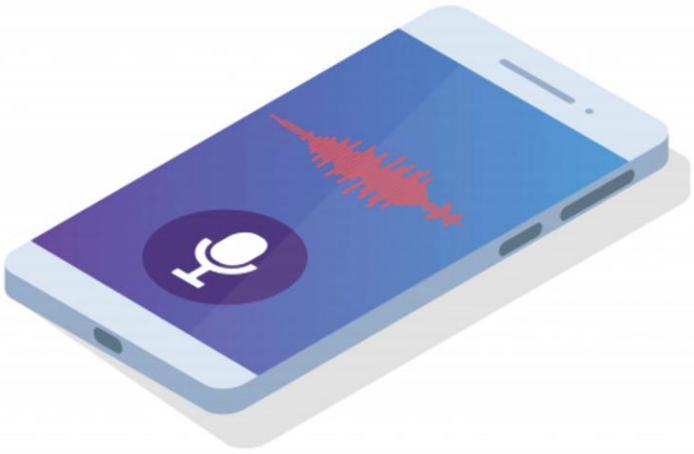
AGENDA 1

TRABALHANDO COM ÁUDIO E UTILIZANDO O ATRIBUTO ONCLICK NO BUTTON



GEEaD - Grupo de Estudos de Educação a Distância

Centro de Educação Tecnológica Paula Souza

GOVERNO DO ESTADO DE SÃO PAULO

EIXO TECNOLÓGICO DE INFORMAÇÃO E COMUNICAÇÃO

CURSO TÉCNICO EM DESENVOLVIMENTO DE SISTEMAS

PROGRAMAÇÃO MOBILE I

Expediente

Autor:

GUILHERME HENRIQUE GIROLLI

Revisão Técnica:

Eliana Cristina Nogueira Barion

Revisão Gramatical:

Juçara Maria Montenegro Simonsen Santos

Editoração e Diagramação:

Flávio Biazim

São Paulo – SP, 2020

MERGULHANDO NO TEMA...

Atributo onClick

Verificamos até aqui que para gerar uma "ação" para um determinado botão ou Button era necessário desenvolver uma codificação Java, utilizando o "setOnClickListener" responsável por gerenciar esse botão. Agora, vamos abordar um novo formato para tratamento do clique do botão, que também é amplamente utilizado pelos desenvolvedores de aplicativos.

O atributo "onClick" é declarado no arquivo XML responsável pela construção da tela ou Activity. Cabe a ele chamar um método da codificação Java que, por sua vez, desenvolve uma ação quando o botão é pressionado. O atributo "onClick" substitui a utilização do "setOnClickListener", e é muito utilizado nas abordagens mais simples de codificação para as ações de um Button.

A figura 3 demonstra um aplicativo de exemplo com dois botões.

O primeiro botão do aplicativo trata o clique através do "setOnClickListener" como podemos observar no código Java da figura 4. O segundo botão utiliza o "onClick" na codificação XML da figura 3, o atributo "onClick" faz referência à execução de um método na codificação Java da figura 4, esse método obrigatoriamente deve ser "public" retornando "void" e recebendo como parâmetro uma "View".

Figura 3 – Exemplo de aplicativo com dois botões.

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MainActivity">

    //Botão com setOnClickListener
    <Button
        android:id="@+id/btnSetOnClickListener"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
```

```
        android:layout_marginTop="152dp"
        android:text="Button com setOnClickListener"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />

    //Botão com atributo "onClick"
    <Button
        android:id="@+id/btnOnClick"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginTop="60dp"
        android:text="Button com Atributo OnClick no XML"

        android:onClick="acaoBtnOnClick"

        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.498"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/btnSetOnClickListener" />
</androidx.constraintlayout.widget.ConstraintLayout>
```

Figura 4 – Arquivo XML do exemplo de aplicativo com dois botões.

```
package com.example.button;

import androidx.appcompat.app.AppCompatActivity;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;

public class MainActivity extends AppCompatActivity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        //Tratamento do clique do button com o setOnClickListener
        Button btnSetOnClickListenertProg = (Button)
findViewById(R.id.btnSetOnClickListener);
        btnSetOnClickListenertProg.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                //Códigos que serão executados no clique do botão
            }
        });
    }

    //Tratamento do clique do button com o método indicado no
    //atributo “onClick” no arquivo XML da Activity
    public void acaoBtnOnClick(View v)
    {
        //Códigos que serão executados no clique do botão
    }
}
```

Figura 5 – Arquivo Java do exemplo de aplicativo com dois botões.

Podemos observar que o sistema por meio do “onClick” oferece uma construção com menos códigos no arquivo Java quando comparado com o sistema “setOnClickListener”.

Os dois processos são eficientes por proporcionar uma ação para o botão, a principal diferença é que o gerenciamento pelo “setOnClickListener” utiliza uma classe Java especializada em “ouvir” e tratar o clique efetuado pelo usuário, esse tratamento é todo realizado por meio de estruturas na programação Java.

Já o atributo “onClick” desenvolve automaticamente, nos “bastidores” da aplicação, uma estrutura para “ouvir” e tratar os cliques do usuário, transferindo apenas para o Java a função de desempenhar uma ação para esse clique, por meio do método associado a ele.

Portanto quando o desenvolvedor necessita interferir ou tratar alguns aspectos deste clique antes de desempenhar uma ação do botão, é aconselhável a utilização do “setOnClickListener”, uma vez que ele consegue manipular essa classe Java. Já nos casos mais simples, como no projeto do jogo “Descubra o número...”, em que o desenvolvedor necessita apenas executar uma ação no clique do botão, é aconselhável a utilização do “onClick”, que ocupa uma quantidade menor de linhas de código na programação Java, onde é necessário apenas a construção de um método.

Classe MediaPlayer

A classe MediaPlayer é um framework multimídia, especializado na reprodução de mídias. Quando a classe é utilizada em um projeto, proporciona, de maneira fácil, a execução de áudio, vídeo e imagens no aplicativo.

O MediaPlayer permite reprodução de áudio ou vídeo de arquivos de mídia armazenados nos recursos do aplicativo, de arquivos na memória do dispositivo ou de um fluxo de dados por meio da Internet.

Com poucos códigos, é possível desenvolver uma ferramenta capaz de executar arquivos dos mais diversos tipos, entre eles MP3, MP4, WAV, entre outros. Para conhecer todos os arquivos compatíveis é necessário acessar o site oficial do Android Studio, disponível em: https://developer.android.com/guide/topics/media/media-formats?hl=pt-br.

O código a seguir apresenta um exemplo que implementa um objeto utilizando a classe MediaPlayer.

```
MediaPlayer mp = MediaPlayer.create(MainActivity.this,R.raw.exemplo);
mp.start();
```

Quando declaramos um objeto do tipo MediaPlayer, utilizamos o recurso create() para informar algumas coisas importantes para a execução do áudio. A primeira parte do create() é destinada à informação do contexto da aplicação, onde vamos informar a tela que vai executar o áudio. Na segunda parte, informamos a localização do arquivo. O arquivo de áudio está disponível em uma pasta "Raw" disponível na pasta de recursos do sistema chamada "Res".

Para conhecer outras formas de utilização do framework MediaPlayer, visite o site oficial para desenvolvedores que utilizam a plataforma Android Studio: https://developer.android.com/reference/android/media/MediaPlayer?hl=pt-br#create(android.content.Context,%20android.net.Uri,%20android.view.SurfaceHolder)

Raw

Quando trabalhamos com arquivos de áudio em um aplicativo, é muito importante que esse arquivo esteja presente dentro da estrutura do projeto, para isto, utilizamos a pasta "raw" para alocar esses arquivos.

A pasta "raw" é uma estrutura desenvolvida para abrigar arquivos que serão acessados por meio do resource id da classe R. Ela não vem por padrão na construção do projeto, sendo necessário o seu desenvolvimento. Clique com o botão direito sobre **"res"** depois em **"New"** e **"Directory",** depois informe o nome em minúsculo da pasta **"raw"** e clique em **"ok".**

Acompanhe a figura 6 que demonstra o processo de criação da pasta **"raw"**.

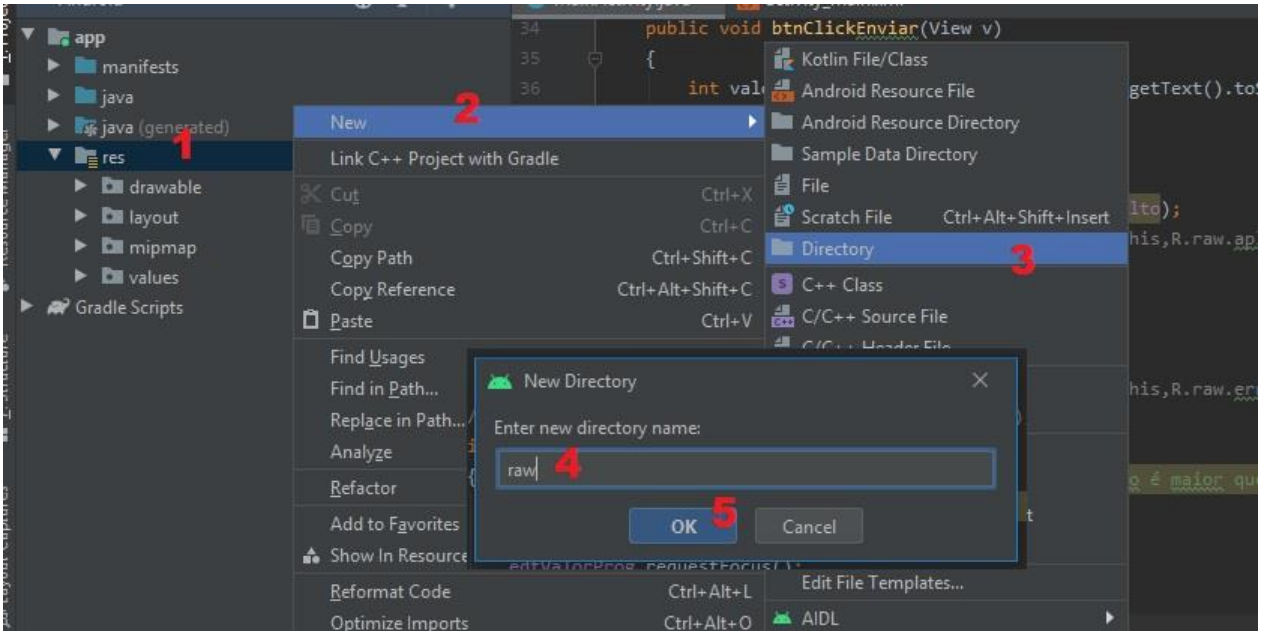


Figura 6 – Criação da pasta "Raw".

VOCÊ NO COMANDO

Antes de começar a codificar o arquivo Java é importante lembrar que essa apostila é uma continuação do projeto de um jogo com nome “Descubra o número...” desenvolvido por Karla. Todo o processo de desenvolvimento da parte visual do projeto foi realizado na agenda anterior, caso você tenha perdido algum detalhe, volte novamente no material anterior e execute os procedimentos. Vamos tratar a partir daqui apenas a parte que envolve a codificação Java.

Veja o vídeo 1 a seguir para realizar a programação do Button por meio do parâmetro onClick no arquivo XML.

Vídeo 1 – Desenvolvimento dos métodos no Java e atributos onClick no arquivo XML. Site: https://youtu.be/ujrh1Gyz-TQ

Random

A classe Random é utilizada para gerar valores aleatórios em um aplicativo Android. Essa classe utiliza alguns métodos de acordo com a necessidade do desenvolvedor, no jogo será necessário utilizar um valor aleatório do tipo inteiro em cada partida. Neste caso, é necessário utilizar o método nextInt().

O nextInt() é responsável por gerar valores inteiros de 0 (zero) até um determinado limite. Para definir o limite, o método utiliza um parâmetro que define o valor máximo que será gerado aleatoriamente, este valor máximo informado nunca será gerado pelo sistema. No exemplo de código a seguir, definimos um gerador de números inteiros de 0 (zero) até 9 (nove), que alimenta uma variável do tipo inteiro, para isso é necessário informar no parâmetro do nextInt() o valor 10 (dez).

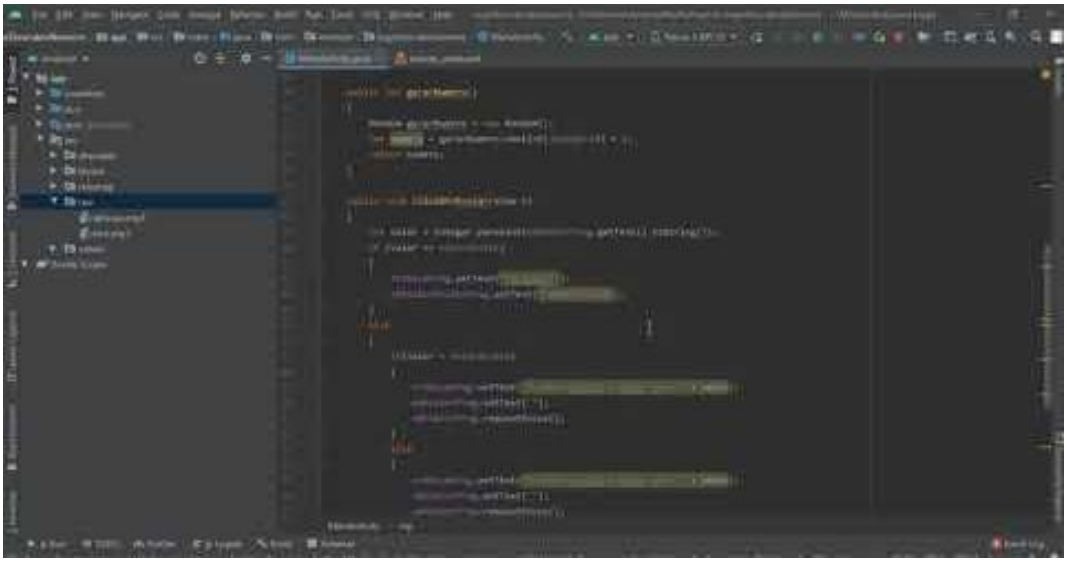
```
Random gerarNumero = new Random();
int numero = gerarNumero.nextInt(10);
```

O Random pode gerar outros tipos de valores como double, float, long, entre outros. Para conhecer mais sobre os tipos de valores utilizados pelo Random, consulte o site oficial para desenvolvedores Android Studio, disponível em: https://developer.android.com/reference/java/util/Random.

Verifique no vídeo 2 a construção do método para gerar os valores aleatórios e ocultos do jogo, do método responsável pelas ações e verificações do botão enviar e do método do botão novo.

Vídeo 2 – Desenvolvimento dos métodos do botão enviar e novo. Site: https://youtu.be/4N4lAbXu4H0

Para finalizar o projeto, Karla vai inserir alguns sons durante a execução do projeto, veja no vídeo 3 a execução dessa etapa final. Os áudios utilizados estão disponíveis em: https://drive.google.com/drive/folders/1m0zfgea56TsqeIQAIYhaqO0H-VHzef_y?usp=sharing



Vídeo 3 – Inserir som ao projeto. Site: https://youtu.be/4VdmUGEeWuw

Veja a seguir como ficou o arquivo “main_activity.xml”.

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MainActivity">

    <androidx.appcompat.widget.AppCompatImageView
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:scaleType="centerCrop"
        android:src="@drawable/code"
        />

    <View
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:background="@drawable/transparentepretocinza"
        />

    <EditText
        android:id="@+id/edtValorOculto"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_margin="10dp"
        android:background="@drawable/editgradienteverdepretoverdeicone"
        android:enabled="false"
        android:text="?"
        android:textAlignment="center"
        android:textColor="@android:color/white"
        app:layout_constraintBottom_toTopOf="@id/edtValor"
        app:layout_constraintEnd_toEndOf="parent"
```

```
        app:layout_constraintHorizontal_bias="0.498"
        app:layout_constraintStart_toStartOf="parent" />

    <EditText
        android:id="@+id/edtValor"
        android:layout_width="match_parent"
        android:layout_height="50dp"
        android:layout_margin="10dp"
        android:background="@drawable/editgradienteverdepretoverde"
        android:hint="Digite o valor!"
        android:inputType="number"
        android:textAlignment="center"
        android:textColor="@android:color/white"
        android:textColorHint="@android:color/white"
        app:layout_constraintBottom_toTopOf="@id/btnEnviar"

        />

    <Button
        android:id="@+id/btnEnviar"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        app:layout_constraintBottom_toTopOf="@+id/btnNovo"
        android:text="Enviar"
        android:onClick="clickBtnEnviar"
        android:background="@drawable/botaoverdeoval"
        android:textColor="@android:color/white"
        android:layout_margin="10dp"
        android:textAllCaps="false"
        />

    <Button
        android:id="@+id/btnNovo"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        app:layout_constraintBottom_toTopOf="@+id/txtDica"
        android:text="Novo"
        android:onClick="clickBtnNovo"
        android:background="@drawable/botaoverdeoval"
        android:textColor="@android:color/white"
        android:layout_margin="10dp"
        android:textAllCaps="false"
        />

    <TextView
        android:id="@+id/txtDica"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="Valor de 1 até 10"
        android:textSize="18dp"
        android:textColor="@android:color/white"
        android:textAlignment="center"
        android:layout_margin="10dp"
        app:layout_constraintBottom_toBottomOf="parent"
        />

</androidx.constraintlayout.widget.ConstraintLayout>
```

Veja a seguir como ficou o arquivo “MainActivity.java”.

```
package com.example.jogodescubranumero;

import androidx.appcompat.app.AppCompatActivity;

import android.media.MediaPlayer;
import android.os.Bundle;
import android.view.View;
import android.widget.EditText;
import android.widget.TextView;

import java.util.Random;

public class MainActivity extends AppCompatActivity {

    TextView txtDicaProg;
    EditText edtValorProg;
    EditText edtValorOcultoProg;
    int valorOculto = gerarNumero();
    MediaPlayer mp;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        txtDicaProg = findViewById(R.id.txtDica);
        edtValorProg = findViewById(R.id.edtValor);
        edtValorOcultoProg = findViewById(R.id.edtValorOculto);
    }

    public int gerarNumero()
    {
        Random gerarNumero = new Random();
        int numero = gerarNumero.nextInt(10) + 1;
        return numero;
    }

    public void clickBtnEnviar(View v)
    {
        int valor = Integer.parseInt(edtValorProg.getText().toString());
        if (valor == valorOculto)
        {
            txtDicaProg.setText("Parabéns!");
            edtValorOcultoProg.setText(""+valorOculto);
            mp = MediaPlayer.create(MainActivity.this,R.raw.aplauso);
            mp.start();
        }
        else
        {
            mp = MediaPlayer.create(MainActivity.this,R.raw.erro);
            mp.start();
            if(valor < valorOculto)
            {
                txtDicaProg.setText("O valor oculto é maior que: " + valor);
                edtValorProg.setText("");
```

```
                edtValorProg.requestFocus();
            }
            else
            {
                txtDicaProg.setText("O valor oculto é menor que: " + valor);
                edtValorProg.setText("");
                edtValorProg.requestFocus();
            }
        }
    }

    public void clickBtnNovo(View v)
    {
        txtDicaProg.setText("Valor de 1 até 10");
        edtValorProg.setText("");
        edtValorOcultoProg.setText("?");
        valorOculto = gerarNumero();
    }
}
```

Envie para o seu tutor um vídeo mostrando a tela do celular com o projeto realizado por você na etapa “Você no comando”. Envie também os códigos utilizados. Aproveite para customizar o projeto **“JogoDescubraNumero”** com suas cores e uma imagem escolhida por você.

Para auxiliar no processo de aprendizagem dos temas discutidos nesta aula, seguem abaixo algumas dicas de vídeo e livro que se relacionam com o conteúdo estudado. Estas dicas são muito importantes para você!

Vídeo:

Canal Vinícius Thiengo - MediaPlayer no Android, Entendendo e Utilizando. Vídeo do Youtube.com: https://www.youtube.com/watch?v=Ddb4P9zS8RQ&feature=emb_logo