Programação para Dispositivos Móveis

Curso de Ciência da Computação

Universidade Paulista (UNIP)

DESENHO E ANIMAÇÃO

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

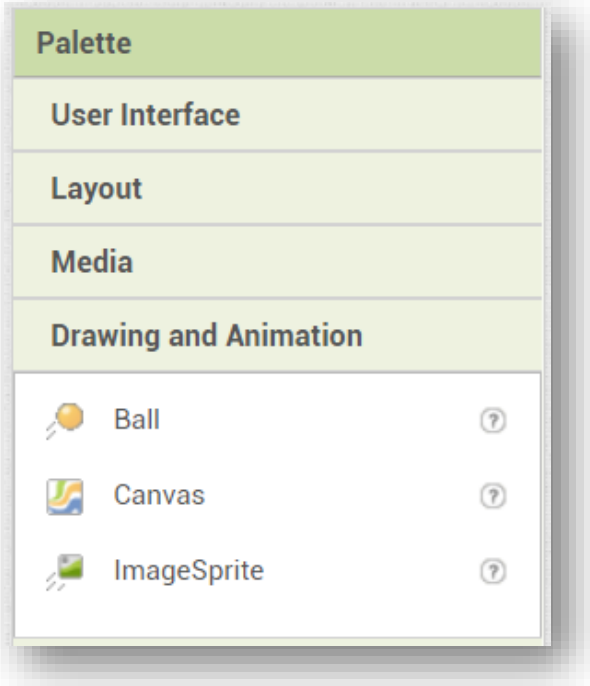
Prof. Ms. Peter Jandl
peter.junior@docente.unip.br

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

AULA 4

A Importância do *Touch Screen*

- O **legal dessa tecnologia** é que ela permite a **interação** com o smartphone de **diferentes maneiras**;
- As **três principais formas** de interação são: **o toque**, **o arraste** e **o arremesso**;
- Como você já sabe, o **App Inventor tem muitos componentes** com **funções específicas**;
- Os componentes que **permitem essas interações** estão na seção: **"*Drawing and Animation*"**.



Eventos de Interação

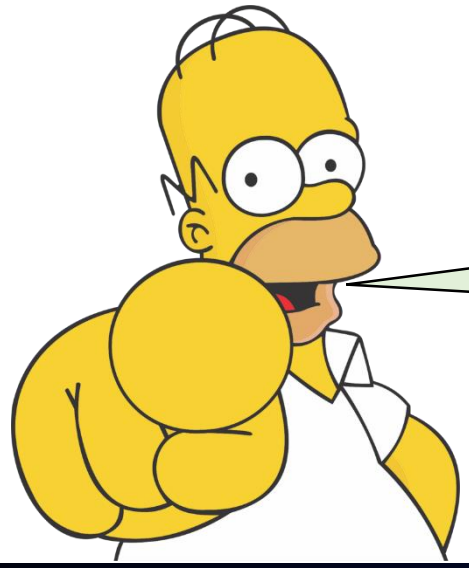
Touched (toque): fornece as coordenadas de posição quando o usuário toca na tela;

Flung (arremesso): considera argumentos como a posição inicial e a velocidade do deslize na tela;

Dragged (arraste): coleta as coordenadas de posição do dedo do usuário quando estiver deslizando na tela.

Eventos por Componente

Tipo de Evento	Canvas	Ball	Sprite
Arraste	✔	✔	✔
Arremesso	✔	✔	✔
Toque	✔	✔	✔



Você sabe qual é a função desses componentes? Vamos ver!

Entendendo *Balls* e *Sprites*

- **Balls e Sprites** são **componentes voltados à animação**, ou seja, **elementos interativos**;
- **Sprites** são usados para descrever **imagens que são capazes de se movimentarem na tela**;
- **Balls são círculos (bolas) animadas** em que somente o seu **tamanho e a sua cor** podem ser **alterados**;
- Salienta-se que **tais componentes** devem **obrigatoriamente** ser **utilizados com o Canvas**.

Mas onde Utilizá-los?

- Um **componente sprite** pode ser um **personagem de um jogo** em movimento;
- Imagine, por exemplo, jogos como o **Sonic**, o **Super Mario** ou o famoso **Pacman**;
- Vários jogos usam o **arraste e o arremesso** para fazer o personagem **virar**, **pular** ou **abaixar**;
- É por esse motivo que é interessante **conhecer a fundo as funções desses componentes**.

Informações Adicionais

- Existem **várias fontes disponíveis** para **checar a fundo a função** de um dado **componente**;
- O **pessoal do MIT**, por exemplo, desenvolveu uma **página** com um **conteúdo bastante rico**;
- Portanto, **sempre que possível** use essa **fonte de recursos** que além de **completa é muito confiável**.

http://ai2.appinventor.mit.edu/reference/components

Sobre o Evento *Flung*

(Arremesso)

- Na **aula passada** foi visto o **funcionamento** dos eventos ***Touched*** e ***Dragged*** no Canvas;
- **Flung é um evento mais complexo** do que os anteriores, pois seus **argumentos são mais técnicos**;
- No entanto, os nossos **procedimentos (métodos) não precisam conhecer todos os argumentos**;
- Assim, pode-se construir **Apps simples** usando apenas a **velocidade e o ângulo de trajetória**.

Argumentos do Evento *Flung*
(Arremesso)

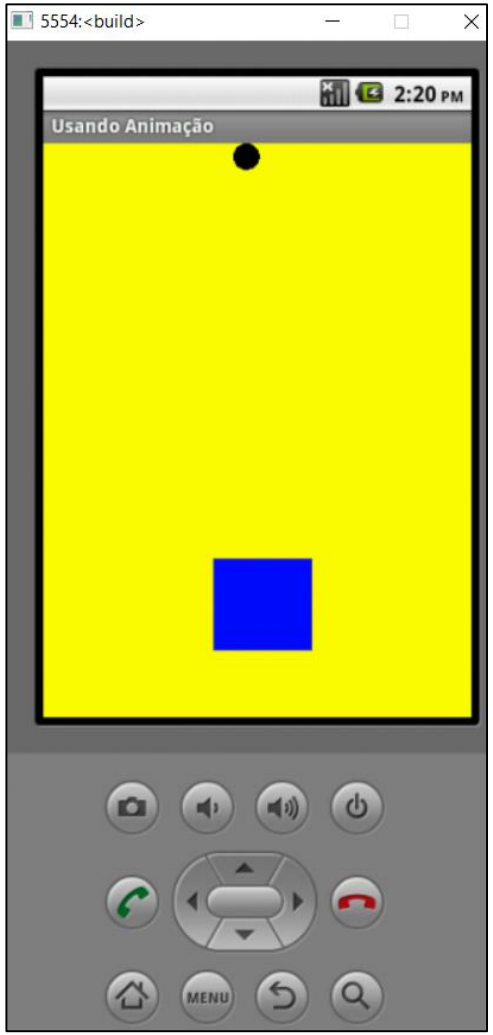
Argumento	O que Significa?
x	A coordenada x (horizontal) indica onde começou o evento. Quanto maior o valor, mais para a direita é a posição da *ball ou sprite*.
y	A coordenada y (vertical) indica onde começou o evento. Quanto maior o valor, mais para baixo é a posição da *ball ou sprite*.
speed	A quantos pixels por segundo a *ball ou o sprite* se movimentará.
heading	A direção em graus em que *a ball ou o sprite* se movimentará (*Para direita é 0 graus*).
xvel	A taxa de mudança *da ball ou do sprite* na direção x.
yvel	A taxa de mudança da *ball ou do sprite* na direção y.

O que são Intervalos?

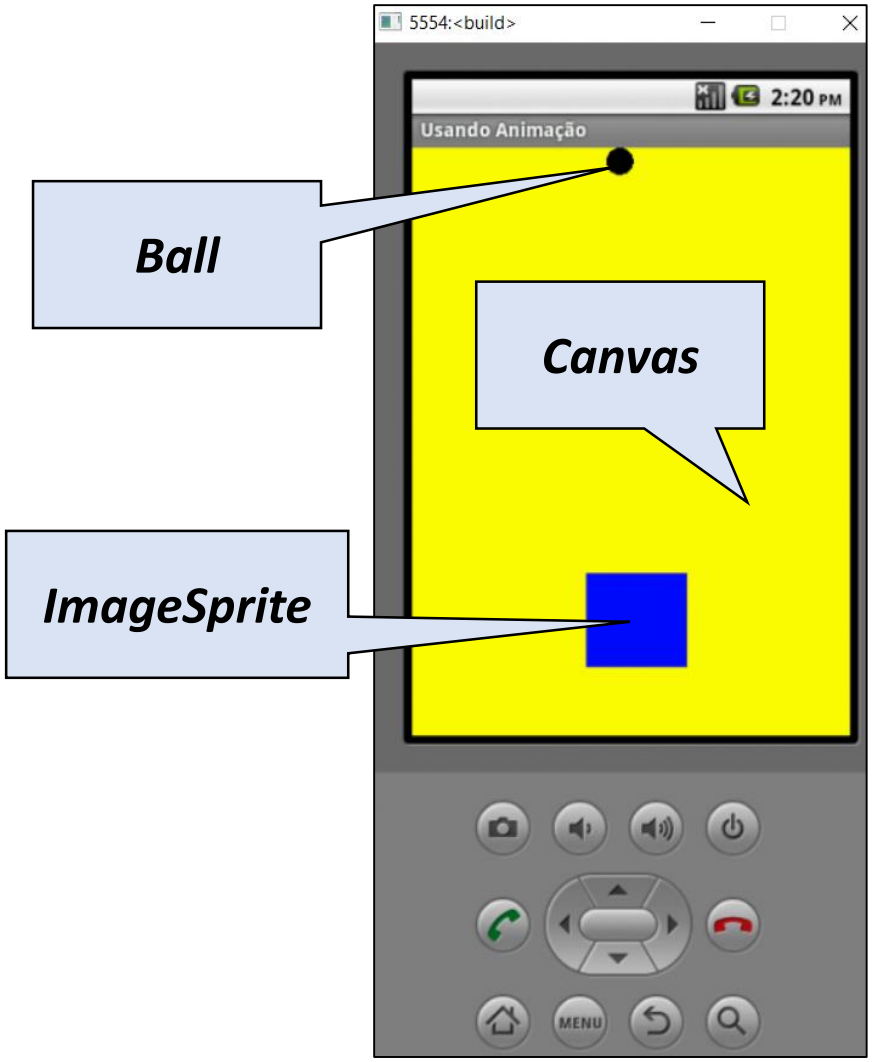
- Um **intervalo** indica a **frequência (em milissegundos)** com que a **posição do *sprite* é atualizada**;
- Se o **intervalo for 50** e a **velocidade for 10**, o *sprite* se movimentará **10 pixels a cada 50 milissegundos**;
- A **velocidade** é determinada com base no **número de pixels por intervalo** (*mil milissegundos = um segundo*).

Vamos Praticar!

Iremos **criar um App** que usa dois componentes: o ***Ball*** e o ***ImageSprite***. A imagem poderá ser **arrastada** para qualquer lugar da tela e a bola poderá ser **arremessada** em direção a imagem. Se houver uma **colisão**, a imagem inicial deverá ser trocada por outra.

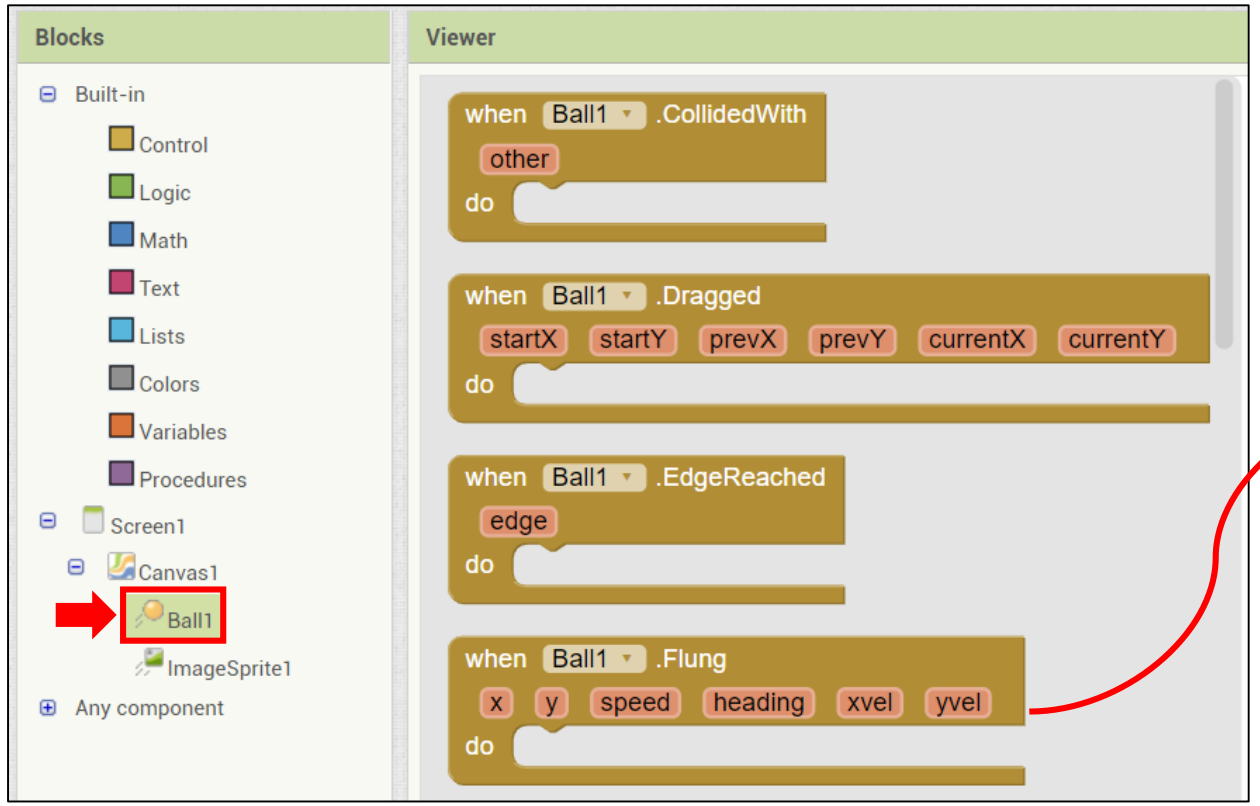


Apresentação do Layout

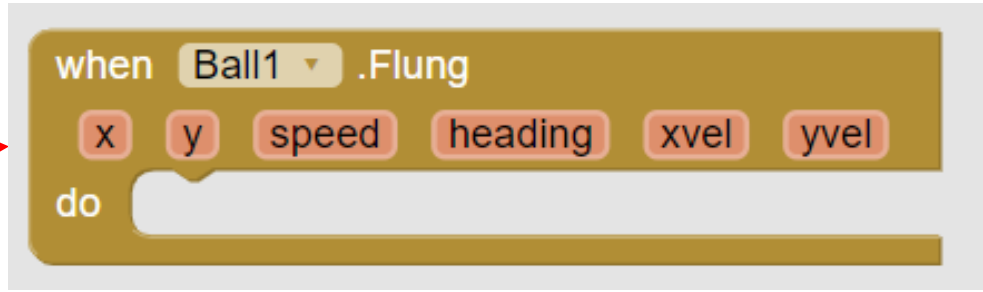


Screen1	**AlignHorizontal:** Center **AlignVertical:** Center **Icon:** Colisao.png **ScreenOrientation:** Portrait **Title:** Usando Animação
Canvas	**BackgroundColor:** Yellow **Height:** Fill parent **Width:** Fill parent
Ball	**Interval:** 10 **Radius:** 10
ImageSprite	**Picture:** SpriteAzul.png

Passo 1: Evento *Flung* da *Ball*
(Arremesso)



Event Handler



Identifica quando a bola é arremessada!

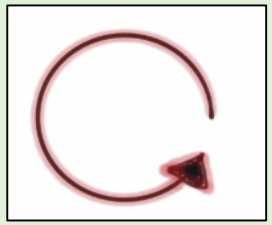
Passo 2: Definindo a Velocidade da *Ball*

Deixa a **movimentação** (pixels) mais rápida!

O ***Speed*** e o ***Heading*** são **calculados automaticamente** no arremesso.

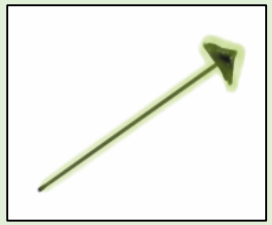
Define a **direção** da movimentação!

Portanto, a **movimentação é sempre composta pela velocidade e direção. Ambos precisam ser definidos** para funcionar!



Velocidade (*speed*)
=
Nº Pixels sem Direção

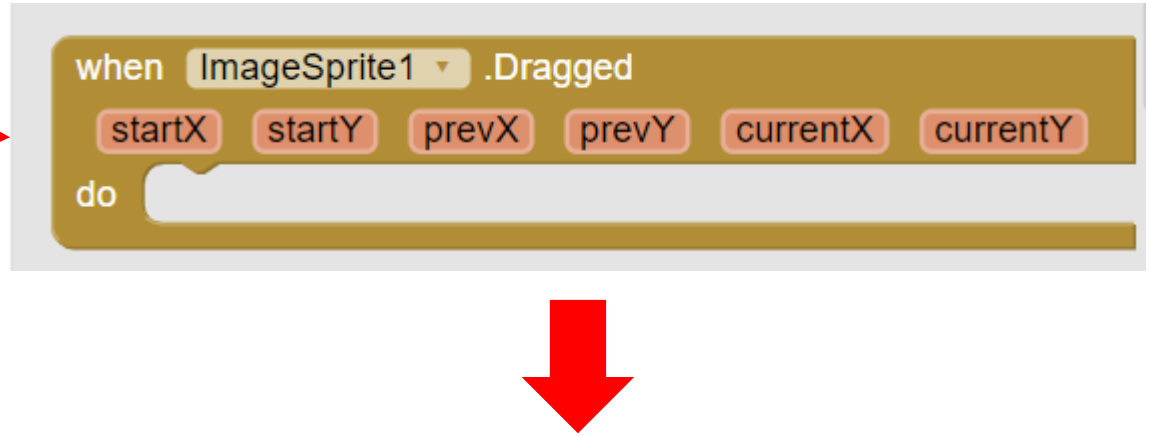
≠



Movimentação
=
Velocidade + Direção

Passo 3: Evento *Dragged* do *ImageSprite*

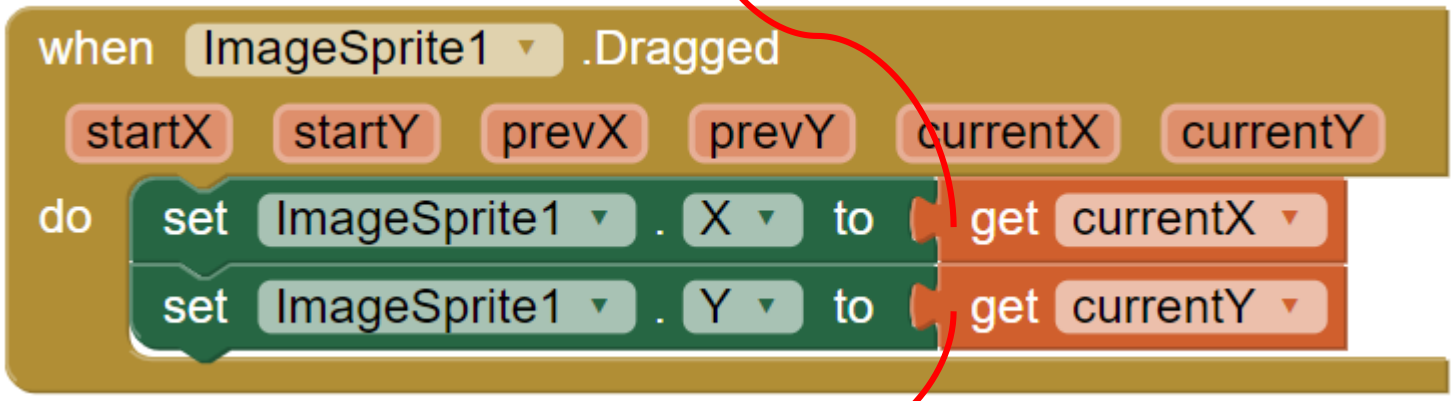
Event Handler



Permite que a imagem (Sprite) seja arrastada!

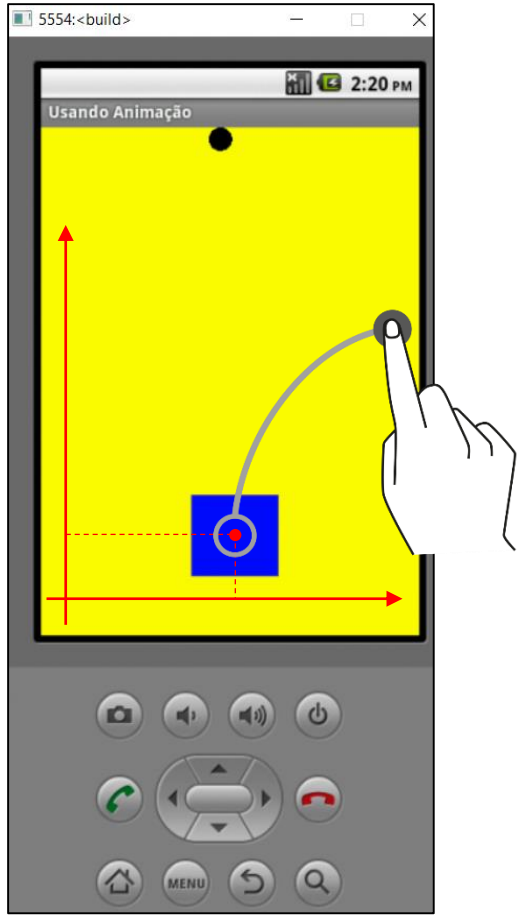
Passo 4: Definindo a Posição do *ImageSprite*

Define a **posição X** do ***ImageSprite***

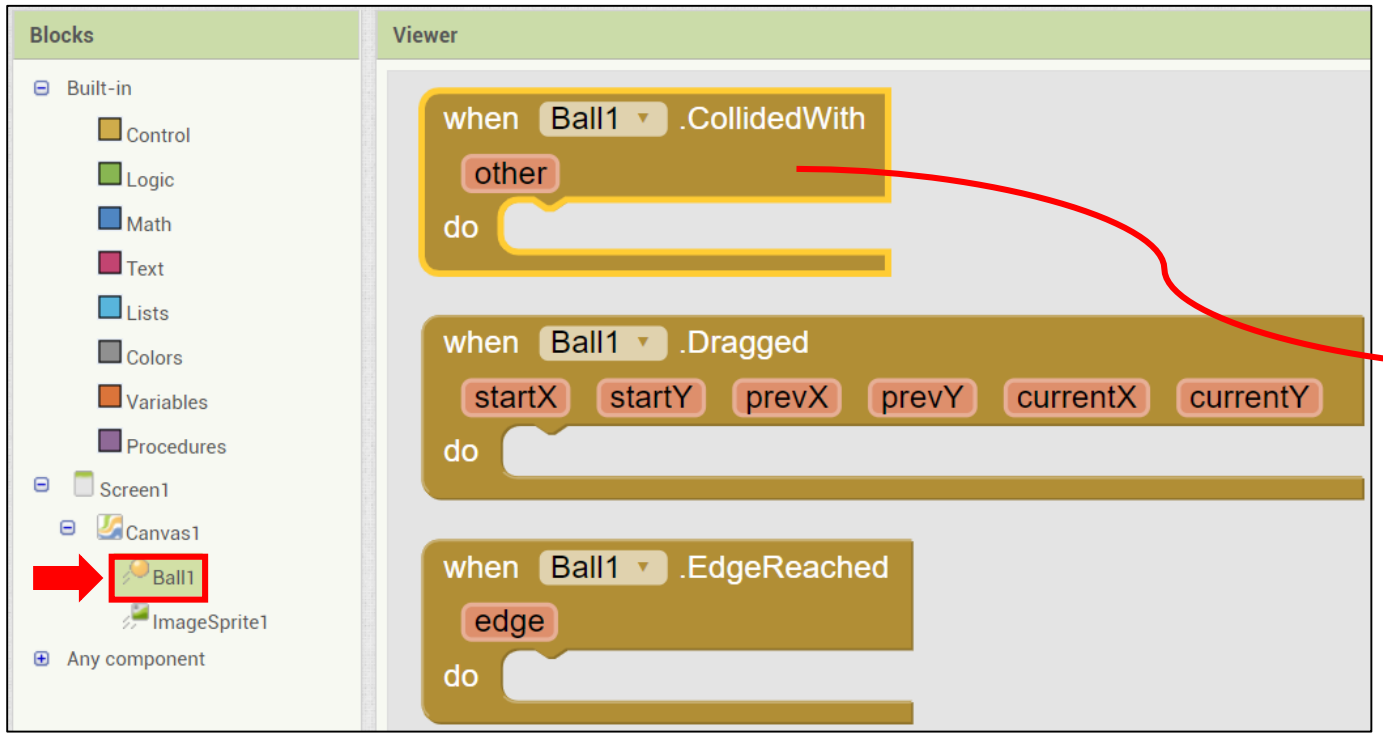


Define a **posição Y** do ***ImageSprite***

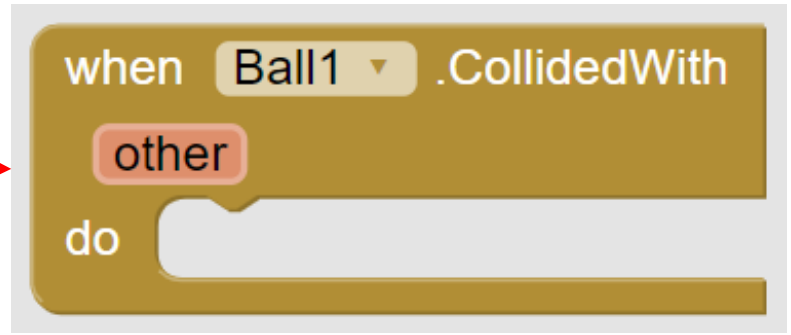
Tais coordenadas consistem no **local onde ficará a imagem** quando o **dedo do usuário deixar a tela**!



Passo 5: Evento *CollidedWith* da *Ball*



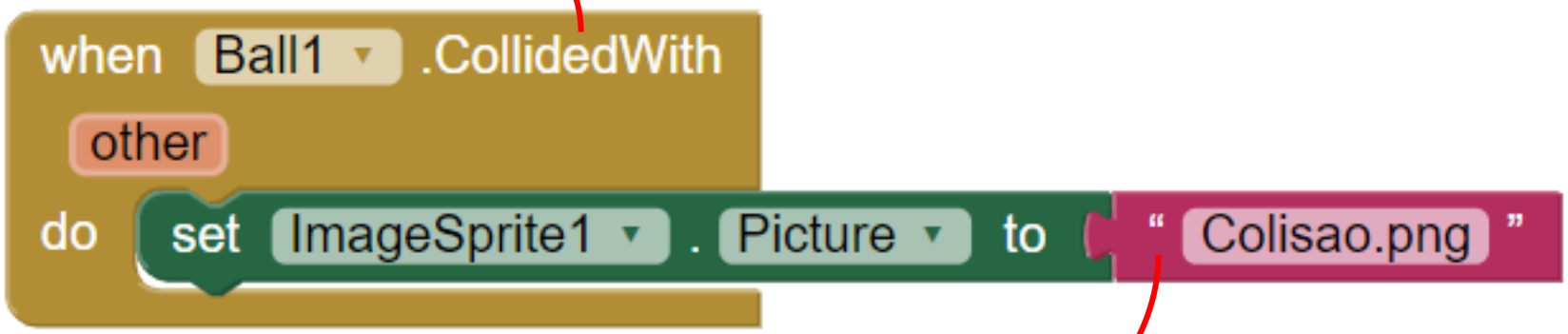
Event Handler

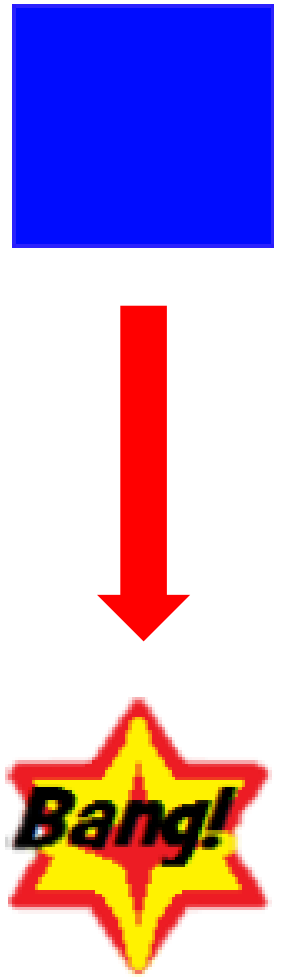


Identifica a colisão da bola com um objeto!

Passo 6: Trocando a Imagem

Identifica a **colisão** da **Bola com a Imagem**





Realiza a **troca da figura SpriteAzul.png para Colisao.png**

Vamos Praticar!

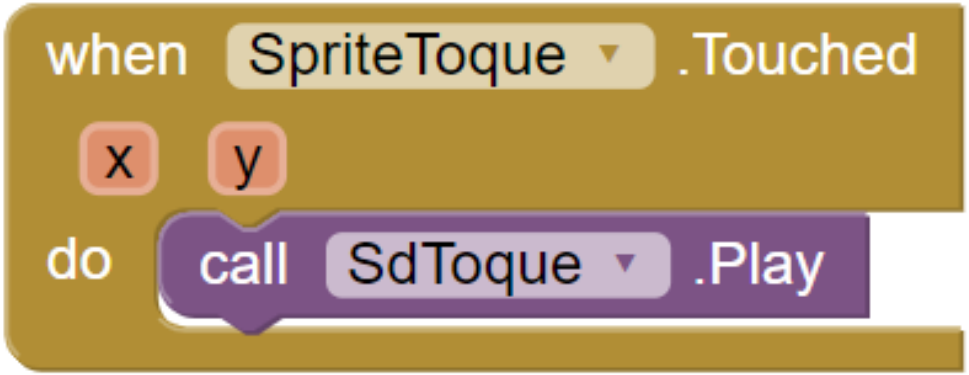
Agora vamos criar um **App mais interativo** que **responderá a diferentes eventos**. Basicamente **o usuário realizará uma ação** (**Toque**, **Arraste** ou **Arremesso**) e o App deverá **executar um som específico** para cada caso. Veja o layout ao lado.

Apresentação do Layout

Screen1	**AlignHorizontal:** Center **AlignVertical:** Center **Icon:** Icone.png **ScreenOrientation:** Portrait **Title:** Diferentes Eventos
Canvas	**BackgroundImage:** fundo.png **Height:** Fill parent **Width:** Fill parent
3 ImageSprite	**Picture:** carregar imagens .png
3 Sound	**Source:** carregar sons .wav

Passo 1: Reagindo ao Toque

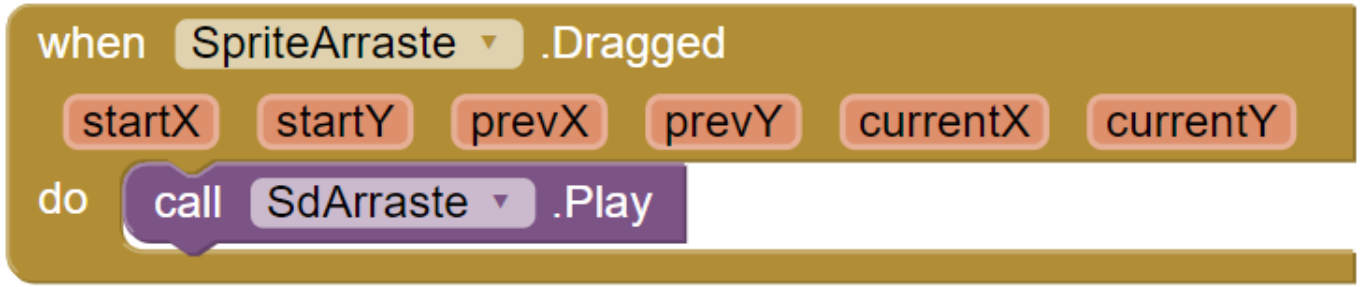
Event Handler



Toca o som do componente Sound SdToque!

Passo 2: Reagindo ao Arraste

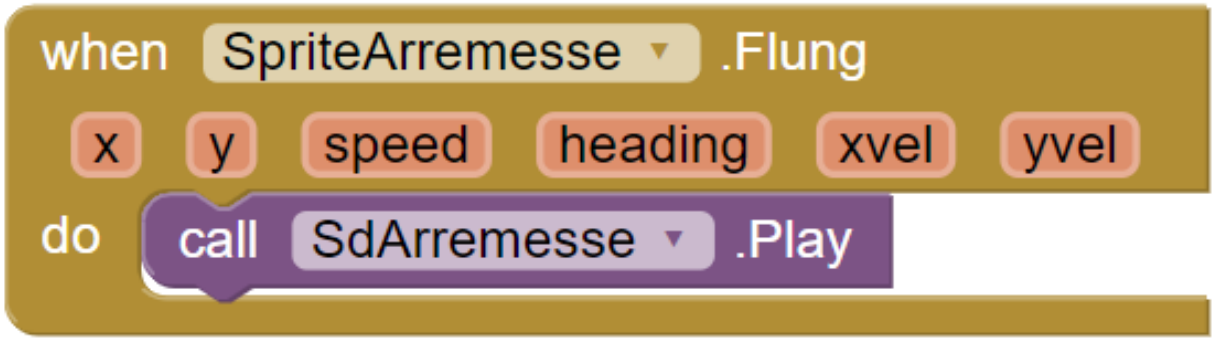
Event Handler



Toca o som do componente Sound SdArraste!

Passo 3: Reagindo ao Arremesso

Event Handler



Toca o som do componente Sound SdArremesse!

Dúvidas?

Programação para Dispositivos Móveis

Curso de Ciência da Computação

Universidade Paulista (UNIP)

DESENHO E ANIMAÇÃO

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

AULA 4