UniFil
EDUCAÇÃO A DISTÂNCIA

Unidade 2

Desenvolvendo Aplicativos Móveis com MIT App Inventor

Introdução da Unidade

Neste módulo veremos como desenvolver aplicativos móveis usando o *MIT App Inventor*, originalmente criado pela *Google*, o *MIT App Inventor* é uma aplicação de código aberto, atualmente mantida pelo *Massachusetts Institute of Technology* (MIT). Ele permite que os recém-chegados à programação de computadores criem aplicativos de *software* para o sistema operacional *Android* através de interface gráfica simples, que permite arrastar e soltar objetos visuais para criar um aplicativo.

Apesar do *MIT App Inventor 2* (*MIT AI2*) ter como foco principal o ensino dos conceitos relacionados ao desenvolvimento de Aplicativos Móveis e introdução a lógica de programação pelo uso de elementos visuais é possível usar esta plataforma para construir MVPs (Produto viável mínimo), desta forma, permitindo a validação de uma ideia.

Saiba mais sobre **Produto viável mínimo**: Disponível em: https://pt.wikipedia.org/wiki/Produto_vi%C3%A1vel_m%C3%ADnimo

Objetivos

- Apresentar o *MIT App Inventor 2* e seus recursos;
- Entender como o *MIT AI2* pode ser usado para criar protótipos e aplicações;
- Construir uma aplicação real de Calculadora usando o *MIT AI2*.

Conteúdo programático

Aula 01 - Introdução e criação de uma aplicação básica

Aula 02 – Criação de uma aplicação de calculadora

Você poderá também assistir às videoaulas em seu celular! Basta apontar a câmera para os **QRCodes** distribuídos neste conteúdo.

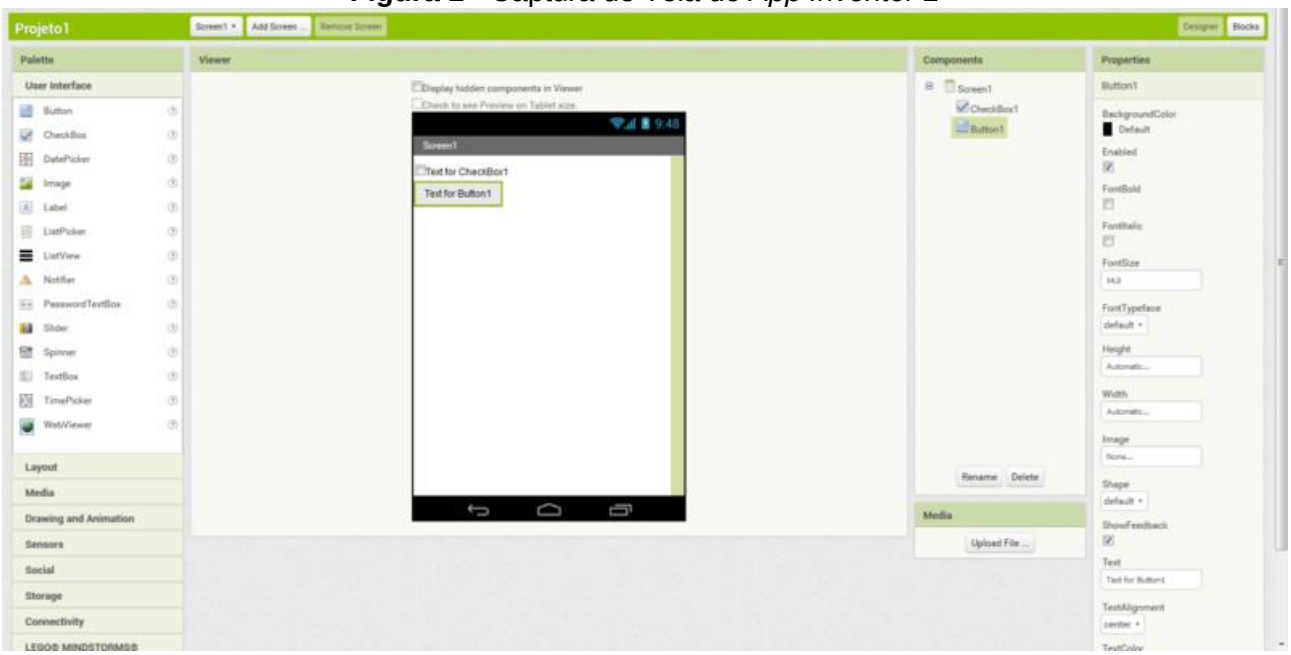
Pode ser necessário instalar um aplicativo de leitura QRcode no celular

Aula 01 – Introdução e criação de uma aplicação básica

O *MIT App Inventor*, é uma aplicação de código aberto originalmente criada pela equipe liderada por Hal Abelson e Mark Friedman do *Google,* em 2010. No segundo semestre de 2011, o *Google* disponibilizou o código fonte e forneceu o financiamento para a criação do *MIT Center for Mobile Learning* liderados pelo criador do *App Inventor* Hal Abelson e colegas professores do MIT, Eric Klopfer e Mitchel Resnick, sendo que a primeira versão MIT foi lançada em março de 2012.

Em 6 de dezembro de 2013, o *MIT App Inventor 2* foi lançado, sendo que o Editor de Blocos na versão original baseado em "*Blocks JAVA*" foi substituído pelo *Open Blocks,* que é distribuído pelo *Scheller Teacher Education Program* (STEP) e é derivado da pesquisa de mestrado de Ricarose Roque. Os professores Eric Klopfer e Daniel Wendel, do programa Scheller, apoiaram a distribuição de *BLOCKS* abertas no âmbito da licença MIT.

Figura 1 - Captura de Tela do *App Inventor 2*



Fonte: disponível em: https://pt.wikipedia.org/wiki/Ficheiro:App_Inventor_2.png. Acesso em: 15 mar. 2021.

O *App Inventor* é uma ferramenta baseada em nuvem, o que significa que você pode criar aplicativos direto no seu navegador da *web* acessando (http://ai2.appinventor.mit.edu), ou acessar (http://appinventor.mit.edu/) e clicar no botão laranja "**Create Apps**!"

Videoaula 1

Agora, assista à videoaula na qual veremos uma introdução ao *MIT App Inventor 2.*

Disponível em: https://www.youtube.com/watch?v=IwGQIA5kr9w. Acesso em: 15 mar. 2021.

Videoaula 1

Utilize o QRcode para assistir!

Agora, assista à videoaula na qual veremos uma introdução ao MIT App Inventor 2.

Você pode configurar o *App Inventor* e começar a criar aplicativos em minutos. O Designer e o Editor de Blocos são executados completamente no navegador. Para ver seu aplicativo em um dispositivo enquanto você o constrói ("*Live Testing*") existem quatro opções:

- Crie aplicativos com um *smartphone* ou *tablet* usando uma conexão Wi-Fi;
- Crie aplicativos com um *Chromebook*;
- Não tem um dispositivo? Use o emulador;
- Sem Wi-Fi? Crie aplicativos com um dispositivo *Android* e cabo USB.

1) Opção - RECOMENDADA - Crie aplicativos com um *smartphone* ou *tablet* usando uma conexão Wi-Fi

Se você tem um computador, *smartphone* ou *tablet* e uma conexão Wi-Fi, esta é a maneira mais fácil de construir e testar seus aplicativos no dispositivo.

Fonte: disponível em: https://img1.daumcdn.net/thumb/R720x0.q80/?scode=mtistory2&fname=http%3A%2F%2Fcfile5.uf.tistory.com%2Fimage%2F246E4E4657DD2015457F1E. Acesso em: 15 mar. 2021.

Você pode usar o *App Inventor* sem fazer *download* de nada para o seu computador!

Você desenvolverá aplicativos em nosso site: ai2.appinventor.mit.edu. Para fazer testes ao vivo em seu *smartphone* ou *tablet*, basta instalar o aplicativo *MIT App Inventor Companion* em seu telefone, *tablet* ou *Chromebooks* compatíveis. Depois que o *Companion* estiver instalado, você pode abrir projetos no *App Inventor* na *web* e pode testar seus aplicativos enquanto os constrói:

Etapa 1: baixe e instale o aplicativo *MIT AI2 Companion* em seu dispositivo.

Abra o leitor de código QR do seu dispositivo e leia o código localizado abaixo à esquerda para baixar o aplicativo *Companion* na *Play Store*. Se você não puder usar a *Play Store*, use o código QR à direita para baixar o aplicativo diretamente em seu telefone.

App Store	Play Store	Arquivo APK
Recomendado	***Recomendado***	**Atualizações manuais**
Leia este código QR (ou clique neste link)	**Leia este código QR (ou clique neste link)**	**Leia este código QR (ou clique neste link)**

Após o *download*, siga as instruções para instalar o aplicativo *Companion* em seu dispositivo. Você precisa instalar o *MIT AI2 Companion* apenas uma vez e, em seguida, deixá-lo em seu telefone ou *tablet* sempre que usar o *App Inventor*.

Observação 1: se você não conseguir usar o código QR, ainda pode instalar *o MIT AI2 Companion* em seu telefone ou *tablet*. Use o navegador da Web em seu dispositivo para acessar a *Google Play Store*; procure o *MIT AI2 Companion* na loja. Depois de encontrar, clique no botão INSTALAR para o aplicativo *Companion*.

Etapa 2: conecte o computador e o dispositivo à MESMA rede Wi-Fi

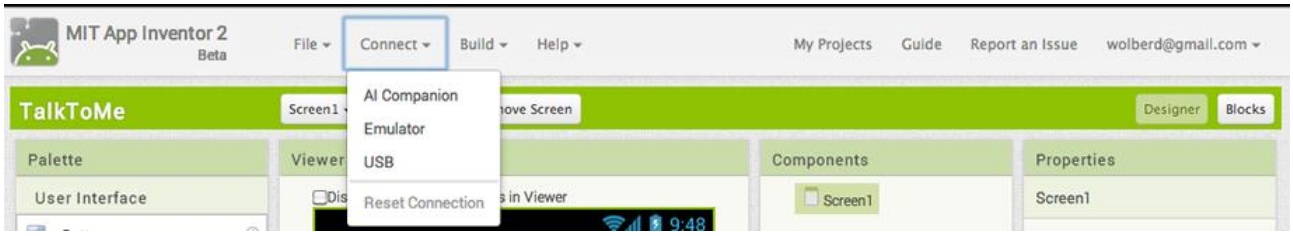
O *App Inventor* mostrará automaticamente o aplicativo que você está construindo, mas apenas se o seu computador (executando o *App Inventor*) e seu dispositivo (executando o *Companion*) estiverem conectados à **mesma** rede Wi-Fi.

Veja uma explicação mais detalhada sobre isso aqui: http://appinventor.mit.edu/explore/support/explain-wifi-connection.

Etapa 3: abra um projeto do *App Inventor* e conecte-o ao seu dispositivo

Vá para o *App Inventor* e abra um projeto (ou crie um novo - use Projeto> Iniciar novo projeto e dê um nome a ele).

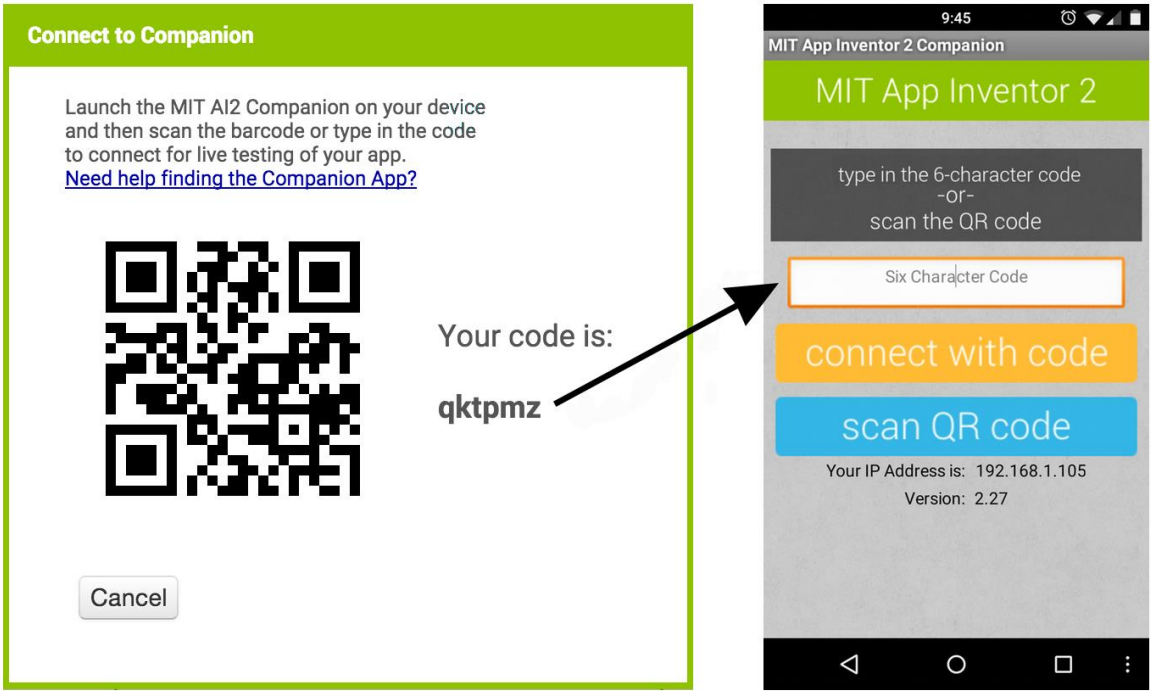
Em seguida, escolha "*Connect*" e "*AI Companion*" no menu superior do navegador AI2:



Fonte: MIT App Inventor 2

Uma caixa de diálogo com um código QR aparecerá na tela do seu PC. Em seu dispositivo, inicie o aplicativo *MIT App Companion* da mesma forma que você faria com

qualquer aplicativo. Em seguida, clique no botão “*Scan QR code*” no *Companion* e digitalize o código na janela do *App Inventor*:



Fonte: MIT App Inventor 2 Companion

Em alguns segundos, você verá o aplicativo que está construindo no seu dispositivo. Ele será atualizado conforme você fizer alterações em seu design e blocos, um recurso chamado “teste ao vivo”.

Se você tiver problemas para escanear o código QR ou se seu dispositivo não tiver um *scanner*, digite o código mostrado no computador na área de texto do *Companion* em seu dispositivo, exatamente como mostrado. O código está diretamente abaixo de onde a tela do seu PC mostra “Seu código é” e consiste em seis caracteres. Digite-os e escolha o botão laranja “Conectar com código”. **Não digite Enter: digite apenas os seis caracteres seguidos pressionando o botão laranja**.

Solução de problemas

Se seu aplicativo não aparecer em seu dispositivo, os problemas mais prováveis são:

Você pode ter uma versão desatualizada do *App Inventor Companion*. Baixe o aplicativo *Companion* mais recente para o *App Inventor 2*, acima.

Seu dispositivo pode não estar conectado ao wi-fi. Certifique-se de ver um endereço IP na parte inferior da tela do *AI Companion App* em seu telefone ou *tablet*.

Seu dispositivo pode não estar conectado à mesma rede wi-fi que seu computador. Certifique-se de que ambos os dispositivos estejam conectados ao mesmo nome de rede wi-fi.

Sua escola ou organização pode ter protocolos de rede em vigor de forma que a conexão wi-fi não seja permitida. Nesse caso, você ainda pode usar o *App Inventor* com o emulador ou pode usar um cabo USB para conectar seu dispositivo. Volte para a página de configuração principal dessas opções.

Para as demais opções de execução, abaixo seguem os respectivos *links* da documentação:

- Crie aplicativos com um Chromebook: http://appinventor.mit.edu/support/chromebooks;
- Não tem um dispositivo? Use o emulador: http://appinventor.mit.edu/explore/ai2/setup-emulator;
- Sem wi-fi? Crie aplicativos com um dispositivo *Android* e cabo USB: http://appinventor.mit.edu/explore/ai2/setup-device-usb

Videoaula 2

Agora, assista à videoaula na qual veremos como criar uma primeira aplicação bem simples no *MIT App Inventor 2*. Na sequência, em nosso material, temos um projeto de exemplo *HelloCodi*, que faz parte da documentação oficial do MIT AI2.

Disponível em: https://www.youtube.com/watch?v=8jPO8UzumGQ. Acesso em: 15 mar. 2021.

Videoaula 2

Utilize o QRcode para assistir!

Agora, assista à videoaula na qual veremos como criar uma primeira aplicação bem simples no MIT App Inventor 2. Na sequência, em nosso material, temos um projeto de exemplo HelloCodi, que faz parte da documentação oficial do MIT AI2.

Construindo seu primeiro aplicativo

Agora que você configurou seu computador e dispositivo, e aprendeu como o Designer e o Editor de Blocos funcionam, você está pronto para construir o aplicativo *HelloCodi*. Neste ponto, você deve ter o Designer ou Editor de Blocos aberto em seu navegador e um dispositivo *Android* ou emulador *Android* conectado ao Editor de Blocos. (Consulte as instruções de configuração do *App Inventor 2* caso você não tenha essas informações em execução). Escolha “Iniciar um novo projeto” no menu “Projetos” e denomine *HelloCodi*.

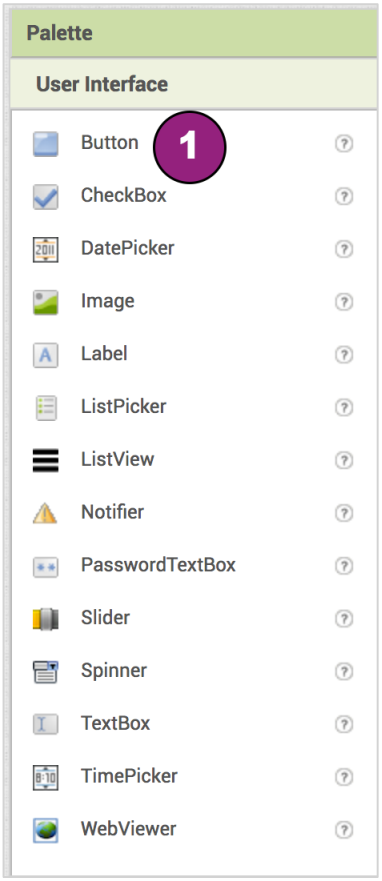
HelloCodi: toque na abelha, ouça o seu zumbido!

HelloCodi é um aplicativo simples que você pode construir em um tempo muito curto. Você deve criar um botão com uma imagem da abelha Codi e, em seguida, programar o botão para que, quando for clicado, o telefone emita um zumbido.

Para construir o *HelloCodi*, você precisará de um arquivo de imagem da abelha Codi. Baixe esses arquivos para o seu computador clicando nos *links* a seguir. Para fazer o *download*: após clicar em um *link*, clique com o botão direito na imagem e selecione "Salvar como". Salve o arquivo em sua área de trabalho ou na pasta de *downloads*, ou em qualquer lugar onde possa encontrá-lo facilmente mais tarde.

- Imagem codi: codi.jpg (clique com o botão direito e salve);
- Som de abelha: Bee-Sound.mp3 (clique com o botão direito e salve).

Selecione os componentes para projetar seu aplicativo:



Fonte: MIT App Inventor 2

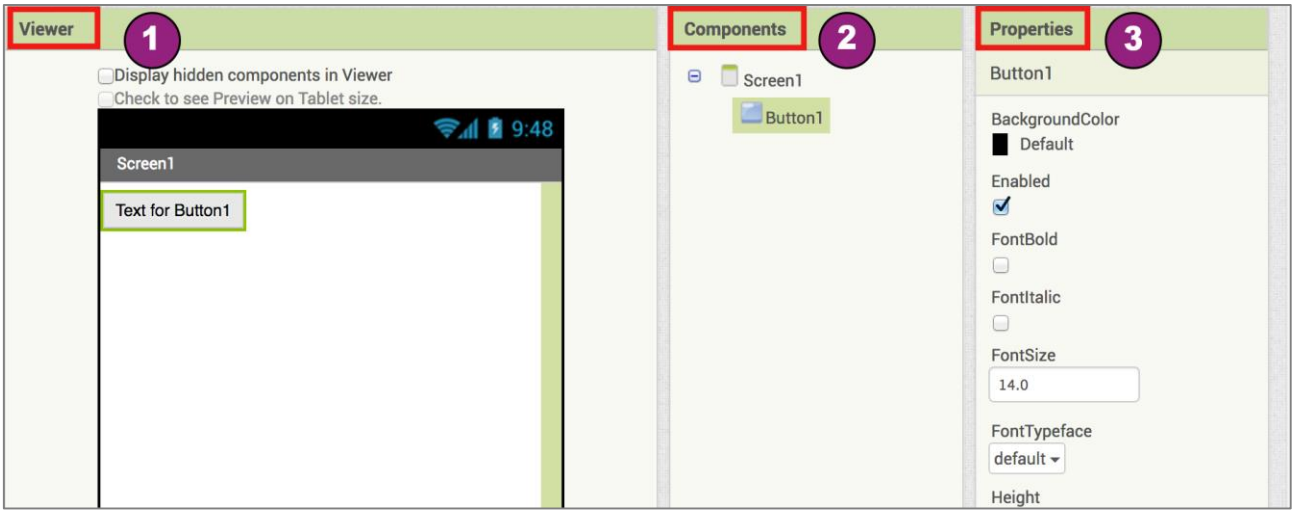
Os componentes do *App Inventor* estão localizados no lado esquerdo da janela Designer denominado “Paleta”. Componentes são os elementos básicos que você utiliza para criar aplicativos no telefone *Android*. Eles são como os ingredientes de uma receita. Alguns componentes são muito simples, como um componente Label, que apenas mostra o texto na tela, ou um componente Button (nº 1 à esquerda) que você toca para iniciar uma ação.

Outros componentes são mais elaborados: uma tela de desenho que pode conter imagens estáticas ou animações, um sensor de acelerômetro que funciona como um controlador de Wii e detecta quando você move ou sacode o telefone, componentes que enviam mensagens de texto, componentes que tocam música e vídeo, componentes que obtêm informações de sites da Web e assim por diante.

Para usar um componente em seu aplicativo, você precisa clicar e arrastá-lo para o visualizador no meio do Designer. Ao adicionar um componente ao *Viewer* (nº 1, abaixo), ele também aparecerá na lista de componentes à direita do *Viewer*.

Os componentes (nº 2, abaixo) têm propriedades ajustáveis. Essas propriedades alteram a forma como o componente aparece ou se comporta no aplicativo. Para visualizar e alterar as propriedades de um componente (n° 3, abaixo), você deve primeiro selecionar o componente desejado em sua lista de componentes.

Figura 2 - Etapas para selecionar componentes e definir propriedades



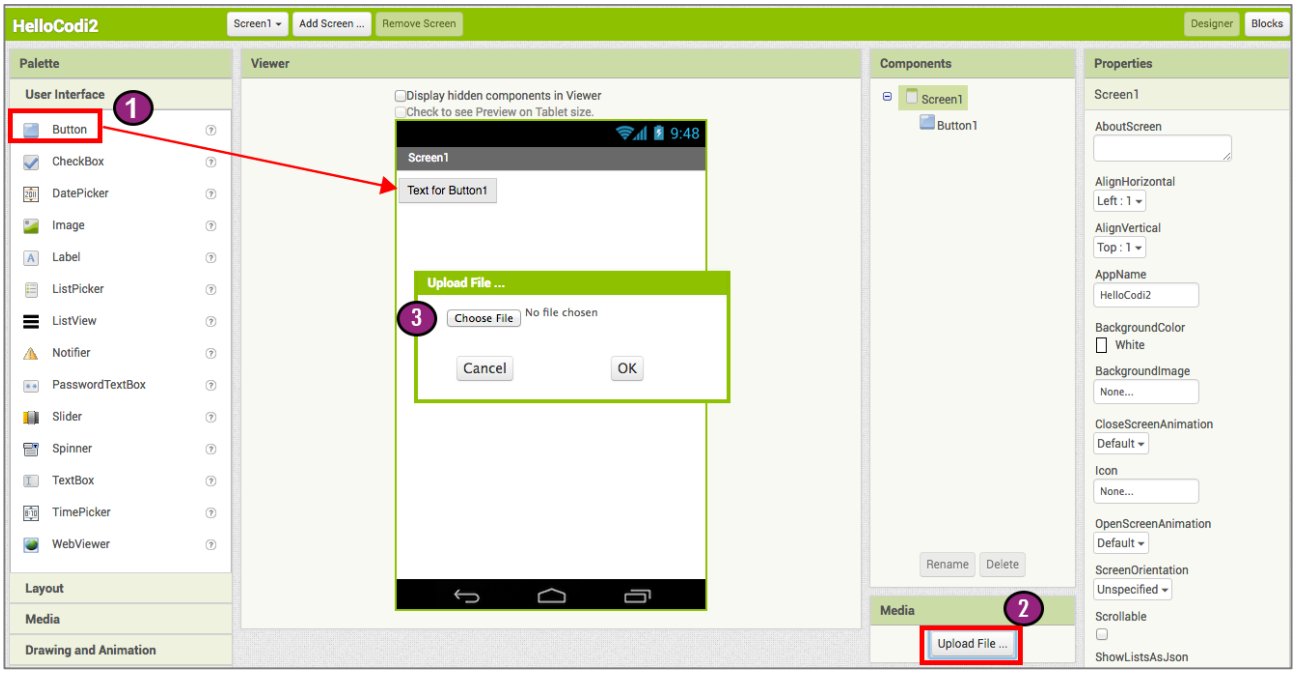
Fonte: MIT App Inventor 2

HelloCodi terá um componente de botão que exibe a imagem da abelha que você baixou anteriormente. Para conseguir isso:

Passo 1a. Na paleta Interface do usuário, arraste e solte o componente Botão na Screen1 (#1).

Passo 1b. Para dar ao botão a imagem da abelha, no painel Propriedades, em Imagem, clique no texto "Nenhum ..." e clique em "Carregar Arquivo ..." (#2). Uma janela aparecerá para permitir que você escolha o arquivo de imagem. Clique em "Procurar" e

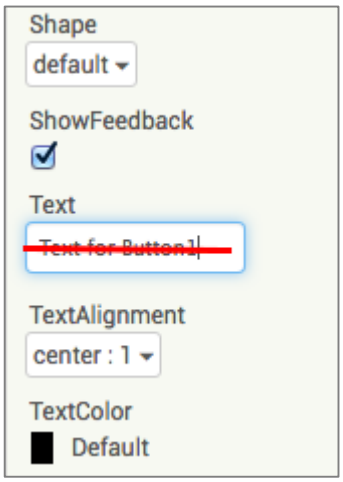
navegue até o local do arquivo codi.jpg que você baixou anteriormente (#3). Clique no arquivo codi.jpg, clique em "Abrir" e em "OK".



Fonte: MIT App Inventor 2

Passo 2. Altere a propriedade Texto do botão:

Exclua "Texto para Botão1", deixando a propriedade texto do Botão em branco a fim de que não haja escrita sobre a imagem da abelha.



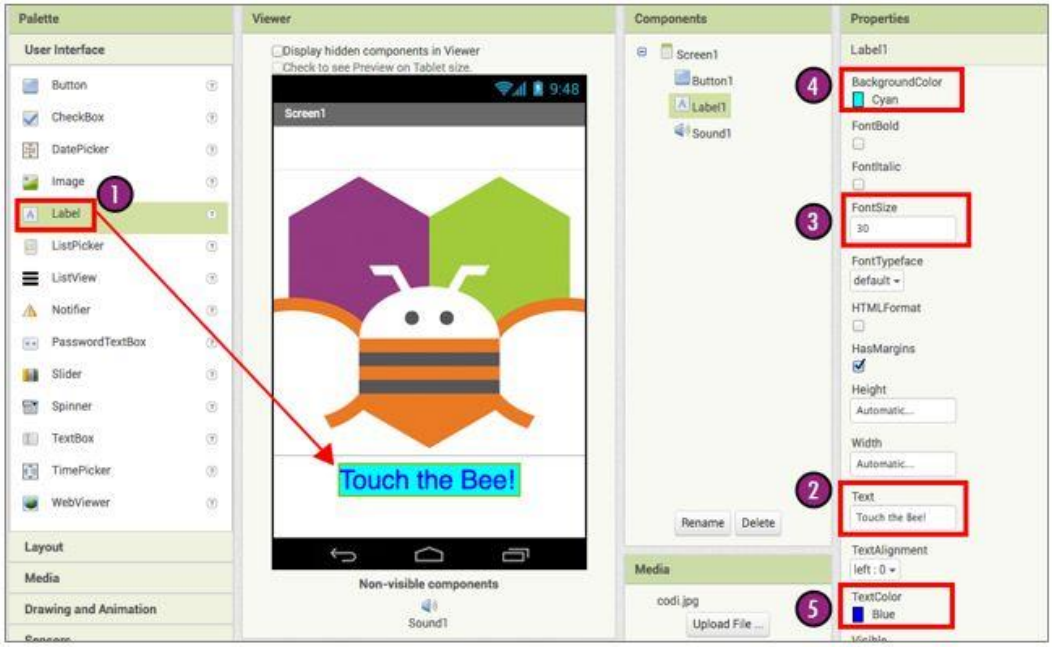
Fonte: MIT App Inventor 2

Passo 3. Na paleta Interface do usuário, arraste e solte o componente Rótulo no Visualizador (# 1), colocando-o abaixo da imagem da abelha. Ele aparecerá na sua lista de componentes como Label1.

No painel “Propriedades”, altere o.

- (2) Propriedade de texto de Label1 para ler "Toque na abelha". Você verá a mudança de texto no Designer e no seu dispositivo;
- (3) FontSize - 30;
- (4) BackgroundColor de Label1 clicando na caixa. Você pode alterá-lo para qualquer cor que desejar;
- (5) TextColor para qualquer cor que você quiser. (Observação: se BackgroundColor e TextColor forem iguais, você não poderá ler seu texto!).

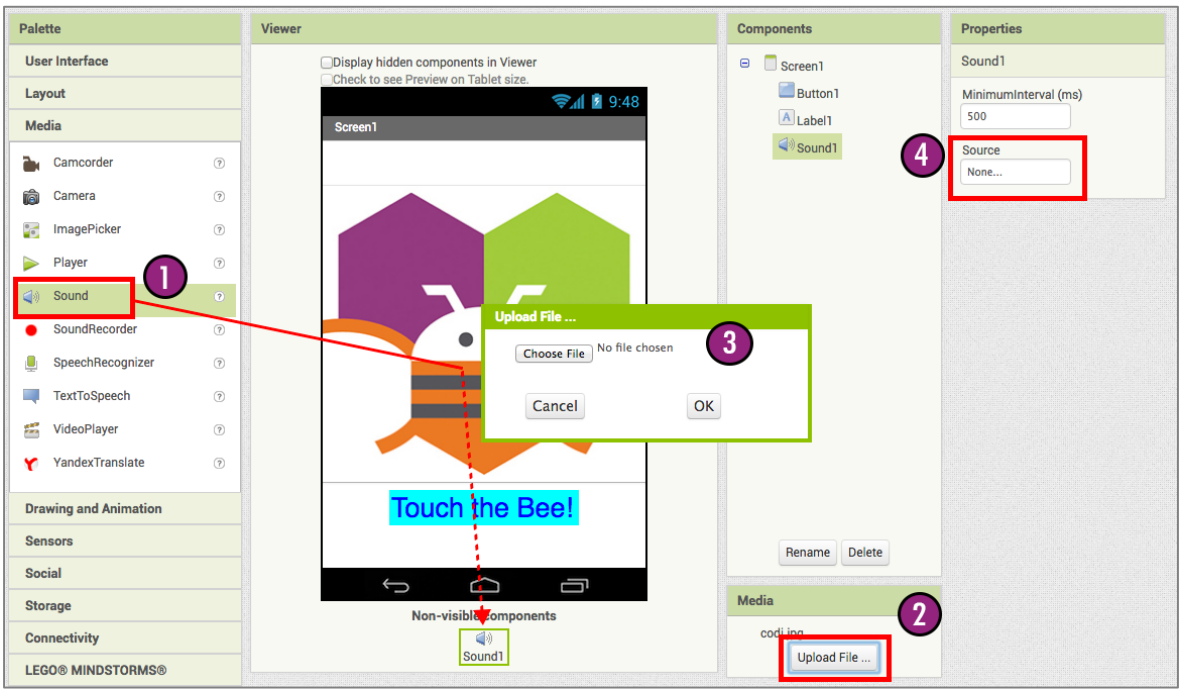
Aqui, a cor de fundo é definida como água e a cor do texto é definida como azul.



Fonte: MIT App Inventor 2

Passo 4. Em “Paleta”, clique na gaveta de mídia e arraste um componente de som e coloque-o no visualizador (#1). Onde quer que você o solte, ele aparecerá na área inferior do Visualizador marcada como “Componentes não visíveis”. No painel “Media”, clique em “*Upload File”* (#2); navegue até o local do arquivo Bee-Sound.mp3 que você baixou anteriormente e carregue-o neste projeto (#3). No painel “Propriedades”, veja que a

propriedade Fonte atualmente diz “Nenhum”; Clique na palavra “Nenhum” para alterar a Fonte do componente Sound1 para Bee-Sound.mp3 (#4).



Fonte: MIT App Inventor 2

Programação com o Editor de Blocos

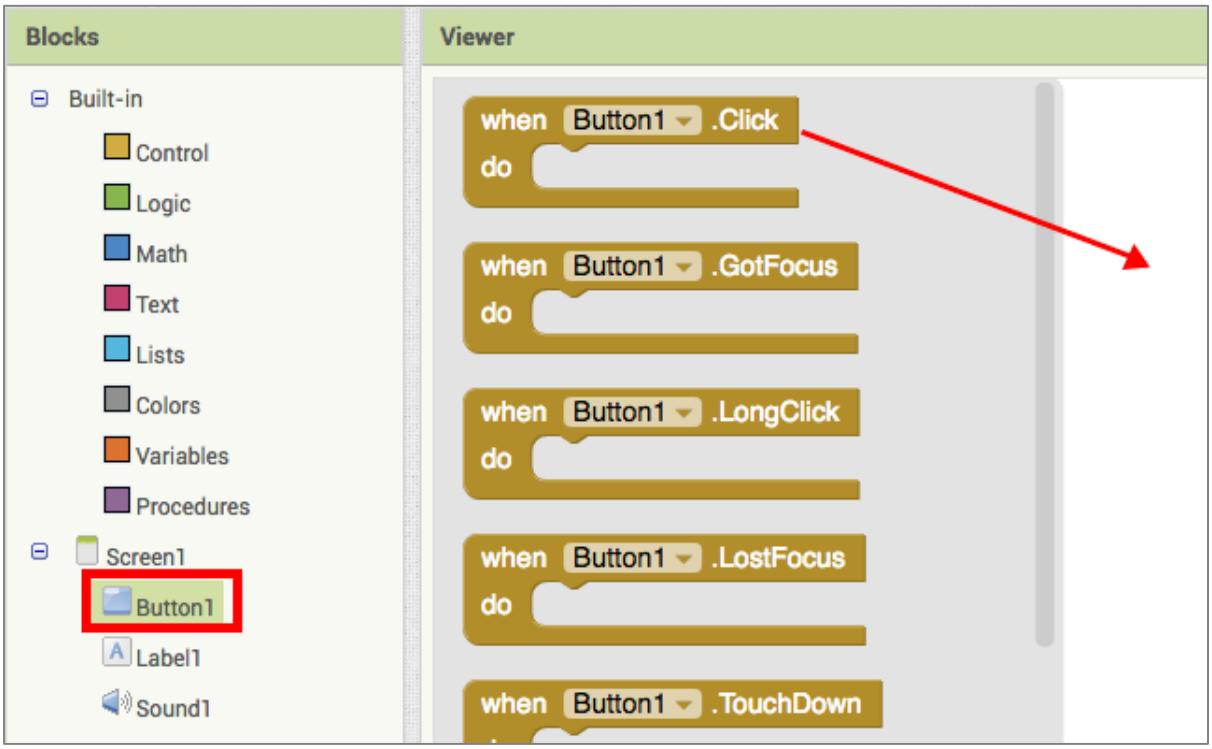
Até agora, você organizou a tela e os componentes do seu aplicativo no Designer, que está em uma janela do navegador da *web*. Para começar a programar o comportamento do aplicativo, você precisa ir ao Editor de Blocos. Clique no botão Blocos, no canto superior direito da tela, para ir para o Editor de blocos.

Fonte: MIT App Inventor 2

Assim que tiver o Editor de blocos a sua frente, passe para a próxima etapa para começar a programar seu aplicativo com blocos.

Tocando o som

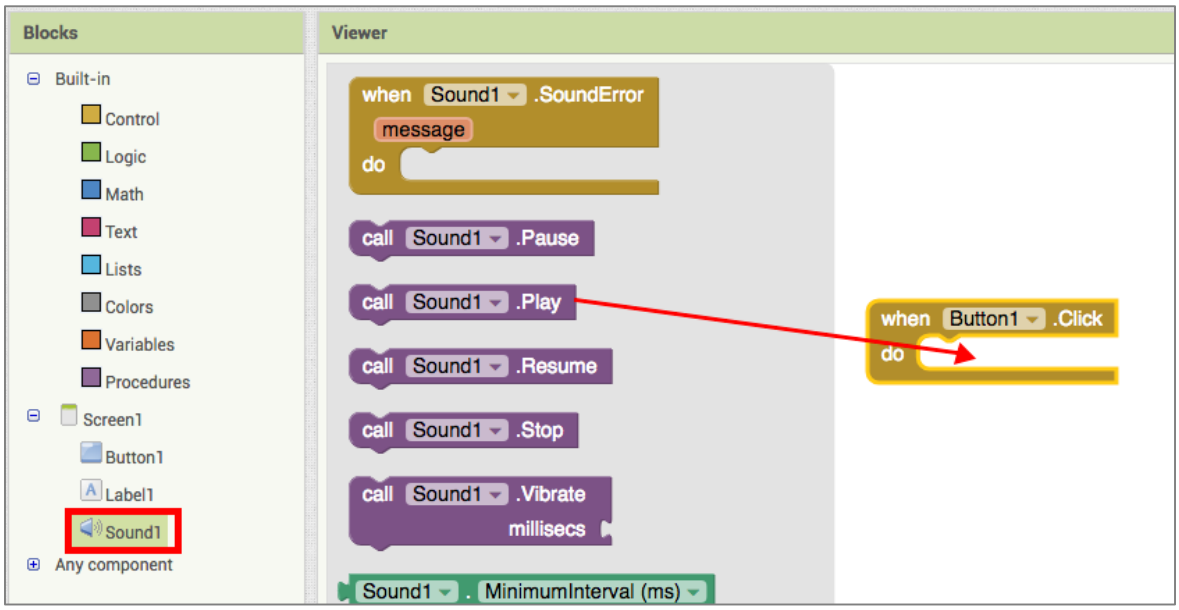
Passo 1. No lado esquerdo do Editor de blocos, clique na gaveta “Button1” para abri-la. Arraste e solte o when Button1.Click, bloco na área de trabalho (a área aberta à direita).



Fonte: MIT App Inventor 2

Esses blocos amarelos mostarda são chamados de blocos manipuladores de eventos. Os blocos do manipulador de eventos especificam como o dispositivo móvel deve responder a certos eventos: um botão foi pressionado, o telefone está sendo sacudido, o usuário está arrastando o dedo sobre uma tela etc. when Button1.Click é um manipulador de eventos.

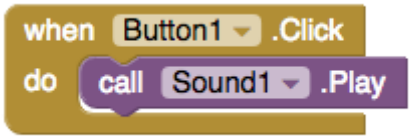
Passo 2a. Clique na “Sound1” e arraste o Sound1.Play para bloquear e conectá-lo à seção "fazer" do bloco when Button1.Click. Os blocos se conectam como peças de um quebra-cabeça e você pode ouvir um som de clique quando eles se conectam.



Fonte: MIT App Inventor 2

Os blocos roxos são chamados de blocos de comando, que são colocados no corpo dos manipuladores de eventos. Quando um manipulador de eventos é executado, ele executa a sequência de comandos em seu corpo. Um comando é um bloco que especifica uma ação a ser realizada (por exemplo, tocar som) quando o evento (por exemplo, pressionar o Button1) é acionado.

Seus blocos devem ter a seguinte aparência neste ponto:



Agora você pode ver que o bloco de comando está no manipulador de eventos. Este conjunto de blocos significa; "quando o botão1 é clicado, o som1 é reproduzido." O manipulador de eventos é como uma categoria de ação (por exemplo, um botão está sendo clicado), e o comando especifica o tipo de ação e os detalhes da ação (por exemplo, reproduzir um som).

Você pode ler mais sobre os blocos e como eles funcionam aqui: App Inventor Built-in Blocks. Disponível em: http://ai2.appinventor.mit.edu/reference/blocks/.

Experimente no seu dispositivo ou com o emulador! Ao pressionar o botão, você deve ouvir o zumbido das abelhas. Parabéns, seu primeiro aplicativo está em execução!

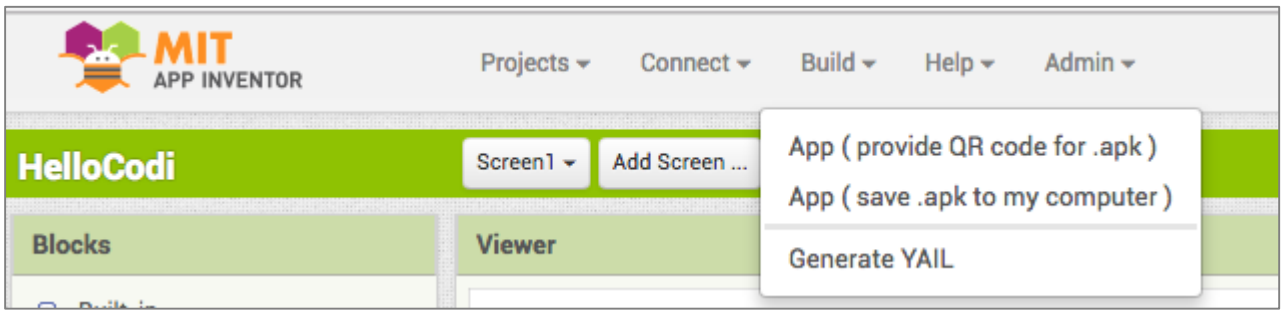
Nota

Há um problema conhecido com o componente de som em alguns dispositivos. Caso veja um "Erro do sistema operacional" e o som não for reproduzido - ou a reprodução estiver muito atrasada, volte ao Designer e tente usar um componente “Player” (encontrado em Mídia) em vez do componente Som.

Empacotando seu aplicativo

Enquanto seu dispositivo (emulador ou telefone/tablet) estava conectado ao *App Inventor*, seu aplicativo estava sendo executado em tempo real no seu dispositivo. Se você desconectar o emulador/dispositivo do Editor de Blocos, o aplicativo irá desaparecer. Você sempre pode fazer com que ele retorne reconectando o dispositivo. Para ter um aplicativo em execução sem estar conectado ao *App Inventor*, você deve "empacotar" o aplicativo para produzir um pacote de aplicativo (arquivo apk).

Para "empacotar" o aplicativo a fim de instalar em seu dispositivo ou enviar para outra pessoa, clique na guia Construir na parte superior da tela. Em Build, há duas opções disponíveis para você escolher:



Fonte: MIT App Inventor 2

1. App (provide QR code for.apk): você pode gerar um código de barras (um código QR), que pode ser usado para instalar o aplicativo em um dispositivo móvel que

possua uma câmera, com o auxílio de um leitor de código de barras (como o leitor de código de barras ZXing, disponível gratuitamente no Google Play).

Nota

Este código de barras só é válido por duas horas. Se você deseja compartilhar seu aplicativo com outras pessoas via código de barras por um período mais longo, você precisará baixar o arquivo.apk para o seu computador e usar um *software* de terceiros para converter o arquivo em um código de barras. Mais informações podem ser encontradas em (http://appinventor.mit.edu/explore/ai2/share)

2. App (save .apk to my computer): você pode baixar o aplicativo para o seu computador como um arquivo apk, que você pode distribuir e compartilhar como quiser instalando-o manualmente em outros dispositivos. (veja mais em: http://www.techrepublic.com/blog/smartphones/how-to-side-load-apps-on-your-android-device/3114).

Análise

Aqui estão as principais ideias abordadas até agora:

- Você cria aplicativos selecionando componentes (ingredientes) e dizendo a eles o que fazer e quando fazer;
- Você usa o Designer para selecionar componentes e definir as propriedades de cada um deles. Alguns componentes são visíveis e outros não;
- Você pode adicionar mídia (sons e imagens) aos aplicativos fazendo o *upload* do seu computador;
- Você usa o Editor de Blocos para montar blocos que definem o comportamento dos componentes;
- when ... do ... são blocos que definem os manipuladores de eventos e que dizem aos componentes o que fazer quando algo acontece;
- call ... são blocos que dizem aos componentes para fazer coisas.

Digitalize o aplicativo de amostra para o seu telefone

Digitalize o seguinte código de barras em seu telefone para instalar e executar o aplicativo de amostra ou baixe o apk.

Baixe o código fonte

Se você gostaria de trabalhar com este exemplo no *App Inventor*, baixe o código-fonte para o seu computador, abra o *App Inventor*, clique em Projetos, escolha Importar projeto (.aia) do meu computador e selecione o código-fonte que você acabou de baixar.

Aula 02 – Criação de uma aplicação de calculadora

Agora nosso foco é inteiramente prático, veremos como criar um aplicativo de Calculadora simples no *MIT App Inventor* e executar no celular. Vamos usar diversos recursos de lógica de programação para implementar o funcionamento de nossa calculadora. Para a construção do nosso projeto temos 4 videoaulas mostrando o passo a passo da construção.

Videoaula 1

Neste primeiro vídeo veremos como criar o projeto e construir a interface de nossa calculadora com base no protótipo que elaboramos.

Disponível em: https://www.youtube.com/watch?v=M3EuNLmwgcc. Acesso em: 15 mar. 2021.

Videoaula 1

Utilize o QRcode para assistir!

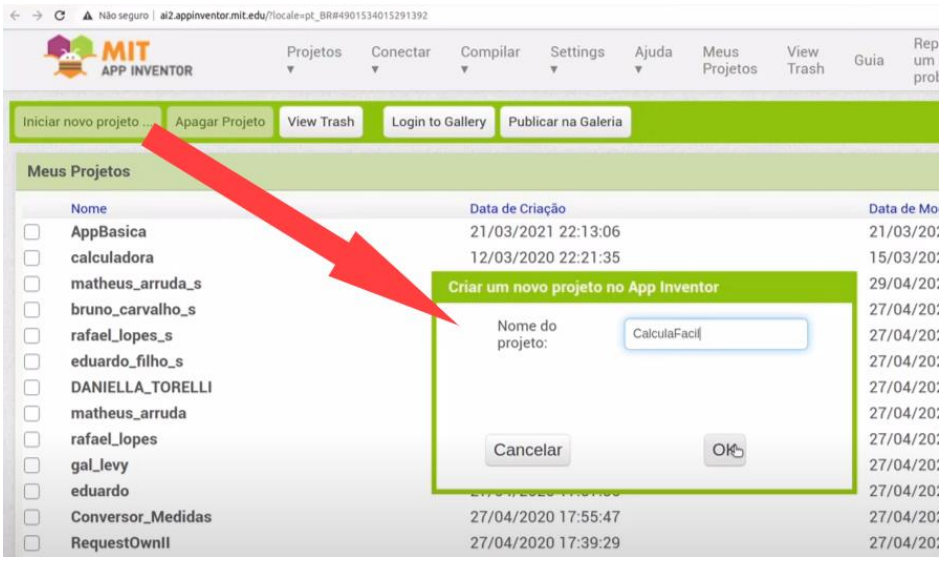
Neste primeiro vídeo veremos como criar o projeto e construir a interface de nossa calculadora com base no protótipo que elaboramos.

Para construir nossa calculadora vamos nos basear no protótipo elaborado usando o Balsamiq. Será uma calculadora simples, mas completamente funcional.

Fonte: MIT App Inventor 2

Nossa calculadora terá um local para apresentar o valor no topo da tela, um quadro com 4 linhas e quatro colunas onde estão os botões numéricos e botões de operações além de um botão para limpar.

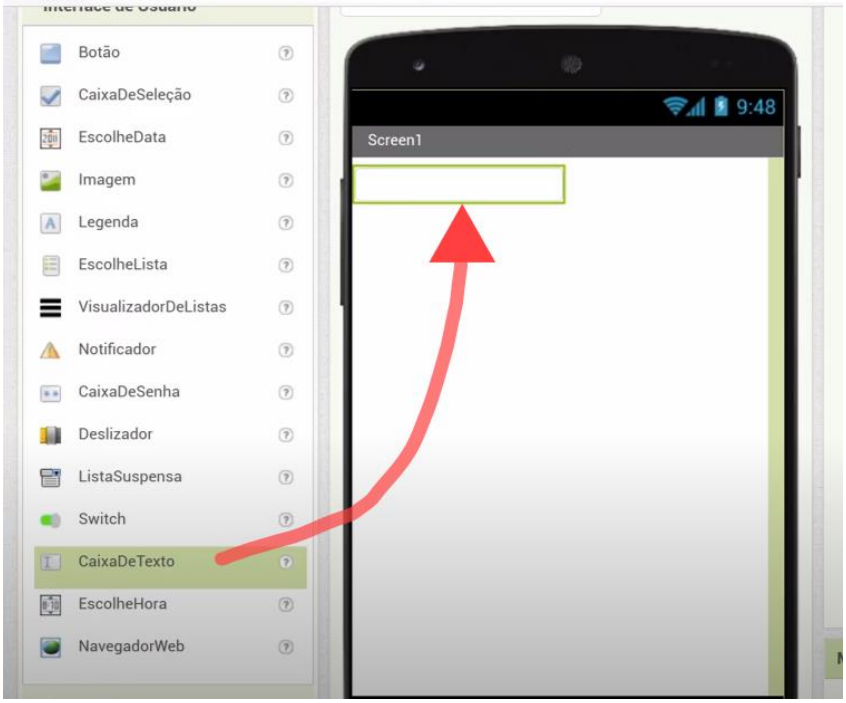
Para iniciar, criamos um novo projeto e chamamos de *CalculaFacil*. Lembre-se que para acessar vá em (https://appinventor.mit.edu/) e clique em (Create Apps!).



Fonte: MIT App Inventor 2

Para agora podemos montar a tela principal da nossa calculadora. O primeiro elemento que vamos usar é a “**CaixaDeTexto**”. Basta clicar e arrastar para a tela. Depois de arrastar podemos ajustar algumas propriedades como:

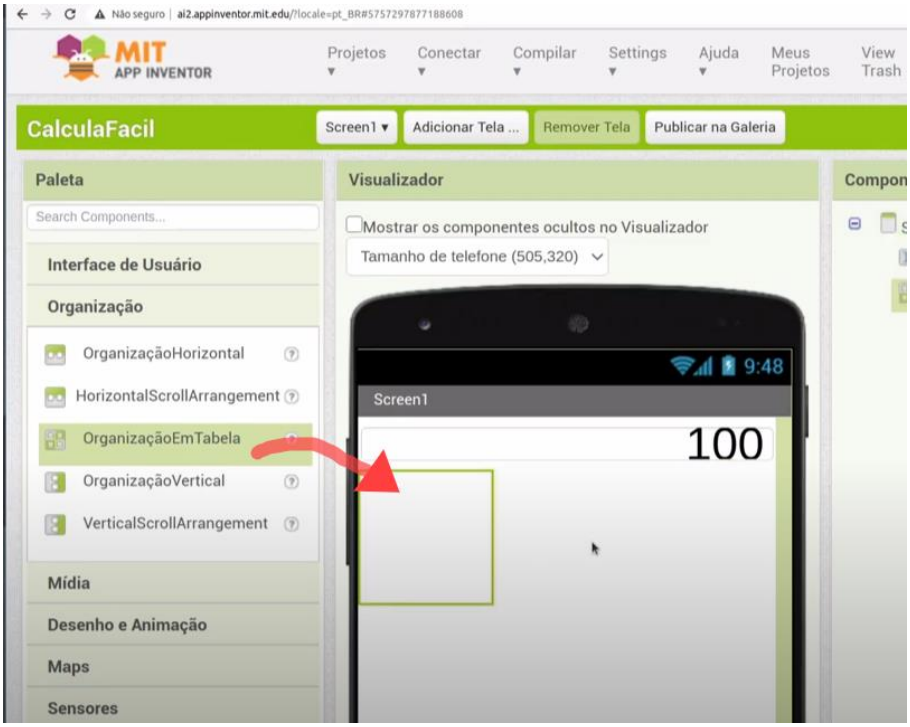
- TamanhoDaFonte: 24;
- Largura: Preencher principal;
- ReadOnly: (deixar a caixa selecionada);
- Texto: 0 (zero).



Fonte: MIT App Inventor 2

Depois de incluir a caixa de texto, vamos adicionar um elemento de organização de *layout*, para conseguir o efeito de quatro linhas e quatro colunas distribuídas proporcionalmente. Vamos usar o elemento “**OrganizaçãoEmTabela**”. Basta arrastar o elemento para a tela, depois da “CaixaDeTexto” que já foi incluída. Depois de incluir o elemento, vamos ajustar as seguintes propriedades dele:

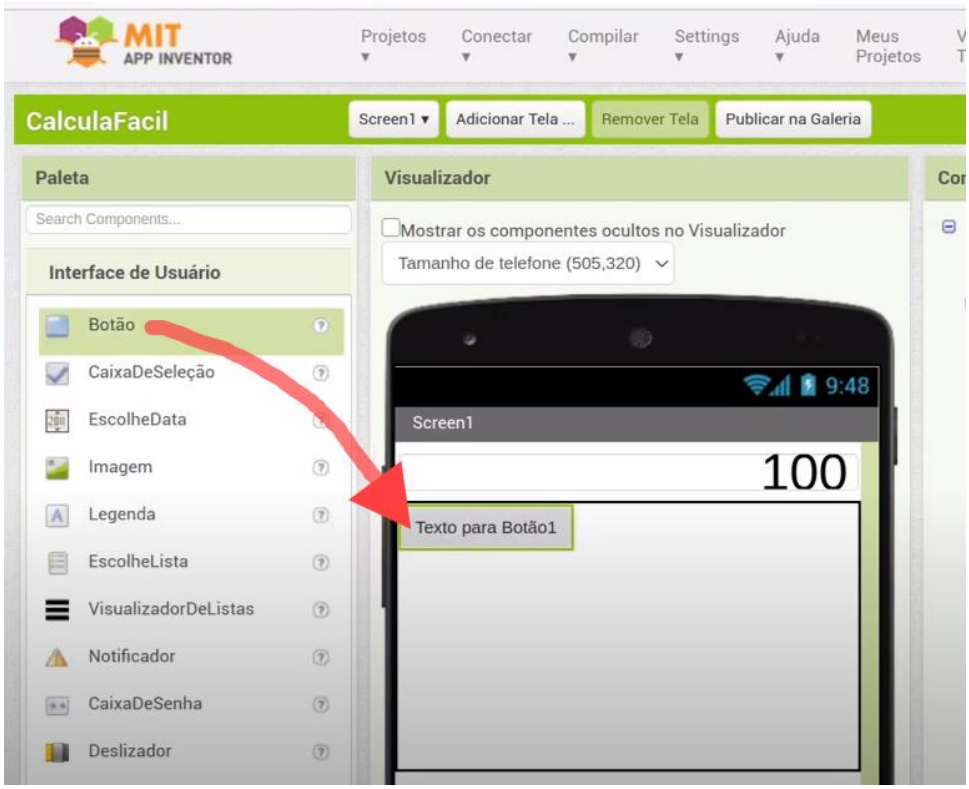
- Colunas: 4;
- Largura: Preencher principal;
- Linhas: 4.



Fonte: MIT App Inventor 2

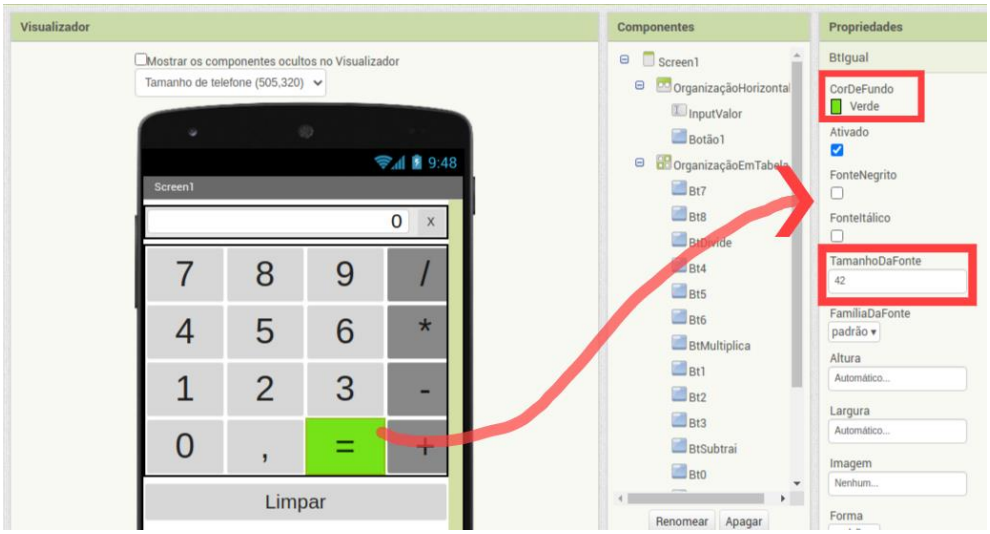
Com o elemento de organização incluído da tela, podemos começar a incluir os botões. Basta arrastar o elemento "**Botão**" para dentro do elemento "**OrganizaçãoEmTabela**". Cada botão inserido deve ter suas propriedades ajustadas para:

- TamanhoDaFonte: 42;
- Largura: 25 percentual;
- Texto: 7 (modificar para cada elemento).



Fonte: MIT App Inventor 2

Podemos melhorar alguns aspectos visuais dos componentes usando cores, fontes diferentes entre outros, no nosso exemplo, os botões de cálculo ficaram com um tom de cinza mais escuro, isso foi feito alterando a propriedade "**CorDeFundo**" para "**Cinza**". O Botão de **"="** (igual) teve a propriedade "**CorDeFundo**" alterada para "**Verde**".



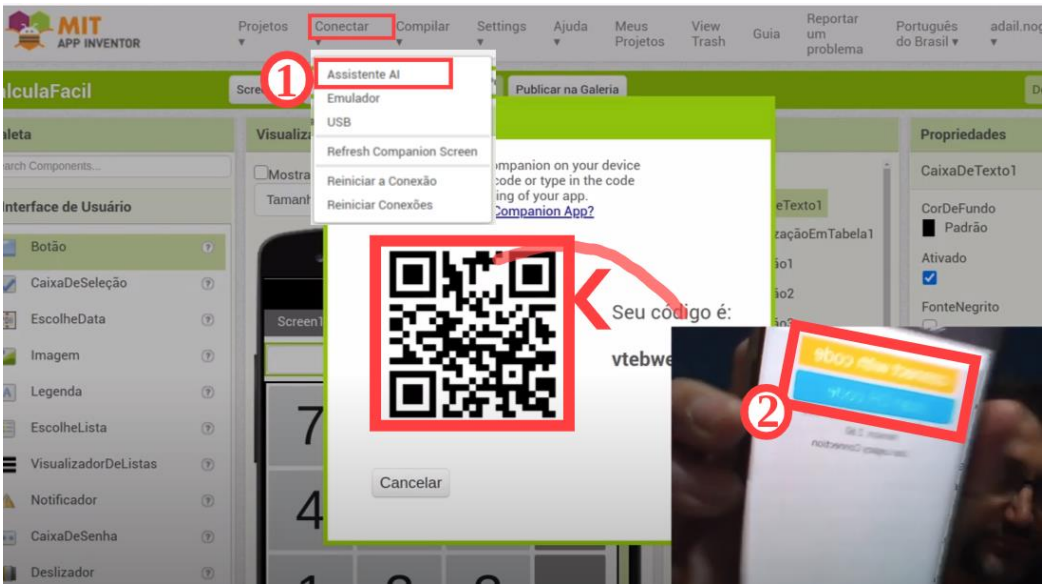
Fonte: MIT App Inventor 2

Agora que a tela de nossa aplicação “**CalculaFacil**” está pronta, podemos executar o projeto em nosso celular *Android*. Para isso, em primeiro lugar é necessário ter o aplicativo MIT AI2 Companion instalado no celular.

Então acessamos o menu “**Conectar ▾**” e selecionamos a opção “**Assistente AI**”, será aberta a tela “**Conectar ao Assistente**” e apresentado um QR Code e um código.

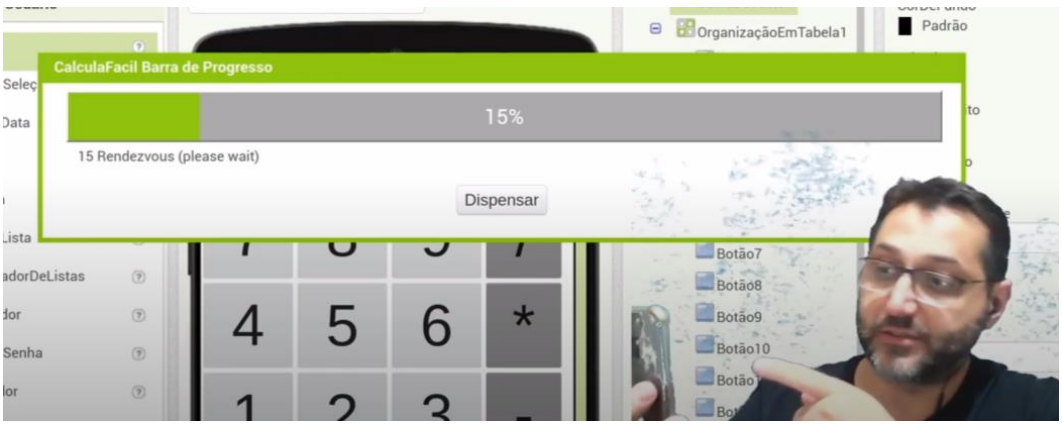
No celular, abra o *MIT AI2 Companion*, e escolha uma das opções:

- connect with code - Nesta opção você deve digitar o código apresentado na tela;
- scan QR code - Nesta opção basta apontar a câmera do celular para o QR Code.



Fonte: MIT App Inventor 2

Agora é só aguardar a barra de progresso terminar e ficar de olho no celular. Quando terminar a aplicação vai abrir automaticamente no celular.



E olha que legal, agora que a aplicação está rodando no celular, você pode alterar a aplicação que ela é atualizada automaticamente.

Ops! Estava funcionando, mas desconectou!

Se você conseguiu rodar sua aplicação normalmente no celular, mas depois de algum tempo ela perdeu a conexão (parou de funcionar) ou parou de refletir as alterações que você está fazendo no projeto não se preocupe, isso ocorre com alguma frequência, para consertar basta acessar "**Conectar** ▾" e "**Reiniciar Conexões**", então feche o aplicativo no seu celular. Agora pode realizar novamente o procedimento para conectar: "**Conectar** ▾" e selecionamos a opção "**Assistente AI**", será aberta a tela "Conectar ao Assistente" etc.

Se seu aplicativo não aparecer em seu dispositivo, os problemas mais prováveis são:

- Você pode ter uma versão desatualizada do *App Inventor Companion*. Baixe o aplicativo *Companion* mais recente, veja mais detalhes em: (https://appinventor.mit.edu/explore/ai2/setup-device-wifi).
- Seu dispositivo pode não estar conectado ao wi-fi. Certifique-se de ver um endereço IP na parte inferior da tela do *AI Companion App* em seu telefone ou *tablet*.
- Seu dispositivo pode não estar conectado à **mesma rede wi-fi** que seu computador. Certifique-se de que ambos os dispositivos estejam conectados ao mesmo nome de rede wi-fi.
- Sua escola ou organização pode ter protocolos de rede em vigor de forma que a conexão wi-fi não seja permitida. Nesse caso, você ainda pode usar o *App Inventor* com o emulador ou pode usar um cabo USB para conectar seu dispositivo. Volte para a página de configuração principal e veja essas opções.

Videoaula 2

Agora, assista à videoaula na qual daremos continuidade com a introdução ao *MIT App Inventor 2*.

Disponível em: https://www.youtube.com/watch?v=Sa8QXg_wOmQ. Acesso em: 15 mar. 2021.

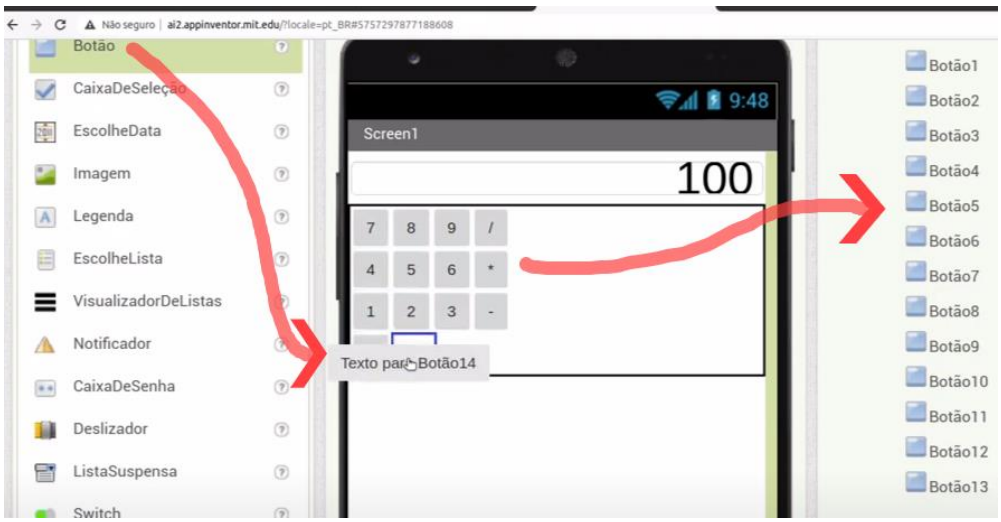
Videoaula 2

Utilize o QRcode para assistir!

Agora, assista à videoaula na qual daremos continuidade com a introdução ao MIT App Inventor 2.

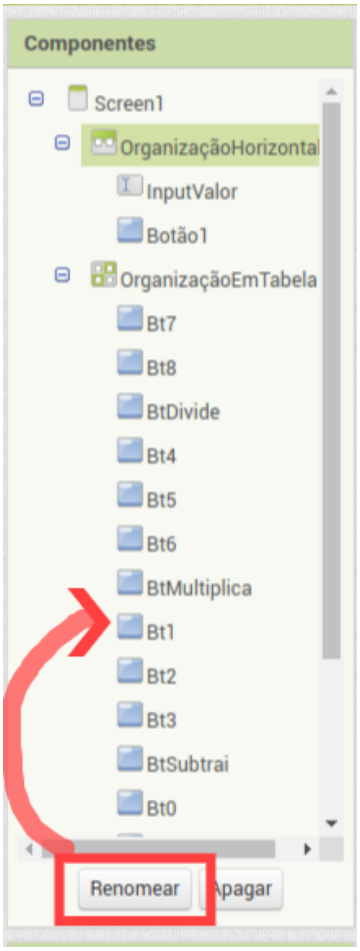
Como vamos passar a trabalhar com a lógica da nossa aplicação, os nomes dos componentes serão muito importantes para simplificar nosso trabalho ao lidar com os blocos de lógica.

Perceba que a cada botão inserido um novo elemento é apresentado na caixa de "Componentes", é importante verificar se eles estão sendo incluídos dentro do elemento "**OrganizaçãoEmTabela1**". Os botões são automaticamente nomeados conforme a sequência de inclusão "Botão1", "Botão2" etc.



Fonte: MIT App Inventor 2

O nome padronizado dos elementos pode ajudar quando estamos criando nossa tela, porém, vai nos confundir quando for necessário incluir os blocos de lógica relacionados a cada botão, desta forma, é importante renomear todos os componentes para nos ajudar na criação da lógica. Na caixa "**Componentes**" basta clicar sobre o botão, depois clicar em "**Renomear**", alterando o nome para um que esteja relacionado ao valor de botão, exemplo "**Bt7**", "**Bt8**", "**BtDivide**" etc. Faça isso para todos os componentes da sua tela.



Fonte: MIT App Inventor 2

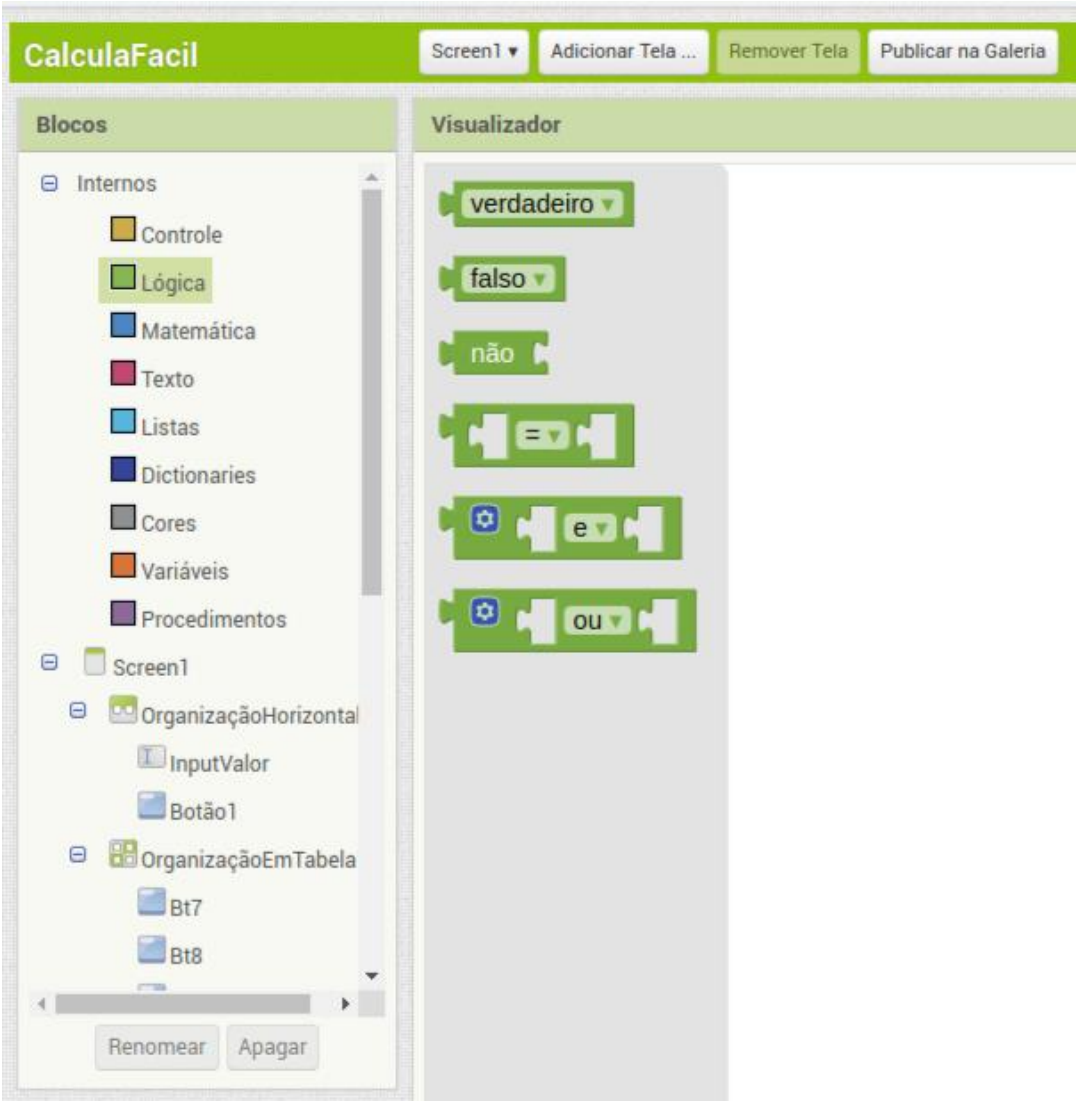
Com os componentes renomeados podemos dar início a construção da lógica, para isso vamos acessar a área de "**Blocos**" do MIT AI2.

Fonte: MIT App Inventor 2

Temos diversos tipos de blocos que podem ser usados para criar a lógica da aplicação, Controle, Lógica, Matemática, Texto, Listas, Dictionaries, Cores, Variáveis e Procedimentos.

Fonte: MIT App Inventor 2

Os blocos de Controle são usados para o controle geral da lógica, como exemplo os blocos de comparação lógica “**Se .. senão**”, estruturas para repetição (laços) como “**para cada… fazer**”, “**enquanto … fazer**”, além de blocos para controlar a abertura e fechamento de telas.



Fonte: MIT App Inventor 2

Os blocos “**Lógica**” são usados para operadores de comparação lógica como igual, maior, menor, diferente, etc., geralmente em conjunto com blocos de controle do tipo “**Se...**”.

Fonte: MIT App Inventor 2

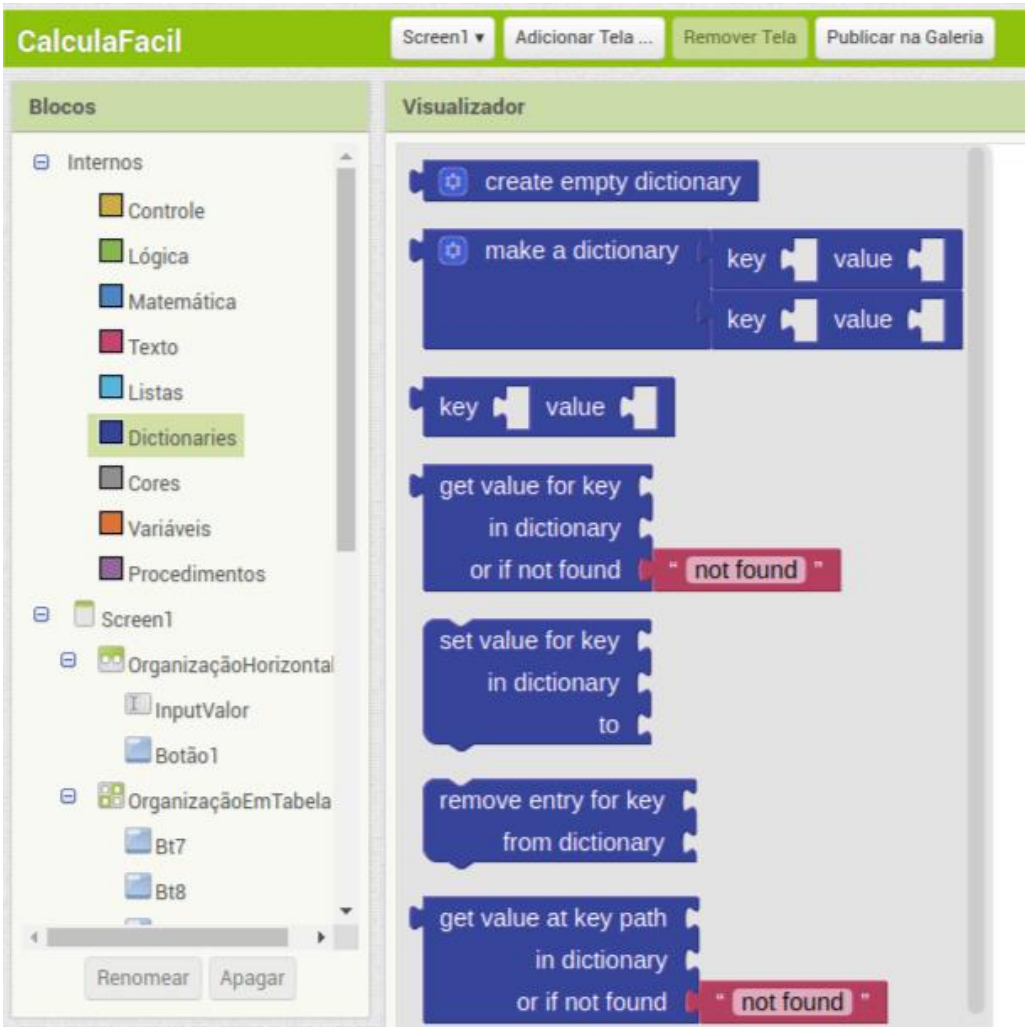
Os blocos “**Matemática**” possibilitam a execução de diversos tipos de cálculos como soma, subtração, multiplicação, divisão, potenciação, raiz quadrada, seno, cosseno, entre outros, além da geração de números ou frações aleatórias.

Fonte: MIT App Inventor 2

Os blocos “**Texto**” possibilitam realizar diversas operações em cadeias de caracteres (variáveis textuais). Permite realizar comparações, concatenações, obtenção de subconjuntos da cadeia textual principal, entre outros.

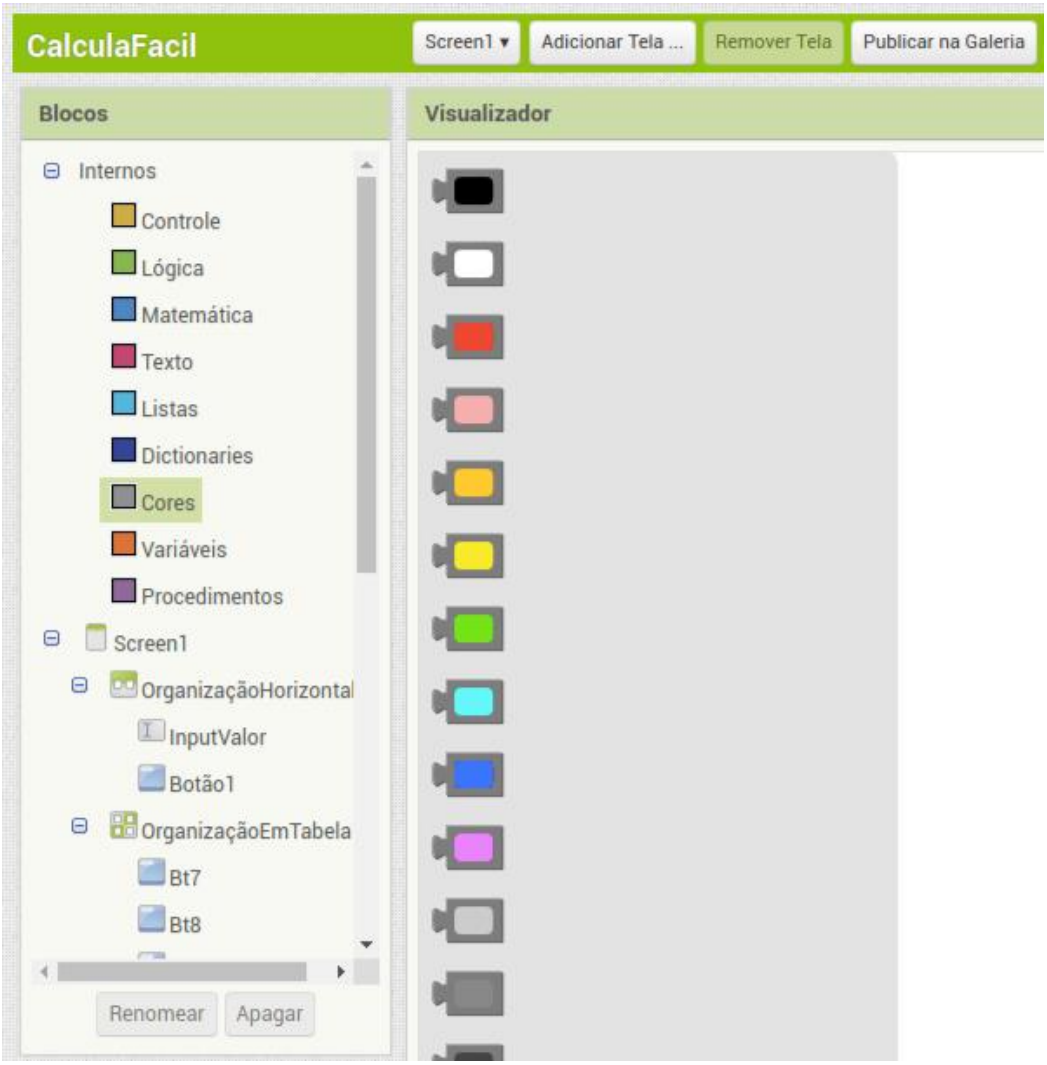
Fonte: MIT App Inventor 2

Os blocos “**Lista**” são destinados à realização de operações com listas de valores (*Arrays*), possibilitando criar listas, procurar um elemento dentro da lista, inserir ou remover elementos da lista, entre outros.



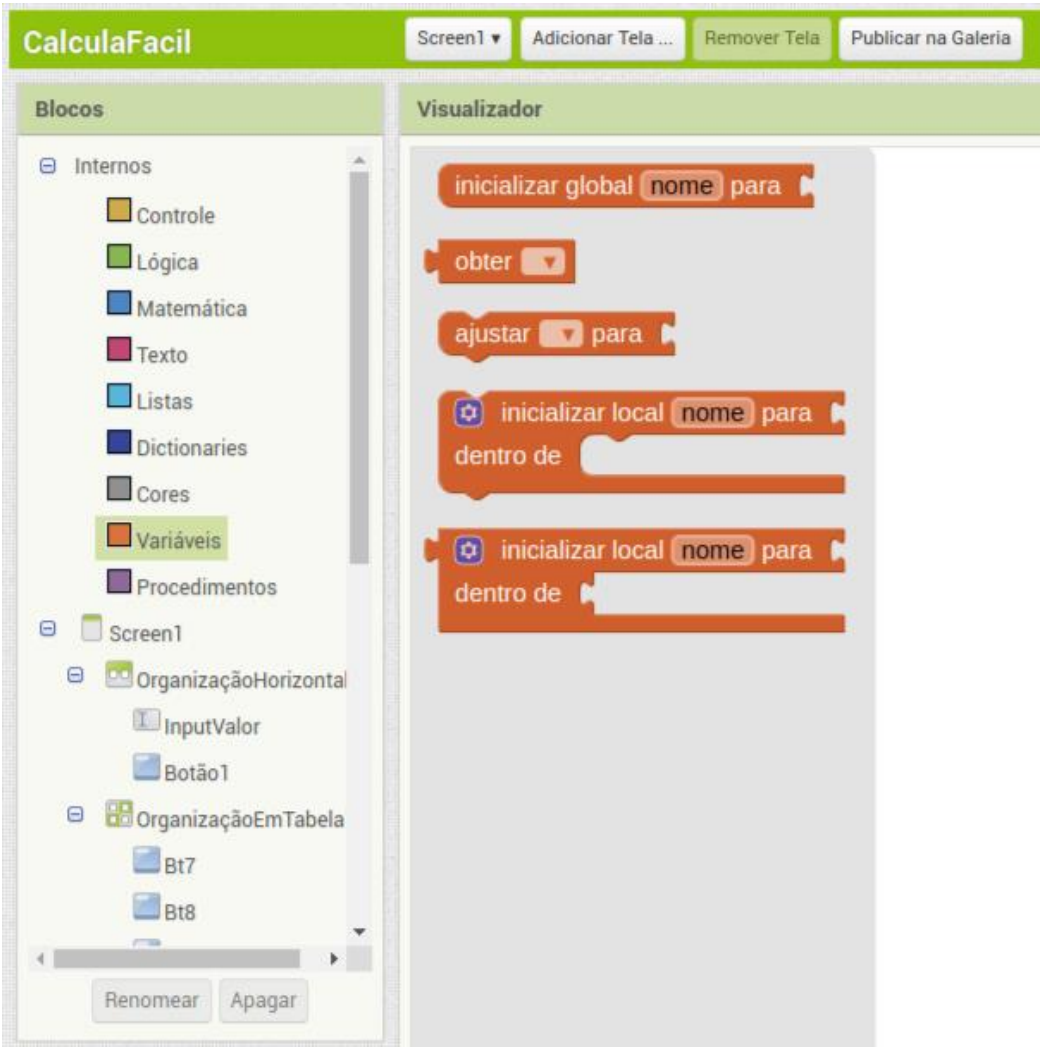
Fonte: MIT App Inventor 2

Os blocos “**Dictionaries**” são destinados ao tratamento de dicionários usando recursos de listas baseadas no conceito chave-valor, estes recursos são usados normalmente para internacionalização (tradução) de aplicações, mas podem ser usados para qualquer situação onde partindo de uma chave (código) deseja-se ter um valor associado.



Fonte: MIT App Inventor 2

Os blocos “**Cores**” são destinados ao tratamento de cores que podem ser usadas nos componentes das telas da aplicação para cor de frente ou fundo. Existem recursos para cores pré-definidas e também para definição de cores pela codificação RGB (veja mais em: https://www.w3schools.com/colors/colors_rgb.asp).



Fonte: MIT App Inventor 2

Os blocos “**Variáveis**” permitem a definição e tratamento de variáveis globais na lógica da aplicação.

Fonte: MIT App Inventor 2

Finalmente, blocos “**Procedimentos**” permitem a definição e tratamento de procedimentos (métodos). Eles são importantes para evitar duplicação de lógica dentro da aplicação.

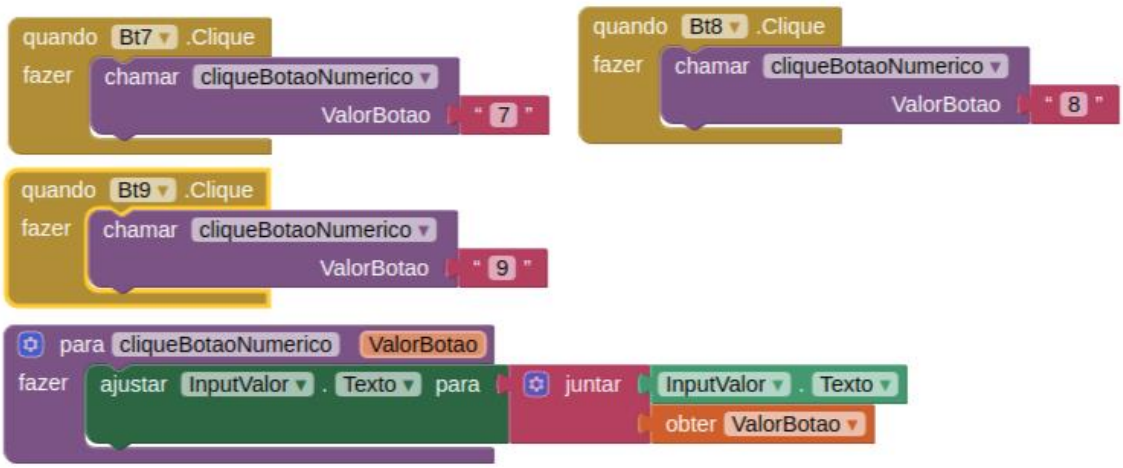
TRATANDO A LÓGICA DA CALCULADORA

Incluindo tratamento para os botões numéricos

Quando um botão numérico é pressionado, exemplos “Bt7”, “Bt8”, “Bt9”, etc., o valor do respectivo botão deve ser concatenado no valor já apresentado no componente “inputValor”.

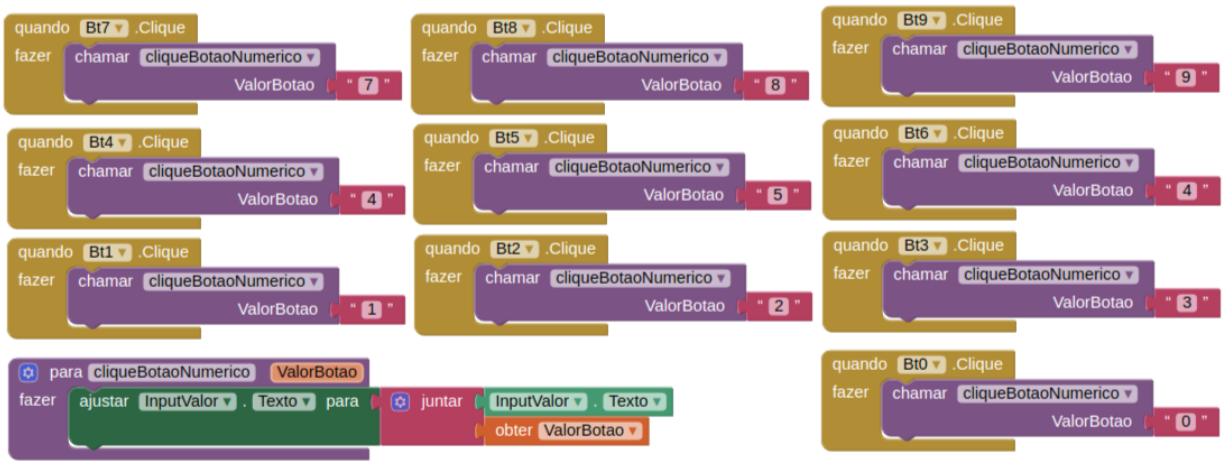
Fonte: MIT App Inventor 2

Porém, incluindo o tratamento desta forma, temos muita repetição de lógica em todos os botões, e se for necessário ajustar alguma coisa, teremos que fazer isso em todos os botões também. Neste caso, podemos criar um procedimento chamado “cliqueBotaoNumerico” e alterar a lógica de clique nos botões para chamar este procedimento. Para passar o valor do botão que está sendo pressionado, incluímos um parâmetro chamado “ValorBotao”.



Fonte: MIT App Inventor 2

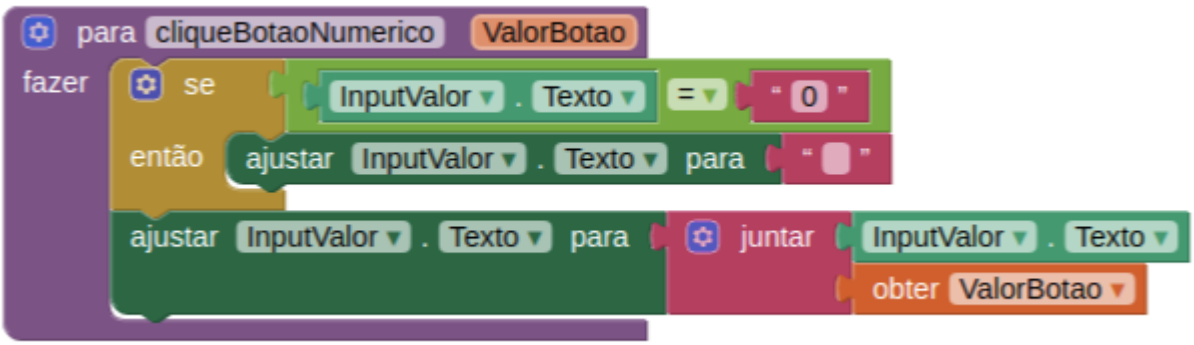
Agora podemos replicar esta lógica de chamada para todos os botões numéricos.



Fonte: MIT App Inventor 2

Porém, geramos um efeito colateral indesejado no caso do botão “0” (zero). Da forma que está a lógica, podemos inserir o valor “00000” que não é interessante, também ficamos com valores “0111” (com o zero à esquerda), assim, podemos verificar se o valor do componente “inputValor” é zero e tratar antes de incluir um novo zero.

Como temos um procedimento, basta incluir o tratamento no procedimento que passa a valer para todos os botões, neste caso, se o valor do “inputValor” é zero, atribuímos vazio “ ” para ele.



Fonte: MIT App Inventor 2

Agora temos todos os nossos valores numéricos, incluindo o 0 (zero) com o funcionamento adequado.

Videoaula 3

Agora, assista à videoaula na qual vamos incluir o tratamento de decimais.

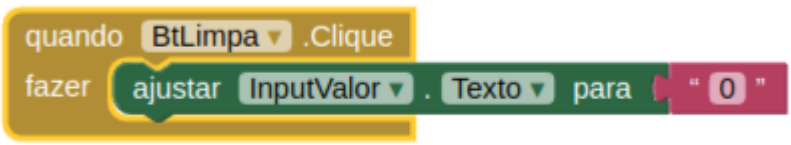
Disponível em: https://www.youtube.com/watch?v=WMRWGR4zwEg. Acesso em: 15 mar. 2021.

Videoaula 3

Utilize o QRcode para assistir!

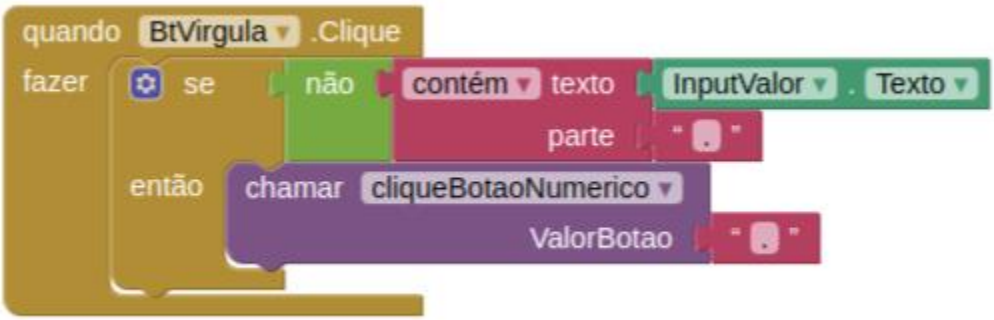
Agora, assista à videoaula na qual vamos incluir o tratamento de decimais.

Antes de iniciar o tratamento das casas decimais, vamos incluir o tratamento para o botão “Limpar”, que vai facilitar bastante os testes. Para isso, basta mover “0” para o valor de “inputValor” quando o botão “BtLimpa” for pressionado.



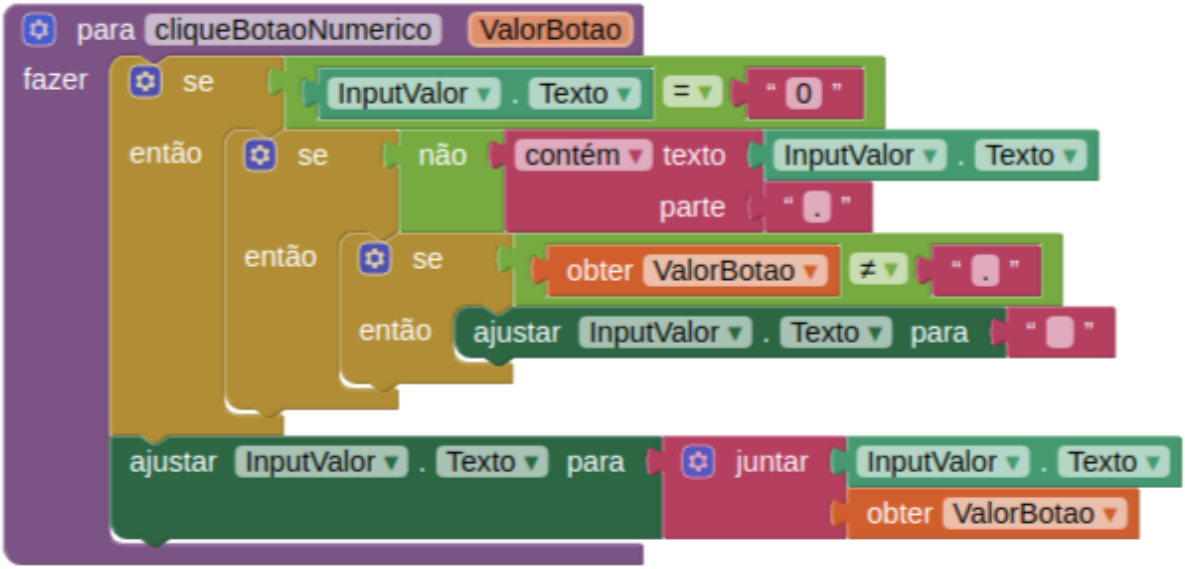
Fonte: MIT App Inventor 2

Agora temos que pensar em um tratamento adequado para a vírgula (ou ponto) decimal. Neste caso, não podemos apenas concatenar o valor, pois ficaríamos com valores “12..3.4..3” que não devem ocorrer. Apenas um delimitador de casa decimal deve ser aceito. Para tratar isso, basta verificar se “inputValor” já contém um separador decimal, e incluir apenas se não tiver.



Fonte: MIT App Inventor 2

Porém, desta forma, o tratamento que realizamos para evitar “0034” passou a gerar um efeito colateral indesejado, pois se o valor atual for “0”, e for pressionada a vírgula, o valor vai ficar “ .” e não “0.” como seria o ideal. Assim, novamente ajustamos o procedimento “cliqueBotaoNumerico” para corrigir isso.



Fonte: MIT App Inventor 2

Ótimo, novamente apenas ajustando o procedimento, corrigimos o comportamento para todos os botões numéricos e para o separador decimal.

Videoaula 4

Agora, assista à videoaula em que veremos como realizar todos os tratamentos que faltam para o funcionamento de nossa calculadora.

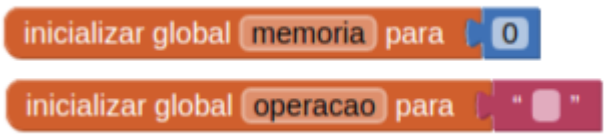
Disponível em: https://www.youtube.com/watch?v=h6AtrGHTMyw. Acesso em: 15 mar. 2021.

Videoaula 4

Utilize o QRcode para assistir!

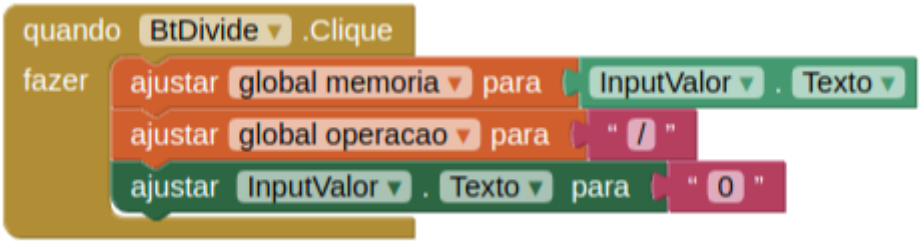
Agora, assista à videoaula em que veremos como realizar todos os tratamentos que faltam para o funcionamento de nossa calculadora.

Para realizar o tratamento dos cálculos necessitamos de duas variáveis, uma para guardar o valor atual na memória e outra para guardar a operação que deve ser executada.



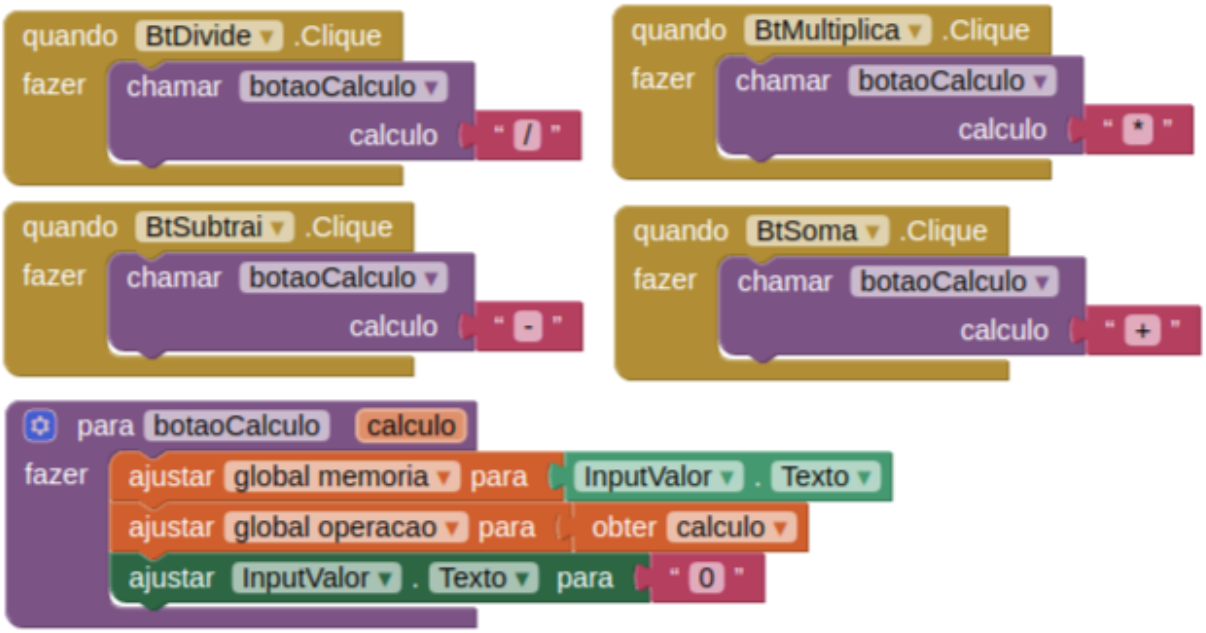
Fonte: MIT App Inventor 2

Basicamente, o tratamento inicial seria guardar o valor que está na tela na memória, guardar a operação que está sendo solicitada na memória e zerar o valor que aparece na tela.



Fonte: MIT App Inventor 2

Porém, da mesma forma que nos botões numéricos, temos que realizar isso em todos os botões de cálculo e também teremos que incluir o tratamento de cálculo propriamente dito, pois da forma que só estamos guardando os valores, ainda não estamos calculando nada. Assim, vamos modificar essa lógica criando um procedimento que será chamado quando o botão de cálculo é clicado.

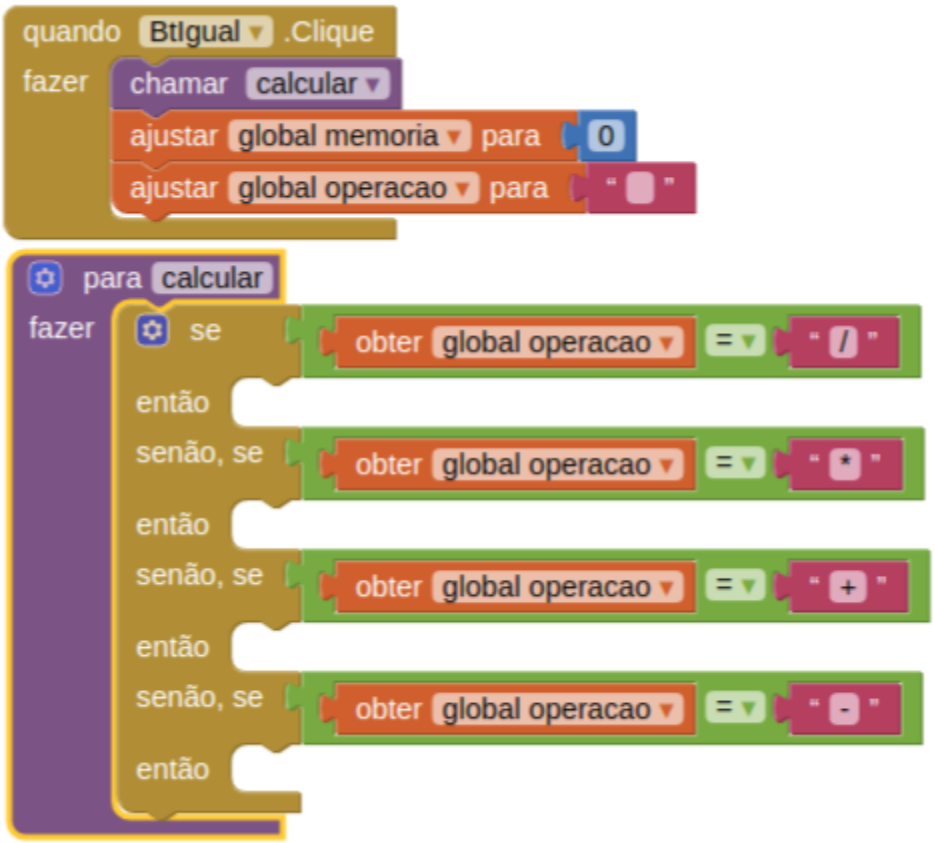


Fonte: MIT App Inventor 2

Ótimo, desta forma temos o procedimento chamado “botaoCalculo” que é chamado quando qualquer botão de operação é pressionado.

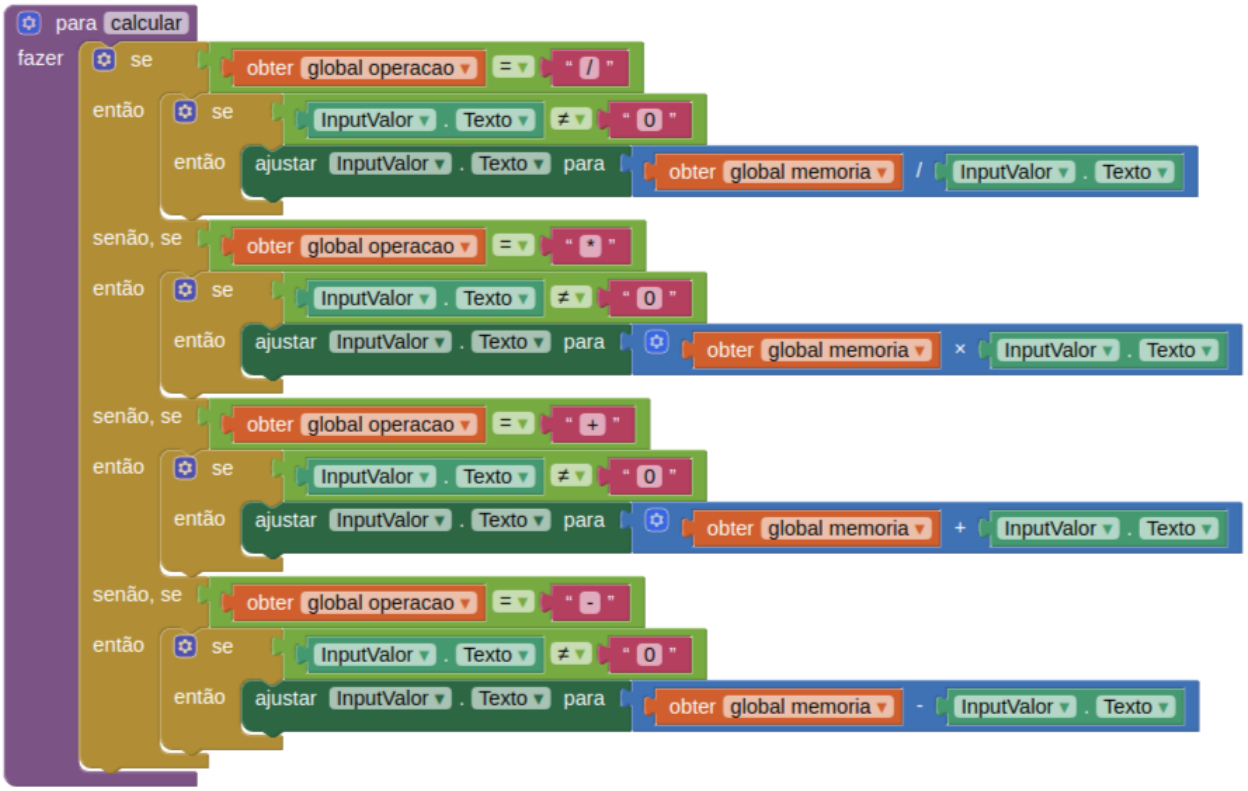
Mas, ainda não realizamos o cálculo, ele vai ocorrer quando pressionarmos novamente um botão de operação ou quando pressionamos o botão “=”, podemos criar um procedimento chamado “calcular”, e chamar este procedimento quando o botão “BtIgual” é

pressionado. Depois que o cálculo é realizado, podemos limpar os dados das duas variáveis de memória.



Fonte: MIT App Inventor 2

Desta forma, podemos incluir um tratamento adequado de cálculo dependendo da operação que está na memória, incluindo a verificação do valor atual para evitar divisão por zero.

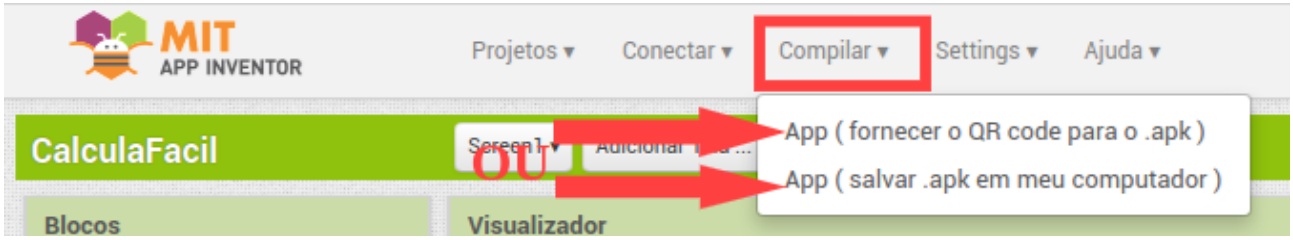


Fonte: MIT App Inventor 2

Com o procedimento de cálculo definido, podemos incluir o tratamento no procedimento “botaoCalculo”, para caso alguma operação esteja na memória, antes de modificar o valor das variáveis de memória, o procedimento de “calcular” seja chamado.

Fonte: MIT App Inventor 2

Agora que temos nossa calculadora funcional, podemos usar as opções de compilação para gerar o arquivo “.apk” e publicar nossa aplicação.



Fonte: MIT App Inventor 2

Lembre-se que o *link* gerado pela opção "**App (fornecer o QR code para o .apk)**" vale por apenas duas horas.

Encerramento da Unidade

Como vimos nesta unidade, podemos usar construtores de aplicação como o *MIT App Inventor 2* para construir aplicações reais, prototipar aplicações e definir um funcionamento simulado ou até criar MVPs para nossos projetos mais complexos validando a ideia principal de nossa aplicação.

Porém, lembre-se que ferramentas como o *MIT App Inventor 2* foram criadas com foco principal no aprendizado do desenvolvimento de aplicações móveis, e não para uso produtivo ou comercial.

Quem tiver interesse em se aprofundar um pouco mais no MIT AI2 pode acessar a **Biblioteca do MIT App Inventor** (http://appinventor.mit.edu/explore/library).

Para criar aplicações de forma mais profissional temos diversas plataformas de desenvolvimento que podem ser usadas, dentre elas: o Ionic que veremos em nosso próximo módulo.

Referências

FÉLIX, Rafael; SILVA, Everaldo Leme. **Arquitetura para computação móvel.** 2. ed. São Paulo: Pearson Education do Brasil, 2019.

FLATSCHART, Fábio. **HTML 5**: Embarque Imediato. Rio de Janeiro: Editora Brasport, 2011.

LEE, Valentino; SCHNEIDER, Heather; SCHELL, Robbie. **Aplicações móveis**: arquitetura, projeto e desenvolvimento. São Paulo: Pearson Makron Books, 2005.

MARINHO, Antonio Lopes (org.). **Desenvolvimento de aplicações para Internet**. São Paulo: Pearson Education do Brasil, 2016.

SEGURADO, Valquiria Santos (org.). **Projeto de interface com o usuário**. 1. ed. São Paulo: Pearson Education do Brasil, 2015.

SILVA, Diego (org). **Desenvolvimento para dispositivos móveis**. São Paulo: Pearson Education do Brasil, 2016.

SOMMERVILLE, Ian. **Engenharia de software**. Tradução Luiz Claudio Queiroz. Revisão técnica Fábio Levy Siqueira. 10. ed. São Paulo: Pearson, 2018. 768 p.

Portal Periódicos. CAPES (http://www.periodicos.capes.gov.br)

Google Acadêmico (http://scholar.google.com.br)

SciELO (http://www.scielo.br)

arXiv.og (http://arxiv.org)

ACM Digital Library (http://dl.acm.org)

IEEE Xplore (http://ieeexplore.ieee.org/Xplore/home.jsp)

ScienceDirect (http://www.sciencedirect.com)


UniFil
EDUCAÇÃO A DISTÂNCIA
UNIFIL.BR