Introdução ao Android

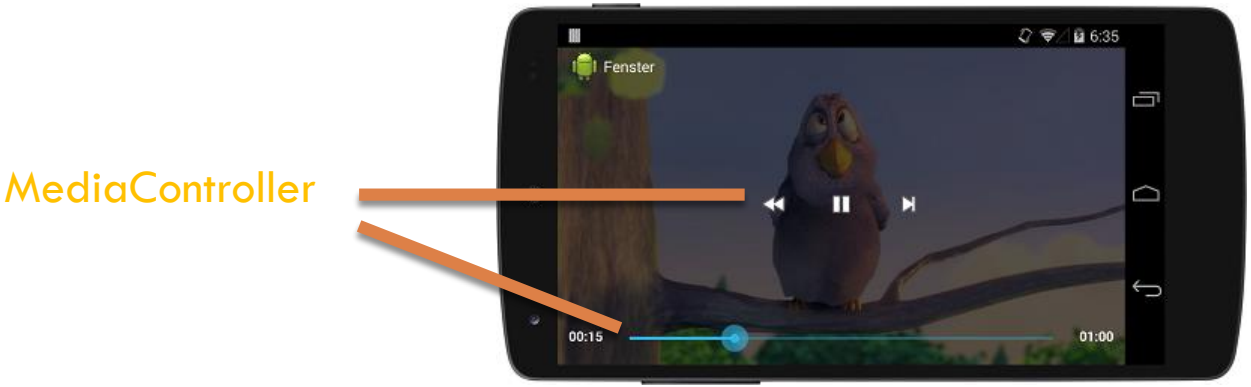
MEDIA PLAYER
SONS E VIDEOS
ROTAÇÃO DO DISPOSITIVO

MediaPlayer

- A Class MediaPlayer, permite reproduzir áudio e vídeo
- Esta class não dispõe de interface, para isso temos de combinar o MediaPlayer com o SuffaceView

VideoView

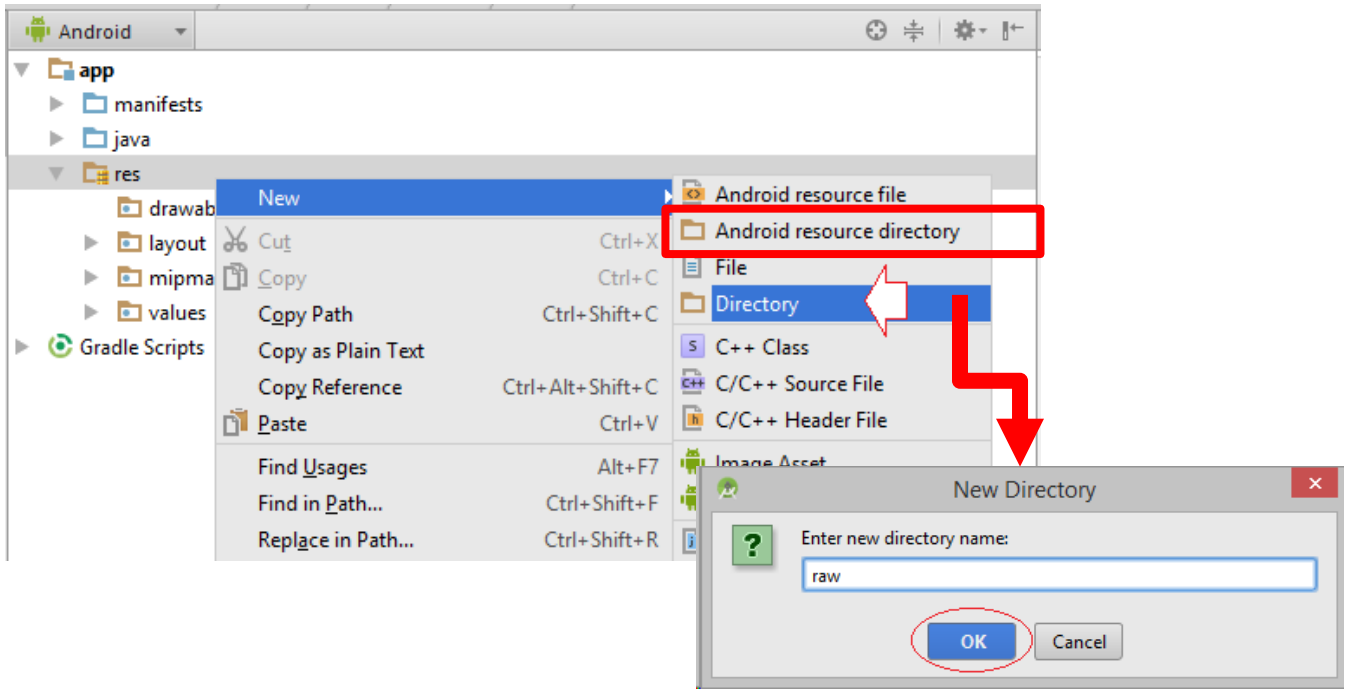
- Para a construção de um player de vídeo, recorremos ao elemento VideoView
- Para aplicar controlos sobre o elemento, devemos de recorrer à implementação da classe MediaController



Pasta raw

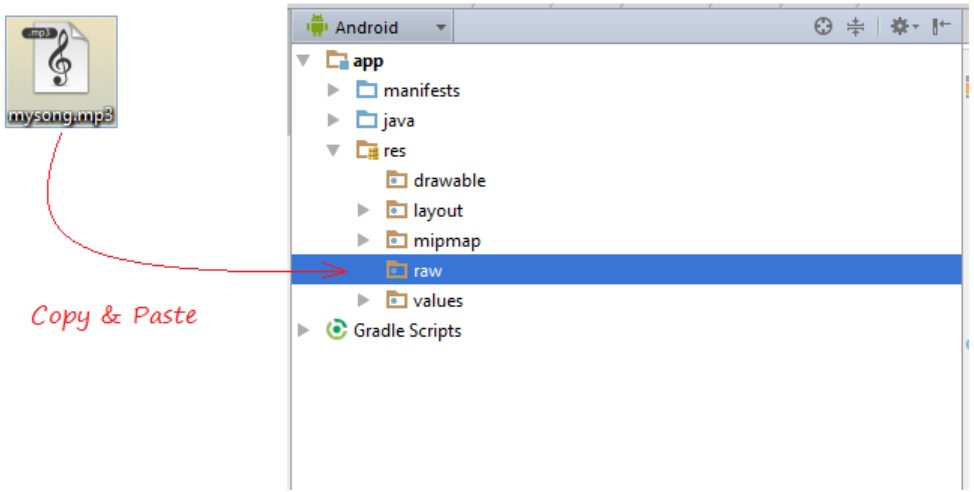
- Para inserir elementos de multimédia numa aplicação android, necessitamos de criar uma pasta 'raw' dentro da pasta 'res', e é nesta pasta que são colocados todos os elementos de multimédia que pretendermos utilizar.
- Por padrão os projetos criados no Android Studio não criam esse diretório, então, o primeiro passo é fazer essa criação

Criar diretório raw

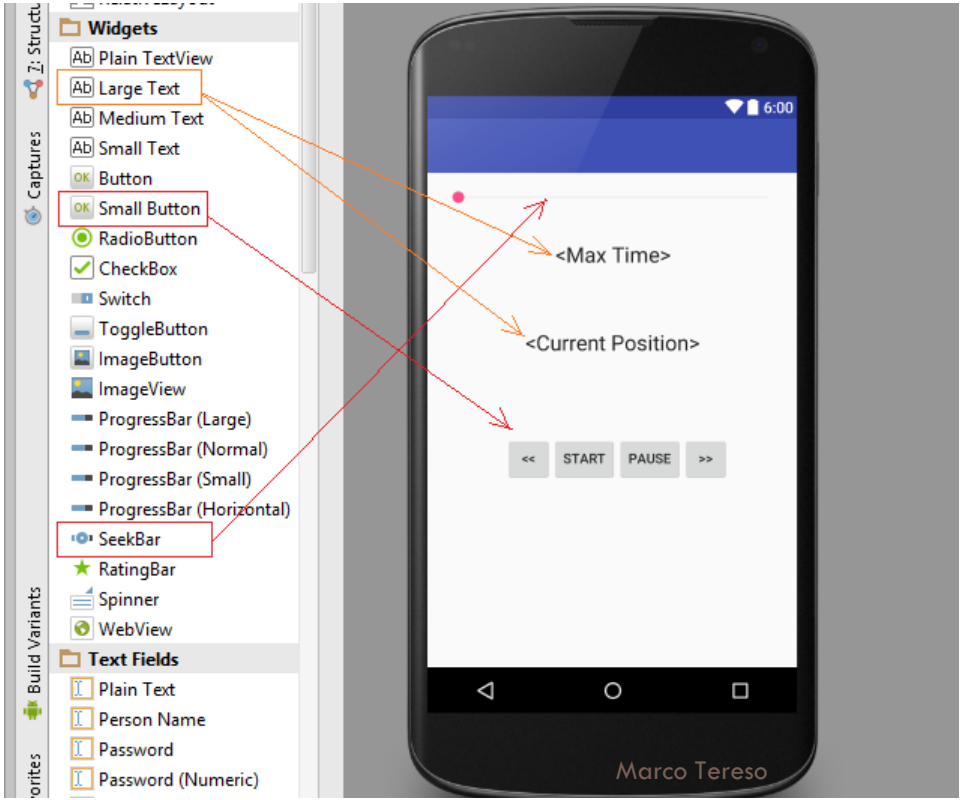


Criar diretório raw

- Também é possível inserir um diretório raw selecionando a opção 'Android resource diretory', assinalada na imagem anterior



Criar um player de música



Declaração dos elementos xml


```
private TextView textMaxTime;
private TextView textCurrentPosition;
private Button buttonPause;
private Button buttonStart;
private SeekBar seekBar;
private Handler threadHandler = new Handler();

private MediaPlayer mediaPlayer;
```

Método onCreate()


```
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);
    //associação dos elementos
    this.textCurrentPosition = (TextView)this.findViewById(R.id.textView_currentPosion);
    this.textMaxTime=(TextView) this.findViewById(R.id.textView_maxTime);
    this.buttonStart= (Button) this.findViewById(R.id.button_start);
    this.buttonPause= (Button) this.findViewById(R.id.button_pause);
    //o botão pause surge desabilitado
    this.buttonPause.setEnabled(false);
    //barra de progresso sem permissões de click
    this.seekBar= (SeekBar) this.findViewById(R.id.seekBar);
    this.seekBar.setClickable(false);

    // selecionar o nome da música a reproduzir, recorrendo a um método getRawResIdByName()
    int songId = this.getRawResIdByName( resName: "som_floresta");

    // Create MediaPlayer.
    this.mediaPlayer=   MediaPlayer.create( context: this, songId);
}
```

Método getRawResIdByName()


```
//encontra o id da música escolhida na pasta raw
public int getRawResIdByName(String resName)  {
    //pega o nome do pacote
    String pkgName = this.getPackageName();
    // devoverá 0 se não encontrar a música na pasta.
    int resID = this.getResources().getIdentifier(resName,  defType: "raw", pkgName);
    //devolve o ID da musica
    return resID;
}
```

Converter unidade temporal em String para apresentar no textView


```
// Converter os milisegundos em String para apresentar no textView
private String millisecondsToString(int milliseconds)  {
    long minutes = TimeUnit.MILLISECONDS.toMinutes((long) milliseconds);
    long seconds =  TimeUnit.MILLISECONDS.toSeconds((long) milliseconds) ;
    return minutes+":"+ seconds;
}
```

Botão start


```
//botão start
public void doStart(View view)  {
    // obter a duração do som
    int duration = this.mediaPlayer.getDuration();
    //obter posição corrente de execução
    int currentPosition = this.mediaPlayer.getCurrentPosition();
    if(currentPosition== 0)  {
        this.seekBar.setMax(duration);
        String maxTimeString = this.millisecondsToString(duration);
        this.textMaxTime.setText(maxTimeString);
    } else if(currentPosition== duration)  {
        // reset para o estado de não iniciado
        this.mediaPlayer.reset();
    }
    this.mediaPlayer.start();
    // atualizar a posição do seek bar
    UpdateSeekBarThread updateSeekBarThread= new UpdateSeekBarThread();
    threadHandler.postDelayed(updateSeekBarThread, delayMillis: 50);
    //ativar os botões de pausa e start
    this.buttonPause.setEnabled(true);
    this.buttonStart.setEnabled(false);
}
```

Atualizar SeekBar


```
// Class que atualiza a posição do seek bar
class UpdateSeekBarThread implements Runnable {

    public void run()  {
        int currentPosition = mediaPlayer.getCurrentPosition();
        String currentPositionStr = millisecondsToString(currentPosition);
        textCurrentPosition.setText(currentPositionStr);

        seekBar.setProgress(currentPosition);
        // avança 50 milisegundos na barra
        threadHandler.postDelayed( r: this,  delayMillis: 50);
    }
}
```

Botão pause


```
// botão pausar
public void doPause(View view)  {
    this.mediaPlayer.pause();
    this.buttonPause.setEnabled(false);
    this.buttonStart.setEnabled(true);
}
```

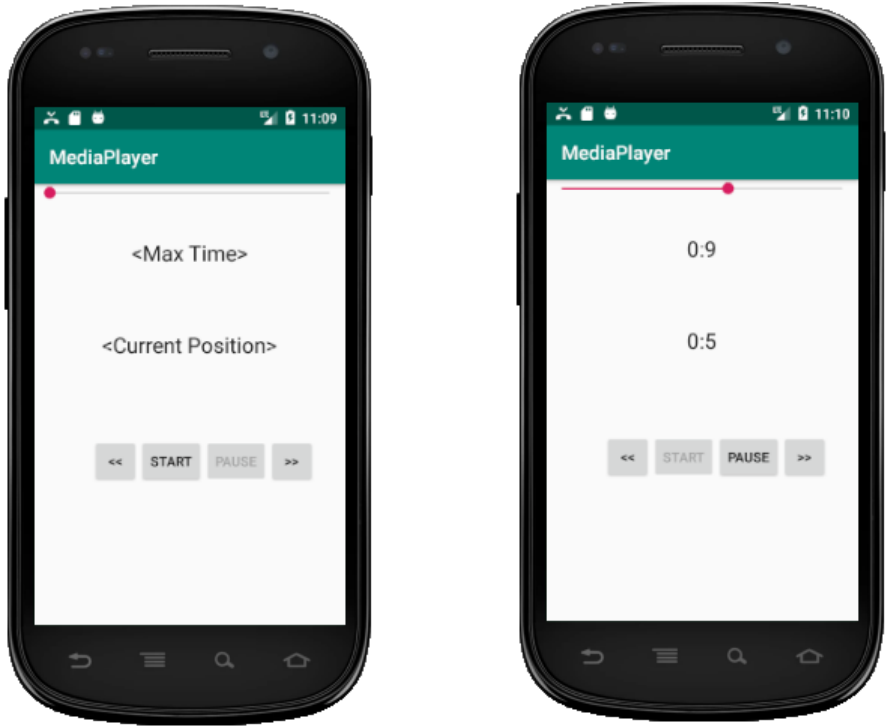
Botão rewind


```
// botão de voltar atrás no tempo de execução
public void doRewind(View view)  {
    int currentPosition = this.mediaPlayer.getCurrentPosition();
    int duration = this.mediaPlayer.getDuration();
    // subtrai 5 segundos.
    int SUBTRACT_TIME = 5000;
    //apenas subtrai 5 segundos quando o tempo atual menos 5 é maior que zero
    if(currentPosition - SUBTRACT_TIME > 0 )  {
        this.mediaPlayer.seekTo( msec: currentPosition - SUBTRACT_TIME);
    }
}
```

Botão forward


```
// Botão andar para a frente.
public void doFastForward(View view)  {
    int currentPosition = this.mediaPlayer.getCurrentPosition();
    int duration = this.mediaPlayer.getDuration();
    // adiciona 5 segundos à posição atual.
    int ADD_TIME = 5000;
    //apenas adiciona 5 segundos quando o tempo atual mais 5 é menor que o total
    if(currentPosition + ADD_TIME < duration)  {
        this.mediaPlayer.seekTo( msec: currentPosition + ADD_TIME);
    }
}
```

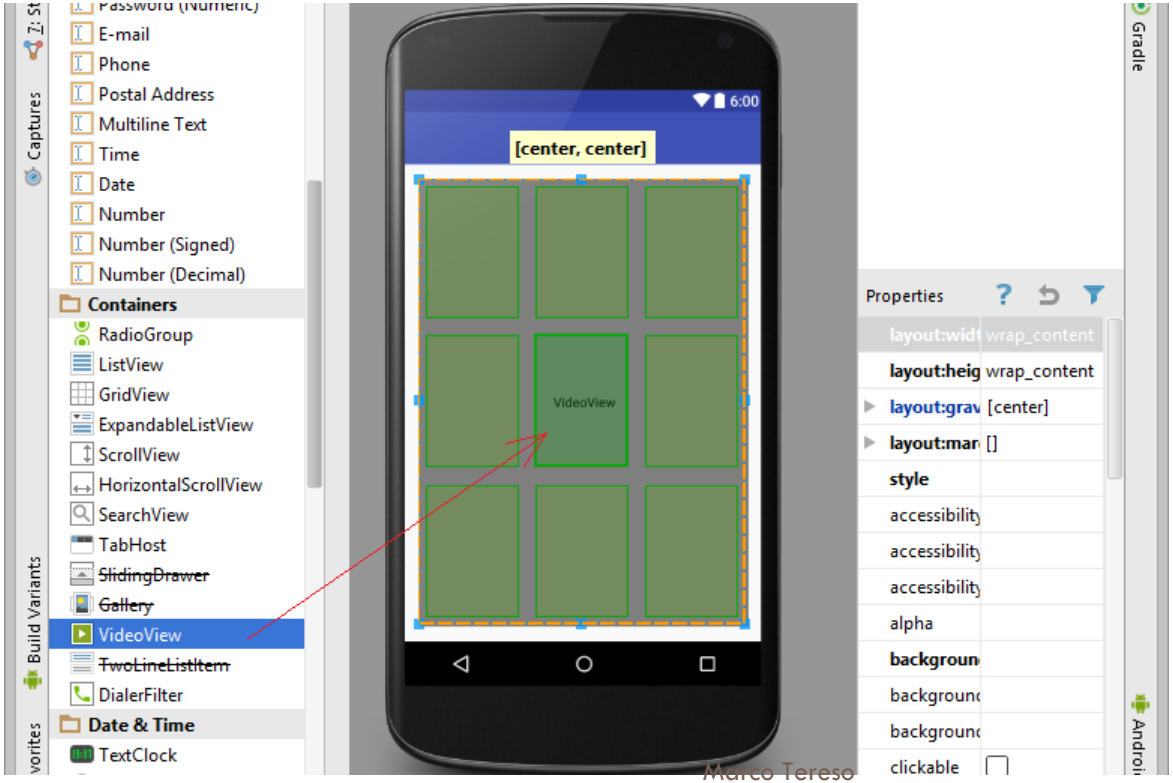
Resultado Final



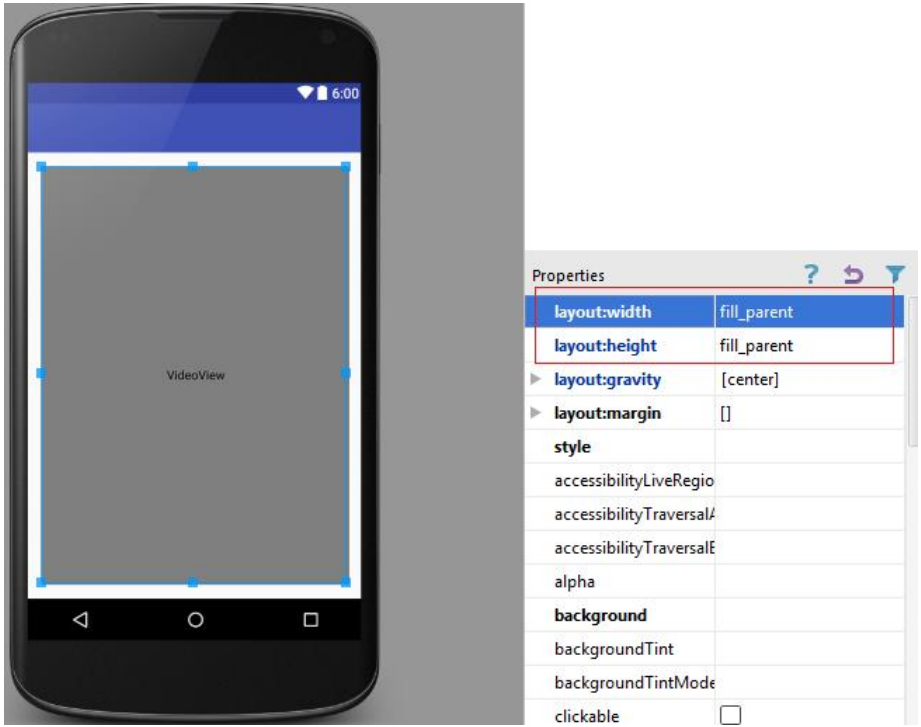
Criar um editor de vídeo

- A visualização de vídeo, segue os mesmos conformes
- O melhor layout para a sua colocação é o FrameLayout

VideoView


E-mail
Phone
Postal Address
Multiline Text
Time
Date
Number
Number (Signed)
Number (Decimal)
Containers
RadioGroup
ListView
GridView
ExpandableListView
ScrollView
HorizontalScrollView
SearchView
TabHost
SlidingDrawer
Gallery
VideoView
TwoLineListItem
DialerFilter
Date & Time
TextClock
6:00
[center, center]
VideoView
Properties
layout:heig wrap_content
layout:grav [center]
layout:mar []
style
accessibility
accessibility
accessibility
alpha
backgroun
backgroun
backgroun
clickable
Gradle
Android
Build Variants
Captures

VideoView


6:00
VideoView
Properties
layout:width
fill_parent
layout:height
fill_parent
layout:gravity
[center]
layout:margin
[]
style
accessibilityLiveRegio
accessibilityTraversalA
accessibilityTraversalB
alpha
background
backgroundTint
backgroundTintMode
clickable

Declaração dos elementos


```
private VideoView videoView;
private int position = 0;
private MediaController mediaController;
```

Método onCreate()


```
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);
    //associar elemento com o xmls
    videoView = (VideoView) findViewById(R.id.videoView);

    // incluir os media controllers no video
    if (mediaController == null) {
        mediaController = new MediaController( context: MainActivity.this);
        // faz linkagem dos controladores para o video
        mediaController.setAnchorView(videoView);
        // coloca os controladores no video
        videoView.setMediaController(mediaController);
    }
    try {
        // ID do ficheiro de video
        int id = this.getRawResIdByName( resName: "tom_jerry");
        videoView.setVideoURI(Uri.parse("android.resource://" + getPackageName() + "/" + id));

    } catch (Exception e) {
        Log.e( tag: "Error", e.getMessage());
        e.printStackTrace();
    }
    videoView.requestFocus();
    // Assim que o video esteja pronto para iniciar
    videoView.setOnPreparedListener(new MediaPlayer.OnPreparedListener() {
```

Método onCreate()


```
        public void onPrepared(MediaPlayer mediaPlayer) {

            videoView.seekTo(position);
            if (position == 0) {
                videoView.start();
            }

            // Quando o video muda o tamanho da tela
            mediaPlayer.setOnVideoSizeChangedListener(new MediaPlayer.OnVideoSizeChangedListener() {
                @Override
                public void onVideoSizeChanged(MediaPlayer mp, int width, int height) {

                    // fazer a colocação dos controladores
                    mediaController.setAnchorView(videoView);
                }
            });
        }
    });

}
```

Encontra o ID do video


```
// Encontra o ID do video
public int getRawResIdByName(String resName) {
    String pkgName = this.getPackageName();
    // devolve 0 senão encontrar
    int resID = this.getResources().getIdentifier(resName, defType: "raw", pkgName);
    Log.i( tag: "AndroidVideoView", msg: "Res Name: " + resName + "==> Res ID = " + resID);
    return resID;
}
```

Se rodar o telefone (antes)


```
//Antes de rodar o telefone
// Guarda a posição corrente do video
@Override
public void onSaveInstanceState(Bundle savedInstanceState) {
    super.onSaveInstanceState(savedInstanceState);

    // guarda a posição corrente.
    savedInstanceState.putInt("CurrentPosition", videoView.getCurrentPosition());
    videoView.pause();
}
```

Se rodar o telefone (depois)


```
// Após rodar o telefone é chamado este método
@Override
public void onRestoreInstanceState(Bundle savedInstanceState) {
    super.onRestoreInstanceState(savedInstanceState);

    // obter a posição guardada
    position = savedInstanceState.getInt( key: "CurrentPosition");
    videoView.seekTo(position);
}
```

Resultado final