

# Programação para Dispositivos Móveis

Curso de Ciência da Computação

Universidade Paulista (UNIP)

## ENTENDENDO E APLICANDO: NÚMEROS ALEATÓRIOS E PROCEDIMENTOS

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 6**



# O que são Números Aleatórios?



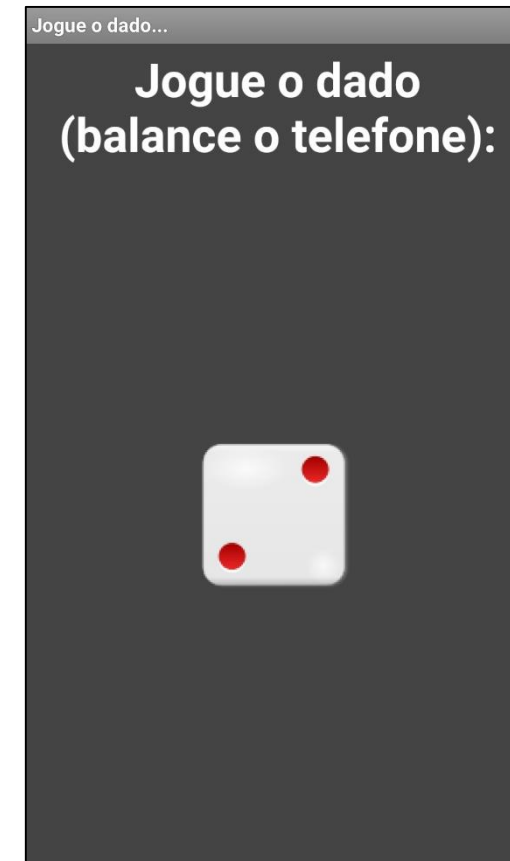
- ❑ Números aleatórios são gerados de forma dinâmica dentro de um intervalo estipulado;
- ❑ Pode-se dizer que ocorre um “sorteio” de um número inteiro localizado dentro do intervalo;
- ❑ Esse é um recurso útil quando queremos que o App realize uma ação aleatória (probabilística);
- ❑ Para usá-lo devemos ir ao bloco **Math** → 

# Aleatoriedade na Prática!

Vamos criar um App que **simule um dado**.

Ao **jogá-lo (balancear o telefone)** será apresentado a **sua face (1 ao 6)**. O jogo irá iniciar **apresentando o valor 1 como padrão**.

A medida que o usuário interage com o celular **novos valores vão sendo sorteados**.



# Sobre o Layout



**Screen1**

**AlignHorizontal:** Center  
**AlignVertical:** Center  
**BackgroundColor:** Dark Gray  
**ScreenOrientation:** Portrait  
**Sizing:** Responsive  
**Title:** Jogue o dado...

**Label**

**FontBold:** True  
**FontSize:** 30  
**Width:** 280 px  
**Text:** Jogue o dado  
**TextAlignment:** Center

**2 Image (Separação)**

**Height:** Fill parent

**1 Image (Dado)**

**Picture:** dado\_1.png

**AccelerometerSensor**

**Nenhuma configuração necessária**

# Programando a Lógica

(Modo mais Difícil)



**Atenção:**  
Carregue primeiramente  
as imagens!



## Media

dado\_1.png  
dado\_2.png  
dado\_3.png  
dado\_4.png  
dado\_5.png  
dado\_6.png

Upload File ...

# Programando a Lógica

(Modo mais Fácil)

initialize global lado to 0

initialize global imagem to ""

```
when Aceletrometro .Shaking
do
  set global lado to random integer from 1 to 6
  set global imagem to join [" dado_ "
    get global lado
    ".png "]
  set imgDado . Picture to get global imagem
```

**Atenção:**  
Carregue primeiramente  
as imagens!



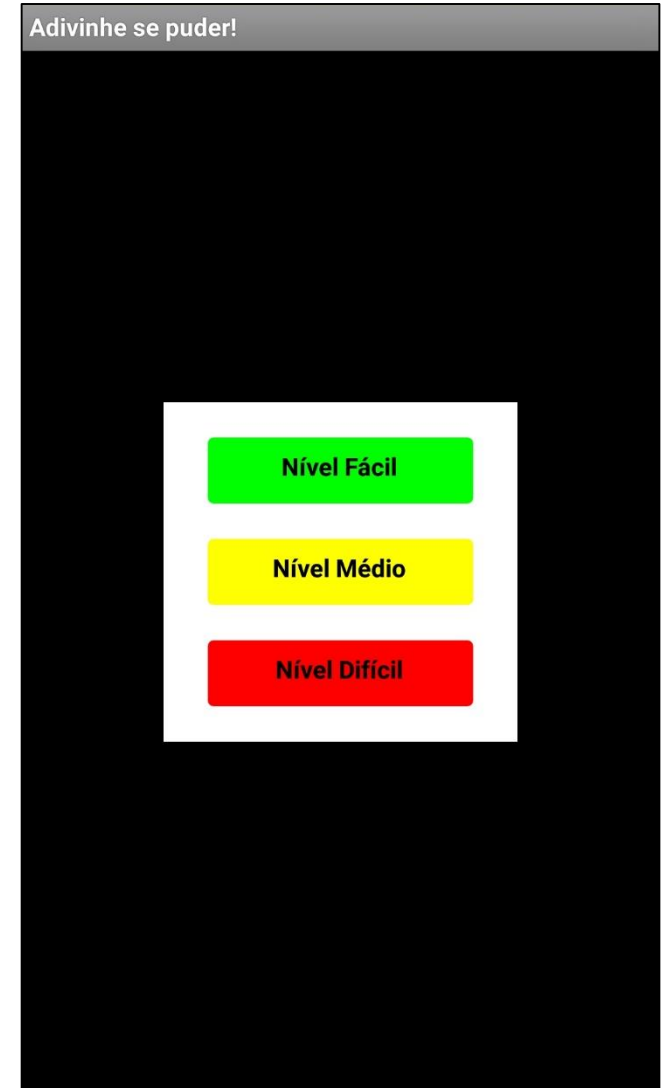
## Media

dado\_1.png  
dado\_2.png  
dado\_3.png  
dado\_4.png  
dado\_5.png  
dado\_6.png

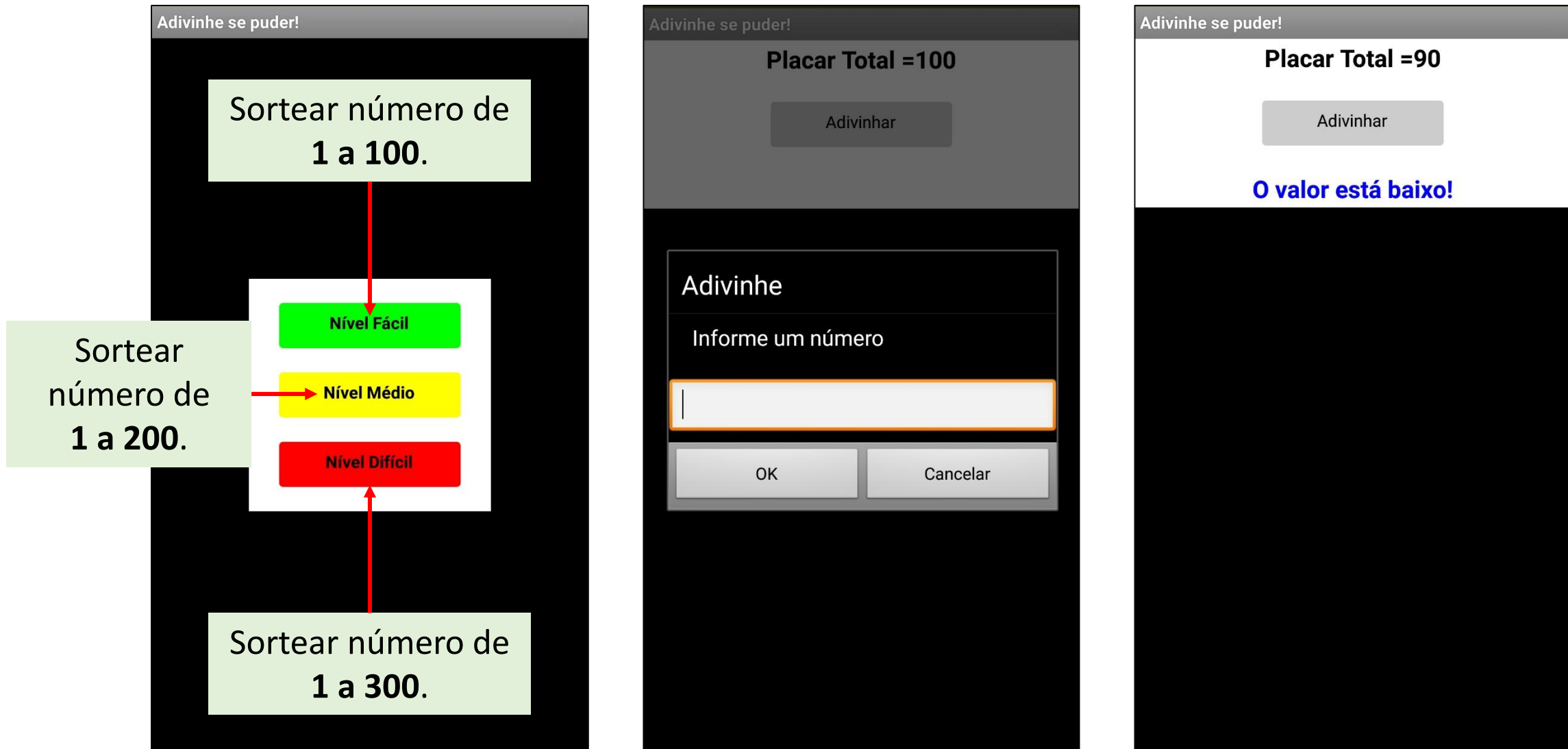
Upload File ...

# Adivinhe o que estou Pensando!

O App *pensará em um número aleatório de acordo com o nível escolhido pelo usuário*. Fácil (1 à 100), Médio (1 à 200) e Difícil (1 à 300). O usuário **começará com 100 pontos** e deverá adivinhar o número sorteado. **A cada tentativa errada o score será reduzido em 10 pontos**, ou seja, 10 tentativas erradas é ***game over***. O usuário deverá ser guiado através de **mensagens para saber se está longe ou perto de adivinhar**. Caso ele acerte apresente uma **mensagem de parabéns e habilite o botão para resetar o jogo**.



# Apresentando o Funcionamento

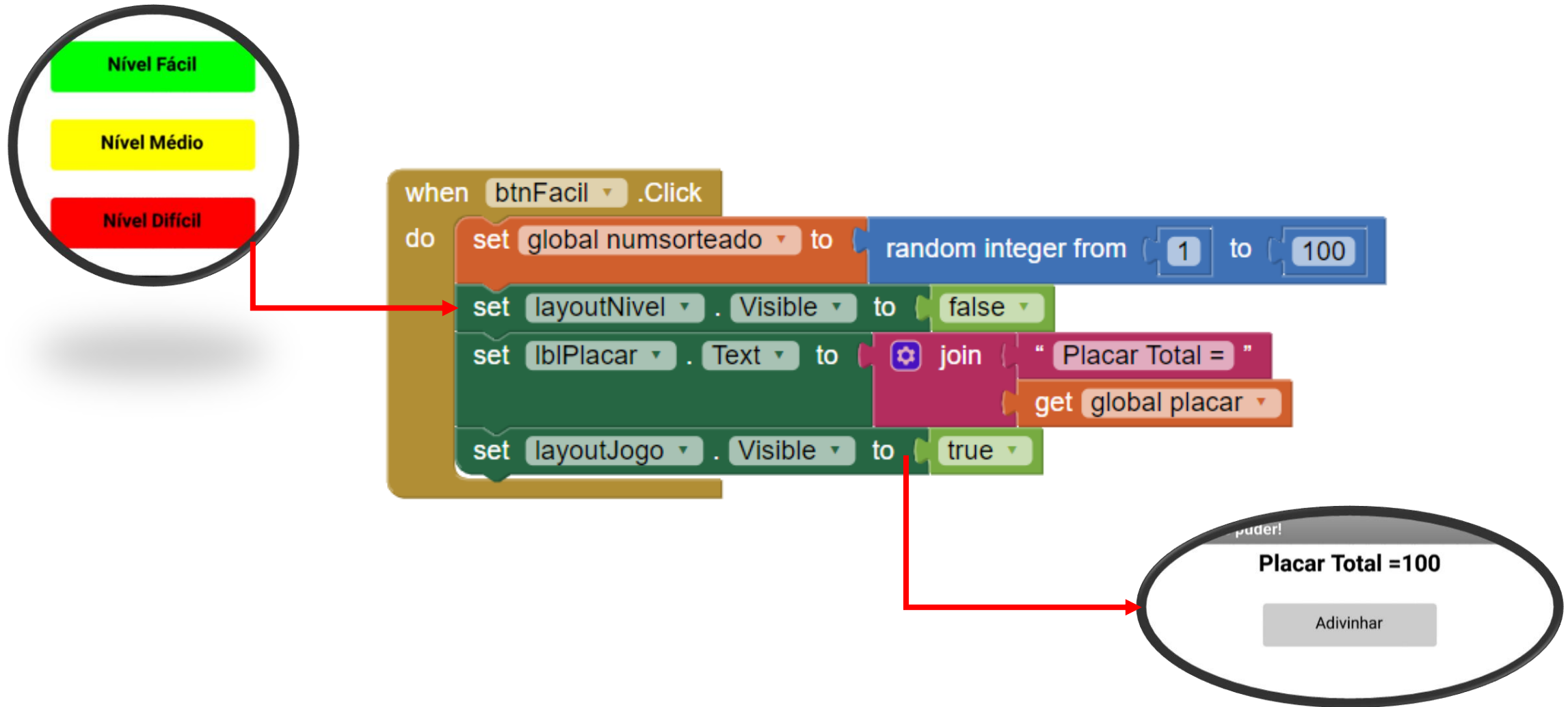




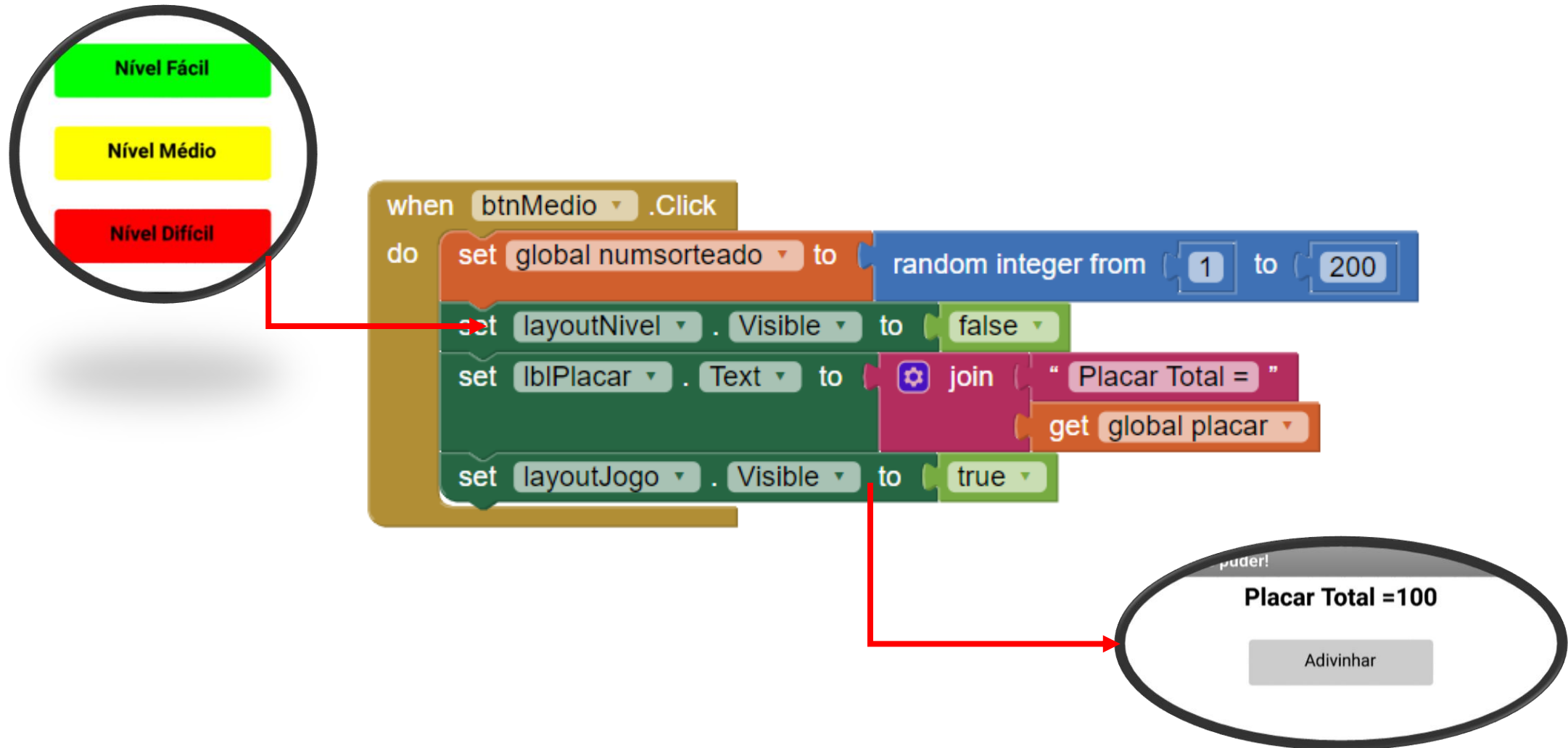
# Passo 1: Criar as Variáveis Necessárias



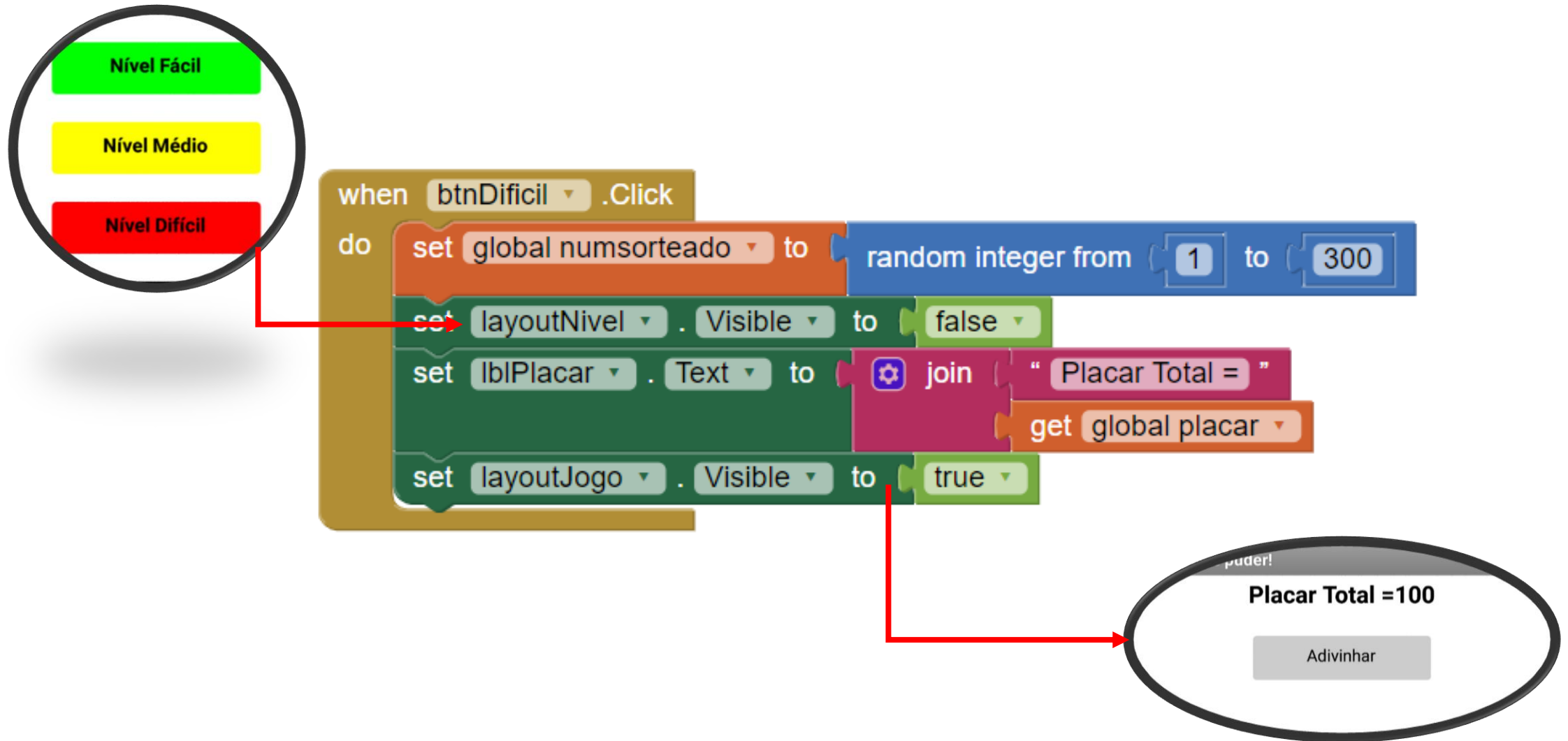
# Passo 2: Programando o Nível Fácil



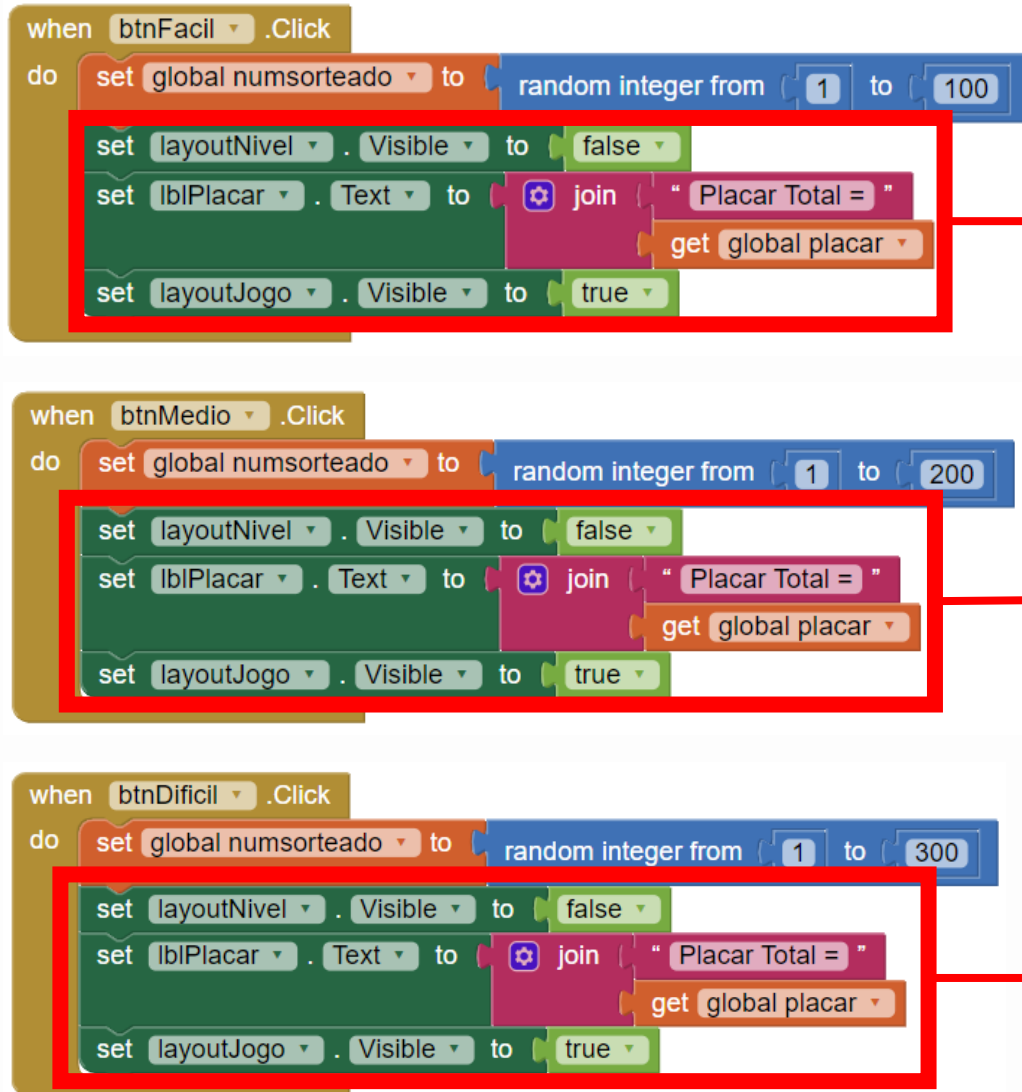
# Passo 3: Programando o Nível Médio



# Passo 3: Programando o Nível Difícil



# Quanta Repetição de Código!



Tudo Igual!

Por que não utilizamos procedimentos?



# O que são Procedimentos?

- ❑ Os *procedimentos* permitem **deixar o código modular**, ou seja, **organizá-lo em partes específicas**;
- ❑ Ao *invés de repetir o bloco três vezes*, poderíamos **criar um único procedimento e apenas chamá-lo**;
- ❑ *Mas se podemos criar qualquer App sem o uso de procedimentos porque devemos usá-los?*
- ❑ A ideia é bastante simples, você estará **economizando tempo e tornando o código mais entendível**.

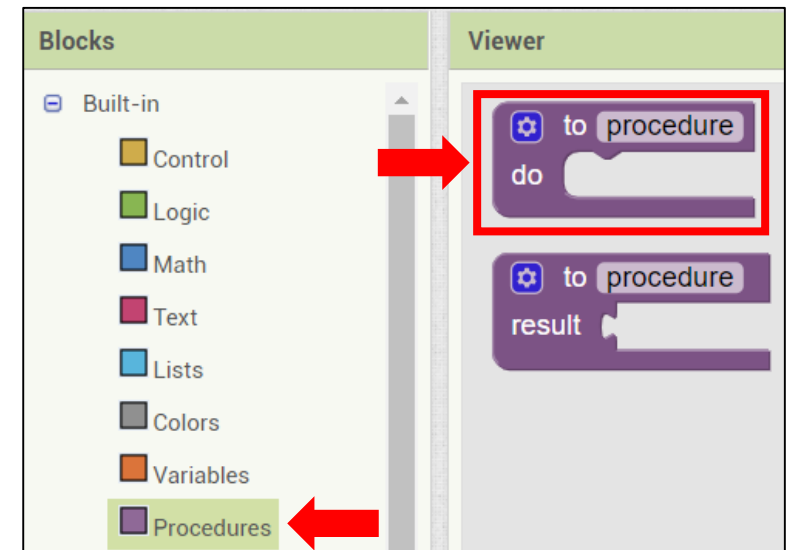
# Vantagens de Usar Procedimentos

- ❑ **Economia de tempo:** funcionam como se fossem atalhos, uma vez montadas sempre poderão ser usadas;
- ❑ **Facilidade de manutenção:** todo o código está centralizado em um único local, sendo fácil de corrigir;
- ❑ **Fácil entendimento:** o fato de possuir nomes igual a variáveis faz com que seja fácil de identificar sua função;
- ❑ **Apps eficientes:** os aplicativos que usam procedimentos possuem um melhor desempenho na execução.



# Criando Procedimentos no App Inventor

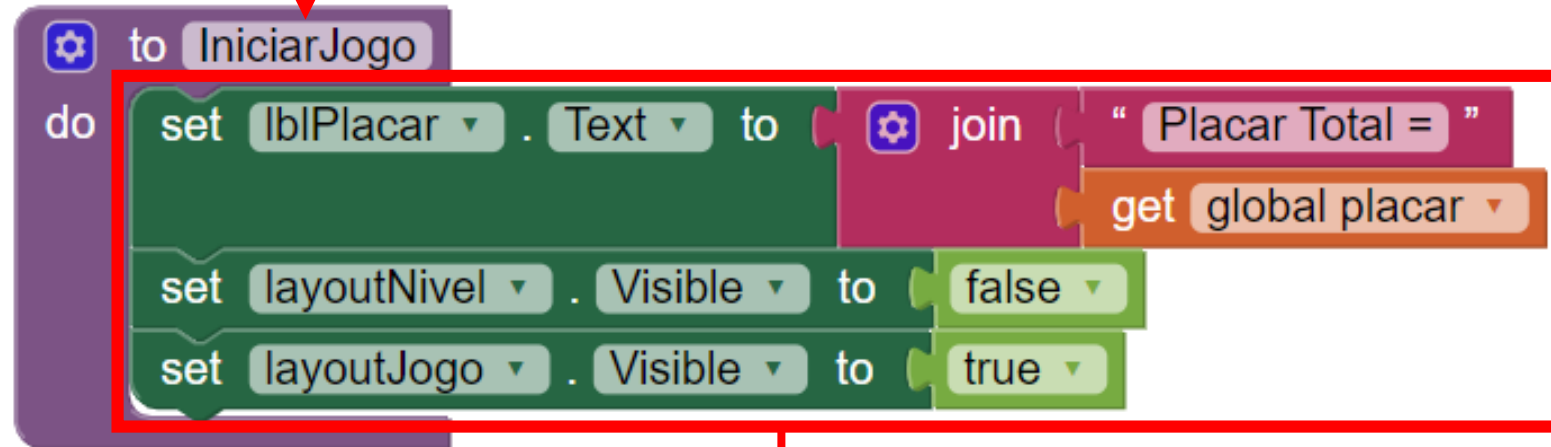
- ☐ O uso de procedimentos é **similar ao uso de variáveis** e estão disponíveis na seção **“Procedures”**;
- ☐ Para utilizá-los **basta arrastar o bloco de código “to procedure do”** para o editor de blocos;
- ☐ Defina um **nome para o procedimento** de modo a deixar claro **qual é a sua função**;
- ☐ Realize a **codificação necessária dentro dele** que em nosso caso é **todo o código repetido**.





# Procedimento Finalizado

Nome do Procedimento!



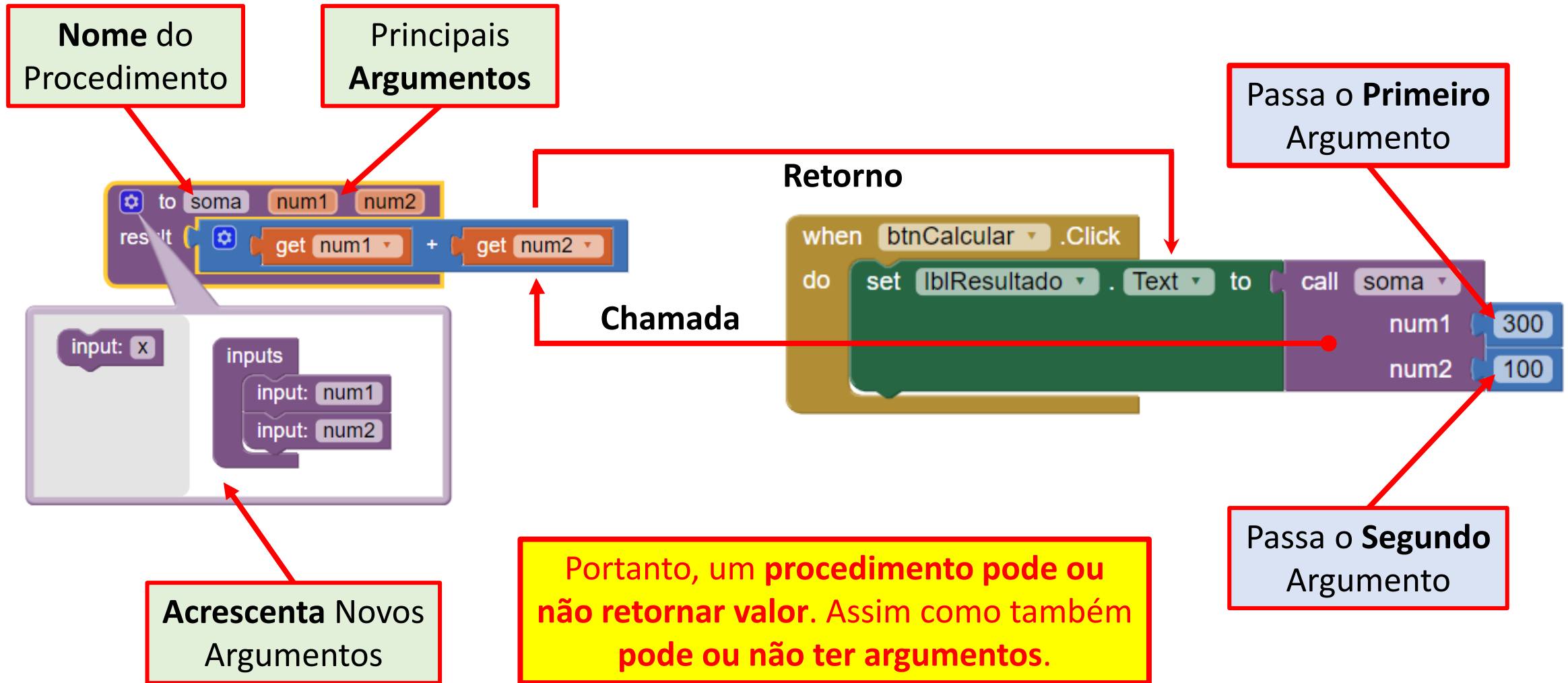
Código que estava repetido!

# Chamando o Procedimento Criado



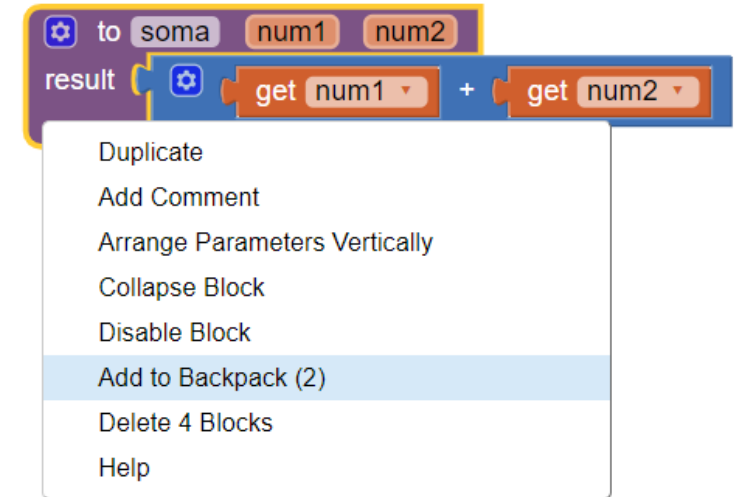
Estou há ~~070~~ 5 dias  
sem fazer gambiarras

# Procedimento que Retorna Valor

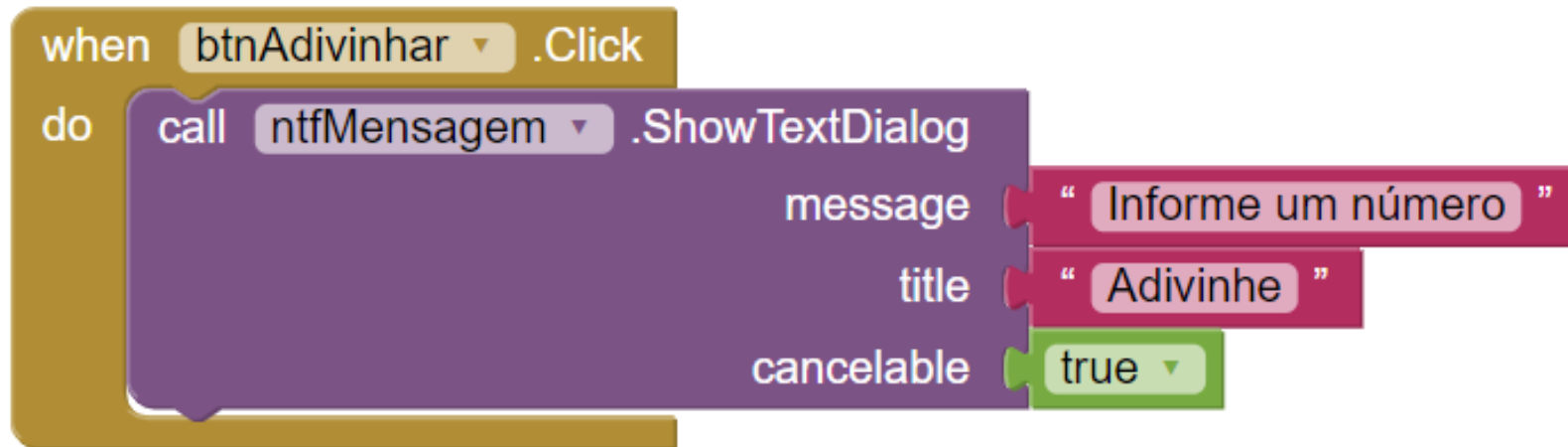


# Reutilizando Procedimentos

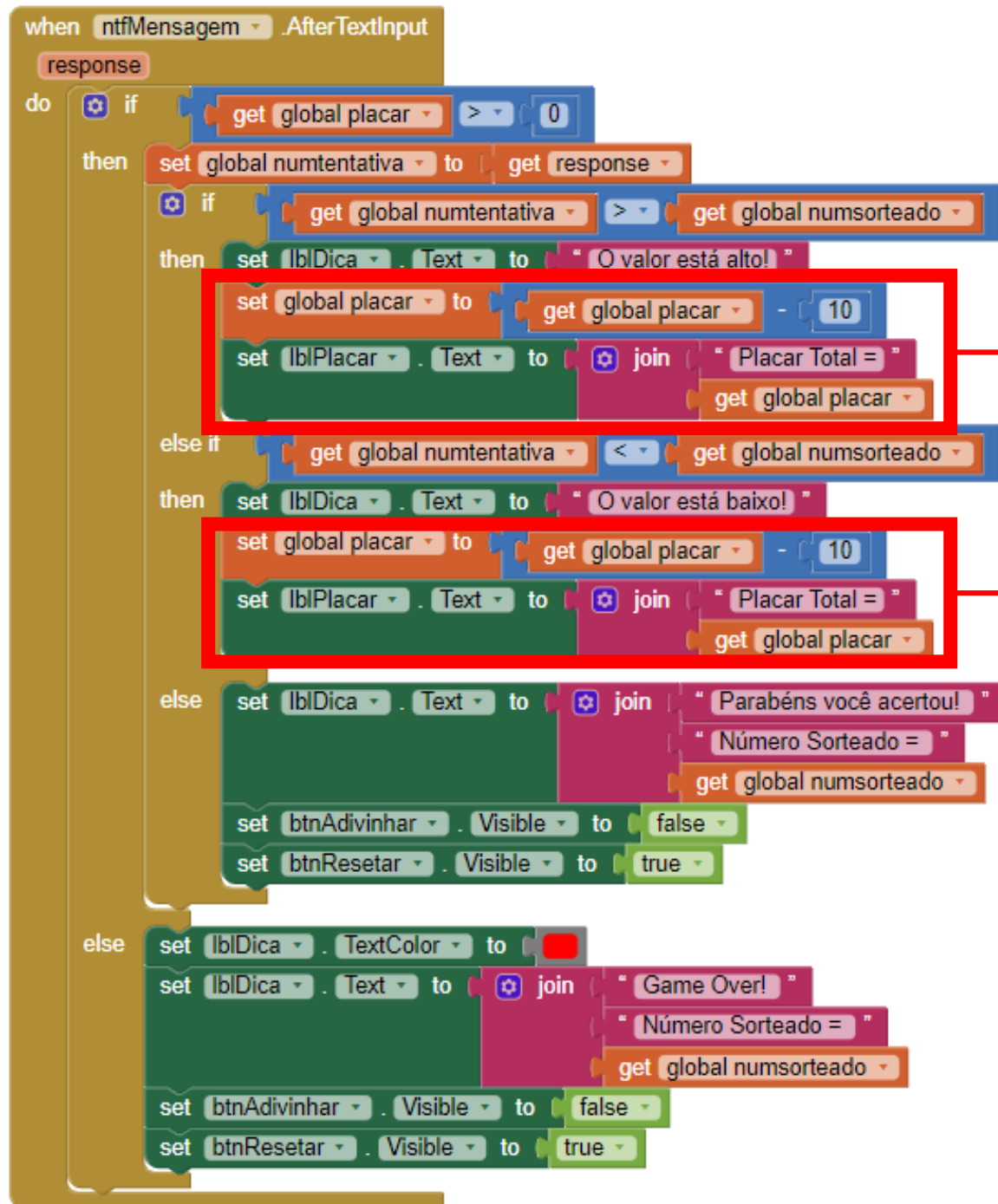
- ❑ Imagine que você desenvolveu um **procedimento para a validar a entrada do e-mail**;
- ❑ Muito provavelmente você irá **utilizá-lo em vários projetos** que necessite dessa função;
- ❑ Nós poderíamos **reescrever o procedimento em cada projeto**, *mas isso é trabalhoso*;
- ❑ Para resolver esse problema iremos **adicioná-lo ao “Backpack”** para que seja **compartilhado**.



## Passo 4: Apresentar o *ShowTextDialog*



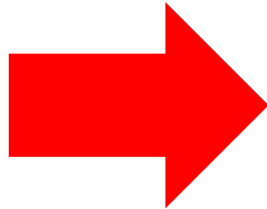
## Passo 5: Capturar a Tentativa de Adivinhação



**Agora é com você!**  
Que tal criar um  
procedimento  
**AtualizaPlacar...**

# Passo 6: Programar o Botão de Reset

A ideia é deixar as coisas **exatamente da mesma forma** como estavam no **início do jogo!**



# Dúvidas?

