آموزش برنامه نویسی اندروید

در محیط اندروید استودیو

متریال دیزاین

بخش پنجم : Floating Action Button یا دکمه شناور

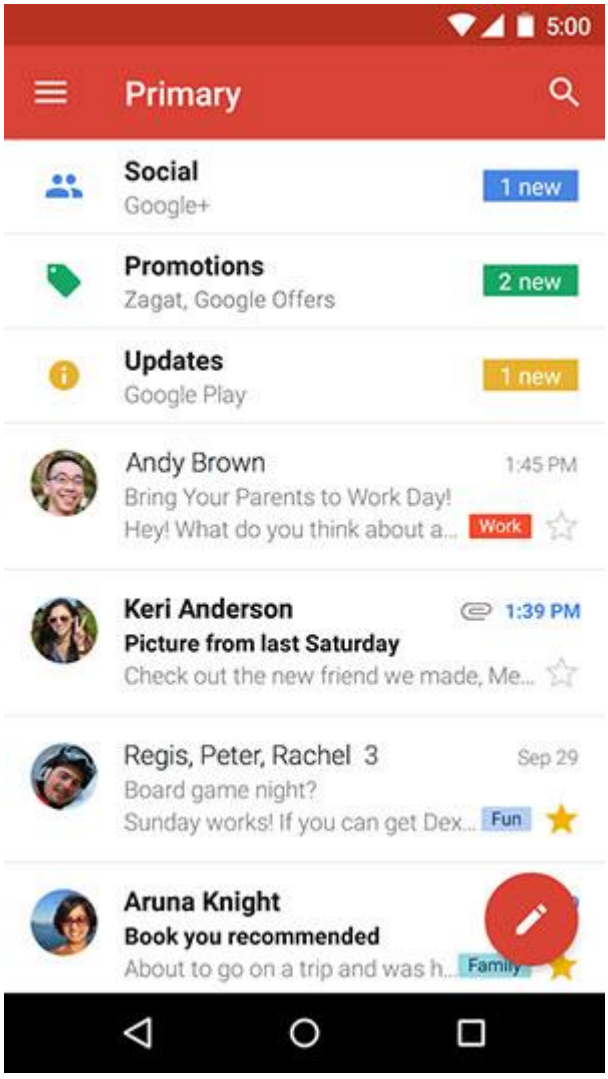
مدرس : سید مهدی مطهری

www.android-studio.ir

به نام خدا

معرفی Floating Action Button:

Floating Action Button یا دکمه شناور که به اختصار FAB نیز نامیده می شود یکی دیگر از کامپوننت های معرفی شده در متریال دیزاین اندروید است. همانطور که از نام آن پیداست، یک دکمه شناور است که مکان قرارگیری آن در صفحه توسط توسعه دهنده تعیین شده و با اسکرول کردن صفحه جابجا نمی شود (مگر به خواست خود توسعه دهنده). این نوع دکمه عموما در اپ هایی مانند پیام رسانها و مدیریت ایمیل استفاده می شود. مانند اپلیکیشن Gmail:



دکمه ای دایره شکل که در سمت راست و پایین صفحه قرار گرفته و یک آیکون مداد را نشان می دهد که کاربر با لمس این دکمه به صفحه ارسال ایمیل هدایت می شود. FAB نیز به کتابخانه appcompat و design نیاز دارد که در ادامه و در قالب یک پروژه به بررسی آن می پردازم.

یک پروژه با نام FAB و MinSDK 17 ایجاد کردم. مانند مباحث قبل، از قسمت Project Structure کتابخانه design را به پروژه اضافه کرده و یا build.gradle (app) را باز کرده و به صورت دستی اینکار را انجام می دهم:

```
dependencies {
    implementation fileTree(dir: 'libs', include: ['*.jar'])
    implementation 'com.android.support:appcompat-v7:27.0.2'
    implementation 'com.android.support:design:27.0.2'
}
```

(کتابخانه های اضافی را حذف کردم)

در این پروژه برای layout اصلی که FAB درون آن قرار می گیرد بجای RelativeLayout یا سایر لایه هایی که قبلا استفاده کردیم، از CoordinatorLayout استفاده می کنیم. توسط این layout مدیریت بهتری روی قسمت های مختلف (از جمله Toolbar و FAB و...) می توان انجام داد. برای این مبحث در همین حد آشنایی با CoordinatorLayout کافی است. ان شا الله پس از تکمیل مباحث متریال دیزاین، یک پروژه کامل خواهیم ساخت و تمامی مباحث عنوان شده از ابتدای آموزشها را در آن بکار خواهم برد.

```
<?xml version="1.0" encoding="utf-8"?>
<android.support.design.widget.CoordinatorLayout
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="ir.android_studio.fab.MainActivity">


</android.support.design.widget.CoordinatorLayout>
```

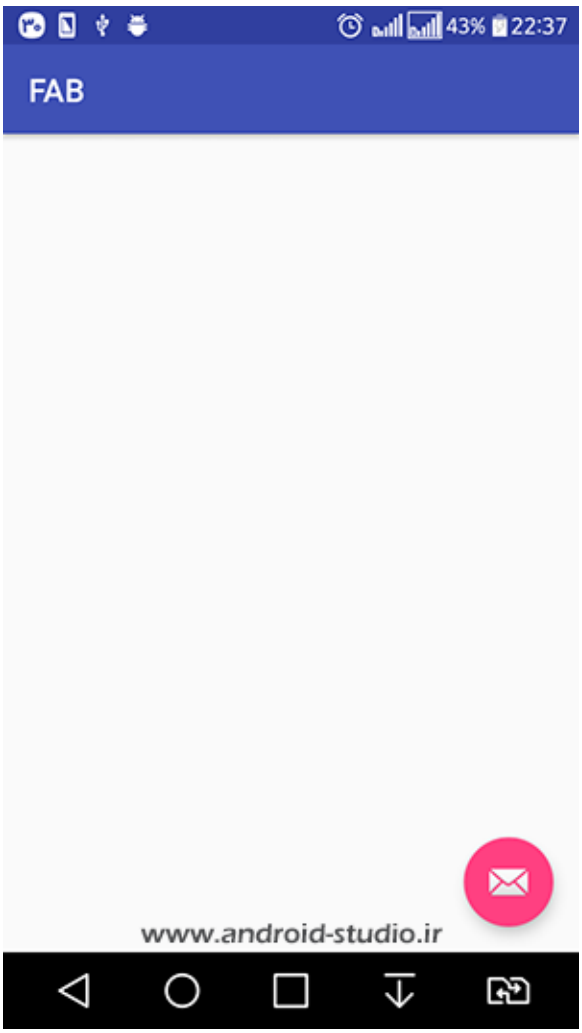
به اینصورت، لایه activity_main.xml را به CoordinatorLayout تبدیل کردم. حالا کافیست یک تگ FloatingActionButton به صورت android.support.design.widget.FloatingActionButton به لایه اضافه کنم:

```
<?xml version="1.0" encoding="utf-8"?>
<android.support.design.widget.CoordinatorLayout
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="ir.android_studio.fab.MainActivity">

    <android.support.design.widget.FloatingActionButton
        android:id="@+id/fab"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_gravity="bottom|end"
        android:layout_margin="16dp"
        android:src="@android:drawable/ic_dialog_email" />

</android.support.design.widget.CoordinatorLayout>
```

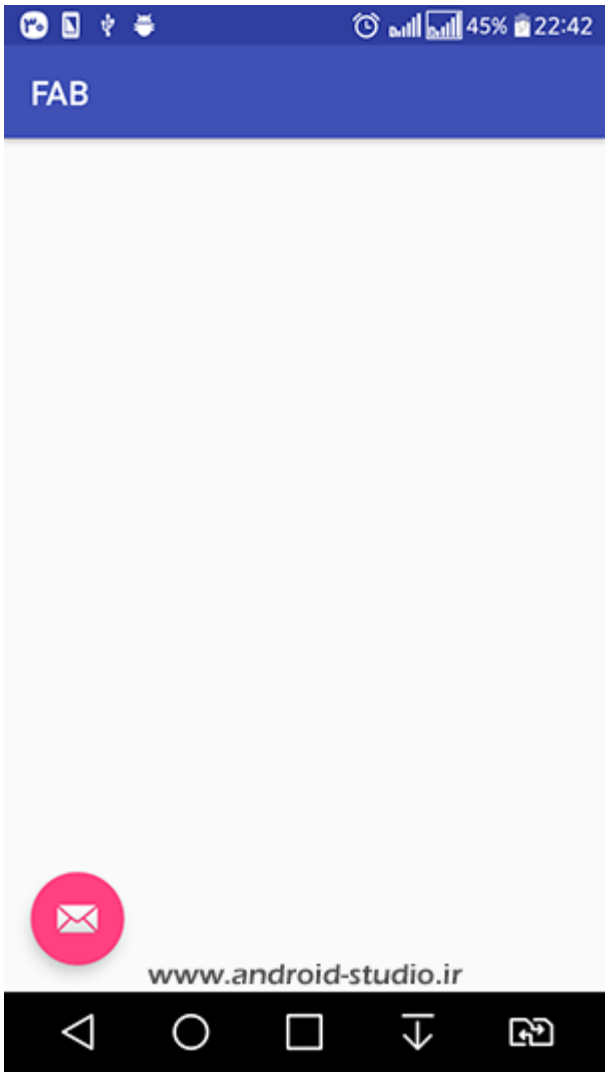
برای layout_gravity دو مقدار bottom (پایین) و end (سمت راست) تعریف شده. layout_margin هم مقدار فاصله ای است که دکمه از کنار صفحه باید داشته باشد که مقدار استاندارد 16dp است. پروژه را اجرا می کنم:



همانطور که انتظار داشتم، Floating Action Button در سمت راست و پایین صفحه با آیکونی که تعریف کرده بودم ظاهر شده است. مسلما اگر layout_gravity را به صورت زیر تغییر دهم، دکمه شناور باید به سمت چپ منتقل شود:

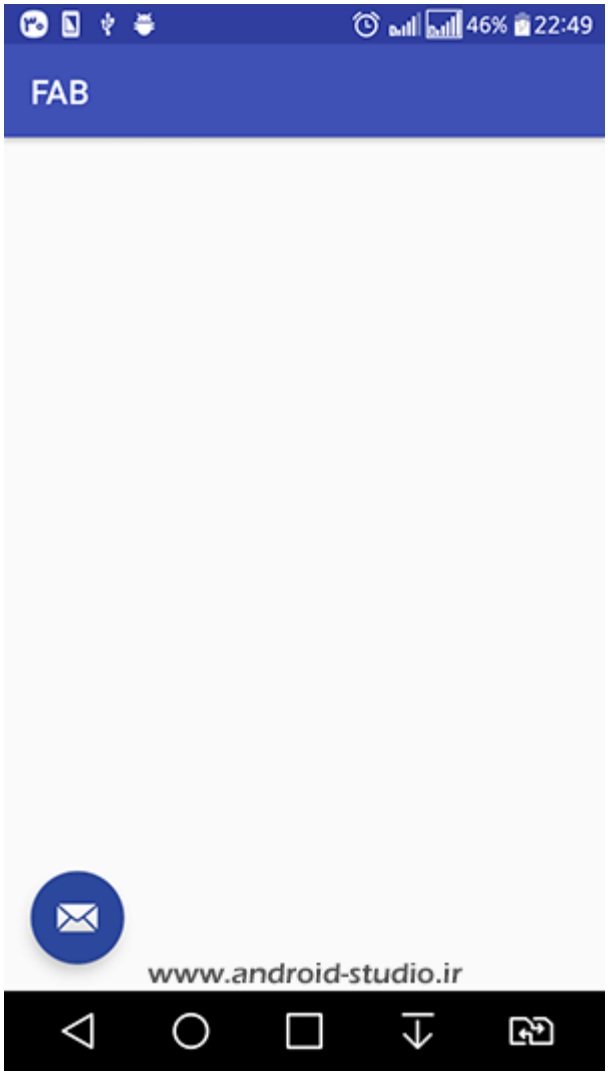
```
android:layout_gravity="bottom|start"
```

start بر خلاف end به معنی ابتدای صفحه نمایش است. یعنی سمت چپ. دوباره اجرا می کنم:



رنگ بک گراند FAB به صورت پیش فرض از colorAccent تم متریال گرفته می شود که با اضافه کردن خاصیت app:backgroundTint امکان تغییر آن وجود دارد:

```
<android.support.design.widget.FloatingActionButton
    android:id="@+id/fab"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_gravity="bottom|start"
    android:layout_margin="16dp"
    android:src="@android:drawable/ic_dialog_email"
    app:backgroundTint="#2c489c" />
```

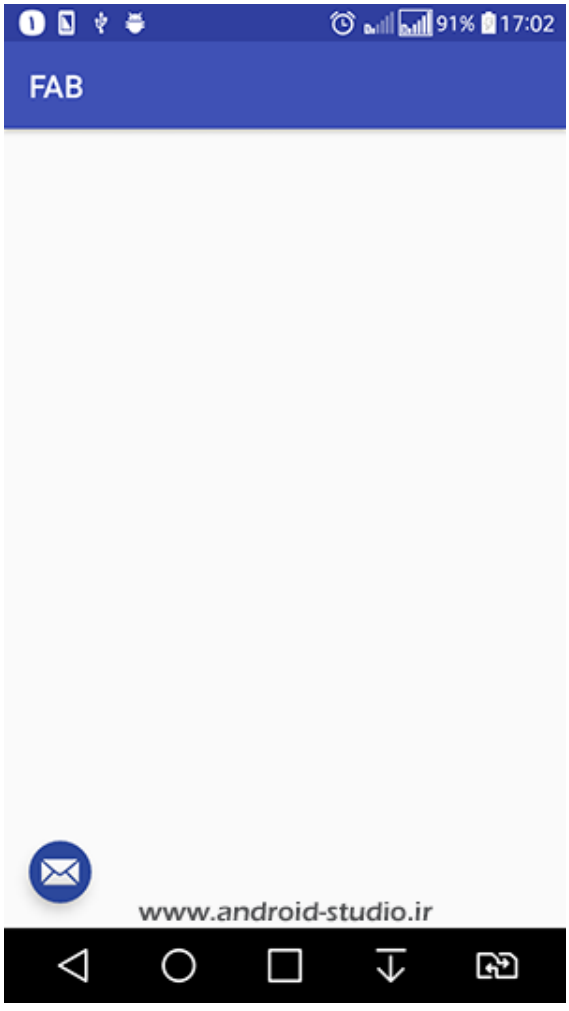


آیتم های دیگری نیز جهت شخصی سازی FAB در اختیار داریم. android:clickable با مقدار true به دکمه یک انیمیشن (Ripple Animation) اضافه می کند که با لمس دکمه ایجاد می شود و کاربر حس بهتری از لمس دکمه دارد. بعد از استفاده از این خاصیت، اندروید استودیو پیغامی نشان می دهد که بهتر است در صورت استفاده از clickable، از android:focusable نیز استفاده شود:

```
<android.support.design.widget.FloatingActionButton
    android:id="@+id/fab"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_gravity="bottom|start"
    android:layout_margin="16dp"
    android:src="@android:drawable/ic_dialog_email"
    app:backgroundTint="#2c489c"
    android:clickable="true"
    android:focusable="true"/>
```

سایز دکمه نیز توسط خاصیت app:fabSize قابل تغییر است که دو مقدار normal و mini را می پذیرد. اگر این خاصیت روی FAB تعریف نشود به صورت پیش فرض مقدار normal اعمال خواهد شد که عموما هم ما همین اندازه را نیاز داریم. بنابراین با دادن مقدار normal به fabSize، سایز دکمه تغییری نکرده اما مقدار mini باعث کوچکتر شدن دکمه می شود:

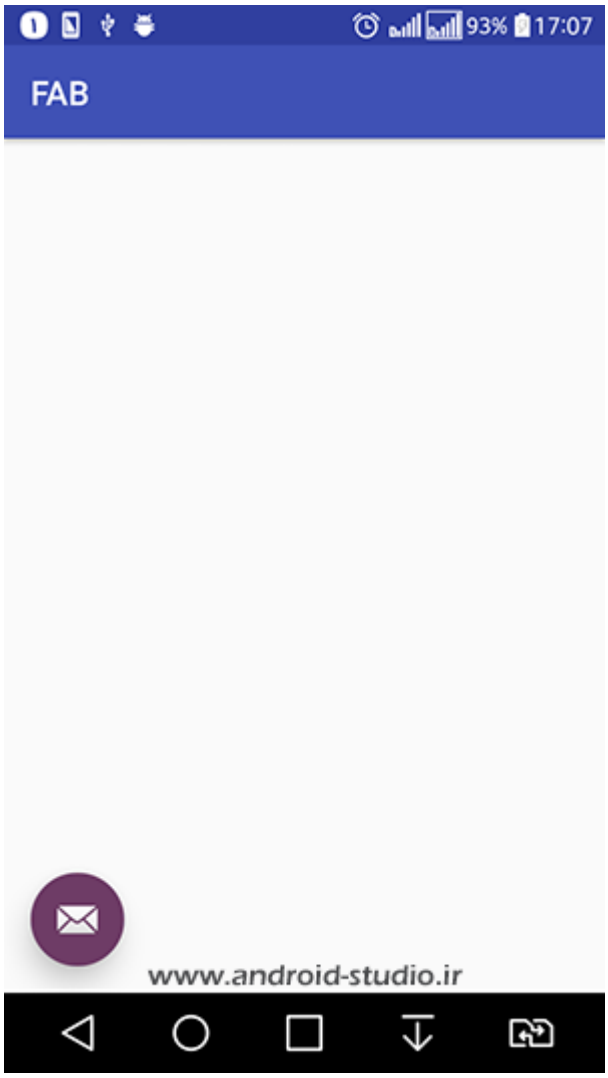
```
app:fabSize="mini"
```



و آخرین موردی که در این مبحث به آن اشاره می کنم، خاصیت app:rippleColor است. در حالت عادی و بدون استفاده از این خاصیت، در صورت استفاده از خاصیت clickable، هنگام لمس دکمه، همان رنگ پس زمینه دکمه قدری تیره تر می شود اما با استفاده از rippleColor و تعریف یک رنگ جدید، ripple Animation لمس دکمه با این رنگ اجرا خواهد شد:

```
<android.support.design.widget.FloatingActionButton
    android:id="@+id/fab"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_gravity="bottom|start"
    android:layout_margin="16dp"
    android:src="@android:drawable/ic_dialog_email"
    app:backgroundTint="#2c489c"
    android:clickable="true"
    android:focusable="true"
    app:fabSize="normal"
    app:rippleColor="#b12f2f"/>
```

(جهت مشاهده واضح تر نتیجه روی اسکرین شات، سایز دکمه را مجدد به نرمال تغییر دادم.)



مرحله نهایی تعریف یک رویداد برای دکمه شناور است که مانند سایر دکمه هاست:

```
package ir.android_studio.fab;

import android.support.design.widget.FloatingActionButton;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.view.View;
import android.widget.Toast;

public class MainActivity extends AppCompatActivity {

    FloatingActionButton mFab;

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

        mFab = findViewById(R.id.fab);

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

                Toast.makeText( MainActivity.this, "FAB Clicked", Toast.LENGTH_SHORT).show();

            }
        });

    }
}
```

به خط زیر دقت کنید:

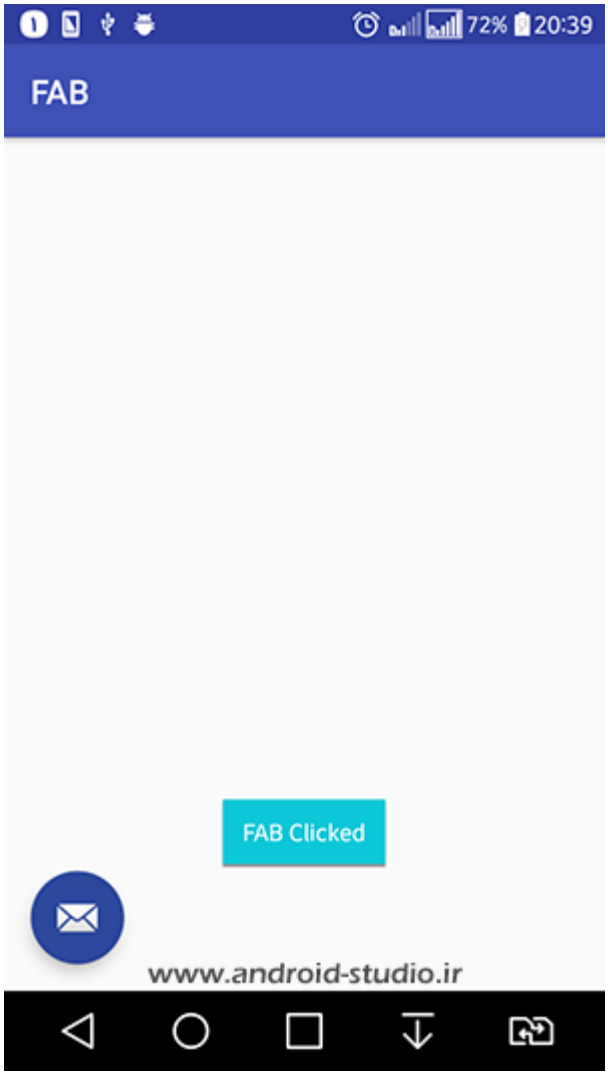
```
mFab = findViewById(R.id.fab);
```

قبلا لازم بود عمل Casting را هنگام تعریف یک view در جاوا انجام دهیم. مانند زیر:

```
mFab = (FloatingActionButton) findViewById(R.id.fab);
```

اما در نسخه جدید اندروید استودیو این عمل ضروری نیست و در صورت استفاده، پیغامی مبنی بر زائد و اضافی بودن Casting نشان می دهد.

حالا پروژه را اجرا و روی FloatingActionButton کلیک می کنم:



پیغام Toast به درستی اجرا شد.

توجه: سورس پروژه درون پوشه Exercises قرار داده شده است

با ارائه انتقادات و پیشنهادات خود، ما را در ارائه آموزش های بهتر یاری فرمائید.
www.android-studio.ir