

SEMINARIO DE LENGUAJES

OPCIÓN ANDROID



Sensores

Esp. Delía Lisandro, Mg. Corbalán Leonardo

Acceso a la cámara de fotos



Cámara de fotos

- Se puede tener acceso a la cámara de fotos invocando alguna aplicación instalada que cumpla tal propósito.
- Se debe indicar en el archivo Manifest que la aplicación a desarrollar utiliza una determinada característica del dispositivo.

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



Cámara de fotos



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

- Si la propiedad `android:required` vale `true` la aplicación podrá ser instalada **solo** en dispositivos que tengan cámara de fotos.
- Por el contrario, si la propiedad `android:required` vale `false` la aplicación podrá ser instalada en **cualquier** dispositivo (disponga o no de cámara de fotos).
 - El desarrollador deberá validar en tiempo de ejecución si el dispositivo cuenta con la característica, y en caso de que no, limitar el funcionamiento de la aplicación
 - `hasSystemFeature(PackageManager.FEATURE_CAMERA)`.

Actividad guiada

- Crear un nuevo proyecto **Android Studio** llamado “CamaraDeFotos”
- En el AndroidManifest.xml incorporar:

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

- En activity_main.xml definir la siguiente vista:

```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical" >
    <Button
        android:id="@+id/button"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="Sacar foto"
        android:onClick="sacarFoto" />
</LinearLayout>
```

Actividad guiada

- En la clase MainActivity incorporar el método sacarFoto

```
public class MainActivity extends AppCompatActivity {  
  
    static final int REQUEST_IMAGE_CAPTURE = 1;  
  
    // ...  
  
    public void sacarFoto(View view){  
        Intent takePictureIntent = new Intent(MediaStore.ACTION_IMAGE_CAPTURE);  
        if (takePictureIntent.resolveActivity(getPackageManager()) != null) {  
            startActivityForResult(takePictureIntent, REQUEST_IMAGE_CAPTURE);  
        }  
    }  
}
```

Actividad guiada

- En la clase MainActivity incorporar el método sacarFoto

```
public class MainActivity extends AppCompatActivity {  
  
    static final int REQUEST_IMAGE_CAPTURE = 1;  
  
    // ...  
  
    public void sacarFoto(View view){  
        Intent takePictureIntent = new Intent(MediaStore.ACTION_IMAGE_CAPTURE);  
        if (takePictureIntent.resolveActivity(getPackageManager()) != null) {  
            startActivityForResult(takePictureIntent, REQUEST_IMAGE_CAPTURE);  
        }  
    }  
}
```

Ya vimos en clases pasadas el concepto de Intent implícitos.
En este ejemplo se delega en Android la búsqueda de una aplicación que pueda sacar fotos.

Actividad guiada

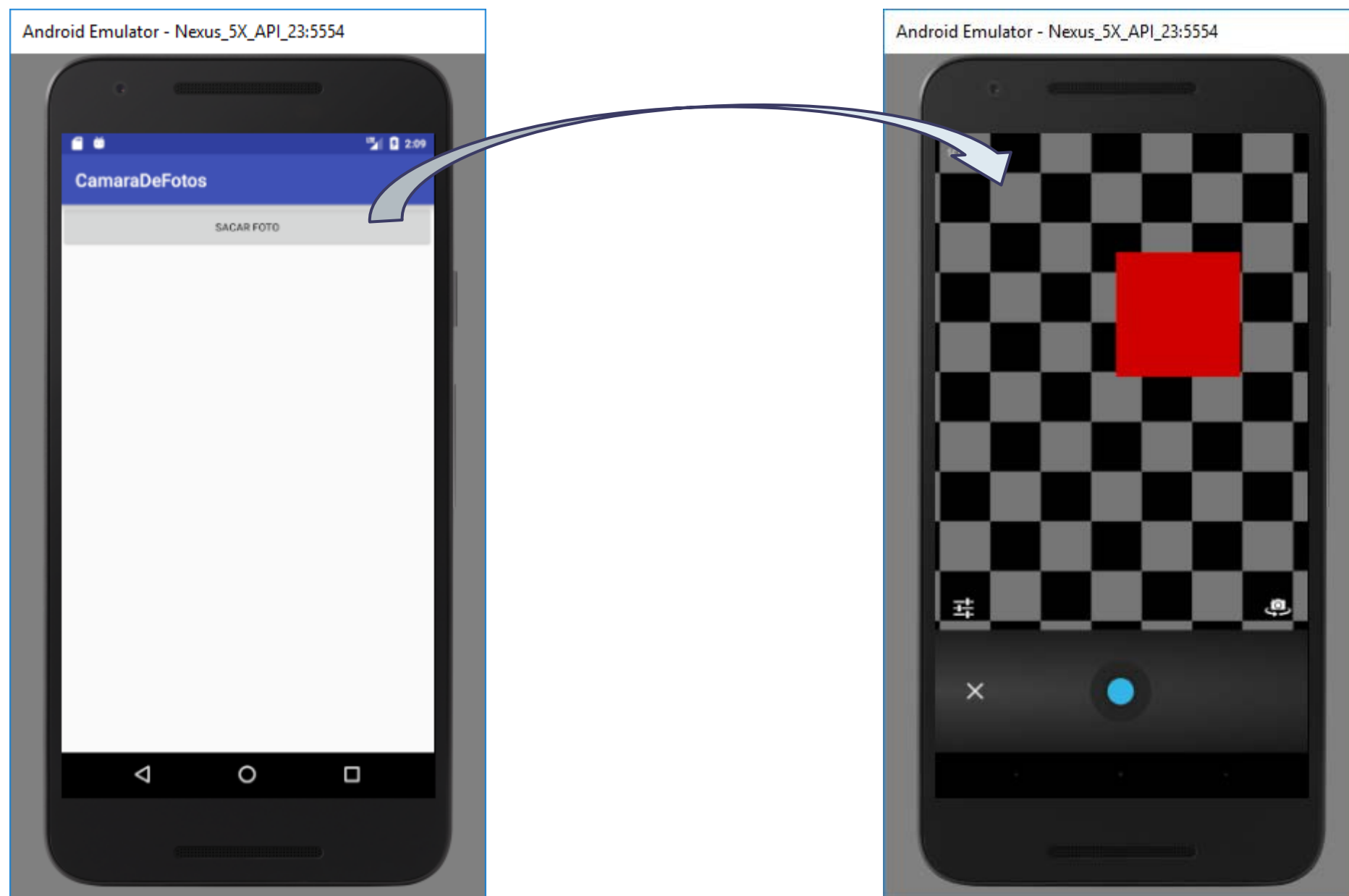
- En la clase MainActivity incorporar el método sacarFoto

```
public class MainActivity extends AppCompatActivity {  
  
    static final int REQUEST_IMAGE_CAPTURE = 1;  
  
    // ...  
  
    public void sacarFoto(View view){  
        Intent takePictureIntent = new Intent(MediaStore.ACTION_IMAGE_CAPTURE);  
        if (takePictureIntent.resolveActivity(getPackageManager()) != null) {  
            startActivityForResult(takePictureIntent, REQUEST_IMAGE_CAPTURE);  
        }  
    }  
}
```

Se debe verificar que efectivamente se puede resolver el Intent (podría no existir una app instalada que lo resuelva), de lo contrario tendríamos un error en ejecución al llamar a `startActivityForResult` y la app se detendrá.

Actividad guiada

- Probar en el emulador o en su dispositivo



Actividad guiada

- Se continuará mostrando un thumbnail de la imagen capturada.

Agregar en el activity_main.xml la definición de un ImageView donde se mostrará el thumbnail:

```
<LinearLayout ... >
    ...
    <ImageView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:id="@+id/imageView1" />
</LinearLayout>
```

Actividad guiada


- En la clase MainActivity incorporar el método onActivityResult

```
public class MainActivity extends AppCompatActivity {  
  
    static final int REQUEST_IMAGE_CAPTURE = 1;  
  
    // ...  
  
    @Override  
    protected void onActivityResult(int requestCode, int resultCode, Intent data) {  
        if (requestCode == REQUEST_IMAGE_CAPTURE && resultCode == RESULT_OK) {  
            Bundle extras = data.getExtras();  
            Bitmap imageBitmap = (Bitmap) extras.get("data");  
  
            ImageView imageView = (ImageView) findViewById(R.id.imageView1);  
            imageView.setImageBitmap(imageBitmap);  
        }  
    }  
}
```

Actividad guiada

- En la clase MainActivity incorporar el método onActivityResult

```
public class MainActivity extends AppCompatActivity {  
  
    static final int REQUEST_IMAGE_CAPTURE = 1;  
  
    // ...  
  
    @Override  
    protected void onActivityResult(int requestCode, int resultCode, Intent data) {  
        if (requestCode == REQUEST_IMAGE_CAPTURE && resultCode == RESULT_OK) {  
            Bundle extras = data.getExtras();  
            Bitmap imageBitmap = (Bitmap) extras.get("data");  
  
            ImageView imageView = (ImageView) findViewById(R.id.imageView1);  
            imageView.setImageBitmap(imageBitmap);  
        }  
    }  
}
```



Verificamos que estamos recibiendo el resultado del Intent adecuado

Actividad guiada

- En la clase MainActivity incorporar el método onActivityResult

```
public class MainActivity extends AppCompatActivity {  
  
    static final int REQUEST_IMAGE_CAPTURE = 1;  
  
    // ...  
  
    @Override  
    protected void onActivityResult(int requestCode, int resultCode, Intent data) {  
        if (requestCode == REQUEST_IMAGE_CAPTURE && resultCode == RESULT_OK) {  
            Bundle extras = data.getExtras();  
            Bitmap imageBitmap = (Bitmap) extras.get("data");  
  
            ImageView imageView = (ImageView) findViewById(R.id.imageView1);  
            imageView.setImageBitmap(imageBitmap);  
        }  
    }  
}
```

Recuperamos el Bitmap asociado al thumbnail

Actividad guiada

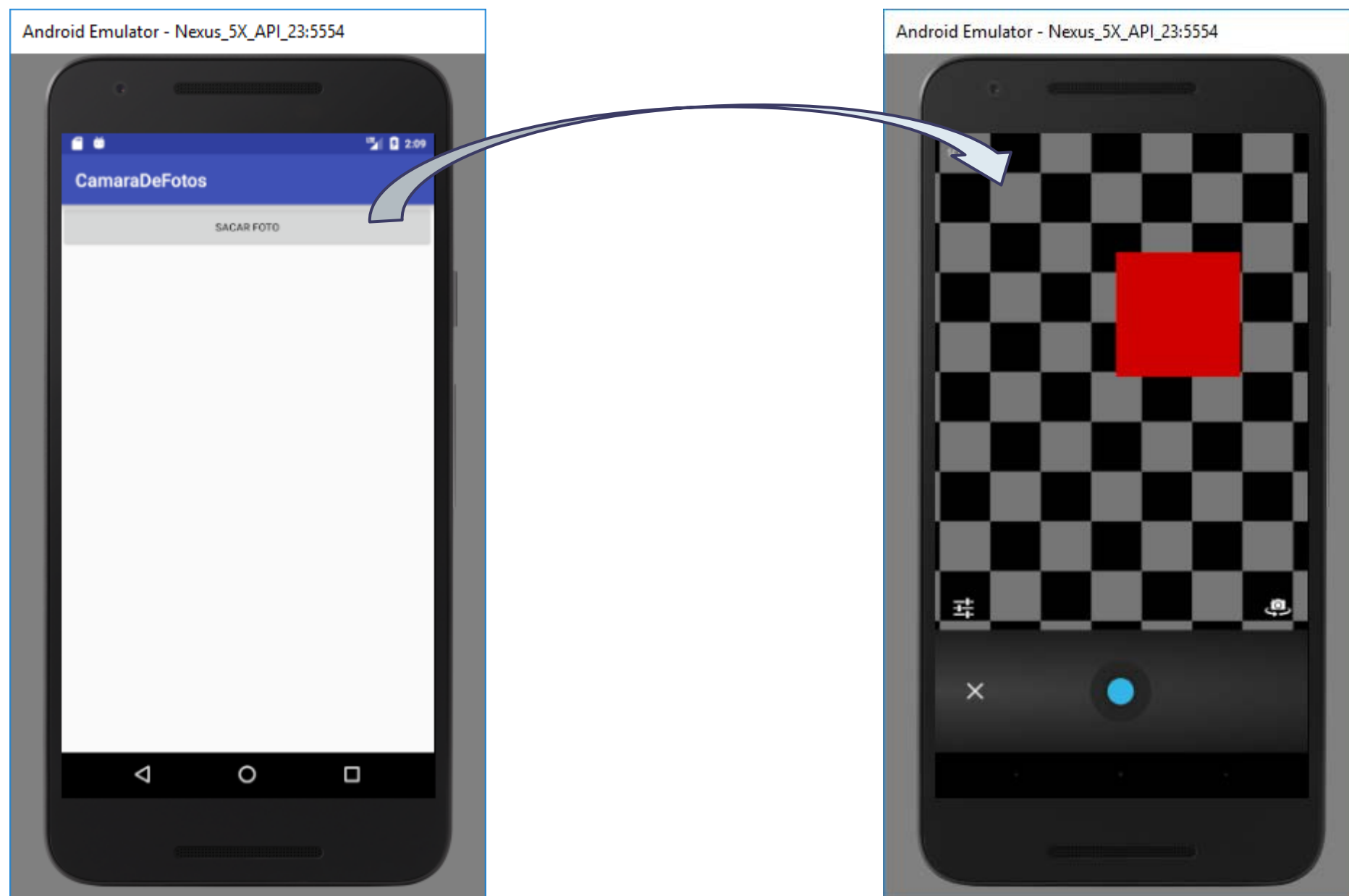
- En la clase MainActivity incorporar el método onActivityResult

```
public class MainActivity extends AppCompatActivity {  
  
    static final int REQUEST_IMAGE_CAPTURE = 1;  
  
    // ...  
  
    @Override  
    protected void onActivityResult(int requestCode, int resultCode, Intent data) {  
        if (requestCode == REQUEST_IMAGE_CAPTURE && resultCode == RESULT_OK) {  
            Bundle extras = data.getExtras();  
            Bitmap imageBitmap = (Bitmap) extras.get("data");  
  
            ImageView imageView = (ImageView) findViewById(R.id.imageView1);  
            imageView.setImageBitmap(imageBitmap);  
        }  
    }  
}
```

Lo asociamos al imageView ubicado en el layout de la actividad

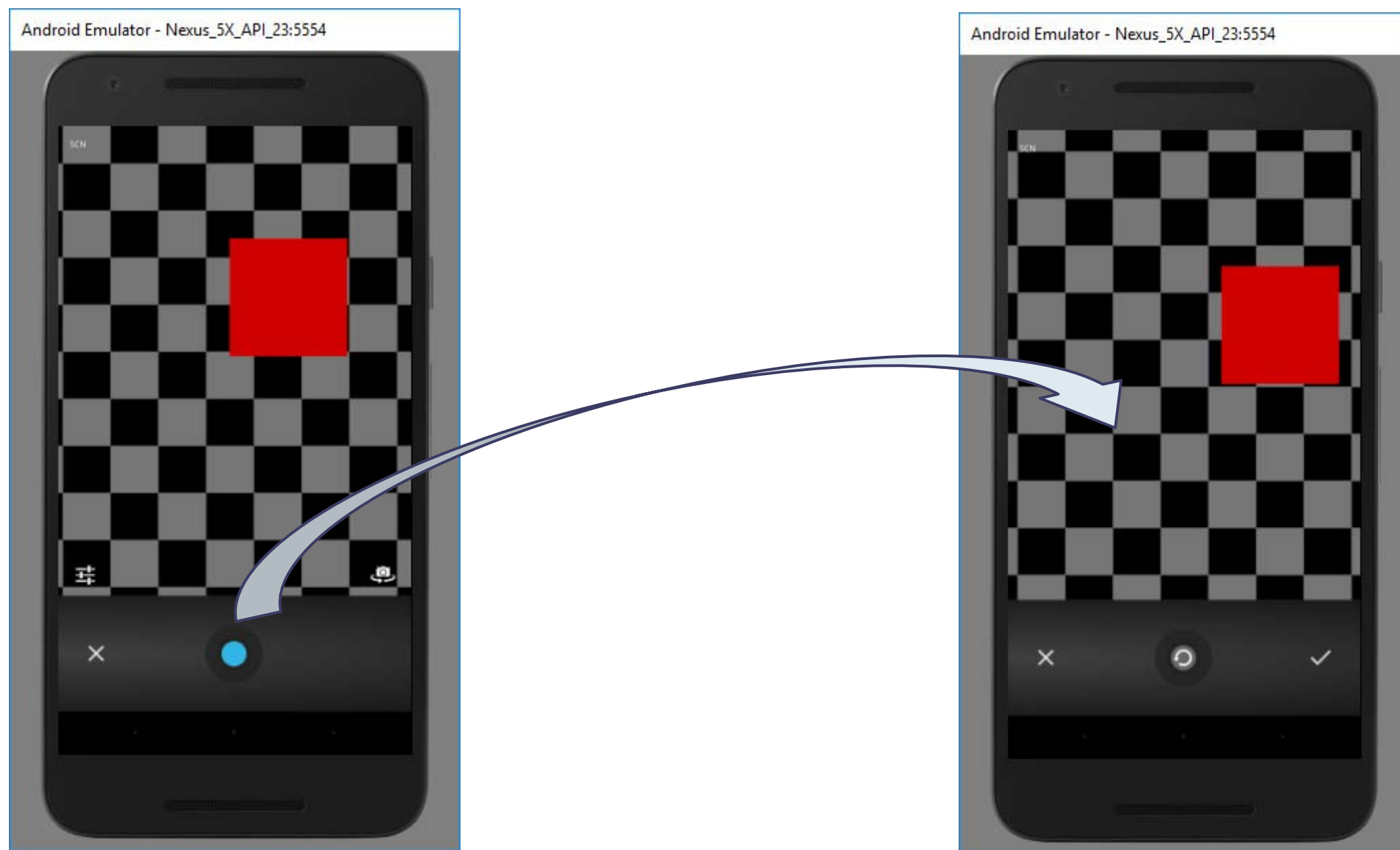
Actividad guiada

- Probar en el emulador o en su dispositivo



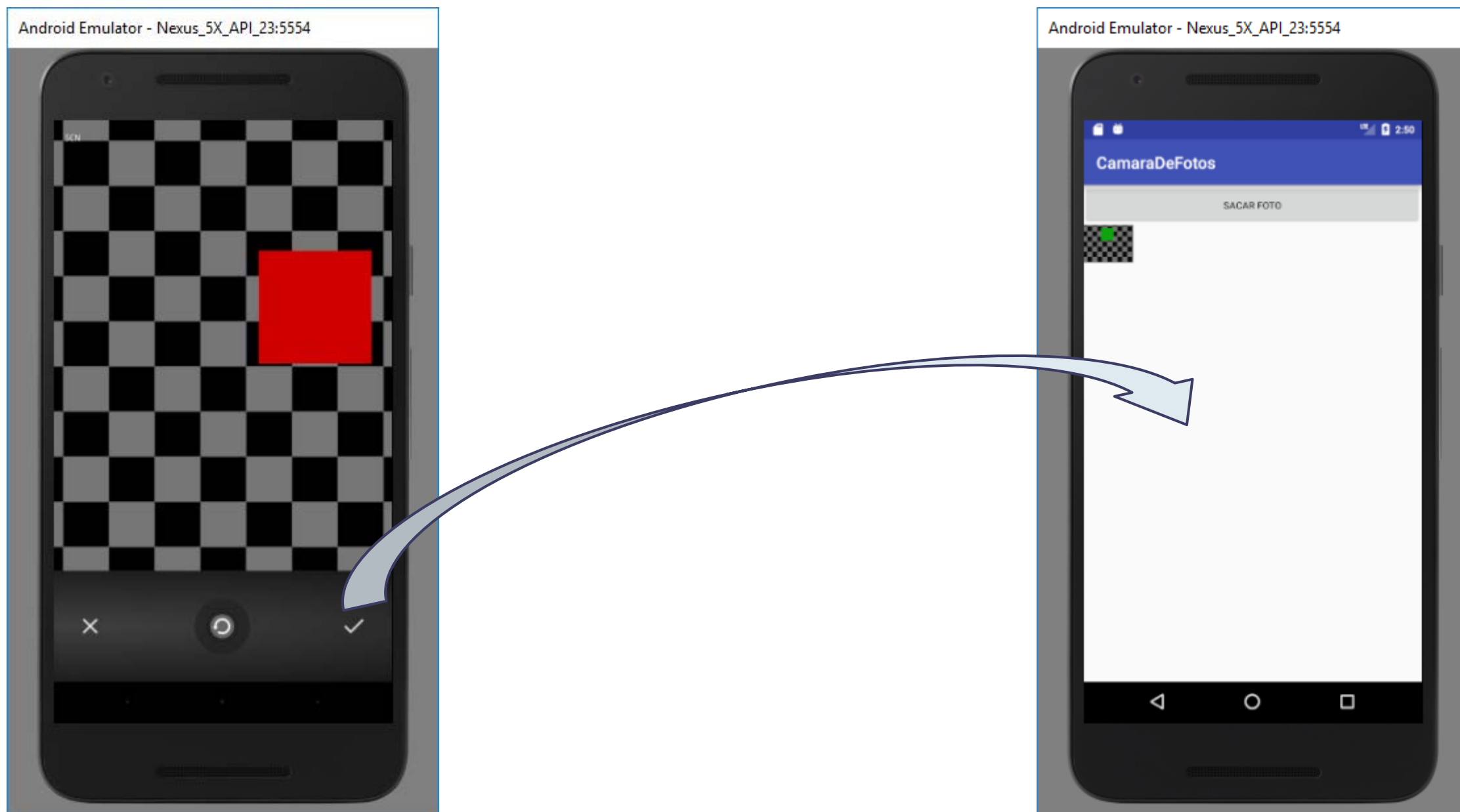
Actividad guiada

- Probar en el emulador o en su dispositivo



Actividad guiada

- Probar en el emulador o en su dispositivo



Localización Geográfica



Google Play Services

- Para obtener la ubicación actual del dispositivo en el que se está ejecutando una aplicación, se debe disponer de Google Play Services
 - En Android Studio, en el menú Tools hacer click en SDK Manager
 - hacer click en SDK Tools
 - expandir Support Repository
 - seleccionar Google Repository
 - hacer click en OK.
- Se podrá probar en
 - Dispositivos que tengan Android 4.0 o superior, y que incluyan Google Play Store
 - Emuladores con Android 4.2.2 o superior con Google APIs

Localización geográfica

- Crear un nuevo proyecto desde Android Studio
- Se debe solicitar permiso al usuario, a través del archivo Manifest.

```
<manifest ... >
    <uses-permission
        android:name="android.permission.ACCESS_FINE_LOCATION" />
    ...
</manifest>
```

- En build.gradle se debe establecer la dependencia con la librería de localización de Google Play Services.

```
dependencies {
    ...
    implementation 'com.google.android.gms:play-services-location:16.0.0'
    ...
}
```

Localización geográfica

- Editar el archivo activity_main.xml

```
<LinearLayout
    android:layout_height="match_parent"
    android:layout_width="match_parent"
    android:orientation="vertical"
    xmlns:android="http://schemas.android.com/apk/res/android">

    <TextView
        android:id="@+id/latitudTextView"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Consultando latitud..." />
    <TextView
        android:id="@+id/longitudTextView"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Consultando longitud..." />

    <Button
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="Obtener ubicación"
        android:onClick="obtenerUbicacion" />

</LinearLayout>
```

Localización geográfica

- Editar la clase MainActivity

```
public class MainActivity extends AppCompatActivity {  
  
    private FusedLocationProviderClient fusedLocationClient;  
    private LocationRequest mLocationRequest;  
  
    int REQUEST_PERMISSIONS_REQUEST_CODE = 10;  
  
    @Override  
    protected void onCreate(Bundle savedInstanceState) {  
        super.onCreate(savedInstanceState);  
        setContentView(R.layout.activity_main);  
  
        fusedLocationClient = LocationServices.getFusedLocationProviderClient(this);  
  
        mLocationRequest = LocationRequest.create()  
            .setInterval(5*1000)  
            .setPriority(LocationRequest.PRIORITY_HIGH_ACCURACY);  
    }  
}
```

Localización geográfica

- Editar la clase MainActivity

Recordar que desde Android 6 en adelante, el usuario acepta los permisos en tiempo de ejecución.

```
public void obtenerUbicacion(View view) {
```

```
    if (ActivityCompat.checkSelfPermission( this,
        Manifest.permission.ACCESS_FINE_LOCATION) != PackageManager.PERMISSION_GRANTED) {

        ActivityCompat.requestPermissions(this,
            new String[]{Manifest.permission.ACCESS_FINE_LOCATION},
            REQUEST_PERMISSIONS_REQUEST_CODE);

        return;
    }
```

```
    LocationCallback locationCallback = new LocationCallback() {
        @Override
        public void onLocationResult(LocationResult locationResult) {
            super.onLocationResult(locationResult);
            Location ubicacionActual = locationResult.getLastLocation();

            TextView latitudTV = findViewById(R.id.latitudTextView);
            latitudTV.setText( String.valueOf(ubicacionActual.getLatitude()) );

            TextView longitudTV = findViewById(R.id.longitudTextView);
            longitudTV.setText( String.valueOf(ubicacionActual.getLongitude()) );
        }
    };

    fusedLocationClient.requestLocationUpdates(mLocationRequest, locationCallback, null);
```

Se recibirán los updates de la ubicación en el método callback

```
}
```

Localización geográfica



Localización geográfica

Cambiar la ubicación desde las opciones del emulador

