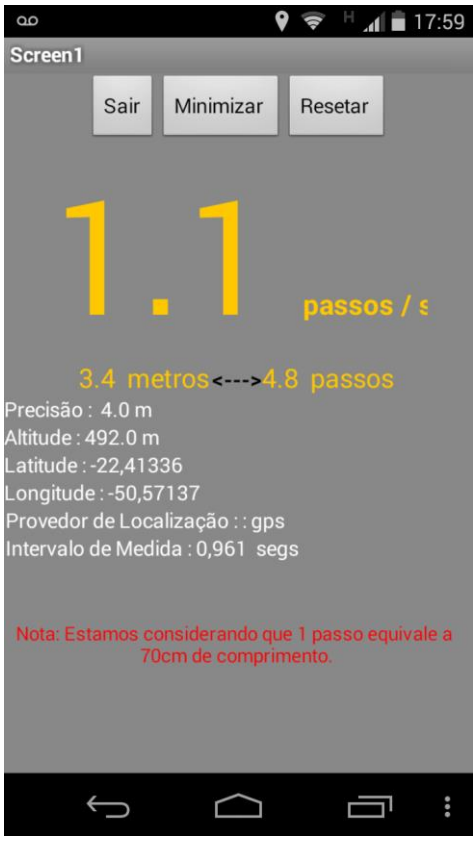

Figura 38 – Foto da tela inicial do simulador "SeMovimentando" em uso no celular.

Enfim, existe uma infinidade de possibilidades para a construção de simuladores para celular utilizando os conceitos de Física abordados. Como vimos os dois simuladores tem ideias muito simples, mas na questão de programação o segundo se tornou bem complexo. Mais é dessa complexidade que queremos que os alunos interajam mais com o professor e com as aulas de Física.

7. Conclusões

Acredito que mesmo os professores tendo problemas em salas de aulas com os celulares, estes pequenos aparelhos podem ser muito bem usados no processo de ensino dos alunos. Se o professor traçar certos objetivos o uso destes aparelhos podem ser estimulantes para o ensino.
O professor não pode achar que o uso destes pequenos aparelhos irá substituir sua aula, na realidade deve ser usado como uma nova ferramenta de auxílio à aprendizagem dos alunos.
Cabe ao professor ir atrás de novas técnicas de ensino envolvendo estas novas tecnologias, e mostrar a eles que os conceitos da Física podem facilmente ser construídos utilizando estes novos aparelhos tecnológicos, pois a cada dia estão mais presentes nas mãos dos alunos.

8. Referências Bibliográficas

GRZESIUK, D. F. **O USO DA INFORMÁTICA NA SALA DE AULA COMO FERRAMENTA DE AUXILIO NO PROCESSO ENSINO-APRENDIZAGEM**.
2008. 39 f. Trabalho de Conclusão de Curso (Especialização em Métodos e Técnicas de Ensino) – Universidade Tecnológica Federal do Paraná, Medianeira, 2008.

COELHO, Rafael O. **O Uso da Informática no Ensino de Física de Nível Médio**.
2002. 101 f. Dissertação (Mestrado em Educação) – Faculdade de Educação da Universidade Federal de Pelotas, Pelotas, 2002.

ALVIM, M. Apesar da frequente proibição, Unesco recomenda o uso de celular em sala de aula: Instituição estimula o acolhimento da tecnologia nas disciplinas. **O GLOBO**. Rio de Janeiro: 28 out. 2014. Disponível em:<http://oglobo.globo.com/rio/bairros/apesar-da-frequente-proibicao-unesco-recomenda-uso-de-celular-em-sala-de-aula-14372630?utm_source=Facebook&utm_medium=Social&utm_campaign=O+Globo>. Acesso em: 15 fev. 2015.

LOPES, Roseli de Deus; et al. **O uso do computador e da internet na escola pública**. Disponível em:< http://www.fvc.org.br/estudos-e-pesquisas/avulsas/estudos1-7-uso-computadores.shtml?page=0>. Data de acesso: 17 fev. 2015.

PEREIRA, Antonio Fabiano. O professor, o aluno e o uso de celular na sala de aula. **A Tribuna**, Mato Grosso, 16 ago. 2014. Disponível em:<http://www.atribunamt.com.br/2014/08/o-professor-o-aluno-e-o-uso-de-celular-na-sala-de-aula/>. Acesso em: 18 fev. 2015.

MARTINS, A. R. **O melhor do computador**. Disponível em:<http://revistaescola.abril.com.br/formacao/melhor-computador-450791.shtml>. Data de acesso: 20 fev. 2015.

MASSACHUSETTS INSTITUTE OF TECHNOLOGY. ***About Us***. Disponível em:< http://appinventor.mit.edu/explore/about-us.html>. Data de acesso: 20 fev. 2015.