Programação para Dispositivos Móveis

Curso de Ciência da Computação

Universidade Paulista (UNIP)

DISPOSIÇÃO DOS ELEMENTOS NA TELA:
APRENDENDO A USAR LAYOUTS E CANVAS

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

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

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

AULA 3

Introdução

- **Posicionar componentes na tela** pode ser uma tarefa difícil devido aos **variados tamanhos e formatos** de dispositivos;
- Além disso, o usuário pode **inclinar o celular**, mudando a aparência de **retrato para paisagem**;
- Na aula anterior, aprendemos a utilizar **um dos componentes** que nos ajuda no posicionamento: o ***VerticalArrangement***;
- Portanto, é **essencial conhecermos** outras formas de **organizar componentes em Grid**.

O que é Layout?

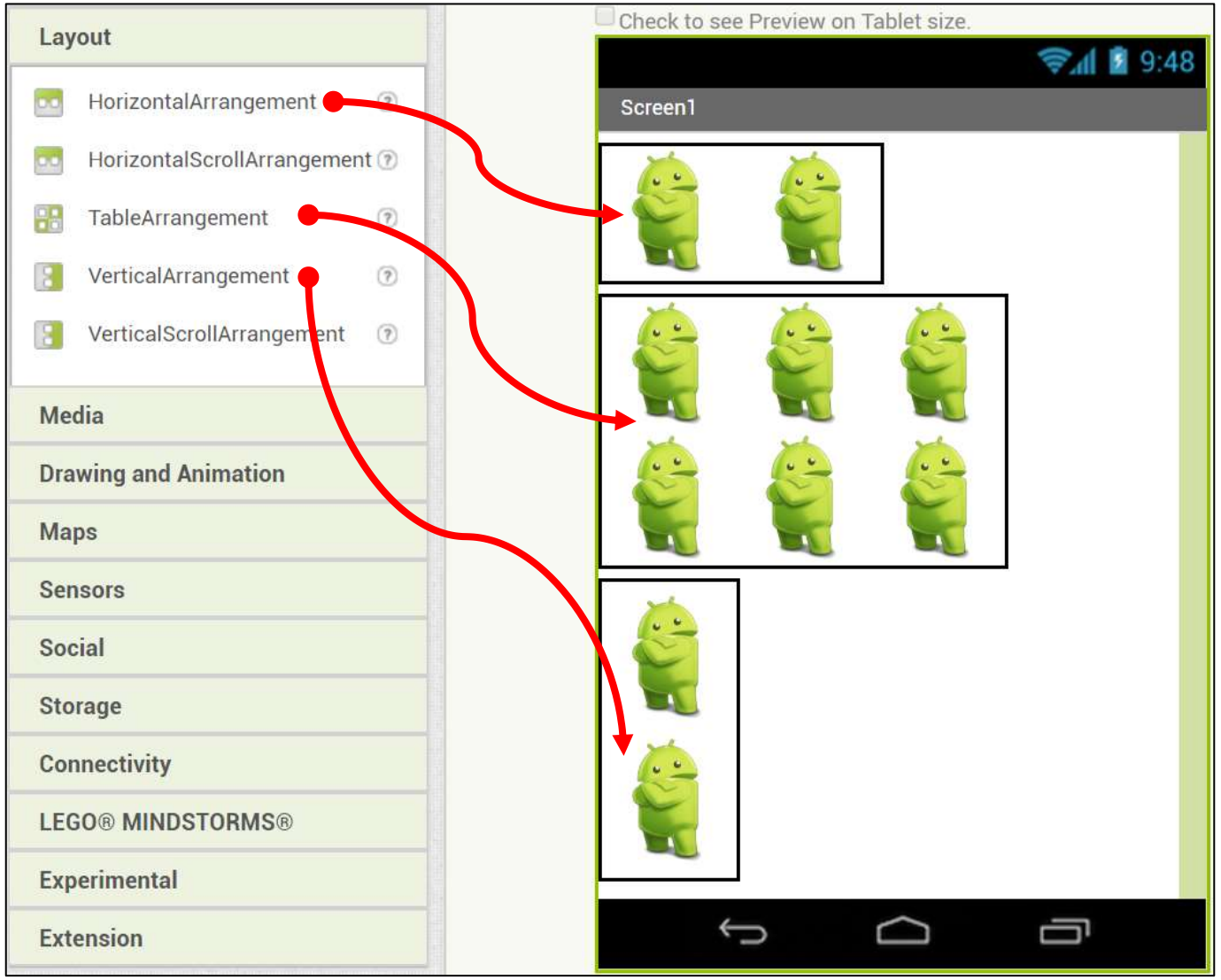
- Layout é a **forma como os componentes estão organizados e apresentados** na tela do usuário;
- Essa **organização** ocorre através de uma **“tabela”** também chamada de **grid** composta por linhas e colunas;
- O **comportamento padrão** do App Inventor é **empilhar os objetos um abaixo do outro**;
- Na forma padrão **conseguimos alterar a ordem dos objetos**, mas e se quisermos que eles fiquem **um do lado do outro**?.

Opções de Layout

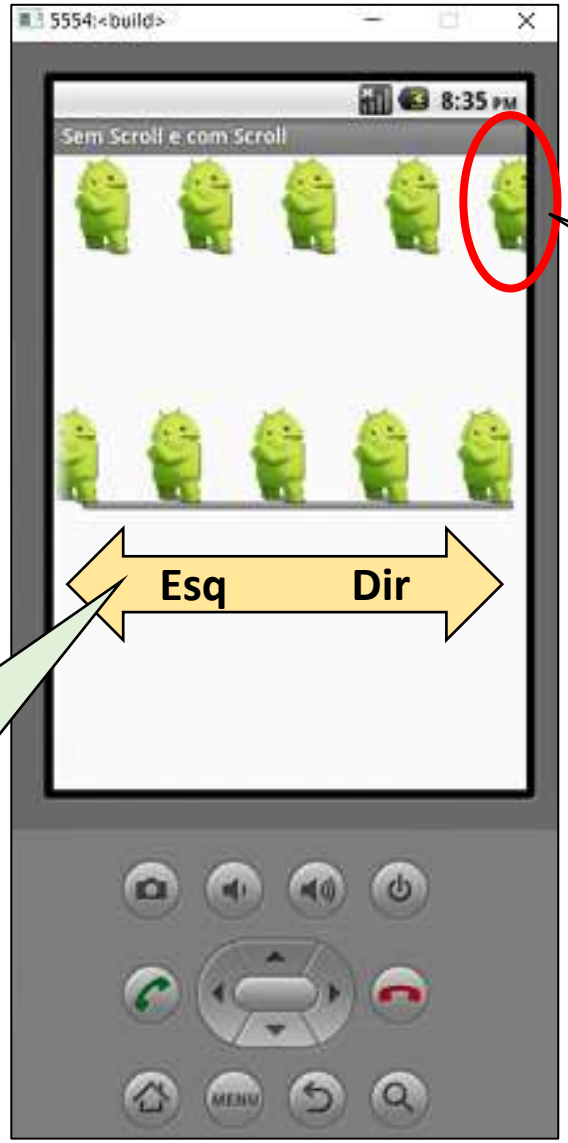
- **VerticalArrangement:** permite que os componentes sejam colocados um em cima do outro (default);
- **HorizontalArrangement:** utilizado para posicionar os componentes da esquerda para a direita (lado-a-lado);
- **TableArrangement:** possibilita a organização dos elementos de maneira tabular (linhas e colunas).

Tanto para o **layout vertical como para o horizontal** existem as **opções com scroll**. Caso os **objetos não caibam na tela** será adicionado uma **barra de rolagem** que permite visualizá-los.

Veja uma Ilustração



Mas o que é o tal do Scroll?



Com Scroll:

A barra de rolagem favorece a **visualização de todos os objetos** através da **movimentação para ambos os lados**.

Sem Scroll:

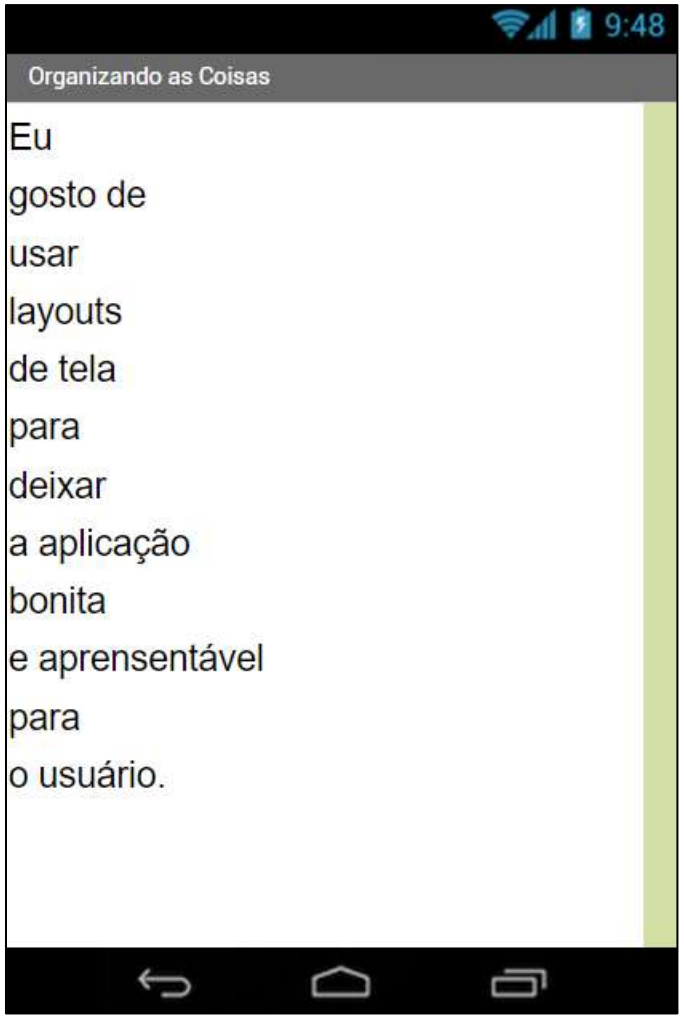
Quando não cabem na tela os objetos são **cortados ou não aparecem**.

Dica sobre Posicionamento

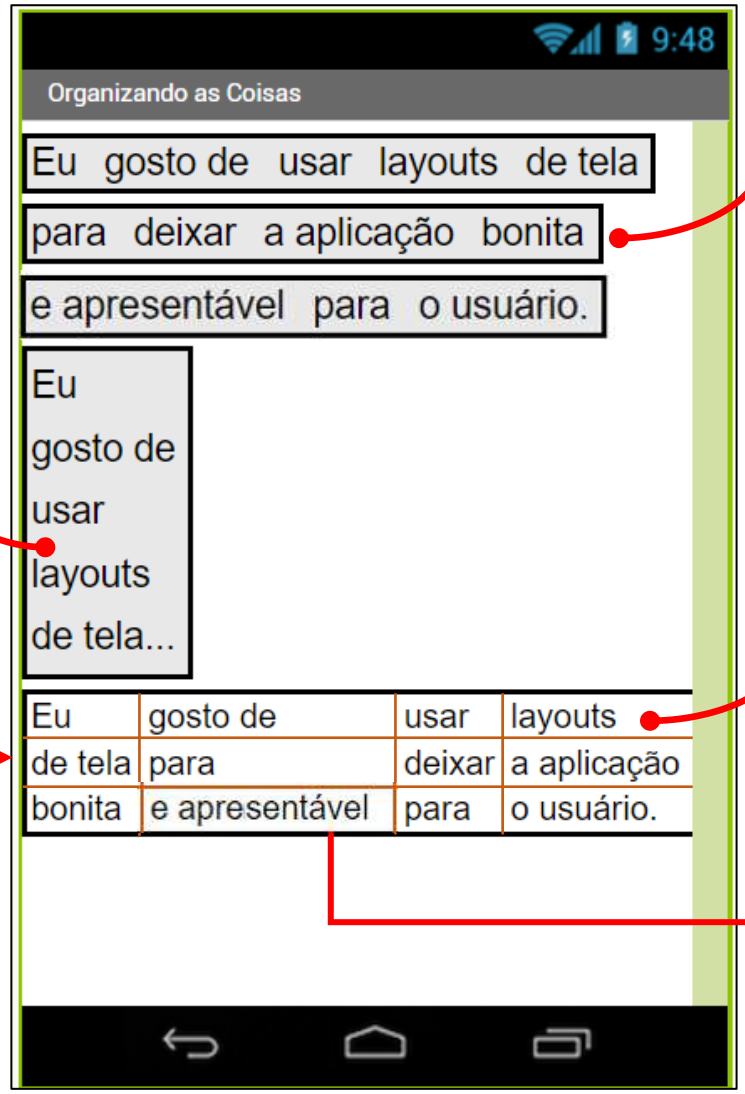
Quando você **arrasta um objeto** para dentro de um grid, note que aparecerá uma **linha azul grossa** indicando o **local onde ficará o objeto** quando você **soltar o botão do mouse**.

Vamos a um Exemplo Prático!

Imagine que você queira **organizar a sentença** ao lado usando ***HorizontalArrangement***, ***VerticalArrangement*** e ***TableArrangement***. Note que **todos os objetos são *labels***. Isso indica que **não importa o tipo de objeto que estamos trabalhando**, pois os **layouts sempre funcionam**.



Veja Alguns Layouts Possíveis



HorizontalArrangement

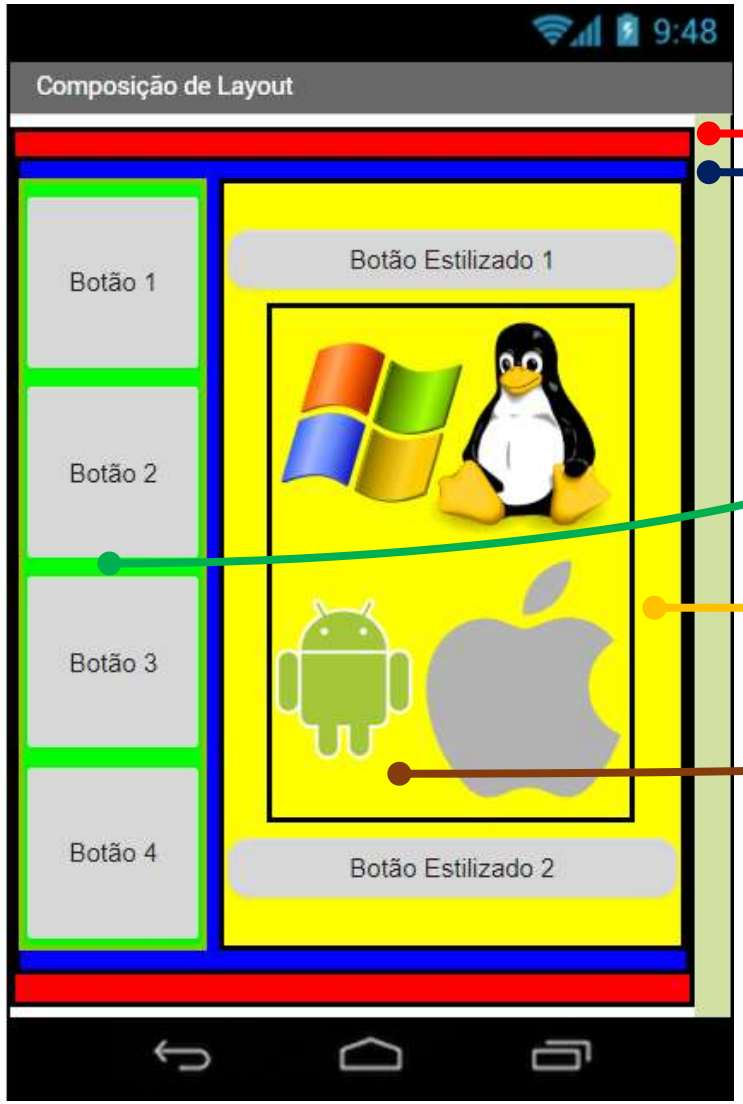
VerticalArrangement

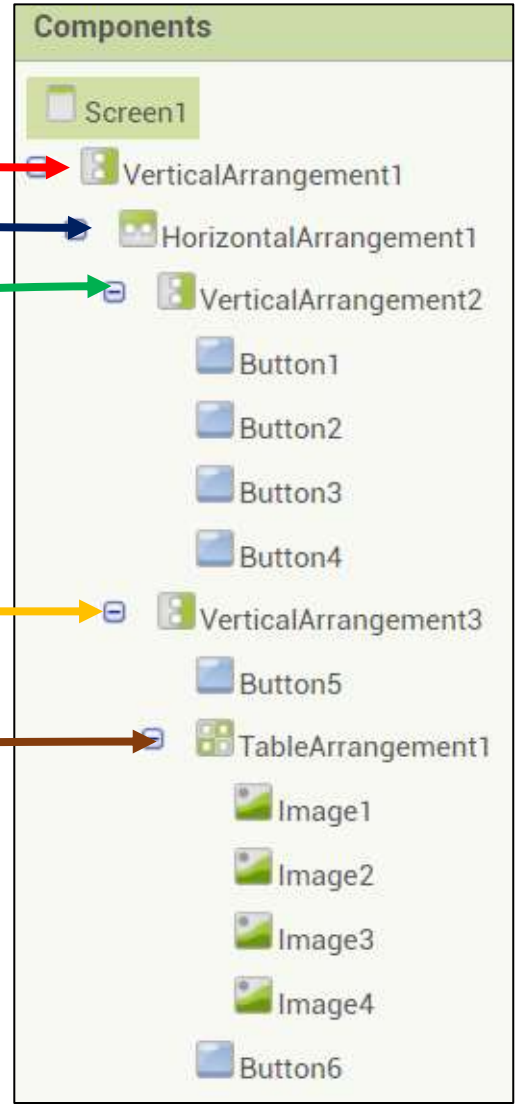
TableArrangement

Necessário definir o **número de linhas (rows)** e **colunas (columns)**.

3 x 4

Composição de Layouts





Será utilizado **vários layouts** para **compor o design** do seu App. Permite uma maior **flexibilidade no posicionamento**!

Vamos a um Exemplo Prático!

Propósito do App:

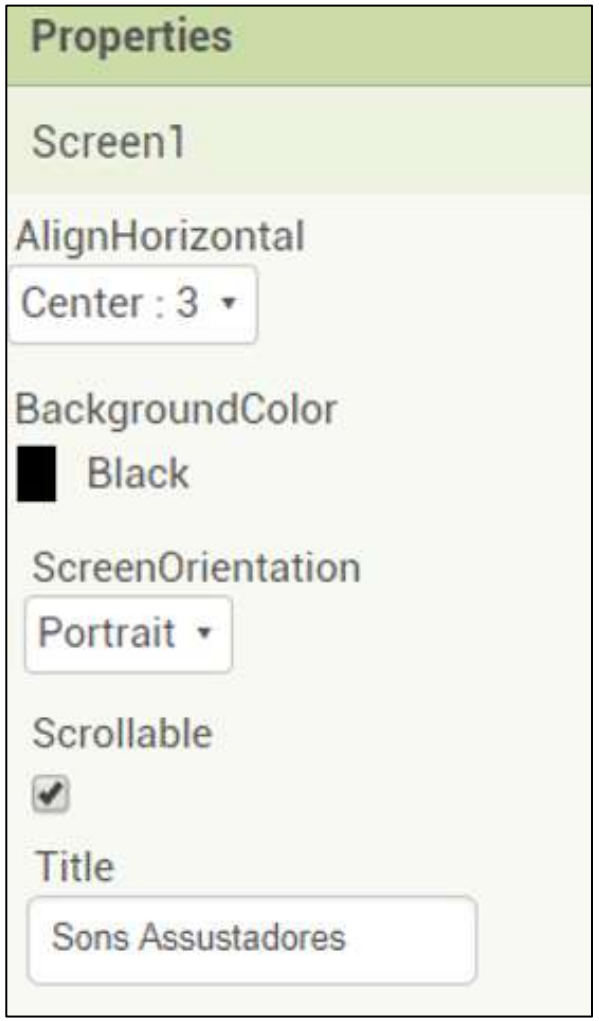
Vamos **criar um app** que executará **diversos sons assustadores**. Para isso, vamos utilizar um layout diferenciado através do ***TableArrangement***.

Recursos Necessários:

Oito sons diferenciados que você queira acrescentar no App.

Passo 1: Configurando as Propriedades de Tela

- Se o usuário girar o celular, o App deve permanecer na **orientação retrato**;
- **Ative o scroll** para garantir que todos os componentes da tela sejam visíveis;
- Ajuste o **alinhamento horizontal para centralizado** e a **cor de plano de fundo para preto**.

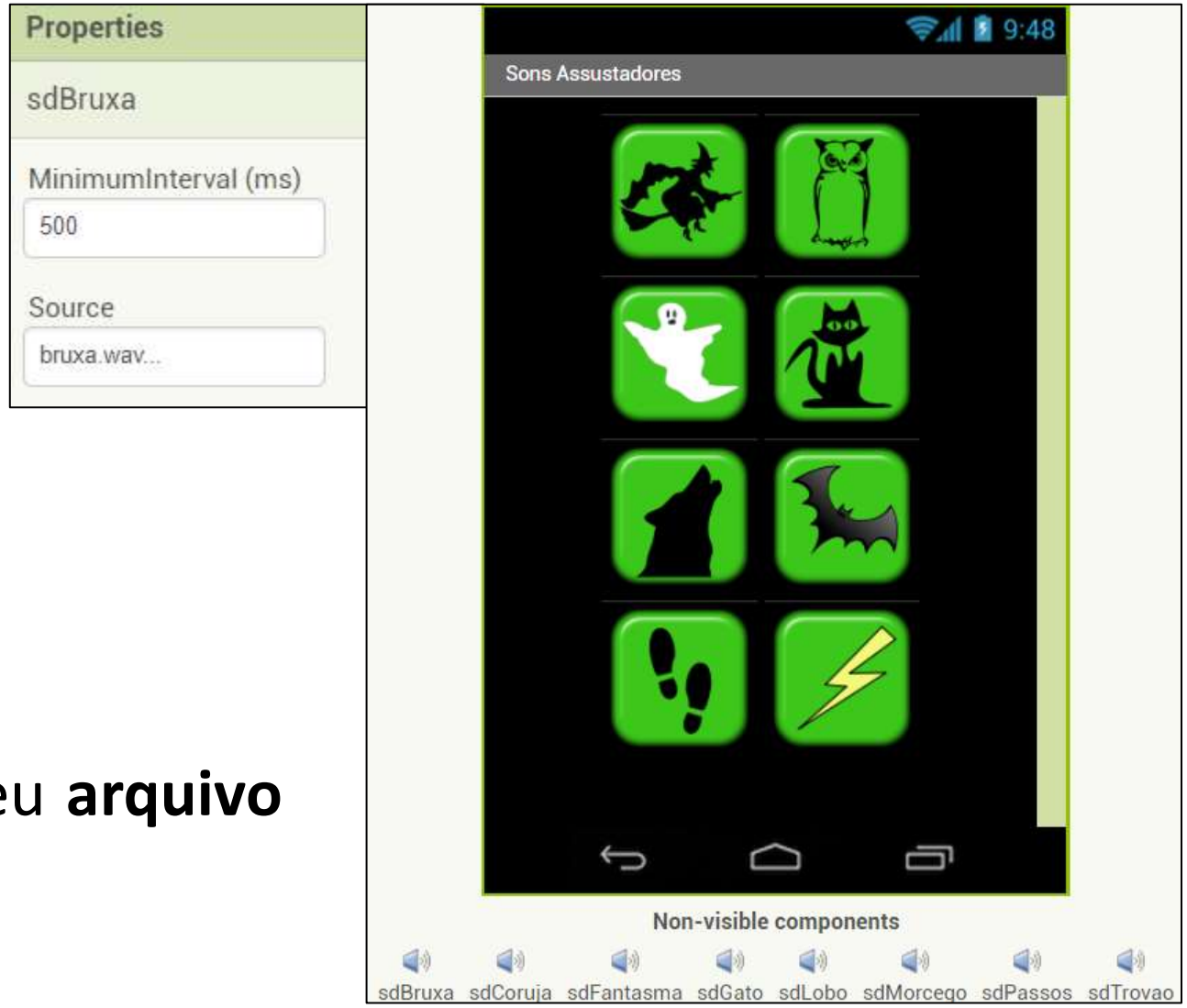


Passo 2: Adicionando o *TableArrangement*

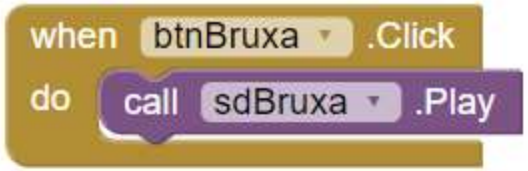
- Adicione o ***TableArrangement*** e configure a sua propriedade **Columns para 2** e **Rows para 4**;
- Acrescente **oito botões** nos espaços definidos, lembrando que trata-se de um **Grid de 2 x 4**;
- Apague o **Text dos botões**, defina a sua **largura e altura para 90 px** e **acrescente as imagens**.

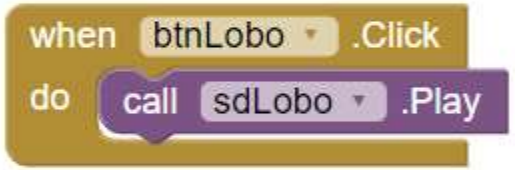
Passo 3: Adicione os Componentes de Som

- Adicione **oito componentes** de som ***"Sound"*** localizados na seção *Media*;
- **Renomeie** cada componente de som para um **nome característico**;
- Para cada um deles informe o seu **arquivo de áudio**, propriedade ***"Source"***.



Passo 4: Programe os Blocos



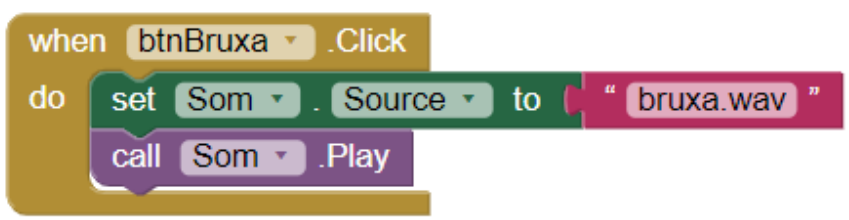


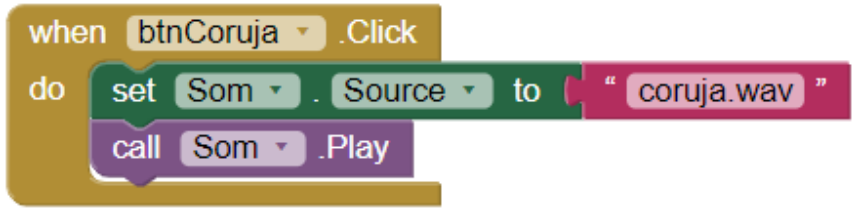
O que vocês acham?

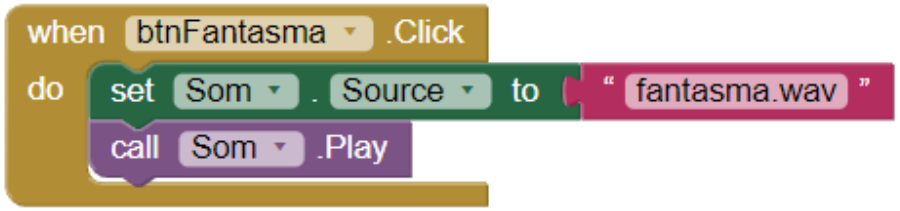
É necessário ter um componente de áudio para cada som?!

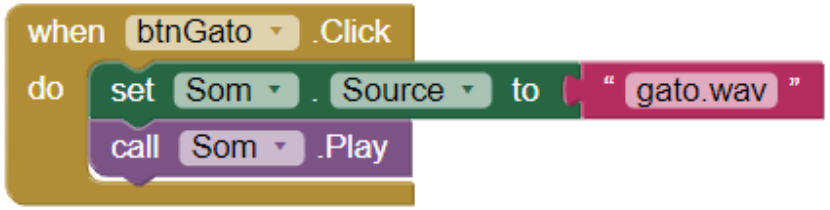
Não! Podemos ter um **único componente de som** e alterar a sua **propriedade *source*** em **tempo de execução**.

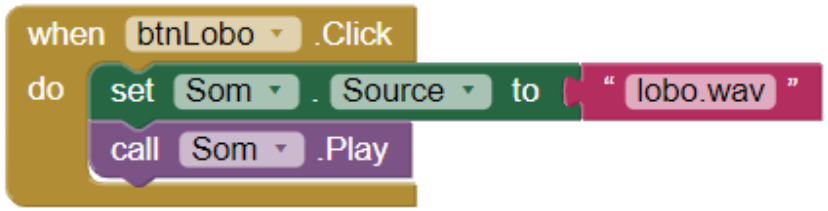
Veja como Ficaria!

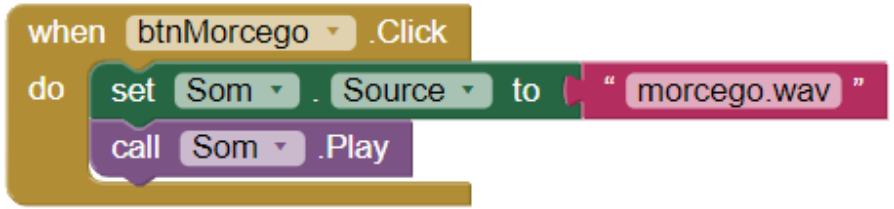


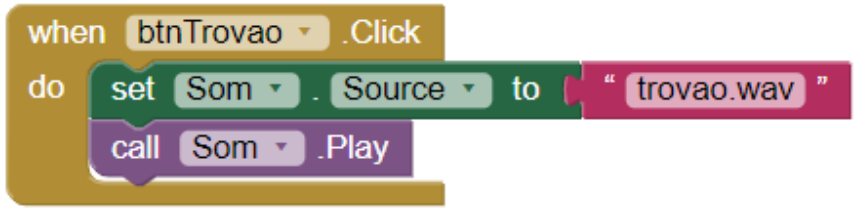












É necessário **já ter carregado** os **arquivos de som** na seção **Media**!

Conhecendo o **Componente Canvas**

- **Canvas** é um componente que fornece **recursos gráficos mais avançados** como **animações e interações**;
- A **tela é um container para os componentes** que o usuário utiliza para interagir com o App;
- Já o **Canvas além de ser um container**, possibilita também **identificar eventos ocorridos**;
- Trata-se portanto de um **painel retangular sensível** que detecta: **toques**, **arrastes** e **arremessos**.

Quando utilizar o Canvas?

- Imagine que você queira **desenhar no seu App**, o Canvas é a solução!;
- **Imagens animadas podem se mover** sobre o Canvas como em um Jogo;
- Portanto, **se você quer animar o seu App**, o Canvas deve ser utilizado.

Vamos a um Exemplo Prático!

Propósito do App:

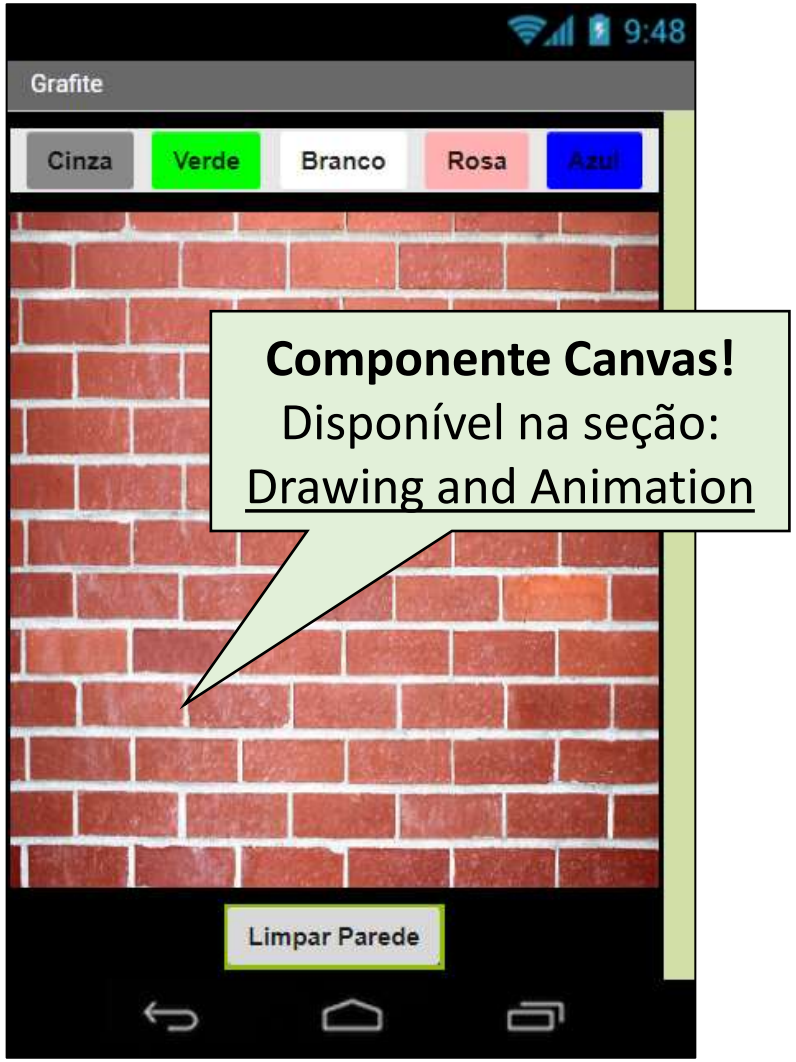
Quando o usuário **deslizar o dedo (ou o mouse) sobre a parede**, **linhas serão desenhadas**. Pode também apenas **tocar na tela para criar pontos**. As **cores são escolhidas utilizando botões**. Há também um **botão que limpa a tela (grafite)**.

Recursos Necessários:

Imagem de uma parede.

Passo 1: Criar o Layout do App

- É interessante que o **componente Canvas ocupe a maior parte possível da tela**;
- Os **botões de cores devem ficar posicionados acima** e separados do Canvas (*HorizontalArrangement*);
- O **botão para limpar o grafite** também deve estar **separado do Canvas na parte inferior**.



Propriedades Utilizadas no App
(Continua)

Screen 1	AlignHorizontal: Center AlignVertical: Center Backgroundcolor: None ScreenOrientation: Portrait Title: Grafite

Componentes	Identificação	Propriedades Utilizadas
HorizontalArrangement	BarraCores	AlignHorizontal: Center Width: Fill Parent Height: Automatic
Button	btnCinza btnVerde btnBranco btnRosa btnAzul	Backgroundcolor: Igual ao nome Text: Igual ao nome

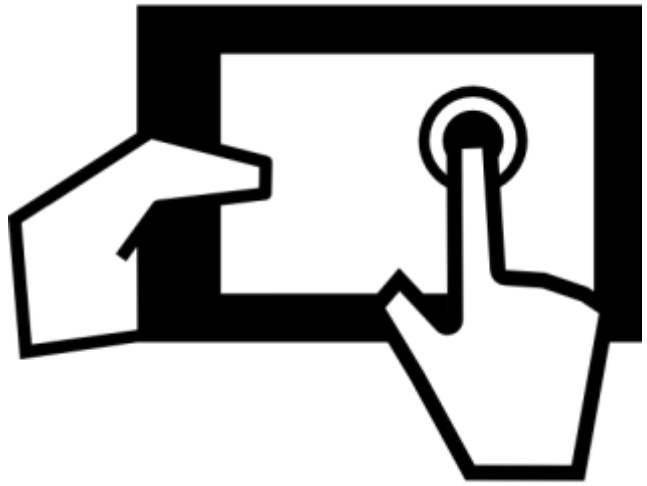
Propriedades Utilizadas no App

Componentes	Identificação	Propriedades Utilizadas
Canvas	ParedeCanvas	BackgroundImage: Wall.png Width: Fill Parent Height: Fill Parent
Button	btnLimpa	Text: Limpar Parede

Passo 2: Programando os Blocos

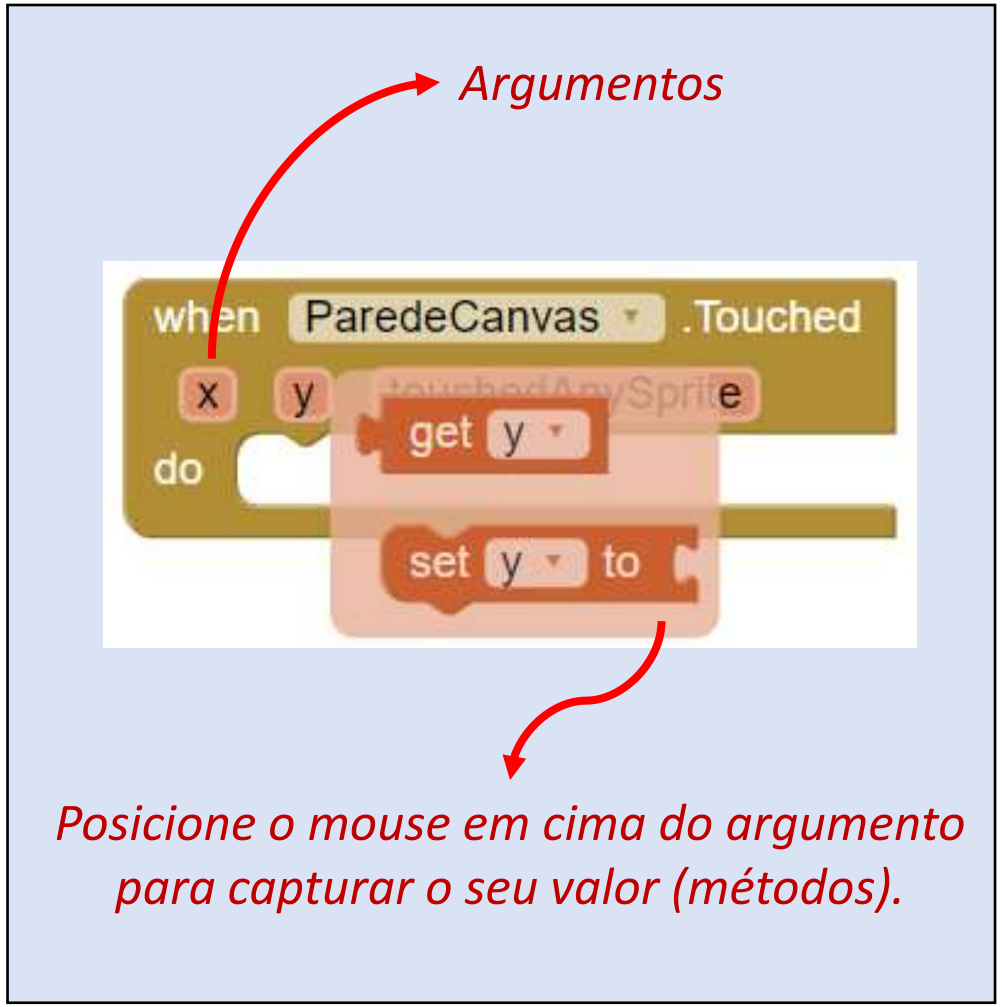
Antes de programar o nosso App é necessário conhecer os principais eventos que podem ser realizados pelo usuário:

- ❑ Criar um único ponto clicando em um local da parede;
- ❑ Desenhar uma linha deslizando o dedo sobre a parede;
- ❑ Trocar a cor clicando em um dos botões de cor;
- ❑ Limpar a parede clicando no botão limpar.

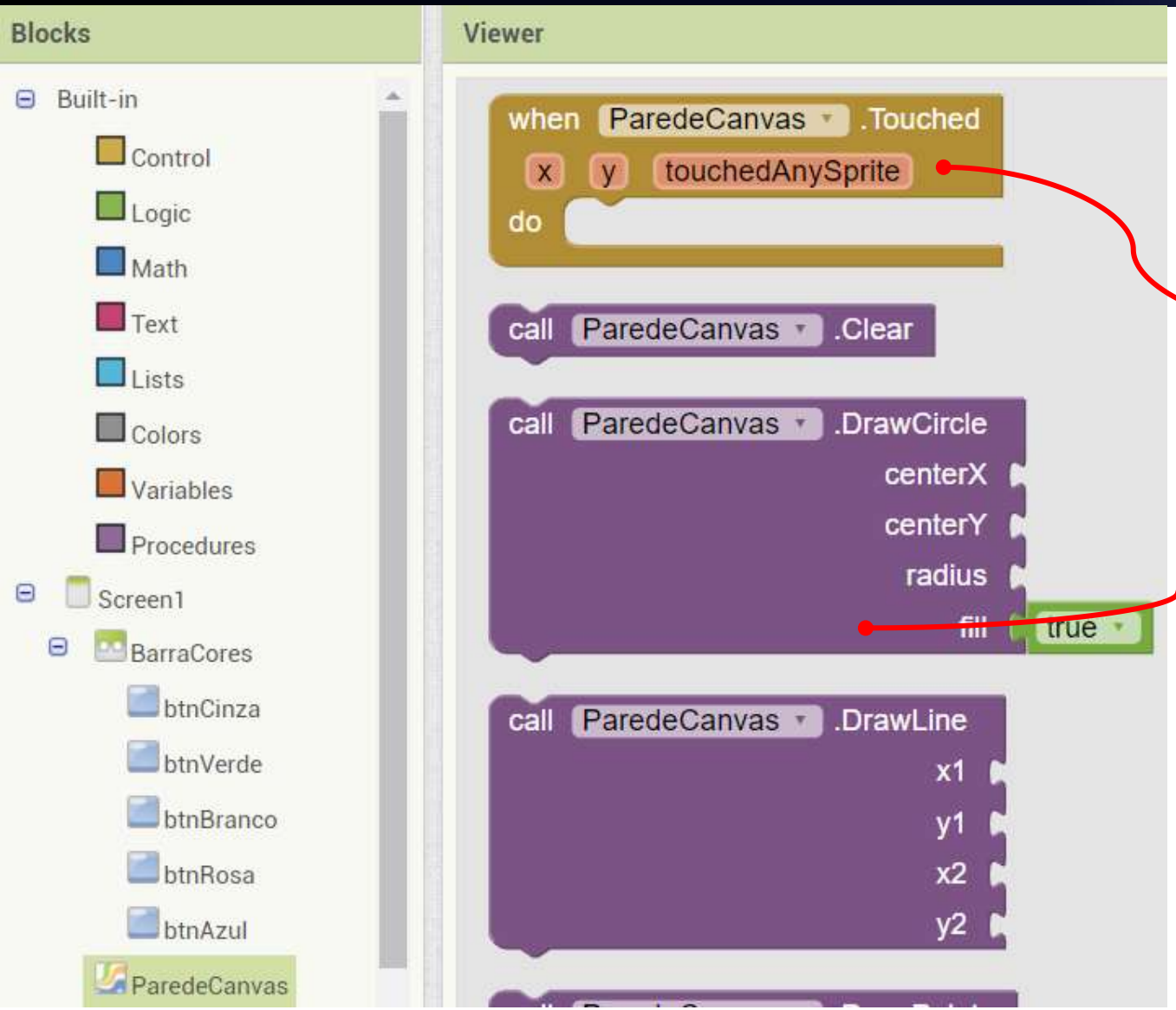


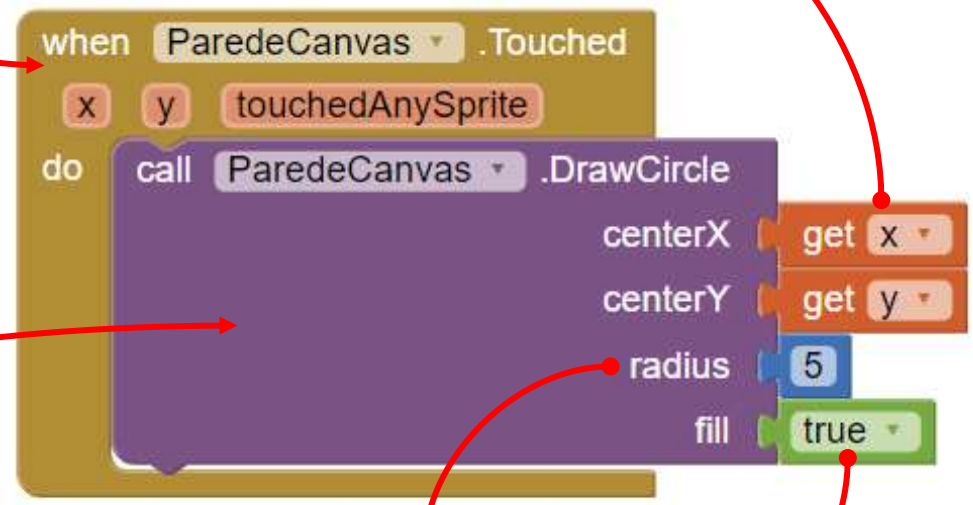
Desenhando **Pontos na Tela**
(Continua)

- Isso é possível através do **evento** chamado **Touched** **do objeto Canvas** que indica o **local tocado**;
- Esse evento traz as **coordenadas do local**, ou seja, **informações extras** chamadas de **argumentos**;
- É o **conhecimento dessas coordenadas** (argumentos) que permite **criar um ponto no local desejado**.



Desenhando **Pontos na Tela**





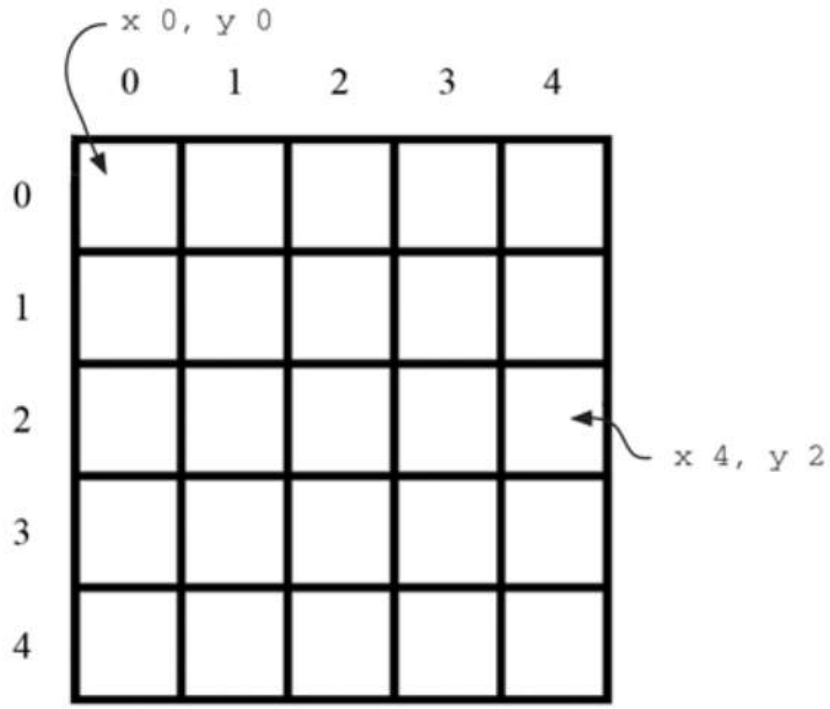
Captura a localização

Raio do Círculo

Círculo Preenchido

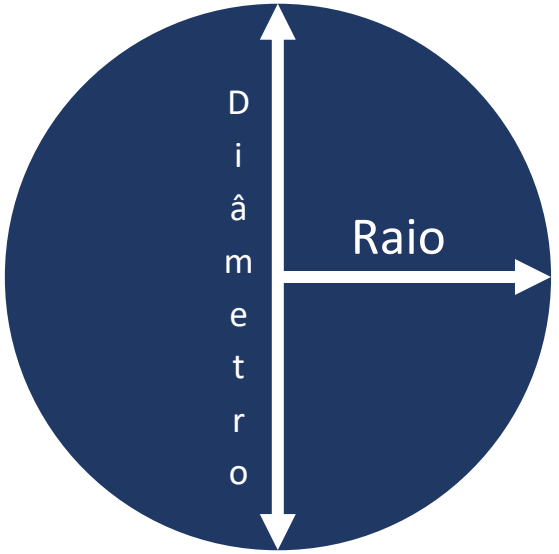
Sobre as Coordenadas X e Y

- Canvas é uma **tabela de pixels** e um pixel é o **menor ponto da tela**;
- Os **pixels são localizados** no canvas através de **coordenadas X, Y**;
- O **X** indica a localização do pixel da **esquerda para direita** (horizontal);
- Já o **Y** indica a localização do pixel de **cima para baixo** (vertical).



Sobre o Raio do Círculo (Ponto)

- O **raio** de um círculo é a **distância do centro desse círculo até a sua borda**;
- O **diâmetro**, por sua vez, é a **distância entre bordas**. Portanto, d = r * 2;
- O canvas utiliza o valor do raio para **determinar o diâmetro** do ponto;
- **Quanto maior o raio**, maior será o **diâmetro e a espessura** do ponto.

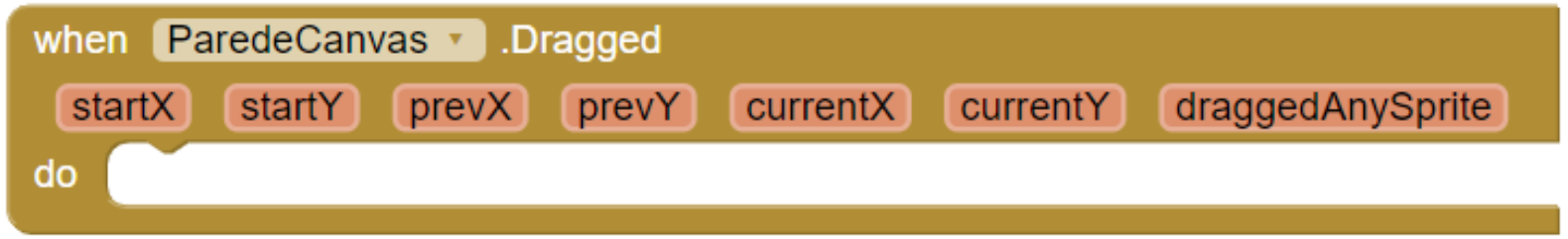


Desenhando Linhas na Tela

- Ao **deslizar o dedo na tela** o usuário está criando um **monte de linhas pequenas e retas**;

- O **evento que é disparado** pelo canvas com essa ação chama-se **Dragged** (arrastar);

- Esse evento é **disparado constantemente enquanto o dedo é movimentado sobre a tela**.

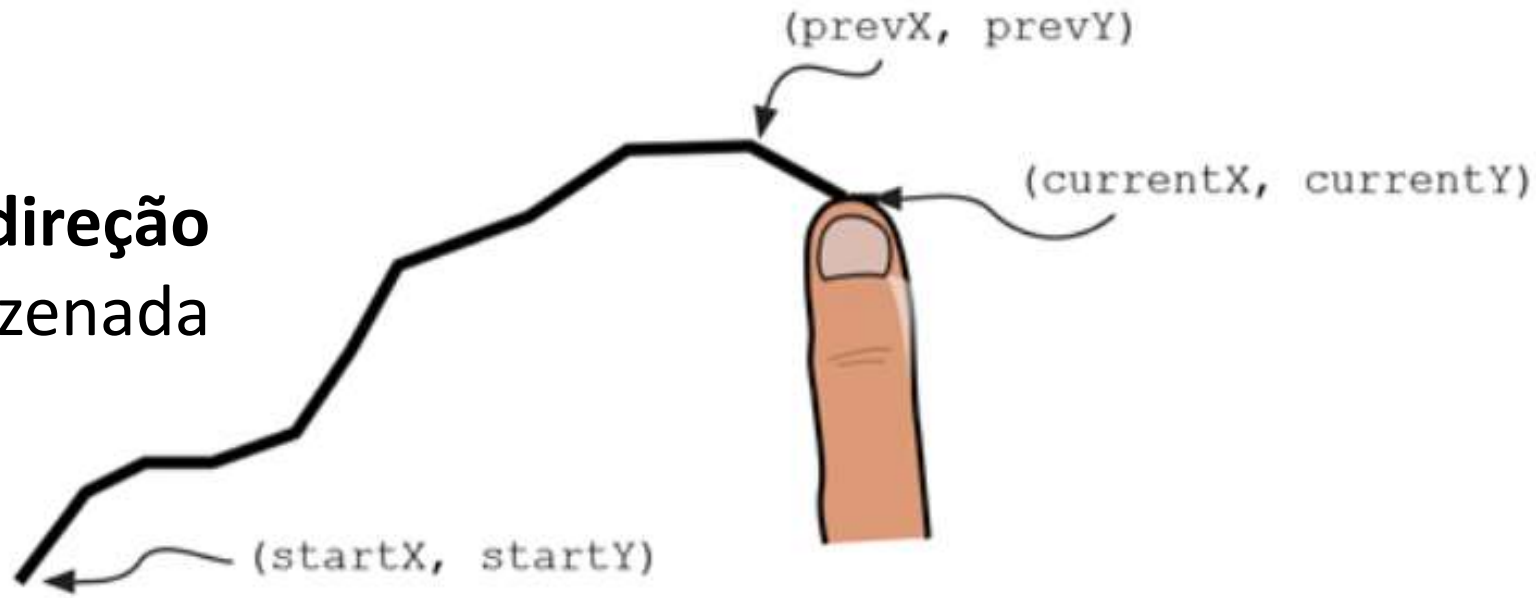


Argumentos do Evento Dragged

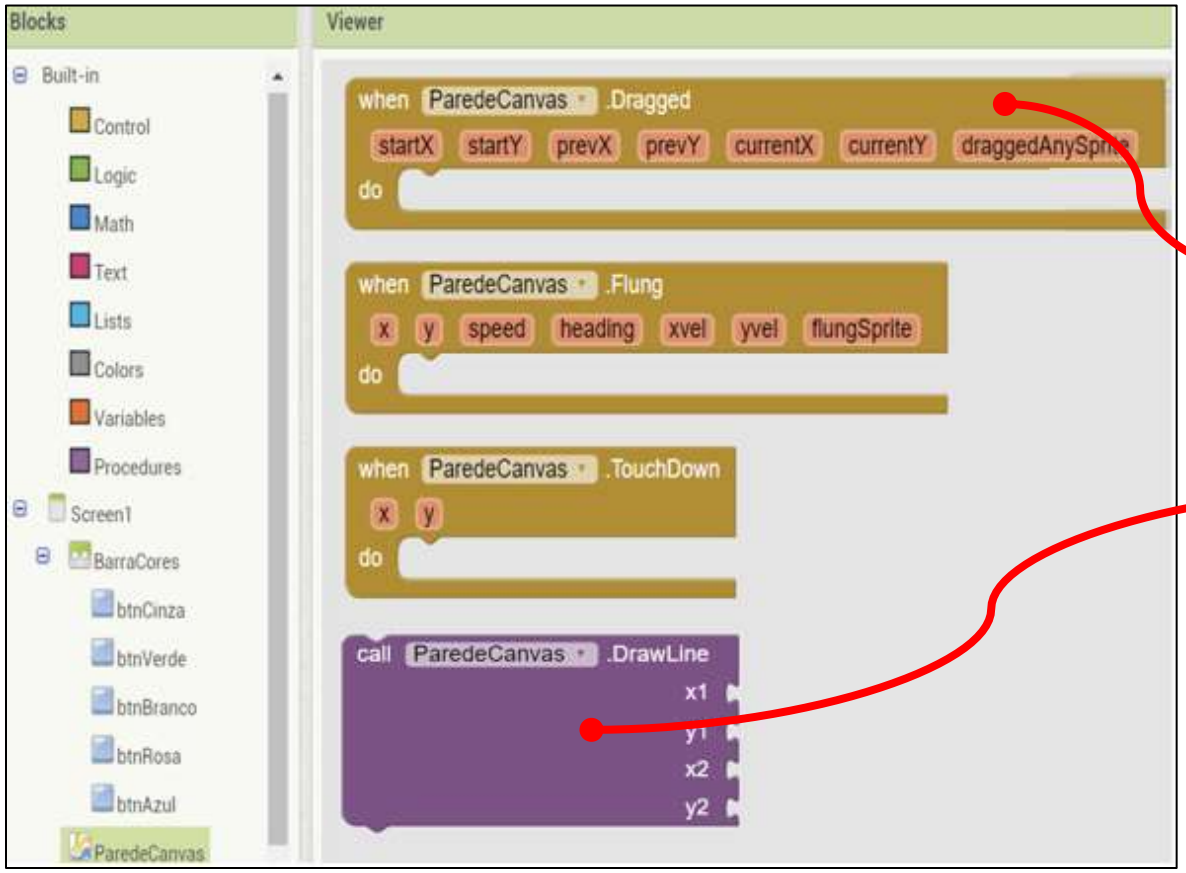
- Os argumentos **startX** e **startY** indicam a **posição inicial** em que a linha **começou a ser traçada**;

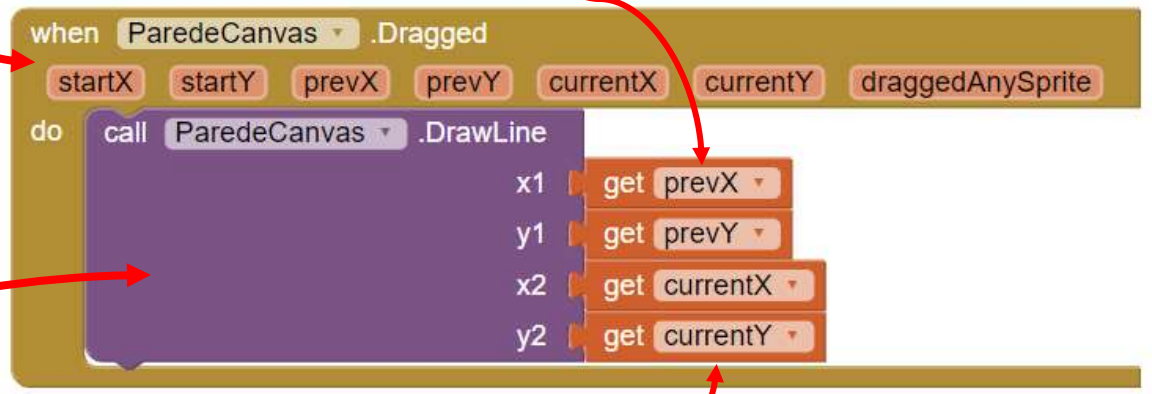
- A **posição atual dos dedos na tela** é sempre armazenada em **currentX** e **currentY**;

- Ao **parar ou mudar de direção** essa posição é armazenada em **prevX** e **prevY**.



Desenhando Linhas na Tela



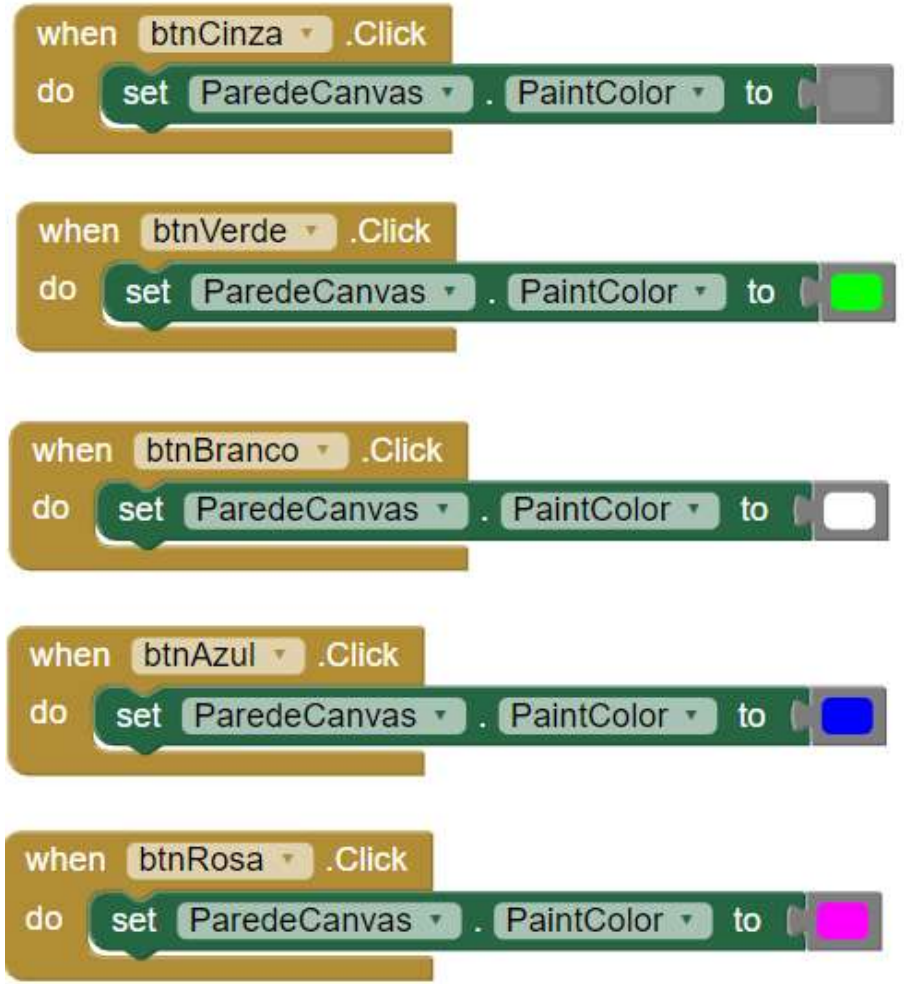


Posição Anterior

Posição Atual

Programando os Botões de Cores

- Para cada um dos **botões de cores** adicione o ***event handler*** **Click**;
- Utilize o **bloco set** para alterar a propriedade **PaintColor do Canvas**;
- Escolha a **cor correspondente** na paleta de cores disponíveis.



Limpando a Parede

- Após utilizar o App a **parede estará toda suja** e será **necessário limpá-la**;
- Para fazer isso é **muito fácil**, basta **chamar a função Canvas.Clear**;
- Essa **chamada** deverá ser feita quando o usuário **clicar no botão limpar**.

Funcionamento Esperado

Dúvidas?

Vamos Trabalhar!

Enviar a lista via Teams conforme indicado pelo professor.
Entregar no prazo combinado!