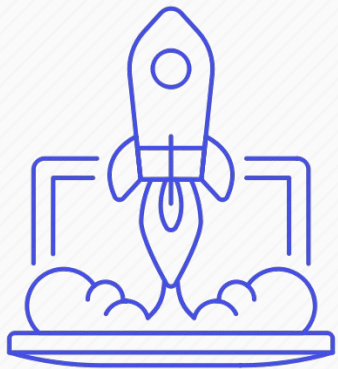


Universidad Nacional de La Matanza



Vitals de una aplicacion

Intro



Cuando lanzamos una app o una actualización, debemos monitorear su comportamiento

La preguntas que nos surgen son varias... ¿Qué herramientas tenemos? ¿Qué debemos mirar? ¿A qué debemos estar atentos? ¿Qué valores son aceptables?

Y otras cuestiones que intentaremos responder

Vitals

“Son un conjunto de métricas referidas mejorar a la estabilidad y el rendimiento de los dispositivos Android”

Ej: tiempo de inicio de la app, consumo de batería, etc.

Vitals

Google define ciertas “métricas esenciales”

- Tasas de ANR
- Tasas de fallas
- Demasiadas activaciones
- Bloqueos de activación parciales

Vitals

¿Donde podemos ver estas métricas?

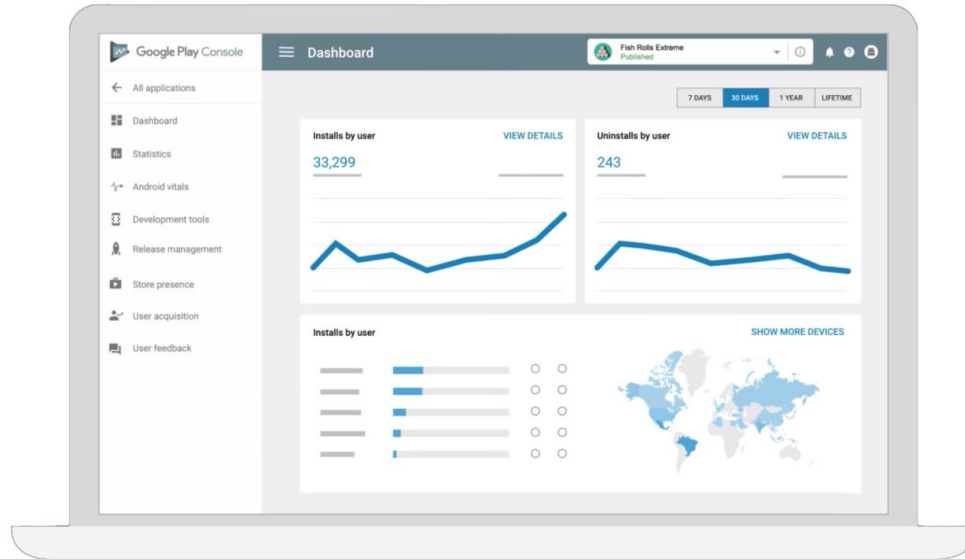


[Google Play Console](#)

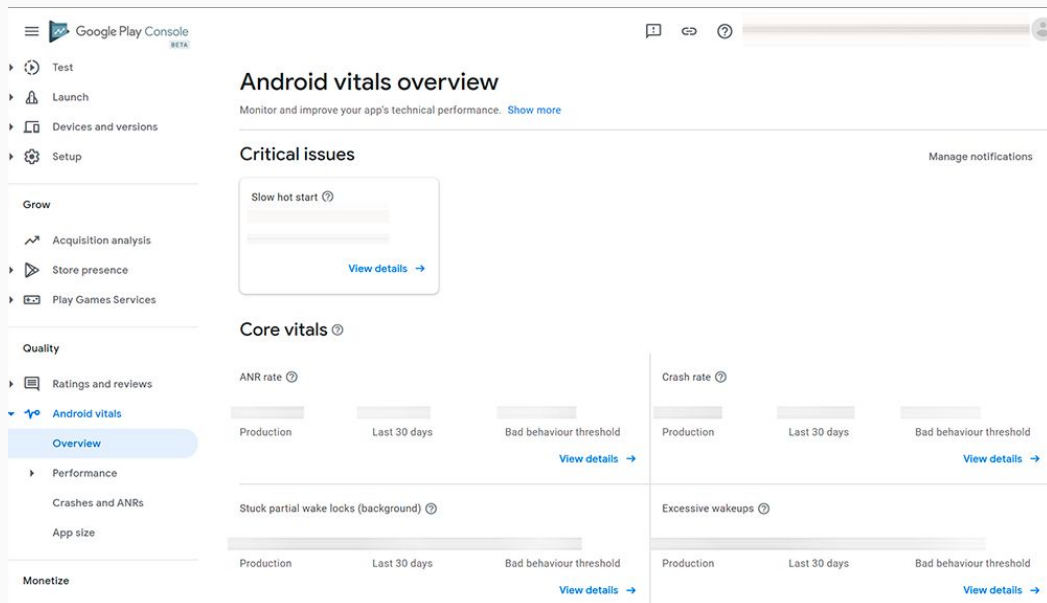


[App Store Connect](#)

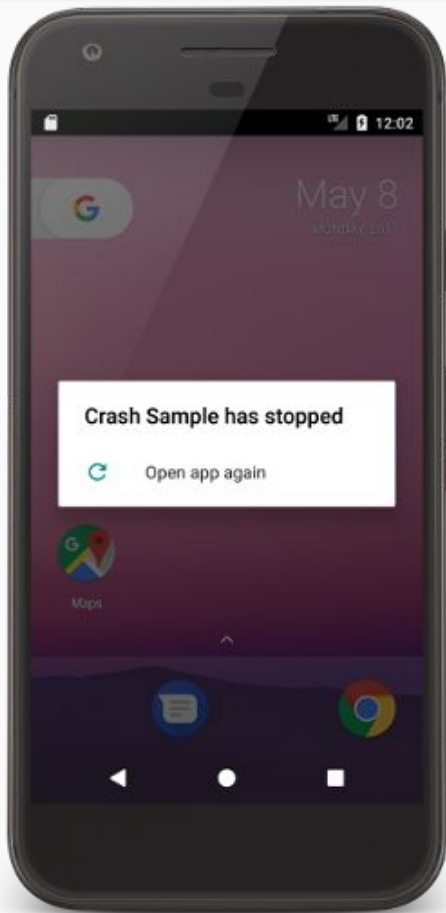
Vitals



Vitals

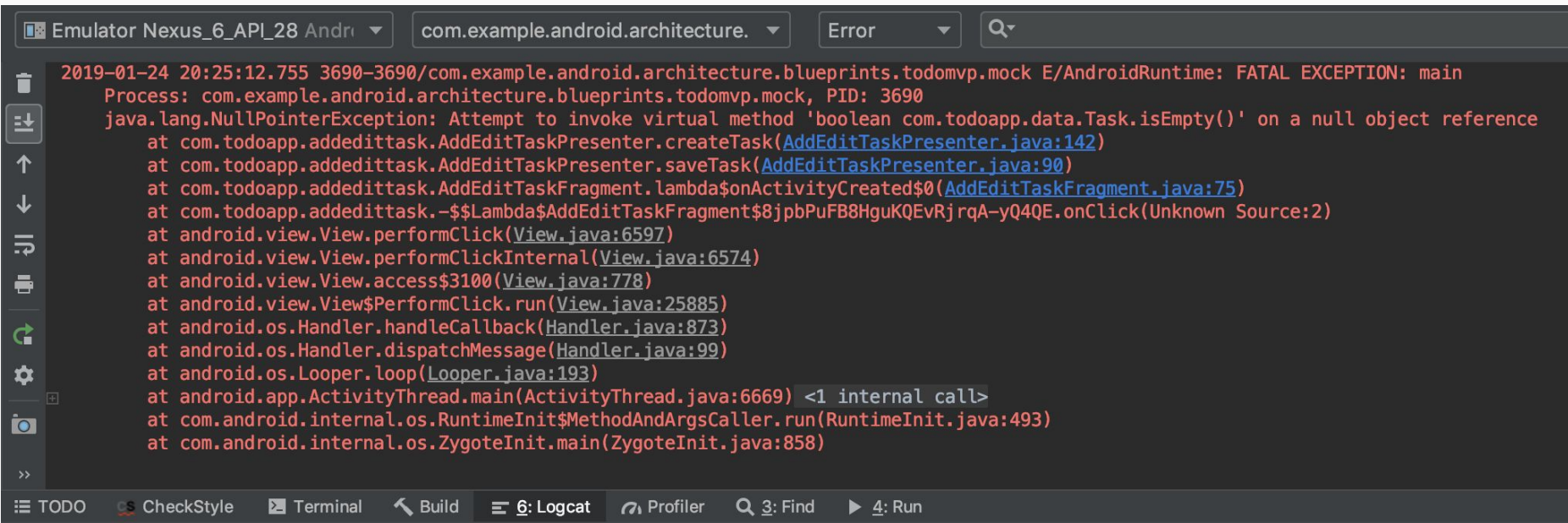


Crashes



- Un crash ocurre cuando se produce un error no controlado (*excepción*)
- El proceso de la aplicación **se cierra**
- Se pierde todo aquello que estaba en memoria y no fue guardado
- La forma de comprender en qué punto de la aplicación está el error es a través del *stacktrace*

Crashes



```
2019-01-24 20:25:12.755 3690-3690/com.example.android.architecture.blueprints.todomvp.mock E/AndroidRuntime: FATAL EXCEPTION: main
Process: com.example.android.architecture.blueprints.todomvp.mock, PID: 3690
java.lang.NullPointerException: Attempt to invoke virtual method 'boolean com.todoapp.data.Task.isEmpty()' on a null object reference
    at com.todoapp.addedittask.AddEditTaskPresenter.createTask(AddEditTaskPresenter.java:142)
    at com.todoapp.addedittask.AddEditTaskPresenter.saveTask(AddEditTaskPresenter.java:90)
    at com.todoapp.addedittask.AddEditTaskFragment.lambda$onActivityCreated$0(AddEditTaskFragment.java:75)
    at com.todoapp.addedittask.-$$Lambda$AddEditTaskFragment$8jpbPuFB8HguKQEvRjrQa-yQ4QE.onClick(Unknown Source:2)
    at android.view.View.performClick(View.java:6597)
    at android.view.View.performClickInternal(View.java:6574)
    at android.view.View.access$3100(View.java:778)
    at android.view.View$PerformClick.run(View.java:25885)
    at android.os.Handler.handleCallback(Handler.java:873)
    at android.os.Handler.dispatchMessage(Handler.java:99)
    at android.os.Looper.loop(Looper.java:193)
    at android.app.ActivityThread.main(ActivityThread.java:6669) <1 internal call>
    at com.android.internal.os.RuntimeInit$MethodAndArgsCaller.run(RuntimeInit.java:493)
    at com.android.internal.os.ZygoteInit.main(ZygoteInit.java:858)
```

Emulator Nexus_6_API_28 Andri com.example.android.architecture. Error 🔍

TODO CheckStyle Terminal Build 6: Logcat Profiler 3: Find 4: Run

Crashes

Crash rate

Percentage of daily sessions during which your users experienced at least one crash. A daily session refers to a day during which your app was used.

[Learn more](#)

0.50%

Last 30 days

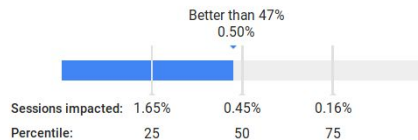
—

Previous 30 days

1.09%

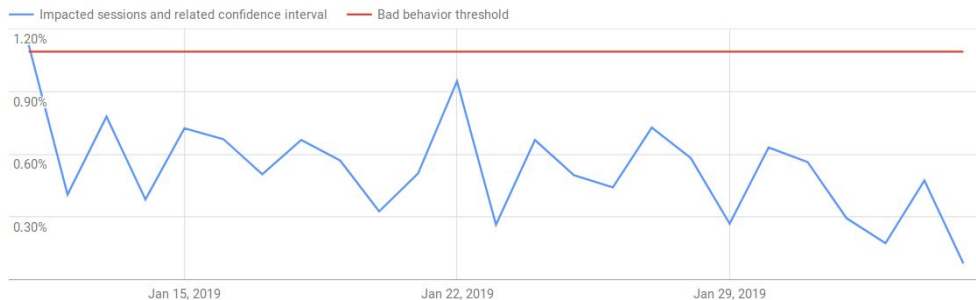
Bad behavior threshold

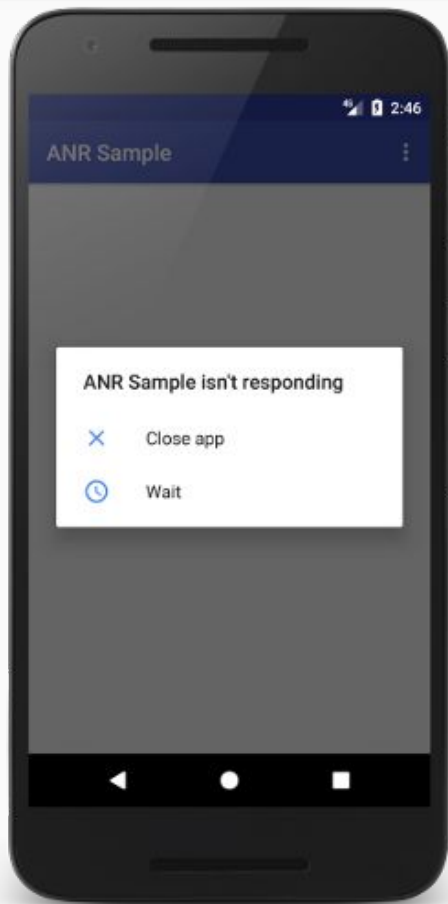
Benchmark ⓘ



Crash rate over time

Show confidence interval ☐





- ANR significa *Application Not Responding*
- Se produce cuando se bloquea por mucho tiempo el thread de la UI de la app
- La aplicación deja de responder al usuario, dando la sensación de que se “congela”
- Aparece un mensaje para cerrar la aplicación o esperar que responda

- [Crashes](#)
- [ANRs](#)
- [Building Mobile At Scale: 6. App Crashes](#)
- [Diagnosing Issues Using Crash Reports and Device Logs](#)

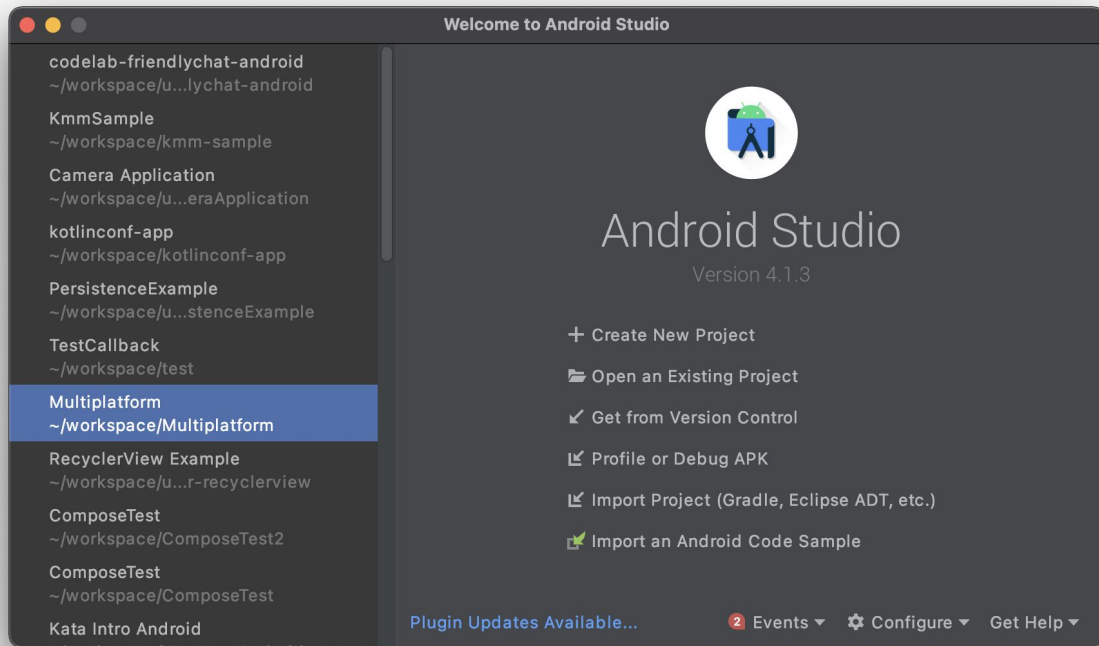
¿Preguntas?

Anexo configuración Firebase

Abrir proyecto existente

Al abrir Android Studio, podremos optar por crear un nuevo proyecto o abrir uno existente

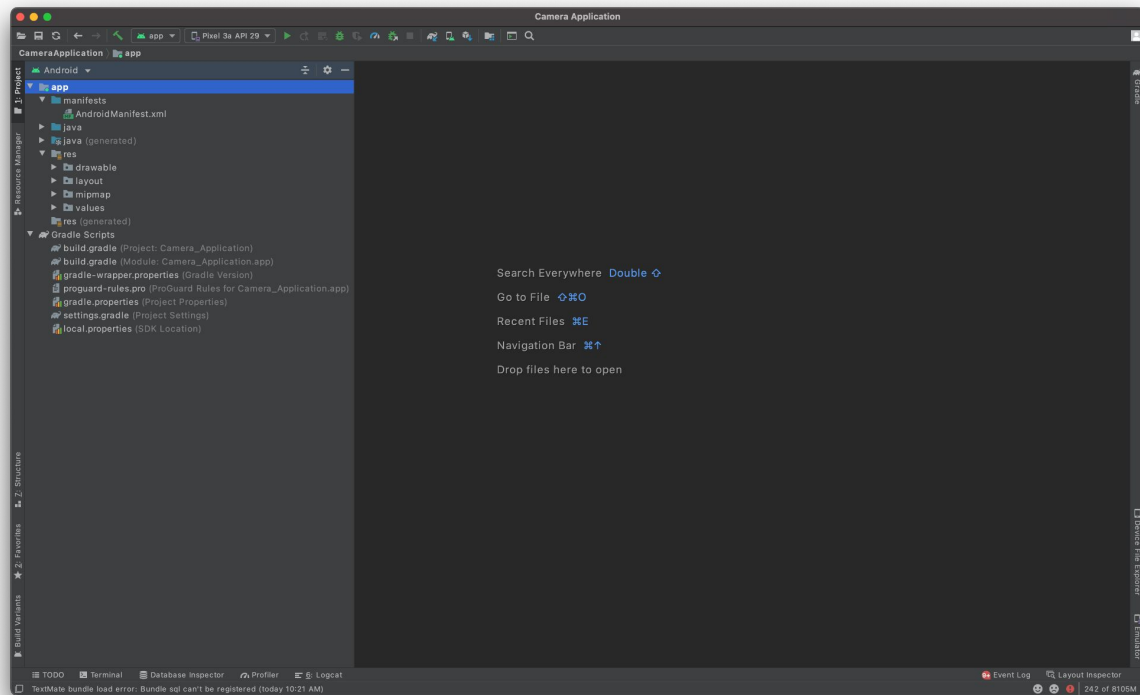
En la izquierda se muestran los proyectos recientes



Elementos de un proyecto

En el panel 1: **Project** y utilizando la vista “**Android**” se muestran los módulos que componen nuestro proyecto

Por defecto, al crear un nuevo proyecto existe uno solo de nombre **app** donde encontraremos el código de nuestra aplicación



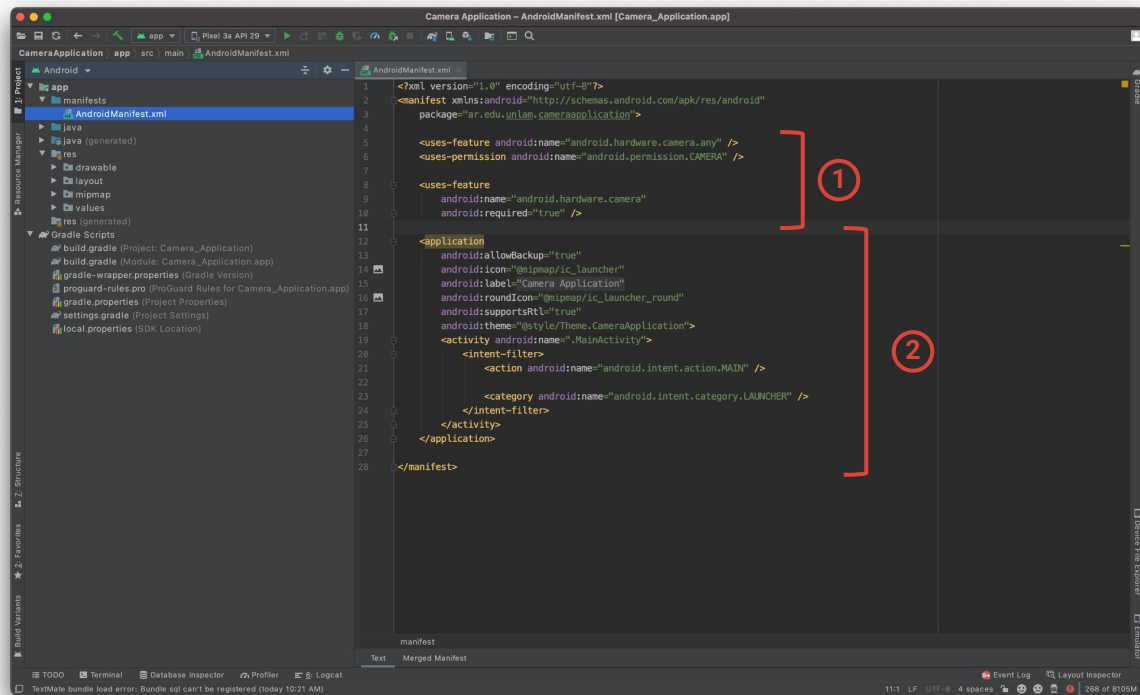
Elementos de un proyecto

Cada modulo contiene un archivo AndroidManifest.xml.

Este archivo, entre otras cosas, incluye:

1 - Declaración de permisos (ej: ubicación, internet) y funcionalidades (camara, bluetooth) que usa ese módulo

2 - Propiedades de la aplicación y componentes (activities, services, broadcast receivers y content providers)

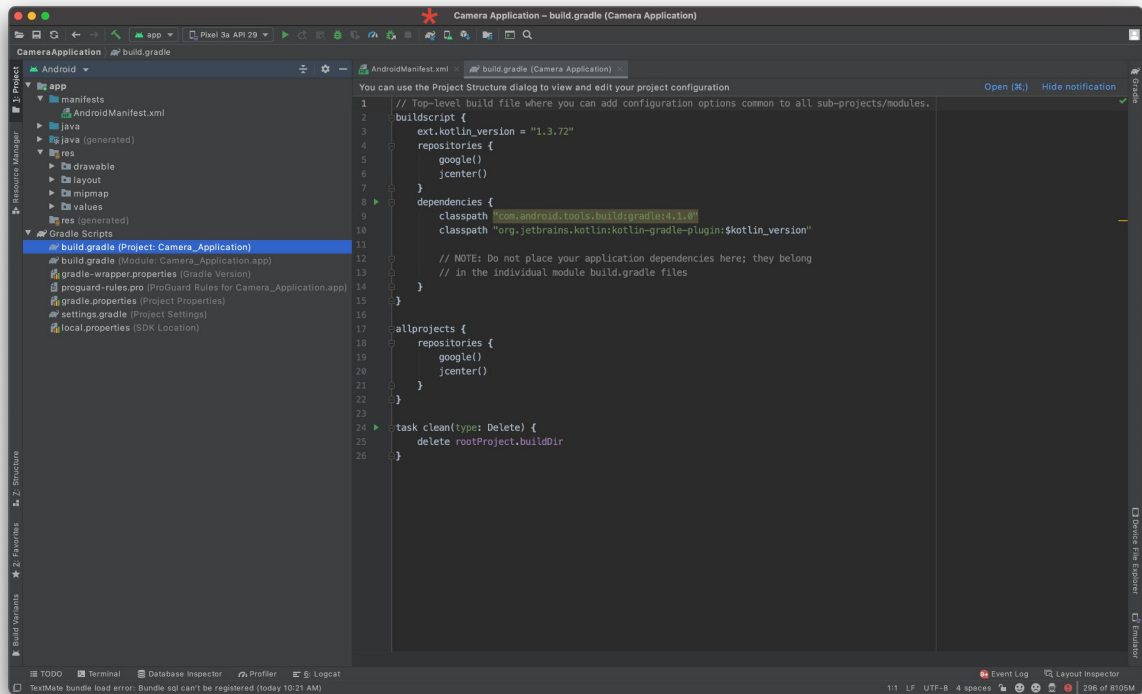


Elementos de un proyecto

Un archivo **build.gradle** general para todo el proyecto, con configuraciones globales.

Ej: versión de plugin de Kotlin para Gradle, versión de plugin de Android para Gradle

** Cada cambio en un archivo **build.gradle**, requiere hacer un “gradle sync”*



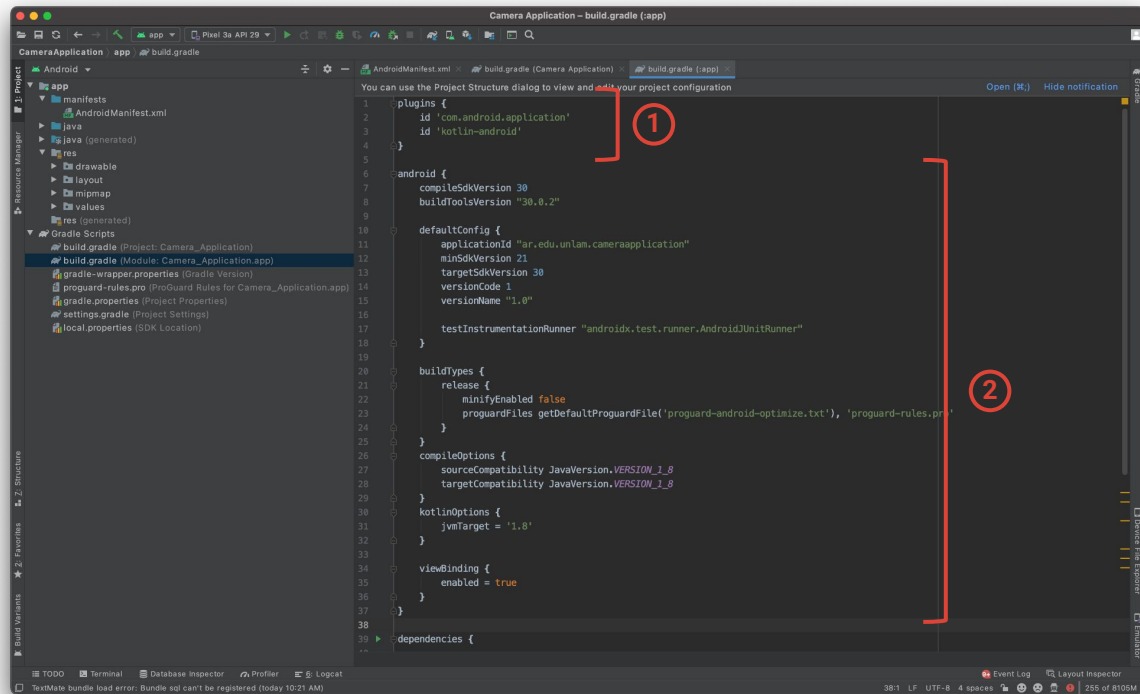
Elementos de un proyecto

Un archivo **build.gradle** por cada módulo del proyecto, con configuraciones en particular para cada uno.

En este caso tenemos uno para el módulo app:

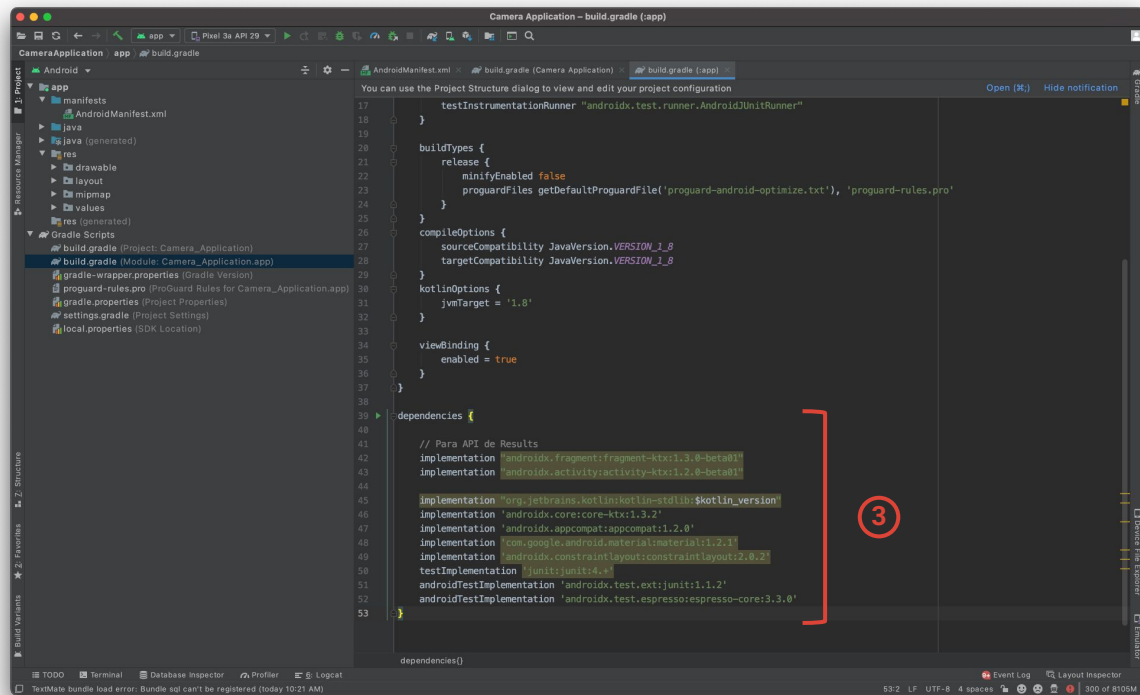
1 - Plugins: Definen que tipo de módulo es (librería, app) y agregan funcionalidades

2 - Android: Configuraciones relativas a Android, como ser: Id de la aplicación, mínima versión de Android soportada, variantes, firmas, etc



Elementos de un proyecto

3 - Dependencias a librerías externas, por ejemplo, de descarga de imagenes, de testing, etc.



Integrar Firebase

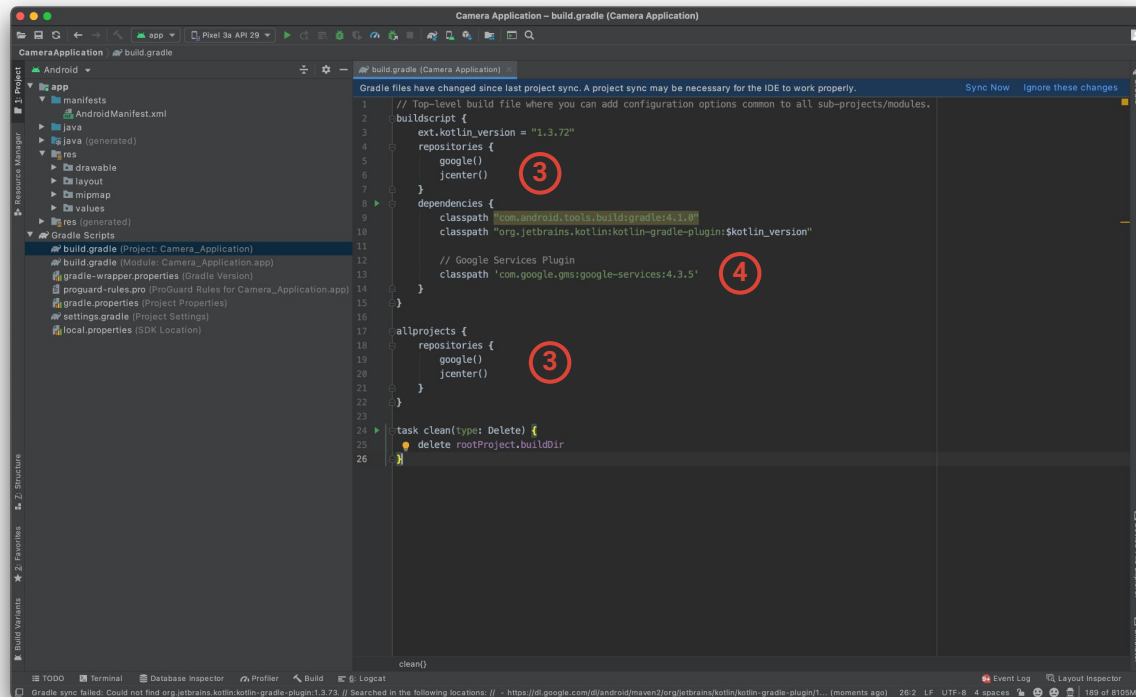
1 - Crear el proyecto en la consola de Firebase

2 - Descargar el archivo *google-services.json* y colocarlo dentro del directorio del módulo *app*

En el archivo *build.gradle* del **proyecto**:

3 - Asegurarse que este agregado **google()** en los repositorios

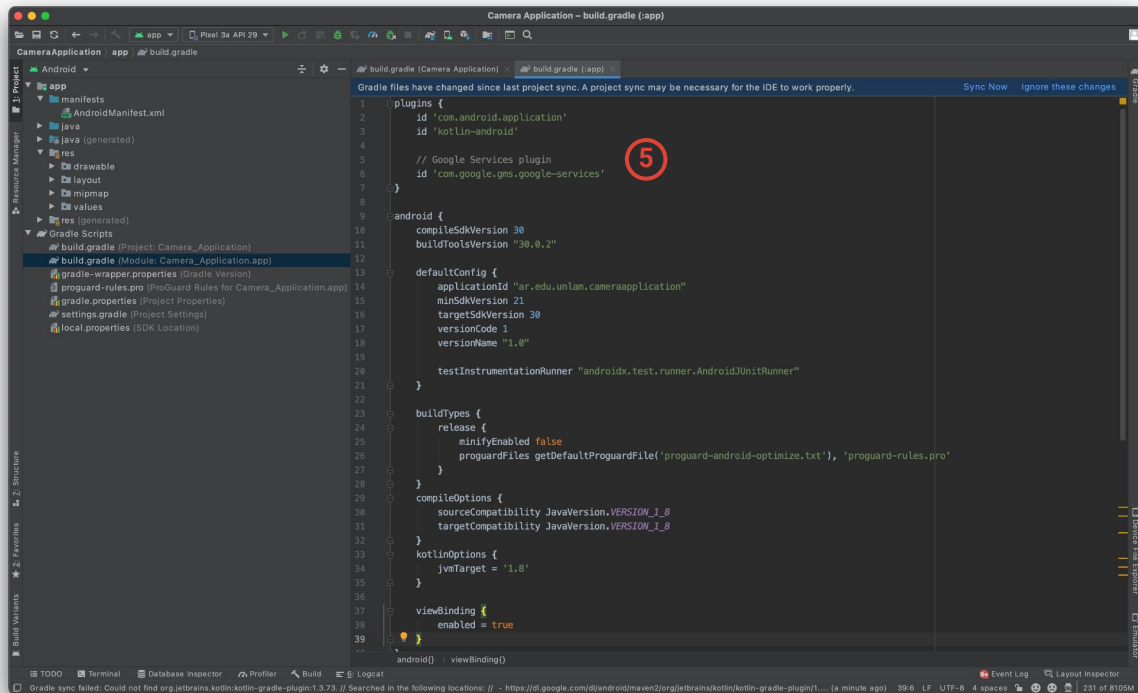
4 - Agregar el plugin de Google Services



Integrar Firebase

En el archivo *build.gradle* del **modulo app**:

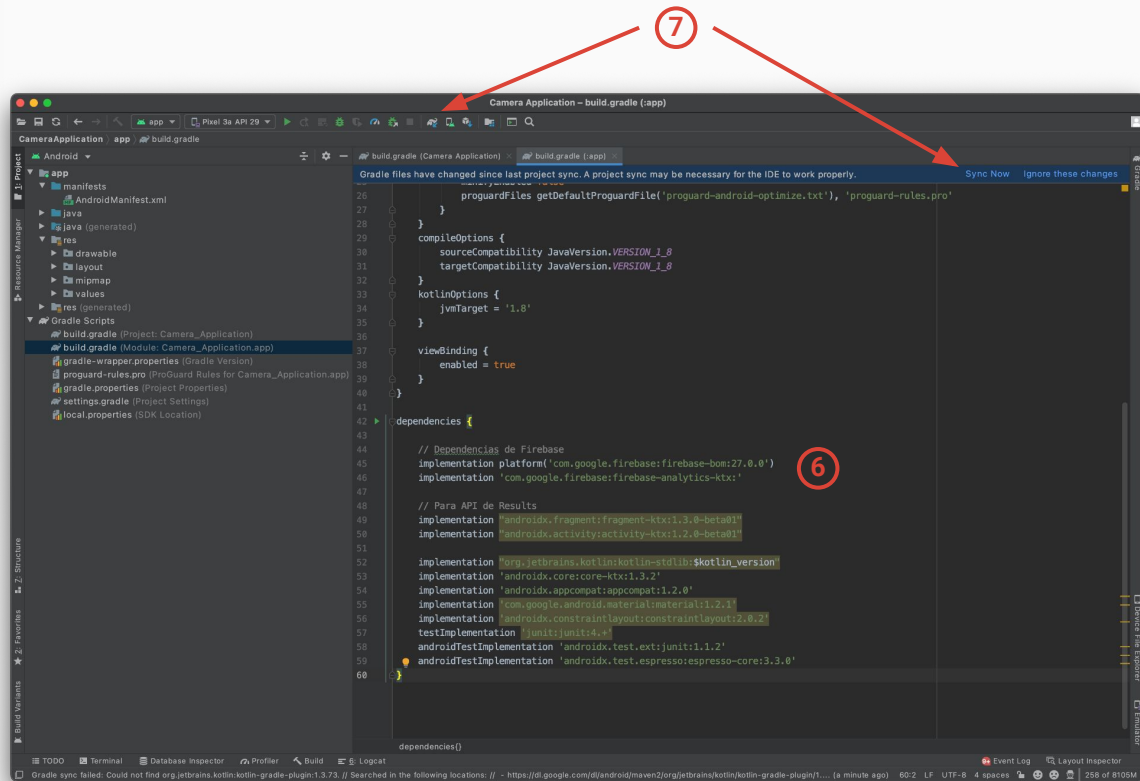
5 - Agregar el plugin de Google Services



Integrar Firebase

6 - Agregar las dependencias de Firebase (En el ejemplo se agrega Analytics)

7 - Realizar un *Gradle Sync* en cualquiera de los lugares señalados



Integrar Firebase

8 - Lanzar la aplicación

Como agregamos la librería de Analytics de Firebase, podemos ver que se registró el inicio de sesión del usuario

