

Programação para Dispositivos Móveis

Curso de Ciência da Computação

Universidade Paulista (UNIP)

UNIP

UNIVERSIDADE PAULISTA

INTRODUÇÃO À DISCIPLINA E CONCEITOS INICIAIS

Prof. Ms. Clayton A. Valdo
clayton.valdo@docente.unip.br

Prof. Ms. Peter Jandl
peter.junior@docente.unip.br

Prof. Ms. Télvio Orrú
telvio.orrú@docente.unip.br

AULA 1



Sobre a Disciplina

“Programação para Dispositivos Móveis”

Programação para Dispositivos Móveis

Curso de Ciência da Computação

Universidade Paulista (UNIP)

INTRODUÇÃO À DISCIPLINA E CONCEITOS INICIAIS



Metodologia e Avaliação

- ❑ Aulas expositivas **dialogadas**;
- ❑ Exercícios **teóricos ou práticos** (em algumas aulas);
- ❑ Proposta de **tarefas extraclasses**;
- ❑ Avaliação **Teórica e Trabalho**.

Avaliação

1º Bimestre

Prova Teórica (6 pts)
+
Trabalho (4 pts)

2º Bimestre

Prova Teórica (6 pts)
+
Trabalho (4 pts)

$$NF = (N1B + N2B) / 2$$

Envio dos Exercícios

- ❑ Todos os exercícios realizados deverão ser encaminhados por e-mail;
- ❑ O formato apresentado ao lado deverá ser seguido para que a atividade não seja descartada;
- ❑ Os exercícios deverão ser feitos em **dupla** ou **individualmente**.

Assunto:

Nome1_Nome2_PDM

Corpo do E-mail:

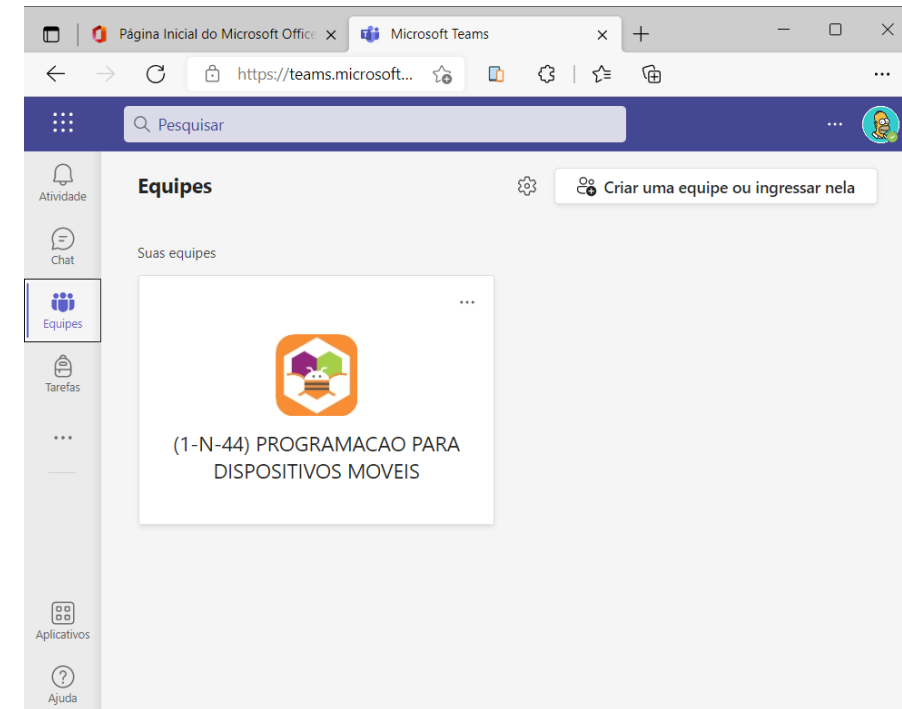
Nome e RA das duplas
Nº da Aula
1/2022

Arquivo (Anexo):

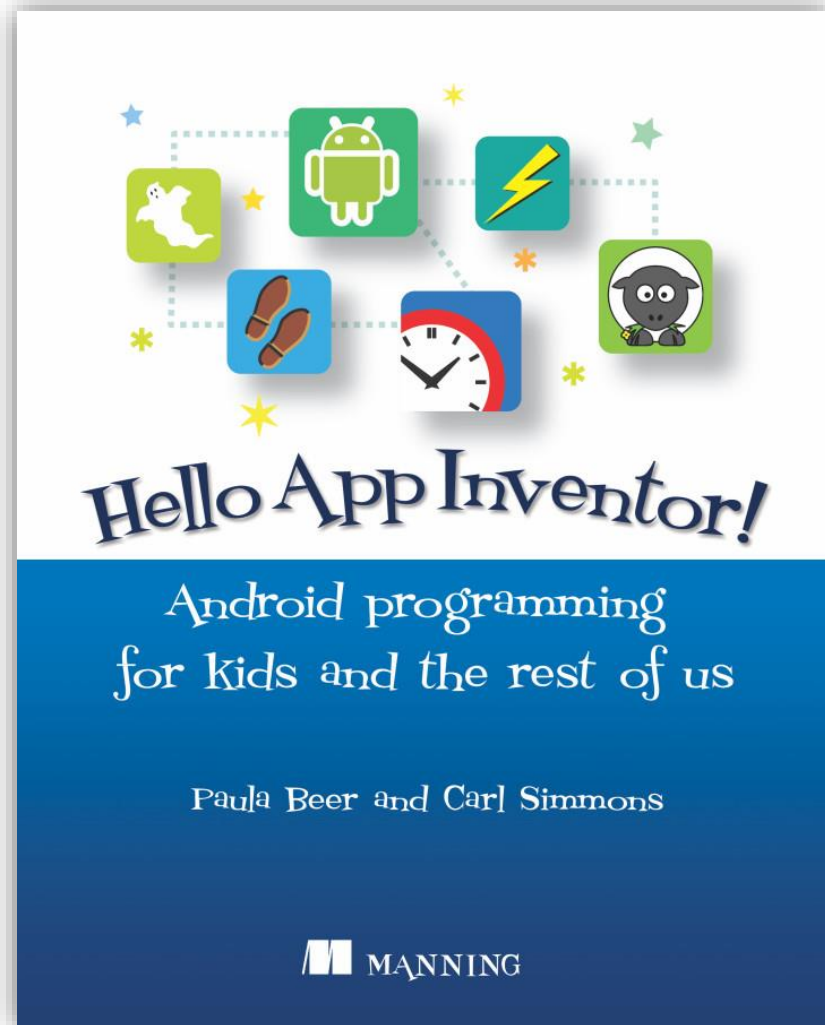
Nome e RA das duplas

Download das Aulas

- ❑ Serão disponibilizadas através de **pasta compartilhada (Teams)**;
- ❑ Estará disponível sempre **após a aula** ou no dia da aula (caso seja necessário);
- ❑ O **endereço de acesso** está dentro da sua área de acesso ao Microsoft Office, dentro do **Portal da Unip**.



Bibliografia Básica



Beer, P; Simmons, C. **Hello App Inventor: Android Programming.** 1ª ed. New York: Manning, 2015.

Encontrado
facilmente pela
Internet!

Vamos ao Conteúdo!

Tudo Acaba Modernizando

- ❑ Os primeiros celulares eram **grandes, pesados** e muito **caros**;
- ❑ Os **smartphones atuais** são **pequenos**, realizam **muitas tarefas** e possuem **alto processamento**;
- ❑ Em outras palavras, as pessoas nunca carregaram antes tanto **poder computacional em seus bolsos**;
- ❑ Um smartphone básico opera **centenas de vezes mais rápido** do que o **computador Apollo** que levou o homem à lua.



O que um Smartphone pode fazer?

- ❑ As **aplicações disponíveis** para os smartphones fazem deles uma **arma poderosa**;
- ❑ É possível, por exemplo, **criar uma aplicação** para **enviar fotos** para a sua lista de contatos;
- ❑ Pode-se também utilizar o recurso de *touch-screen* e orientação para **criar jogos divertidos**;
- ❑ Por que não criar um **ranking no jogo** e disponibilizar na internet para que todos vejam?.



O que é Android?



- ☐ Android é um dos **sistemas operacionais (SO)** disponíveis no mercado para **dispositivos móveis**;
- ☐ Para que uma **aplicação funcione corretamente** é necessário que haja um **SO capaz de gerenciá-la**;
- ☐ Em outras palavras, o SO faz a **ponte entre as aplicações e o hardware** do dispositivo;
- ☐ **Android, iOS e Windows Phone** são os **principais SOs** encontrados em dispositivos móveis.

O que é um App?

- ❑ **App é um programa**, ou seja, uma **lista de instruções** que diz o que o dispositivo deve fazer;
- ❑ Esse termo é usado para se referir a **programas desenvolvidos para dispositivos móveis**;
- ❑ Para a criar esses Apps é necessário a utilização de uma **linguagem de programação**;
- ❑ Existem várias linguagens disponíveis para serem utilizadas, como: **Java, C# e Swift**.



Programar Faz Bem!

- ❑ A programação **além de ser engraçada** também permite **desenvolver a lógica** (raciocínio computacional);
- ❑ Através dela é possível **resolver muitos problemas** que temos em nosso **mundo real**;



- ❑ As **habilidades geradas pela programação** podem ser úteis para **qualquer tipo de profissional**;
- ❑ **Criatividade, resolução de problemas, lógica, entendimento e paciência** são alguns exemplos.

Sobre o App Inventor

- ❑ **App Inventor** é uma **linguagem de programação** utilizada via **browser** para criar Apps para Android;
- ❑ Desenvolvido inicialmente pelo **Google**, agora é mantido pelo **Instituto de Tecnologia de Massachusetts (MIT)**;
- ❑ O seu ambiente gráfico **facilita a codificação** por meio da técnica de **arrastar e soltar objetos**;
- ❑ Tais objetos podem ser **componentes ou blocos de código** sendo muito similar ao **Scratch**.



Por que usar o App Inventor?

- ❑ Há várias linguagens de programação, mas o App Inventor é **voltado para dispositivos móveis**;
- ❑ Além disso, a sua **facilidade de codificação** permite que **iniciantes assimilem melhor** os conteúdos;
- ❑ A **taxa de erros tende a ser muito menor** do que em linguagens de programação tradicionais (**mais técnicas**);
- ❑ Ao usar o App Inventor você **não precisa se preocupar com letras maiúsculas, ponto e vírgula**, etc.

Requisitos Necessários

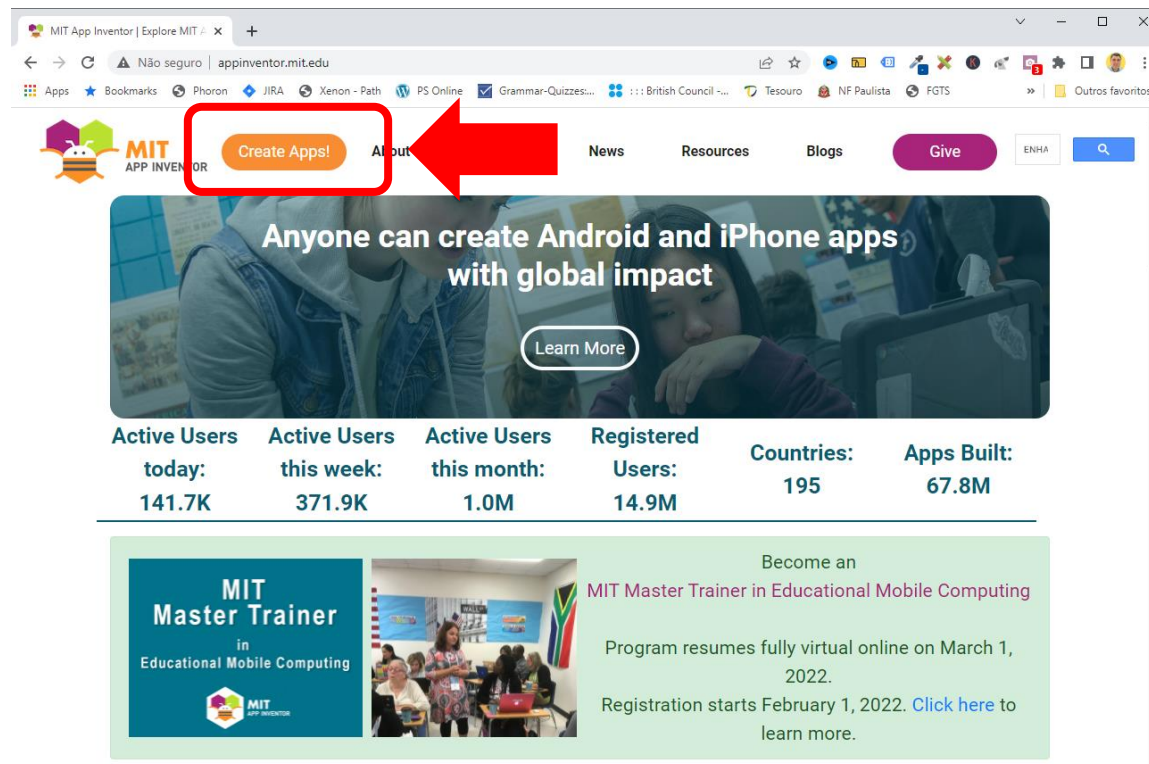
- ❑ Um **computador** rodando **Windows, Mac OS X** ou **Linux**;
- ❑ Um **browser**: **Chrome, Firefox** ou **Safari** (IE não funciona);
- ❑ Uma **conta do Google** para entrar no App Inventor;
- ❑ Um **smartphone com android** não é requisito, mas torna as coisas mais interessantes;
- ❑ **Boas ideias** e um pouco de **paciência!**.



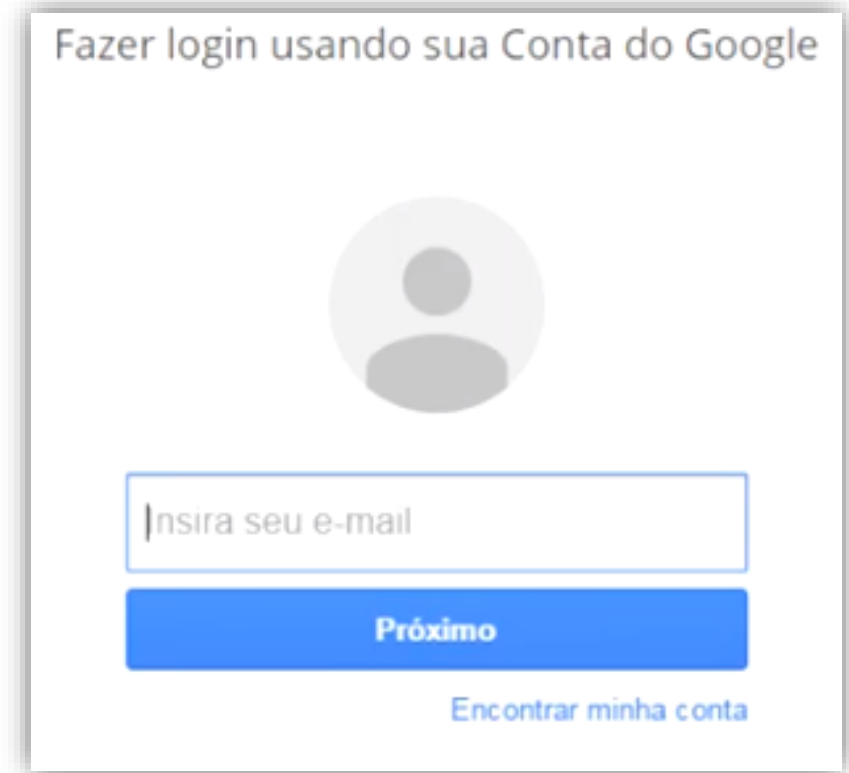
Acessando o App Inventor

1º) Acesse o site:

<http://appinventor.mit.edu/explore/>



2º) Logar com a conta do Google: (Crie uma caso necessário)



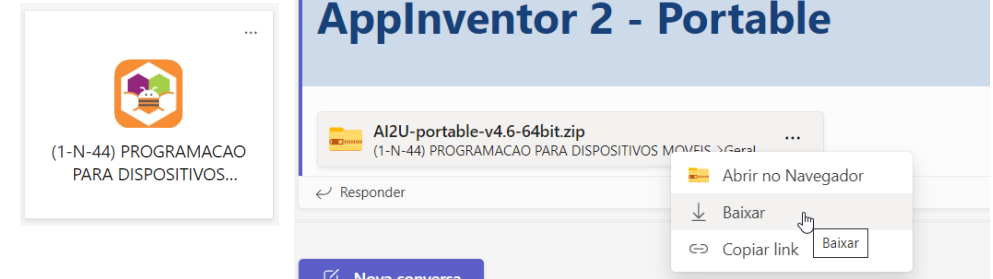
Ops, Temos um Problema!

- ❑ Como mencionado anteriormente, o **App Inventor** é uma **plataforma de ensino bastante simples**;
- ❑ Essa plataforma, por sua vez, **apresenta instabilidades** ficando **fora do ar** alguma vezes;
- ❑ Para contornar essa situação, iremos disponibilizar uma **versão *portable* (*pasta compartilhada*)**;
- ❑ Essa versão **possui algumas limitações** em relação a oficial, mas **funciona muito bem**.

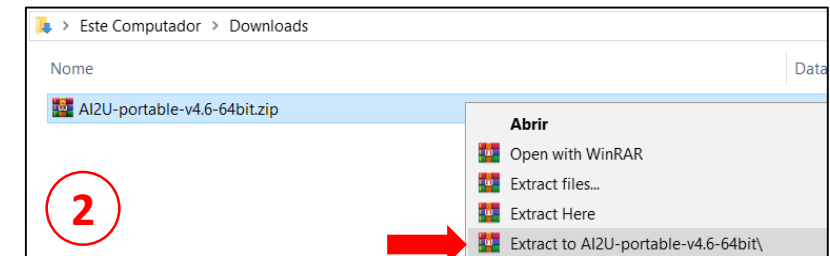


Então como Proceder?

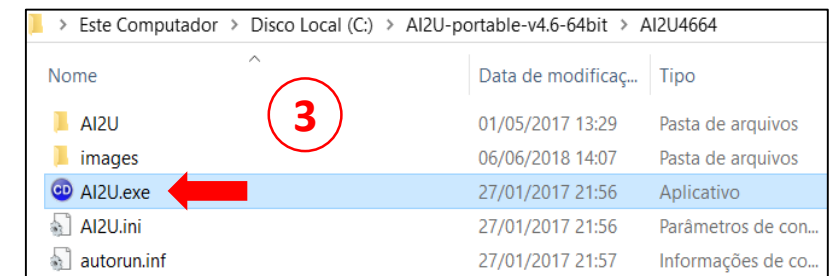
- ❑ **Passo 1:** Acesse o Teams na Equipe de Programação para Dispositivos Móveis → aba Geral → Publicação do link do Arquivo.



- ❑ **Passo 2:** Descompacte a pasta do App Inventor na unidade **C:**;



- ❑ **Passo 3:** Crie um **atalho** na área de trabalho para o executável do programa.

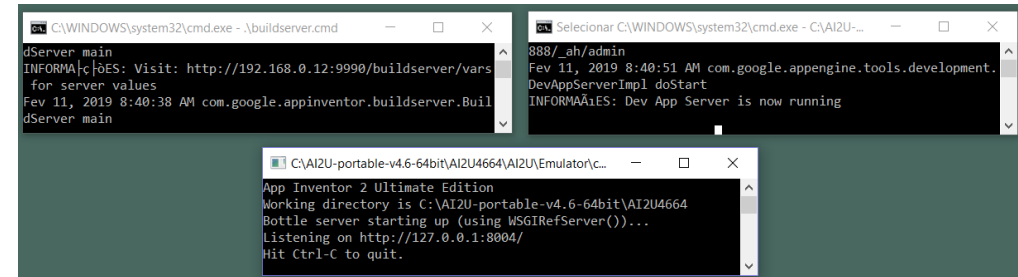


Executando o App Inventor

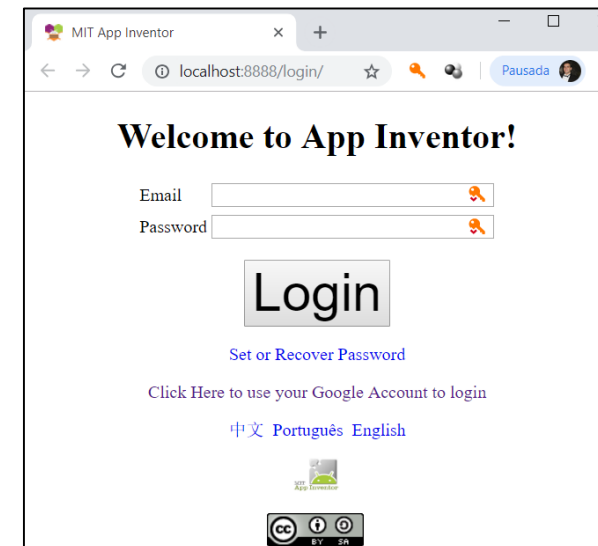
Passo 1: duplo clique no atalho criado



Passo 2: inicie o servidor da aplicação



Passo 3: execute o App Inventor



Entrando no Sistema

Welcome to App Inventor!

Email


Password

Login

[Set or Recover Password](#)

[Click Here to use your Google Account to login](#)

[中文](#) [Português](#) [English](#)

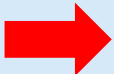




Not logged in

Email:

☐ Sign in as Administrator



Atenção:

Na versão portable não é necessário informar nenhum e-mail para logar. Basta clicar em “Log In”.

Tela Inicial do App Inventor



1 – Gerenciamento dos Projetos

4 – Conteúdos Adicionais (Tutoriais)

2 – Conexão com Emulador ou Dispositivo

5 – Documentação Oficial

3 – Gerar o APK da aplicação

6 – Projetos Desenvolvidos

Formas de Trabalhar com o App Inventor

(Conexão Wi-Fi com AI Companion)

- ❑ A aplicação pode ser testada **diretamente em um smartphone** com android;

- ❑ O celular e o computador devem estar **conectados à mesma rede (Wi-Fi)**;



- ❑ Deve ser instalado no celular o software chamado **AI2 Companion** disponível no Google Play Store.

Essa opção não funciona no App Inventor Portable, somente na versão oficial.

Formas de Trabalhar com o App Inventor

(Conexão via Cabo USB)

- ❑ Para utilizar o **cabo USB** é necessário instalar o **software chamado aiStarter** (*disponível no site do App Inventor*);
- ❑ Para que o computador possa **reconhecer o smartphone** também deve ser instalado os **drivers do celular** (*fabricante*);
- ❑ A vantagem é que o **celular carrega** enquanto você trabalha.



Formas de Trabalhar com o App Inventor

(Utilizando o Emulador)

- ❑ Como cada pessoa possui um **celular diferente**, a instalação de **drivers específicos** torna-se difícil;
- ❑ Manter **todos os equipamentos** em uma **mesma rede** também **não é tarefa fácil** em um ambiente corporativo;
- ❑ Portanto, iremos usar **durante as aulas o emulador** da própria aplicação;
- ❑ Precisamos apenas do **aiStarter** instalado e a **versão portable já contém esse software**.



Passos para criar um App

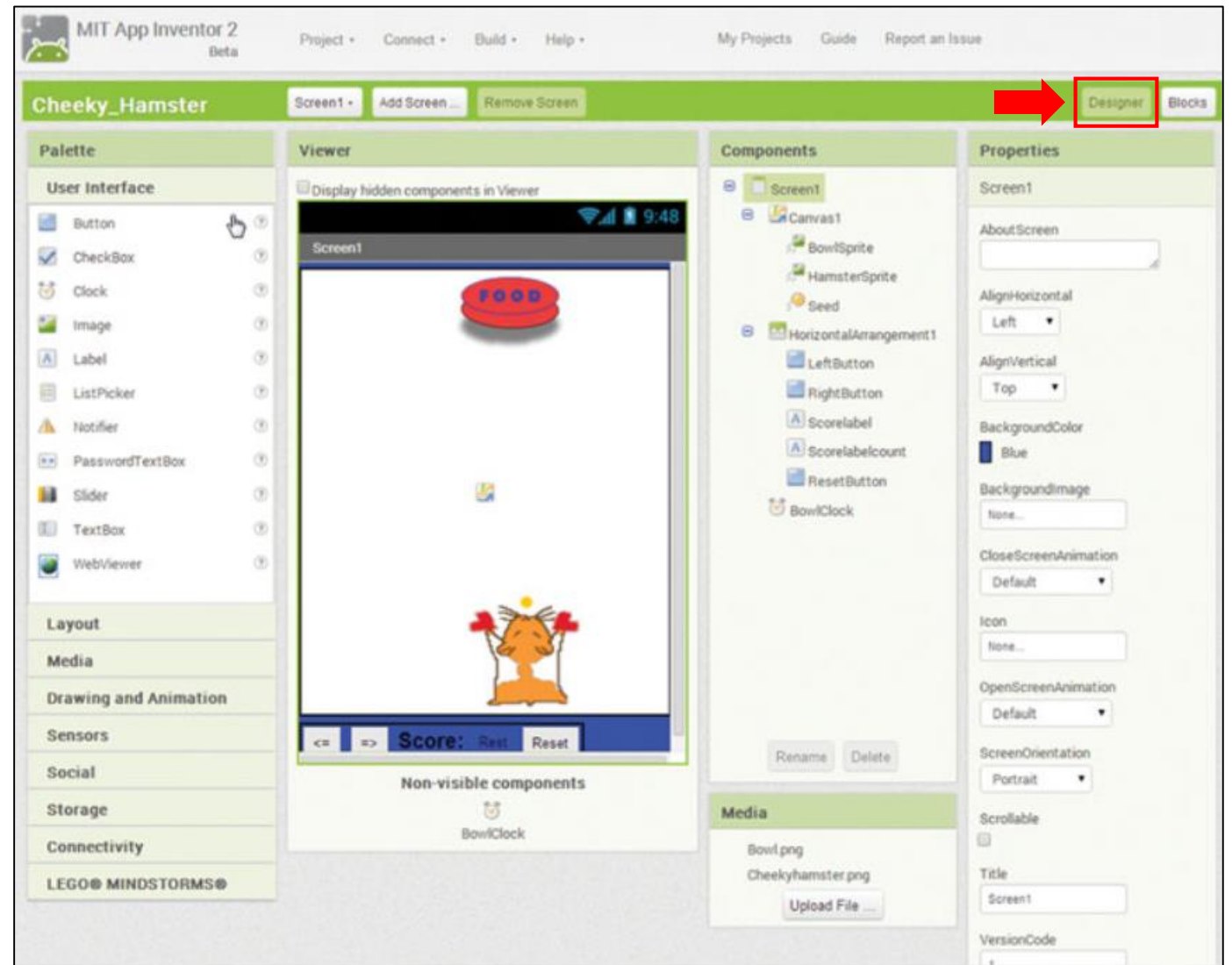
Para construir qualquer tipo de app no App Inventor é necessário executar três passos em telas diferentes:

- ❑ **Passo 1:** desenhar a tela do app usando a tela de designer do App Inventor;
- ❑ **Passo 2:** codificar a funcionalidade do app através do editor de blocos do App Inventor;
- ❑ **Passo 3:** testar o programa usando um smartphone ou o emulador.



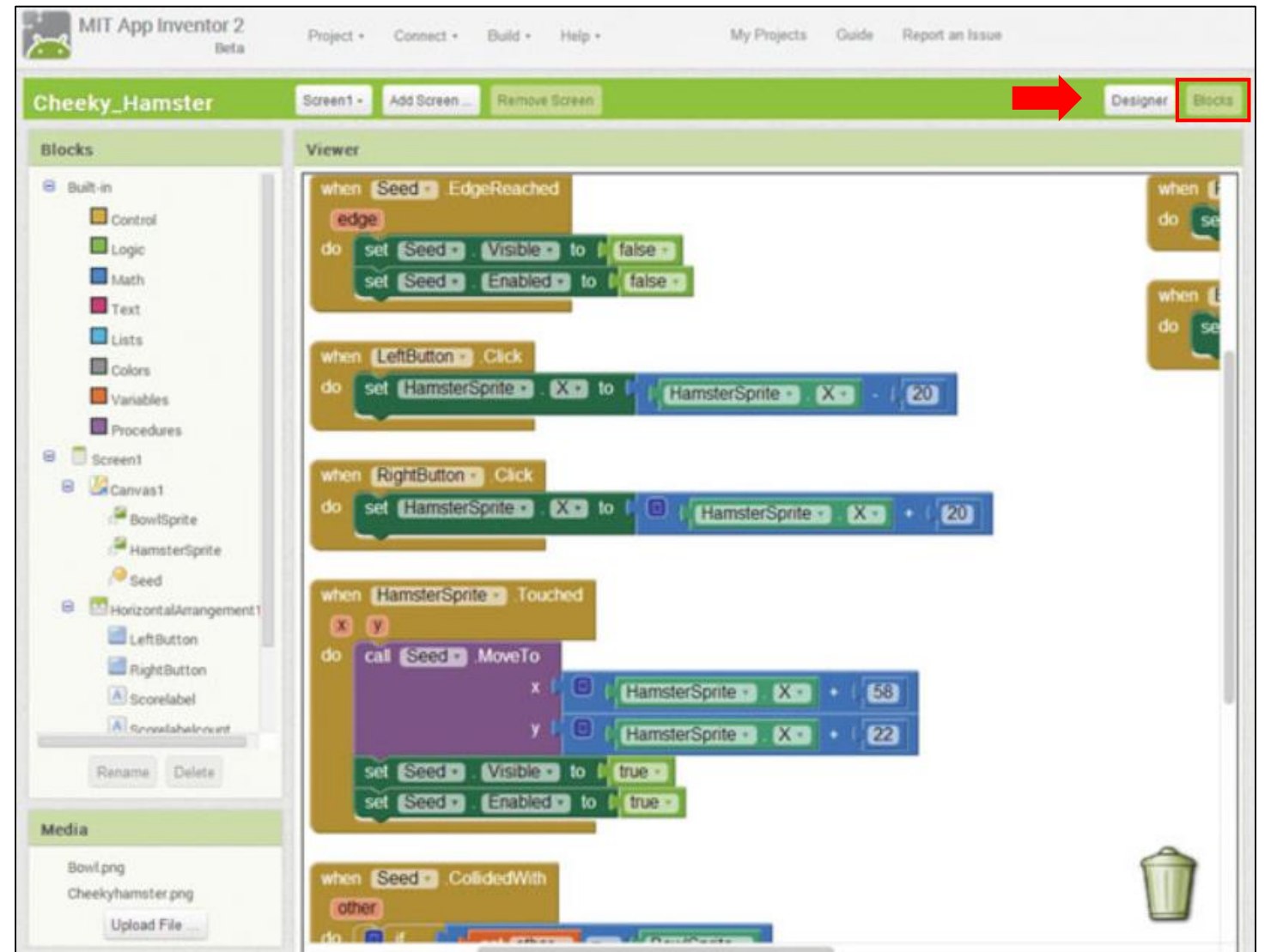
Tela de Designer

- ❑ Acessada através do botão: “Designer”;
- ❑ Permite que você adicione componentes ao projeto;
- ❑ É aqui que será construída a aparência do App.



Editor de Blocos

- ☐ Acessado através do botão: “Blocks”;
- ☐ Serve para especificar o que o App deve fazer (funcionalidade);
- ☐ Consiste na lógica do App, ou seja, como ele deve trabalhar.



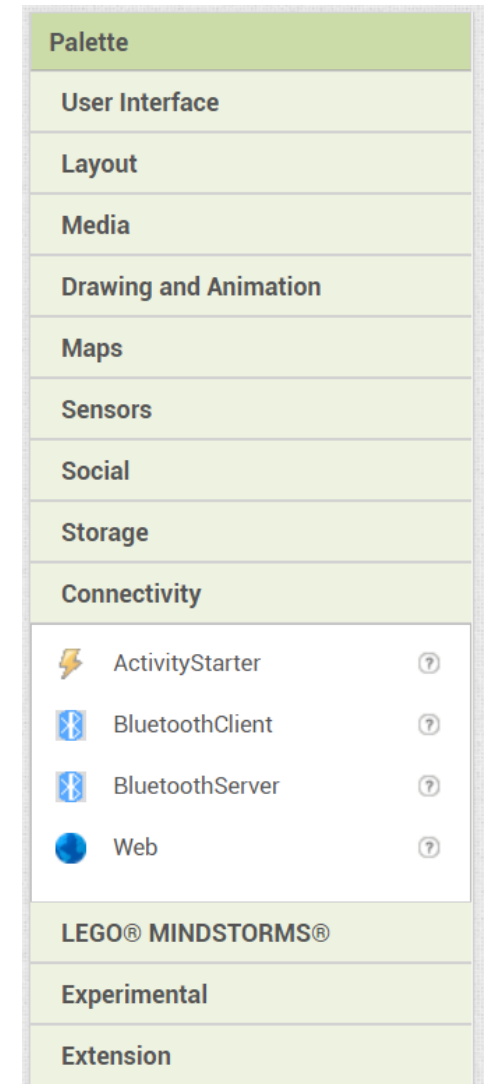
Testando o Programa

- ☐ O teste do App pode ser feito no celular ou no emulador;
- ☐ O emulador é um celular virtual que aparece na tela do computador;
- ☐ Em ambos os casos é possível testar o programa em tempo real.



Sobre os Componentes

- ❑ Os **componentes do App Inventor** estão organizados em uma **seção chamada “Paleta”**;
- ❑ Nessa seção é possível encontrar **componentes** para as mais **variadas finalidades** (funções);
- ❑ Os **componentes** que apresentam **funções similares** são agrupados em **subseções dentro da paleta**;
- ❑ Cada componente possui **propriedades específicas**, como: cor, tamanho, texto, etc.



Sobre os Blocos de Programas

- ☐ O **editor de blocos** fornece uma “**espécie de arquivo**” que contém **blocos de programas**;
- ☐ Cada “**gaveta do arquivo**” contém **blocos específicos** para uma **dada funcionalidade**;
- ☐ Existem basicamente **dois tipos de blocos** de programas: **os embutidos e os específicos** do componente;
- ☐ Os **embutidos** são **estáticos** (sempre disponíveis). Já os **específicos** aparecem ao **adicionar componentes**.



Exemplo de Blocos de Programas

The image shows a screenshot of a block-based programming environment. It is divided into two main panels: 'Blocks' on the left and 'Viewer' on the right. The 'Blocks' panel contains a list of categories under 'Built-in': Control (orange), Logic (green), Math (blue), Text (pink), Lists (light blue), Colors (grey), Variables (orange), and Procedures (purple). Below these are 'Screen1' (a screen icon) and 'Button1' (a button icon). At the bottom of the 'Blocks' panel is 'Any component' (a plus icon). A callout box labeled 'Blocos Embutidos' points to the 'Built-in' category. Another callout box labeled 'Blocos Específicos' points to the 'Button1' category. The 'Viewer' panel displays a series of event-driven blocks. Each block starts with 'when' followed by a dropdown menu showing 'Button1' and an event name, followed by a 'do' block. The events shown are: '.Click', '.GotFocus', '.LongClick', '.LostFocus', '.TouchDown', and '.TouchUp'. A callout box labeled 'Visualização dos Blocos' points to the 'Viewer' panel.

Blocos Embutidos

Blocos Específicos

Visualização dos Blocos

Visão Geral dos Blocos Embutidos

(Continua)

Control: Oferece **blocos de decisão** que dizem ao programa o que fazer baseado em testes condicionais. Já os **blocos de repetição** é uma maneira eficiente de repetir tarefas.

Logic: Disponibiliza os valores lógicos **true** ou **false**, além dos testes lógicos: **and**, **or**, **not** e **=**.

Math: Os blocos dessa categoria são utilizados para **criar e comparar números**, além de permitir a representação de **todas as funções** encontradas em uma calculadora.

Visão Geral dos Blocos Embutidos

(Continua)

Text: Consiste em blocos que **criam**, **modificam**, **juntam**, **separam**, **substituem** e **comparam** textos.

Lists: São blocos que possibilitam **armazenar**, **manipular** e **mostrar** listas de coisas, como por exemplo um menu.

Colors: Os seus blocos permitem **escolher** e **trocar** a cor de qualquer coisa que aparece na tela, como um texto ou botão.

Visão Geral dos Blocos Embutidos

Variables: Os seus blocos **criam variáveis**. Uma variável é um **espaço de armazenamento** para uma **pequena quantidade de dados**.

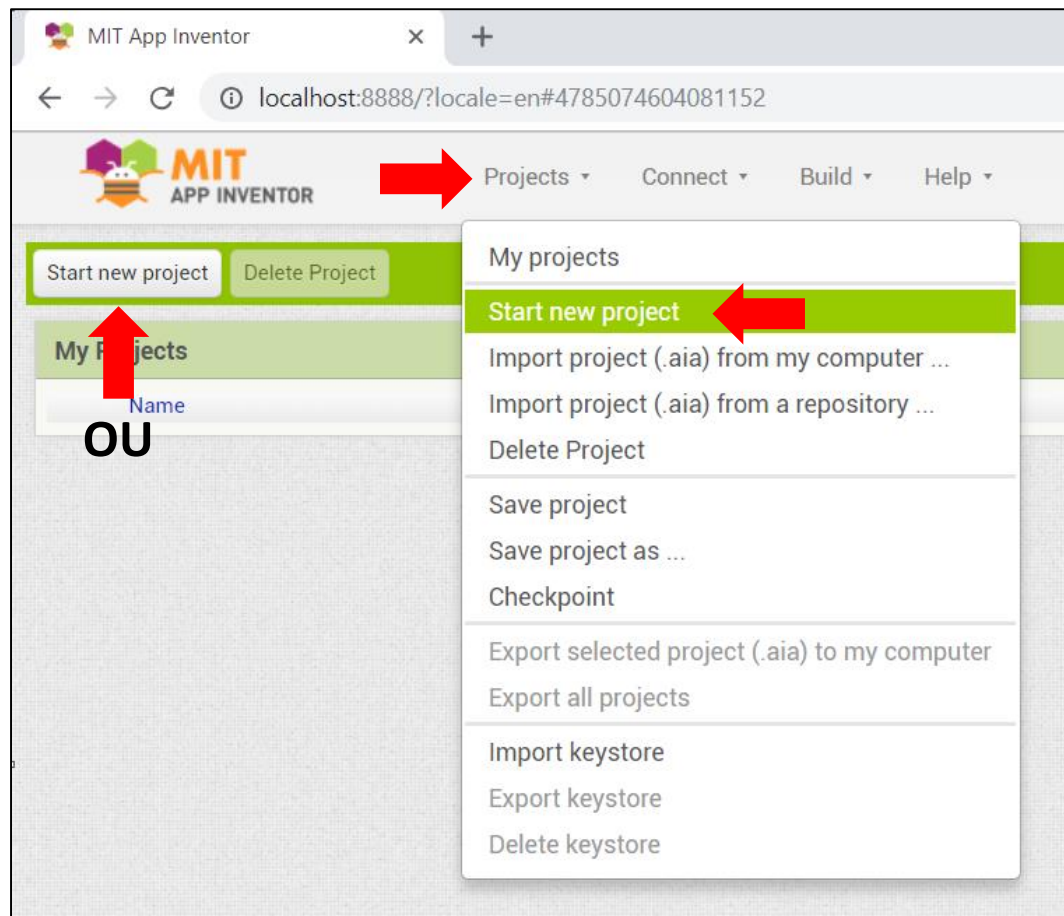
Procedures: Os blocos dessa categoria **criam e executam** procedimentos. Um procedimento é uma rotina (mini programa) que pode ser executada para realizar uma tarefa.

Seu Primeiro App: Olá Mundo!



Para ilustrar os conceitos aprendidos, vamos criar um **App simples**. Basicamente teremos um **botão** que **quando o usuário clicar** será apresentado a seguinte mensagem: **“Olá Mundo!”**. Isso não fará de você um especialista em App, porém é o **início da aprendizagem**.

Crie um Novo Projeto

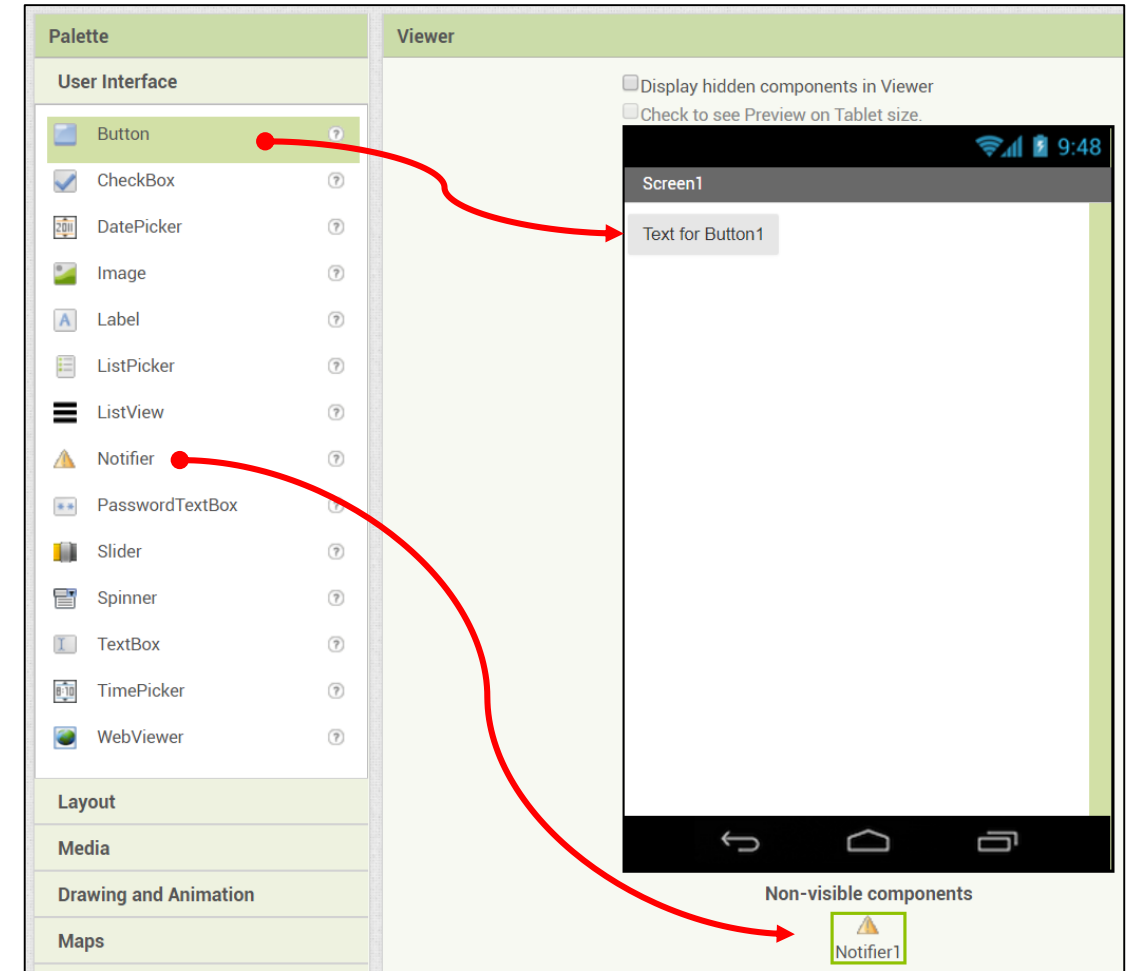


Atenção:

Não é permitido **espaços** e nem **caracteres especiais** em **nomes de projetos**. Isso também vale para **nomes de componentes**. Um modo de separar palavras é utilizar a convenção ***Camel Case***.

Adicionando os Componentes

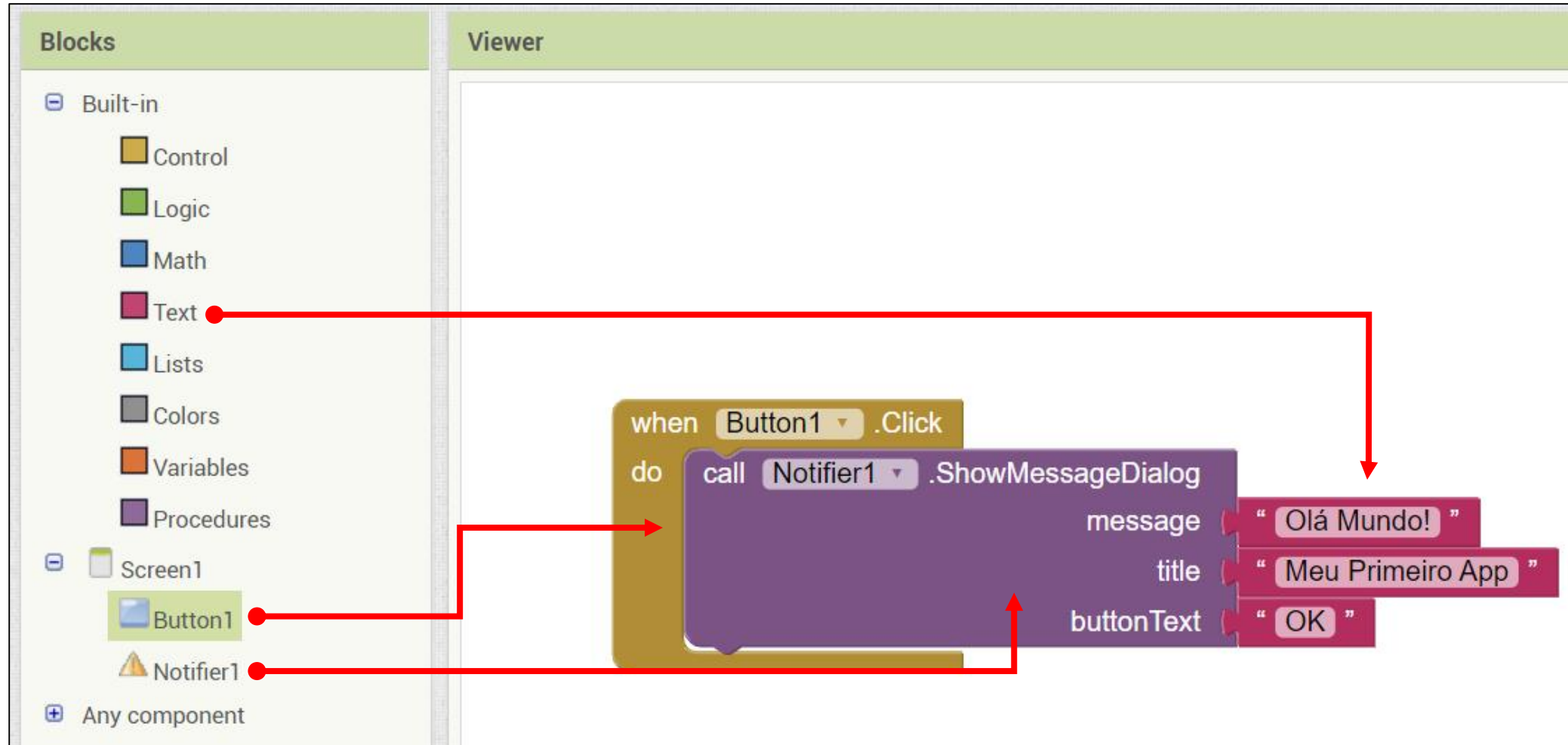
- ☐ Usaremos apenas dois componentes: o **Button** e o **Notifier**;
- ☐ O **Button** como o próprio nome sugere é um **botão**;
- ☐ O **Notifier** é utilizado para qualquer tipo de **mensagem pop-up e avisos**.



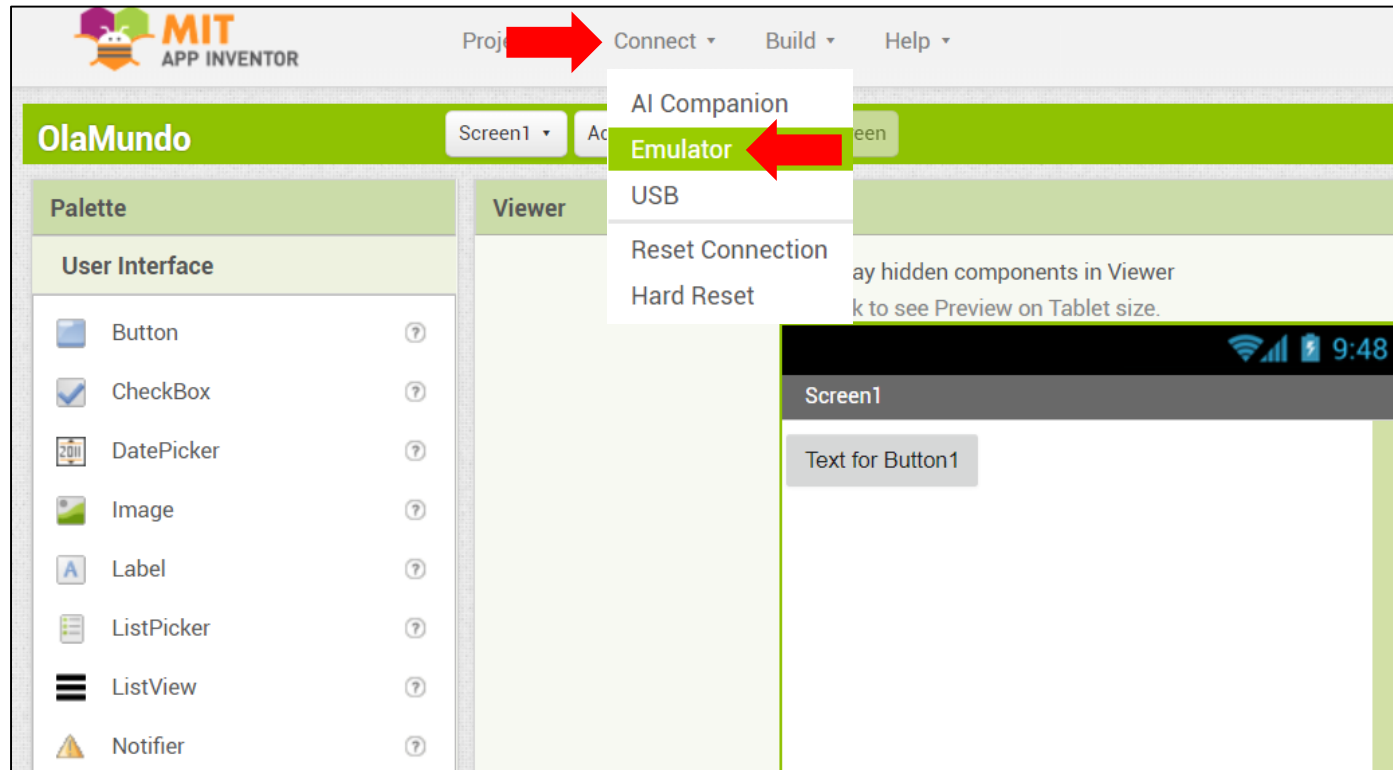
Como Detectar o Click do Botão?

- ❑ Normalmente os **Apps não fazem nada** até que o **usuário realize uma ação** (clique no botão);
- ❑ Toda **ação realizada** é chamada de **evento** e existem **blocos capazes de detectá-los** (*event handlers*);
- ❑ Um ***event handler*** é reconhecido pela sua estrutura que possui as palavras **when** e **do**;
- ❑ O **when** refere-se ao **componente e ao seu evento** (gatilho), já o **do** diz respeito a **ação a ser realizada**.

Escrevendo a Lógica do Programa



Testando o App

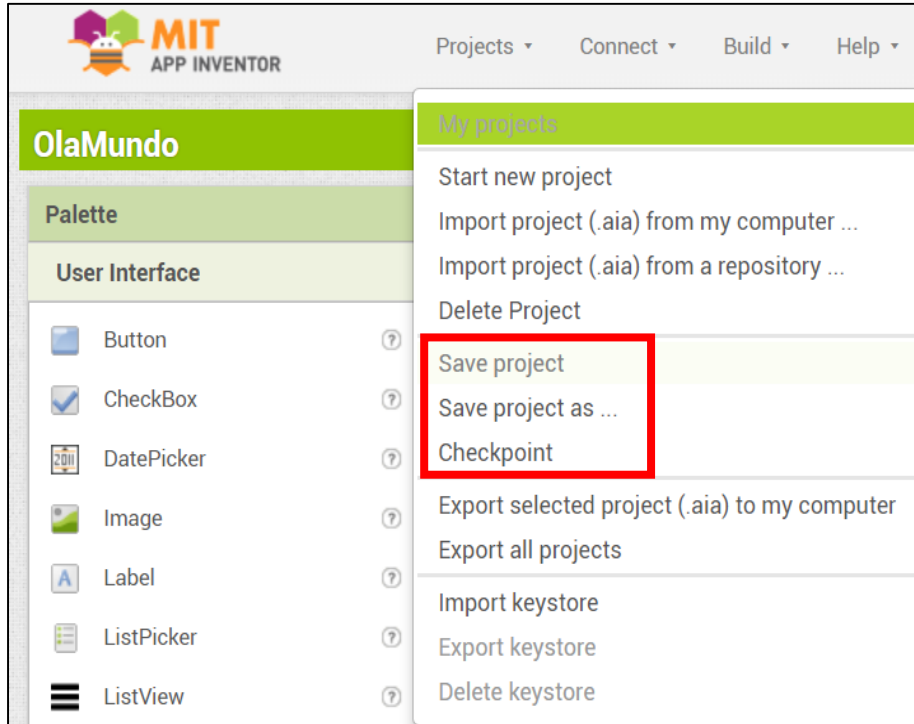


Atenção:
Antes de **destravar o emulador**
espere ele terminar de **preparar o**
SD Card.

Como salvar o Projeto?

- ☐ App Inventor **salva automaticamente** os projetos na medida em que você trabalha neles;
- ☐ No entanto, os projetos são **salvos no servidor**, ou seja, na nuvem. **Mas e se o servidor parar?**
- ☐ Para resolver esse problema o App Inventor **permite realizar cópias do projeto** em seu computador;
- ☐ Também é possível criar **pontos de restauração** caso queira voltar a uma **versão anterior**.

Modos de Salvar o Projeto



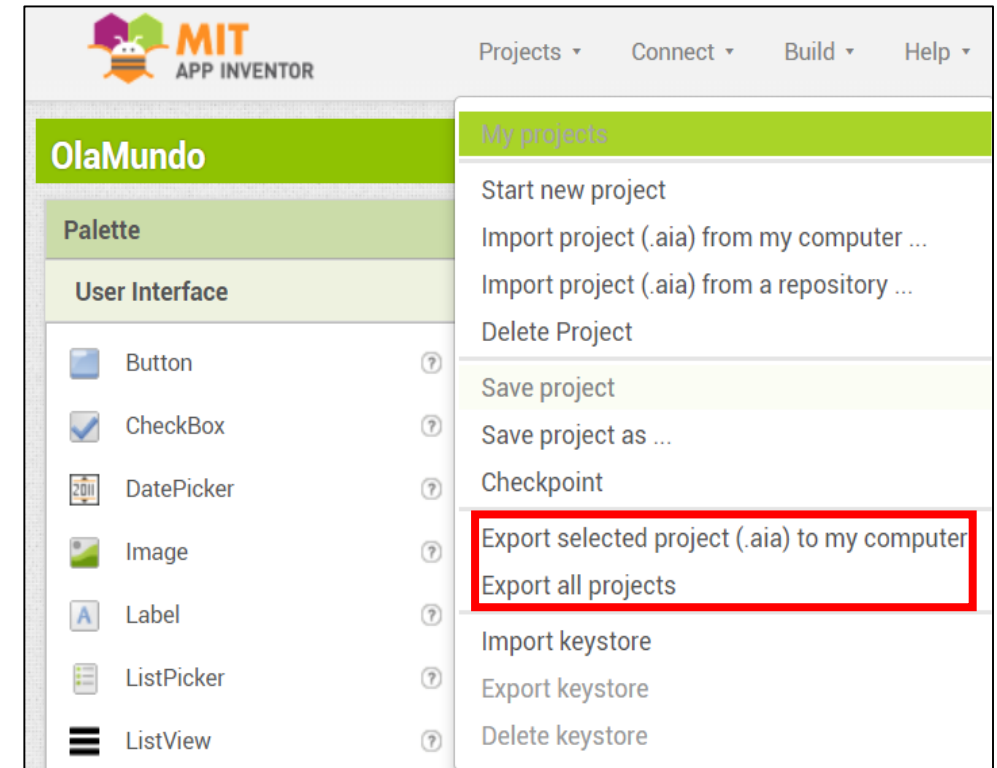
Save Project: é o mesmo trabalho que o App Inventor já faz automaticamente. Mas se mesmo assim você não confiar, ele permite que você salve manualmente.

Save Project As: cria uma cópia do projeto com um novo nome. O projeto antigo permanece disponível. Esse recurso é muito utilizado para criar versões diferentes.

Checkpoint: cria pontos de restauração no projeto que permite voltar ao seu estado anterior. Se for fazer alguma coisa complicada, crie um checkpoint.

Cópia Local (Backup)

- ❑ O App Inventor permite exportar **um projeto** selecionado ou **todos os projetos** existentes;
- ❑ Uma **cópia dos projetos** ficará em sua **máquina** facilitando a **importação** quando necessária;
- ❑ O resultado da exportação é um arquivo com o **formato (.aia)**.



Dúvidas?



Vamos Trabalhar!

Enviar a lista por e-mail conforme as regras apresentadas na aula 1. Entregar ainda hoje!



Programação para Dispositivos Móveis

Curso de Ciência da Computação

Universidade Paulista (UNIP)

INTRODUÇÃO À DISCIPLINA E CONCEITOS INICIAIS

Todos os Créditos para Nathan Cirillo e Silva
nathan.silva@docente.unip.br

