

صوت و تصویر در اندروید:

برای پخش صدا ابتدا یک دکمه با رویداد `onclick` در فایل `activity_main.xml` با نام تابع `btnPlay_Click` میسازیم و در داخل `main activity` آن تابع را تعریف میکنیم.

```
<Button
    android:id="@+id/button5"
    android:onClick="btnPlay_Click"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:text="Play" />
```

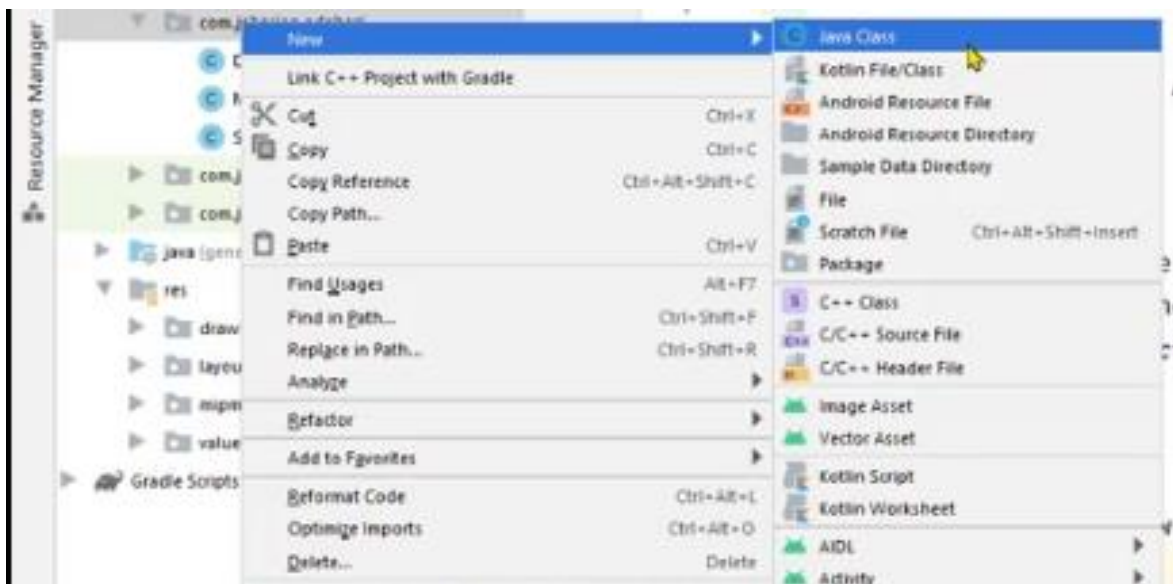
1

```
public void btnPlay_Click(View view) {
    |
}
```

2

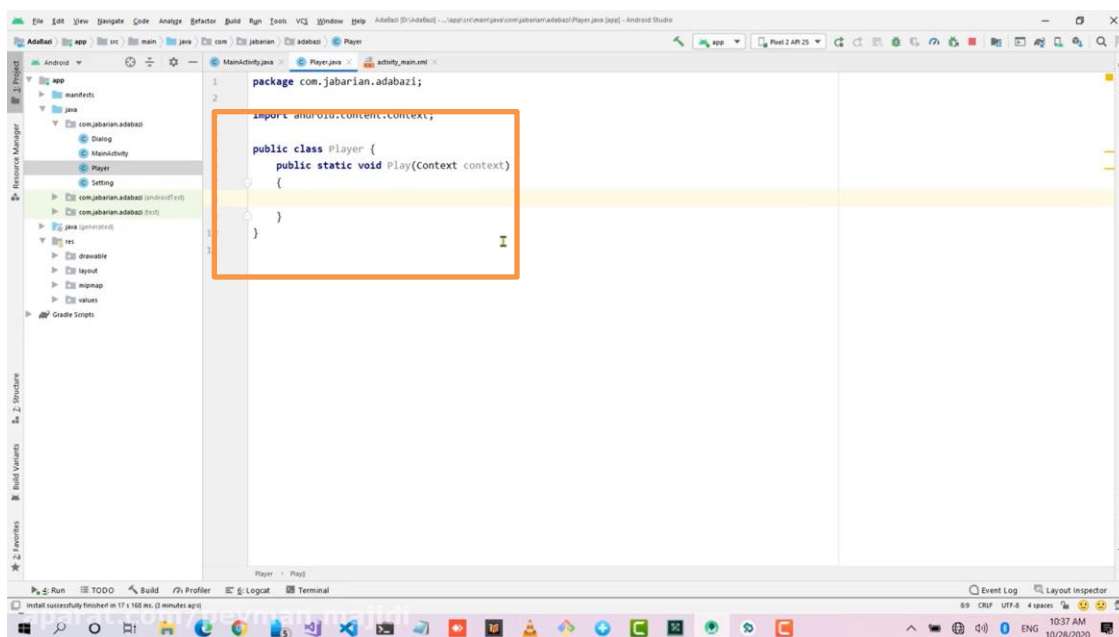
برای طبقه بندی کد های برنامه و نظم بیشتر یک کلاس جدید بوجود می آوریم.

نکته : کلمه `class` کوتاه شده کلمه `classification` به معنی طبقه بندی است.



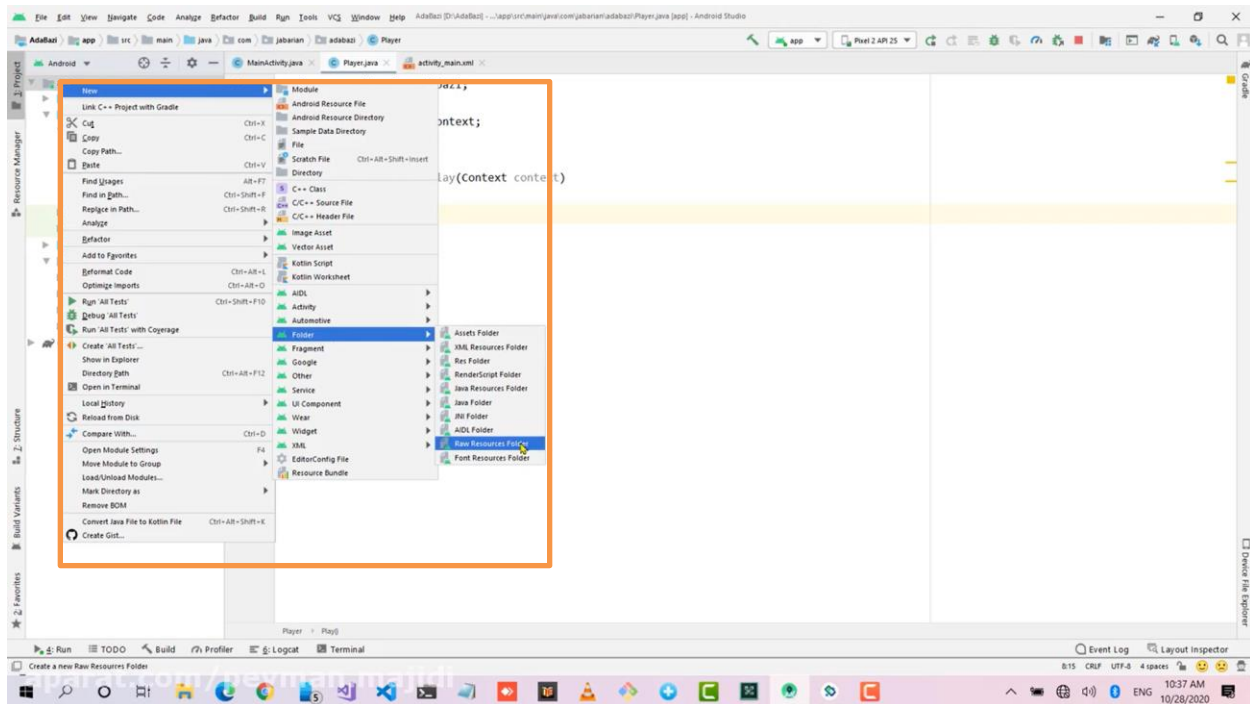
3

یک کلاس **public** به نام **player** ایجاد میکنیم که بتوانیم از فایل **main activity** به آن دسترسی داشته باشیم. و داخل این کلاس یک متد (تابع) به نام **play** تعریف میکنیم و اون رو بصورت **void** (بدون مقدار برگشتی) قرار میدیم. به عنوان ورودی برای این تابع مقدار **context** را پاس میدهیم.



4

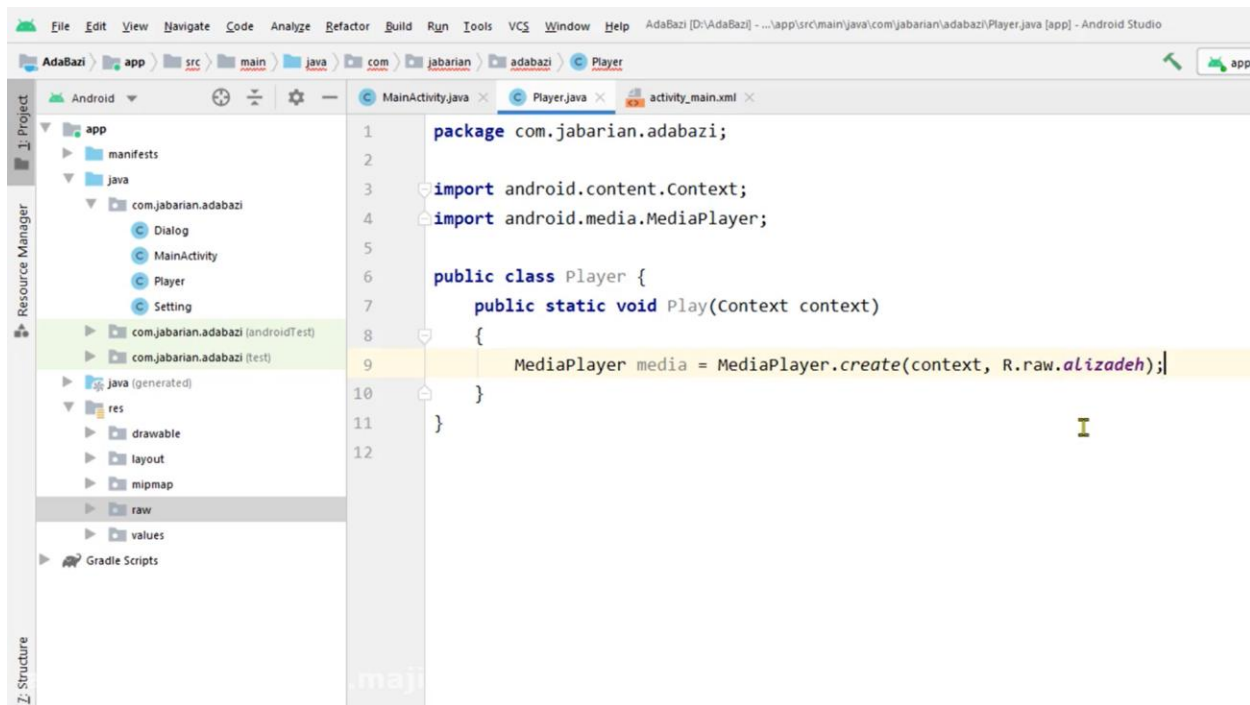
در پروژه های اندروید فایل هایی مثل موزیک باید در یک نوع پوشه به عنوان **raw** قرار بگیرند. برای ذخیره سازی فایل های **raw** پوشه بندی استاندارد وجود دارد که در تصویر زیر میبینید.



5

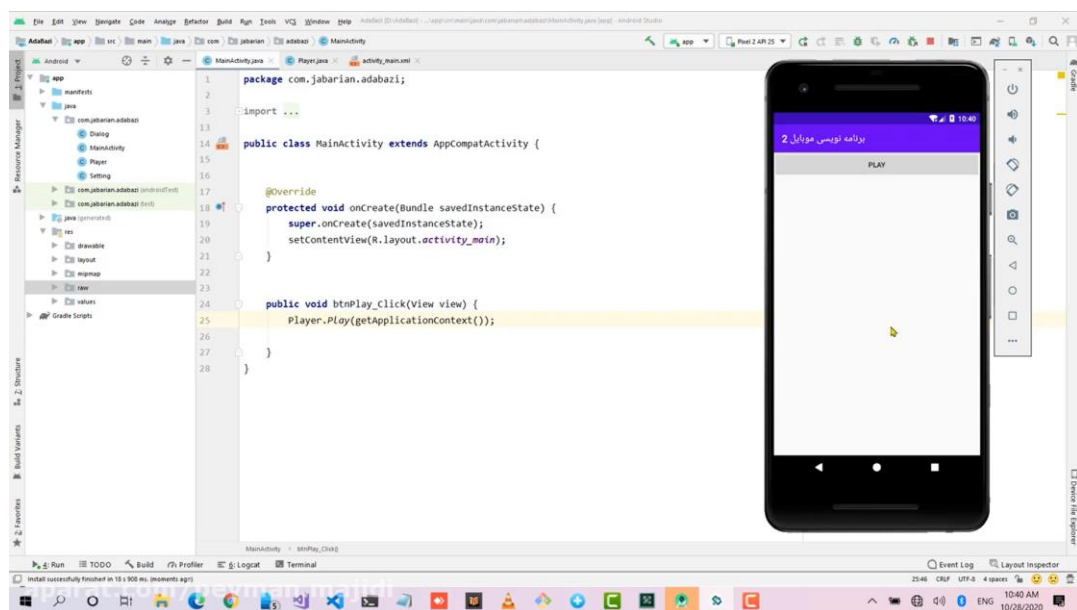
وقتی از این مسیر اقدام به ساخت پوشه میکنیم در پوشه **res** یک پوشه بانام **raw** ایجاد میشود و میتوانیم فایل های **mp3** را در آن قرار بدهیم. برای ذخیره سازی موزیک ها و فیلم ها باید حروفی که برای ذخیره سازی انتخاب میکنیم همه حروف کوچک باشند.

در ادامه برای پخش موزیک از شیء **mediaplayer** یک نمونه جدید با نام **media** میسازیم و آن را برابر با تابع **create()** شیء **mediaplayer** قرار میدهیم. در تابع **create** دو ورودی باید قرار بدهیم: اول نام **context** و آدرس پخش فایل که هم میتواند بصورت **url** و هم بصورت آدرس محلی باشد.



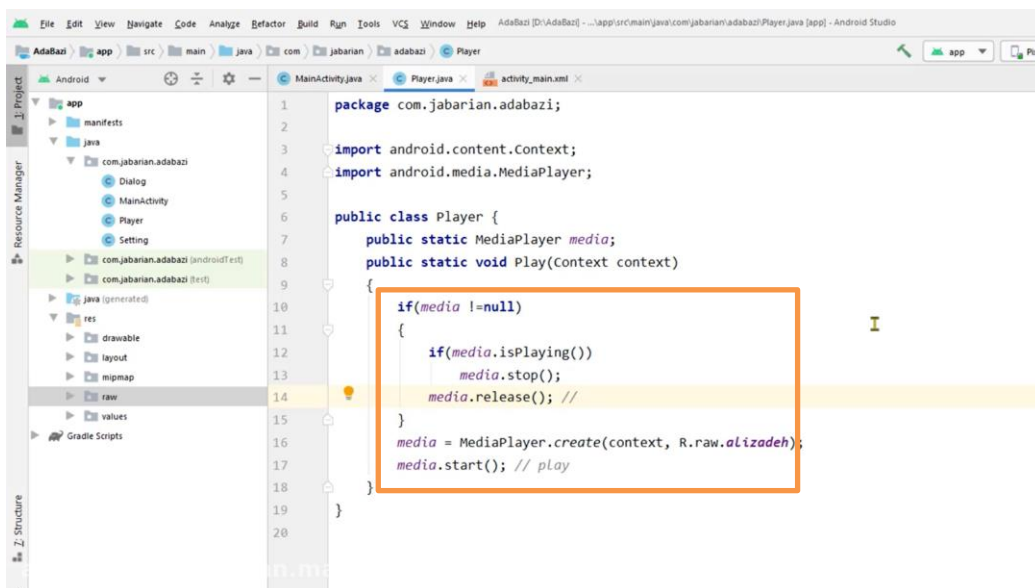
6- اجرای فایل بصورت محلی

برای play کردن فایل ابتدا خود media را صدا میزنیم و بعد تابع start را صدا میزنیم. سپس داخل فایل main_activity تابع play از کلاس player را صدا میزنیم. داخل این تابع، تابع getApplicationContext() را به عنوان ورودی پاس می‌دهیم.



7- با کلیک روی دکمه play شاهد پخش موزیک هستیم

اگر چند بار روی play کلیک کنیم متوجه میشویم که چند خط اجرای موزیک همزمان کار میکنند و هیچکدام متوقف نمیشود. برای حل این مشکل ابتدا تعریف media را داخل تابع خارج میکنیم و آنرا بصورت global داخل کلاس معرفی میکنیم. داخل یک شرط میگیریم که اگر دوباره روی این دکمه کلیک شد و media خالی نبود پخش media متوقف شود.



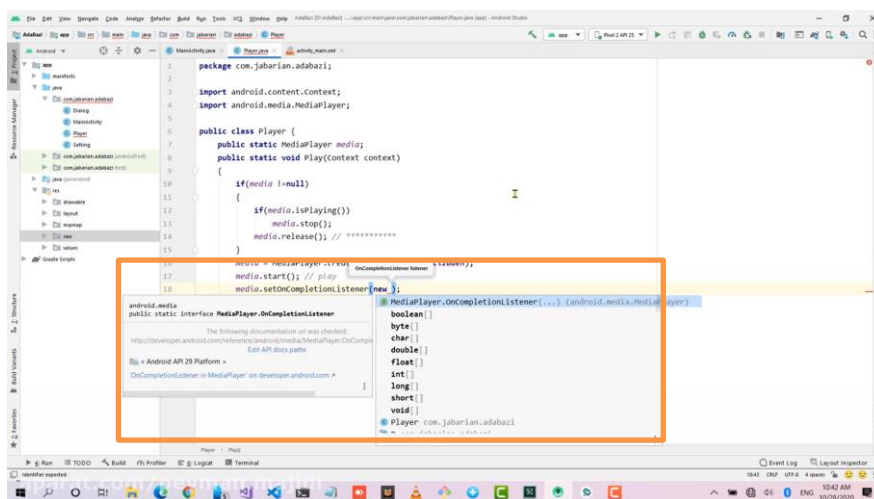
```

1 package com.jabarian.adabazi;
2
3 import android.content.Context;
4 import android.media.MediaPlayer;
5
6 public class Player {
7     public static MediaPlayer media;
8     public static void Play(Context context)
9     {
10         if(media != null)
11         {
12             if(media.isPlaying())
13                 media.stop();
14             media.release(); //
15         }
16         media = MediaPlayer.create(context, R.raw.alizadeh);
17         media.start(); // play
18     }
19 }
20

```

8-دستور media.release() برای پاک کردن media از حافظه

برای اینکه بعد از اتمام موسیقی هم این دستور اجرا شود از تابع setOnClickListener() استفاده میکنیم و داخل تابع برای جلوگیری از نوشتن کد تکراری بعد از نوشتن کلمه new خود برنامه گزینه رویداد mediaPlayer.oncompletionlistener را پیشنهاد میدهد و روی آن کلیک میکنیم

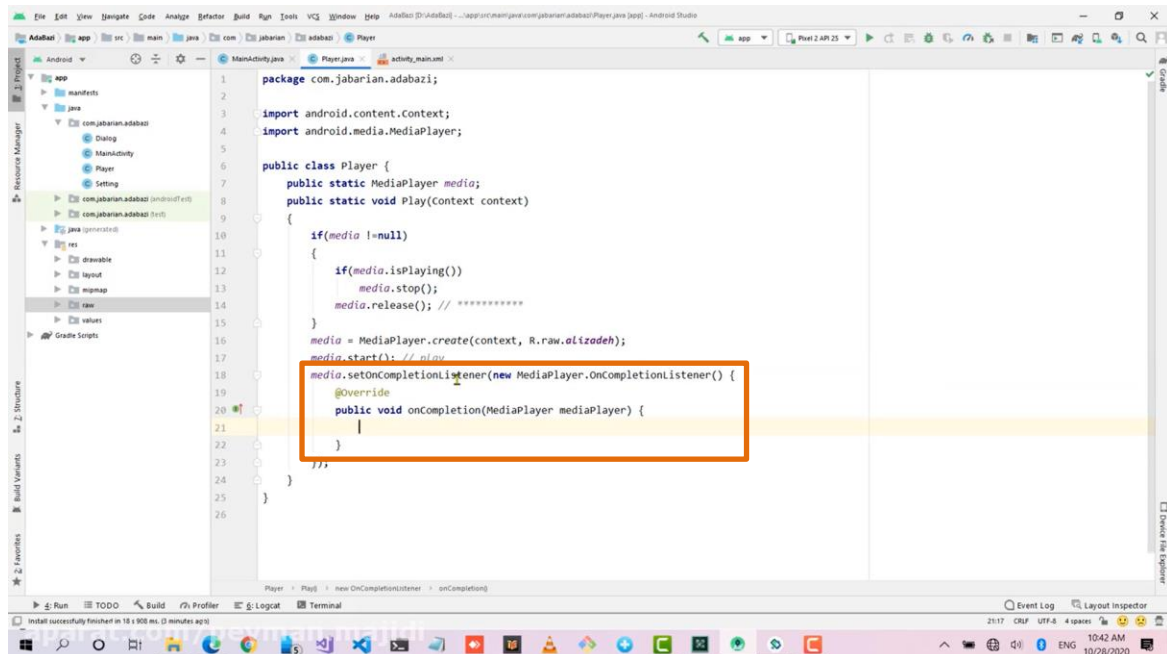


```

1 package com.jabarian.adabazi;
2
3 import android.content.Context;
4 import android.media.MediaPlayer;
5
6 public class Player {
7     public static MediaPlayer media;
8     public static void Play(Context context)
9     {
10         if(media != null)
11         {
12             if(media.isPlaying())
13                 media.stop();
14             media.release(); //
15         }
16         media = MediaPlayer.create(context, R.raw.alizadeh);
17         media.start(); // play
18         media.setOnCompletionListener(new MediaPlayer.OnCompletionListener() {
19             // ...
20         });
21     }
22 }
23

```

و بعد این قطعه کد را خواهیم داشت:



بعد لازم است که دستور `media.release()` را در داخل متد `onCompletion` هم قرار بدهیم که بعد از پایان موزیک هم این موزیک از حافظه موقت خارج شود. سپس توابع `stop` و `pause` را هم از روی همین تابع کپی میکنیم.

```

public static void Stop()
{
    if(media != null)
    {
        if(media.isPlaying())
        {
            media.stop();
            media.release(); // *****
        }
    }
}

public static void Pause()
{
    if(media != null)
    {
        if(media.isPlaying())
        {
            media.pause();
        }
    }
}

```

برای جلوگیری از تکرار کد تابع `play` را دست خوش تغییر میکنیم.

تابع `media.stop` را از تابع `play` حذف میکنیم. داخل یک شرط به برنامه میگوییم که اگر موزیک قبلاً شروع به پخش شده بود از شرط با کلمه `return` خارج شو و اگر نه که یعنی موزیک قبلاً `pause` شده است، دوباره شروع به پخش موزیک کن.

```

if(media != null)
{
    if(media.isPlaying())
    {
        return;
    }
    media.start();
    return;
}

```


درنهایت برای تست متد ها باید دو دکمه ایجاد کنیم و در رویداد **onClick** آنها توابع را به آنها پاس بدهیم.

تابع **media.seekTo()** :

برای اینکه بخواهیم به یک جای دلخواه از موزیک برویم از این تابع استفاده میکنیم . این تابع یک ورودی زمانی بر اساس میلی ثانیه میگیرد.

استفاده از صداهای استاندارد اندروید در برنامه :

برای استفاده از صداهای استاندارد اندروید ابتدا یک تابع با نام **beep** ایجاد میکنیم. سپس یک **mediaActionSound** با نام **sound** ایجاد میکنیم و آنرا برابر با نمونه جدید از این کلاس میکنیم.

```
MediaActionSound sound = new MediaActionSound();
```

در مرحله بعد از متد **sound.play** استفاده میکنیم و به عنوان ورودی به آن **mediaActionSound** را پاس میدهیم و از داخل این آبجکت صدای مورد نظرمون رو که ما در این مثال از صدای پایان ضبط کردن استفاده کردیم انتخاب میکنیم.

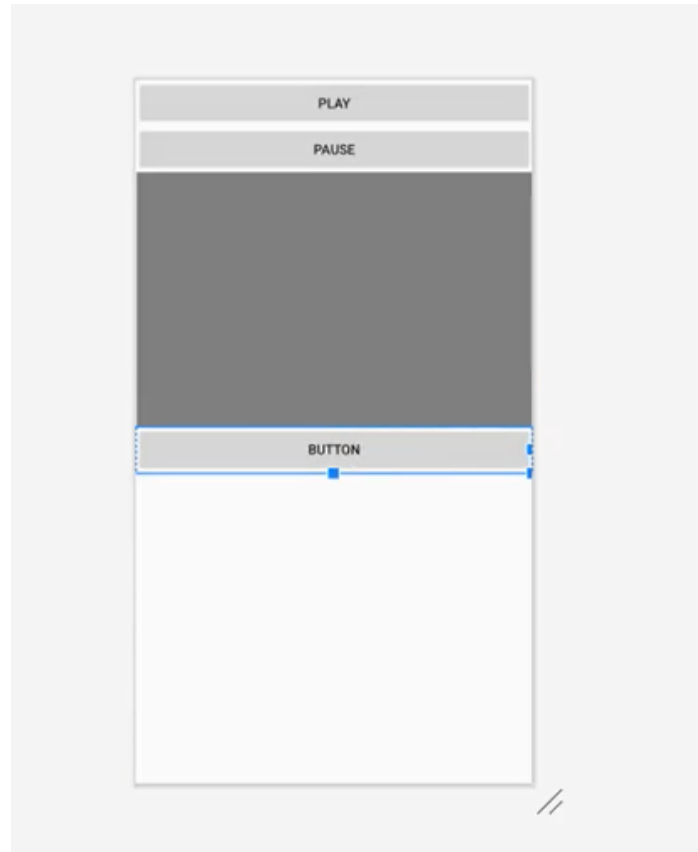
```
public static void Beep()  
{  
    MediaActionSound sound = new MediaActionSound();  
    sound.play(MediaActionSound.STOP_VIDEO_RECORDING); // beeeeeeeeeep  
}
```

در مرحله آخر از **main_activity** ابتدا کلاس و سپس متد کلاس رو صدا میزنیم.

```
public void btnPause_Click(View view) {  
    Player.Pause();  
    Player.Beep();  
}
```

آموزش پخش ویدیو به وسیله `videoView` :

ابتدا در کامپوننت های آماده اندروید استودیو کامپوننت اندروید استودیو را انتخاب و یک دکمه هم برای پخش ویدیو استفاده میکنیم.



نمایی از کامپوننت های موجود آمده

داخل فایل `main_activity` میایم یک متد با نام برای رویداد کلیک تعریف میکنیم و از طریق `xml` این متد را به برنامه متصل میکنیم.

باید توجه داشته باشیم که استاندارد پخش ویدیو با کدک `h.264` باید در ویدئو ویو قرارداده بشود تا پخش شود. غیر از این استاندارد ممکن است پخش ویدیو با مشکل مواجه شود.


```

public void btnPlayVideo_Click(View view) {
    VideoView video = findViewById(R.id.videoView);
    Uri uri = Uri.parse("android.resource://" + getPackageName() + "/" + R.raw.mask );
    video.setVideoURI(uri);
    video.start(); // play
}

```

در ادامه ارتباط ویدیو ویو را با دستور **findbyid** با فایل **main_activity** برقرار میکنیم و بعد یک **uri** با همین نام تعریف میکنیم تا به برنامه آدرس ویدیو را برای پخش اعلام کنیم و با دستور **setVideoURI()** این **uri** تعریف شده را به آن به عنوان ورودی پاس میدهیم.

و در آخر هم از دستور **video.start** پخش ویدیو استفاده میکنیم.