



# Interativa

## Programação para Dispositivos Móveis

**Autor:** Prof. Olavo Tomohisa Ito

**Colaboradoras:** Profa. Vanessa Santos Lessa  
Profa. Christiane Mazur Doi

## **Professor conteudista: Olavo Tomohisa Ito**

Mestre em Engenharia de Produção pela Universidade Paulista (UNIP), bacharel em Física pelo Instituto de Física da Universidade de São Paulo (USP), licenciado em Ciências pela Faculdade de Educação da USP, e bacharel em Língua e Literatura Japonesa pela Faculdade de Filosofia, Letras e Ciências Humanas (FFLCH) da USP. Professor dos cursos de graduação tradicional de Sistemas de Informação e Ciência da Computação, bem como do curso superior tecnológico de Análise e Desenvolvimento de Sistemas. Das disciplinas que ministra, *Programação para Dispositivos Móveis* é especial, pois durante muitos anos atuou no ensino de programação para crianças e jovens, desde a Pré-Escola, com ScratchJr, até o Ensino Médio, com App Inventor, entre outros. Em 2015, recebeu a certificação Mobile Computing with App Inventor – CS Principles pelo Trinity College, Connecticut.

### **Dados Internacionais de Catalogação na Publicação (CIP)**

I89p Ito, Olavo Tomohisa.

Programação para Dispositivos Móveis / Olavo Tomohisa Ito. – São Paulo: Editora Sol, 2021.

296 p., il.

Nota: este volume está publicado nos Cadernos de Estudos e Pesquisas da UNIP, Série Didática, ISSN 1517-9230.

1. APP. 2. Identificação. 3. Criação. I. Título.

CDU 004.512

U511.34 – 21

Prof. Dr. João Carlos Di Genio  
**Reitor**

Prof. Fábio Romeu de Carvalho  
**Vice-Reitor de Planejamento, Administração e Finanças**

Profa. Melânia Dalla Torre  
**Vice-Reitora de Unidades Universitárias**

Profa. Dra. Marília Ancona-Lopez  
**Vice-Reitora de Pós-Graduação e Pesquisa**

Profa. Dra. Marília Ancona-Lopez  
**Vice-Reitora de Graduação**

### **Unip Interativa – EaD**

Profa. Elisabete Brihy  
Prof. Marcello Vannini  
Prof. Dr. Luiz Felipe Scabar  
Prof. Ivan Daliberto Frugoli

### **Material Didático – EaD**

Comissão editorial:

Dra. Angélica L. Carlini (UNIP)  
Dr. Ivan Dias da Motta (CESUMAR)  
Dra. Kátia Mosorov Alonso (UFMT)

Apoio:

Profa. Cláudia Regina Baptista – EaD  
Profa. Deise Alcantara Carreiro – Comissão de Qualificação e Avaliação de Cursos

Projeto gráfico:

Prof. Alexandre Ponzetto

Revisão:

Ricardo Duarte  
Ana Fazzio



# Sumário

## Programação para Dispositivos Móveis

APRESENTAÇÃO .....	7
INTRODUÇÃO .....	9

### Unidade I

1 INTRODUÇÃO AO APP INVENTOR .....	12
1.1 Acesso ao ambiente de desenvolvimento .....	15
1.2 Apresentação dos componentes do ambiente .....	18
1.3 Instalação e configuração do emulador .....	21
1.4 Teste do emulador.....	33
2 DESIGNER .....	34
2.1 Janelas .....	35
2.1.1 Paleta .....	37
2.1.2 Visualizador.....	48
2.1.3 Componentes .....	49
2.1.4 Mídia .....	50
2.1.5 Propriedades.....	51
2.2 Criação do app para a demonstração de componentes.....	52
3 BLOCOS .....	58
3.1 Conceito de programação: comportamento.....	62
3.2 Tipos de evento .....	63
3.3 Componentes dos blocos .....	65
3.4 Criação do app para a demonstração de componentes.....	82
4 TESTE DE APP .....	92

### Unidade II

5 APlicativo, TELA SIMPLES E APIs .....	98
5.1 Projetando o app1 .....	98
5.1.1 Identificação da demanda e definição das características do app1 .....	98
5.1.2 Criação do app1 .....	114
5.1.3 Teste do app1 .....	127
5.2 Projetando o app2 .....	136
5.2.1 Identificação da demanda e definição das características do app2 .....	136
5.2.2 Criação do app2 .....	146
5.2.3 Teste do app2 .....	160

<b>6 APlicativo, telas mÚltiplas e procedimentos .....</b>	<b>163</b>
<b>6.1 Projetando o app3 .....</b>	<b>163</b>
<b>6.1.1 Identificação da demanda e definição das características do app3 .....</b>	<b>163</b>
<b>6.1.2 Criação do app3 .....</b>	<b>177</b>
<b>6.1.3 Teste do app3 .....</b>	<b>202</b>

## **Unidade III**

<b>7 TEORIA .....</b>	<b>210</b>
<b>7.1 Planejando jogos .....</b>	<b>210</b>
<b>7.1.1 Identificação da demanda e definição das características do jogo .....</b>	<b>211</b>
<b>7.1.2 Apresentação da metodologia de game design document (GDD) .....</b>	<b>213</b>
<b>7.1.3 Criação de imagens para jogos .....</b>	<b>219</b>
<b>8 JOGOS NO APP INVENTOR.....</b>	<b>222</b>
<b>8.1 Projetando o jogo 1.....</b>	<b>222</b>
<b>8.1.1 Design.....</b>	<b>222</b>
<b>8.2 Projetando o jogo 2.....</b>	<b>241</b>
<b>8.2.1 Design.....</b>	<b>241</b>
<b>8.3 Projetando o jogo 3.....</b>	<b>253</b>
<b>8.3.1 Design.....</b>	<b>253</b>
<b>8.3.2 Spritelagem: eventos e propriedades.....</b>	<b>265</b>
<b>8.4 Finalizando o jogo.....</b>	<b>278</b>
<b>8.4.1 Pontuação, tela de início, ranking e créditos .....</b>	<b>278</b>
<b>8.4.2 Atribuição de ícone e título.....</b>	<b>279</b>
<b>8.4.3 Geração de arquivo APK .....</b>	<b>280</b>
<b>8.4.4 Distribuição do jogo .....</b>	<b>280</b>
<b>8.5 Preparando para a publicação na Play Store.....</b>	<b>284</b>
<b>8.5.1 Informações dos detalhes do app .....</b>	<b>284</b>
<b>8.5.2 Versões do app .....</b>	<b>284</b>
<b>8.5.3 Classificação do aplicativo .....</b>	<b>285</b>
<b>8.5.4 Preços e distribuição.....</b>	<b>285</b>
<b>8.5.5 Conta do desenvolvedor .....</b>	<b>285</b>

## **APRESENTAÇÃO**

Caro aluno,

A disciplina *Programação para Dispositivos Móveis* visa a propiciar o entendimento e o desenvolvimento de pequenos projetos de aplicativos e de jogos para celular por meio de ferramentas visuais de uso livre.

Assim, objetiva-se que os estudantes tenham contato direto com o ambiente mobile, de modo que possam desenvolver projetos com o emprego do App Inventor e se familiarizar com a realidade multidisciplinar dos trabalhos profissionais realizados em empresas da área de computação.

Em suma, este livro-texto tem como propósito apresentar aos alunos conceitos relacionados ao desenvolvimento de aplicações para dispositivos móveis com o auxílio de uma ferramenta básica, a qual dispõe de módulos de software que agilizam o processo de criação e de implementação de projetos.

A computação móvel está cada vez mais atuante no cotidiano das pessoas. Nesse cenário, faz-se necessária uma disciplina capaz de reunir a praticidade de um dispositivo bastante versátil com a aprendizagem de conceitos e tecnologias computacionais.

A própria ferramenta utilizada no curso, o App Inventor, foi desenvolvida para possibilitar o acesso de pessoas sem formação em informática ao mundo da criação de aplicativos.

Para apresentar o curso, vamos ler as palavras do idealizador do App Inventor, Hal Abelson, em uma entrevista ao canal do YouTube The Brainwaves Video Anthology (2015).

Eu ensino computação no MIT e estou aqui para dizer que a computação móvel pode ser democratizada. Em outras palavras, as pessoas poderão escolher como vão querer usar a tecnologia móvel, se nesse novo mundo da tecnologia móvel elas serão criadoras, construtoras ou produtoras, ou se serão apenas consumidoras de coisas prontas, que outras pessoas fizeram para elas.

Quando eu vim para o MIT, no início da era do computador pessoal, era uma época incrível, porque aqui ainda estavam as máquinas que até então eram usadas exclusivamente por agências governamentais, grandes indústrias, e de repente, em alguns anos, tudo isso se transformou em ferramentas para o uso de pessoas comuns, tornando a computação pessoal uma ferramenta com usos variados na infraestrutura e na indústria, como vemos atualmente.

Os líderes da tecnologia hoje, ou as pessoas que estão agora na casa dos 30, 40 e 50 anos, eram adolescentes naquela época, que manusearam e jogaram com esses brinquedos computacionais.

Agora estamos passando por uma revolução ainda maior, a tecnologia móvel. Esse é um tipo de computação pessoal que é pessoal como nunca antes. Ele diz respeito a quem você é, para onde vai; tem informações sobre o que dizem de você, o que faz, com quem fala, como se comunica com os seus amigos. Mas a questão é se as pessoas seriam capazes de fazer um aplicativo.

Portanto, aqui no MIT, estamos examinando a questão da abertura da tecnologia móvel, de modo que todos possam se tornar criadores de aplicativos móveis, sejam ou não programadores. Fizemos um sistema no meu grupo de pesquisa no MIT chamado App Inventor, que permite que todos criem aplicativos móveis, mesmo que nunca tenham programado antes.

Hoje temos cerca de 2 ou 3 milhões de criadores de aplicativos, e seus aplicativos pessoais lhes permitem produzir impacto em sua comunidade. Por exemplo, um grupo de alunos do Ensino Médio na Califórnia fez um aplicativo que permite às pessoas da comunidade encontrar lugares que precisam ser limpos, paredes pichadas, locais com lixo e coisas assim. Você tira uma foto com o aplicativo, e isso ajuda a agendar reuniões com a comunidade para programar a limpeza. Um grupo de estudantes do Ensino Médio na Moldávia permitiu às pessoas em seu país identificar onde a água está limpa e onde está poluída, e compartilhar isso, de modo que todo o país saiba onde encontrar água pura. Um grupo de meninas no Texas criou um aplicativo para ajudar um colega cego a encontrar o caminho e a se movimentar no entorno da escola. Uma estudante em Boston fez um aplicativo para os ciclistas da cidade tirarem fotos e informarem às autoridades municipais sobre carros estacionados em pontos que bloqueiam as ciclovias. Todos são pequenos aplicativos que as pessoas criaram, coisas que para elas são pessoalmente significativas, mas também mostram que podem ter um impacto na comunidade, na família e nos amigos.

Bom, esse tipo de democratização da tecnologia é muito divertido, e muitas coisas interessantes acontecem, mas acho que também é importante para a nossa sociedade democrática. Como sabemos, estamos no meio de uma tecnologia muito, mas muito poderosa. Estamos todos preocupados com a privacidade. Estamos todos preocupados com o que esses aplicativos podem fazer. Uma forma de garantir que nossa sociedade continue a ser responsável à cidadania é colocar o poder de criar aplicativos nas mãos de todos, e não apenas nas mãos de alguns profissionais e grandes corporações. A visão do nosso projeto não diz respeito somente a educar, a ensinar pessoas a programar, mas a garantir que, à medida que essa tecnologia continue a moldar nossa vida, ela atenda aos desejos de todos.

Boa leitura!

## **INTRODUÇÃO**

No dia a dia da maioria das pessoas, os dispositivos móveis (smartphones e tablets) estão cada vez mais presentes e incorporados à sua rotina, o que mostra a importância da disciplina *Programação para Dispositivos Móveis*.

Nesta disciplina, usamos o App Inventor, ferramenta que permite programar, ou melhor, montar aplicativos bastante sofisticados utilizando os recursos embutidos nos aparelhos. Logicamente, o App Inventor não foi projetado para programas específicos ou de alta performance, tanto em termos de processamento quanto em termos gráficos, mas ele possibilita o uso de interfaces externas, bem como a conectividade e a importação de extensão de terceiros.

Este livro-texto mostra a montagem de alguns aplicativos e jogos. Cada um deles é feito com o objetivo de abordar conceitos, tecnologias ou práticas que envolvem o dia a dia da computação móvel, relacionada com a área das ciências da computação e da informação.

O livro é dividido didaticamente em três unidades.

Na unidade I, apresentamos uma introdução ao App Inventor, mostrando o acesso ao ambiente e seus componentes, explicando a instalação, a configuração e o teste do emulador e estudando os componentes do Designer e dos Blocos. Assim, a unidade I focaliza a instalação, a configuração e, principalmente, a compreensão dos ambientes e dos princípios gerais de funcionamento da ferramenta, com a construção de aplicativos simples.

Na unidade II, abordamos a identificação da demanda e a definição das características dos apps e falamos sobre os testes dos aplicativos, incluindo editores de blocos e testes. Logo, a unidade II volta-se para a construção de aplicativos completos, que fazem a interação do mundo exterior de forma transparente com o usuário, e para o estudo de técnicas não intuitivas de programação.

Na unidade III, sobre planejamento de jogos, apresentamos a metodologia de game design document (GDD), explicamos a criação de imagens e de designs para jogos, abordamos a pontuação, a tela de início, o ranking, os créditos, a atribuição de ícone e título, a geração de arquivo APK, a distribuição do jogo e a sua publicação na Play Store. Desse modo, a unidade III centra-se no planejamento e na construção de jogos e avalia seu ciclo de vida, desde a ideia inicial até a publicação.



# Unidade I

Nesta unidade, veremos um pouco da história do App Inventor e suas opções de configuração. Exploraremos o ambiente de trabalho, a interface com a máquina e os conceitos iniciais do processo de programação.

Em 2008, os celulares estavam adquirindo status de smartphones e ganhando sistema operacional (IOS e Android, por exemplo). Antes, os celulares executavam apenas programas que já vinham de fábrica, sem a liberdade de o usuário escolher o que instalar, como se fazia nos computadores. Com a vinda do sistema operacional, abriu-se a possibilidade de executar programas conhecidos hoje como aplicativos. Os aparelhos equipados com processadores velozes, a possibilidade de armazenamento de memória e os diversos sensores – enfim, os smartphones – estavam permitindo que as pessoas interagissem com o mundo como nunca antes. Nessa época, Hal Abelson, professor do Instituto de Tecnologia de Massachusetts (MIT), teve a ideia de criar uma linguagem de programação fácil de usar para fazer aplicativos móveis que aproveitassem o poder da tecnologia emergente dos smartphones. O objetivo de Abelson era democratizar o processo de desenvolvimento de aplicativos móveis, tornando acessível para qualquer pessoa a sua criação. Durante um período sabático no Google em Mountain View, CA, Abelson trabalhou com o engenheiro Mark Friedman e alguns outros desenvolvedores para criar o Google App Inventor (KAMRIANI; ROY, 2016).

Pedagogicamente, o App Inventor segue uma escola que começou com o trabalho de Seymour Papert e do MIT Logo Group na década de 1960 (WOLBER et al., 2011), cuja influência persiste até hoje por meio de muitas atividades, como o Scratch e os programas projetados para apoiar o pensamento computacional, todos unindo computadores e educação.



### Saiba mais

O Scratch é uma linguagem de programação desenvolvida por Mitchel Resnick e sua equipe no grupo de pesquisa Lifelong Kindergarten, no Media Lab do MIT. É próprio para crianças, por ser baseado em blocos de encaixe coloridos que eliminam possíveis erros de sintaxe, o que facilita a conexão correta dos comandos. Permite o desenvolvimento de animações, simulações, jogos digitais, entre outros. É um software livre e está disponível gratuitamente para multiplataforma, podendo ser utilizado offline nas versões Scratch 1.4 e Scratch 2.0, além de possibilitar a criação online na versão 3.0 pela página indicada a seguir (SHIMOHARA; SOBREIRA; ITO, 2016).

*Scratch.* Disponível em: <http://bit.ly/3eCE7ot>. Acesso em: 19 mar. 2021.

Para tornar a estrutura de programação visual próxima da estrutura do Scratch, mas sem "reinventar a roda", a implementação específica do App Inventor foi montada sobre a estrutura do Open Blocks. Posteriormente, o próprio App Inventor serviu como ponto de partida para o Google Blockly (METZ, 2012).



## Observação

O Open Blocks é distribuído pelo Scheller Teacher Education Program do MIT. O compilador que traduz a linguagem de blocos visuais para implementação no Android usa o Kawa Language Framework e o dialeto de Kawa da linguagem de programação Scheme, desenvolvida por Per Bothner, distribuída como parte do sistema operacional GNU pela Free Software Foundation.



### Saiba mais

A Free Software Foundation (FSF) é uma organização sem fins lucrativos fundada para apoiar o movimento do software livre. Ela promove a liberdade universal de estudar, distribuir, criar e modificar softwares de computador, com a preferência da organização por softwares distribuídos sob copyleft (compartilhar da mesma forma), como ocorre com sua própria general public license (GNU). A FSF foi constituída em Boston, Massachusetts, EUA, onde também tem sede. Infelizmente a página do site em português está bem defasada. Conheça mais sobre a FSF no link indicado.

*Free Software Foundation.* Disponível em: <http://bit.ly/2OzZ48L>. Acesso em: 19 mar. 2021.

Em 2011, Abelson levou o App Inventor para o MIT e, juntamente com o Media Lab e o Laboratório de Ciência da Computação e Inteligência Artificial (CSAIL), criou o Center for Mobile Learning. Em dezembro de 2013, Abelson e sua equipe de desenvolvedores lançaram o MIT App Inventor 2, uma versão do aplicativo baseada na web ainda mais fácil de usar e que utiliza um ambiente de desenvolvimento integrado (KAMRIANI; ROY, 2016).

## 1 INTRODUÇÃO AO APP INVENTOR

A plataforma de desenvolvimento do App Inventor é um aplicativo na nuvem, executado em um navegador. Os navegadores recomendados são Chrome, Firefox ou Safari. O próprio site não recomenda o uso do Microsoft Edge, por incompatibilidade não esclarecida.

Para estudarmos esta disciplina, temos de atender a alguns pré-requisitos:

# PROGRAMAÇÃO PARA DISPOSITIVOS MÓVEIS

- PC Windows, Mac ou GNU/Linux conectado à internet;
- conta do Gmail (Google);
- dispositivo Android, que pode ser:
  - smartphone ou tablet conectado ao PC por Wi-Fi;
  - smartphone ou tablet conectado ao PC por cabo USB.
- emulador Android instalado no PC, Mac ou Linux.

De posse dos equipamentos, temos, então, alguns caminhos, em função da disponibilidade e da situação individual, conforme indicado na figura a seguir.

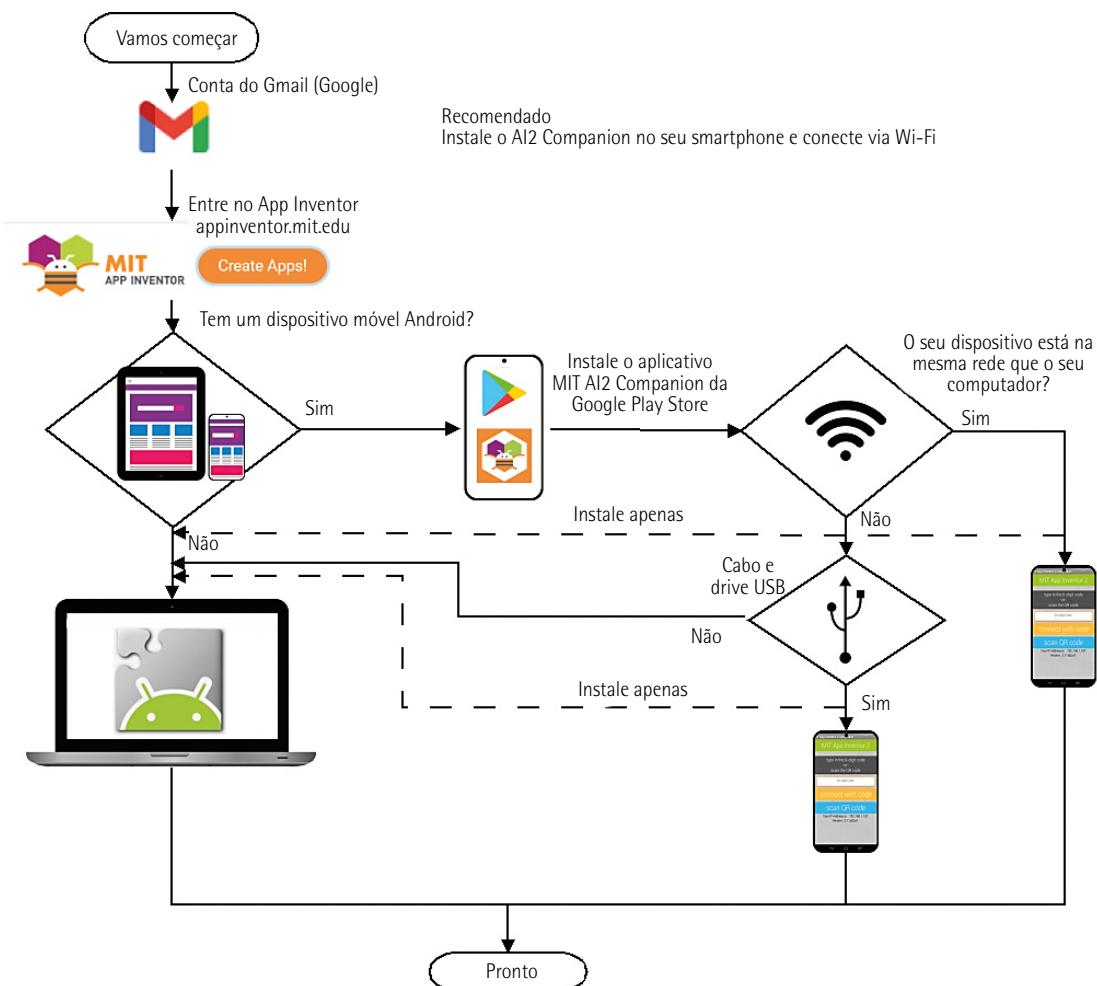


Figura 1 – Fluxo de instalação do App Inventor

As duas primeiras fases do processo de configuração são estas:

- Obtenha uma conta do Gmail ou uma conta do celular Android (se ainda não tiver).
- Registre-se no site do App Inventor.

Vejamos as etapas a seguir.

## Etapa 1: configure uma conta do Gmail

Para construir aplicativos com o App Inventor, você precisará de uma conta do Gmail. Se ainda não tiver uma, no browser, visite o site <http://mail.google.com> e escolha o link para criar uma nova conta. Será aberto o formulário de inscrição do Google. Siga as instruções para configurar sua conta.

## Etapa 2: faça login no App Inventor

O App Inventor é um exemplo de ambiente de desenvolvimento integrado (IDE), o que significa que todas as ferramentas de software necessárias para projetar, desenvolver e testar um aplicativo móvel Android estão integradas na plataforma App Inventor. IDEs são parte do kit de ferramentas-padrão que os programadores usam para desenvolver programas.

No browser, entre no site <http://appinventor.mit.edu>, conforme mostrado na figura a seguir. Será solicitado que você faça login usando suas credenciais do Gmail. Isso é tudo que você precisa para se registrar. Se essa for a sua primeira visita, verá uma caixa de diálogo pop-up informando que você não tem nenhum projeto e sugerindo que crie um novo projeto. Não há necessidade de fazer isso agora. Voltaremos a isso mais adiante. Neste momento, vamos apenas fazer login e continuar a explorar a configuração.

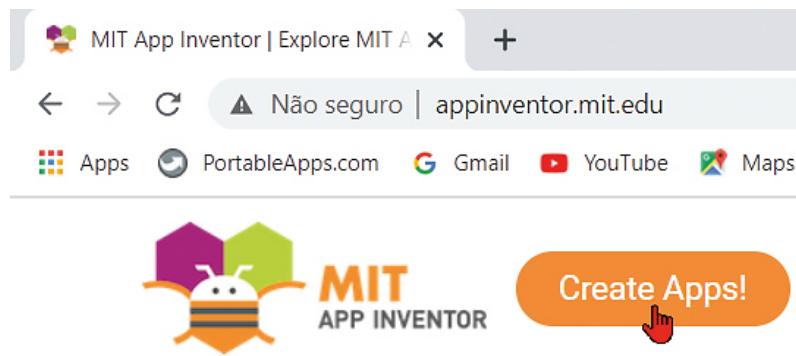


Figura 2 – Entrada no IDE do App Inventor

## Etapa 3: escolha o modo de emulador ou dispositivo móvel

Um dos recursos mais interessantes da plataforma App Inventor é a possibilidade de fazer programação ao vivo, ou seja, a capacidade de executar seu aplicativo em seu dispositivo. Isso permite

# PROGRAMAÇÃO PARA DISPOSITIVOS MÓVEIS

que você acompanhe o progresso sem ter de construir o aplicativo e instalá-lo no dispositivo. Dessa forma, a experiência será melhor com um dispositivo móvel Android, seja um tablet, seja um smartphone conectado na mesma rede via Wi-Fi.

No caso de ter dispositivo móvel, será necessário instalar o aplicativo MIT AI2 Companion. Mesmo sem a disponibilidade de Wi-Fi na mesma rede do computador, é possível utilizar a conexão via USB caso tenha o drive correto do smartphone.

De qualquer forma, o aiStarter deve ser instalado no computador, como veremos adiante.

## 1.1 Acesso ao ambiente de desenvolvimento

Como vimos, para entrar no App Inventor basta digitar appinventor.mit.edu e, ao ser aberta a tela de boas-vindas do MIT App Inventor, clicar no botão **Create Apps**. Veja a figura.



Figura 3 – Na tela inicial do App Inventor, clique em Create Apps

Depois, abrirá a tela de login e senha, conforme mostrado na figura a seguir. Preencha tais requisições com os dados da sua conta do Google.

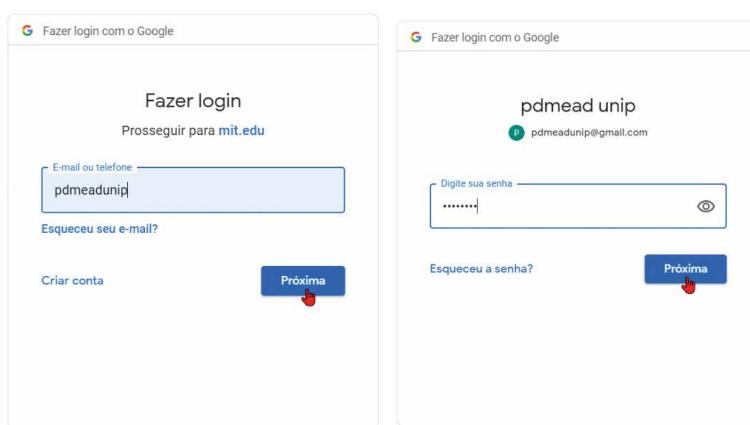


Figura 4 – Telas de login e senha

# Unidade I

No primeiro acesso, aparecerá a tela de contrato de termos de uso. Para aceitar, clique no botão **I accept the terms of service**. Veja a figura.

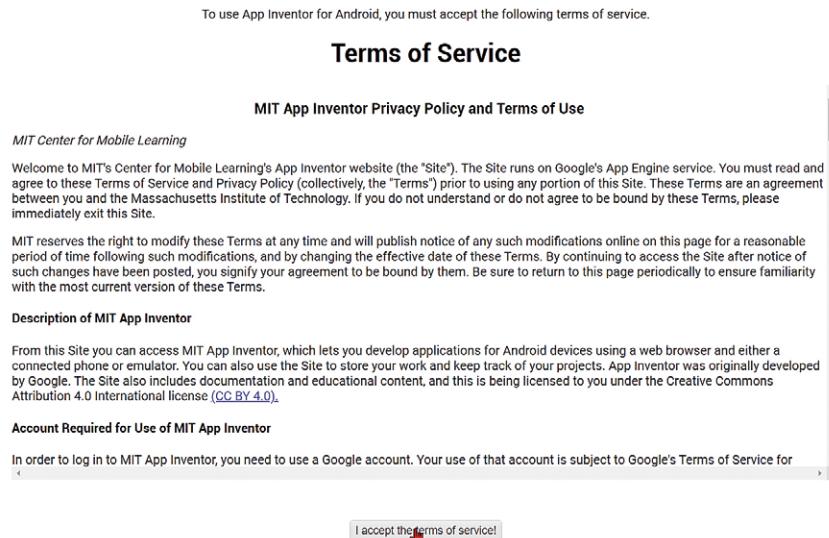


Figura 5 – Tela dos termos de uso do App Inventor

Uma janela de boas-vindas será aberta. Marque a opção **Do Not Show Again** e clique no botão **Continue**. Em seguida, abrirá a janela dos tutoriais. Para fechá-la, clique no botão **Close**. Veja a figura.

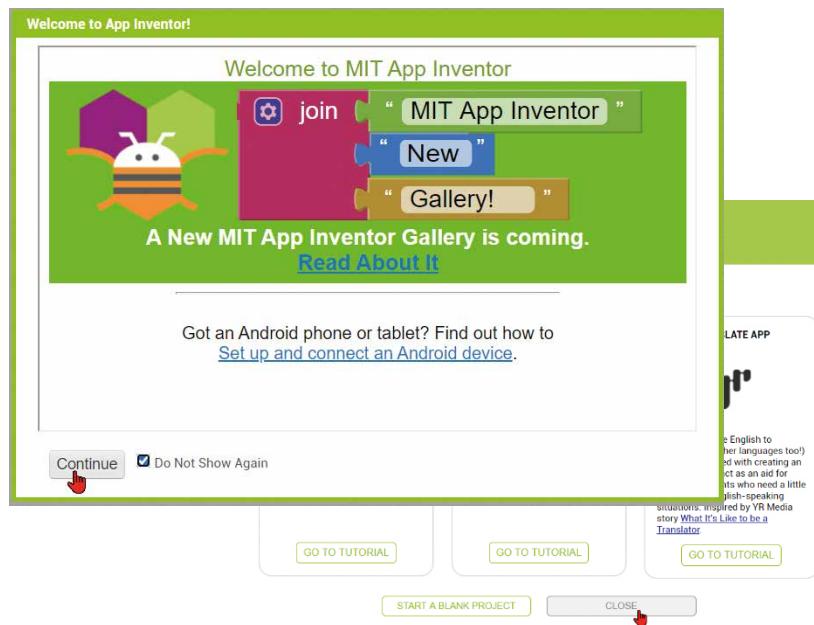


Figura 6 – Janela de boas-vindas e tutoriais

Independentemente de ser ou não o primeiro acesso, após a tela do tutorial, a primeira tela em inglês será apresentada como mostrado na figura a seguir. Essa tela é chamada de **My Projects**. Neste momento, a lista de projetos está vazia, porque estamos iniciando o curso, mas, conforme formos avançando, a lista será preenchida.

# PROGRAMAÇÃO PARA DISPOSITIVOS MÓVEIS

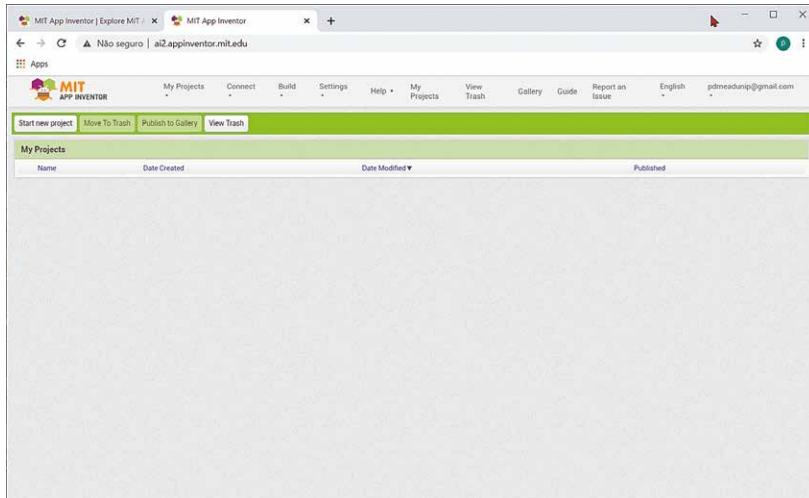


Figura 7 – Tela inicial ou tela My Projects

A primeira providência é configurar o ambiente para o português do Brasil. Na barra de menus, ao clicar em **English**, descerá uma cortina de opções. Escolha **Português do Brasil**. Com a interface alterada, os menus e os blocos em sua maioria estarão traduzidos. Todas as vezes que entrar no App Inventor, essa operação deverá ser feita, pois as nossas lições serão todas trabalhadas em português. Veja a figura.

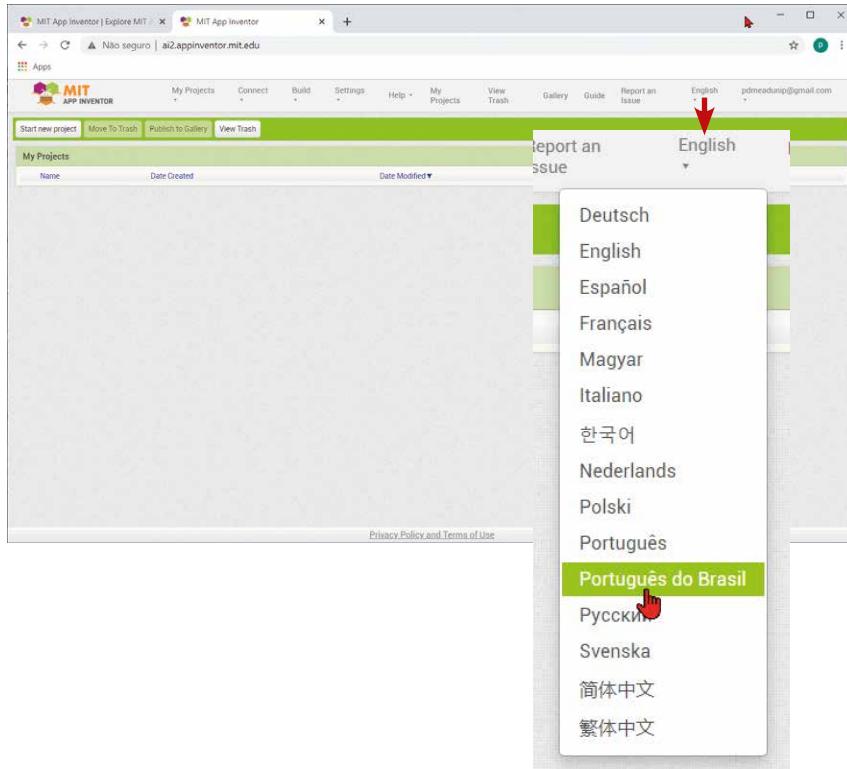


Figura 8 – Alterando o idioma do IDE

Neste momento, estamos prontos para explorar o ambiente de trabalho do App Inventor.

## 1.2 Apresentação dos componentes do ambiente

A primeira tela apresentada é a página Meus Projetos (My Projects), que contém os projetos associados à conta Gmail. A tela da próxima figura apresenta uma lista com as informações de alguns projetos que já foram criados. Nessa tela, temos também os botões para:

- iniciar um novo projeto;
- excluir um projeto assinalado;
- publicar um aplicativo na Galeria dos usuários do App Inventor, a fim de compartilhar o projeto com a comunidade dos desenvolvedores;
- exibir os projetos que estão na lixeira.

O último botão serve para recuperar um projeto que foi apagado acidentalmente. A barra superior de menu disponibiliza outras funcionalidades – por exemplo, mudar o idioma para o português do Brasil, como fizemos anteriormente.

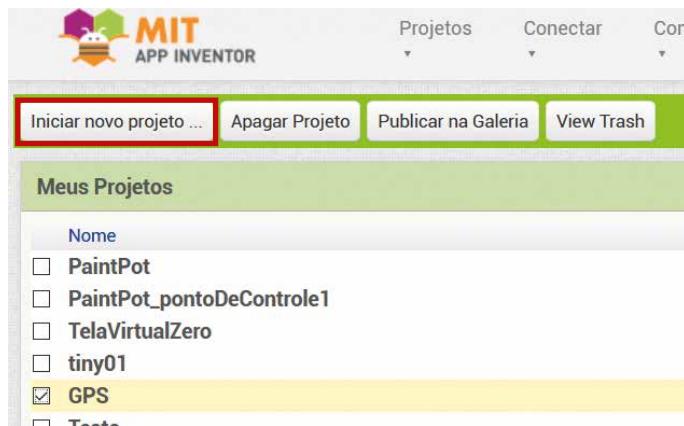


Figura 9 – Tela Meus Projetos com alguns projetos já gravados

Para iniciar, clique no botão **Iniciar novo projeto**. Assim, será aberta a janela para dar nome ao projeto. Veja a figura.



Figura 10 – Batizando um novo projeto

# PROGRAMAÇÃO PARA DISPOSITIVOS MÓVEIS

Desse modo, entramos na tela de edição e criação, que está inicialmente vazia. Veja a figura.

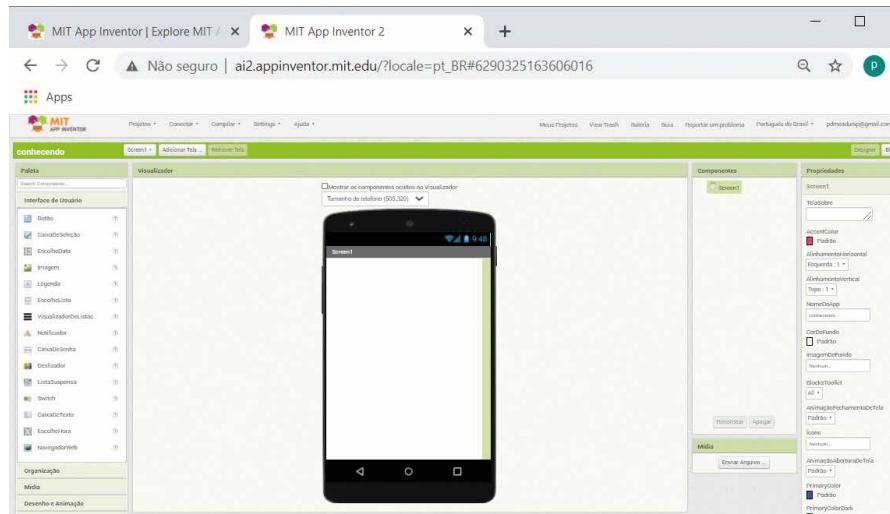


Figura 11 – Tela do Designer vazia

A interface do MIT App Inventor tem duas telas principais:

- a tela do Designer, interface gráfica do usuário (GUI) em que é criada a aparência do aplicativo (e os componentes extras que o aplicativo necessite);
- a tela dos Blocos, em que é definido o comportamento do aplicativo codificando-o com blocos coloridos.

Ao criarmos um novo projeto, entramos direto na tela do Designer.

A navegação entre as interfaces é feita pelos botões localizados na parte superior direita da tela. Veja a figura a seguir.

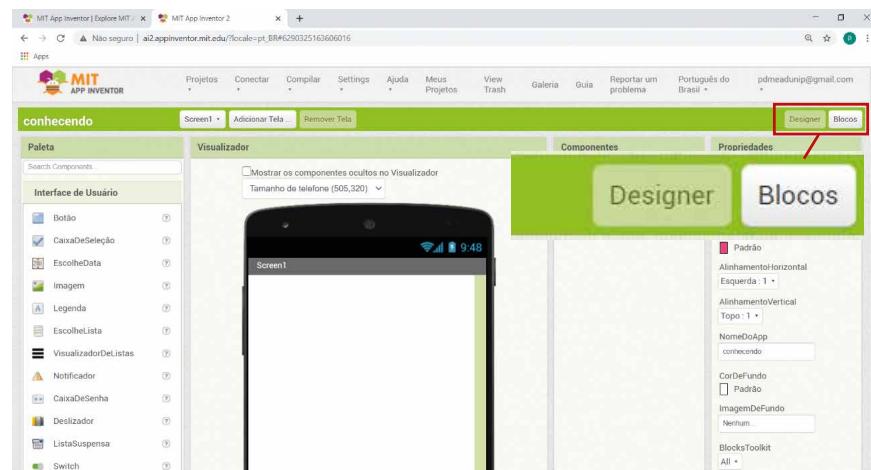


Figura 12 – Botões de alternância entre as telas do Designer e dos Blocos

# Unidade I

No editor de design, mostrado na figura a seguir, você seleciona visualmente os componentes e faz o layout das telas do aplicativo. No editor de blocos, esses componentes estão disponíveis para programação. Cada componente (por exemplo, botão) na tela apresenta um conjunto de propriedades predefinidas e manipuladores de eventos que são mostrados na forma de blocos.

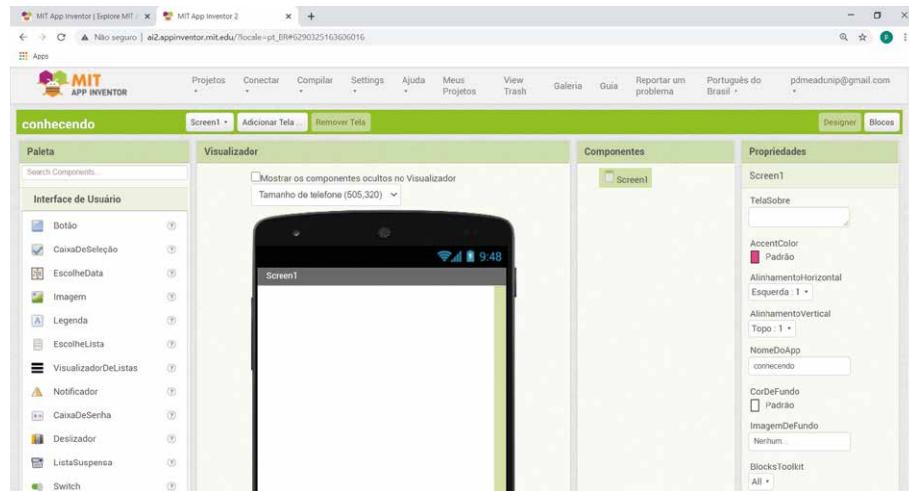


Figura 13 – Tela do Designer

É no editor de blocos que se programa a lógica dos aplicativos e dos seus comportamentos. Nesse editor, você verá todos os componentes que foram adicionados no editor de design. O editor de blocos oferece uma interface de arrastar e soltar para programar o comportamento do aplicativo juntando blocos semelhantes às peças de um quebra-cabeça. Veja a figura.

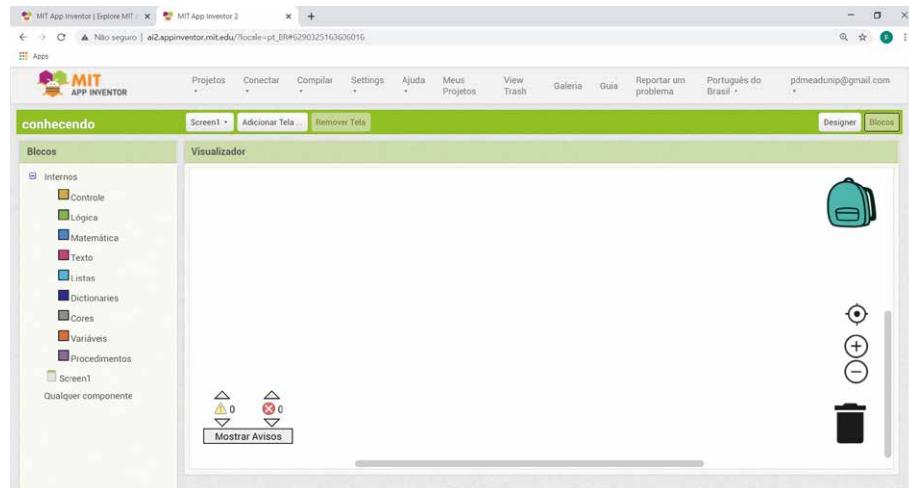


Figura 14 – Tela dos Blocos

Assim, conseguimos obter uma ideia geral do ambiente.

## 1.3 Instalação e configuração do emulador

Neste item, vamos realizar a instalação do emulador. Em primeiro lugar, o que é um emulador? É um programa que serve para simular a experiência de executar um sistema operacional diferente dentro de outro sistema de computador, como o Windows. No nosso caso, emulamos não só um sistema operacional, mas também um celular Android.

Instalado o programa aiStarter no computador, instala-se o software App Inventor Setup. Ele faz a comunicação entre o App Inventor que está sendo executado no navegador e outras partes do App Inventor. Sempre que quiser usar o emulador ou o cabo USB, é preciso se certificar de que o aiStarter está ativo. Esse programa permite que o navegador se comunique com o emulador ou com o cabo USB. Você não precisa do aiStarter se estiver usando apenas o AI2 Companion com Wi-Fi.

Para instalar o emulador, é necessário retornar à página appinventor.mit.edu e escolher o link **Get Started** ou, no menu superior, clicar em **Resources** e, depois, em **Get Started**. Veja a figura.

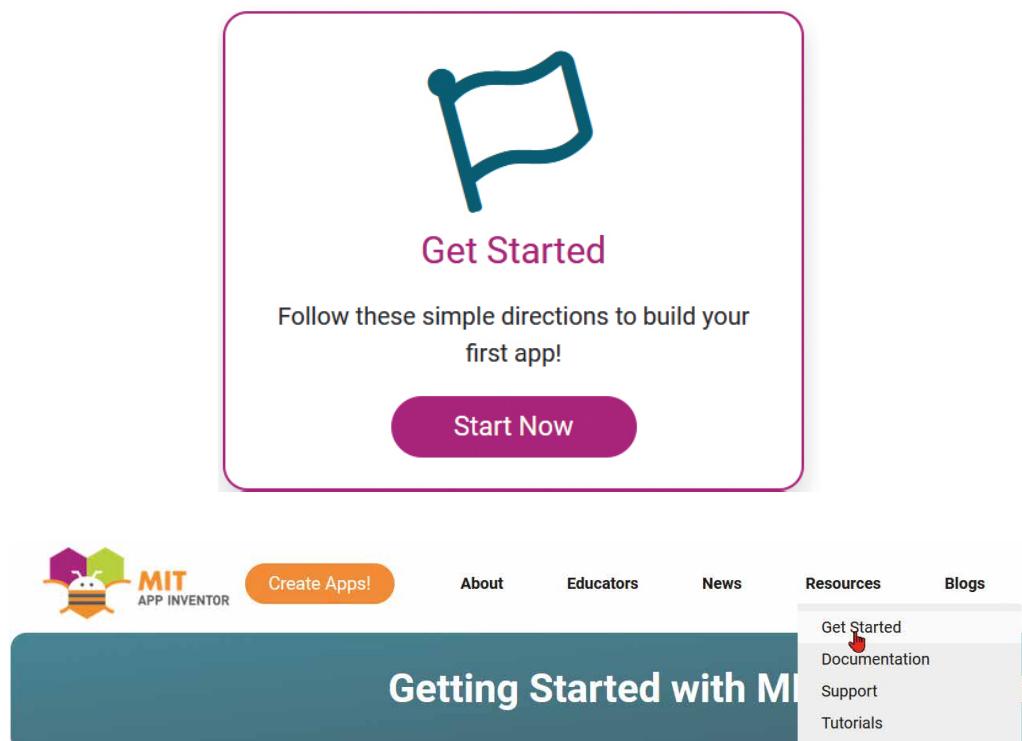


Figura 15 – Acesso para a página de inicialização

Na tela **Getting Started**, clique na opção **Setup Instructions**, em que será feita a escolha do tipo de instalação. Veja a figura a seguir.

# Unidade I

The screenshot shows the MIT App Inventor website. At the top, there's a navigation bar with links for 'About', 'Educators', 'News', 'Resources', 'Blogs', 'Donate', and 'ENHANCED'. On the left, there's a logo for 'MIT APP INVENTOR' featuring a stylized bee icon. A prominent orange button labeled 'Create Apps!' is visible. Below the navigation, a large teal header bar says 'Getting Started with MIT App Inventor'. Underneath, a text block explains that App Inventor is a cloud-based tool for building apps in a web browser, with a link to 'a2.appinventor.mit.edu'. A numbered list starts with '1. Setup Instructions: Set up your phone or tablet for live testing (or start the emulator if you don't have a mobile device)'.

Figura 16 – Instruções de configuração na tela de inicialização

A Opção Um (Option One), recomendada, destina-se ao caso em que se tem um dispositivo Android. Basta fazer uma busca na Google Play Store e instalar o aplicativo MIT AI2 Companion. Uma vez instalado, execute a primeira vez e entre no aplicativo. Veja a figura.

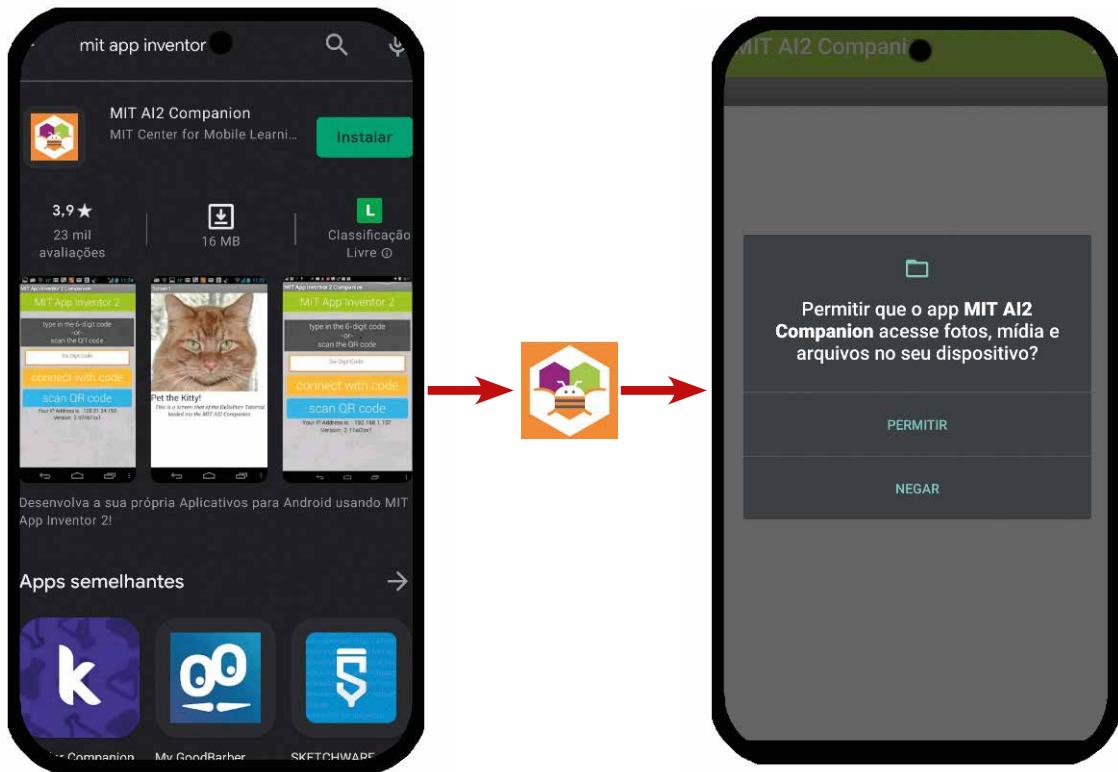


Figura 17 – Instalação do AI2 Companion e liberação dos seus recursos no sistema operacional

No computador, entre na tela do Designer e, no menu superior, ao clicar em **Conectar**, escolha **Assistente AI**. É com essa opção que será feita a ligação entre o App Inventor e o dispositivo móvel sempre que estiverem na mesma rede. Veja a figura a seguir.

# PROGRAMAÇÃO PARA DISPOSITIVOS MÓVEIS

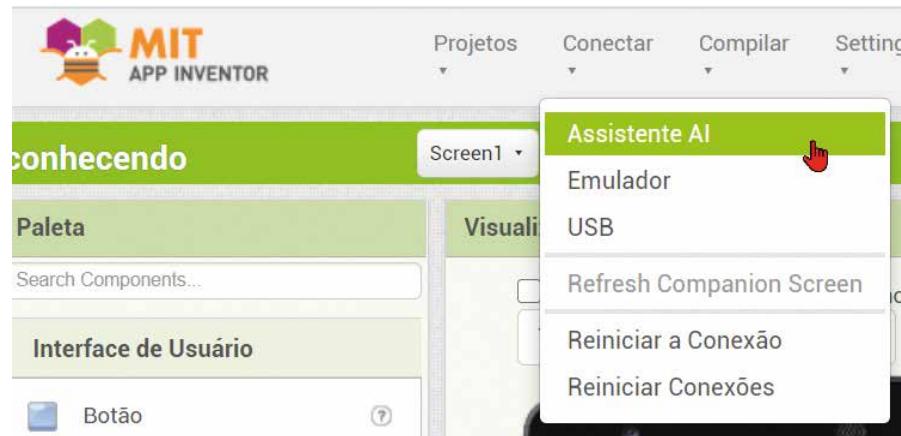


Figura 18 – A opção para ativar a ligação entre o navegador e o celular



## Lembrete

Nessa opção, é necessário que o computador em que o App Inventor está sendo executado e o dispositivo móvel com o AI2 Companion instalado estejam na mesma rede, ou seja, conectados ao mesmo roteador.

Ao clicar na opção Assistente AI, uma janela com um QR code e um código se abrirá. No dispositivo móvel, selecione a opção **scan QR code**. Caso seja o primeiro acesso, será aberta uma janela no dispositivo para autorizar que o AI2 Companion tire fotos e grave vídeos. Autorize isso clicando em **Permitir**. Veja a figura.

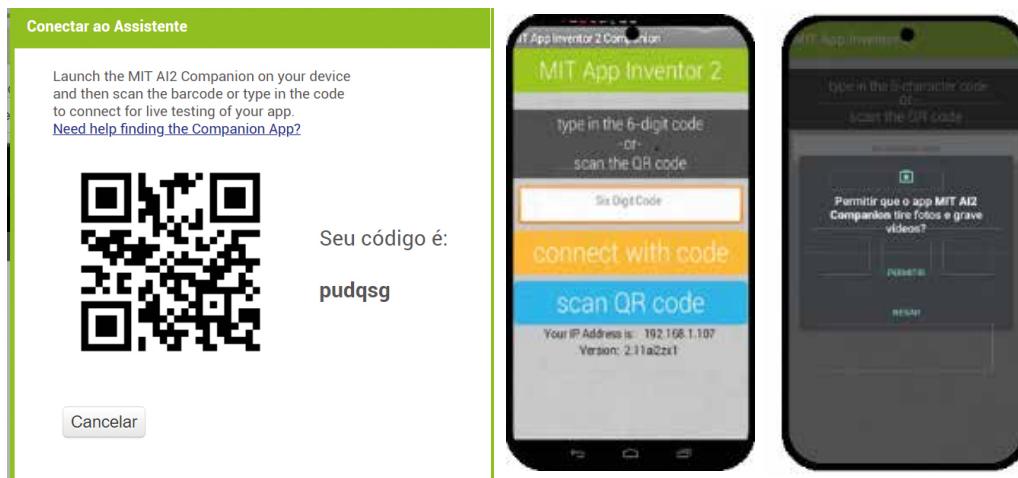


Figura 19 – Conexão entre o projeto e o smartphone via Wi-Fi

Ao posicionar o dispositivo sobre o QR code, conforme mostrado na figura a seguir, se o IDE e o emulador estiverem na mesma rede, o dispositivo irá espelhar a tela do visualizador da tela do Designer.



Figura 20 – Leitura do QR code no dispositivo móvel

Com isso, em princípio, já é possível começar a desenvolver projetos no App Inventor. Essa é a melhor e a mais simples configuração para trabalhar com o App Inventor.

As configurações mostradas a seguir são alternativas à instalação feita. Por segurança, é aconselhável deixar instalado o emulador para lidar com quaisquer problemas futuros.

### Instalação: emulador

A partir da página de configuração, no quadro da Opção Três (Option Three), clique no link **Instructions** para instalar o emulador. Veja a figura.

### Option Three Don't have an Android device? Use the Emulator: [Instructions](#)

If you don't have an Android phone or tablet handy, you can still use App Inventor. Have a class of 30 students? Have them work primarily on emulators and share a few devices.



Build your project on your computer      Test it in real-time on your computer with the onscreen emulator

Figura 21 – Escolha a instalação do emulador

# PROGRAMAÇÃO PARA DISPOSITIVOS MÓVEIS

Na página que abrir, clique no link **Instructions for Windows**, que está na guia **Step 1. Install the App Inventor Setup Software**. Na página seguinte, estará o link para baixar o arquivo de instalação do emulador. Veja a figura.

## Step 1. Install the App Inventor Setup Software

- [Instructions for Mac OS X](#)
- [Instructions for Windows](#)
- [Instructions for GNU/Linux](#)

Figura 22 – Localização do link para a página do download

Na página seguinte, clique em **Download the Installer** e baixe o arquivo do instalador. Uma vez baixado o programa, garanta que salvou tudo que estiver aberto e feche todos os outros programas, pois o computador será reiniciado ao final do processo. Execute a instalação do arquivo **MIT\_App\_Inventor\_Tools\_x.x.x\_win\_setup.exe** como administrador. Conforme as telas de instalação forem se sucedendo, escolha o botão **Next** ou o botão **I agree**, conforme o caso. Na última tela, permita fazer o **Reboot Now**, que é a reinicialização do computador, como mostrado na figura a seguir.

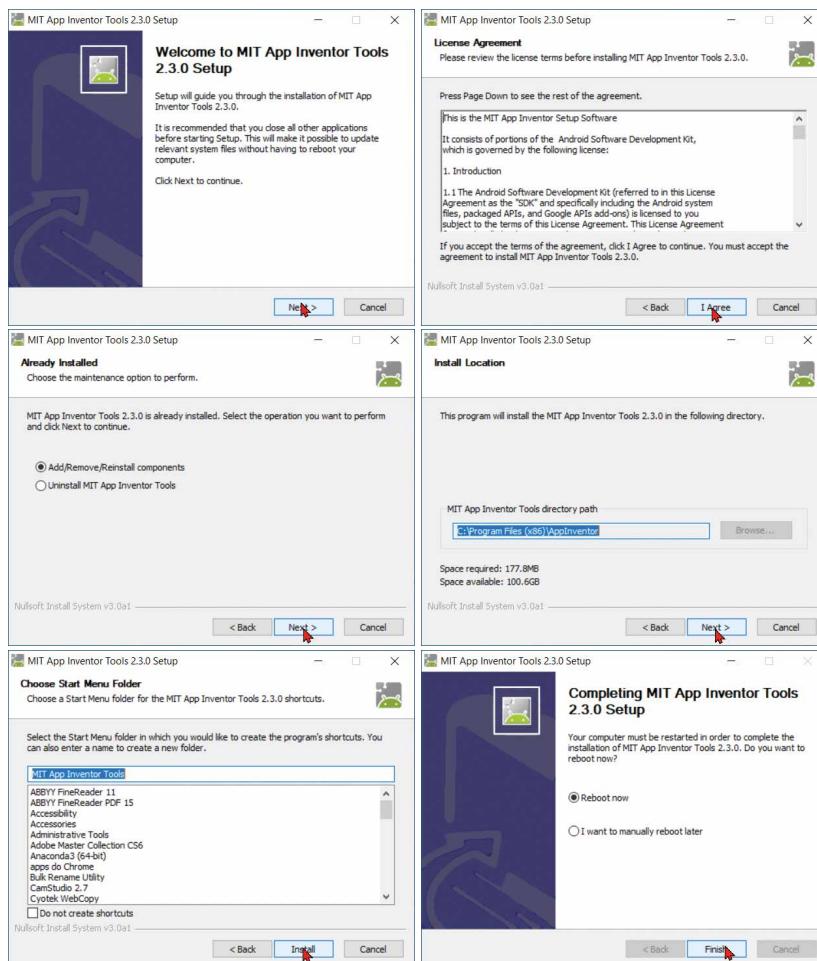


Figura 23 – Sequência de instalação do aiStarter

Com o computador devidamente reiniciado, o ícone do aiStarter deverá aparecer na área de trabalho do seu computador. Ao dar um clique, o aiStarter já estará ativo para o uso. Nenhuma tela irá aparecer, pois o software é executado em segundo plano. A indicação da sua atividade é o ícone na barra de trabalho. Veja as figuras a seguir.

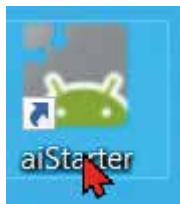


Figura 24 – Ícone do aiStarter

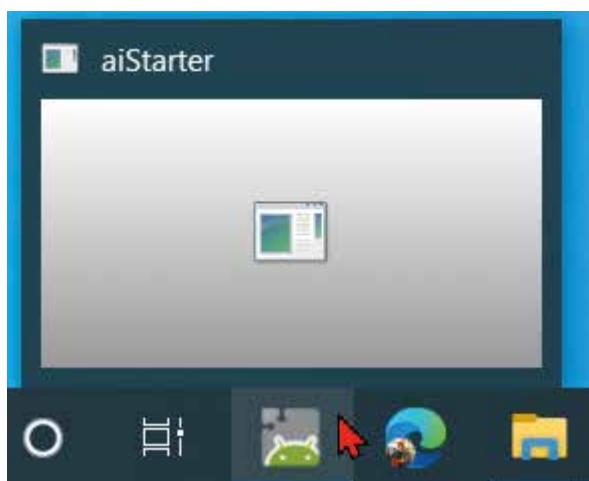


Figura 25 – aiStarter na barra de trabalho

### Observação

Ao dar um clique no ícone do aiStarter na barra de trabalho, abre-se uma janela do prompt do DOS, conforme mostrado na figura a seguir. É nele que está o programa ativo. O programa é um servidor web virtual, que recebe as instruções do navegador. Para encerrar o aiStarter, é importante que seja dado Control + C nessa tela, pois, se fizermos o fechamento simplesmente clicando no botão em forma de X, o processo continuará funcionando em segundo plano.

A imagem mostra uma janela de terminal com o título "aiStarter". O conteúdo da janela é o seguinte:

```
Platform = Windows
AppInventor tools located here: "C:\Program Files (x86)"
Bottle server starting up (using WSGIRefServer())
Listening on http://127.0.0.1:8004/
Hit Ctrl-C to quit.
```

Figura 26 – Tela do aiStarter

# PROGRAMAÇÃO PARA DISPOSITIVOS MÓVEIS

Até a edição deste livro, o emulador interno do aiStarter tem vindo com o Companion desatualizado. Dessa forma, é preciso um cuidado especial e um pouco de paciência e atenção. Não existe nada crítico, mas a desatenção pode acarretar a necessidade de refazer todo o processo, o que resulta em perda de tempo.

Uma vez ativado o aiStarter, abra o site do App Inventor e entre na tela do Designer. Nessa tela, no menu superior, escolha **Conectar** e, no menu suspenso, **Emulador**. O item Conectar do menu superior permite escolher o meio para acompanhar e simular em tempo real o aplicativo que está sendo desenvolvido no App Inventor. No caso em que escolhemos o emulador, o App Inventor envia uma mensagem ao servidor virtual que processa e ativa o emulador. Veja a figura.

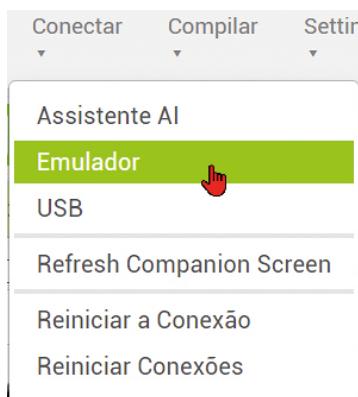


Figura 27 – Menu suspenso da conexão entre o site e a interface do dispositivo móvel

Uma janela será aberta avisando que o processo irá demorar. Nesse momento, o emulador estará sendo carregado e executado no computador. A indicação da execução do emulador é o ícone na barra de trabalho. Veja as figuras.

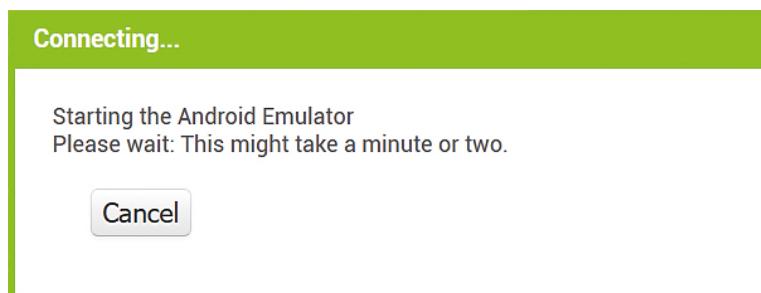


Figura 28 – Janela que se abre enquanto o emulador é ativado

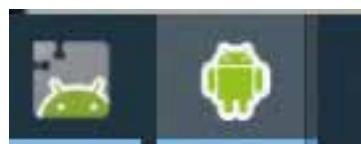


Figura 29 – Ícones do aiStarter e do emulador

Ao ser carregado, o emulador verifica se o AI Companion instalado está atualizado. Normalmente, não está. Clique em **OK** para permitir a atualização. Veja a figura.

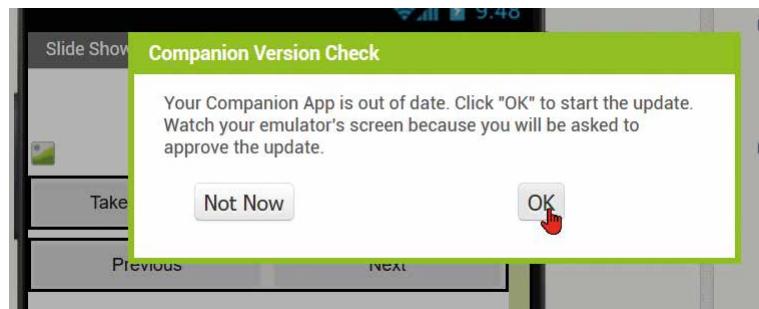


Figura 30 – Permissão para atualizar o AI Companion do emulador

Simultaneamente, no emulador, uma janela será aberta para permitir a troca do aplicativo por um mais novo. Permita a troca dando um clique no botão **OK** e, depois, no botão **Install**. Veja a figura.

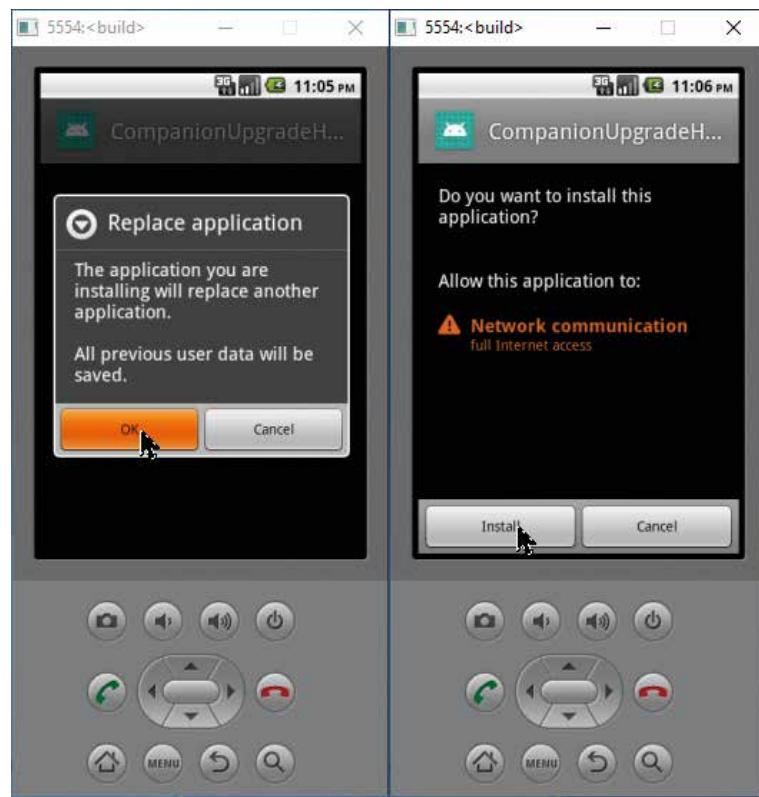


Figura 31 – Autorização para a troca de versão da instalação

A próxima mensagem é importante. Veja a figura a seguir.

# PROGRAMAÇÃO PARA DISPOSITIVOS MÓVEIS

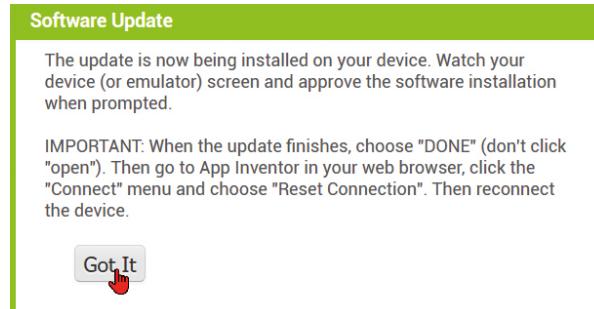


Figura 32 – Aviso importante durante a instalação

Seu conteúdo em português significa o que se traduz a seguir.

A atualização está sendo instalada agora no seu dispositivo. Observe a tela do seu dispositivo (ou emulador) e aprove a instalação do software quando solicitado.

Importante: quando a atualização terminar, escolha **Done** (não clique em **Open**). Em seguida, vá para o App Inventor em seu navegador da web. Clique no menu **Conectar** e escolha **Reiniciar a Conexão**. Em seguida, reconecte o dispositivo.

Conforme o aviso dado, quando for informado que a aplicação está instalada, deve-se clicar no botão **Done**. Veja a figura.

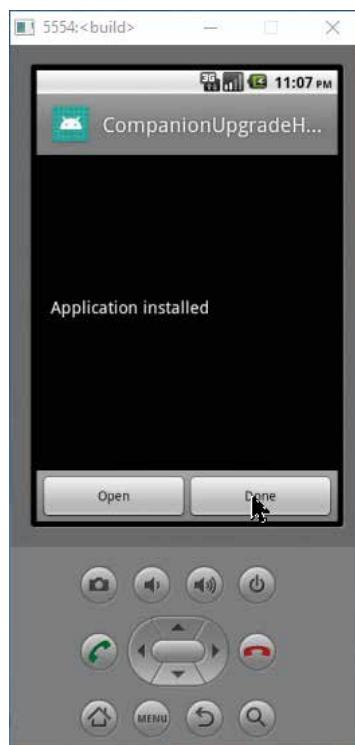


Figura 33 – É importante clicar em Done quando a aplicação estiver sendo instalada pela primeira vez

O emulador será fechado. Como recomendado anteriormente, no navegador, no menu superior, na opção **Conectar**, acione **Reiniciar a Conexão**. Veja a figura.

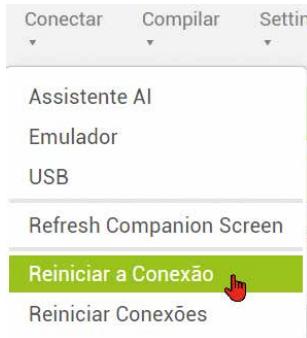


Figura 34 – Opção Reiniciar a Conexão

Neste momento, é importante estudar a diferença entre as opções Reiniciar a Conexão e Reiniciar Conexões. Deve-se tomar muito cuidado, pois a segunda opção (Reiniciar Conexões) restaura o aiStarter para a condição inicial. Isso significa que toda a atualização realizada no AI Companion será perdida. A primeira opção (Reiniciar a Conexão), diferentemente, apenas encerra o emulador.

Reiniciada a conexão, abra novamente o emulador e espere a tela para a instalação do novo emulador. Não se preocupe com a tela avisando o tempo para abrir o emulador.

Por diversas vezes, conforme a velocidade da conexão da internet, a página do App Inventor poderá apresentar aviso de falha de conexão. Nesse caso, clique no botão **Keep Trying** (Continue Tentando). Veja as figuras a seguir.

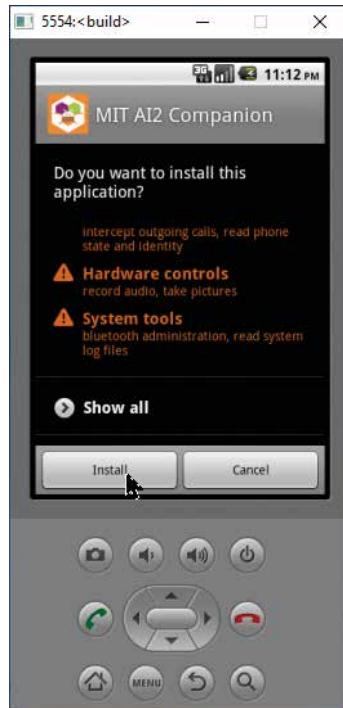


Figura 35 – O Android pedirá a autorização para instalar o novo AI Companion

# PROGRAMAÇÃO PARA DISPOSITIVOS MÓVEIS

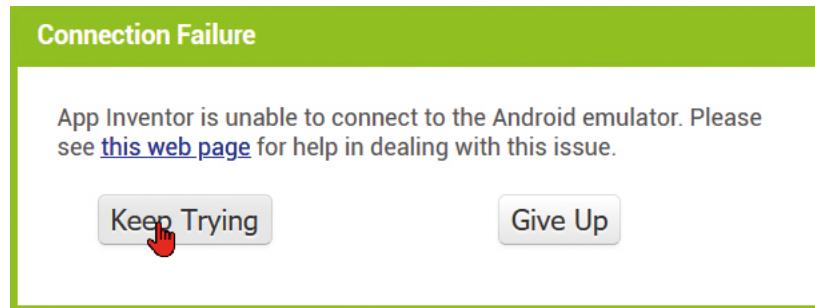


Figura 36 – Mensagem de falha de conexão

Esse processo pode ser demorado e acidentado, mas a instalação é cumulativa. Isso significa que aquilo que já foi baixado no emulador não será perdido. Muitas vezes, a tela do emulador poderá apresentar mensagens de erro. Caso erros apareçam na tela do emulador, reinicie a conexão, espere o emulador fechar e abra-o novamente, enquanto a instalação é retomada. Se a internet for adequada, não deverá acontecer nenhum problema.

Ao terminar a instalação, a mesma tela mostrada antes reaparece, mas com o aplicativo instalado ("Application installed"). Assim, pode-se acionar o botão **Open**. Veja a figura.

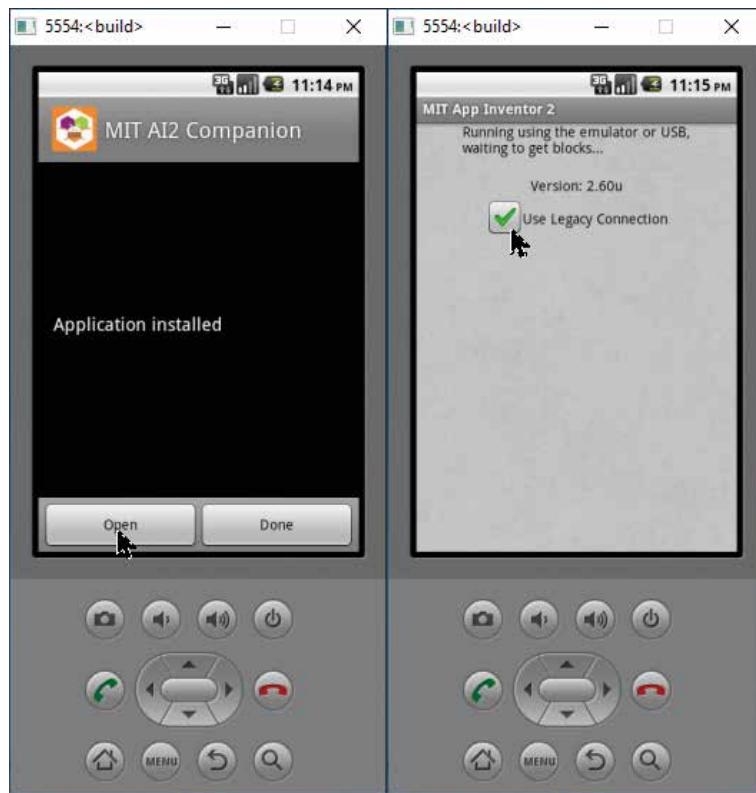


Figura 37 – Finalizando a atualização do emulador

Assim, o emulador estará pronto para uso com a tela em branco. Veja a figura a seguir.

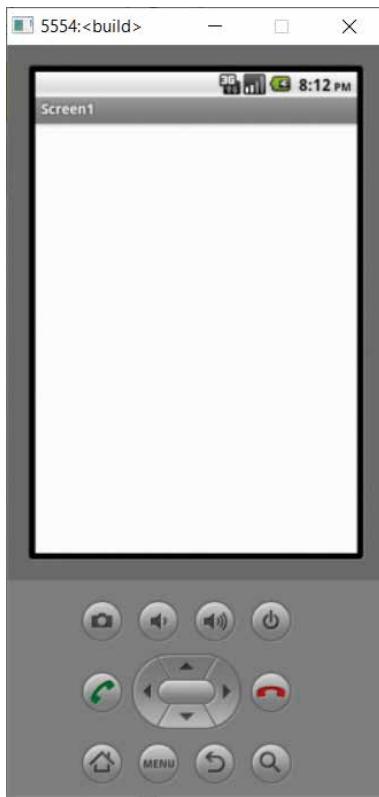


Figura 38 – Emulador ativo com a tela em branco

Em caso de uso do cabo USB para conectar com o computador, a instalação do aiStarter é a mesma. No dispositivo, é preciso permitir a depuração do USB. Para isso, é necessário seguir estes passos:

- Entre na configuração.
- Clique em **Sobre o Dispositivo**.
- Clique sete vezes sobre o número da versão.

Neste momento será ativado o modo desenvolvedor no dispositivo. Assim, é possível concluir a sequência de passos com as opções a seguir:

- sistema (no menu principal da configuração);
- avançado;
- opções de desenvolvedor;
- depuração USB ativar.

A sequência pode mudar conforme a versão e o fabricante do dispositivo.

# PROGRAMAÇÃO PARA DISPOSITIVOS MÓVEIS

## 1.4 Teste do emulador

Para fazer o teste do emulador no browser, digite appinventor.mit.edu/test na barra de endereço do navegador. Veja a figura.

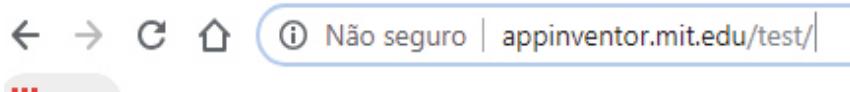


Figura 39 – Chamada do teste na homepage do App Inventor

Se tudo estiver certo, deverá aparecer a mensagem "YES, aiStarter [...] running". Veja a figura.

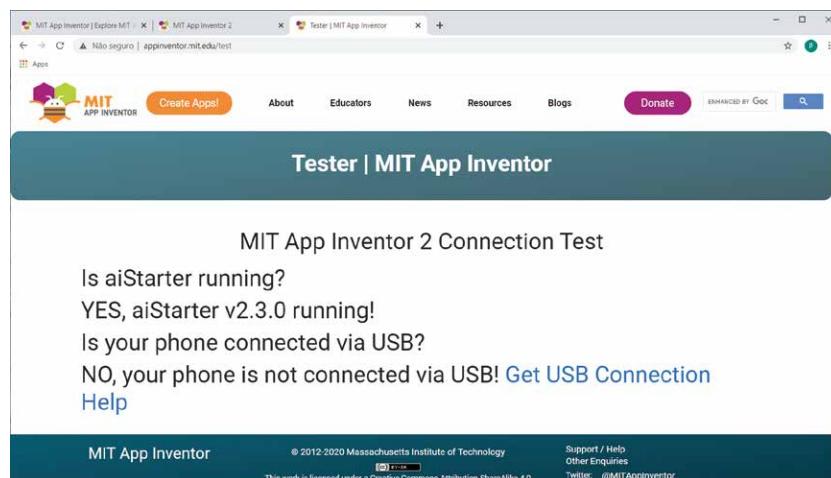


Figura 40 – Resultado do teste de funcionamento do aiStarter

Caso o dispositivo esteja conectado por meio do cabo USB, aparecerá a mensagem "YES, your phone is connected via USB". Veja a figura.

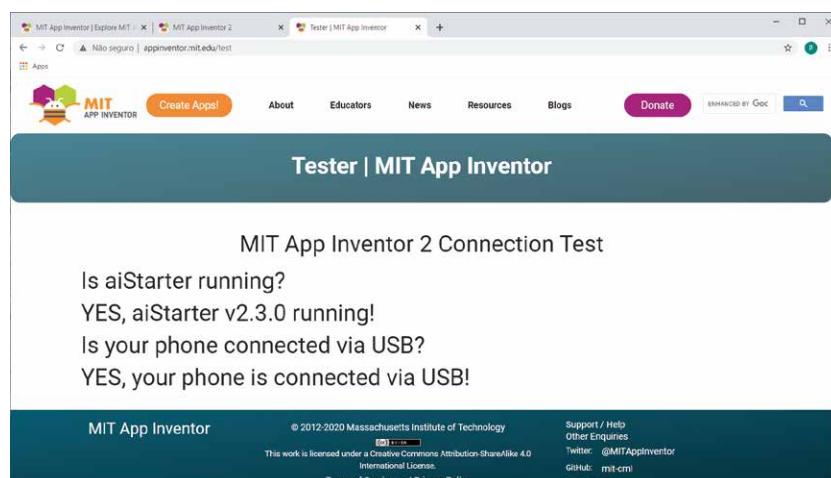


Figura 41 – Resultado do teste com o dispositivo funcionando com a conexão USB

Um problema comum em redes muito fechadas está nas portas fechadas. O App Inventor utiliza a porta 5554. Portanto, ela deve estar aberta.

## 2 DESIGNER

Vamos estudar a estrutura interna do App Inventor para compreender como projetar os aplicativos. A melhor maneira de descrever a construção de um aplicativo é separá-lo em duas partes: seus componentes e seus comportamentos. No App Inventor, as janelas Designer correspondem ao projeto dos objetos (componentes) do aplicativo, e o editor de blocos corresponde à programação de como o aplicativo responde ao usuário e aos eventos externos (comportamento do aplicativo). Veja a figura.

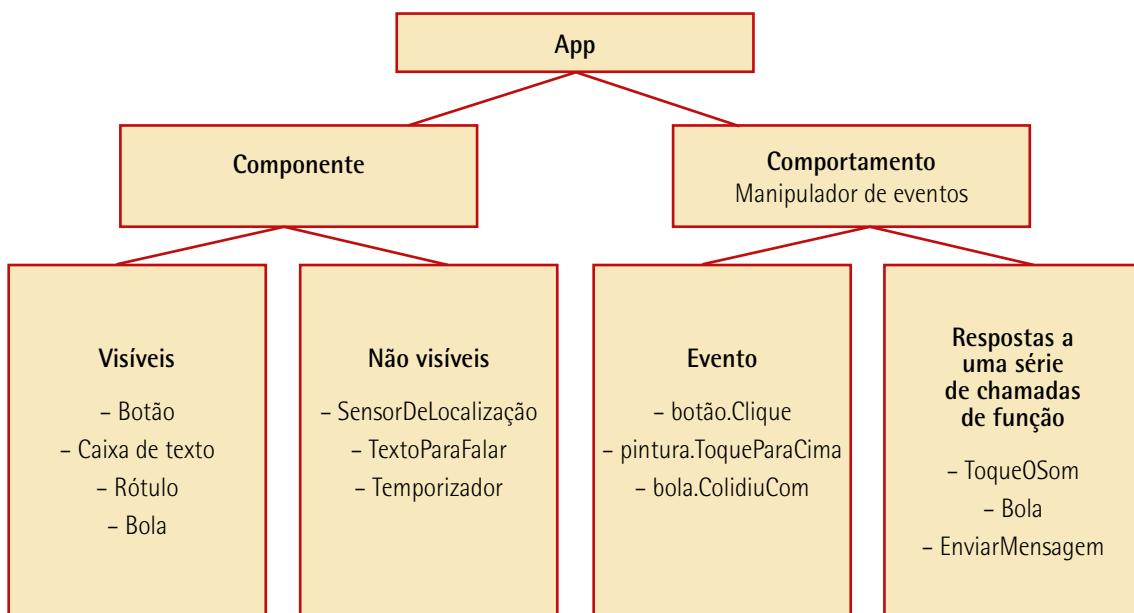


Figura 42 – Estrutura interna do App Inventor

Neste item do livro-texto, vamos estudar a interface do ramo dos componentes, ou seja, o modo Designer.

Temos dois tipos principais de componentes em um aplicativo:

- componentes visíveis;
- componentes não visíveis.

Os componentes visíveis do aplicativo são aqueles que você pode ver quando o aplicativo é iniciado: botões, caixas de texto e rótulos. Também podem ser chamados de interface do usuário.

Os componentes não visíveis são aqueles que você não pode ver. Portanto, eles não fazem parte da interface do usuário; funcionam nos "bastidores" do aplicativo. Eles dão funcionalidade aos componentes integrados do dispositivo. Por exemplo, o componente **SensorDeLocalização**

funciona como um GPS e determina a localização do dispositivo, e o componente **TextoParaFalar** converte texto em fala. Os componentes não visíveis são a tecnologia embarcada no dispositivo.

Ambos os tipos de componente, visíveis e não visíveis, são definidos por um conjunto de propriedades. Propriedades são espaços de memória para armazenar informações sobre o componente. Componentes visíveis, como botões e rótulos, têm propriedades como largura, altura e alinhamento, que definem sua aparência.

As propriedades dos componentes são campos que podem ser alterados no Designer para definir sua aparência inicial. Você também pode alterar os valores com a montagem dos blocos.

## 2.1 Janelas

A tela do Designer está dividida em cinco janelas, descritas a seguir e mostradas na próxima figura.

- **Paleta:** apresenta todos os componentes disponíveis de interface do usuário. Os componentes são agrupados em categorias. Você pode arrastar e soltar qualquer componente no Visualizador. Os componentes são as peças do seu aplicativo que executam ações.
- **Visualizador:** mostra como será o aplicativo. Ele é usado para projetar a interface do usuário do aplicativo e como ela será apresentada no dispositivo. Os componentes da Paleta podem ser colocados no Visualizador. Se o componente for visível, ele será colocado na tela do smartphone e ficará visível para o usuário. Se o componente não for visível, ele será colocado logo abaixo do Visualizador.
- **Componentes:** mostra todos os componentes que foram adicionados ao aplicativo. Os componentes que foram arrastados da Paleta e colocados no Visualizador serão exibidos na janela Componentes. Essa janela também mostrará a organização (ordem, relacionamento pai-filho) dos componentes. Nesta seção, um componente pode ser selecionado, excluído ou renomeado.
- **Mídia:** exibe os arquivos de mídia carregados adicionalmente para uso do aplicativo, como imagens e sons.
- **Propriedades:** permite visualizar ou alterar qualquer uma das propriedades (características) do componente atualmente selecionado. O estado inicial de tais propriedades pode ser definido nessa janela. No entanto, muitas dessas propriedades também podem ser definidas dinamicamente durante a execução do programa (algo que será abordado posteriormente).

# Unidade I

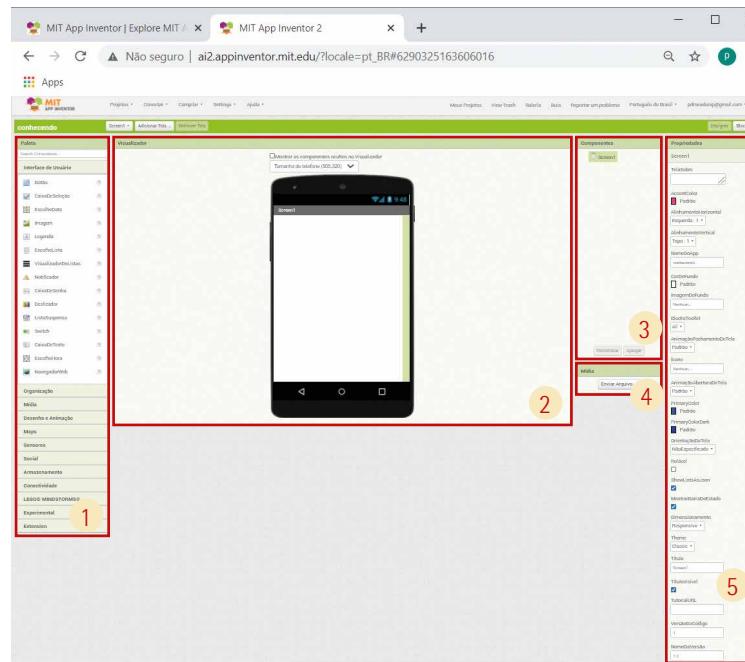


Figura 43 – Tela do Designer e suas cinco janelas: 1) Paleta; 2) Visualizador; 3) Componentes; 4) Mídia; 5) Propriedades

As janelas estão conectadas. Uma peça arrastada da Paleta para o Visualizador automaticamente ocupa o seu lugar na árvore da janela Componentes, e a janela Propriedades é preenchida com os dados do componente. Veja a figura.



Figura 44 – Dinâmica da interação entre as janelas

# PROGRAMAÇÃO PARA DISPOSITIVOS MÓVEIS

## 2.1.1 Paleta

A Paleta inclui componentes de interface do usuário, aqueles que aparecem na tela do dispositivo. Os componentes são as peças do seu aplicativo que executam ações para você. O ponto inicial de tudo que será utilizado no aplicativo é a Paleta, pois é nela que estão todos os componentes.

Na figura a seguir, é possível ver a região da tela denominada Paleta, em que encontramos itens que podem ser adicionados à tela. A Paleta está subdividida em categorias, conforme indicado no próximo quadro.



Figura 45 – Paleta ou interface com o usuário

**Quadro 1 – Funcionalidades das guias da Paleta**

Guia	Finalidade
Interface de Usuário	Contém componentes visíveis com os quais o usuário pode interagir, como botão, rótulo e imagem
Organização	Contém componentes de layout que são usados para organizar os componentes
Mídia	Contém componentes não visíveis, como câmera, filmadora, SoundRecorder e TextToSpeech
Desenho e Animação	Contém componentes que podem ser usados para animação
Maps	Permite a inclusão de mapas para possibilitar ao usuário do aplicativo interagir com eles
Sensores	Inclui componentes que podem se comunicar com os sensores Android, como bússola e acelerômetro

Guia	Finalidade
Social	Inclui componentes sociais, como mensagens de texto e Twitter
Armazenamento	Contém recursos de armazenamento para a leitura ou para a gravação de dados
Conectividade	Inclui componentes para a conexão com a web ou com outros aplicativos; nesta área, o programador pode adicionar as funções relacionadas ao bluetooth do smartphone
Lego® Mindstorms®	Inclui componentes que permitem criar uma relação entre o aplicativo e os equipamentos da Lego, que fornecem controle de robôs Lego Mindstorms NXT
Experimental	Por enquanto, inclui recursos de armazenamento de dados na internet, ainda em caráter experimental
Extension	Permite incluir funções desenvolvidas por outros programadores ao aplicativo, como a barra deslizante de menu

## Paleta: Interface de Usuário e Organização

Os dois principais grupos da Paleta que têm componentes visíveis são a Interface de Usuário e a Organização. Veja a figura.

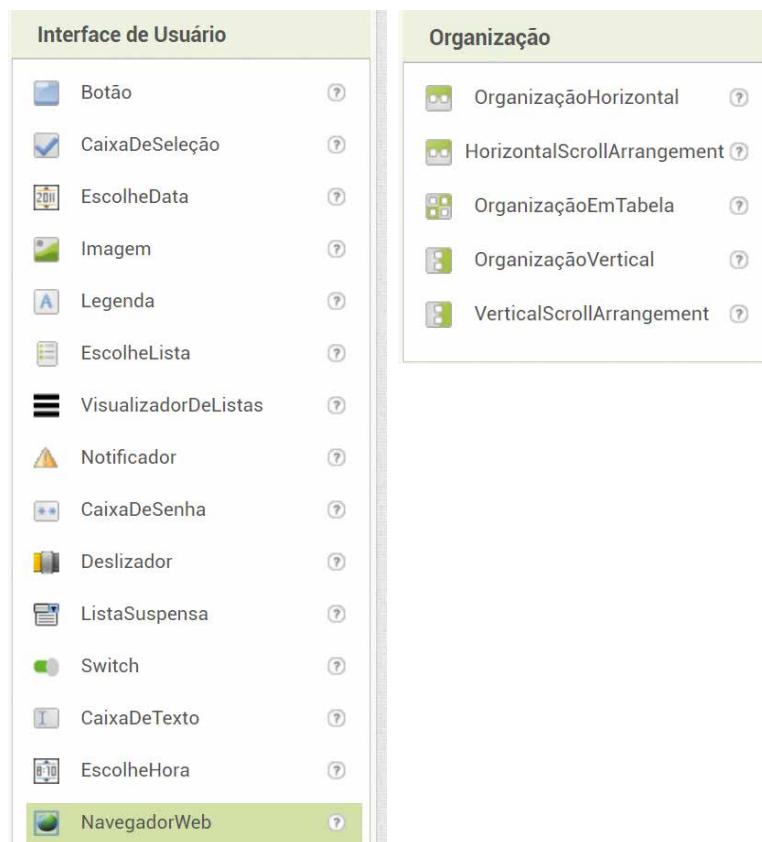
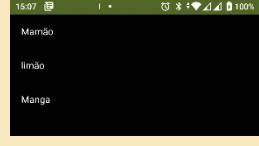
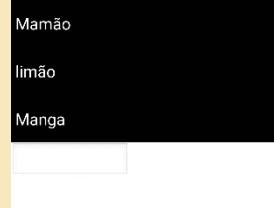


Figura 46 – Guias Interface de Usuário e Organização

# PROGRAMAÇÃO PARA DISPOSITIVOS MÓVEIS

No quadro a seguir, temos as funções e as imagens dos componentes da Interface de Usuário. Alguns componentes ativam outras telas. Temos um componente não visível, o Notificador, que é mostrado apenas por meio da ação dos blocos.

**Quadro 2 – Funções dos componentes do grupo Interface de Usuário**

Componente	Função	Imagen	Interação
Botão	Botão com capacidade de detecção de cliques		
CaixaDeSeleção	Detecta toque do usuário e altera seu estado booleano (selecionado ou não selecionado) em resposta		
EscolheData	Botão que, quando clicado, abre uma caixa de diálogo para permitir ao usuário selecionar uma data no calendário		
Imagen	Componente para exibição de imagens e animações básicas		
Legenda	Componente usado para mostrar texto	Texto para Legenda1	
EscolheLista	Botão que, ao ser clicado, mostra uma lista de textos para o usuário escolher em uma tela inteira		
VisualizadorDeListas	Componente visível que permite colocar uma lista de elementos de texto do dispositivo para exibição		
Notificador	Abre uma janela de mensagens de alerta e cria entradas de log do Android	Não visível	
CaixaDeSenha	Caixa de texto que oculta o texto que foi digitado		
Deslizador	Mostra um controle deslizante		

# Unidade I

Componente	Função	Imagem	Interação
ListaSuspensa	Exibe uma caixa de diálogo com uma lista de elementos para escolha		
Switch	O componente tem um estado ligado (verdadeiro) e um estado desligado (falso); também gera um evento quando o usuário o toca para alternar entre os estados		
CaixaDeTexto	Os usuários inserem texto por este componente		
EscolheHora	Botão que, ao ser clicado, abre uma caixa de diálogo para permitir que o usuário selecione um horário		
NavegadorWeb	Componente para visualização de páginas da web		

O grupo Organização da Paleta é importante na organização da tela. Os componentes desse grupo servem para posicionar, na tela, componentes como imagens ou botões colocados ou até mesmo outros organizadores dentro dele. Esses objetos servem para alinhar nossos componentes na tela, conforme mostra o quadro a seguir.

**Quadro 3 – Funções do grupo Organização da Paleta**

Organizador	Função
OrganizaçãoHorizontal	Exibe um grupo de componentes dispostos da esquerda para a direita
HorizontalScrollArrangement	Faz a OrganizaçãoHorizontal com barra deslizante
OrganizaçãoEmTabela	Exibe um grupo de componentes de forma tabular
OrganizaçãoVertical	Exibe um grupo de componentes dispostos de cima para baixo
VerticalScrollArrangement	Faz a OrganizaçãoVertical com barra deslizante

Os arranjos ou organizadores exigem um pouco de atenção para o bom entendimento do seu uso.

Na OrganizaçãoHorizontal, os componentes são dispostos ao longo do eixo horizontal, alinhados verticalmente no centro.

# PROGRAMAÇÃO PARA DISPOSITIVOS MÓVEIS

Se a propriedade Altura da OrganizaçãoHorizontal for definida como **Automático**, a altura real do arranjo será determinada pelo componente mais alto do arranjo, cuja propriedade Altura não está definida como **Preencher principal**.

Por exemplo, na figura a seguir, temos uma OrganizaçãoHorizontal com um botão, uma caixa de seleção e um switch colocados no seu interior. Na janela Componentes, temos a **Screen1**, que é a tela do nosso dispositivo. Dentro dela, temos uma OrganizaçãoHorizontal e, dentro dessa OrganizaçãoHorizontal, temos um botão, uma caixa de seleção e um switch.



Figura 47 – OrganizaçãoHorizontal com um botão, uma caixa de seleção e um switch

A altura da OrganizaçãoHorizontal está ajustada ao botão, que é o elemento mais alto, e os três estão alinhados ao topo. Veja a figura.

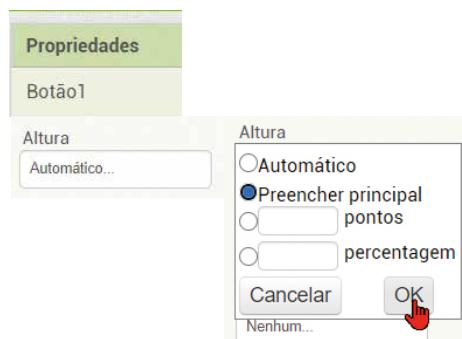


Figura 48 – Alteração da propriedade Altura do botão

Ao trocar a propriedade Altura do botão, temos um resultado em princípio inesperado. Veja a figura.



Figura 49 – Resultado ao alterar a propriedade Altura do botão

A explicação do resultado apresentado na figura anterior é a esperada. A propriedade Preencher principal ajusta a dimensão do organizador em que o componente Botão está incluído. No caso, a altura não é mais fixa. Assim, o componente mais alto passa a ser a caixa de seleção.

Um exemplo extremo é alterar as propriedades Altura e Largura do componente OrganizaçãoHorizontal para Preencher principal. Veja a figura.

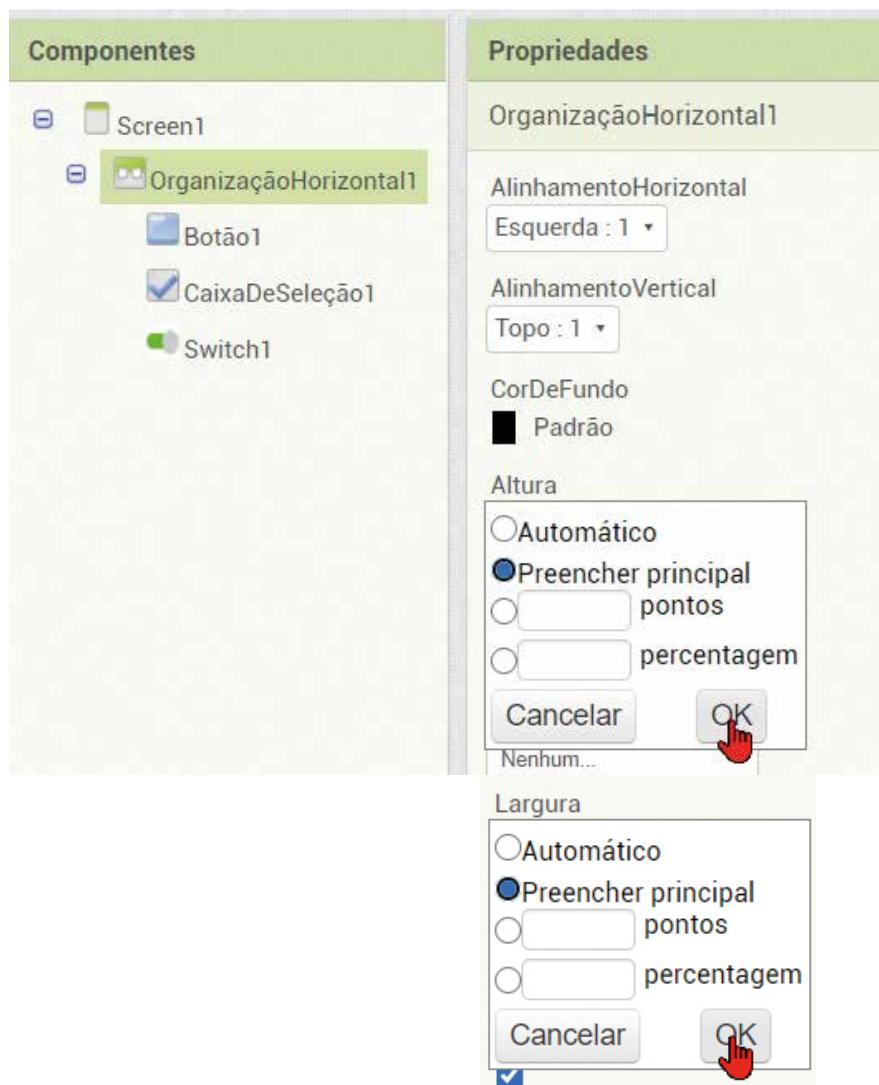


Figura 50 – Alteração das propriedades Altura e Largura da OrganizaçãoHorizontal

O resultado pode ser visto na figura a seguir.

# PROGRAMAÇÃO PARA DISPOSITIVOS MÓVEIS

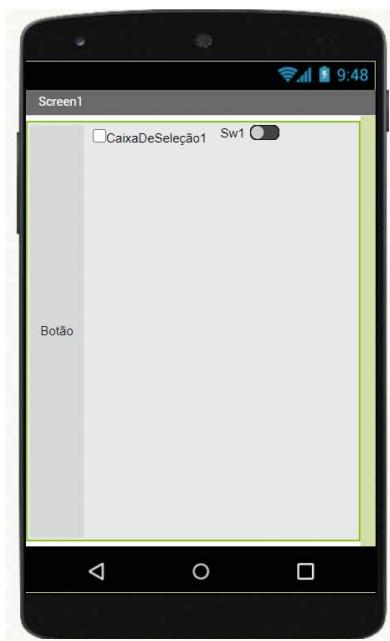


Figura 51 – OrganizaçãoHorizontal com a altura e a largura preenchendo o principal

Com as propriedades da OrganizaçãoHorizontal alteradas, observamos que as bordas da organização são as bordas da própria tela. Como a propriedade Altura do botão não foi alterada (Preencher principal), temos um botão alto, pois o principal do botão é a altura da OrganizaçãoHorizontal.

O exemplo da OrganizaçãoHorizontal também pode ser aplicado à OrganizaçãoVertical. Veja a figura.

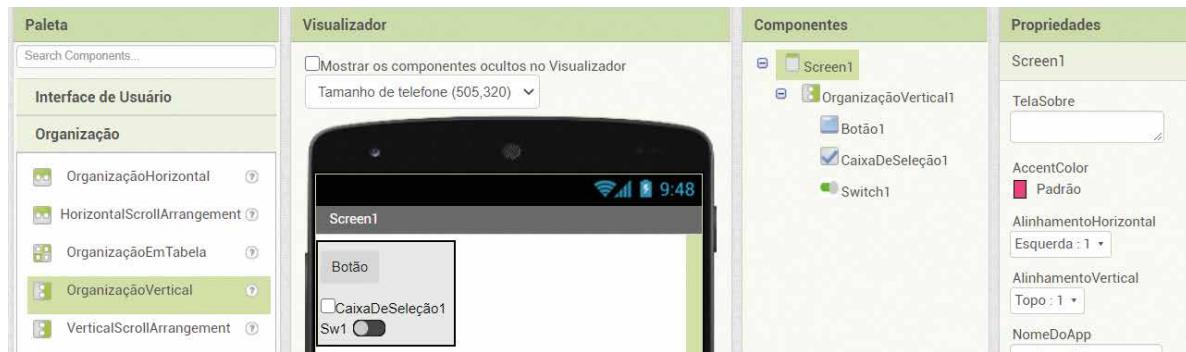


Figura 52 – OrganizaçãoVertical com um botão, uma caixa de seleção e um switch

## Paleta: outros componentes

Outro grupo da Paleta é a Mídia, em que temos os componentes de áudio e vídeo.

No grupo Desenho e Animação estão os componentes que permitem a interação dinâmica do usuário com a tela para fazer jogos.

O grupo Maps corresponde aos componentes que envolvem mapas e marcadores.

Os componentes de geolocalização estão na guia Sensores, juntamente com as outras interfaces do dispositivo com o mundo externo.

O grupo Social apresenta os componentes de relacionamento interpessoal.

Há um grupo que trata da persistência ou do armazenamento de dados ao sair do aplicativo.

A ligação entre dispositivos via bluetooth está no grupo Conectividade.

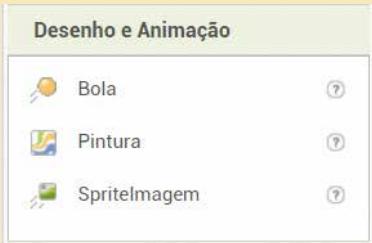
O grupo Experimental, por enquanto, trata do acesso ao banco de dados Firebase.

O que dissemos pode ser visto nos quadros a seguir.

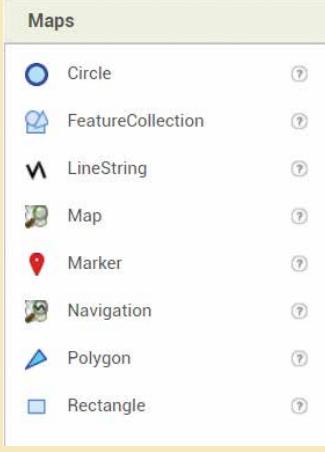
## Quadro 4 – Mídia: grupo de componentes

Grupo	Componente	Finalidade
Mídia	CâmeraDeVídeo	Componente para gravar um vídeo usando a câmera de vídeo do dispositivo
	Câmera	Componente da câmera para tirar uma foto no dispositivo
	Escolhemagem	Componente com a finalidade de exibir a galeria de imagens do dispositivo, de modo que o usuário possa escolher uma imagem
	Tocador	Componente multimídia que reproduz áudio e controla o toque do telefone
	Som	Componente que reproduz arquivos de som e, opcionalmente, vibra pelo número de milissegundos (milésimos de segundo) especificado no editor de blocos
	Gravador	Componente multimídia que grava áudio
	ReconhecedorDeVoz	Componente que ouve o usuário falando e converte o áudio falado em texto usando o recurso de reconhecimento de voz do dispositivo
	TextoParaFalar	Reproduz determinado texto em voz alta
	ReprodutorDeVideo	Componente para reproduzir vídeos
	TradutorYandex	Componente para traduzir palavras e frases entre diferentes idiomas; este componente necessita de acesso à internet, pois solicitará traduções ao serviço Yandex.Translate

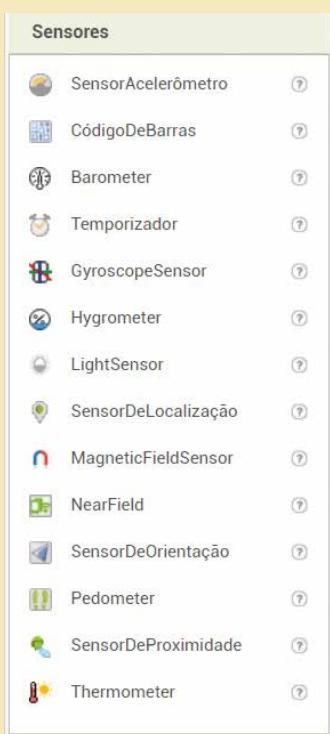
**Quadro 5 – Desenho e Animação: grupo de componentes**

Grupo	Componente	Finalidade
	Bola	Um sprite redondo que pode ser colocado na tela de pintura, em que pode reagir a toques e arrastar, interagir com outros sprites (outras bolas) e as bordas, além de se mover de acordo com os valores definidos na propriedade ou tempo de execução
	Pintura	Um painel retangular sensível ao toque bidimensional no qual o desenho pode ser feito e os sprites podem ser movidos
	Spritelgem	Uma imagem que pode ser colocada na tela de pintura, em que pode reagir a toques e arrastar, interagir com outros sprites (outras bolas) e as bordas, além de se mover de acordo com os valores definidos na propriedade ou tempo de execução

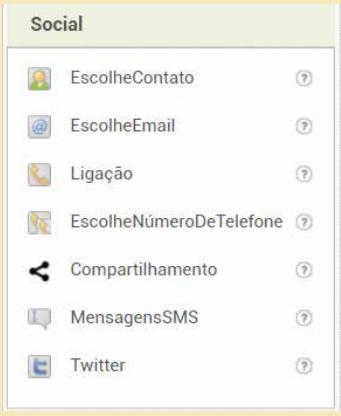
**Quadro 6 – Maps: grupo de componentes**

Grupo	Componente	Finalidade
	Circle	Visualiza um círculo de dado raio, em metros, centrado na latitude e na longitude
	FeatureCollection	Agrupa um ou mais recursos do mapa; quaisquer eventos que ocorram em um recurso na coleção também irão disparar o evento correspondente no componente da coleção; FeatureCollections podem ser carregados de recursos externos para preencher mapas com conteúdo; GeoJSON é o único formato compatível no momento
	LineString	Serve para desenhar uma sequência aberta e contínua de linhas no mapa
	Map	Insere um mapa na tela do dispositivo
	Marker	Indica pontos no mapa que podem ser personalizados de várias maneiras, como usando imagens personalizadas dos ativos do aplicativo
	Navigation	Gera direções entre dois locais usando um serviço chamado OpenRouteService; é necessário fornecer uma chave de API, a ser obtida em <a href="https://openrouteservice.org">https://openrouteservice.org</a>
	Polygon	Inclui uma área bidimensional arbitrária no mapa; serve para desenhar um perímetro
	Rectangle	É um polígono com latitudes e longitudes fixas para os limites norte, sul, leste e oeste

## Quadro 7 – Sensores: grupo de componentes

Grupo	Componente	Finalidade
 <b>Sensores</b> <ul style="list-style-type: none"> <li> SensorAcelerometro</li> <li> CódigoDeBarras</li> <li> Barometer</li> <li> Temporizador</li> <li> GyroscopeSensor</li> <li> Hygrometer</li> <li> LightSensor</li> <li> SensorDeLocalização</li> <li> MagneticFieldSensor</li> <li> NearField</li> <li> SensorDeOrientação</li> <li> Pedometer</li> <li> SensorDeProximidade</li> <li> Thermometer</li> </ul>	SensorAcelerômetro	Pode detectar tremores e medir a aceleração aproximadamente em três dimensões
	CódigoDeBarras	Reconhece um QR code e recupera o texto codificado
	Barometer	Mede a pressão do ar ambiente se houver o hardware necessário
	Temporizador	Fornece o instante no tempo usando o relógio interno do dispositivo; pode disparar um cronômetro em intervalos regulares e realizar cálculos de tempo, manipulações e conversões
	GyroscopeSensor	Fornece dados do sensor giroscópio do dispositivo
	Hygrometer	Mede a umidade relativa do ar ambiente, se compatível com o hardware
	LightSensor	Mede o nível de luz
	SensorDeLocalização	Informa a localização, incluindo latitude, longitude, altitude (se disponibilizada pelo hardware), velocidade (se disponibilizada pelo hardware) e endereço; também pode realizar a geocodificação, convertendo determinado endereço em um par de coordenadas (latitude e longitude)
	MagneticFieldSensor	Mede a força e o vetor do campo magnético se o dispositivo permitir
	NearField	Componente não visível para fornecer recursos NFC (tecnologia de troca de dados sem fio por aproximação entre dois dispositivos); por enquanto, este componente oferece apenas suporte à leitura e gravação de tags de texto (se compatível com o dispositivo)
	SensorDeOrientação	Usa um componente de sensor de orientação para determinar a orientação espacial do dispositivo
	Pedometer	Contagem dos passos usando o acelerômetro
	SensorDeProximidade	Pode medir a proximidade de um objeto (em cm) em relação à tela de visualização de um dispositivo; é normalmente usado para determinar se um auricular do celular está sendo colocado perto do ouvido de uma pessoa, ou seja, permite determinar a que distância um objeto está de um dispositivo
	Thermometer	Mede a temperatura do ar ambiente, se o hardware permitir

**Quadro 8 – Social: grupo de componentes**

Grupo	Componente	Finalidade
 <b>Social</b> <ul style="list-style-type: none"> <li> EscolheContato <a href="#">?</a></li> <li> EscolheEmail <a href="#">?</a></li> <li> Ligação <a href="#">?</a></li> <li> EscolheNúmeroDeTelefone <a href="#">?</a></li> <li> Compartilhamento <a href="#">?</a></li> <li> MensagensSMS <a href="#">?</a></li> <li> Twitter <a href="#">?</a></li> </ul>	EscolheContato	Botão que, ao ser clicado, exibe uma lista para escolher o contato
	EscolheEmail	Caixa de texto que, ao ser clicada, exibe uma lista para escolher o email de contato
	Ligação	Componente não visível que faz chamada telefônica controlado por blocos
	EscolheNúmeroDeTelefone	Botão que, ao ser clicado, exibe uma lista para escolher o número de telefone dos contatos
	Compartilhamento	Componente não visível que permite o compartilhamento de arquivos e/ou mensagens entre o seu aplicativo e outros aplicativos instalados em um dispositivo; o componente exibirá uma lista dos aplicativos instalados que podem lidar com as informações fornecidas e permitirá ao usuário escolher um para compartilhar o conteúdo – por exemplo, um aplicativo de email, um aplicativo de rede social, um aplicativo de texto, SMS, Facebook etc.
	MensagensSMS	Ativa o envio de mensagem SMS
	Twitter	Componente não visível que permite a comunicação com o Twitter

**Quadro 9 – Armazenamento: grupo de componentes**

Grupo	Componente	Finalidade
 <b>Armazenamento</b> <ul style="list-style-type: none"> <li> CloudDB <a href="#">?</a></li> <li> Arquivo <a href="#">?</a></li> <li> TinyDB <a href="#">?</a></li> <li> TinyWebDB <a href="#">?</a></li> </ul>	CloudDB	Componente não visível que permite armazenar dados em um servidor de banco de dados conectado à internet (usando o software Redis)
	Arquivo	Componente não visível para armazenamento e recuperação de arquivos; use este componente para gravar ou ler arquivos no dispositivo
	TinyDB	Componente não visível que armazena dados para um aplicativo
	TinyWebDB	Componente que se comunica com um serviço da web para armazenar e recuperar informações; ele é muito limitado e tem como objetivo principal uma demonstração para pessoas que desejam criar seus próprios componentes que se comunicam com a web

## Quadro 10 – Conectividade: grupo de componentes

Grupo	Componente	Finalidade
Conectividade	IniciadorDeAtividades	Componente que inicia outro aplicativo App Inventor for Android
	ClienteBluetooth	Componente que conecta seu dispositivo a outros dispositivos usando bluetooth
	ServidorBluetooth	Componente que transforma seu dispositivo em um servidor que recebe conexões de outros aplicativos que usam o ClienteBluetooth
	Serial	Componente para porta serial
	Web	Componente não visível que fornece funções para solicitações HTTP GET, POST, PUT e DELETE

## Quadro 11 – Experimental: grupo de componentes

Grupo	Componente	Finalidade
Experimental	FirebaseDB	Componente que se comunica com um serviço da Web para armazenar e recuperar informações

### 2.1.2 Visualizador

O Visualizador é uma previsão da aparência da tela do nosso aplicativo no dispositivo após a instalação. Conforme os componentes são arrastados para o Visualizador, eles são automaticamente adicionados ao projeto. A característica do componente inserido será introduzida na tela, caso seja visível, como o botão da figura a seguir, ou sob a tela, caso seja não visível, como o reconhecedor de voz.

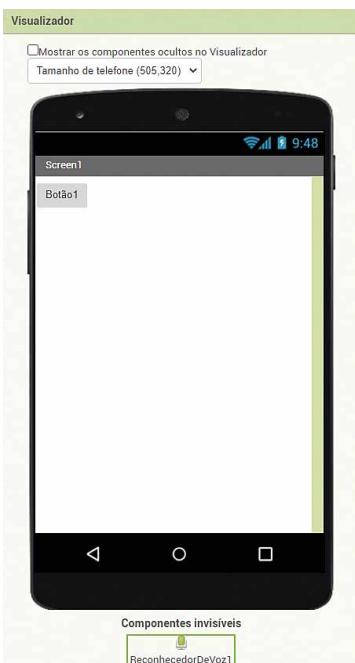


Figura 53 – Tela do Visualizador com componentes visível e não visível

## 2.1.3 Componentes

A janela Componentes lista todos os componentes adicionados ao Visualizador. Todos os componentes são listados abaixo da tela na estrutura de árvore. Por meio do aninhamento de organizadores, é possível montar estruturas complexas de layout na tela do dispositivo. Veja a figura.

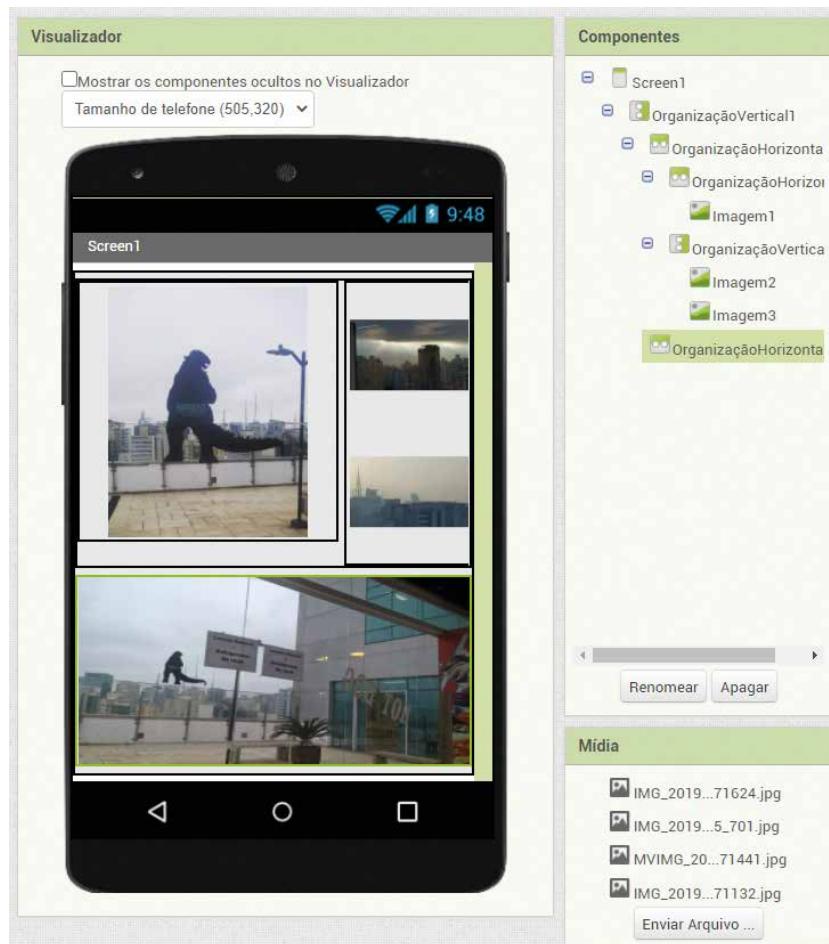


Figura 54 – Layout montado e estrutura montada na janela Componentes

A janela Componentes também permite o acesso rápido aos objetos. Ao dar um clique, o objeto ficará selecionado também nas janelas do Visualizador e das Propriedades.

Abaixo da árvore, existem dois botões: Renomear e Apagar.

Renomear objetos com um nome descritivo é uma prática recomendável, pois:

- torna mais fácil entender o código do nosso aplicativo;
- auxilia a compreender sua função no aplicativo quando se renomeiam os componentes de acordo com seu tipo (botão) e sua função (atualizar);

- ajuda outros programadores a ler e a entender nosso código, principalmente em projetos colaborativos;
- facilita a identificação dos objetos dentro do conjunto do aplicativo, seja na árvore de componentes, seja na tela de blocos.

Para realizar o procedimento, basta escolher o componente e clicar no botão **Renomear**, que abrirá o notificador da figura.

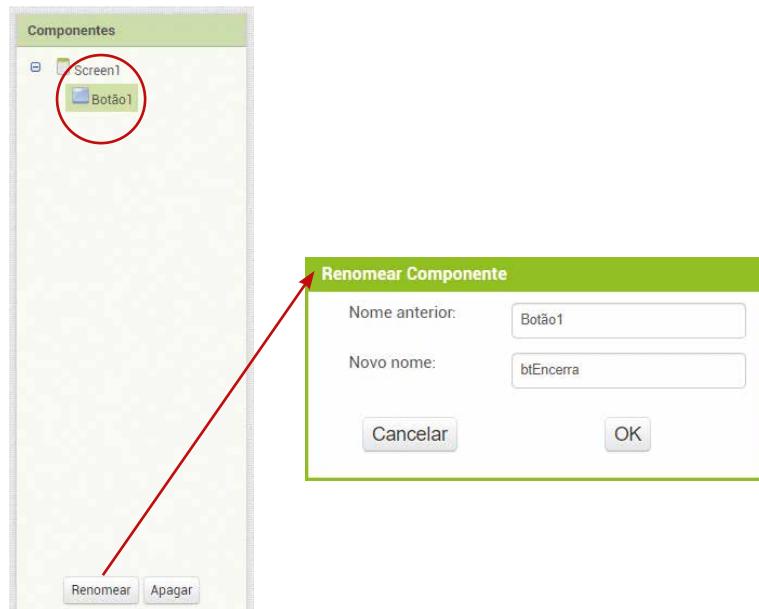


Figura 55 – Notificador de Renomear Componente

## 2.1.4 Mídia

A Mídia é uma janela que fica abaixo da janela Componentes.

As mídias não são componentes propriamente ditos, mas anexos externos que são associados como propriedades dos componentes. Esses anexos são arquivos de áudio, foto e vídeo. Veja a figura.

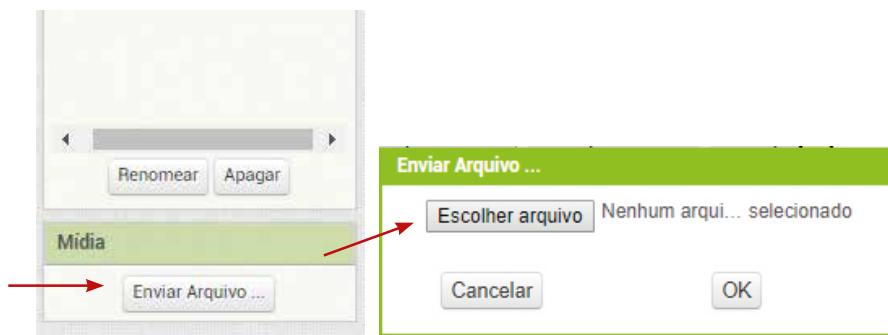


Figura 56 – Sequência para a inserção de mídia



## Saiba mais

Quais são as mídias suportadas pelo App Inventor? A compatibilidade das mídias é definida pelo sistema operacional. São muitos os tipos e os formatos de mídia. A relação completa para Android está disponível na página indicada a seguir.

FORMATOS de mídia compatíveis. *Developers*, 27 dez. 2019. Disponível em: <http://bit.ly/3qjcOBW>. Acesso em: 19 mar. 2021.

### 2.1.5 Propriedades

A janela Propriedades permite definir diferentes propriedades para os componentes, como altura, largura, cor, texto e fundo. Quando qualquer componente é adicionado ao projeto ou é selecionado na janela Componentes, suas respectivas propriedades são mostradas na janela Propriedades. Essas propriedades podem ser configuradas de acordo com o requisito.

#### Propriedades: como configurar

Os objetos e os componentes são definidos por um conjunto de propriedades. Cada um dos componentes tem o seu conjunto de propriedades, e cada objeto colocado no projeto tem os valores das propriedades independentes dos outros objetos do projeto. Assim, dois botões têm o mesmo conjunto de propriedades, mas o valor contido nas propriedades é exclusivo de cada um deles.

As propriedades são slots (espaços) de memória para armazenar informações sobre os objetos.

Objetos visíveis, por exemplo, têm grandezas, como altura e largura, ou características, como cor e alinhamento, que em conjunto definem como eles aparecem na tela. Veja a figura.



Figura 57 – Tela de Propriedades de um objeto visível

Cada uma das propriedades pode ser alterada no editor de blocos durante a execução do aplicativo. Portanto, as propriedades são apenas as condições iniciais do aplicativo.

## 2.2 Criação do app para a demonstração de componentes

Vamos mostrar detalhadamente a construção de uma tela semelhante a uma tela que já fizemos.

Um novo projeto deve ser criado no App Inventor. Na tela **Meus Projetos**, clique no botão **Iniciar novo projeto** e batize o projeto como **Exercicio01** (ou use um nome alternativo). Veja a figura.

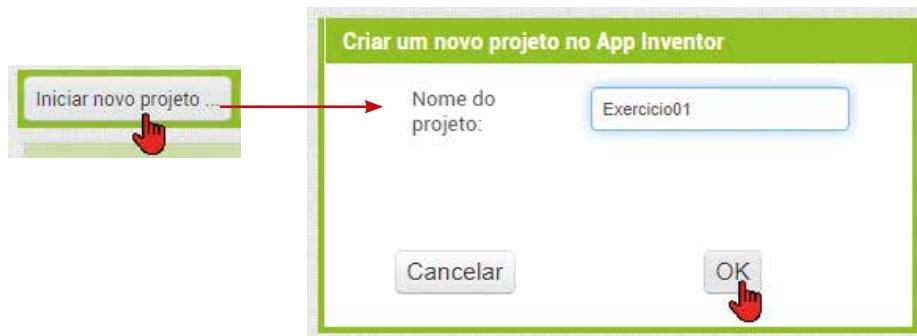


Figura 58 – Iniciando um novo projeto

Na tela do Designer, do grupo Organização da Paleta, arraste o componente **OrganizaçãoVertical** para **Screen1**. Veja a figura.

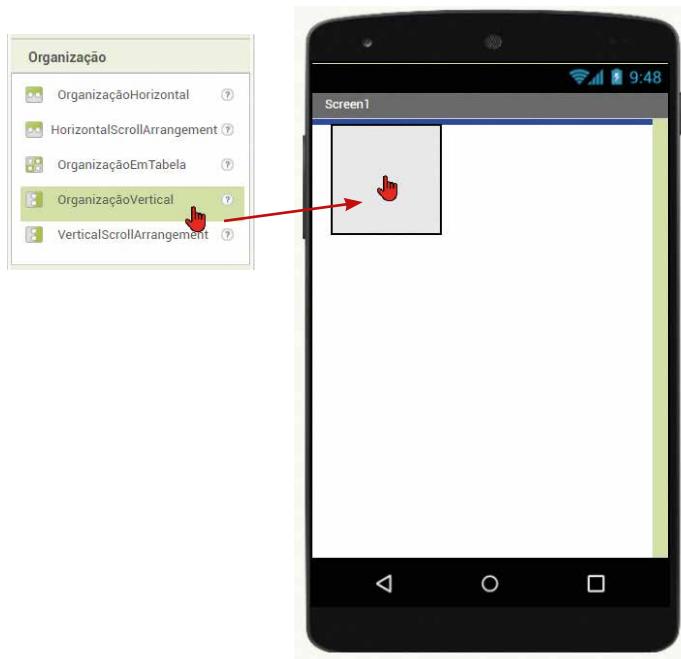


Figura 59 – Inserindo uma OrganizaçãoVertical na tela

# PROGRAMAÇÃO PARA DISPOSITIVOS MÓVEIS

Na janela Propriedades, altere a largura da organização para 100%. Os dispositivos móveis não têm tamanho de tela padronizada; vão desde pequenos celulares até grandes tablets. Assim, a formatação deve ser flexível (dada por um percentual do tamanho da tela). Colocar um botão ou um componente fixo em um ponto da tela de um dispositivo não significa que isso ficará esteticamente melhor. Dessa forma, o importante é trabalhar com proporções. Veja a figura.

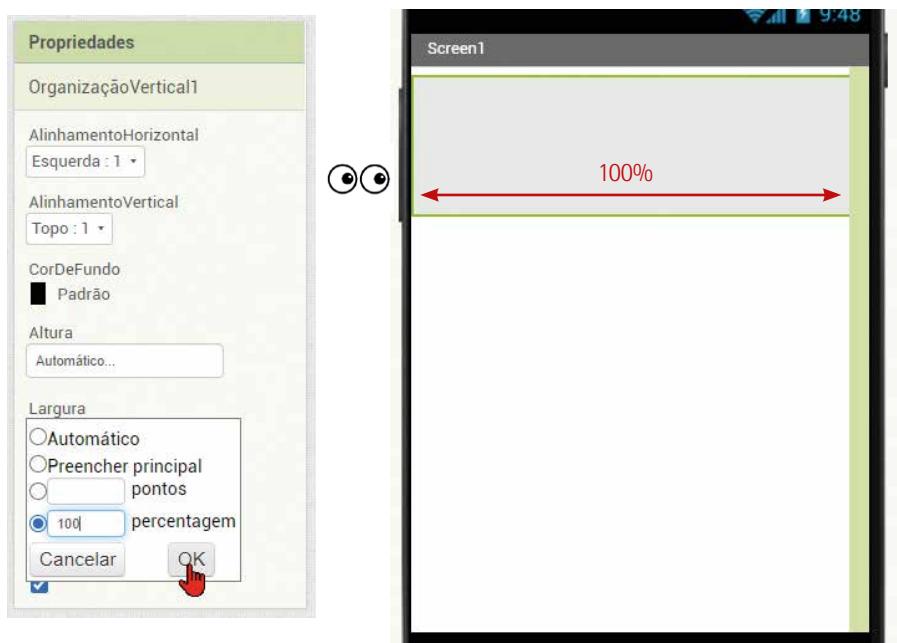


Figura 60 – Mudança da propriedade Largura da organização para 100%

Para entender o funcionamento das dimensões relativas, temos nas Propriedades o seguinte: Automático, Preencher principal, pontos e percentagem. Veja o quadro.

## Quadro 12 – Ajustes das propriedades dimensionais

Referência	Finalidade
Automático	Ajusta ao conteúdo
Preencher principal	Ajusta ao espaço livre em relação ao componente imediatamente pai
pontos	Tamanho fixo em pixels
percentagem	Percentual em relação ao tamanho da tela do dispositivo

A altura também precisa ser alterada para 100%. Assim, a OrganizaçãoVertical ocupará todo o fundo da tela do dispositivo, conforme se vê na figura.

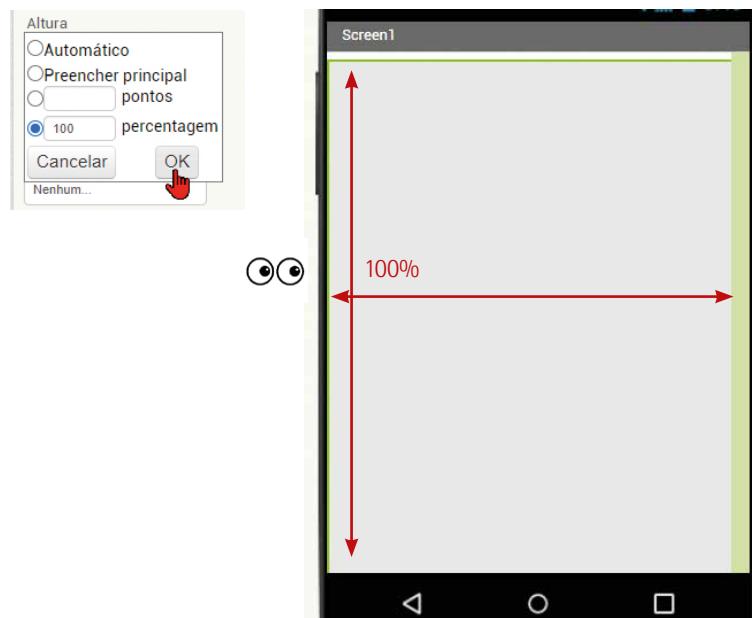


Figura 61 – OrganizaçãoVertical ocupando 100% da altura e 100% da largura

A partir do grupo Organização da Paleta, arraste duas OrganizaçõesHorizontais para dentro da OrganizaçãoVertical. Observe duas coisas. As OrganizaçõesHorizontais ficaram dispostas verticalmente no Visualizador. Na janela Componentes, as duas OrganizaçõesHorizontais aparecem em um nível hierárquico abaixo da OrganizaçãoVertical. Veja a figura.

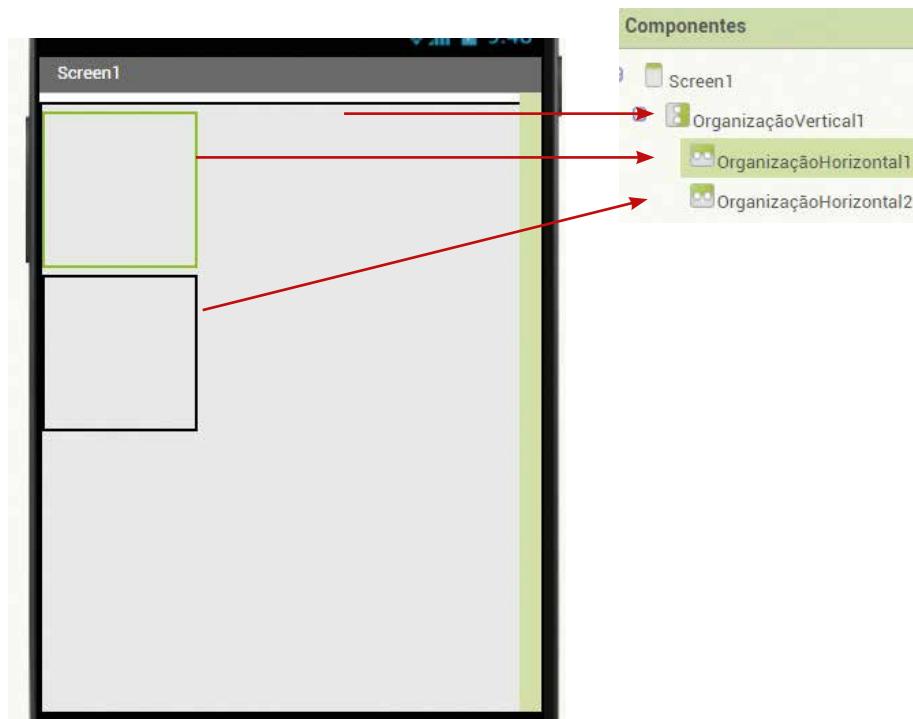


Figura 62 – Inclusão de duas OrganizaçõesHorizontais

# PROGRAMAÇÃO PARA DISPOSITIVOS MÓVEIS

Nos dois organizadores horizontais, faça a alteração das dimensões. Veja o quadro.

**Quadro 13 – Propriedades das OrganizaçõesHorizontais**

Objeto	Propriedade	Valor
OrganizaçãoHorizontal1	Largura	Preencher principal
OrganizaçãoHorizontal1	Altura	50%
OrganizaçãoHorizontal2	Largura	Preencher principal
OrganizaçãoHorizontal2	Altura	Preencher principal

Para alterar as propriedades dos componentes, basta clicar no objeto na janela Componentes, conforme mostra a figura a seguir.

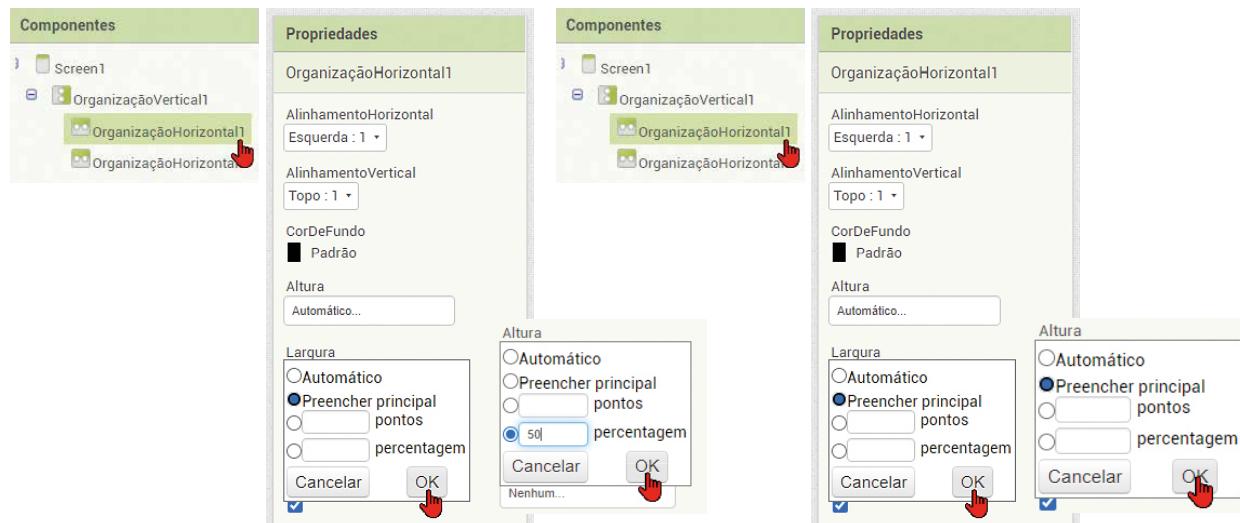


Figura 63 – Alteração das propriedades das OrganizaçõesHorizontais

Na OrganizaçãoHorizontal1, arraste uma imagem da paleta Interface de Usuário. Observe que a imagem está abaixo da OrganizaçãoVertical1 e da OrganizaçãoHorizontal1. Veja a figura.

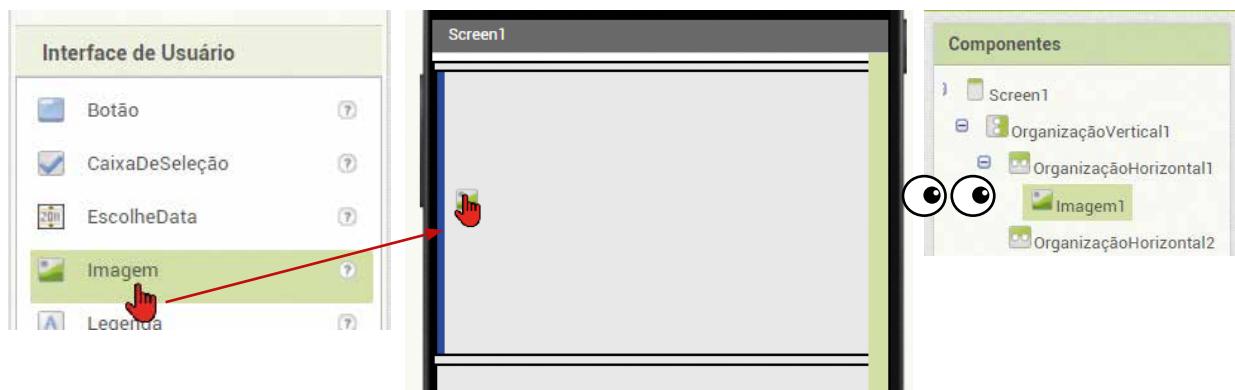


Figura 64 – Inserindo uma imagem na OrganizaçãoHorizontal

Na metade superior, queremos ter uma foto do lado esquerdo e duas fotos dispostas verticalmente do lado direito. Já temos a imagem do lado esquerdo. Para colocarmos as duas fotos na vertical, é necessário colocar um organizador vertical do lado direito. Assim, arraste uma OrganizaçãoVertical para o lado direito da Imagem1. Altere as propriedades da dimensão. Veja as figuras e o quadro a seguir.

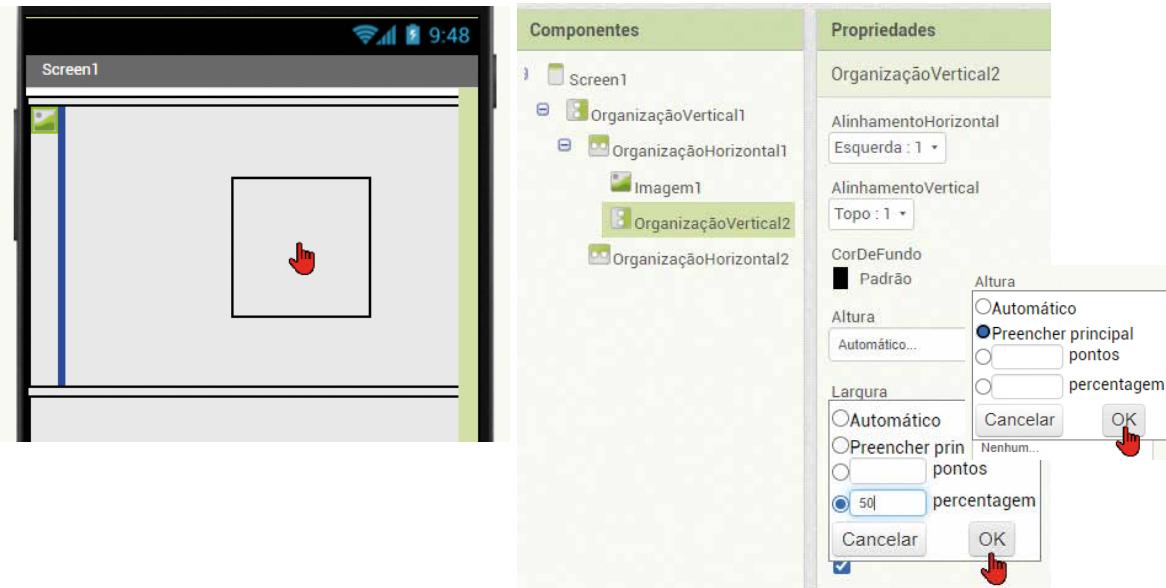


Figura 65 – Inserindo uma OrganizaçãoVertical depois da Imagem1

## Quadro 14 – Propriedades alteradas da OrganizaçãoVertical2

Objeto	Propriedade	Valor
OrganizaçãoVertical2	Largura	50%
OrganizaçãoVertical2	Altura	Preencher principal

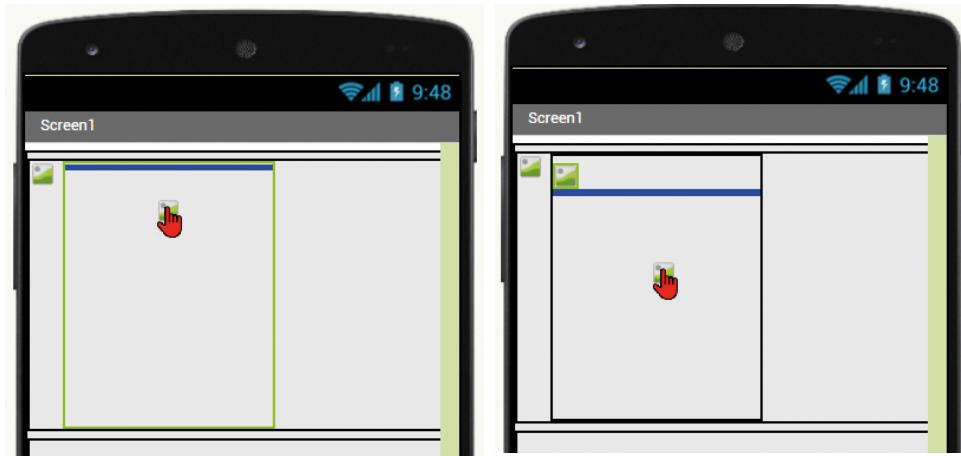


Figura 66 – Inserção de duas imagens na OrganizaçãoVertical2

Insira uma legenda na OrganizaçãoHorizontal2, conforme as indicações do quadro a seguir.

**Quadro 15 – Alteração de propriedades dos objetos**

Objeto	Propriedade	Valor
Imagen1	Altura	Preencher principal
Imagen1	Largura	Preencher principal
Imagen1	Imagen	pdm0001.jpg
Imagen2	Altura	Preencher principal
Imagen2	Largura	Preencher principal
Imagen2	Imagen	pdm0002.jpg
Imagen3	Altura	Preencher principal
Imagen3	Largura	Preencher principal
Imagen3	Imagen	pdm0003.jpg
OrganizaçãoHorizontal2	Imagen	pdm0004.jpg
OrganizaçãoHorizontal2	AlinhamentoHorizontal	Centro
Legenda1	FonteNegrito	Marcar
Legenda1	FonteLálico	Marcar
Legenda1	TamanhoDaFonte	60
Legenda1	Texto	PDM
Legenda1	CorDeTexto	Amarelo
Screen1	Título	Projeto Exercício

Feitas as configurações apontadas, o projeto estará pronto. Na figura a seguir, vê-se a situação do projeto. Observe a janela Componentes e a árvore da hierarquia das organizações e das imagens.

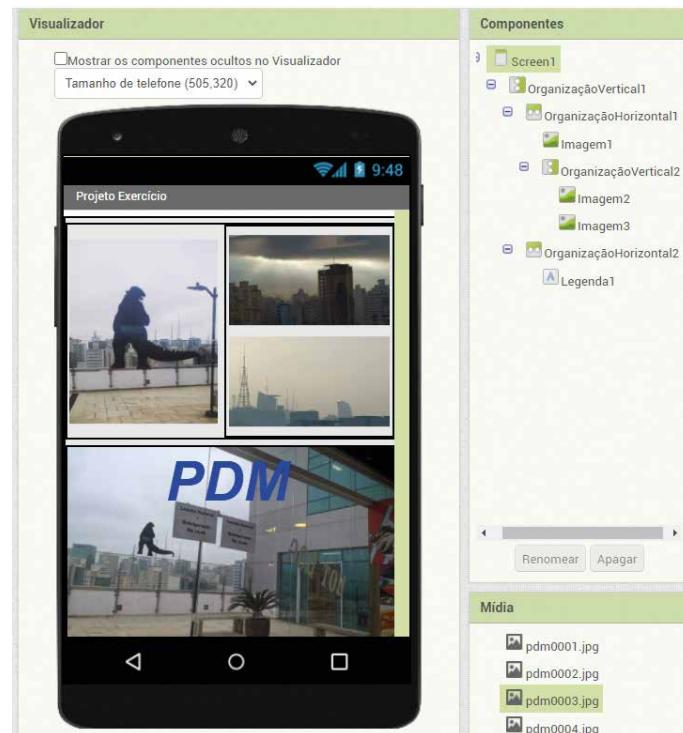


Figura 67 – Situação final do projeto

A figura a seguir mostra a visualização do projeto na tela de um dispositivo móvel.



Figura 68 – Screenshot do projeto pronto

## 3 BLOCOS

É no editor de blocos que são programados a lógica dos aplicativos e seu comportamento. Nesse editor, estão presentes todos os componentes que foram adicionados no Designer.

Assim como o Designer, o editor de blocos oferece uma interface de arrastar e soltar. Nesse caso, para programar o comportamento do aplicativo, juntam-se blocos, de maneira semelhante ao que fazemos em um quebra-cabeça.



### Saiba mais

Antes de começar este módulo, é recomendável jogar o labirinto do site indicado a seguir.

*Jogos do Blockly.* Disponível em: <http://bit.ly/3qQwexY>. Acesso em: 19 mar. 2021.

# PROGRAMAÇÃO PARA DISPOSITIVOS MÓVEIS

Em ambientes de programação tradicionais, você precisa digitar o código de programação para desenvolver o aplicativo em uma linguagem computacional, como Python, C ou Java. O App Inventor permite que os desenvolvedores iniciantes criem aplicativos de forma relativamente rápida, arrastando e soltando blocos de código na tela, em vez de digitando o programa.

Os blocos do App Inventor incorporam a mesma lógica de programação que seria usada na escrita do código em ambientes baseados em texto. Isso significa que os conceitos de programação aqui utilizados serão úteis futuramente na codificação nas linguagens tradicionais.

A programação por blocos segue uma sequência linear de comandos de cima para baixo, e cada bloco executa uma ação. Veja a figura.

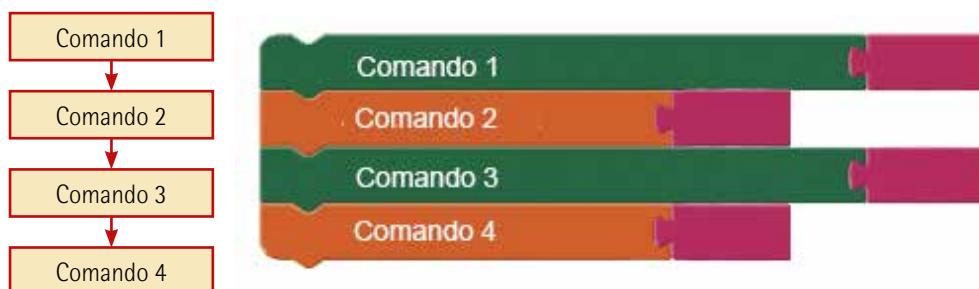


Figura 69 – Na programação tradicional, os comandos seguem uma sequência linear.  
O mesmo conceito corresponde à sequência dos blocos

Assim, os blocos são a linguagem de programação do App Inventor. Os encaixes horizontais são feitos para a passagem de valores. O resultado do bloco da direita é passado para o bloco da esquerda pelo encaixe. Veja a figura.

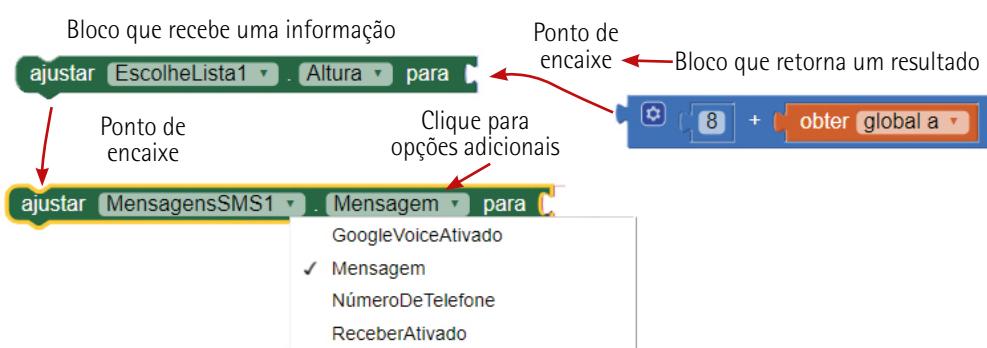


Figura 70 – Montagem dos blocos de programação

A maioria dos blocos tem formato estático, ou seja, sua forma não pode ser alterada. Mas existem alguns blocos que são mutantes, isto é, sua forma pode variar sem perder o seu conceito básico. Conforme mostrado na figura a seguir, os blocos mutantes têm um ícone azul.



Figura 71 – Alteração do formato básico do bloco

O comutador pode mudar também a quantidade de parâmetros de um bloco. Veja a figura.

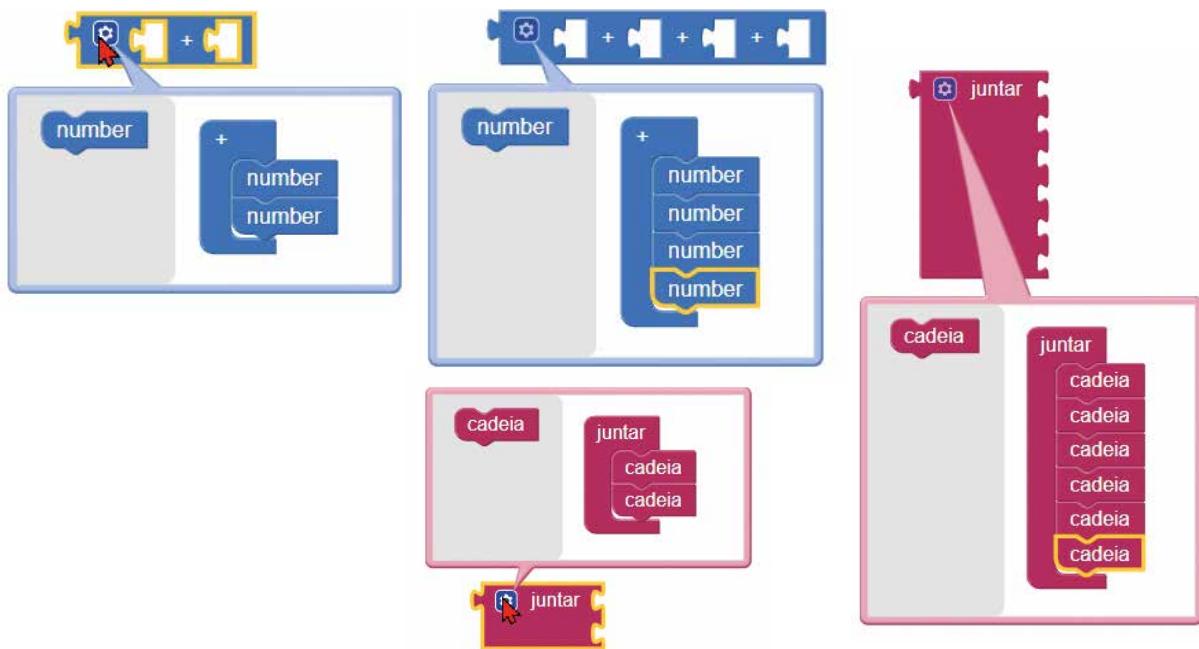


Figura 72 – O comutador pode ser utilizado para aumentar o número de parâmetros

O estilo de programação do App Inventor é, muitas vezes, chamado de programação dirigida por eventos (WOLBER *et al.*, 2011). Isso significa que a execução dos aplicativos e das funções é baseada em reações a eventos.

## Observação

Vejamos definições de termos pertinentes aos assuntos aqui tratados.

- **Evento:** algo que acontece em um aplicativo por meio da ação do usuário (por exemplo, clicar em um botão, tocar na tela ou mover o telefone) ou por meio de alguma outra ação do mundo real (por exemplo, receber uma mensagem de texto ou chamada telefônica).

# PROGRAMAÇÃO PARA DISPOSITIVOS MÓVEIS

- **Manipulador de eventos:** elemento de software que responde a um evento, como o bloco Botão.
- **Programação orientada a eventos ou dirigida por eventos:** refere-se à escrita de aplicativos em que o controle depende de como o programa responde a vários eventos.
- **Loop de eventos:** indica que os dispositivos móveis são projetados para escutar constantemente a ocorrência de eventos e, em seguida, responder a esses eventos.
- **Interface gráfica de usuário (GUI):** contém janelas, botões e outros componentes que o usuário pode usar para interagir com um aplicativo.
- **Sistemas operacionais:** softwares, como os sistemas operacionais Android ou iOS, que intermedeiam a interação entre os aplicativos móveis e o hardware subjacente.

Conforme os eventos ocorrem, o aplicativo reage, chamando uma sequência de funções. Vale notar que funções são ações que você pode executar com um ou mais componentes. Podem ser operações do tipo enviar um texto SMS ou operações do tipo alterar uma propriedade (como alterar o texto em um rótulo da interface do usuário). Um conjunto de funções é executado em resposta ao manipulador de eventos. Veja a figura.



Figura 73 – Laço (loop) do evento

Os eventos não são iniciados necessariamente pela ação do usuário final. Um aplicativo pode:

- reagir a eventos que acontecem dentro do dispositivo, como alterações em seu sensor de orientação e no relógio (ou seja, temporizadores);
- responder a eventos com origem externa ao dispositivo, como uma mensagem de texto, uma chamada recebida de outro telefone ou a chegada de dados da web.

Nesse sentido, observe a figura.

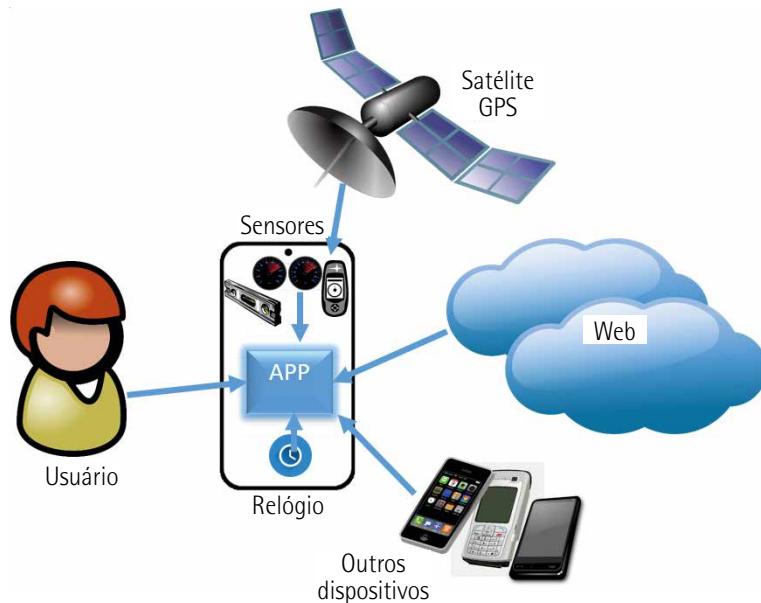


Figura 74 – O aplicativo pode responder tanto a eventos internos quanto a eventos externos

## 3.1 Conceito de programação: comportamento

O comportamento define como o aplicativo deve responder a eventos.

Conforme vimos, a programação segue, sequencialmente, de cima para baixo. Vamos usar esse conceito de programação como comportamento. No caso do App Inventor, todo comportamento sempre é iniciado por um evento. Veja a figura.

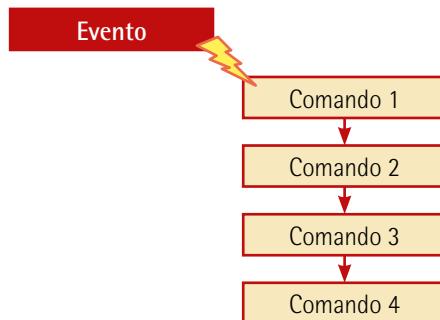


Figura 75 – Um comportamento é disparado por um evento

# PROGRAMAÇÃO PARA DISPOSITIVOS MÓVEIS

Um aplicativo, então, é uma coleção de comportamentos que são acionados por eventos. Veja a figura.

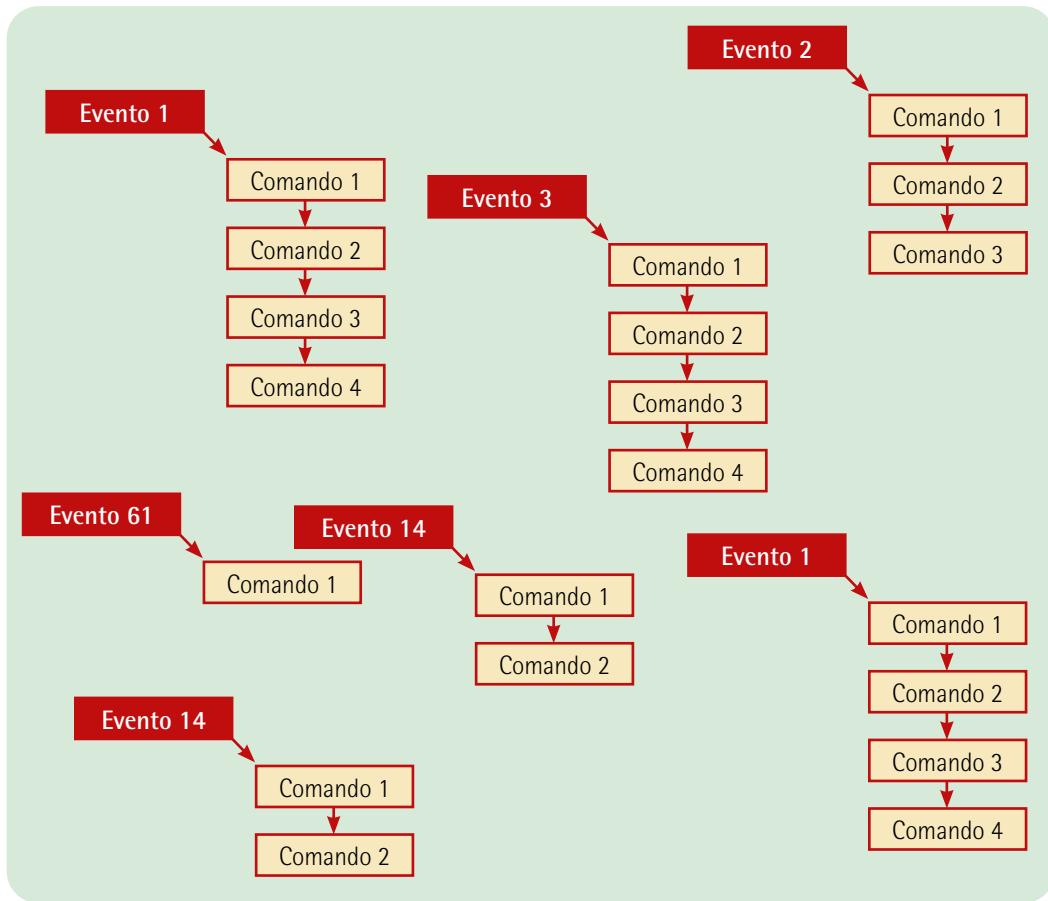


Figura 76 – Aplicativo é um conjunto de comportamentos

## 3.2 Tipos de evento

O App Inventor tem cinco tipos de evento. Como dito anteriormente, não é somente o usuário que inicia um evento. Antes de começar a colocar os blocos, é interessante saber mais sobre esses eventos. Uma maneira fácil de procedermos a esse entendimento é pensando em termos de exemplos, como mostrado no quadro a seguir.

Quadro 16 – Eventos e exemplos

Tipo de evento	Exemplo
Evento iniciado pelo usuário	Quando o usuário clicar no Botão1, faça...
Evento de inicialização	Quando o aplicativo for iniciado, faça...
Evento de timer	Após 20 milissegundos, faça...
Evento de animação	Quando dois objetos colidirem, faça...
Evento externo	Quando o telefone receber uma mensagem de texto, faça...

Formalmente, vamos ver cada um dos tipos de evento.

## Evento iniciado pelo usuário

Eventos iniciados pelo usuário são os tipos mais comuns. Veja a figura.



Figura 77 – Evento iniciado pelo usuário: clique no botão

O conjunto de blocos da figura, ao receber o evento acionado pelo usuário (clique no botão), executa sequencialmente os dois blocos internos.

## Evento de inicialização

Às vezes, o aplicativo precisa executar determinadas funções automaticamente quando o aplicativo começa. Para isso, utilizam-se os eventos de inicialização. Veja a figura.

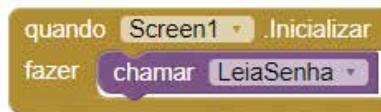


Figura 78 – Quando a Screen1 é aberta, o evento de inicialização é disparado

Situações como a leitura de senha e a verificação de dados do usuário cadastrados necessitam ser conferidas sempre que o aplicativo for acionado.

Na figura, temos o bloco que captura o evento disparado quando a Screen1 é aberta.

## Evento de timer

No evento de timer, alguma atividade em um aplicativo é acionada pela passagem do tempo. Veja a figura.

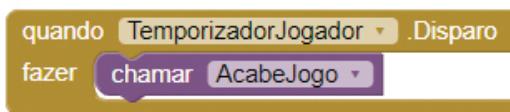


Figura 79 – Evento acionado por temporizador

# PROGRAMAÇÃO PARA DISPOSITIVOS MÓVEIS

O relógio interno do dispositivo, o temporizador, dispara gatilhos, pulsos e eventos de tempos em tempos, conforme determinado pela propriedade. No bloco da figura, ao receber o evento do relógio, chama-se a rotina de encerrar o jogo.

## Evento de animação

Os eventos de animação correspondem a atividades que envolvem objetos gráficos (sprites) dentro de telas para acioná-los. Veja a figura.

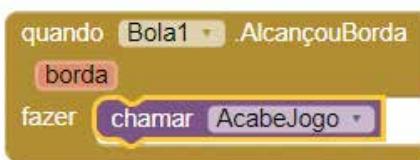


Figura 80 – Tratamento de evento gráfico

Na figura, temos um dos blocos de eventos gráficos. Choques, interação entre cores e posicionamentos dentro da tela de pintura geram eventos de animação.

## Evento externo

Os eventos externos são atividades iniciadas por uma mensagem, uma ligação, uma resposta da internet etc. Veja a figura.



Figura 81 – Evento disparado exteriormente

Na figura anterior, o bloco somente será executado se uma mensagem de texto for recebida.

## 3.3 Componentes dos blocos

Como vimos em item anterior, o acesso à tela do editor de blocos é feito ao darmos um clique no botão Blocos no canto superior direito da tela. A tela é composta por três janelas, indicadas a seguir.

- **Mídia.** Lista os arquivos de mídia carregados.
- **Blocos.** Disponibiliza os blocos internos e os blocos das ações e respostas dos componentes utilizados no Designer.

- **Visualizador.** É a área de trabalho em que os blocos são montados. Nele, destacam-se a lata de lixo, em que são descartados os blocos desnecessários, e a mochila, em que são armazenados blocos para o compartilhamento entre projetos.

Observe a figura.



Figura 82 – Janelas do editor de blocos

## Blocos internos

Os blocos internos são adicionados para alterar os comportamentos gerais do seu aplicativo. Para isso, arraste-os para o visualizador de blocos e responda ao seu evento correspondente. Veja a figura.

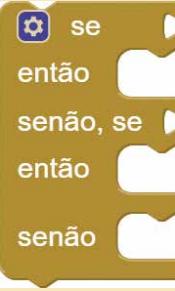
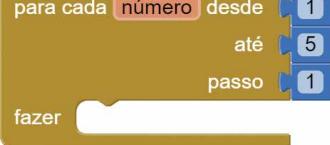
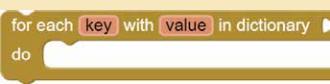


Figura 83 – Blocos internos

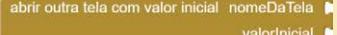
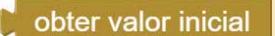
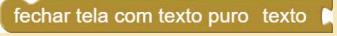
Os blocos internos estão divididos em grupos segundo suas características, como na Paleta do Designer.

No grupo Controle estão os blocos que alteram o fluxo do programa, conforme o quadro a seguir.

## Quadro 17 – Grupo Controle

Bloco	Comando	Comportamento
	se/então	Testa determinada condição. Se a condição for verdadeira, executa as ações em certa sequência de blocos; caso contrário, os blocos são ignorados
	se/então/senão	Testa determinada condição. Se a condição for verdadeira, executa as ações na sequência de blocos do então; caso contrário, executa as ações na outra sequência de blocos
	se/então/senão, se/então/senão	Testa determinada condição. Se o resultado for verdadeiro, executa as ações na sequência de blocos do primeiro então; caso contrário, testa a instrução na seção "senão, se". Se o resultado for verdadeiro, executa as ações na sequência de blocos do segundo então; caso contrário, executa as ações na sequência de blocos do senão
	para cada número desde	Executa repetidas vezes os blocos do fazer, contando o número para cada valor numérico no intervalo, começando desde o valor informado e terminando até o valor informado, realizando incrementos pelo passo definido. Use o nome de variável fornecido, número, para se referir ao valor atual. Você pode alterar o nome número para outro nome
	para cada item na lista	Executa repetidamente os blocos na seção fazer para cada item da lista. Use o nome de variável fornecido, item, para se referir ao item da lista encaixada. Você pode alterar o nome item para outro nome conforme a necessidade
	for each key with value in dictionary do	Executa os blocos na seção fazer para cada entrada de valor key (chave) no dicionário. Use as variáveis fornecidas, key e value, para se referir à chave e ao valor da entrada do dicionário atual. Você pode alterar os nomes key e value
	enquanto	Testa a condição lógica testar. Se verdadeiro, executa a ação fornecida em fazer e testa novamente. Quando o teste é falso, o bloco termina e a ação fornecida em fazer não é mais executada
	se/então/senão	Testa determinada condição lógica. Se for verdadeira, executa as ações da sequência de blocos de então e devolve o valor do resultado; caso contrário, executa as ações na sequência do senão e devolve o valor do resultado

# Unidade I

Bloco	Comando	Comportamento
	fazer/resultado	Às vezes, em um procedimento, o resultado de vários blocos de códigos precisa ser retornado por um conector. Nesse caso, utiliza-se este bloco. Ele pode ser alternativa à criação de um novo procedimento
	avaliar, mas ignorar resultado	Fornece um "soquete fictício" para encaixar um bloco que tem um plugue à esquerda e que retorna um valor, o qual não será considerado. O bloco que você encaixa será executado, mas o resultado retornado será ignorado. O uso típico é a execução de funções que são necessárias, mas que retornam valores
	abrir outra tela	Abre a tela com o nome fornecido. O nomeDaTela, que deve ser de uma das telas criadas com o Designer, deve ser inserido em um componente Texto e digitado exatamente como nomeado no Designer. As maiúsculas e minúsculas são importantes. Os desenvolvedores de aplicativos nunca devem fechar a Screen1 ou usar este bloco para retornar à Screen1. Se você abrir outra tela, deverá fechá-la ao retornar à tela principal para liberar memória do sistema. Deixar aberta a outra tela ao sair pode causar problemas de memória
	abrir outra tela com valor inicial	Abre outra tela e passa um valor para ela
	obter texto puro inicial	Retorna o texto simples que foi passado para esta tela quando foi iniciado por outro aplicativo. Se nenhum valor for passado, ele retorna o texto vazio. Para aplicativos de tela múltipla, use obter valor inicial em vez de obter texto puro inicial
	obter valor inicial	Retorna o valor inicial fornecido para a tela atual. Esse valor é fornecido usando-se abrir outra tela com valor inicial ou fechar tela com valor
	fechar tela	Fecha a tela atual
	fechar tela com texto puro	Fecha a tela atual e passa o texto para o aplicativo que a abriu. Este comando é para retornar texto para atividades que não sejam do App Inventor, não para telas do App Inventor. Para telas do App Inventor, como em vários aplicativos de tela, use fechar tela com valor
	fechar tela com valor	Fecha a tela atual e retorna um valor para a tela que a abriu
	fechar aplicação	Fecha o aplicativo
	break	Utilizado para interromper repetições como para, para cada, enquanto e sair sem terminar

Por uma questão de organização, veremos o grupo Variáveis antes dos outros.

# PROGRAMAÇÃO PARA DISPOSITIVOS MÓVEIS

Define-se uma variável quando seu aplicativo precisa lembrar algo que não está sendo armazenado dentro de uma propriedade de componente.

Considere o quadro a seguir.

**Quadro 18 – Grupo Variáveis**

Bloco	Comando	Comportamento
	inicializar global para	É usado para criar variáveis globais. A variável aceita qualquer tipo de valor como argumento. O bloco é independente porque as variáveis globais são usadas em todos os procedimentos ou eventos. As variáveis podem ser renomeadas neste bloco, e quaisquer blocos associados terão a referência ao nome antigo atualizada automaticamente
	obter	Devolve o valor armazenado na variável
	ajustar para	Atribui um valor para a variável
	inicializar local para	É um modificador que permite criar novas variáveis, as quais são usadas apenas no procedimento que você executa dentro do bloco. Dessa forma, todas as variáveis neste procedimento começarão com o mesmo valor cada vez que o procedimento for executado

No próximo exemplo, duas variáveis (pontos e incremento) são criadas e inicializadas com os seus valores. Depois, o conteúdo dessas variáveis é somado, e o resultado é armazenado na variável pontos. Veja a figura.



Figura 84 – Criação e operação com variáveis

Em comparações lógicas, podem ser feitas operações. Os blocos do grupo interno Lógica fazem comparações e geram como resultado verdadeiro ou falso. Veja o quadro.

**Quadro 19 – Grupo Lógica**

Bloco	Comando	Comportamento
	verdadeiro	Representa o valor constante verdadeiro
	falso	Representa o valor constante falso
	não	Executa a negação lógica, retornando falso se a entrada for verdadeira e verdadeiro se a entrada for falsa

Bloco	Comando	Comportamento
	=	<p>Testa se seus argumentos são iguais:</p> <ul style="list-style-type: none"> <li>- Dois números são iguais se forem numericamente iguais; por exemplo, 1 é igual a 1,0</li> <li>- Dois blocos de texto são iguais se tiverem os mesmos caracteres na mesma ordem, e maiúsculas e minúsculas iguais; por exemplo, banana não é igual a Banana</li> <li>- Números e texto são iguais se o número for numericamente igual a um número que seria impresso com aquele texto; por exemplo, 12,0 é igual ao resultado de juntar o primeiro caractere de 1A ao último caractere de dia2</li> <li>- Duas listas são iguais se tiverem o mesmo número de elementos e os elementos correspondentes forem iguais</li> </ul>
	≠	Testa para verificar se dois argumentos não são iguais
	e	Testa se todas as condições lógicas de um conjunto são verdadeiras
	ou	Testa se alguma das condições de um conjunto de condições lógicas é verdadeira

As operações lógicas obedecem a regras comuns. Essas regras são chamadas de **tabela-verdade**.

A operação negação utiliza o bloco "não" e inverte o valor dos resultados lógicos. Como vemos na figura a seguir, caso a condição seja não falsa (isto é, verdadeira), será executada a ação do então, apresentando, na Legenda1, Verdadeiro. Caso a condição seja não verdadeira (isto é, falsa), será executará a ação do senão, apresentando, na Legenda1, Falso.

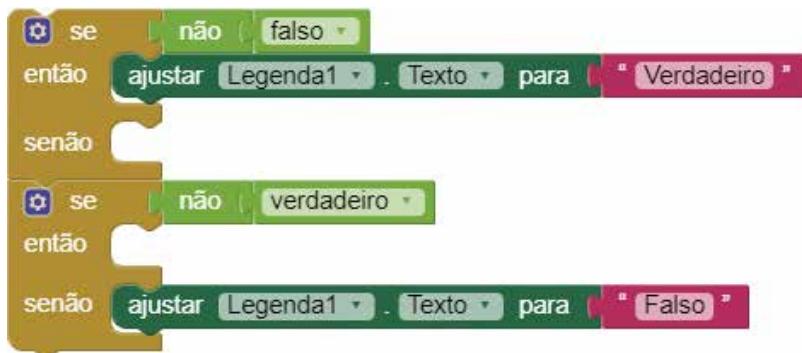


Figura 85 – Tabela-verdade da negação

O operador "e" resulta em verdadeiro apenas quando todos os seus operandos forem verdadeiros. Veja a figura.

# PROGRAMAÇÃO PARA DISPOSITIVOS MÓVEIS



Figura 86 – Tabela-verdade da operação "e"

O operador "ou" resulta em verdadeiro quando qualquer um dos operadores for verdadeiro. Veja a figura.

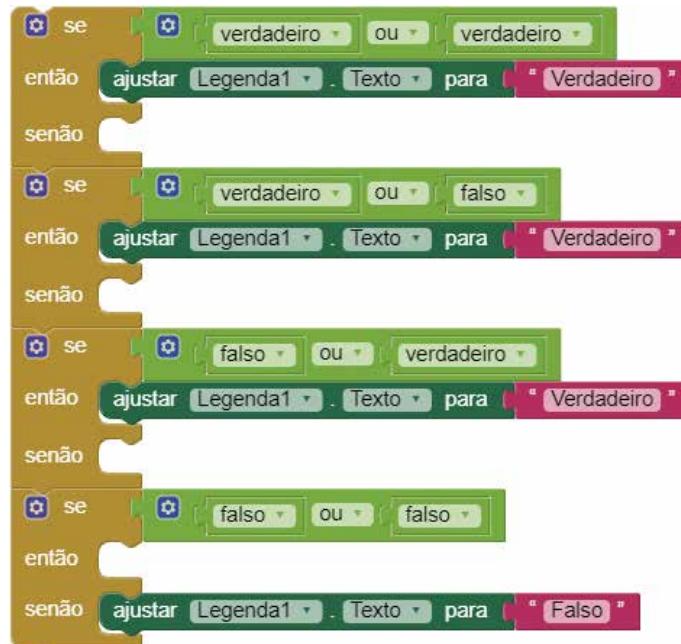


Figura 87 – Tabela-verdade da operação "ou"

Vamos montar um simples teste para verificar as operações lógicas. Analisaremos algumas condições de sobrevivência entre um super-herói e um ser humano normal. Para montar esse simples exemplo, siga os passos mostrados no quadro a seguir.

## Quadro 20 – Sequência de operações do Exemplo2

Passo	Paleta	Componente	Operação
1	Interface de Usuário	Legenda	Arrastar para Screen1
2	Interface de Usuário	Botão	Arrastar para Screen1

Na figura a seguir, temos o resultado dos passos do quadro.

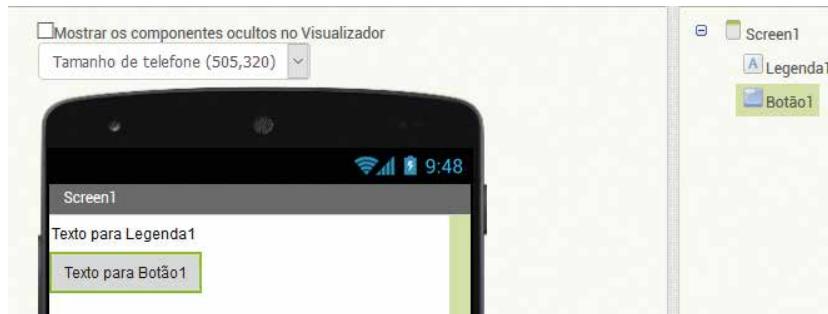


Figura 88 – Tela do Exemplo2

Na tela do editor de blocos, efetue a montagem dos blocos, conforme o que se vê no quadro.

## Quadro 21 – Montagem dos blocos

Passo	Paleta	Componente	Operação	
1	Variáveis	inicializar global	Colocar o nome Tiro	
2	Lógica	verdadeiro	Encaixar em inicializar Tiro	
3	Variáveis	inicializar global	Colocar o nome Facada	
4	Lógica	verdadeiro	Encaixar em inicializar Facada	
5	Botão1	.Clique		
6	Controle	se/então/senão	fazer de Botão1.Clique	
7	Lógica	e	Encaixar em se de se/então/senão	
8	Variáveis	obter	Preencher o 1º operador de "e"	Tiro
9	Variáveis	obter	Preencher o 2º operador de "e"	Facada
10	Legenda1	ajustar.Texto	Encaixar em então de se/então/senão	
11	Texto	“ ”	Encaixar em Legenda1	“Morreu”
12	Legenda1	ajustar.Texto	Encaixar em senão de se/então/senão	
13	Texto	“ ”	Encaixar em Legenda1	“Vivo”

Uma vez editados os blocos, o visualizador se apresenta como mostrado na figura a seguir.

# PROGRAMAÇÃO PARA DISPOSITIVOS MÓVEIS



Figura 89 – Blocos montados

Para fazermos o teste, podemos utilizar tanto o emulador quanto o AI2 Companion.

Na atual configuração, ao darmos um clique no botão da tela, o resultado é Morreu. Isso significa que a pessoa que levou um tiro (verdadeiro) e uma facada morreu. Veja a figura.



Figura 90 – Tela ao clicar no Botão1

Dessa forma, alterando a inicialização das variáveis, podemos montar este quadro.

## Quadro 22 – Resultado do exemplo utilizando “e”

Initializations	Result	Description
<pre>iniciar global Tiro para verdadeiro iniciar global Facada para verdadeiro</pre>	<p>Screen1 Morreu Texto para Botão1</p>	Tomando tiro e facada, morre
<pre>iniciar global Tiro para verdadeiro iniciar global Facada para falso</pre>	<p>Screen1 Vivo Texto para Botão1</p>	Só tomado tiro, sobrevive
<pre>iniciar global Tiro para falso iniciar global Facada para verdadeiro</pre>	<p>Screen1 Vivo Texto para Botão1</p>	Só tomado facada, sobrevive
<pre>iniciar global Tiro para falso iniciar global Facada para falso</pre>	<p>Screen1 Vivo Texto para Botão1</p>	Sem tomar tiro e facada, sobrevive

Repetindo as alterações da inicialização das variáveis, porém mudando a operação para "ou", obtemos o quadro a seguir.

**Quadro 23 – Resultado do exemplo utilizando "ou"**

<pre>obter global Tiro ou obter global Facada</pre>	<pre>Screen1 Morreu Texto para Botão1</pre>	Tomando tiro e facada, morre
<pre>inicializar global Tiro para verdadeiro inicializar global Facada para verdadeiro</pre>	<pre>Screen1 Morreu Texto para Botão1</pre>	Só tomado tiro, morre
<pre>inicializar global Tiro para falso inicializar global Facada para verdadeiro</pre>	<pre>Screen1 Morreu Texto para Botão1</pre>	Só tomado facada, morre
<pre>inicializar global Tiro para falso inicializar global Facada para falso</pre>	<pre>Screen1 Vivo Texto para Botão1</pre>	Sem tomar tiro e facada, sobrevive

No nosso exemplo, no caso do "e", a condicional pode ser utilizada para super-heróis que, para morrer, precisam que ambas as condições aconteçam, ou seja, tomar facada e tomar tiro.

No exemplo, o "ou" se aplica ao caso de um personagem comum. Basta tomar ou uma facada ou um tiro que ele é abatido.

As funções matemáticas são blocos para montar expressões. Como há muitas funções, somente as principais e alguns blocos matemáticos são menus suspensos e podem ser convertidos em diferentes blocos. Veja o quadro.

**Quadro 24 – Grupo Matemática**

Bloco	Comando	Comportamento
	Valor	Insere um valor numérico
	Operações aritméticas	+ - × ÷

# PROGRAMAÇÃO PARA DISPOSITIVOS MÓVEIS

Bloco	Comando	Comportamento
	Potenciação	$a^b$
	bitwise	Toma dois números e compara cada par de bits
	inteiro aleatório	Retorna um valor inteiro aleatório entre os valores fornecidos, inclusive os extremos do intervalo
	fração	Retorna um valor aleatório entre 0 e 1
	ajusta	Gera uma semente, que é uma sequência repetível de números aleatórios. Assim, sempre gera a mesma sequência de números aleatórios a partir da primeira semente do conjunto aleatório com o mesmo valor. É útil para testar programas que envolvem valores aleatórios
	mín/máx	Retorna o maior ou o menor valor e um conjunto de valores
	Funções matemáticas	
	Divisões	Retorna o módulo, o resto ou o quociente da divisão de um número por outro
	Funções trigonométricas	
	formatar decimal	Determina o número de casas decimais de um número
	Converte	O número será "traduzido" em outra base numérica, que pode ser decimal (também conhecido como base 10), binária, octal ou hexadecimal
	Conversão trigonométrica	Conversão angular

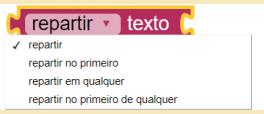
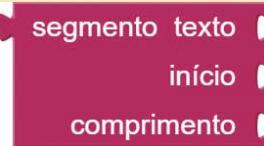
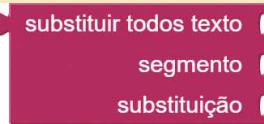
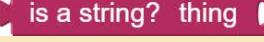
Bloco	Comando	Comportamento
	Conversão computacional	
	é	Retorna verdadeiro ou falso, conforme o tipo do número de entrada
	Comparação de grandezas	Retorna verdadeiro ou falso, conforme o resultado da expressão avaliada

Os blocos de texto servem para manipular elementos textuais, ou seja, cadeias de caracteres (letras, números ou símbolos). No App Inventor, os textos são considerados objetos. Veja o quadro.

### Quadro 25 – Grupo Texto

Bloco	Comando	Comportamento
	""	Contém uma cadeia de texto
	juntar	Acrescenta todas as entradas para fazer um único texto. Se não houver entradas, retorna um texto vazio
	comprimento	Retorna o número de caracteres, incluindo espaços, no texto. Esse é o comprimento do texto
	é vazio?	Retorna se o texto contém ou não caracteres (incluindo espaços). Quando o comprimento do texto é 0, retorna verdadeiro; caso contrário, retorna falso
	comparar textos	Retorna se o primeiro texto é lexicograficamente <, >, = ou ≠ em relação ao segundo texto, dependendo de qual lista suspensa está selecionada. Um texto é considerado caractere a caractere maior que outro se for alfabeticamente maior que o outro texto. Essencialmente, viria depois no dicionário. Todas as letras maiúsculas são consideradas menores que as minúsculas
	apurar espaços	Retorna removendo todos os espaços à esquerda ou à direita do texto de entrada
	maiúsculas/minúsculas	Retorna uma cópia de seu argumento de texto convertido em maiúsculas/minúsculas
	começa em	Retorna a posição do caractere em que o primeiro trecho da parte aparece pela primeira vez no texto ou 0 se não estiver presente. Por exemplo, a localização de "ete" em "Sorvete derrete" é 5

# PROGRAMAÇÃO PARA DISPOSITIVOS MÓVEIS

Bloco	Comando	Comportamento
	contém	Retorna verdadeiro se algum dos trechos na lista de partes aparecer no texto; caso contrário, retorna falso. Este bloco pode ser obtido alterando-se a lista suspensa no bloco contém. As opções são todos ou qualquer
	repartir	Divide o texto fornecido em partes usando a localização ocorrência de uma cadeia especificada, retornando uma lista de itens
	repartir	Divide o texto dado em qualquer ocorrência de um espaço, produzindo uma lista das palavras
	segmento	Extrai parte do texto começando na posição inicial e continuando para caracteres de comprimento
	substituir todos	Retorna um novo texto obtido pela substituição de todas as ocorrências do segmento trocadas pelo texto dado em substituição
	Texto Ofuscado	Produz texto como um bloco de texto. A diferença é que o texto não é facilmente detectável examinando o conteúdo do aplicativo. Isso é criado em aplicativos para distribuir que incluem informações confidenciais – por exemplo, chaves de API. (Aviso: isso fornece apenas uma segurança muito baixa contra adversários experientes)
	é texto?	Retorna verdadeiro se thing for um objeto de texto; caso contrário, retorna falso
	reverso	Inverte o texto fornecido
	substituir todos os mapeamentos	Dado um dicionário de mapeamentos como entrada, substitui as entradas principais no texto pelos valores correspondentes no dicionário. Retorna o texto com os mapeamentos aplicados

Listas são um tipo de estrutura de dados usado em todas as linguagens de programação, não apenas no App Inventor. Elas são empregadas para criar e manipular diferentes conjuntos de valores e seus elementos. Por exemplo, um jogo pode manter uma lista de pontuações mais altas, e seu aplicativo do Facebook mantém uma lista de seus amigos. Para facilitar o acesso a um elemento da lista, utilizamos o índice. No App Inventor, o primeiro elemento em uma lista é o índice 1. Na figura a seguir, a tem o índice 1, b tem o índice 2 e c tem o índice 3. Isso significa que podemos fazer referência a um elemento específico dentro de nossa lista se soubermos qual índice ele possui e qual é o nome da lista.

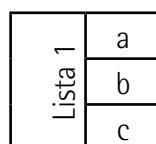
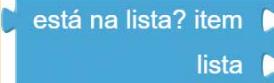
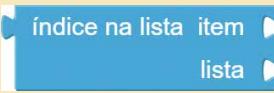
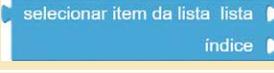
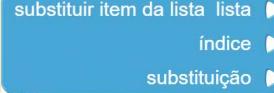
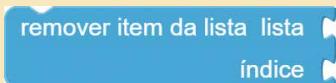


Figura 91 – Exemplo de lista

Para a criação e a manipulação das listas, temos as referências dos blocos mostradas no quadro a seguir.

## Quadro 26 – Grupo Listas

Bloco	Comando	Comportamento
 criar lista vazia	criar lista vazia	Cria uma lista vazia sem elementos em uma variável
 criar lista	criar lista	Cria uma lista dos blocos fornecidos
 adicionar itens à lista lista item	adicionar itens à lista	Adiciona os itens fornecidos ao final da lista. A diferença de anexar à lista é que este leva os itens a serem anexados como uma única lista, enquanto adicionar itens à lista leva os itens como argumentos individuais
 está na lista? item lista	está na lista?	Se item for um dos elementos da lista, retorna verdadeiro; caso contrário, retorna falso
 comprimento da lista lista	comprimento da lista	Retorna o número de itens da lista
 é lista vazia? lista	é lista vazia?	Se a lista não tiver itens, retorna verdadeiro; caso contrário, retorna falso
 escolher um item aleatoriamente lista	escolher um item aleatoriamente	Escolhe um item aleatoriamente da lista
 índice na lista item lista	índice na lista	Retorna a posição do item na lista. Se não estiver na lista, retorna 0
 selecionar item da lista lista índice	selecionar item da lista	Seleciona o item no índice fornecido, na lista fornecida. O primeiro item da lista está no índice 1
 inserir item na lista índice item	inserir item na lista	Insere um item na lista na posição fornecida
 substituir item da lista índice substituição	substituir item da lista	Insere substituição na lista fornecida no índice de posição. O item anterior naquela posição é removido
 remover item da lista índice	remover item da lista	Remove o item na posição fornecida
 acrescentar à lista lista1 lista2	acrescentar à lista	Adiciona os itens da segunda lista ao final da primeira lista
 copiar lista lista	copiar lista	Faz cópia de uma lista, incluindo todas as sublistas
 é lista? coisa	é lista?	Se coisa for uma lista, retorna verdadeiro; caso contrário, retorna falso

# PROGRAMAÇÃO PARA DISPOSITIVOS MÓVEIS

Bloco	Comando	Comportamento
	lista reversa	Faz uma cópia da lista com itens na ordem inversa. Por exemplo, reverse list [1,2,3] retorna [3,2,1]
	de lista para linha csv	Interpreta a lista como uma linha de uma tabela e retorna um texto CSV (valor separado por vírgula) que representa a linha. Cada item na lista de linhas é considerado um campo e é citado com aspas duplas no texto CSV resultante. Os itens são separados por vírgulas
	de lista para tabela csv	Interpreta a lista como uma tabela no formato de linha principal e retorna um texto CSV (valor separado por vírgula) que representa a tabela. Cada item da lista deve ser uma lista que representa uma linha da tabela CSV. Cada item na lista de linhas é considerado um campo e é citado com aspas duplas no texto CSV resultante
	de linha csv para lista	Analisa um texto como uma linha formatada em CSV (valor separado por vírgula) para produzir uma lista de campos
	de tabela csv para lista	Analisa um texto como uma tabela formatada em CSV (valor separado por vírgula) para produzir uma lista de linhas, cada uma delas sendo uma lista de campos
	busca nos pares	Pesquisa informações em uma estrutura semelhante a um dicionário representada como uma lista. Esta operação usa três entradas, uma chave, uma lista de pares e um resultado nãoEncontrado, que por padrão é definido como not found. Aqui, os pares devem ser uma lista de pares, ou seja, uma lista em que cada elemento é uma lista de dois elementos. Busca nos pares encontra o primeiro par na lista cujo primeiro elemento é a chave e retorna o segundo elemento
	juntar itens usando separador	Une todos os elementos na lista especificada pelo separador especificado, produzindo texto como resultado

Dicionários, chamados de termos em outras linguagens, como mapas, matrizes associativas ou listas, são estruturas de dados que associam um valor, geralmente chamado de chave, a outro valor. Um exemplo disso está nas agendas telefônicas, pois podemos associar uma chave nome com uma série de números. Os blocos para trabalhar com dicionários estão no quadro a seguir.

**Quadro 27 – Grupo Dicionários**

Bloco	Comando	Comportamento
	criar dicionário vazio	Cria um dicionário vazio. As entradas podem ser adicionadas ao dicionário vazio usando set value for key. Também pode ser transformado em make a dictionary usando o botão modificador azul para adicionar pairs de entradas
	fazer um dicionário	Cria um dicionário com um conjunto de pairs conhecidos de antemão. Entradas adicionais podem ser adicionadas usando set value for key

# Unidade I

Bloco	Comando	Comportamento
	chave/valor	Bloco de propósito especial usado para construir dicionários
	obter valor para a chave	Verifica se o dicionário contém um valor correspondente para a chave fornecida. Em caso afirmativo, o valor é retornado; caso contrário, o valor not found é retornado
	definir valor para a chave	Define o valor correspondente para uma key (chave) em dictionary (nome do dicionário) para value (valor). Se nenhum mapeamento existir para key, uma nova chave será criada; caso contrário, o valor existente será substituído pelo novo valor
	deletar entrada para chave	Remove o mapeamento de valor-chave no dicionário para a chave fornecida
	obter valor no caminho da chave	É uma versão mais avançada do get value for key. Ao contrário de obter o valor de uma chave específica, ele obtém uma lista de chaves e números válidos que representam um caminho por meio de uma estrutura de dados
	definir valor para o caminho da chave	Atualiza o valor em um key path específico em uma estrutura de dados
	pegar as chaves	Retorna uma lista de chaves no dicionário. Modificar o conteúdo de um valor na lista também o modificará no dicionário
	a chave está no dicionário?	Testa se a chave existe no dicionário
	tamanho do dicionário	Retorna o número de pares de valores-chave presentes no dicionário
	lista de pares para dicionário	Converte uma lista associativa do formulário (key1 value1, key2 value2 etc.) em um dicionário que mapeia as chaves para seus valores
	dicionário para lista de pares	Converte um dicionário em uma lista associativa. Este bloco reverte a conversão realizada pelo list of pairs to dictionary
	copiar dicionário	Faz uma cópia detalhada do dicionário fornecido. Isso significa que todos os valores são copiados recursivamente e que alterar um valor na cópia não o alterará no original
	fundir no dicionário	Copia os pares de valores-chave de um dicionário para outro, sobrescrevendo quaisquer chaves no dicionário de destino
	listar pelo percurso do caminho da chave do dicionário ou da lista	Atua de forma semelhante ao get value at key path, mas cria uma lista de valores em vez de retornar um único valor. Funciona começando no dicionário fornecido e descendo a árvore de objetos, seguindo o caminho fornecido
	andar por todo o nível	É um bloco especializado que pode ser usado no caminho da chave de list by walking key path. Quando encontrado durante uma caminhada, ele faz com que todos os itens daquele nível sejam explorados
	é um dicionário?	Testa se o bloco conectado a ele é um dicionário ou não

# PROGRAMAÇÃO PARA DISPOSITIVOS MÓVEIS

O App Inventor faz a manipulação de cores armazenando cada cor como um único número. Esse número é formado pela combinação das cores vermelho, verde e azul em termos de intensidade entre 0 e 255. Para simplificar, o App Inventor oferece uma série de blocos com as cores básicas. Os blocos relativos a cores estão no quadro.

**Quadro 28 – Grupo Cores**

Bloco	Comando	Comportamento
	Cor básica	Representa a cor armazenada internamente no bloco. Se você clicar na cor do meio, um pop-up aparecerá na tela com uma tabela de setenta cores à sua escolha. Ao clicar em uma nova cor, a cor do seu bloco de cores básico será atualizada
	criar cor	Recebe uma lista de três números. Esses números na lista representam valores em um código RGB. Os códigos RGB são usados para fazer cores na internet
	repartir cor	Recebe uma cor – bloco de cor, variável que contém uma cor ou propriedade de um dos componentes que representam uma cor – e retorna uma lista dos valores RGB no código RGB dessa cor

O procedimento é um subprograma, uma sequência de blocos ou um código armazenado sob um nome. Ele evita a repetição de código. A comunicação com o procedimento é feita por meio de parâmetros. Para a construção dos procedimentos, a referência está no quadro a seguir.

**Quadro 29 – Grupo Procedimentos**

Bloco	Comando	Comportamento
	para procedimento/fazer	Executa a sequência de blocos em fazer e, se tiver argumentos, eles serão criados usando o botão modificador do bloco
	Função	Executa a sequência de blocos em fazer e, se tiver argumentos, eles serão criados usando o botão modificador do bloco, retornando um valor ao final da execução
	chamar procedimento	O App Inventor gera automaticamente um bloco de chamada e o coloca no grupo de procedimentos. Esse bloco executa o procedimento e retoma a execução no retorno do procedimento
	= chamar função	O App Inventor gera automaticamente um bloco de chamada e o coloca no grupo de procedimentos. Esse bloco executa a função e retoma a execução recebendo o valor devolvido pela função

## 3.4 Criação do app para a demonstração de componentes

Vamos criar uma calculadora simples que faça as quatro operações básicas da aritmética.

Inicie um novo projeto e chame-o de **QuatroOperacoes**.

Vamos alterar a imagem de fundo e o título do aplicativo seguindo os passos que aparecem na tela. Veja o quadro e a figura a seguir.

**Quadro 30 – Tela de fundo**

Passo	Objeto	Propriedade	Valor
1	Screen1	ImagenDeFundo	pdm0004.jpg
2	Screen1	Título	As 4 operações



Figura 92 – Imagem de fundo e título do aplicativo alterados

A inserção de novos componentes ocorrerá sempre na parte superior esquerda da área de visualização.

Para evitar a sobreposição de componentes, necessita-se de um componente para organizar a visualização da mensagem.

Os componentes da guia Organização têm esta finalidade. Veja o quadro a seguir.

**Quadro 31 – Montagem do cabeçalho**

Passo	Janela	Componente	Propriedade	Operação
1	Paleta	Organização	OrganizaçãoHorizontal1	Arrastar para o Visualizador
2	Propriedades	OrganizaçãoHorizontal1	AlinhamentoHorizontal	Centro
3	Propriedades	OrganizaçãoHorizontal1	AlinhamentoVertical	Base
4	Propriedades	OrganizaçãoHorizontal1	CorDeFundo	Nenhum (altera a cor de fundo para transparente)
5	Propriedades	OrganizaçãoHorizontal1	Altura	80 pontos
6	Propriedades	OrganizaçãoHorizontal1	Largura	Preencher principal
7	Paleta	Interface de Usuário	Legenda1	Arrastar para a OrganizaçãoHorizontal
8	Propriedades	Legenda1	FonteNegrito	Marcar
9	Propriedades	Legenda1	TamanhoDaFonte	25
10	Propriedades	Legenda1	Largura	Preencher principal
11	Propriedades	Legenda1	Texto	AS 4 OPERAÇÕES
12	Propriedades	Legenda1	AlinhamentoDoTexto	Centro
13	Propriedades	Legenda1	CorDeTexto	Branco

Uma vez montada a primeira OrganizaçãoHorizontal, temos o layout mostrado na figura a seguir.

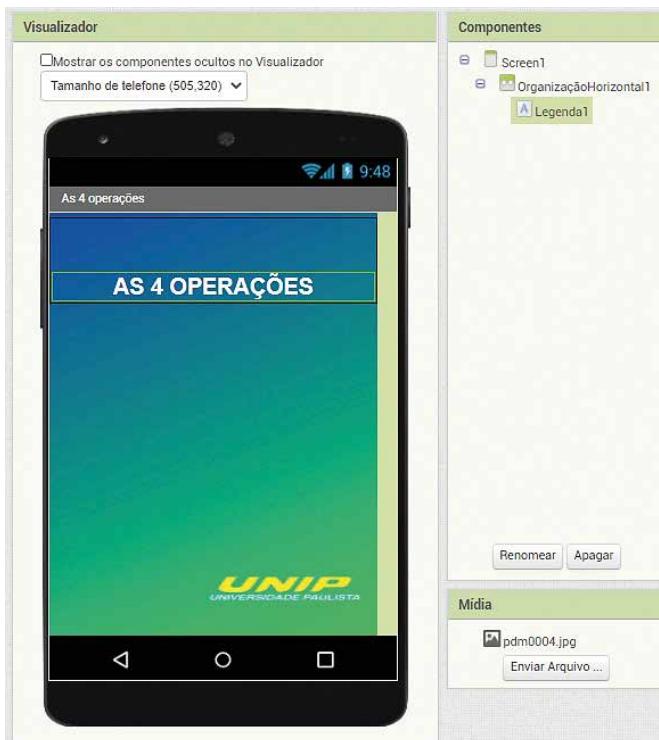


Figura 93 – Primeira OrganizaçãoHorizontal montada

Abaixo do cabeçalho será colocado o campo para a leitura do primeiro número. Veja o quadro a seguir.

## Quadro 32 – Organização da entrada do primeiro número

Passo	Janela	Componente	Propriedade	Operação
1	Paleta	Organização	OrganizaçãoHorizontal2	Arrastar para o Visualizador
2	Propriedades	OrganizaçãoHorizontal2	AlinhamentoHorizontal	Centro
3	Propriedades	OrganizaçãoHorizontal2	AlinhamentoVertical	Centro
4	Propriedades	OrganizaçãoHorizontal2	CorDeFundo	Nenhum (altera a cor de fundo para transparente)
5	Propriedades	OrganizaçãoHorizontal2	Largura	Preencher principal
6	Paleta	Interface de Usuário	Legenda2	Arrastar para a OrganizaçãoHorizontal2
7	Propriedades	Legenda2	FonteNegrito	Marcar
8	Propriedades	Legenda2	TamanhoDaFonte	16
9	Propriedades	Legenda2	Largura	Preencher principal
10	Propriedades	Legenda2	Texto	Número
11	Propriedades	Legenda2	CorDeTexto	Branco
12	Paleta	Interface de Usuário	CaixaDeTexto1	Arrastar para a OrganizaçãoHorizontal2, à direita de Legenda2
13	Propriedades	CaixaDeTexto1	Dica	Apagar o texto
14	Propriedades	CaixaDeTexto1	SomenteNúmeros	Marcar
15	Propriedades	CaixaDeTexto1	Texto	Número
16	Componentes	Renomear	CaixaDeTexto1	Txt_n1

Realizados os passos mostrados no quadro, observe que a OrganizaçãoHorizontal da segunda altura, por estar em automático, ajustou-se à altura da caixa de texto. Na janela Componentes, verifique a hierarquia dos componentes e a caixa de texto renomeado. Veja a figura.

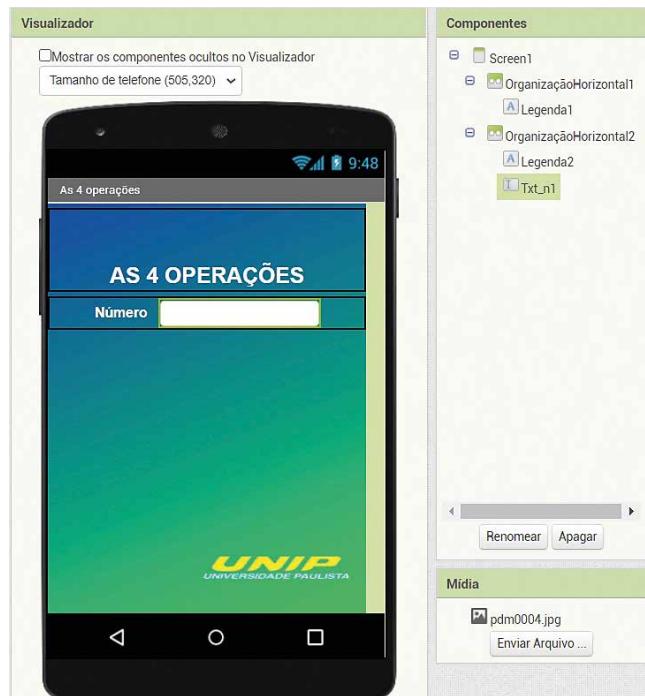


Figura 94 – Tela com o primeiro número

# PROGRAMAÇÃO PARA DISPOSITIVOS MÓVEIS

Reita o procedimento para a criação de outra legenda para a entrada de outro número. Renomeie a segunda CaixaDeTexto de Txt\_n2. Veja o quadro.

**Quadro 33 – Inserção de uma nova entrada de número**

Passo	Janela	Componente	Propriedade	Operação
1	Paleta	Organização	OrganizaçãoHorizontal3	Arrastar para o Visualizador
2	Propriedades	OrganizaçãoHorizontal3	AlinhamentoHorizontal	Centro
3	Propriedades	OrganizaçãoHorizontal3	AlinhamentoVertical	Centro
4	Propriedades	OrganizaçãoHorizontal3	CorDeFundo	Nenhum (altera a cor de fundo para transparente)
5	Propriedades	OrganizaçãoHorizontal3	Largura	Preencher principal
6	Paleta	Interface de Usuário	Legenda3	Arrastar para a OrganizaçãoHorizontal3
7	Propriedades	Legenda3	FonteNegrito	Marcar
8	Propriedades	Legenda3	TamanhoDaFonte	16
9	Propriedades	Legenda3	Largura	Preencher principal
10	Propriedades	Legenda3	Texto	Número 2
11	Propriedades	Legenda3	CorDeTexto	Branco
12	Paleta	Interface de Usuário	CaixaDeTexto2	Arrastar para a OrganizaçãoHorizontal3, à direita de Legenda3
13	Propriedades	CaixaDeTexto2	Dica	Apagar o texto
14	Propriedades	CaixaDeTexto2	SomenteNúmeros	Marcar
15	Propriedades	CaixaDeTexto2	Texto	Número 2
16	Componentes	Renomear	CaixaDeTexto2	Txt_n2

Feitas as operações acima, teremos um layout semelhante ao apresentado na figura.



Figura 95 – As duas entradas numéricas criadas

Adicionalmente, vamos implementar um separador (OrganizaçãoHorizontal) entre a entrada dos números e os botões das operações. Veja o quadro e a figura a seguir.

**Quadro 34 – Inserção do separador e dos botões das operações**

Passo	Janela	Componente	Propriedade	Operação
1	Paleta	Organização	OrganizaçãoHorizontal4	Arrastar para o Visualizador
2	Propriedades	OrganizaçãoHorizontal4	Altura	Preencher principal
3	Propriedades	OrganizaçãoHorizontal4	Largura	Centro
4	Paleta	Organização	OrganizaçãoHorizontal5	Arrastar para o Visualizador
5	Propriedades	OrganizaçãoHorizontal5	AlinhamentoHorizontal	Centro
6	Propriedades	OrganizaçãoHorizontal5	Largura	Preencher principal
7	Paleta	Interface de Usuário	Botão1	Arrastar para a OrganizaçãoHorizontal5
8	Propriedades	Botão1	TamanhoDaFonte	30
9	Propriedades	Botão1	Altura	50
10	Propriedades	Botão1	Largura	40
11	Propriedades	Botão1	Texto	+
12	Componentes	Renomear	Botão1	Btn_Soma
13	Paleta	Interface de Usuário	Botão1	Arrastar para a OrganizaçãoHorizontal5, à direita de Btn_Soma
14	Propriedades	Botão1	TamanhoDaFonte	30
15	Propriedades	Botão1	Altura	50
16	Propriedades	Botão1	Largura	40
17	Propriedades	Botão1	Texto	-
18	Componentes	Renomear	Botão1	Btn_Sub
19	Paleta	Interface de Usuário	Botão1	Arrastar para a OrganizaçãoHorizontal5, à direita de Btn_Sub
20	Propriedades	Botão1	TamanhoDaFonte	30
21	Propriedades	Botão1	Altura	50
22	Propriedades	Botão1	Largura	40
23	Propriedades	Botão1	Texto	X
24	Componentes	Renomear	Botão1	Btn_Mult
25	Paleta	Interface de Usuário	Botão1	Arrastar para a OrganizaçãoHorizontal5, à direita de Btn_Mult
26	Propriedades	Botão1	TamanhoDaFonte	30
27	Propriedades	Botão1	Altura	50
28	Propriedades	Botão1	Largura	40
29	Propriedades	Botão1	Texto	/
30	Componentes	Renomear	Botão1	Btn_Div

# PROGRAMAÇÃO PARA DISPOSITIVOS MÓVEIS

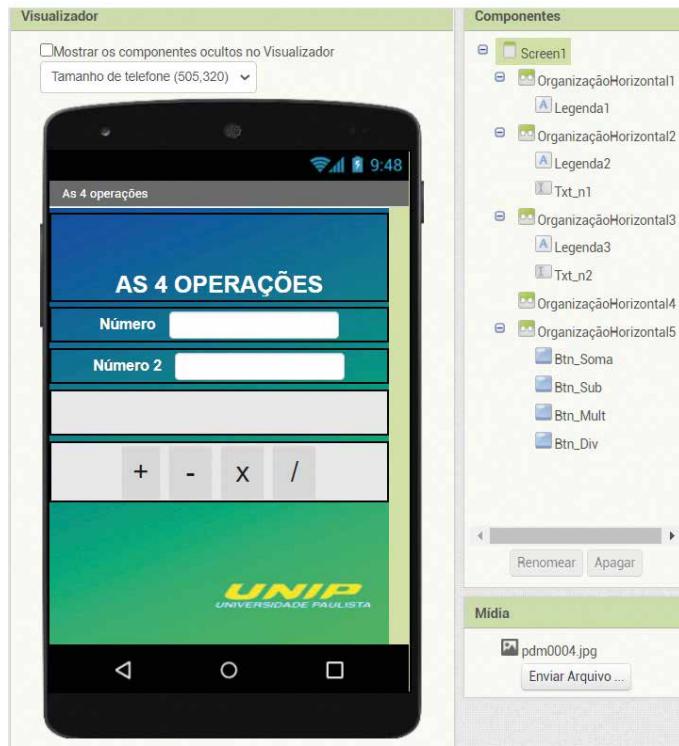


Figura 96 – Botões das operações

No quadro a seguir estão os passos para apresentar o resultado.

## Quadro 35 – Montagem da seção do resultado

Passo	Janela	Componente	Propriedade	Operação
1	Paleta	Organização	OrganizaçãoHorizontal6	Arrastar para o Visualizador
2	Propriedades	OrganizaçãoHorizontal6	AlinhamentoHorizontal	Centro
3	Propriedades	OrganizaçãoHorizontal6	AlinhamentoVertical	Centro
4	Propriedades	OrganizaçãoHorizontal6	Altura	50
5	Propriedades	OrganizaçãoHorizontal6	Largura	Preencher principal
6	Paleta	Interface de Usuário	Legenda4	Arrastar para a OrganizaçãoHorizontal6
7	Propriedades	Legenda4	TamanhoDaFonte	20
8	Propriedades	Legenda4	Texto	Resultado
9	Paleta	Interface de Usuário	Legenda5	Arrastar para a OrganizaçãoHorizontal6, à direita de Legenda4
10	Propriedades	Legenda5	TamanhoDaFonte	20
11	Propriedades	Legenda5	Texto	Resultado
12	Componentes	Renomear	CaixaDeTexto1	Lbl_Resultado

Observe a figura.

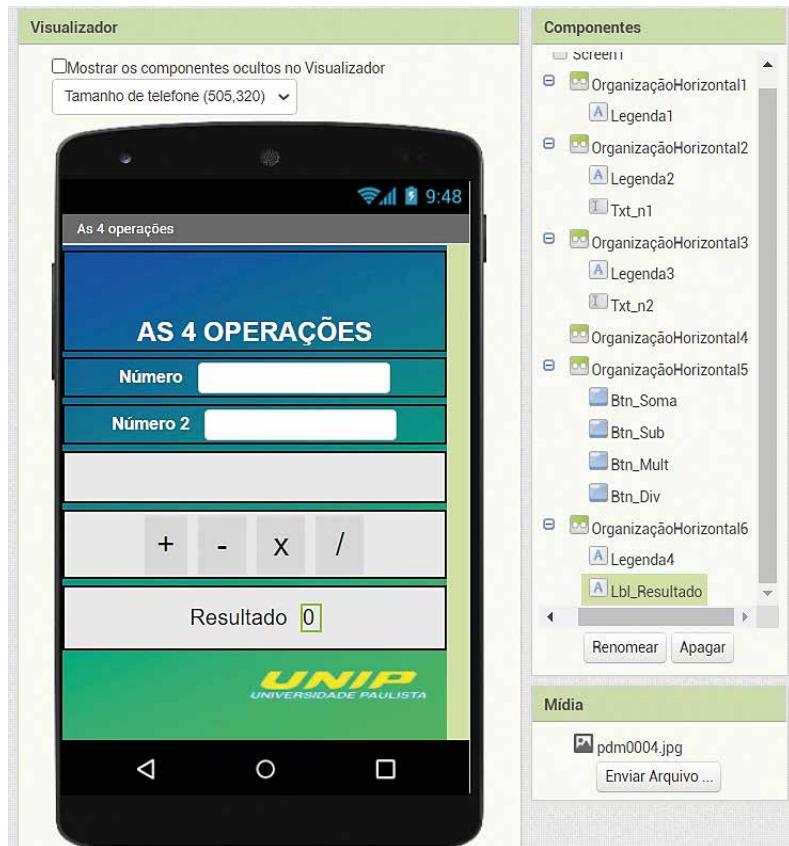


Figura 97 – Tela do design pronto

Na figura, temos a tela do design pronto. Observe a janela Componentes e a sua disposição. A importância de renomear os componentes pode ser vista no caso dos botões das operações. Caso não tivessem sido renomeados, eles seriam Botão1, Botão2, Botão3 e Botão4, o que dificultaria identificar as funcionalidades deles na programação.

Vamos agora acessar o editor de blocos. Utilizaremos duas variáveis, cada uma para receber o valor (conteúdo) de cada número que será digitado na caixa de texto. Elas serão inicializadas com o valor zero.

## Observação

Vale observar o que segue.

- As variáveis são entidades para o armazenamento de conteúdos na memória do computador.
- As variáveis são constituídas por um identificador, um tipo de conteúdo e um endereço de memória.

# PROGRAMAÇÃO PARA DISPOSITIVOS MÓVEIS

Veja o quadro e a figura a seguir.

**Quadro 36 – Inicialização das variáveis**

Passo	Paleta	Componente	Operação	
1	Variáveis	inicializar global	Colocar o nome numero1	
2	Matemática	Valor	0	Encaixar em inicializar numero1
3	Variáveis	inicializar global	Colocar o nome numero2	
4	Matemática	Valor	0	Encaixar em inicializar numero2



Figura 98 – Blocos da inicialização montados

Uma vez montadas as variáveis, chegamos ao coração do programa, que é a resposta do aplicativo ao clique nos botões das operações.

Na janela de blocos, abaixo dos internos, está uma reprodução exata da árvore da janela Componentes. Nela estarão os botões e, como estão renomeados, é muito fácil localizar o botão desejado. Vamos editar o botão de soma (Btn\_Soma). Ao clicarmos sobre ele, irão aparecer todos os blocos pertinentes a eventos sobre esse botão.

Arrastamos para o Visualizador a operação relacionada ao evento de clicar sobre o botão. Veja a figura.

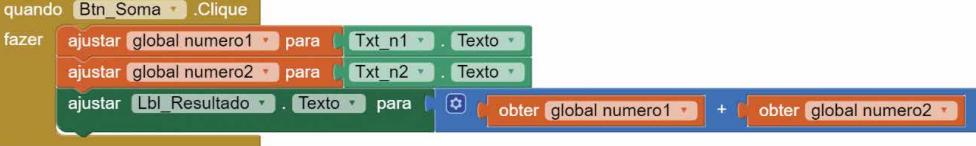


Figura 99 – Localização do botão Btn\_Soma e o evento Clique associado a ele

O tratamento do comportamento ao evento Clique no Btn\_Soma acontecerá no espaço **fazer**, que é ajustável.

Nesse espaço, os valores digitados nas caixas de texto Txt\_n1 e Txt\_n2 serão armazenados nas variáveis numero1 e numero2 respectivamente. Depois, a legenda Lbl\_resultado receberá o resultado da operação aritmética entre numero1 e numero2, exibindo imediatamente o resultado. Veja o quadro.

## Quadro 37 – Montagem do bloco de tratamento do evento Clique no botão soma

Passo	Paleta	Componente	Operação	
1	Btn_Soma	Btn_Soma.Clique	Arrastar para o Visualizador	
2	Variáveis	ajustar para	numero1	fazer de Btn_Soma.Clique
3	Txt_n1	Txt_n1.Texto	Encaixar em ajustar numero1	
4	Variáveis	ajustar para	numero2	Encaixar abaixo de ajustar numero1
5	Txt_n2	Txt_n2.Texto	Encaixar em ajustar numero2	
				
6	lbl_Resultado	lbl_Resultado.Texto	Encaixar abaixo de ajustar numero2	
7	Matemática	+	Encaixar em lbl_Resultado.Texto	
				
8	Variáveis	obter	Preencher o 1º operador de +	numero1
9	Variáveis	obter	Preencher o 2º operador de +	numero2
				

O editor permite alguns atalhos na edição. No bloco montado, ao darmos um clique com o botão direito do mouse, abre-se uma aba com opções. Veja a figura a seguir.

# PROGRAMAÇÃO PARA DISPOSITIVOS MÓVEIS

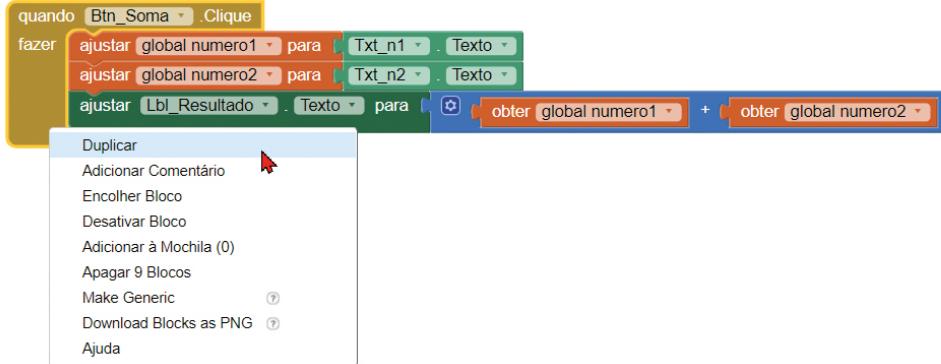


Figura 100 – Clique no bloco com o botão direito do mouse

Ao duplicarmos, todo o bloco será copiado e será apresentado um erro, pois um botão está tratando o mesmo evento. Para corrigir, basta alterarmos o nome do objeto para Btn\_Sub. Veja a figura.

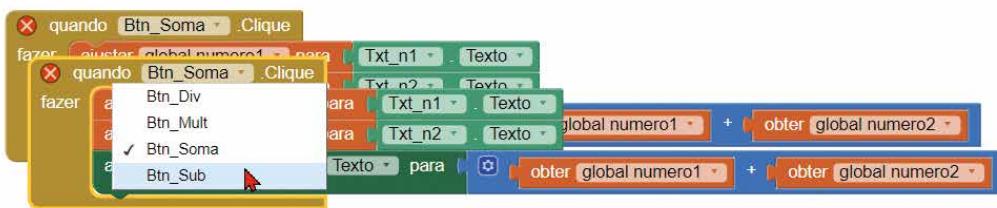


Figura 101 – Trocando o objeto fonte

No novo bloco, deve ser corrigida a operação aritmética. O bloco excedente pode ser descartado. Veja a figura.

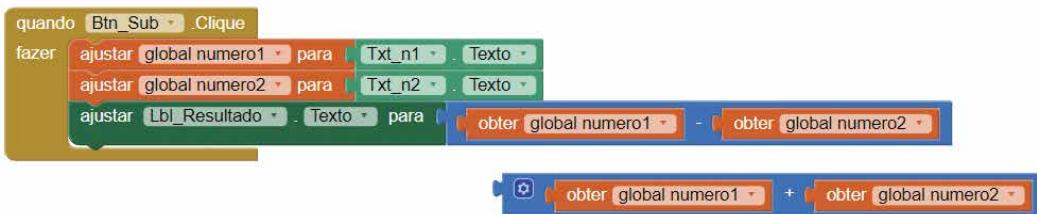


Figura 102 – Trocando a operação

A mesma estratégia pode ser feita para os botões Btn\_Mult e Btn\_Div. Veja a figura a seguir.

# Unidade I



Figura 103 – Situação final dos blocos do app As 4 operações

## 4 TESTE DE APP

Para testar o aplicativo, conectaremos o AI2 Companion via Wi-Fi. Assim que o dispositivo sincronizar, veremos a tela inicial com os campos em branco e o valor do resultado em branco. No topo da tela, a identificação "As 4 operações", não o tradicional Screen1, é a consequência da alteração da propriedade Título do componente Screen1.



### Lembrete

Para executar, utilize a opção Assistente AI para ler o QR code no dispositivo móvel e executar no MIT AI2 Companion do seu dispositivo móvel, ou a opção Emulador para aiStarter do computador do menu superior Conectar.

# PROGRAMAÇÃO PARA DISPOSITIVOS MÓVEIS

Veja as figuras.

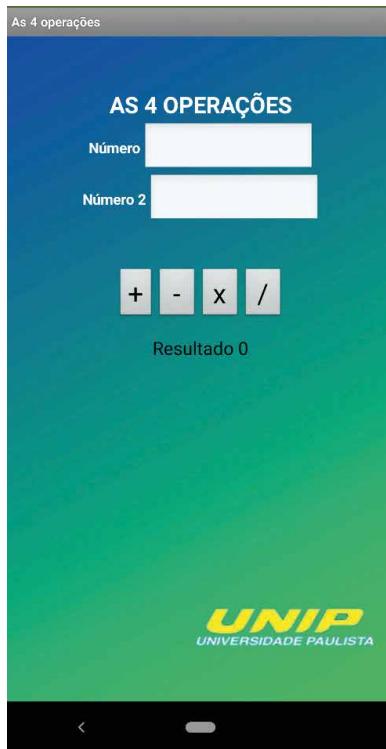


Figura 104 – Tela de abertura

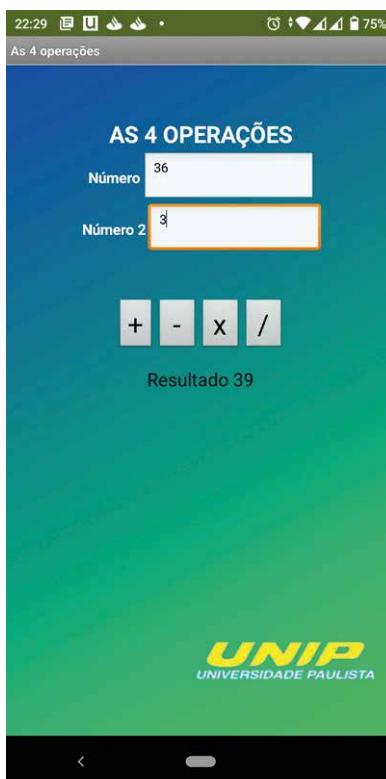


Figura 105 – Resultado da soma



## Resumo

Nesta unidade, começamos a estudar o App Inventor. Tivemos um breve contato com a história do desenvolvimento do produto e focalizamos a popularização da programação para dispositivos móveis.

Inicialmente, vimos a grande divisão do ambiente de desenvolvimento, o exterior, o design, a interface com o usuário e o editor de blocos, em que o comportamento do aplicativo é definido.

Um tópico foi dedicado ao Designer e ao uso dos organizadores, elementos flexíveis de layout que modelam a disposição dos objetos na tela do aplicativo. Seu uso é aconselhável para evitar problemas da diversidade de dimensões das telas dos dispositivos. Todos os componentes da Paleta foram referenciados. Por meio do módulo Designer, é possível editar e visualizar todos os componentes do aplicativo em desenvolvimento. Deve-se ressaltar novamente a importância de dar nomes intuitivos aos componentes, uma boa prática de programação que facilita a revisão do código pelo próprio criador e por outros colaboradores.

Outro tópico foi dedicado ao editor de blocos e à compreensão do funcionamento do App Inventor e do seu paradigma, a orientação a evento. Os blocos internos e os seus principais mecanismos foram apresentados. Os blocos são unidades de programação que possibilitam ao usuário criar variáveis, fazer operações matemáticas, editar textos, listas e cores, elaborar dicionários, configurar eventos e realizar operações lógicas.

Finalmente, fizemos um exercício de aplicação de programação com a criação de uma calculadora que realizar as quatro operações fundamentais da aritmética.



## Exercícios

**Questão 1.** Existem diversas formas de utilizarmos o MIT App Inventor para o desenvolvimento de aplicativos para celulares. Nesse contexto, avalie as afirmativas a seguir.

I – Em todas as possibilidades de desenvolvimento, são necessárias a instalação e a utilização de um emulador Android executado no celular.

II – A criação de projetos e de programas para celulares é feita por meio de um aplicativo que executa na nuvem, acessado diretamente dos navegadores de internet, como o Google Chrome ou o Firefox.

III – Caso se disponha de um celular com o sistema operacional Android, é possível instalar nele o aplicativo MIT AI2 Companion. Se esse dispositivo estiver conectado a uma rede sem fio, e se ela for a mesma rede na qual o computador utilizado para o desenvolvimento estiver conectado, é possível conectar o computador diretamente ao celular e testar o programa em desenvolvimento por meio da opção Assistente AI.

É correto o que se afirma em:

- A) I, apenas.
- B) II, apenas.
- C) III, apenas.
- D) I e III, apenas.
- E) II e III, apenas.

Resposta correta: alternativa E.

### Análise das afirmativas

I – Afirmativa incorreta.

Justificativa: a instalação do emulador não é sempre necessária, uma vez que podemos executar o programa diretamente no celular por meio do aplicativo MIT AI2 Companion.

II – Afirmativa correta.

Justificativa: uma das grandes vantagens do MIT App Inventor é que a sua interface de desenvolvimento é acessada diretamente pelo navegador, não sendo necessária a realização de uma instalação local. Com isso, amplia-se a gama de dispositivos que podem ser utilizados para o desenvolvimento de aplicativos e jogos.

III – Afirmativa correta.

Justificativa: uma das vantagens em se utilizar o MIT AI2 Companion é que é possível testar o programa diretamente no celular durante o seu desenvolvimento. Isso facilita a programação e possibilita o desenvolvimento em máquinas mais antigas e limitadas, que poderiam não ser capazes de executar um emulador com o Android.

**Questão 2.** Considere o programa feito no MIT App Inventor mostrado na figura a seguir.



Figura 106

Com base no programa e nos seus conhecimentos, avalie as afirmativas.

I – Quando o usuário clicar no Botão1, a propriedade Texto da CaixaDeTexto1 será modificada para o valor contido na propriedade Texto do componente Legenda4.

II – Quando o usuário clicar no Botão1, a propriedade Texto da CaixaDeTexto2 será modificada para o valor contido na propriedade Texto do componente Legenda4.

III – Quando o usuário clicar no Botão1, a propriedade Texto do componente Legenda4 será modificada para a soma dos valores obtidos das propriedades Texto dos componentes CaixaDeTexto1 e CaixaDeTexto2.

É correto o que se afirma em:

- A) I, apenas.
- B) II, apenas.
- C) III, apenas.
- D) II e III, apenas.
- E) I, II e III.

Resposta correta: alternativa C.

## Análise das afirmativas

I – Afirmativa incorreta.

Justificativa: a propriedade Texto da CaixaDeTexto1 não será modificada, apenas terá o seu valor lido para a realização do cálculo.

# PROGRAMAÇÃO PARA DISPOSITIVOS MÓVEIS

II – Afirmativa incorreta.

Justificativa: a propriedade Texto da CaixaDeTexto2 não será modificada, apenas terá o seu valor lido para a realização do cálculo.

III – Afirmativa correta.

Justificativa: os objetivos do pequeno programa são:

- ler os dois números que entram nas duas caixas de diálogo;
  - fazer a soma desses dois números;
  - colocar o resultado da soma na propriedade Texto do componente Legenda4.