آموزش برنامه نویسی اندروید
در محیط اندروید استودیو

ساخت صفحه Splash Screen

مدرس : سید مهدی مطهری

www.android-studio.ir

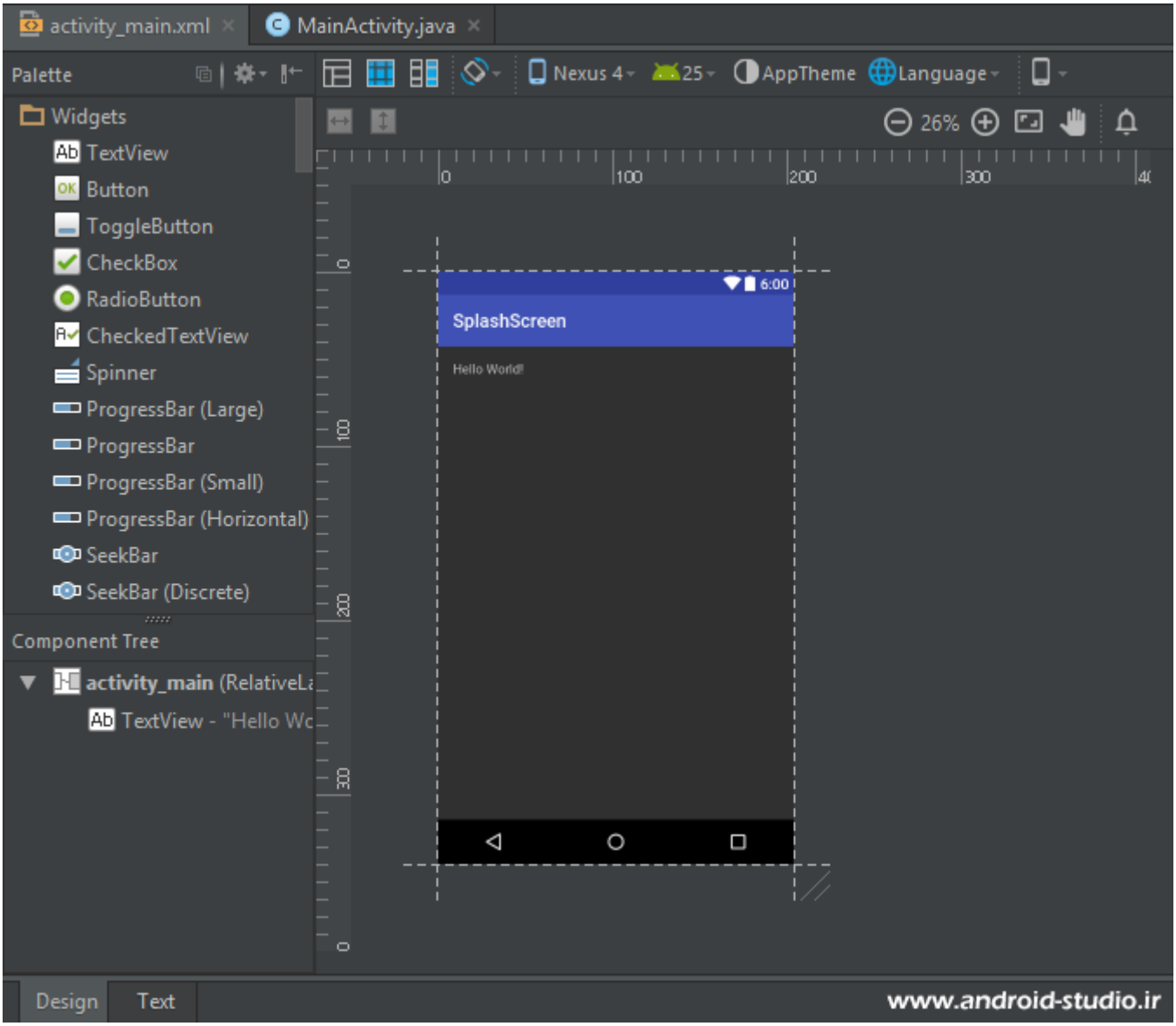
به نام خدا

حتما با اپلیکیشن هایی برخورد داشته اید که در هنگام اجرا برای چند ثانیه صفحه ای که عموما شامل لوگوی شرکت سازنده هست، نمایش داده شده و پس از اتمام زمان مشخص به صفحه اصلی اپلیکیشن منتقل می شود. این صفحه Splash Screen (اسپلش اسکرین) نام دارد. این صفحه یک اکتیویتی است که باید تعیین کنیم پس از یک زمان مشخص بعد از اجرای اپلیکیشن توسط کاربر (مثلا ۳ ثانیه) محو شده و اکتیویتی نخست جایگزین شود. مثال Splash Screen را می توان در اپلیکیشن های متعددی از جمله نمونه وطنی آن یعنی دیجیکالا دید:

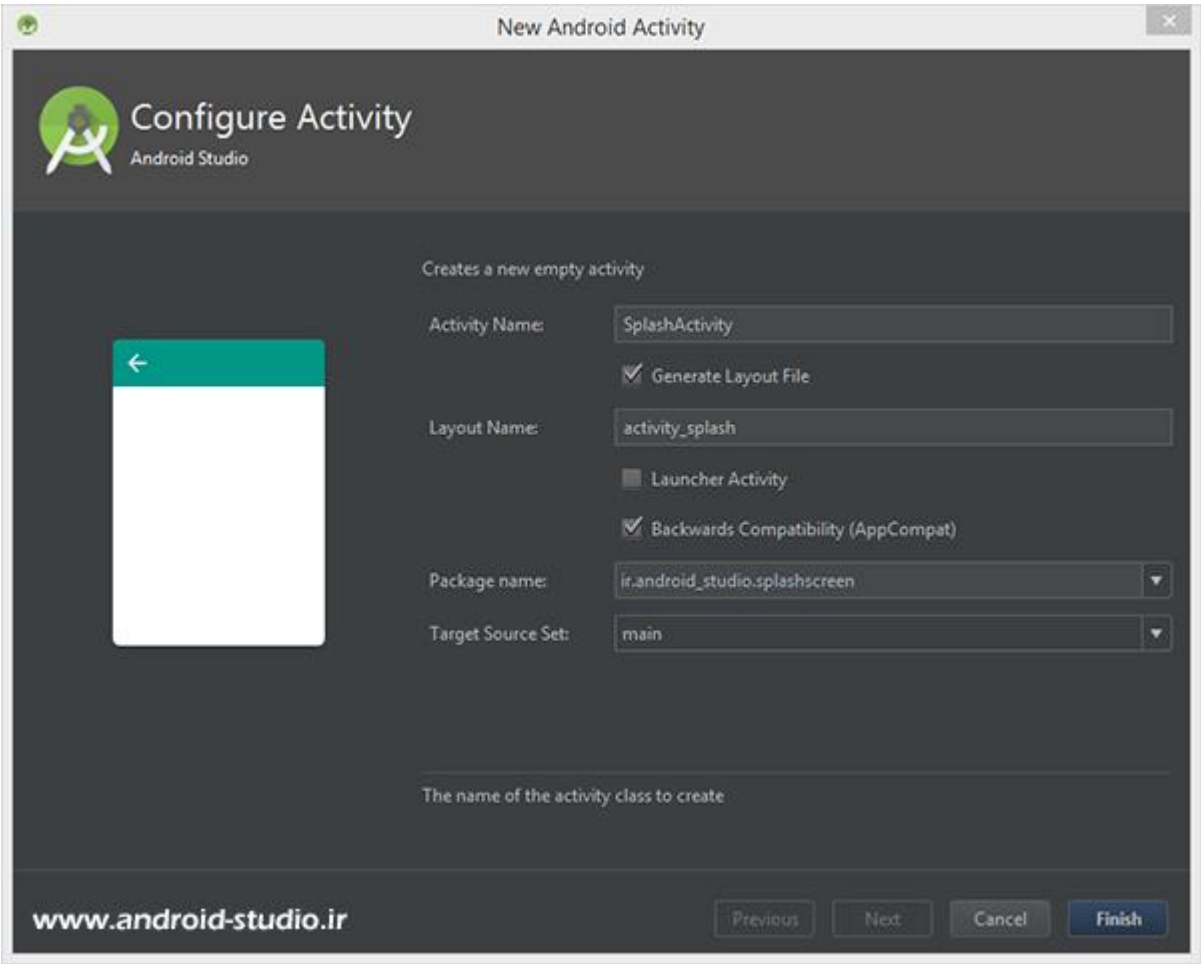
توسعه دهندگان اپلیکیشن دیجیکالا برای صفحه Splash Screen زمانی حدود ۱ ثانیه را در نظر گرفته اند. همانطور که مشاهده می کنید این صفحه بسیار ساده بوده و فقط شامل لوگو در وسط و یک

ProgressBar در انتهای صفحه می شود. با این حال اینکه اجزای این صفحه چه باشد کاملا در اختیار برنامه نویس بوده و مانند یک اکتیویتی معمولی می توان هر چیزی را درون آن قرار داد. اسپلش اسکرین را به چند روش می توان ساخت که ما در اینجا روش بهینه تر را در قالب یک مثال و به صورت عملی بررسی می کنیم.

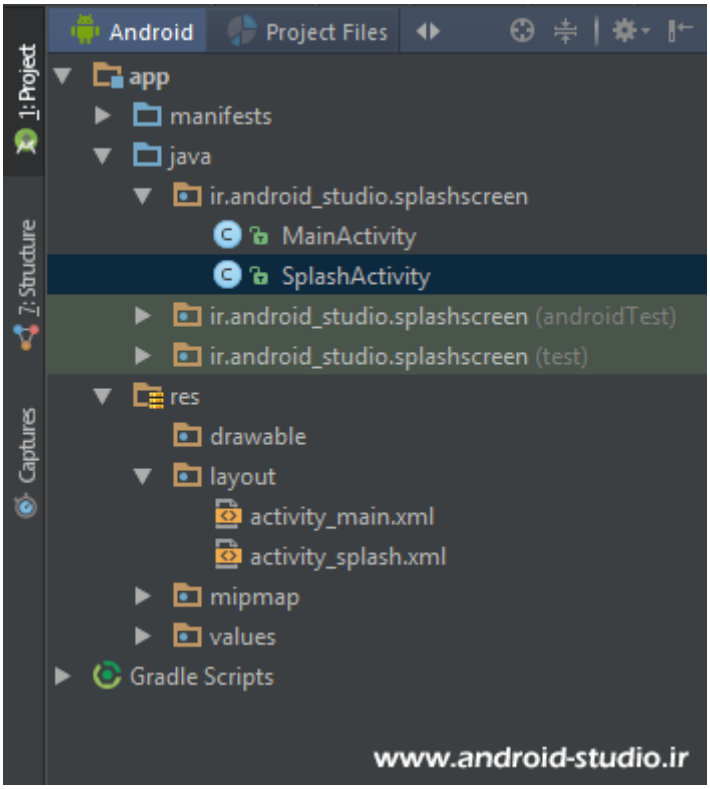
یک پروژه جدید با نام SplashScreen با API10 و اکتیویتی از نوع Empty Activity ساختیم:



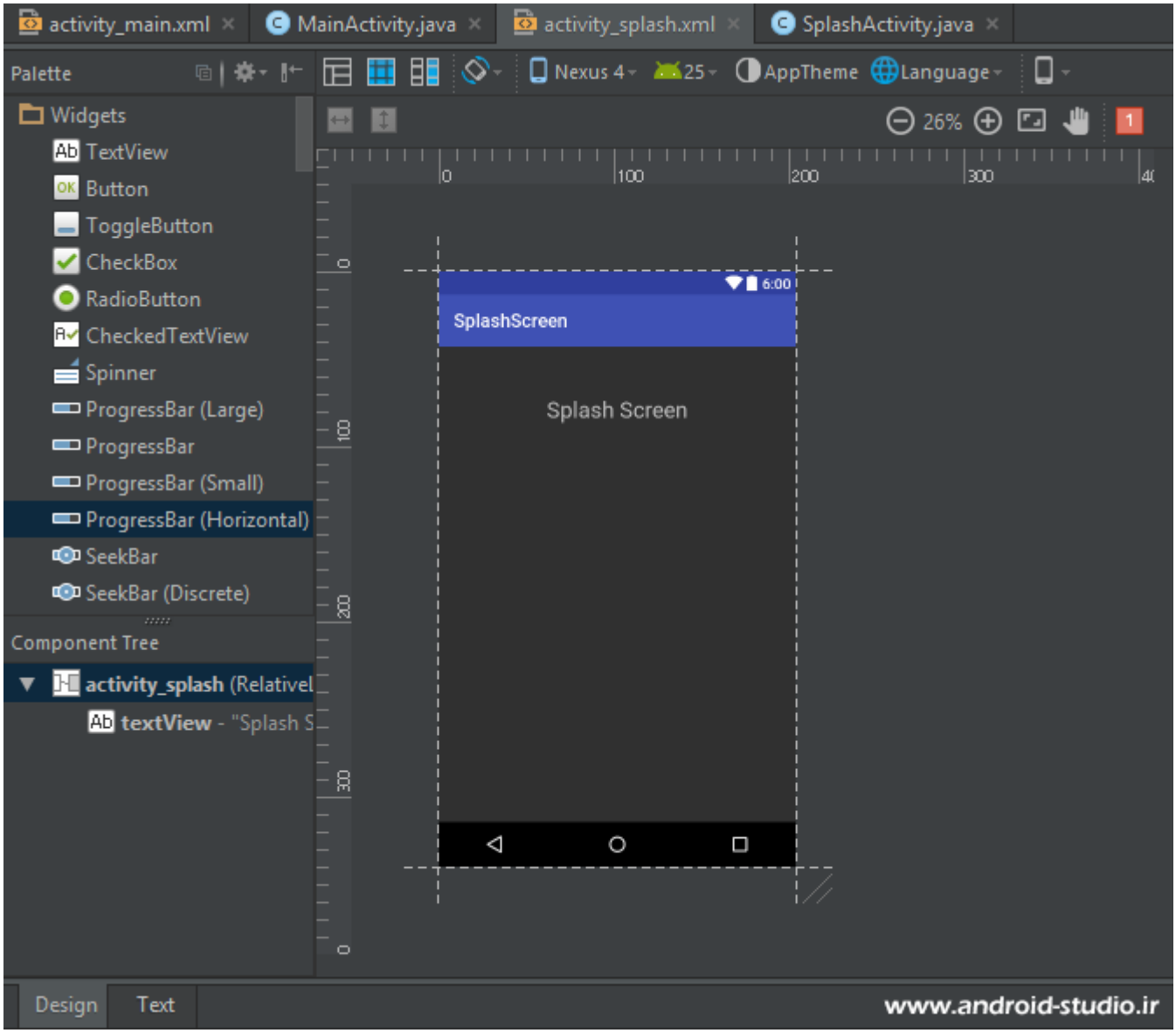
در حال حاضر پروژه ما یک اکتیویتی با نام MainActivity دارد. حال می‌بایست یک اکتیویتی جدید نیز ایجاد کنیم و یکی از دو اکتیویتی را برای Splash Screen درنظر بگیریم. به جهت اینکه روال مباحث قبل را حفظ کنیم، اکتیویتی جدید را به اسپلش اسکرین اختصاص داده و MainActivity همچنان نقش صفحه اصلی اپلیکیشن را بر عهده خواهد داشت. یک اکتیویتی جدید از نوع Empty Activity با نام دلخواه SplashActivity می سازیم:



کلاس جاوا و فایل xml اکتیویتی جدید به پروژه اضافه شد:



ابتدا به سراغ کلاس جاوای اکتیویتی می رویم تا کد موردنیاز برای اجرای اسپلش اسکرین را پیاده سازی کنیم. فقط برای تشخیص تفاوت دو اکتیویتی، یک ویجت از نوع TextView با عبارت Splash Screen درون اکتیویتی Splash اضافه کردیم:



همانطور که در ابتدای مبحث اشاره شد، کاربرد صفحه اسپلش اسکرین این است که بتوانیم قبل از باز شدن اکتیویتی اصلی اپلیکیشن، به مدت چند ثانیه محتوایی را به کاربر نمایش دهیم. ما این کار را توسط کلاس Handler انجام می دهیم. کلاس Handler را new کرده و همانطور که مشاهده می کنید کلاس باید ایمپورت شود:

```
activity_main.xml × | MainActivity.java × | activity_splash.xml × | SplashActivity.java ×
1  package ir.android_studio.splashscreen;
2
3  import android.support.v7.app.AppCompatActivity;
4  import android.os.Bundle;
5
6  public class SplashActivity extends AppCompatActivity {
7
8      @Override
9      protected void onCreate(Bundle savedInstanceState) {
10         super.onCreate(savedInstanceState);
11         setContentView(R.layout.activity_splash);
12
13         new Handler()
14
15     }
16 }
17
www.android-studio.ir
Class to Import
Handler (android.os)          < Android API 25 Platform > (android.jar)
Handler (java.util.logging)   < Android API 25 Platform > (android.jar)
```

سپس از متد postDelayed برای تعیین زمان توقف استفاده می کنیم (Delay به معنی تاخیر است):

```
activity_main.xml × | MainActivity.java × | activity_splash.xml × | SplashActivity.java ×
1  package ir.android_studio.splashscreen;
2
3  import android.os.Handler;
4  import android.support.v7.app.AppCompatActivity;
5  import android.os.Bundle;
6
7  public class SplashActivity extends AppCompatActivity {
8
9      @Override
10     protected void onCreate(Bundle savedInstanceState) {
11         super.onCreate(savedInstanceState);
12         setContentView(R.layout.activity_splash);
13
14         new Handler().postDelayed()
15
16     }
17 }
18
Runnable r, long delayMillis
www.android-studio.ir
```

طبق راهنمای اندروید استودیو پارامتر اولی Runnable و دومی زمان مدنظر بوده که باید برحسب میلی ثانیه وارد شود. با new کردن Runnable و انتخاب گزینه نخست، اندروید استودیو به صورت خودکار متد run را ایجاد می کند:

```
activity_main.xml × | MainActivity.java × | activity_splash.xml × | SplashActivity.java ×
1   package ir.android_studio.splashscreen;
2
3   import android.os.Handler;
4   import android.support.v7.app.AppCompatActivity;
5   import android.os.Bundle;
6
7   public class SplashActivity extends AppCompatActivity {
8
9       @Override
10      protected void onCreate(Bundle savedInstanceState) {
11          super.onCreate(savedInstanceState);
12          setContentView(R.layout.activity_splash);
13
14          new Handler().postDelayed(new Ru
15                                        Runnable{...} (java.lang)
16      }                                 RunnableFuture<V>{...} (java.util.concurrent)
17  }                                     RunnableScheduledFuture<V>{...} (java.util.concurrent)
18                                        RuntimeException (java.lang)
                                          Runtime (java.lang)
                                          RuntimePermission (java.lang)
                                          InterruptedException (java.lang)
                                          Press Ctrl+Shift+Space to show only variants that are suitable by type
www.android-studio.ir
```

کلاس SplashScreen.java تا به اینجای کار:

```
package ir.android_studio.splashscreen;

import android.os.Handler;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;

public class SplashActivity extends AppCompatActivity {

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

        new Handler().postDelayed(new Runnable() {
            @Override
            public void run() {

            }
        })

    }
}
```

در این مرحله خطا می‌گیرید که علت آن عدم تعیین زمان است. بعد از Runnable یک ویرگول گذاشته و تایم دلخواه را اضافه می کنیم (ما ۳۰۰۰ را وارد کردیم یعنی ۳ ثانیه):

```
activity_main.xml × | MainActivity.java × | activity_splash.xml × | SplashActivity.java ×
1  package ir.android_studio.splashscreen;
2
3  import android.os.Handler;
4  import android.support.v7.app.AppCompatActivity;
5  import android.os.Bundle;
6
7  public class SplashActivity extends AppCompatActivity {
8
9      @Override
10     protected void onCreate(Bundle savedInstanceState) {
11         super.onCreate(savedInstanceState);
12         setContentView(R.layout.activity_splash);
13
14         new Handler().postDelayed(new Runnable() {
15             @Override
16             public void run() {
17
18             }
19         }, 3000)
20
21     }
22 }
23                                                www.android-studio.ir
```

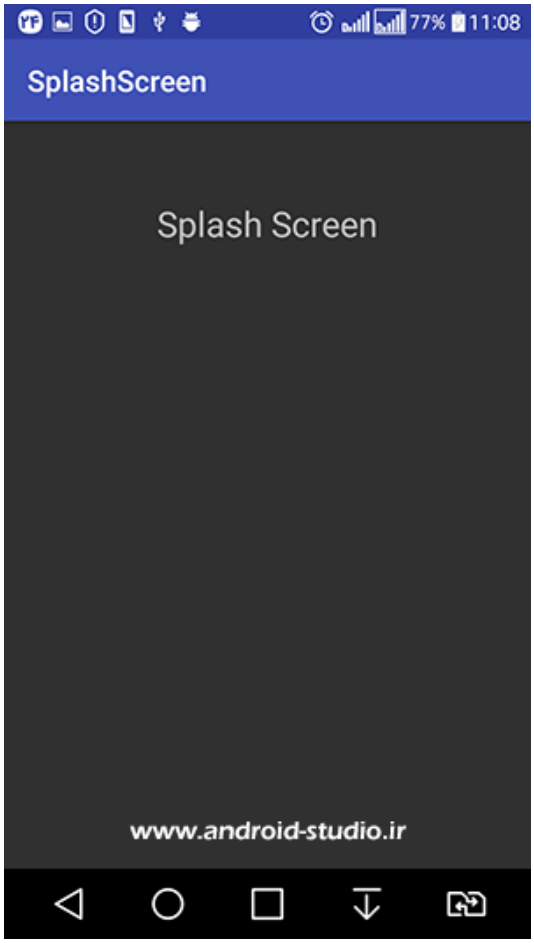
هنوز یک خطای دیگر هم مشاهده می شود که با بردن نشانگر موس روی ناحیه خطا مشخص می شود سمی کالن ";" از قلم افتاده. در قدم بعد باید تعیین کنیم بعد از اتمام زمان تعیین شده کاربر به چه صفحه ای منتقل شود که این کار را توسط intent انجام می دهیم. هدف ما انتقال از اکتیویتی SplashActivity پس از اتمام زمان تعیین شده به اکتیویتی MainActivity است. متد run را به اینصورت تکمیل می کنیم:

```
new Handler().postDelayed(new Runnable() {
    @Override
    public void run() {
        Intent i = new Intent(SplashActivity.this, MainActivity.class);
        startActivity(i);
    }
}, 3000);
```

اگر به خاطر داشته باشید در مبحث intent ها به اینکه چگونه یک اکتیویتی به عنوان اکتیویتی اصلی تعیین می شود، اشاره شد. هنگام ساخت پروژه MainActivity به عنوان اکتیویتی اصلی تعیین شده که لازم است این ویژگی را به اکتیویتی SplashActivity اختصاص دهیم. به مانیفست رفته و intent-filter مربوط به MainActivity را به اکتیویتی اسپلش منتقل می کنیم:

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

    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="@string/app_name"
        android:supportsRtl="true"
        android:theme="@style/AppTheme">
        <activity android:name=".MainActivity">

        </activity>
        <activity android:name=".SplashActivity">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

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

</manifest>
```

حالا پروژه را اجرا می کنیم:



ابتدا Splash Screen به مدت ۳ ثانیه نمایش و سپس MainActivity جایگزین آن شد:

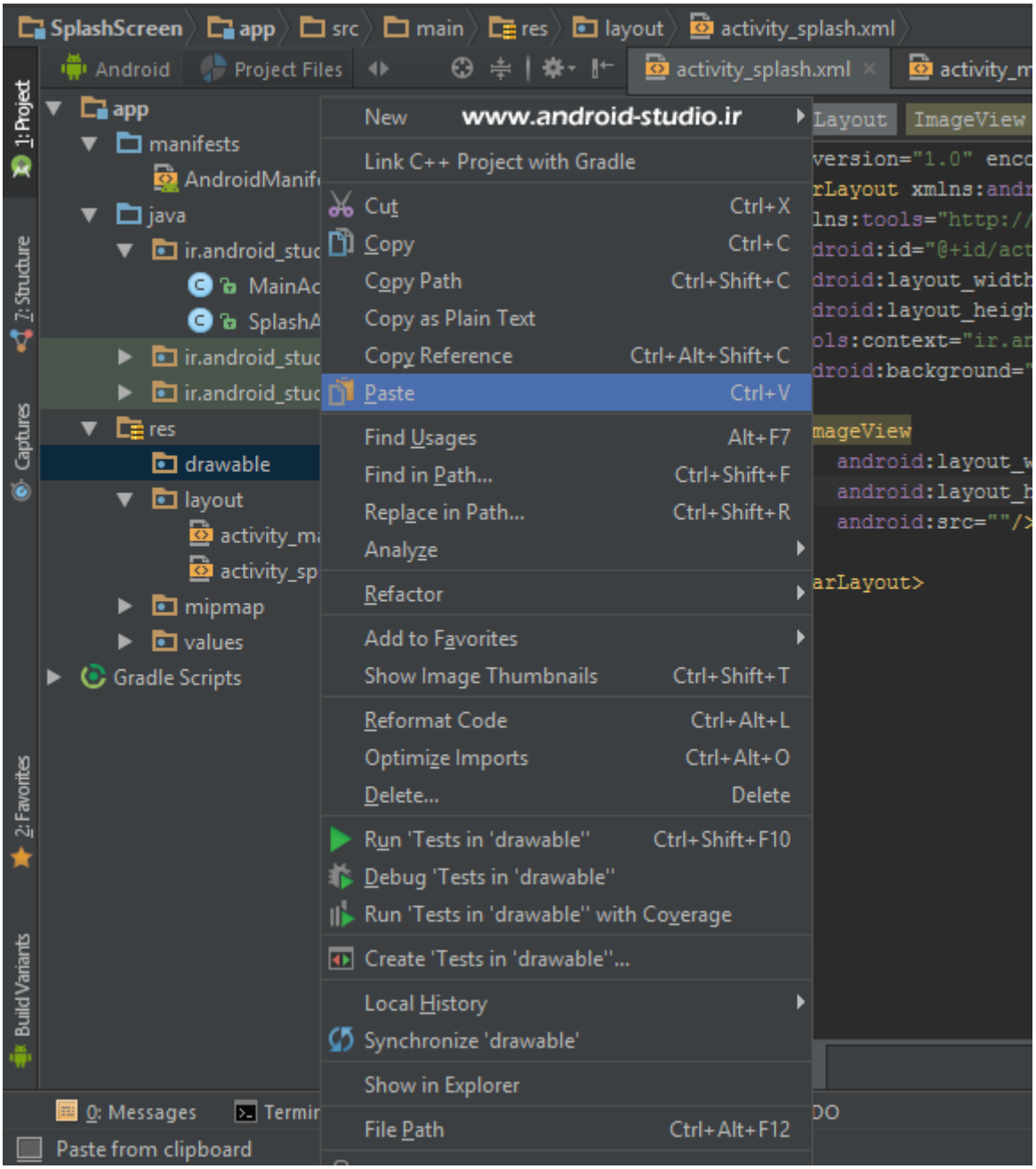
تا به اینجای کار ظاهرا همه چیز خوب پیش رفته. اما کافیست یکبار امتحان کنید و بعد از انتقال به اکتیویتی اصلی، دکمه Back یا برگشت اندروید را بزنید. مشاهده می کنید به جای اینکه از اپلیکیشن خارج شوید، مجددا اسپلش اسکرین نمایش داده می شود! در صورتی که ماهیت اسپلش بر این است که فقط و فقط یک بار در هر اجرا نمایش داده شود. کافی است بعد از دستور مربوط به اینتنت، با دستور finish() این مشکل را برطرف کنیم:

```
new Handler().postDelayed(new Runnable() {
    @Override
    public void run() {
        Intent i = new Intent(SplashActivity.this, MainActivity.class);
        startActivity(i);

        finish();
    }
}, 3000);
```

دستور finish باعث می شود تا اکتیویتی بعد از سپری شدن تایم مشخص شده، بسته شود و امکان برگشت به آن وجود نداشته باشد.

هدف ما در این آموزش اجرای اسپلش اسکرین بود اما طبق ضرب المثل "کار را که کرد آن که تمام کرد"، بد نیست ما هم کار را تمام کرده و یک نمونه دیزاین صفحه SplashScreen را هم تمرین کنیم.

قصد داریم اسپلش ما شامل یک رنگ پس زمینه و لوگو باشد. در مباحث قبل مطرح شد که تصاویری که قصد استفاده از آنها درون اپلیکیشن داریم را باید درون پوشه drawable قرار داد. فایل تصویر لوگو را کپی کرده و روی فولدر drawable راست کلیک و سپس paste می کنیم:



البته خارج از محیط اندروید استودیو نیز می توان تصویر یا هر فایل دیگر را در مسیری که پروژه ساخته شده قرار داد. توسط ویژگی android:src نیز مسیر تصویر را تعیین می کنیم (عکس مدنظر ما android_logo.png نام دارد).

activity_splash.xml :

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/activity_splash"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context="ir.android_studio.splashscreen.SplashActivity"
    android:background="#b3b3b3"
    android:orientation="vertical">

        <ImageView
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_gravity="center"
            android:src="@drawable/android_logo"/>

</LinearLayout>
```

مجدد پروژه را اجرا می کنیم:

بهتر است نوار Toolbar مربوط به اکتیویتی اسپلش اسکرین را نیز حذف کنیم. عموما صفحات اسپلش اسکرین بدون هرگونه تولبار و... هستند. برای حذف تولبار لازم است یک Theme (تم) بدون تولبار (اکشن بار) برای اکتیویتی مدنظر تعریف کنیم. این مباحث مربوط به طراحی تم می شود که در آینده به آن خواهیم پرداخت اما فعلا به جهت رفع نیاز به توضیح اجمالی اکتفا می کنیم. ویژگی theme را درون تگ باز <activity> به اینصورت اضافه کردیم:

```
android:theme="@style/Theme.AppCompat.NoActionBar"
```

@style به این معنی‌ست که قصد استفاده از استایلی را داریم. استایل Theme.AppCompat.NoActionBar همانطور که از نامش پیداست تولبار یا اکشن بار را شامل نمی شود. در نهایت تگ اکتیویتی به اینصورت شد:

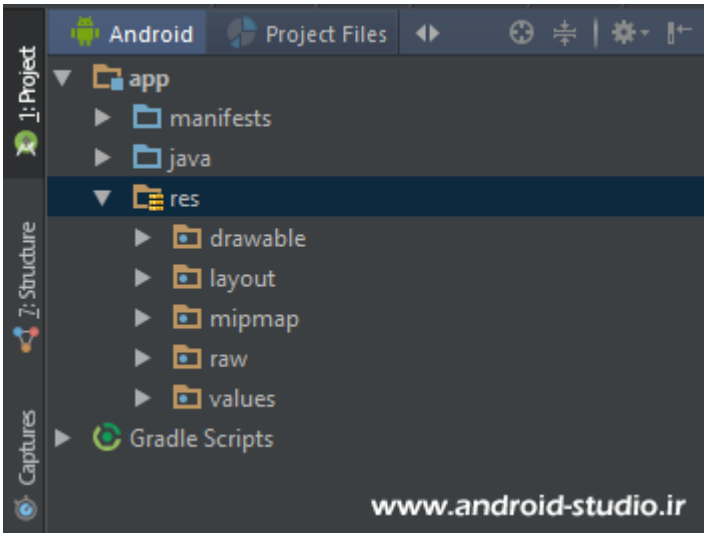
```
<activity android:name=".SplashActivity"
    android:theme="@style/Theme.AppCompat.NoActionBar">
    <intent-filter>
        <action android:name="android.intent.action.MAIN" />

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

مجدد پروژه را اجرا می کنیم:

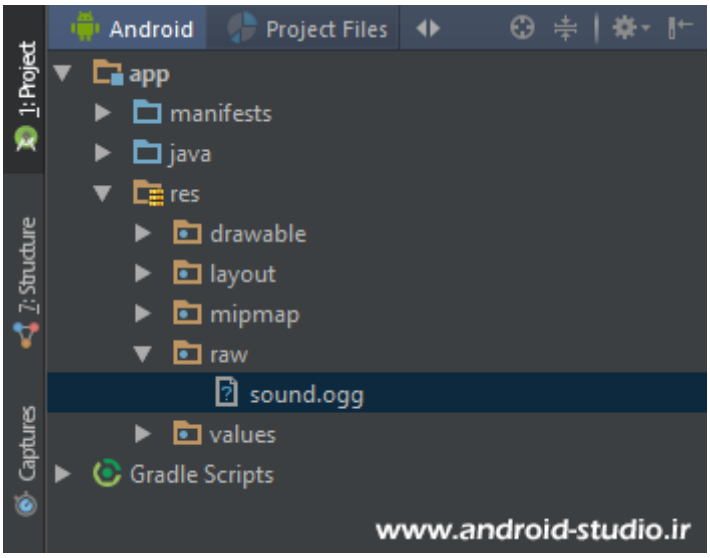
تغییرات به درستی اعمال شد و مانند دفعات قبل بعد از نمایش ۳ ثانیه اسپلش اسکرین، به اکتیویتی اصلی منتقل می شویم.

قدم آخر در این آموزش اضافه کردن فایل صوتی به اسپلش اسکرین است به اینصورت که همزمان با اجرای اپلیکیشن و نمایش صفحه اسپلش، یک فایل صوتی نیز در پس زمینه اجرا شود. برای اضافه کردن

فایلهای صوتی به پروژه اندرویدی لازم است یک فولدر با نام raw درون فولدر res ایجاد شود که با راست کلیک روی پوشه res و سپس New > Directory قابل انجام است.



حالا می بایست فایل صوتی مدنظر را کپی کرده و درون پوشه raw قرار دهیم (اکثر فرمت های رایج صوتی پشتیبانی می شود):



فایل صوتی با نام sound.ogg که انتخاب کردیم یک افکت ۲ ثانیه ای است. اکنون باید فایل صوتی را به اکتیویتی اضافه کنیم. ابتدا یک شئ از کلاس MediaPlayer با نام دلخواه splashSound ساختیم:

```
private MediaPlayer splashSound;
```

سپس با دستور create، کانتکست this و بعد از آن نیز مسیر قرارگیری فایل صوتی که به صورت R.raw قابل دسترسی است. در نهایت start می کنیم:

```
splashSound = MediaPlayer.create(this, R.raw.sound);
splashSound.start();
```

با اجرای مجدد پروژه هنگام نمایش اسپلش اسکرین، افکت صوتی نیز در پس زمینه اجرا شد. دقت داشته باشید مدت زمان فایل صوتی از تایمی که برای اسپلش تعریف کرده اید بیشتر نباشد.

کد نهایی SplashActivity.java :

```
package ir.android_studio.splashscreen;

import android.content.Intent;
import android.media.MediaPlayer;
import android.os.Handler;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;

public class SplashActivity extends AppCompatActivity {

    private MediaPlayer splashSound;

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

        splashSound = MediaPlayer.create(this, R.raw.sound);
        splashSound.start();

        new Handler().postDelayed(new Runnable() {
            @Override
            public void run() {
                Intent i = new Intent(SplashActivity.this, MainActivity.class);
                startActivity(i);

                finish();
            }
        }, 3000);

    }
}
```

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