آموزش برنامه نویسی اندروید
در محیط اندروید استودیو

کار با دوربین
بخش اول: استفاده از برنامه داخلی دوربین

مدرس : سیدمهدی مطهری

www.android-studio.ir

به نام خدا

در این جلسه از مباحث آموزش برنامه نویسی اندروید به آموزش کار با دوربین در برنامه نویسی اندروید می‌پردازیم که در آن از برنامه پیش فرض دوربین موجود روی دستگاه (گوشی یا تبلت) برای گرفتن تصویر و انتقال آن به اپلیکیشن اصلی استفاده شده است.

قطعا نیازی به معرفی قابلیت‌های دوربین و موارد کاربرد آن در گوشی‌های موبایل و تبلت‌ها نیست! همه ما روزانه با این قسمت از دیوایس خود سروکار داریم و برای گرفتن عکس، اسکن QR Code یا بارکد قبوض و... نیازمند وجود سخت افزار دوربین روی دیوایس هستیم. امروزه تقریبا تمامی دستگاه‌های اندرویدی و در سطح بالاتر، تمامی تلفن‌های همراه هوشمند و همچنین تبلت‌ها از حداقل یک دوربین برخوردار هستند. درصد زیادی از این دستگاه‌ها علاوه بر دوربین اصلی که در پشت قرار دارد، دارای یک دوربین جانبی نیز هستند که اصطلاحا دوربین سلفی نامیده می‌شودکه در جلوی دستگاه تعبیه شده است.

روش‌های استفاده از دوربین در برنامه نویسی اندروید

بطور کلی برای بکارگیری دوربین دیوایس‌های اندرویدی دو راهکار وجود دارد:

۱- **برنامه واسط:** استفاده از یک برنامه واسط برای گرفتن تصویر یا ویدئو و سپس انتقال آن به اپلیکیشنی که به خروجی آن نیاز دارد.
۲- **API:** استفاده مستقیم از دوربین بدون دخالت برنامه‌های واسط توسط API دوربین (Camera API) که در سیستم عامل اندروید تعبیه شده.

همه ما برای گرفتن تصویر و یا ضبط ویدئو از اپلیکیشن پیش فرض دوربین دیوایس خود و یا یک برنامه جانبی مشابه آن استفاده می‌کنیم. این برنامه‌ها جزء دسته دوم هستند؛ یعنی به طور مستقیم از API دوربین اندروید استفاده می‌کنند. زیرا عملیات ضبط تصاویر و ویدئو در محیط همان برنامه انجام می‌شود و نیاز به انتقال به برنامه دیگری نیست. اینستاگرام را می‌توان به عنوان یک نمونه دیگر ذکر کرد که ثبت تصویر یا ویدئو درون خود برنامه انجام می‌پذیرد.

اما در این مبحث فعلا به دسته دوم و کار با API کاری نداریم و صرفا یک برنامه ساده می‌نویسیم که برای گرفتن تصویر توسط یک Intent به برنامه دوربین گوشی منتقل شده و پس از گرفتن عکس، نتیجه را به اپلیکیشن ما برمی‌گرداند.

کار با دوربین در اندروید توسط برنامه پیش فرض

یک پروژه جدید با نام Camera و یک اکتیویتی از نوع Blank Activity در اندروید استودیو ایجاد می‌کنم. در بخش رابط کاربری برنامه به یک دکمه یا Button و یک ImageView نیاز دارم.

در قدم اول یک تگ <uses-feature> به مانیفست پروژه اضافه می‌کنم:

```
<uses-feature android:name="android.hardware.camera" android:required="true" />
```

در صورتی که با این تگ آشنایی ندارید مبحث کاربرد تگ <uses-feature> را مطالعه کنید. با تعریف خط فوق در مانیفست پروژه اندرویدی، به فروشگاه Google Play و سایر فروشگاه‌هایی که این تگ را بررسی می‌کنند اعلام می‌کنیم وجود سخت افزار دوربین (android.hardware.camera) برای کارایی این اپلیکیشن ضروری است بنابراین این اپ برای دیوایس‌هایی که از این قابلیت پشتیبانی نمی‌کنند در لیست برنامه‌ها نمایش داده نشود.

AndroidManifest.xml:

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="ir.android_studio.camera">

    <uses-feature android:name="android.hardware.camera" android:required="true" />

    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="@string/app_name"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:supportsRtl="true"
        android:theme="@style/AppTheme">
        <activity android:name=".MainActivity">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
    </application>

</manifest>
```

نکته: در این پروژه ما به مجوز دسترسی (Permission) دوربین نیازی نداریم زیرا همانطور که اشاره شد، این برنامه از یک برنامه ثانویه برای گرفتن تصویر استفاده می‌کند. زمانی نیاز به اخذ مجوز دسترسی به دوربین از کاربر را داریم که بخواهیم به طور مستقیم و به واسطه API از سخت افزار مدنظر استفاده کنیم.

در مرحله بعد layout اکتیویتی را به اینصورت ویرایش می‌کنم:

:activity_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout 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"
    android:orientation="vertical"
    tools:context=".MainActivity">


    <ImageView
        android:id="@+id/imageView"
        android:layout_width="match_parent"
        android:layout_height="400dp"
        android:src="@drawable/ic_launcher_background" />

    <Button
        android:id="@+id/btn_photo"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="Take Picture" />
</LinearLayout>
```

یک ImageView با شناسه imageView و Button با شناسه btn_photo. برای لی اوت همین کافیست.

به سراغ فایل جاوای اکتیویتی می‌روم. ابتدا دو ویجتی که در layout قرار گرفته را اینجا تعریف می‌کنم:

:MainActivity.java

```
package ir.android_studio.camera;

import androidx.appcompat.app.AppCompatActivity;

import android.os.Bundle;
import android.widget.Button;
import android.widget.ImageView;

public class MainActivity extends AppCompatActivity {

    ImageView imgView;
    Button btnPhoto;

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

        imgView = findViewById(R.id.imageView);
        btnPhoto = findViewById(R.id.btn_photo);

    }
}
```

سپس برای دکمه‌ی btnPhoto درون متد onCreate یک رویداد setOnClickListener تعریف می‌کنم تا با لمس یا کلیک روی آن، صفحه‌ی مربوط به اپلیکیشن دوربین باز شود:

```
btnPhoto.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View view) {

        Intent camIntent = new Intent(MediaStore.ACTION_IMAGE_CAPTURE);
        startActivityForResult(camIntent, CameraRequest);

    }
});
```

قبلا در مبحث آموزش intent در اندروید با اینتنت‌ها آشنا شدیم. یک آبجکت (شیء) از Intent با نام دلخواه camIntent ساختم.

جهت برقراری ارتباط بین اپلیکیشن خودمان و برنامه‌ای که خروجی مدنظر را برای ما تهیه می‌کند؛ یعنی برنامه دوربین، از کلاس MediaStore استفاده می‌کنیم. محتوای خروجی می‌تواند شامل عکس، ویدئو و صوت باشد که در اینجا ما به دریافت عکس نیاز داریم. MediaStore زیرمجموعه‌ی android.provider است.

اکشن مدنظر ما گرفتن عکس و ارسال آن به برنامه است. بنابراین از ACTION_IMAGE_CAPTURE استفاده شده. در خط بعد متد startActivityForResult() تعریف شده که دو پارامتر ورودی می‌گیرد. ورودی اول intent ای که قبلا تعریف شده و مورد دوم یک "کد درخواست" از جنس int می‌باشد.

لغت Result به معنی نتیجه و حاصل است بنابراین کاربرد این متد از نام آن مشخص می‌شود؛ استارتِ اکتیویتی برای برگرداندن یک نتیجه. این متد توسط ورودی اول تشخیص می‌دهد باید دنبال چه نوع اکتیویتی باشد. ورودی اول intent ای از نوع MediaStore.ACTION_IMAGE_CAPTURE است بنابراین این متد ما را به سمت یک اکتیویتی در یک برنامه‌ای هدایت خواهد کرد که قابلیت گرفتن عکس را داشته باشد.

ورودی دوم همانطور که اشاره شد یک عدد دلخواه است. به جهت اینکه این کد در ادامه کار برای برگرداندن نتیجه استفاده می‌شود، یک متغیر از جنس int در بدنه اصلی کلاس اکتیویتی با نام دلخواه CameraRequest و با مقدار دلخواه "1" اضافه کردم:

```
protected static final int CameraRequest = 1;
```

مقدار این متغیر همواره ثابت بوده و نباید تغییر کند، همچنین فقط در همین کلاس فراخوانی می‌شود بنابراین آنرا به صورت protected static final تعریف کرده‌ام.

کد مربوط به رویداد دکمه برنامه کامل شد.

تا اینجای کار، با کلیک روی دکمه، کاربر از اپلیکیشن ما به اکتیویتی اپلیکیشنی منتقل می‌شود که قابلیت دریافت تصویر از دوربین را داشته باشد.

:MainActivity.java

```
package ir.android_studio.camera;

import androidx.appcompat.app.AppCompatActivity;
import android.content.Intent;
import android.os.Bundle;
import android.provider.MediaStore;
import android.view.View;
import android.widget.Button;
import android.widget.ImageView;

public class MainActivity extends AppCompatActivity {

    ImageView imgView;
    Button btnPhoto;
    protected static final int CameraRequest = 1;

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

        imgView = findViewById(R.id.imageView);
        btnPhoto = findViewById(R.id.btn_photo);

        btnPhoto.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {

                Intent camIntent = new Intent(MediaStore.ACTION_IMAGE_CAPTURE);
                startActivityForResult(camIntent, CameraRequest);

            }
        });

    }

}
```

در قدم بعد باید داده‌ی مدنظر یعنی تصویری که کاربر گرفته را به اپلیکیشن برگردانده و درون ImageView نمایش دهیم. بدین منظور درون کلاس MainActivity و بعد از متد ()onCreate یک متد دیگر با نام ()onActivityResult اضافه می‌کنم:

```
protected void onActivityResult(int requestCode, int resultCode, Intent data) {



}
```

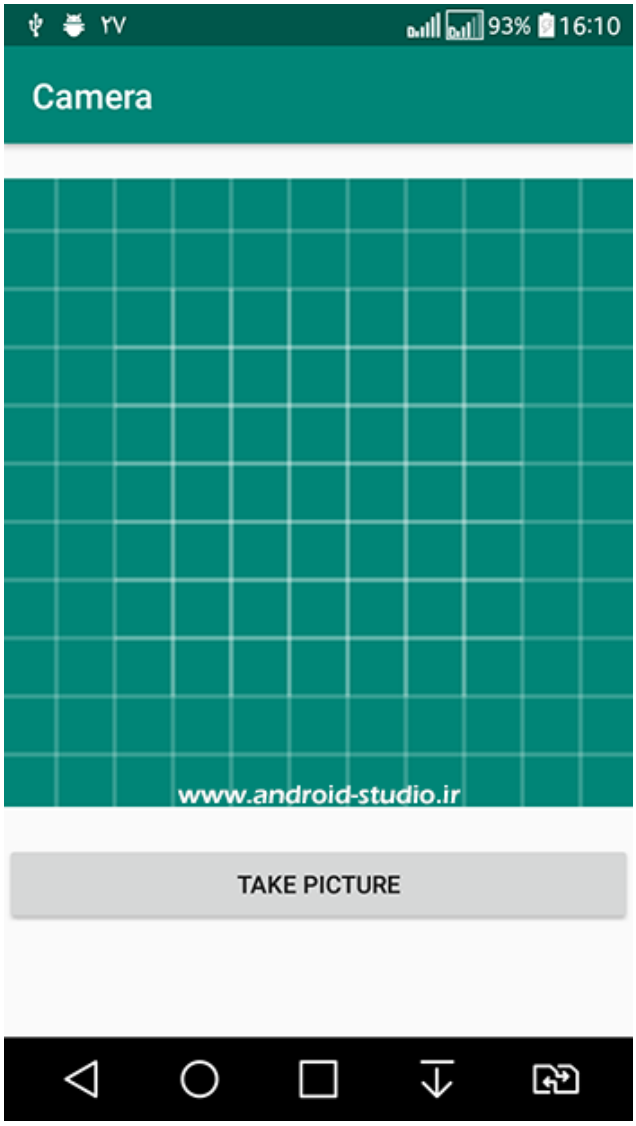
برخلاف متد onCreate() که در زمان ساخته شدن اکتیویتی اجرا می‌شود، متد onActivityResult() زمانی در اندروید اجرا خواهد شد که از اکتیویتی فعلی به اکتیویتی دیگری منتقل شده، نتیجه (Result) را گرفته و به اکتیویتی اول برگشته‌ایم. بنابراین بعد از آنکه کاربر تصویر موردنظر خود را گرفت، بلافاصله به MainActivity برنامه منتقل شده و کدهای درون متد onActivityResult() اجرا می‌گردد. این متد سه پارامتر requestCode، resultCode و data دارد که در اینجا به موارد اول و سوم نیاز داریم. متد را به اینصورت تکمیل می‌کنم:

```
protected void onActivityResult(int requestCode, int resultCode, Intent data) {

    if (requestCode == CameraRequest) {

        Bitmap resultPhoto = (Bitmap) data.getExtras().get("data");
        imgView.setImageBitmap(resultPhoto);

    }

}
```

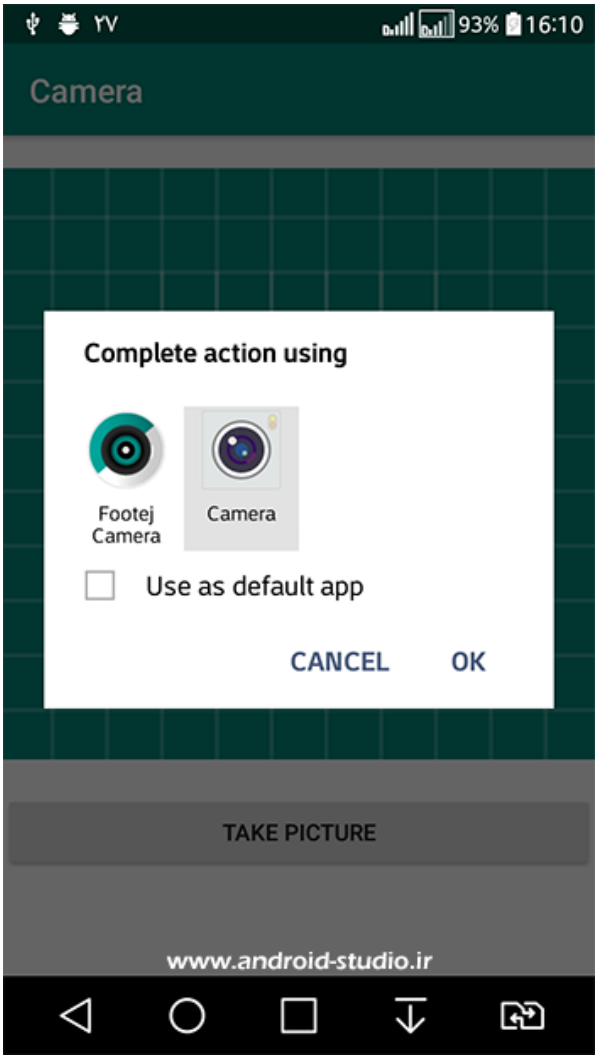
در ابتدا یک شرط تعریف شده به اینصورت که چک شود اگر requestCode با CameraRequest برابر بود آنگاه کد درون بلاک را اجرا کن. در واقع requestCode حاوی کدی است که قبلا توسط متد startActivityForResult() در متد onCreate() به اکتیویتی دوربین ارسال شده و حالا توسط onActivityResult دریافت شده است. به این ترتیب مطمئن می‌شویم در حال مدیریت نتیجه‌ای هستیم که با کد درخواست "1" ارسال شده بود.

تصویر دریافتی را می‌بایست به داده‌ای از جنس Bitmap تبدیل کرد تا بتوان آنرا درون ImageView نمایش داد. یک آبجکت از Bitmap با نام resultPhoto ساخته‌ام که اطلاعات دریافتی با کلید data یعنی get("data") را در خود ذخیره می‌کند. در خط بعد توسط setImageBitmap داده‌ی دریافتی در قالب تصویر به imgView ارسال می‌گردد.

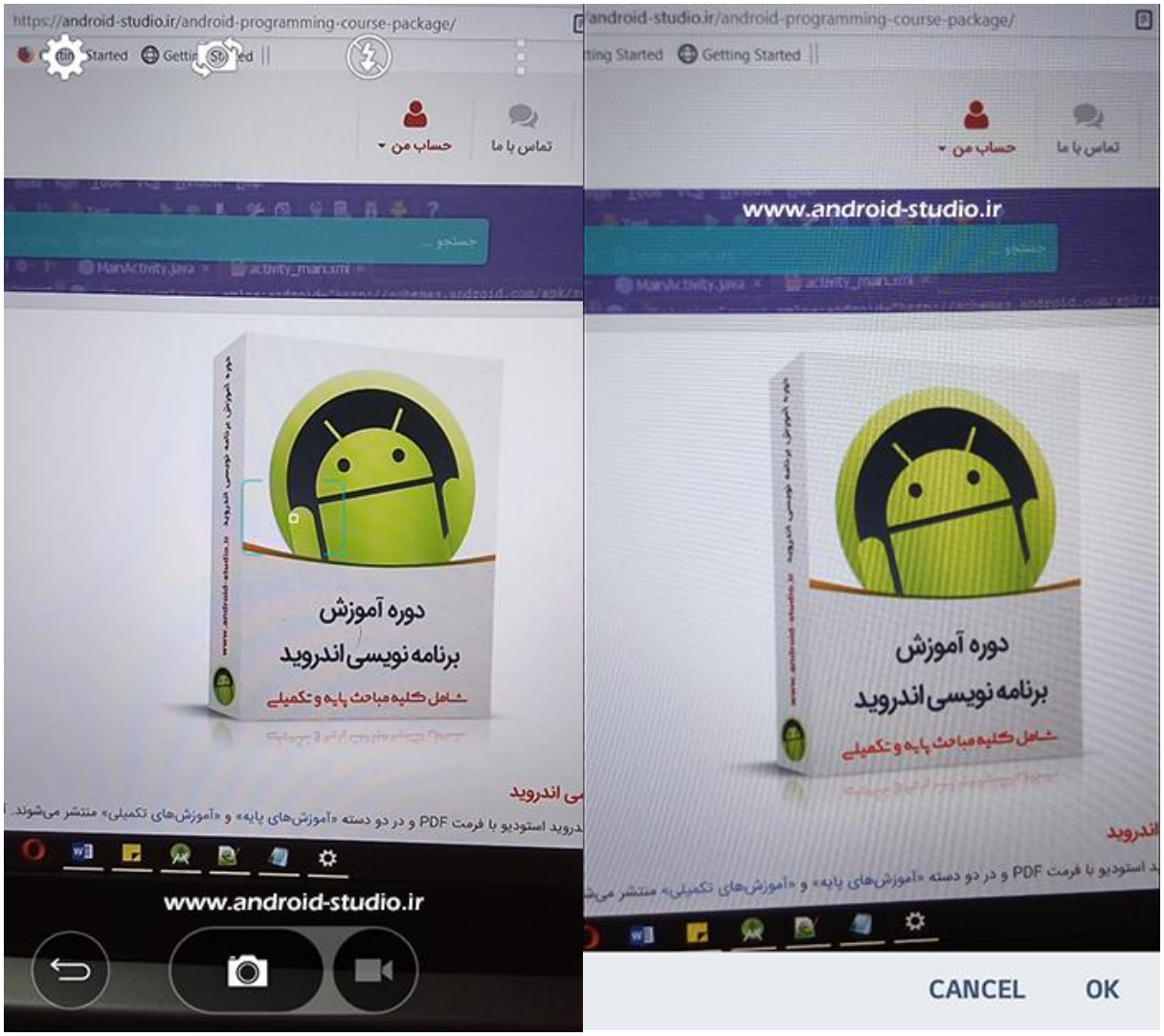
پروژه را اجرا می‌کنم. شبیه سازهای اندروید ازجمله Genymotion و یا AVD اندروید استودیو قابلیت دوربین مجازی را دارند به اینصورت که به طور مجازی می‌توان از یک تصویر متحرک عکس گرفت. در جنی موشن، لوگوی نرم افزار به صورت متحرک در کادر دوربین در چهار جهت مختلف حرکت می‌کند. اما من برای درک بهتر شما، از یک دیوایس حقیقی استفاده می‌کنم:



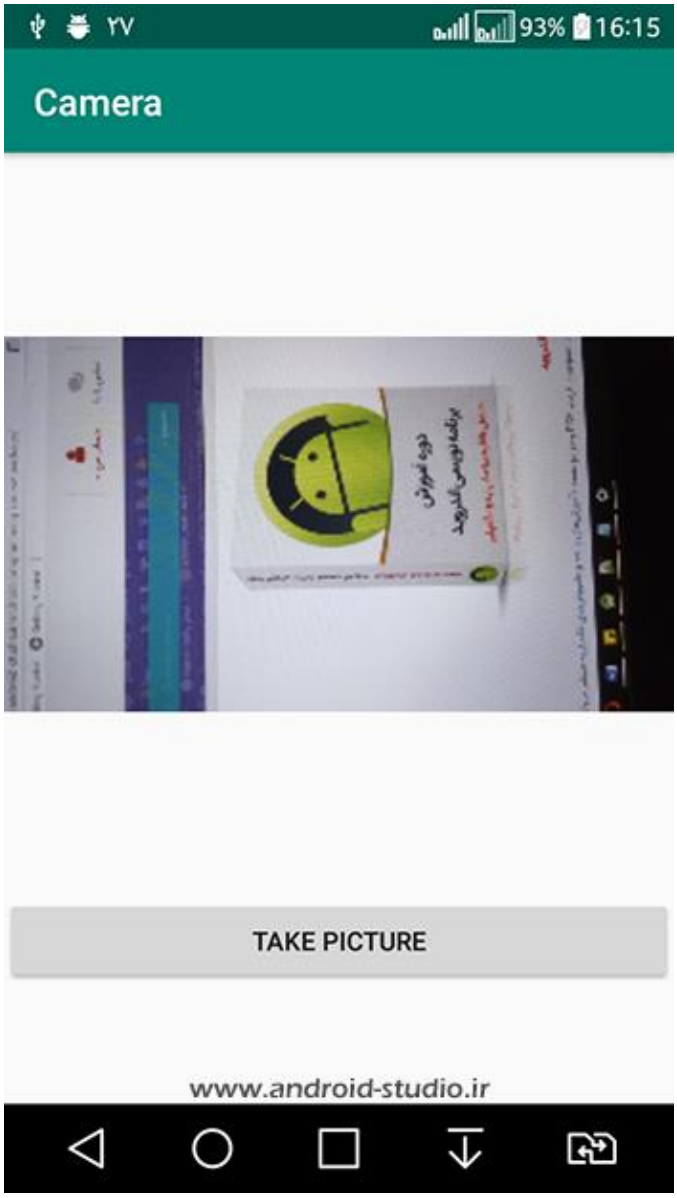
در تصویر فوق، اکتیویتی برنامه را مشاهده می‌کنید که شامل یک ImageView با تصویری پیش فرض (تصویر ic_launcher_background در پوشه drawable پروژه) و یک Button است. دکمه را لمس می‌کنم:



من روی این دیوایس علاوه بر برنامه اصلی مدیریت دوربین، یک برنامه جانبی دیگر نیز استفاده می‌کنم بنابراین سیستم عامل ابتدا از من می‌خواهد برنامه مدنظرم را انتخاب کنم. یک مورد را انتخاب کرده و به صفحه دوربین منتقل می‌شوم:



پس از ثبت تصویر و تایید (OK) آن، به برنامه اصلی برگشته و تصویر نمایش داده می‌شود:


۲۷
93%
16:15
Camera
TAKE PICTURE
www.android-studio.ir

کد کامل MainActivity.java:

```
package ir.android_studio.camera;

import androidx.appcompat.app.AppCompatActivity;
import android.content.Intent;
import android.graphics.Bitmap;
import android.os.Bundle;
import android.provider.MediaStore;
import android.view.View;
import android.widget.Button;
import android.widget.ImageView;

public class MainActivity extends AppCompatActivity {

    ImageView imgView;
    Button btnPhoto;
    protected static final int CameraRequest = 1;

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

        imgView = findViewById(R.id.imageView);
        btnPhoto = findViewById(R.id.btn_photo);

        btnPhoto.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {

                Intent camIntent = new Intent(MediaStore.ACTION_IMAGE_CAPTURE);
                startActivityForResult(camIntent, CameraRequest);

            }
        });

    }

    protected void onActivityResult(int requestCode, int resultCode, Intent data) {

        if (requestCode == CameraRequest) {

            Bitmap resultPhoto = (Bitmap) data.getExtras().get("data");
            imgView.setImageBitmap(resultPhoto);

        }

    }


}
```

در فایل‌های آموزش زبان جاوا (فایل شماره ۵۵) دستور try catch توضیح داده شده. در ادامه قصد دارم

از این دستور برای مدیریت یا به اصطلاح handle کردن خطا استفاده کنم. خطاهایی که هنگام بروز یک یا چند استثناء (Exception) رخ می‌دهند. البته قطعا در قسمت‌های بعد بطور اختصاصی آموزشی برای نحوه مدیریت خطاها در اندروید تهیه خواهم کرد.

ممکن است دیوایسی که برنامه ما روی آن اجرا شده، برنامه دوربینی روی آن نصب و فعال نباشد. در این صورت موقع کلیک روی دکمه Take photo برنامه به دلیل عدم یافتن اکتیویتی دوربین، دچار مشکل شده و اصطلاحا کرش (Crash) خواهد کرد. برای جلوگیری از این مسئله، کد مربوط به اجرای اکتیویتی را درون دستور Try catch قرار می‌دهم:

```
btnPhoto.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View view) {

        Intent camIntent = new Intent(MediaStore.ACTION_IMAGE_CAPTURE);

        try {

            startActivityForResult(camIntent, CameraRequest);

        } catch (ActivityNotFoundException e) {

            Toast.makeText(MainActivity.this, "برنامه دوربین پیدا نشد",
Toast.LENGTH_SHORT).show();

        }

    }
});
```

با اضافه شدن try catch، متد startActivityForResult یعنی کد موجود در بدنه try اجرا می‌شود. در صورتی که به نتیجه‌ی مدنظر (پیدا کردن اکتیویتی دوربین) رسید، مانند قبل، صفحه‌ی دوربین باز خواهد شد اما در صورت عدم حصول نتیجه، بلاک مربوط به catch اجرا می‌شود که من یک پیغام از جنس Toast تعریف کرده‌ام. از نام ActivityNotFoundException پیداست که برای مدیریت خطای مربوط به استثناء "عدم یافتن اکتیویتی" استفاده می‌گردد.

کد نهایی MainActivity.java:

```
package ir.android_studio.camera;

import androidx.appcompat.app.AppCompatActivity;

import android.content.ActivityNotFoundException;
import android.content.Intent;
import android.graphics.Bitmap;
import android.os.Bundle;
import android.provider.MediaStore;
import android.view.View;
```

```
import android.widget.Button;
import android.widget.ImageView;
import android.widget.Toast;

public class MainActivity extends AppCompatActivity {

    ImageView imgView;
    Button btnPhoto;
    protected static final int CameraRequest = 1;

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

        imgView = findViewById(R.id.imageView);
        btnPhoto = findViewById(R.id.btn_photo);

        btnPhoto.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {

                Intent camIntent = new Intent(MediaStore.ACTION_IMAGE_CAPTURE);

                try {

                    startActivityForResult(camIntent, CameraRequest);

                } catch (ActivityNotFoundException e) {

                    Toast.makeText(MainActivity.this, "نشد پیدا دوربین برنامه",
Toast.LENGTH_SHORT).show();

                }

            }
        });


    }

    protected void onActivityResult(int requestCode, int resultCode, Intent data) {

        if (requestCode == CameraRequest) {

            Bitmap resultPhoto = (Bitmap) data.getExtras().get("data");
            imgView.setImageBitmap(resultPhoto);

        }

    }


}
```

مطالعه بیشتر:

https://developer.android.com/reference/android/provider/MediaStore

https://developer.android.com/guide/topics/manifest/uses-feature-element

توجه: سورس پروژه درون پوشه Exercises قرار دارد