Android Developer Fundamentals

What's Next?

Lesson 13

13.1 Permissions, performance and security

Contents

- Permissions
- Performance
- Security

This lesson has concepts only. It does not have a practical.

Permissions

Permissions

You have already used permissions in some of your apps:

- Permission to connect to the Internet
- Permission to use data from a Content Provider

A basic Android application has no permissions so that it cannot do anything that adversely impacts the user experience or data on the device

When to use permissions

App must get permission to do anything that

- Uses data or resources that the app did not create
- Uses network, hardware, features that do not belong to it
- Affects the behavior of the device
- Affects the behaviour of other apps

If it isn't yours, get permission!

How apps declare permissions they need

List permissions in the manifest

- <uses-permission>

```
<manifest ...>
 <uses-permission android:name="android.permission.READ_CONTACTS" />
 <uses-permission android:name="android.permission.READ_CALENDAR" />
 <uses-permission android:name="android.permission.CALL_PHONE"/>
```

Example permissions

- ACCESS_COARSE_LOCATION
- ACCESS_FINE_LOCATION
- ACCESS_NETWORK_STATE
- ACCESS_WIFI_STATE
- GET_ACCOUNTS
- RECEIVE_SMS
- CAMERA
- RECORD_AUDIO
- MODIFY_AUDIO_SETTINGS
- ADD_VOICEMAIL

See more at

https://developer.android.com/reference/android/Manifest.permission.html

More examples

- BLUETOOTH
  Connect to paired bluetooth devices.
- BODY_SENSORS
  Access data from sensors that the user uses to measure what is happening inside user's body, such as heart rate.
- USE_FINGERPRINT
  Use fingerprint hardware.

Normal and dangerous permissions

- **Normal** permissions do not directly risk the user's privacy

  Example: Set the time zone

  Android automatically grants normal permissions.

- **Dangerous** permissions give access to user's private data

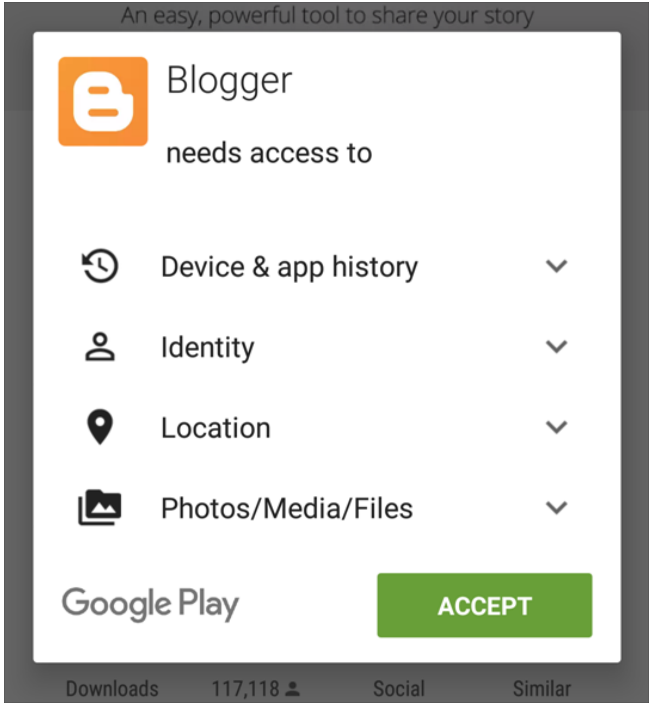
  Example: Read the user's contacts

  Android asks user to explicitly grant dangerous permissions

How users grant permission

For apps created before Marshmallow

- Users grant permission before installing



An easy, powerful tool to share your story

Blogger

needs access to

Device & app history

Identity

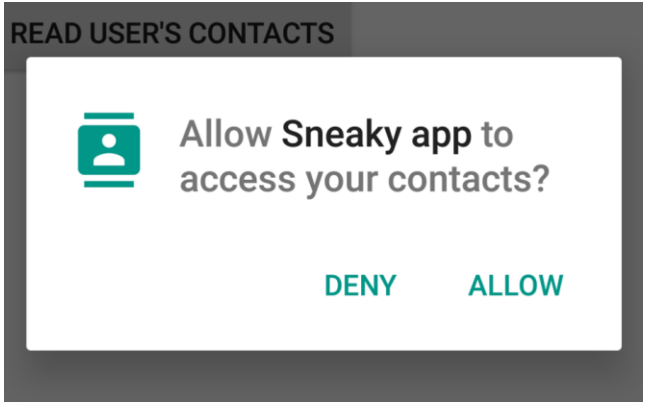
Location

Photos/Media/Files

How Users grant permission

Marshmallow onwards

- Installation doesn't ask user to give permissions
- App must get runtime permission



READ USER'S CONTACTS

Allow **Sneaky app** to access your contacts?

How users revoke permission

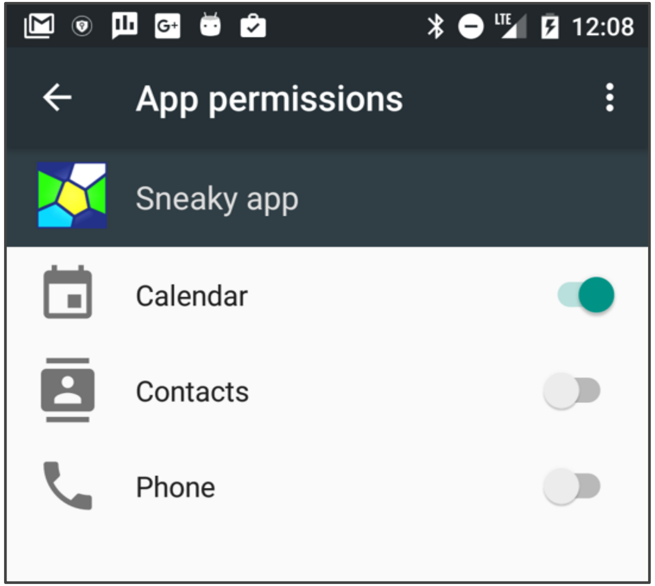
- Before Marshmallow

  Uninstall app!

- Marshmallow onwards

  Revoke individual permissions

  Settings > apps > permissions



12:08

App permissions

Sneaky app

Calendar

Contacts

Phone

Differences in permission models

Before Marshmallow

- If app is running, it can assume that user granted permissions during installation

After Marshmallow

- App needs to get permission at runtime
- Must check if it still has permission every time
- User can revoke permissions at any time

Framework versus support library

- Android framework 6.0 (API level 23) + provides permission methods
- Better to use Android Support Library

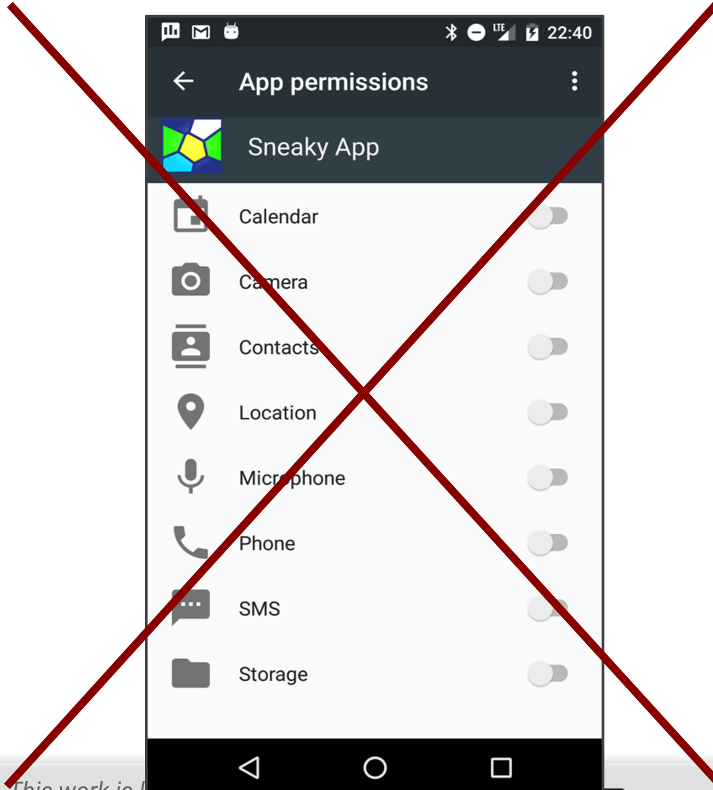
Support library permission methods

Support library permission methods check Android version

- If runtime permissions model is supported, requests the appropriate permission from the user
- Otherwise, checks if the permission was granted at install time

Best practices for permissions

- Ask for the least amount of permissions that you need
- Don't overwhelm the user
- Consider using an Intent instead—For example, send an Intent to use the camera



22:40

App permissions

Sneaky App

Calendar

Camera

Contacts

Location

Microphone

Phone

SMS

Storage

Learn more about permissions

- List of permissions defined in Android:
  developer.android.com/reference/android/Manifest.permission.html
- Permissions in Android:
  developer.android.com/guide/topics/security/permissions.html
- Best practices:
  developer.android.com/training/permissions/best-practices.html
- Blog entry about runtime permissions:
  android-developers.blogspot.com/2015/08/ building-better-apps-with-runtime.html

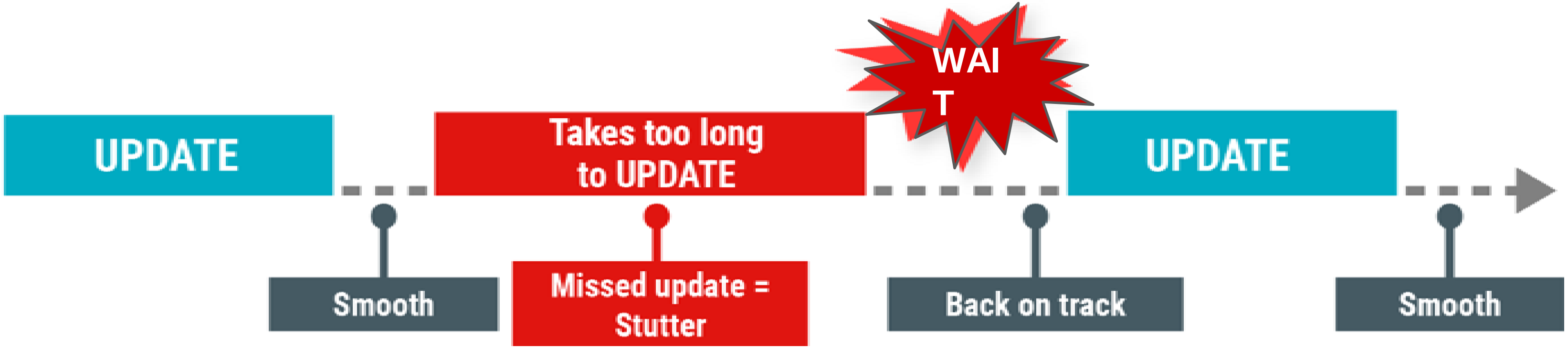
App Performance

What is app performance?

- Speed
- Responsiveness
- Smoothness
- Consistency
- Resource-efficiency

The Main thread must be fast

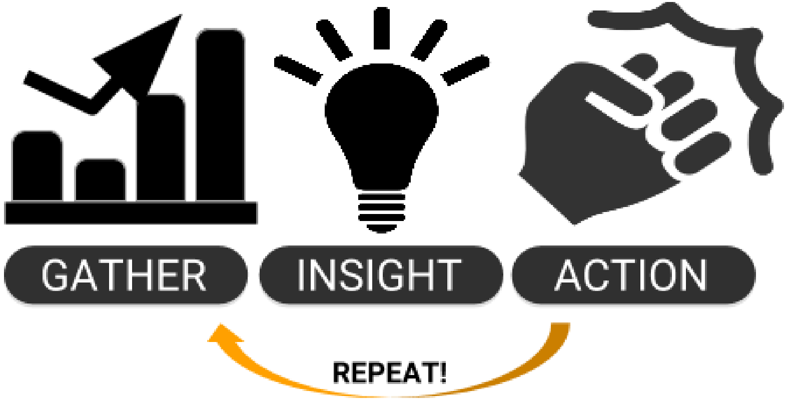
- Hardware updates screen every 16 milliseconds
- UI thread has 16 ms to do all its work
- If it takes too long, app stutters or hangs



Improving performance

- Be systematic
  - Gather information
  - Gain insight
  - Take action
- Iterate
- Use tools

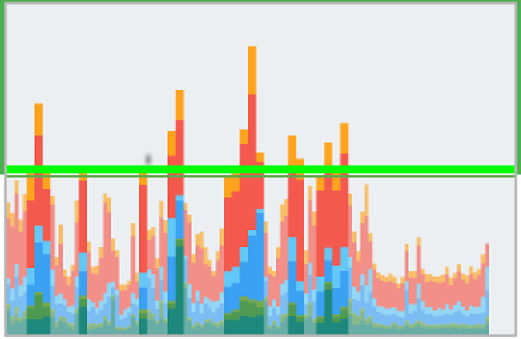
Performance Improvement



Profile GPU Rendering tool

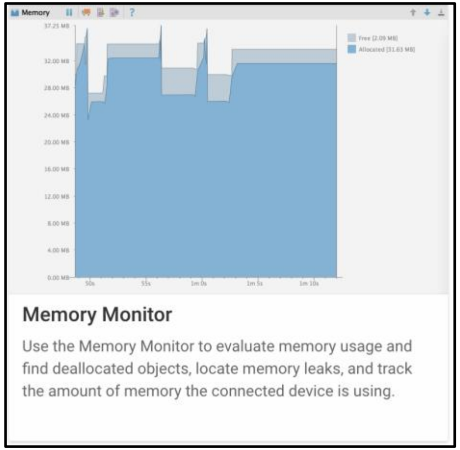
1. On your mobile device, go to Settings > Developer Options
2. In Monitoring section, select **Profile GPU Rendering**.
3. In Profile GPU Rendering popup, choose **On screen as bars**
4. Go to the app that you want to profile
5. See the bars at the bottom of the screen

Interpret the bars



- One bar represents one screen rendered
- If a bar goes above the green line, it took > 16 ms to render
- Many bars above the line, or heavy spiking indicate problems
- User will see stuttering or inconsistent responsiveness
- Analyze the bar and fix problems

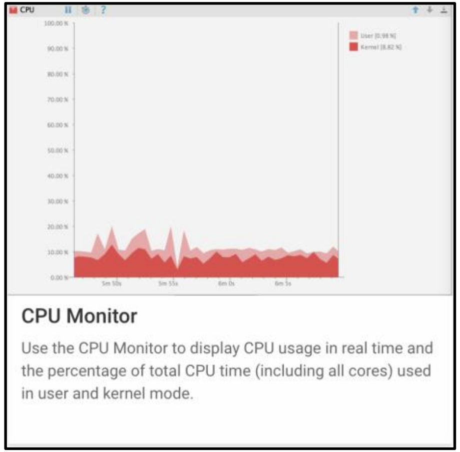
Android Studio > Android Monitor



Memory Monitor

Use the Memory Monitor to evaluate memory usage and find deallocated objects, locate memory leaks, and track the amount of memory the connected device is using.

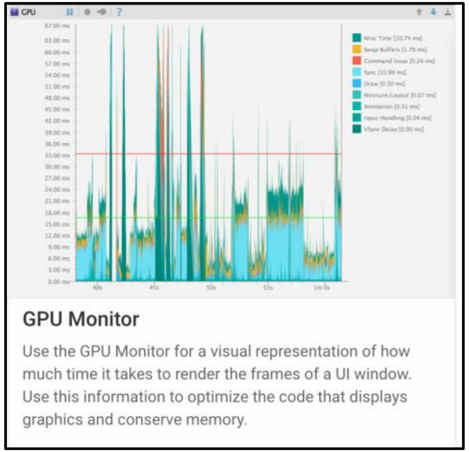
Memory



CPU Monitor

Use the CPU Monitor to display CPU usage in real time and the percentage of total CPU time (including all cores) used in user and kernel mode.

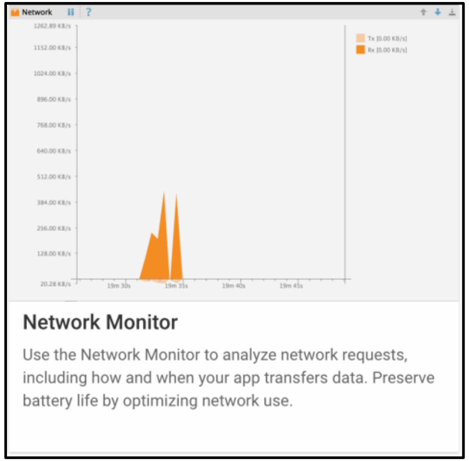
CPU



GPU Monitor

Use the GPU Monitor for a visual representation of how much time it takes to render the frames of a UI window. Use this information to optimize the code that displays graphics and conserve memory.

GPU



Network Monitor

Use the Network Monitor to analyze network requests, including how and when your app transfers data. Preserve battery life by optimizing network use.

Network

Don't make users wait

- Load data in background
- Pre-fetch data
- Move work off the UI thread
- Optimize UI—draw less and faster
- Eliminate overdraw and optimize view hierarchy

Don't waste user's resources

- WiFi and mobile radio use lots of battery
  - Batch requests
  - Schedule to run when phone is being charged
- Large images consume lots of memory
  - Use smallest images possible
  - Always use compressed image formats. Use WebP when possible
- Getting and putting data on the internet uses up data plans
  - When possible download data when on WiFi

Learn more about Performance — Introductory

- You, Your App and Performance
- Exceed the Android Speed Limit
- Only Draw What You See
- Simplify Complex View Hierarchies
- Understanding Compression
- Android Performance Patterns (YouTube series)

Learn more about performance — Tools

- Profile GPU Rendering Walkthrough
- Debug GPU Overdraw Walkthrough
- Hierarchy Viewer Walkthrough
- Battery Stats and Battery Historian
- Android Monitor Overview
- Performance Profiling Tools

Learn more about performance — Guides

- Analyzing with Profile GPU Rendering
- Optimizing for Battery Life
- Optimize Memory Use
- … and much more

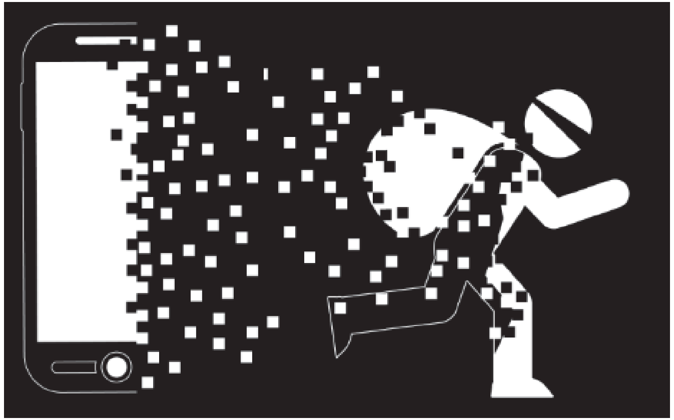
Security Best Practices

Android's got you covered (mostly)

- Android has built-in security features
- Significantly reduces the frequency and impact of application security issues
- You can typically build apps with default system and file permissions

Your app's responsibility

- Keep user's private data safe
- Do not leak secret things
- Treat user's data with integrity
- Keep your own app and data safe



Handling user data

- Minimize access to sensitive or personal user data
- Do not store or transmit user data if possible
- But if you must, consider using a hash or non-reversible form of the data

  For example, use hash of an email address as a primary key, so you do not store or transmit the email address

Comply with Personal Data Policies

If your app accesses personal information like passwords or usernames, it might need a privacy policy explaining how it uses and stores user data

Be careful what you log

- Android logs are a shared resource, and are available to an application with the READ_LOGS permission
- Inappropriate logging of user information could leak user data to other applications

Encrypt sensitive data

- Encrypt local files that contain sensitive data
- Store the key so it is not accessible by the app
  For example, use a KeyStore protected with a user password stored off the device

External storage

Do not store sensitive information on external storage

- Files created on external storage, such as SD Cards, are globally readable and writable
- External storage can be removed by the user
- External storage can be modified by any application
- Validate input on data from external storage

IP Networking

- Networking on Android is similar to other Linux environments
- Minimize network transactions that transmit private data
- Use HTTPS over HTTP wherever it's supported on the server
- Mobile devices often connect on networks that are not secured, such as public Wi-Fi hotspots
- You can implement authenticated, encrypted socket-level communication using the SSLSocket class

Learn more about security in Android

- Security tips:
  developer.android.com/training/articles/security-tips.html
- Saving files:
  developer.android.com/training/basics/data-storage/files.html
- Sharing files:
  developer.android.com/training/secure-file-sharing/index.html
- Full disk encryption:
  source.android.com/security/encryption/

What's Next?

- Concept Chapter: 13.1 C Permissions, performance and security
- No practical!

END

Android Developer Fundamentals

What's Next?

Lesson 14

This work is licensed under a Creative Commons Attribution-NonCommercial 4.0 International License

14.1 Firebase and Monetization

Contents

- Firebase
- Make money from your app

Firebase

What is Firebase?

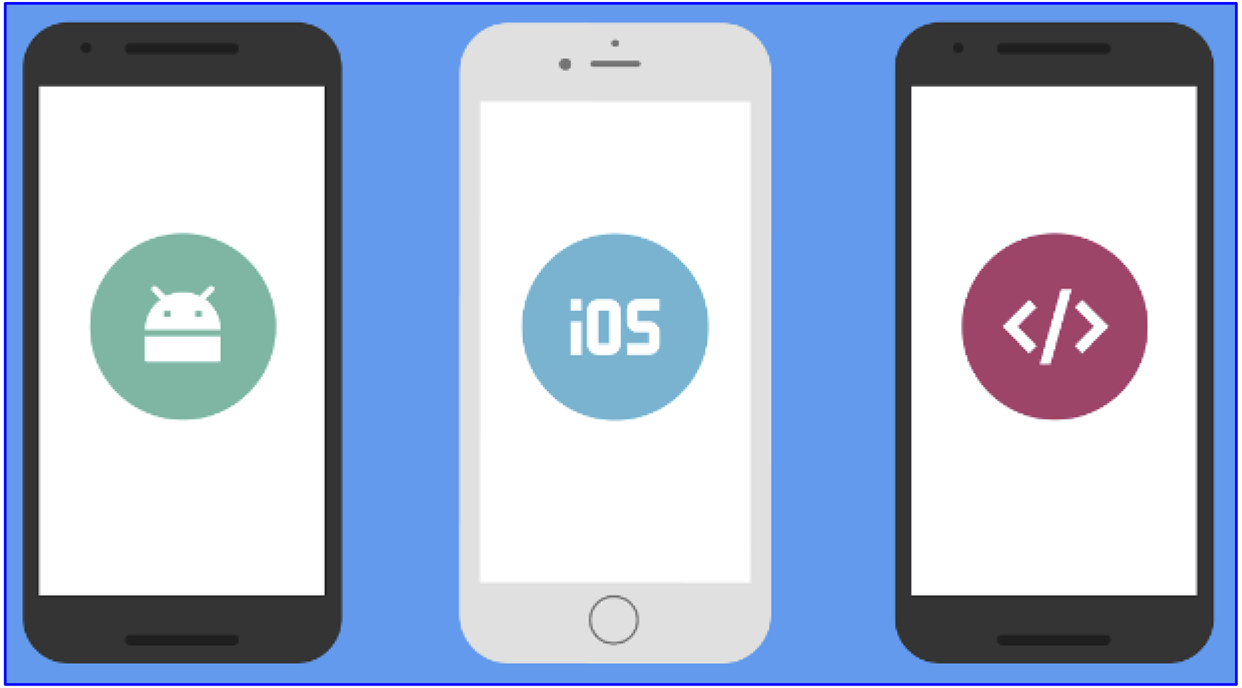
Firebase is a platform that provides tools to help you

- develop your app
- grow your user base
- earn money from your app

Tools for all platforms

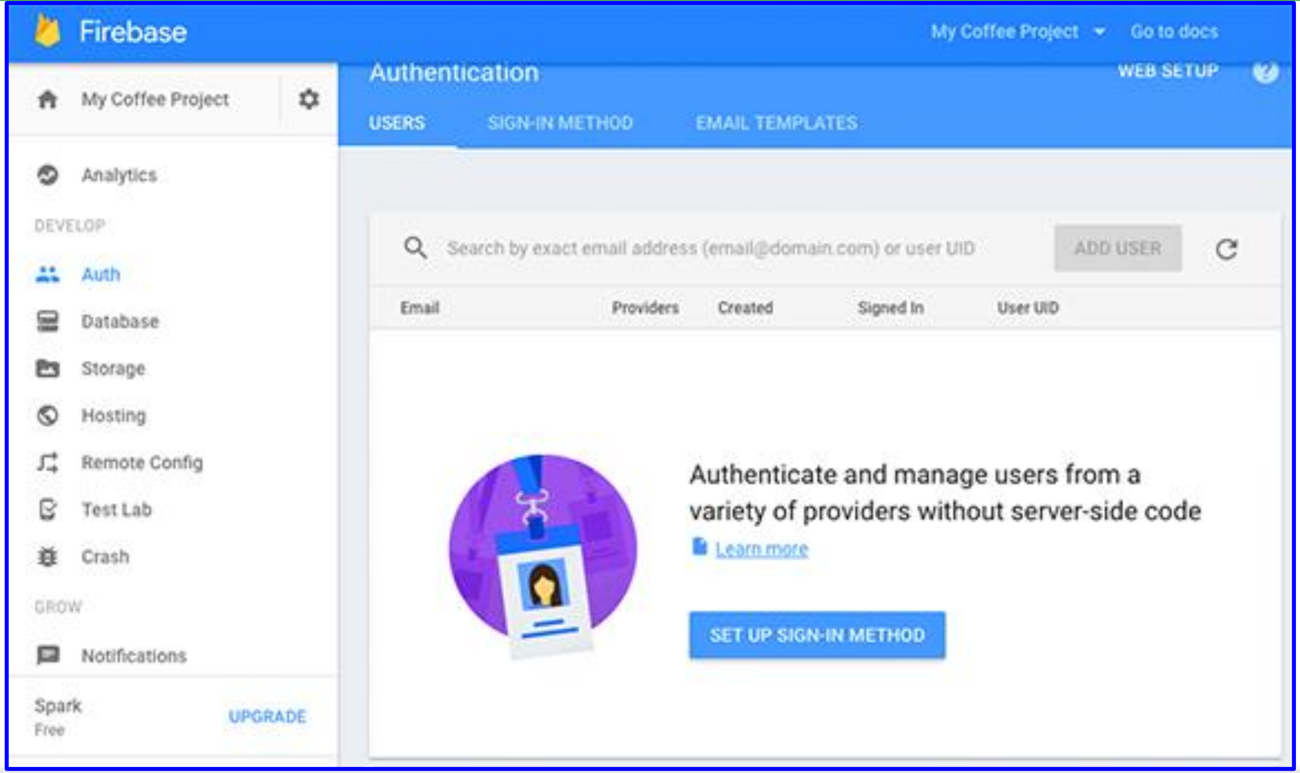
Firebase features are available for

- Android
- iOS
- Web



Firebase console

Firebase Console is a web front end for managing your Firebase projects



Authenticate and manage users from a variety of providers without server-side code

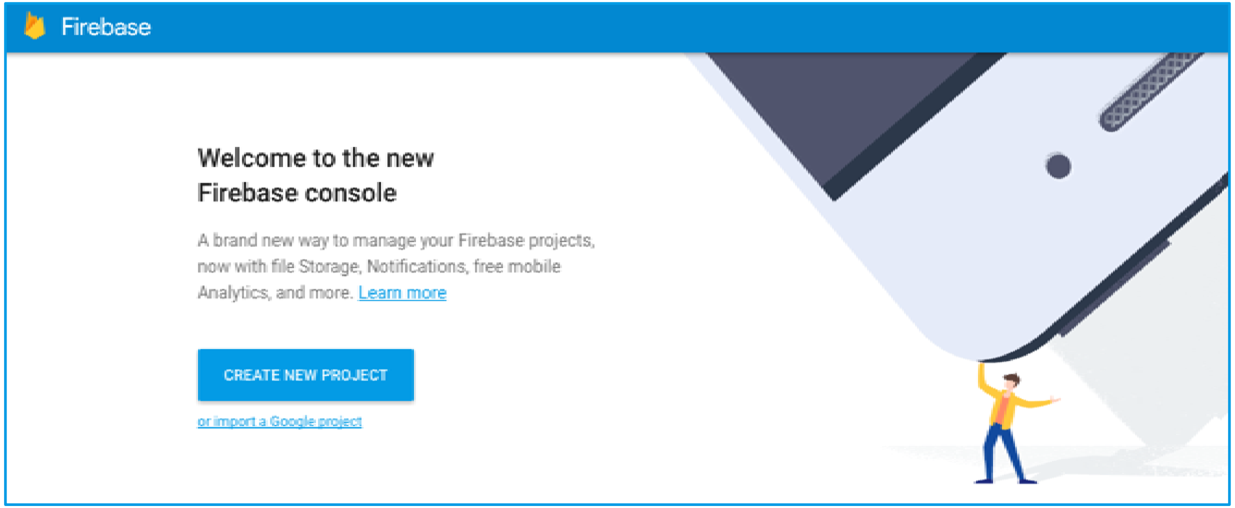
Using Firebase

1. Connect your app to your Firebase project
2. Enable Firebase features in the console
3. Add code to your app (where needed)

Get started with Firebase

Firebase console

firebase.google.com

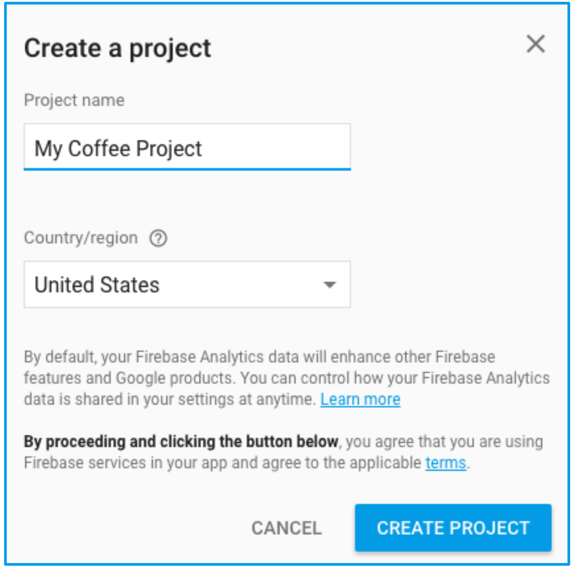


Firebase

Welcome to the new Firebase console

A brand new way to manage your Firebase projects, now with file Storage, Notifications, free mobile Analytics, and more. Learn more

Create new project



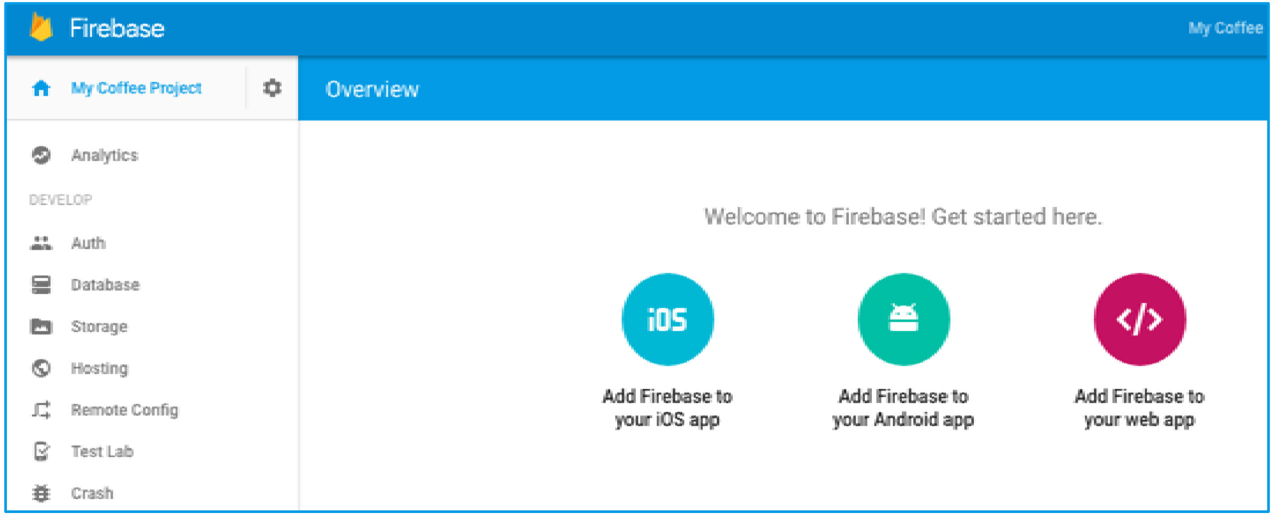
Create a project

Project name: My Coffee Project

Country/region: United States

By default, your Firebase Analytics data will enhance other Firebase features and Google products. You can control how your Firebase Analytics data is shared in your settings at anytime. Learn more

By proceeding and clicking the button below, you agree that you are using Firebase services in your app and agree to the applicable terms.

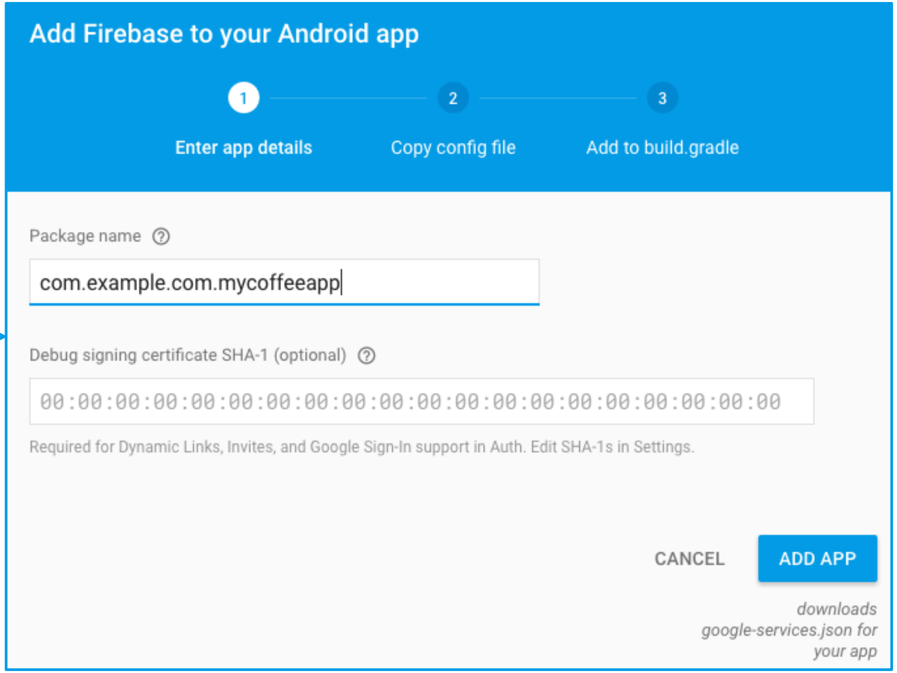


Welcome to Firebase! Get started here.

Connect your app to the Firebase project

Add your Android app to your Firebase project

Add Firebase to your Android app



Add Firebase to your Android app

Package name

com.example.com.mycoffeeapp

Debug signing certificate SHA-1 (optional)

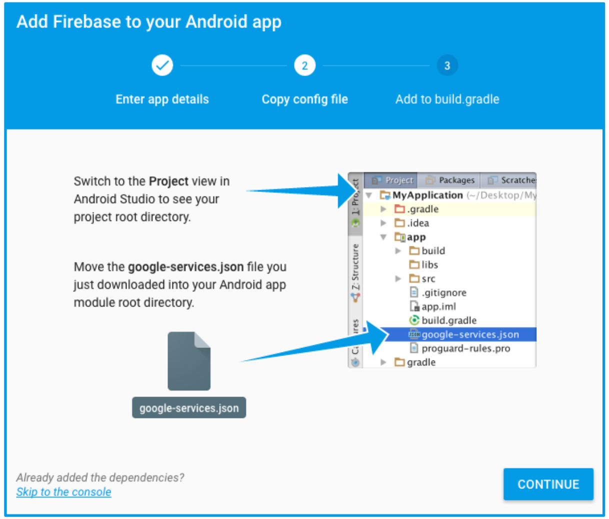
00:00:00:00:00:00:00:00:00:00:00:00:00:00:00:00:00:00:00:00

Required for Dynamic Links, Invites, and Google Sign-In support in Auth. Edit SHA-1s in Settings.

downloads google-services.json for your app

Firebase config file

- Firebase creates a config file for your app
- It contains all the information your app needs to integrate with the Firebase project
- google-services.json file



Add Firebase to your Android app

✓ Enter app details — 2 Copy config file — 3 Add to build.gradle

Switch to the **Project** view in Android Studio to see your project root directory.

Move the **google-services.json** file you just downloaded into your Android app module root directory.

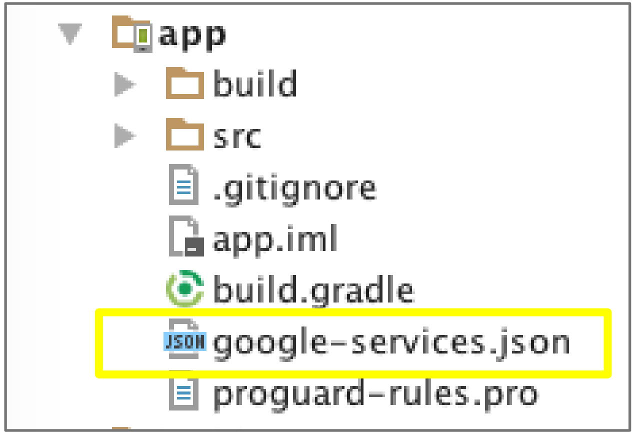
google-services.json

Already added the dependencies?
Skip to the console

CONTINUE

Add config file to your project

- Open your app in Android Studio
- Drag and drop google-services.json to the app module



Update code dependencies in your app

- The Google services plugin for Gradle in your project loads google-services.json
- Modify your build.gradle files to use the plugin

Follow the instructions in the Firebase wizard to update code dependencies

Update build.gradle

- Project-level build.gradle:

```
buildscript {
  dependencies {
    // Add this line
    classpath 'com.google.gms:google-services:n.n.n'
```

Get exact number from Firebase wizard

- App-level build.gradle...

```
  // Add to the bottom of the file
  apply plugin: 'com.google.gms.google-services'
```

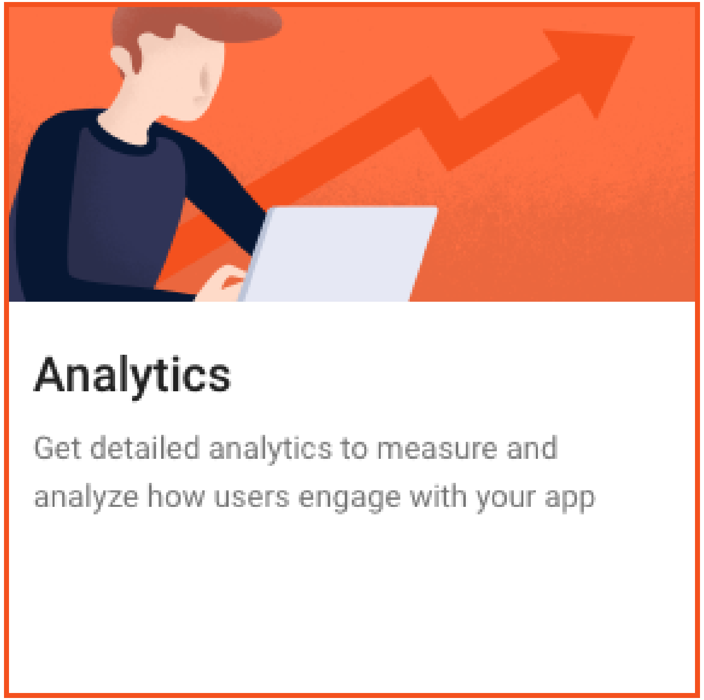
Firebase Analytics

Firebase Analytics

- Unlimited free reporting
- Audience segmentation

  Define custom audiences based on device data, custom events, or user properties

firebase.google.com/docs/analytics/



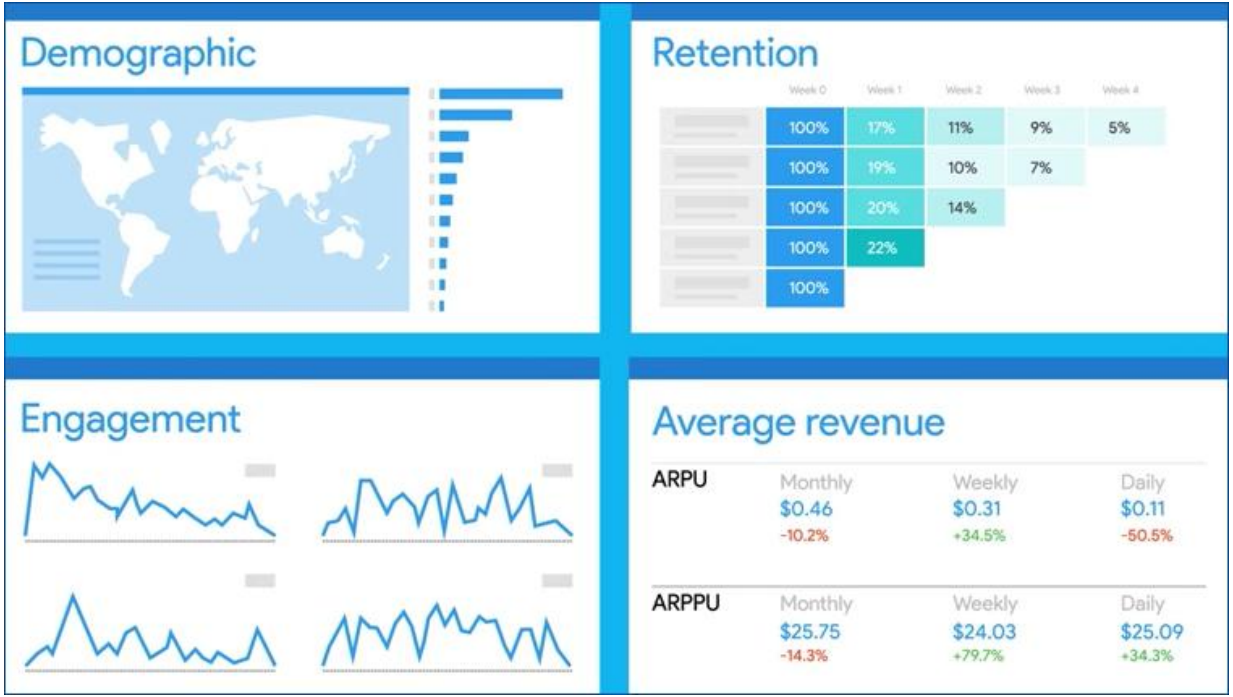
Analytics

Get detailed analytics to measure and analyze how users engage with your app

Firebase Analytics: default reports

Get reports without adding code

- geographic
- demographic
- engagement
- revenue



Retention

	Week 0	Week 1	Week 2	Week 3	Week 4
	100%	17%	11%	9%	5%
	100%	19%	10%	7%	
	100%	20%	14%		
	100%	22%			
	100%				

Average revenue

	Monthly	Weekly	Daily
ARPU	$0.46 -10.2%	$0.31 +34.5%	$0.11 -50.5%
ARPPU	$25.75 -14.3%	$24.03 +79.7%	$25.09 +34.3%

Firebase Analytics: custom reports

Add code to log events in your app to get more reports

- Predefined events such as: user adds an item to their cart
- Custom events such as: user achieves a level in a game

In your app

- In your app project, add dependency in app/build.gradle:

```
compile 'com.google.firebase:firebase-analytics:n.n.n'
```

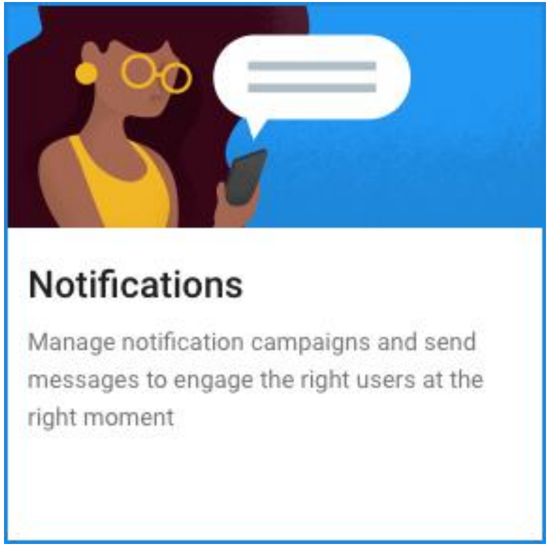
- No need to write code for default reports
- Write code for custom events if you want them

Firebase Notifications

Firebase notifications

In the Notifications lesson, you learned how ***your app*** can send notifications to the user.

The Firebase Console lets ***you*** send notifications to your users.



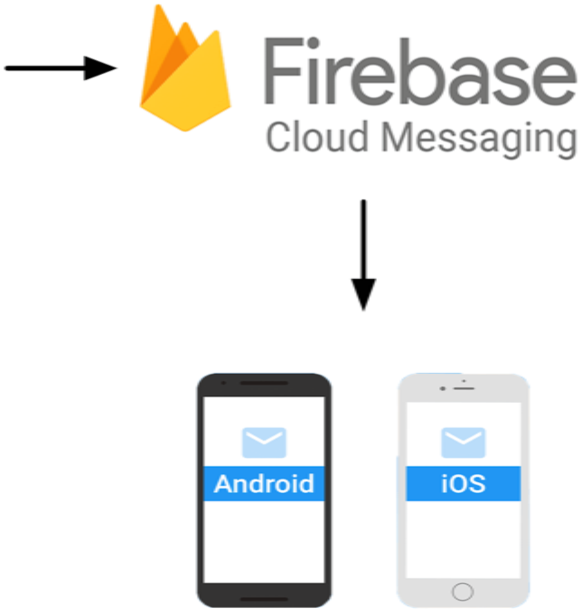
Notifications

Manage notification campaigns and send messages to engage the right users at the right moment

Firebase Cloud Messaging sends the msg

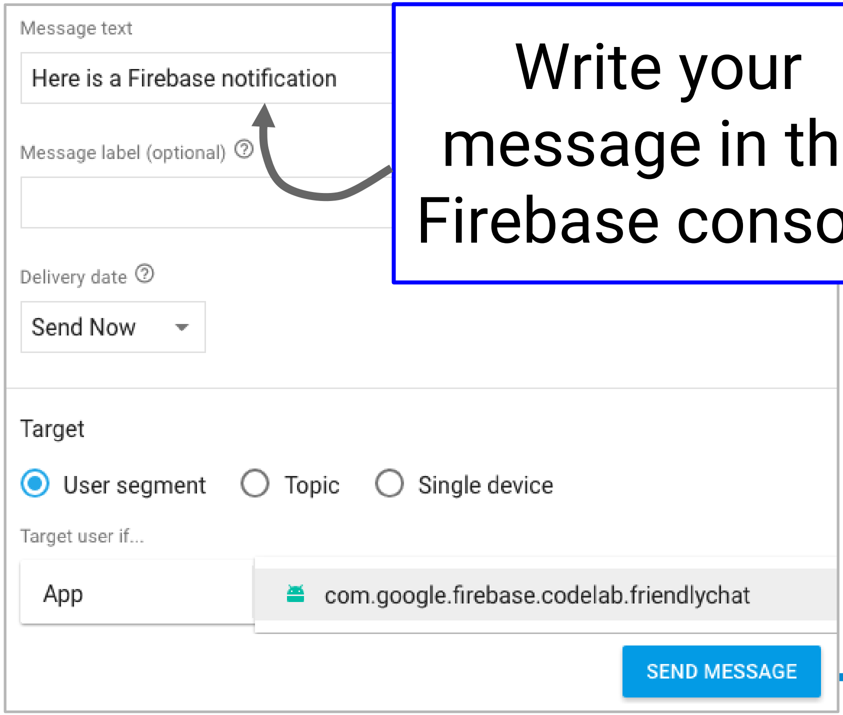
- You write notification messages in the Firebase Console

Notifications Console GUI



- Firebase Cloud Messaging delivers the notifications to the target audience

Send message



Message text

Here is a Firebase notification

Message label (optional)

Delivery date

Send Now

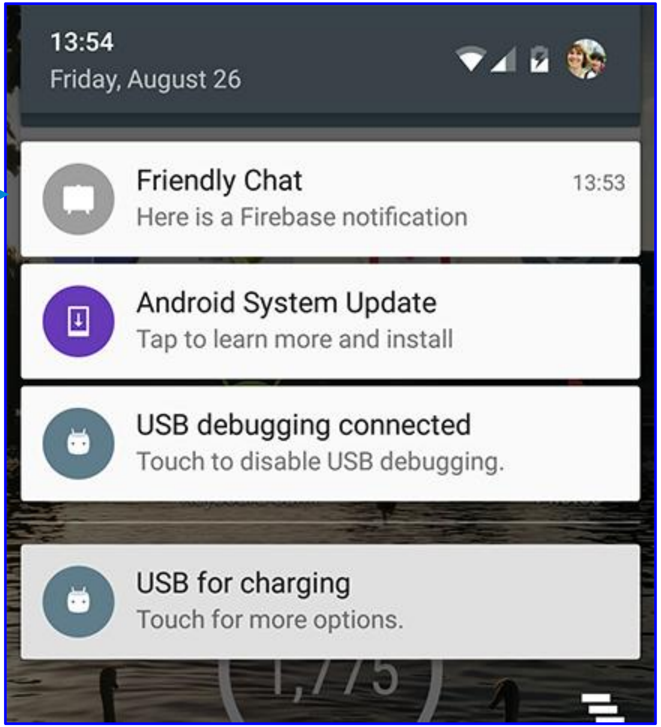
Target

● User segment ○ Topic ○ Single device

Target user if...

App	com.google.firebase.codelab.friendlychat

Write your message in the Firebase console



13:54
Friday, August 26

Friendly Chat 13:53
Here is a Firebase notification

Android System Update
Tap to learn more and install

USB debugging connected
Touch to disable USB debugging.

USB for charging
Touch for more options.

Firebase Database

Storing and sharing data

You have already learned your app can

- save data in an SQLite database on the device
- use a ContentProvider to share data with other apps

How do you enable different users using different devices, to share and update data?

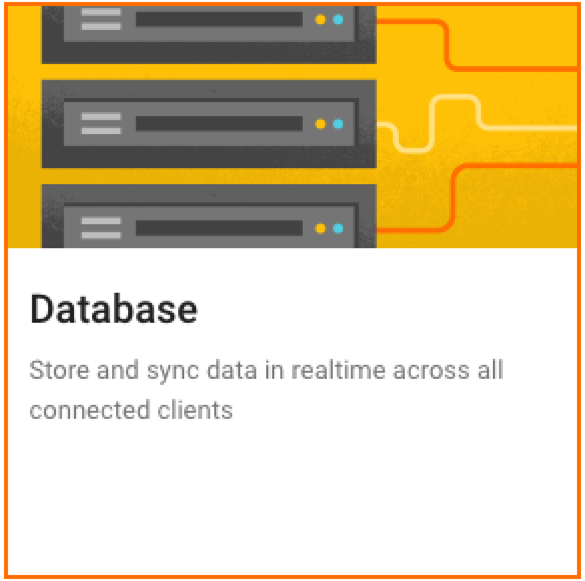
- Use Firebase Database

Firebase Realtime database

Store and sync data with the Firebase cloud database

Data is synced across all clients, and remains available when your app goes offline

firebase.google.com/docs/database/



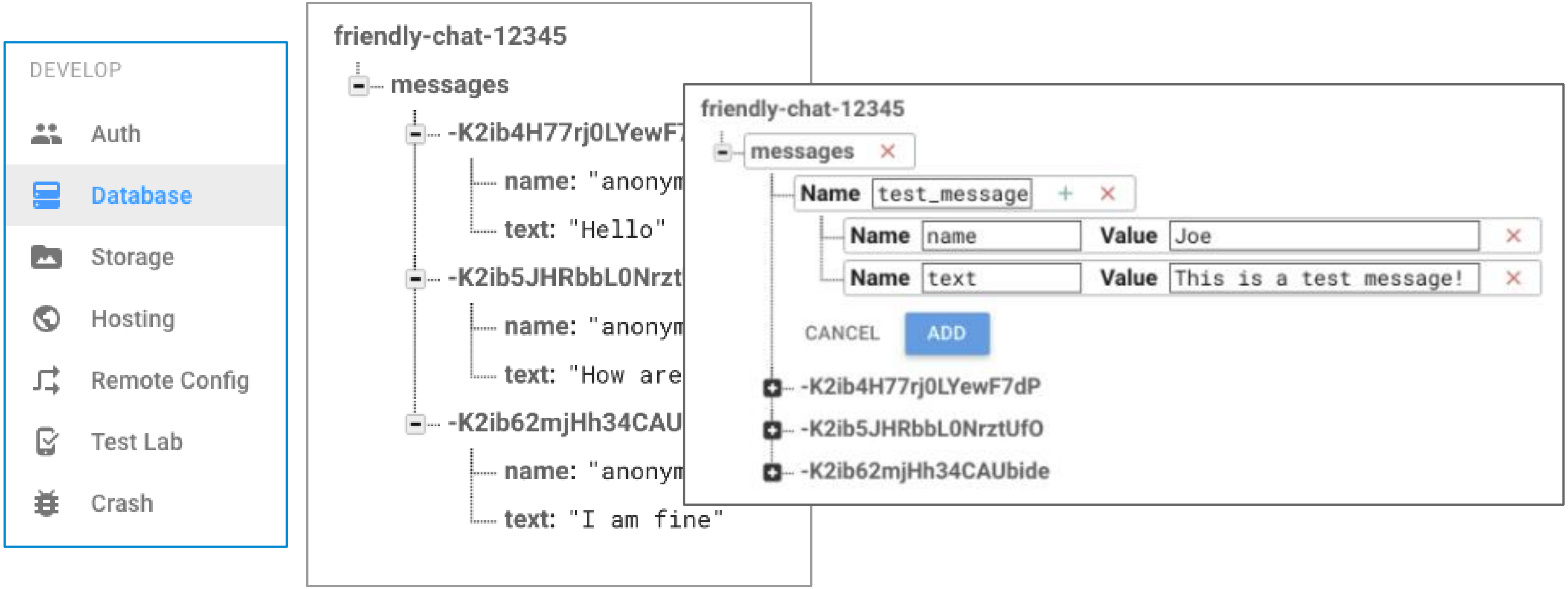
Database

Store and sync data in realtime across all connected clients

Connected apps share data

- Firebase Realtime Database is hosted in the cloud
- Data is stored as JSON
- Data is synchronized in realtime to every connected client.

View and edit data in Firebase Console



```
friendly-chat-12345
  messages
    -K2ib4H77rj0LYewF7
      name: "anonym
      text: "Hello"
    -K2ib5JHRbbL0Nrzt
      name: "anonym
      text: "How are
    -K2ib62mjHh34CAU
      name: "anonym
      text: "I am fine"
```

```
friendly-chat-12345
  messages
    Name test_message
      Name name    Value Joe
      Name text    Value This is a test message!
    CANCEL  ADD
    -K2ib4H77rj0LYewF7dP
    -K2ib5JHRbbL0NrztUfO
    -K2ib62mjHh34CAUbide
```

In your app

- In your app project, add dependency in app/build.gradle:

  ```
  compile 'com.google.firebase:firebase-database:n.n.n'
  ```

- In your app source code, put data in the database, and get data from the database (API Reference)

Example:

```
FirebaseDatabase database = FirebaseDatabase.getInstance();
DatabaseReference myRef = database.getReference(path);
myRef.setValue("New value");
```

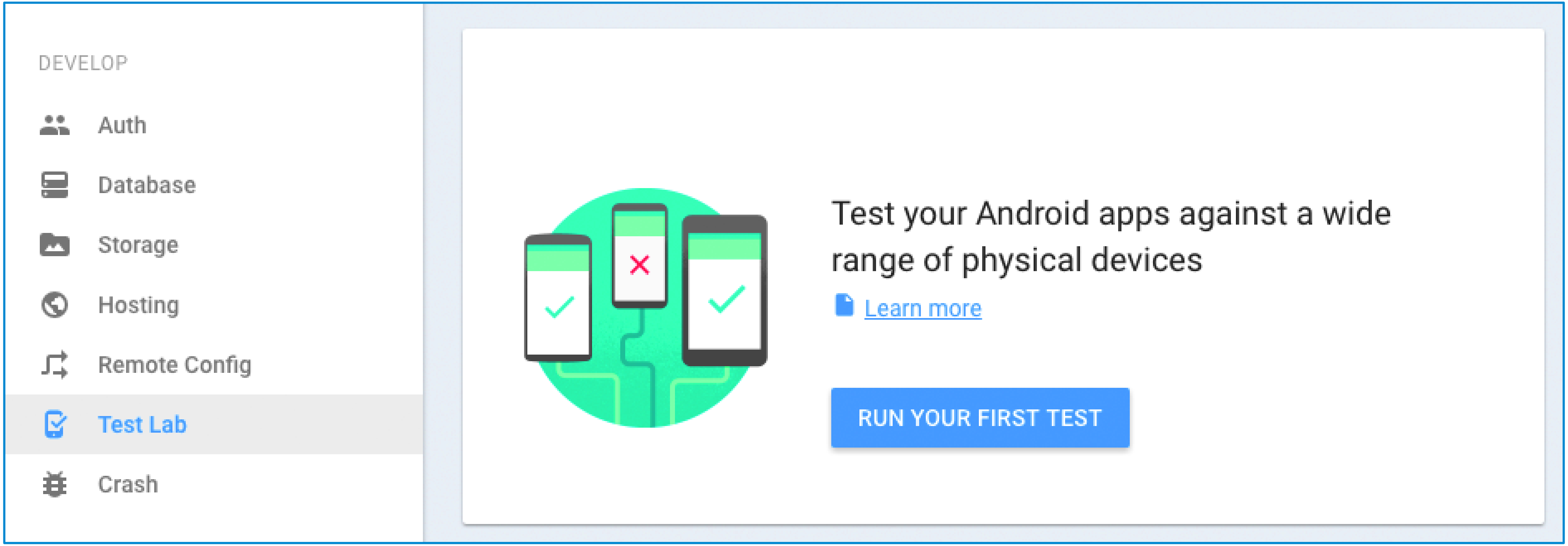
Firebase Cloud Test Lab

Firebase Cloud Test Lab

Test your app on real devices in a Google data center

firebase.google.com/docs/test-lab/

Run your tests in the console



DEVELOP

- Auth
- Database
- Storage
- Hosting
- Remote Config
- Test Lab
- Crash

Test your Android apps against a wide range of physical devices

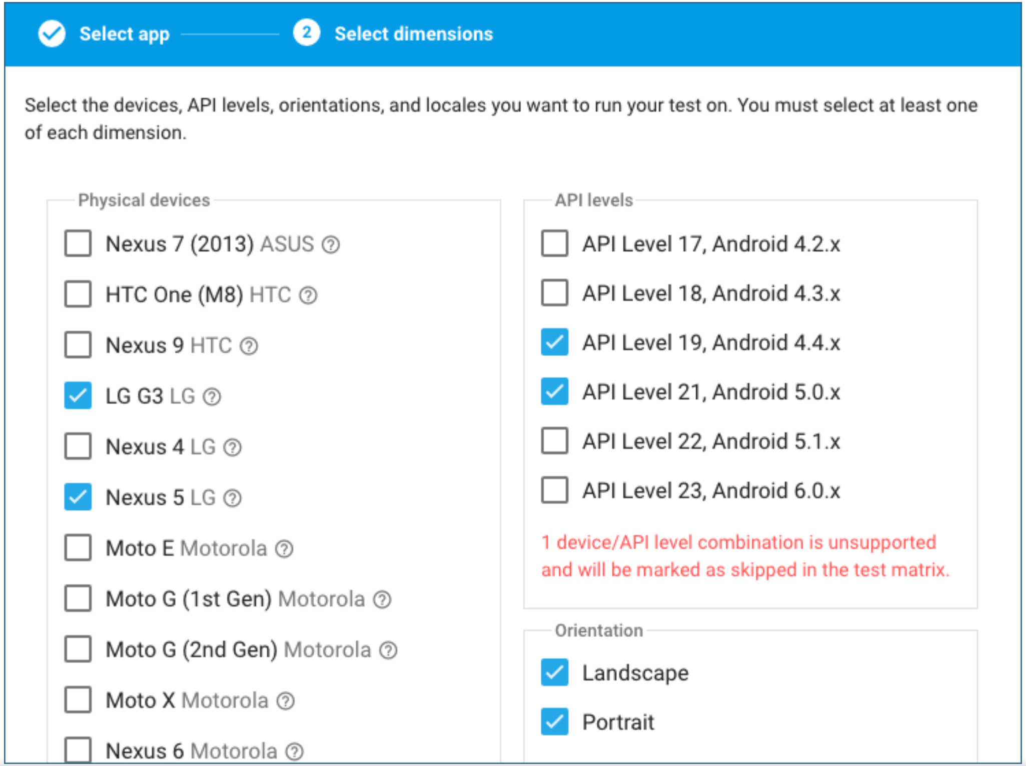
Learn more

RUN YOUR FIRST TEST

Choose the devices to test your app on

Set test targets:

- Devices
- API levels
- Orientations
- Locales



Select the devices, API levels, orientations, and locales you want to run your test on. You must select at least one of each dimension.

Physical devices

- □ Nexus 7 (2013) ASUS ⓘ
- □ HTC One (M8) HTC ⓘ
- □ Nexus 9 HTC ⓘ
- ☑ LG G3 LG ⓘ
- □ Nexus 4 LG ⓘ
- ☑ Nexus 5 LG ⓘ
- □ Moto E Motorola ⓘ
- □ Moto G (1st Gen) Motorola ⓘ
- □ Moto G (2nd Gen) Motorola ⓘ
- □ Moto X Motorola ⓘ
- □ Nexus 6 Motorola ⓘ

API levels

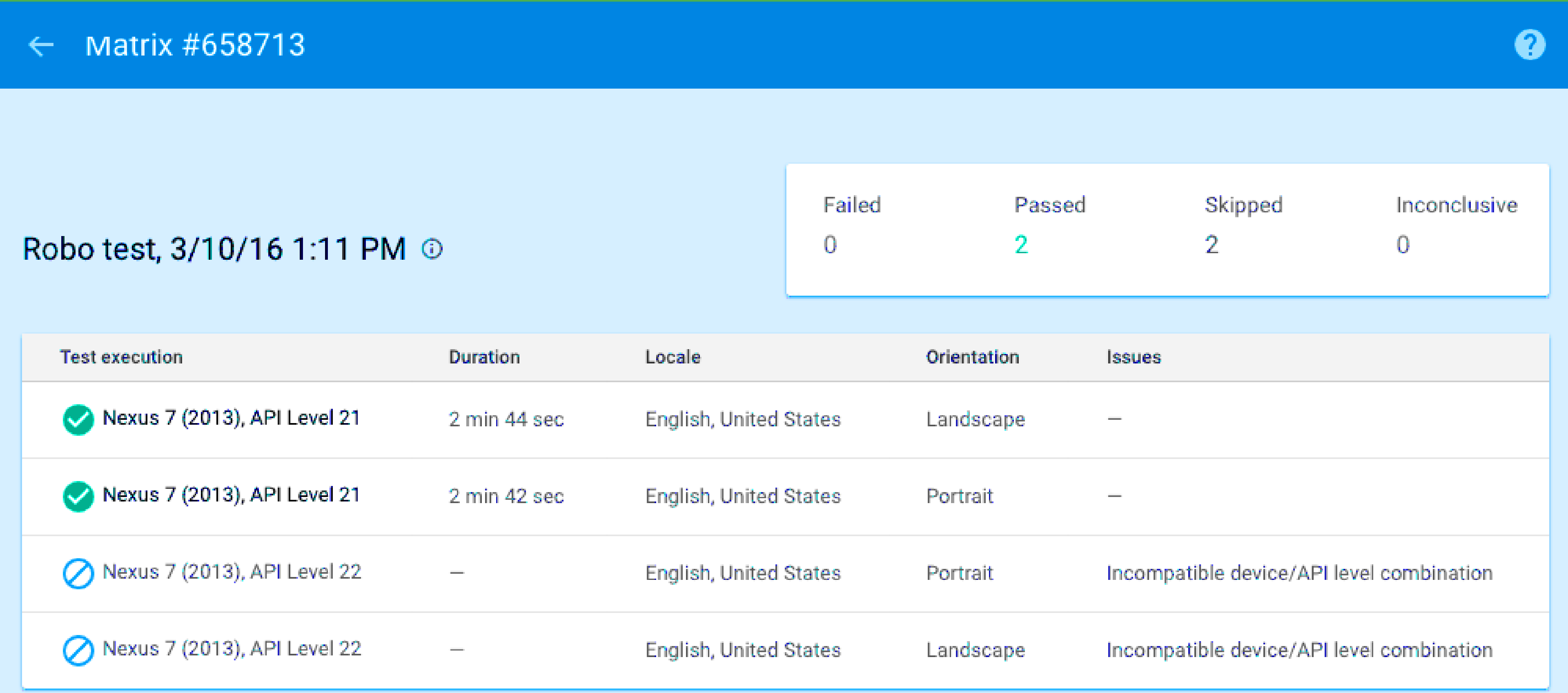
- □ API Level 17, Android 4.2.x
- □ API Level 18, Android 4.3.x
- ☑ API Level 19, Android 4.4.x
- ☑ API Level 21, Android 5.0.x
- □ API Level 22, Android 5.1.x
- □ API Level 23, Android 6.0.x

1 device/API level combination is unsupported and will be marked as skipped in the test matrix.

Orientation

- ☑ Landscape
- ☑ Portrait

Get test results



← Matrix #658713

Robo test, 3/10/16 1:11 PM ⓘ

Failed	Passed	Skipped	Inconclusive
0	2	2	0

Test execution	Duration	Locale	Orientation	Issues
✓ Nexus 7 (2013), API Level 21	2 min 44 sec	English, United States	Landscape	—
✓ Nexus 7 (2013), API Level 21	2 min 42 sec	English, United States	Portrait	—
⊘ Nexus 7 (2013), API Level 22	—	English, United States	Portrait	Incompatible device/API level combination
⊘ Nexus 7 (2013), API Level 22	—	English, United States	Landscape	Incompatible device/API level combination

And more...

More Firebase tools

- Firebase storage —Store images, audio, video, or other user-generated content.
- Store terabytes of data!
- Authentication—Enable users to sign in to your app
- Get crash reports

… and even more!

- App indexing -- enable Google search to include results from your app
- Dynamic links -- deep links into an app that work whether or not users have installed the app yet.
- App Invites -- allow users to invite others to your app
- AdMob -- we'll talk about that next

Learn more about Firebase

- Firebase
- Firebase Testing Lab
- Getting started with Firebase for Android
- Firebase in a weekend online course
  www.udacity.com/course/ud0352

Learn more contd...

- Firebase console
  console.firebase.google.com/
- Firebase developer documentation
  firebase.google.com/docs/
- Firebase codelab

  codelabs.developers.google.com/codelabs/firebase-android

Highly recommended!

Video Resources

- Introducing Firebase:
  https://www.youtube.com/watch?v=O17OWyx08Cg
- Playlist of intro videos to Firebase features
  http://goo.gl/qo4Frq

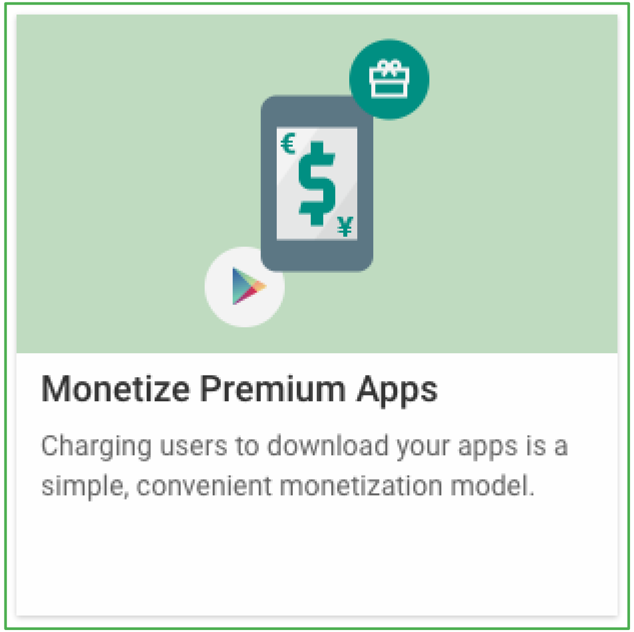
Make Money from your apps

Ways to monetize apps in Google Play

- **Premium** model—users pay to download app
- **Freemium** model
  - downloading app is free
  - users pay for upgrades or in-app purchases
- **Subscriptions**—users pay recurring fee for app
- **Ads**—app is free but displays ads

Premium apps

- Charge users to download your app
- Set prices in the Developer Console
- Good model for apps that address a market niche
- BUT users often won't download an app if they have to pay for it



Monetize Premium Apps

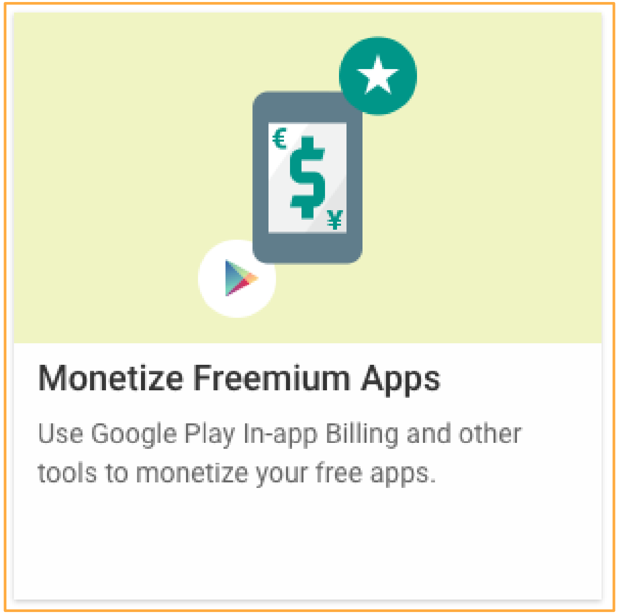
Charging users to download your apps is a simple, convenient monetization model.

Freemium Apps

Offer a free download with

- limited features
- full features for a limited time

Let users upgrade to full, unlimited app with an in-app purchase



Monetize Freemium Apps

Use Google Play In-app Billing and other tools to monetize your free apps.

developer.android.com/google/play/billing/index.html

In-app purchases

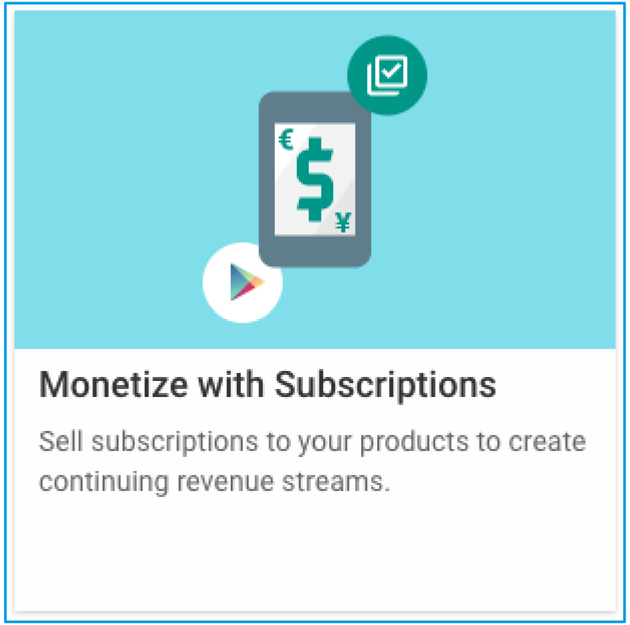
Use In-app purchases to sell extra features

- New features
- Additional content
- Skins
- New levels, powers, attacks...

developer.android.com/distribute/monetize/freemium.html

Subscriptions

- Subscriptions let users use apps or features for a recurring monthly or annual fee
- Offer a free trial subscription to allow users to explore your app

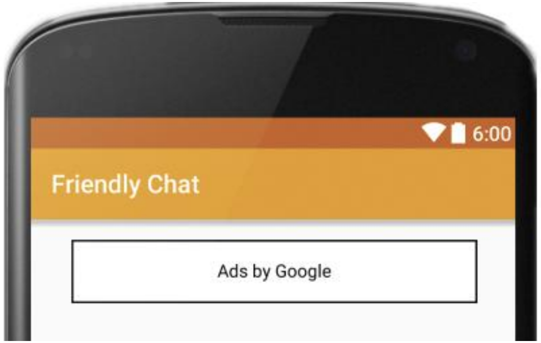


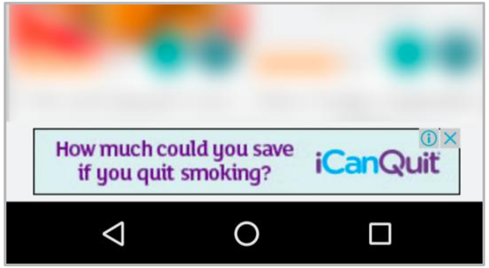
Monetize with Subscriptions

Sell subscriptions to your products to create continuing revenue streams.

Run ads using AdMob

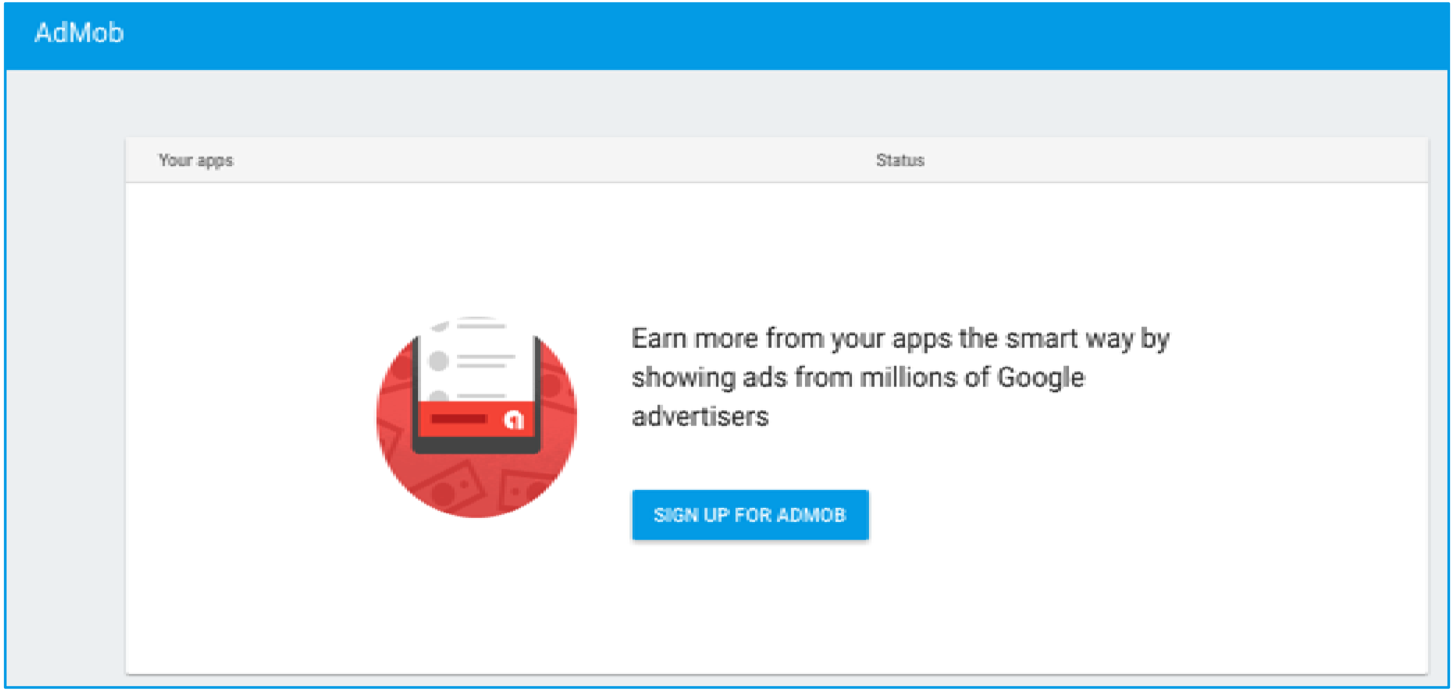
- Runs ads in your app to earn revenue
- 650,000 + apps use AdMob
- $1 billion+ paid to developers in the last 2 years




How much could you save if you quit smoking?

developer.android.com/distribute/monetize/ads.html

Sign up for AdMob in Firebase Console



AdMob

Your apps

Status

Earn more from your apps the smart way by showing ads from millions of Google advertisers

Learn more about monetizing your app

- Earn
- Monetizing with Ads
- In-App Purchases
- In-App Billing

What's next?

This course does not have a practical for Firebase and AdMob

Get hands on experience by taking these online courses:

- Firebase in a Weekend online course
  www.udacity.com/course/ud0352

- Firebase codelab
  codelabs.developers.google.com/codelabs/firebase-android

What's next?

- Concept Chapter: 14.1 C Firebase and monetization
- This lesson does not have a practical

Get practical experience by taking these online courses:

  - Firebase in a Weekend
    www.udacity.com/course/ud0352
  - Firebase codelab
    codelabs.developers.google.com/codelabs/firebase-android

END

Android Developer Fundamentals

What's next?

Lesson 15

This work is licensed under a Creative Commons Attribution-NonCommercial 4.0 International License

15.1 Prepare and Publish Your App!

Contents

- Prepare your app for release
- Publish!

Steps for publishing your app

- Prepare app for release
- Generate signed APK
- Upload to Google Play
- Run alpha and beta tests
- Publish to the world

What is an APK?

- **A**ndroid **A**pplication **P**ackage file → .apk file
- It's like the executable
- Each Android application is compiled and packaged in a single file that includes all the app's code, resources, assets, and manifest file
- You need an APK to publish on Google Play

Share your app during development

Ways to distribute your app

- Zip it up
- Share the source code
- Publish to github
- Make an APK

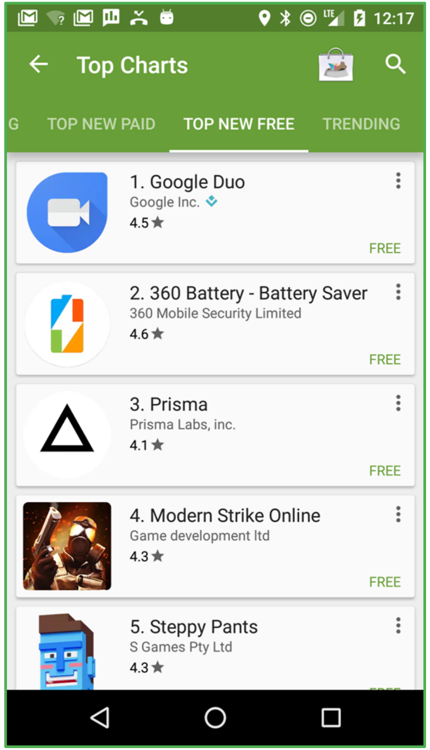
Share with friends and family

Use alpha and beta tests to share your app with friends and family

Share your production app

Publish on Google Play

- Make an APK
- Upload to Google Play
- Run alpha and beta tests
- Publish!



1. Google Duo
Google Inc.
4.5 ★
FREE

2. 360 Battery - Battery Saver
360 Mobile Security Limited
4.6 ★
FREE

3. Prisma
Prisma Labs, inc.
4.1 ★
FREE

4. Modern Strike Online
Game development ltd
4.3 ★
FREE

5. Steppy Pants
S Games Pty Ltd
4.3 ★

Prepare Your App for Release

Prepare your app for release

- Test, test, test!
- Add an icon
- Make sure your app has the correct filters
- Choose an Application ID
- Specify API levels targets
- Clean up your app
- Generate a signed APK for release

developer.android.com/studio/publish/preparing.html

Test, test, test!

First, make sure your app works correctly...

- Test your app on different devices and screen size
- Test your app on older devices

  Use support library for backwards compatibility
- Test, test, test!

Test on devices in a data center

Test on real devices in a data center using the

Firebase Cloud Test Lab

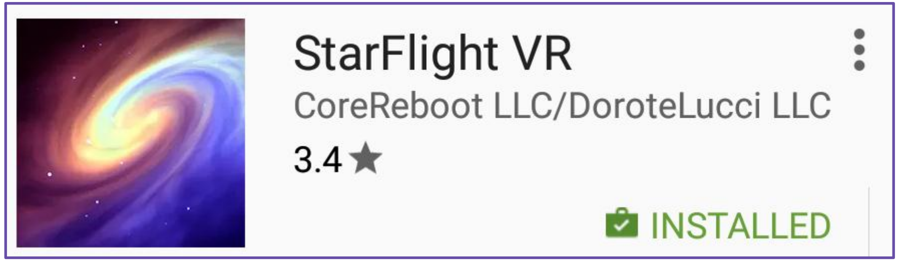
Add launcher app icon

The launcher icon appears in

- Google Play

On the device

- Home screen
- Manage Applications
- My Downloads

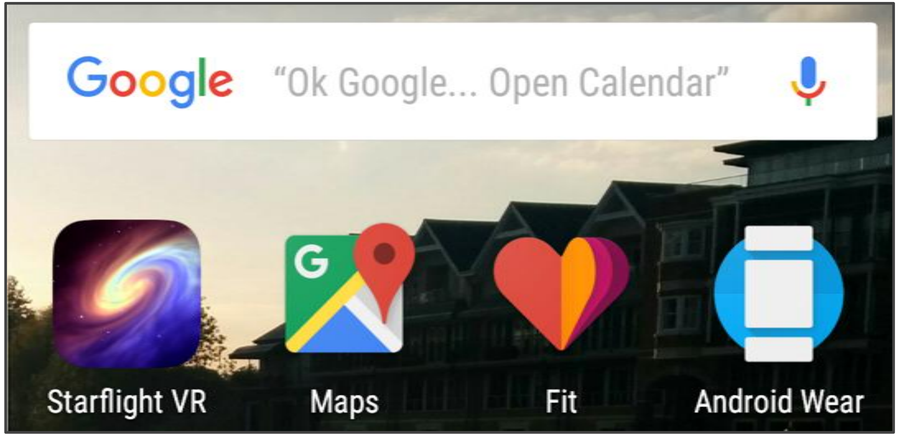


StarFlight VR

CoreReboot LLC/DoroteLucci LLC

3.4 ★

INSTALLED



Google Play filters search results

Google Play search results only show apps that are compatible with the user's device.

If an app uses a camera, Google Play only shows the apps to devices that have a camera.

developer.android.com/google/play/filters.html

Types of filters

- Hardware features
- API level
- Manifest settings such as
  - <supports-screens>
  - <uses-feature>
- Countries (selected during APK upload)

developer.android.com/google/play/filters.html

Choose your Application ID

- Application ID defines your application's identity
- Must be unique across all apps from everyone!
- If you change App ID and re-publish
  - The app becomes a different application
  - Users of the previous app cannot update to the new app

Application ID versus package name

- Initial Application ID is set to the package
- You can change Application ID in build.gradle independently of package name

tools.android.com/tech-docs/new-build-system/applicationid-vs-packagename

Specify Application ID in build.gradle

```
android {
  compileSdkVersion 23
  buildToolsVersion "23.0.3"
  defaultConfig {
    applicationId "android.mydomain.com.myappid"
    ...
  }
  ...
}
```

Initial Application ID is same as package in Android manifest

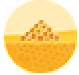
Set min and target API level

- **minSdkVersion** — minimum version of the Android platform that the app runs on

- **targetSdkVersion** — API level that the app is designed for

API levels: developer.android.com/guide/topics/manifest/uses-sdk-element.html#ApiLevels

Android versions

Codename		Version	Release Date	API Level
Honeycomb		3.0 - 3.2.6	Feb 2011	11 - 13
Ice Cream Sandwich		4.0 - 4.0.4	Oct 2011	14 - 15
Jelly Bean		4.1 - 4.3.1	July 2012	16 - 18
KitKat		4.4 - 4.4.4	Oct 2013	19 - 20
Lollipop		5.0 - 5.1.1	Nov 2014	21 - 22
Marshmallow		6.0 - 6.0.1	Oct 2015	23
Nougat		7.0	Sept 2016	24

There were earlier versions before Feb 2011.

Clean up your app

- Remove logging statements
- Disable debugging
- Clean up your project directories
- Update URLs for servers and services
- Reduce APK size

Why does APK size matter?

The size of your APK affects:

- how fast your app loads
- how much memory it uses
- how much power it consumes

developer.android.com/topic/performance/reduce-apk-size.html

Why reduce APK size?

Users might abandon "large" apps, particularly:

- in areas with unreliable 2G and 3G networks
- on devices that work on pay-by-the-byte plans

developer.android.com/topic/performance/reduce-apk-size.html

Reducing APK size

- Remove unused resources
- Reuse resources
- Minimize resource use from libraries
- Reduce native and Java code
- Reduce space needs for images

developer.android.com/topic/performance/reduce-apk-size.html

Reduce image sizes

- Reduce animation frames
- Use Drawable objects
- Crunch PNG files
- Use lowest quality and size JPEG files that look good
- Use WebP File Format
- Use vector graphics

developer.android.com/topic/performance/reduce-apk-size.html

Clean up project folders

- Clean up project folders and files
- Stray or orphaned files can prevent your application from compiling and cause it to behave unpredictably

developer.android.com/studio/projects/index.html#ApplicationProjects

Clean up jni, lib, and src folders

- **src**: source files for your app (.java and .aidl files)
  NO jar files
- **lib**: third-party or private library files, including prebuilt shared and static libraries (such as .so files)
- **jni**: source files associated with the Android NDK, such as .c, .cpp, .h, and .mk files

Clean up assets, resources, and tests

- Remove unused private or proprietary data files

  For example, delete unused drawable files, layout files, and values files from res/ folder

- Review the assets and res/raw directories for raw asset files and static files to update or remove

- Remove unused test directories

Generate APK

Generate signed APK for release

- Android apps must be digitally signed before users can install them
- Use Android Studio to generate and sign your APK

Digital certificates

A public-key certificate contains:

- the public key of a public/private key pair,
- other metadata identifying the owner of the key such as name and location

The owner of the certificate holds the private key.

Public and private key for app

- When Android Studio signs the app, it creates the public certificate and the private key.
- It attaches the public certificate to the APK.
- You must securely store the private key in a keystore

Why you need to sign your app

The public-key certificate serves as as a "fingerprint" that uniquely associates the APK to you and your corresponding private key.

This helps Android ensure that any future updates to your APK are authentic and come from the original author.

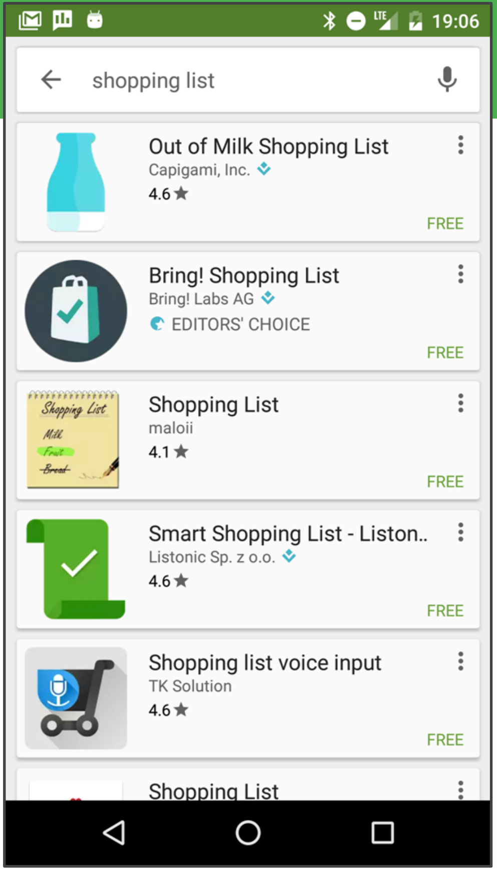
Publish to Google Play

Publish your app!

Publish your Android apps on Google Play

Users can

- search
- download
- review



19:06

shopping list

Out of Milk Shopping List
Capigami, Inc.
4.6 ★
FREE

Bring! Shopping List
Bring! Labs AG
EDITORS' CHOICE
FREE

Shopping List
maloii
4.1 ★
FREE

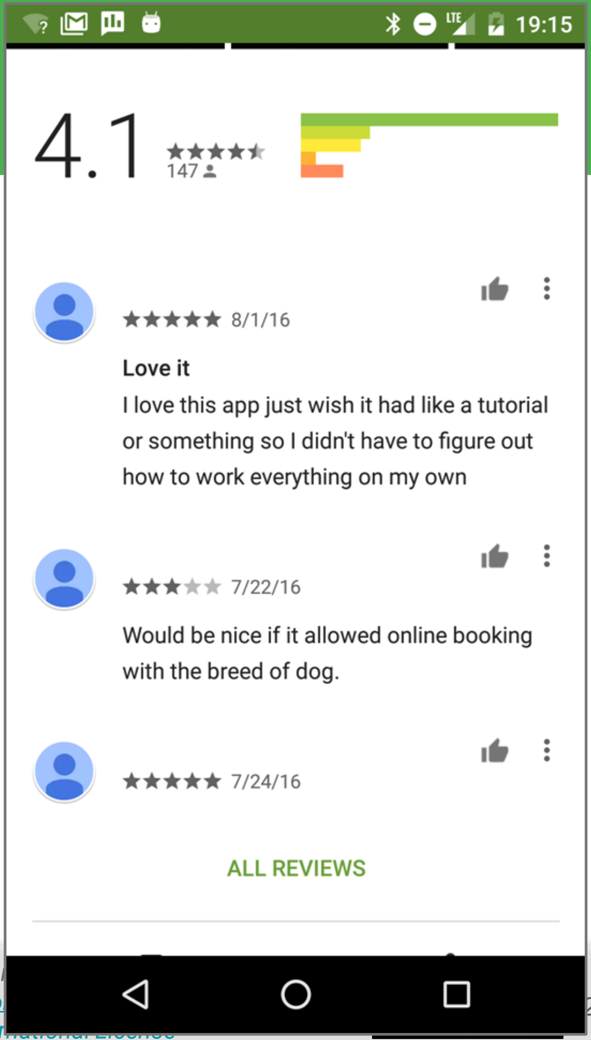
Smart Shopping List - Liston..
Listonic Sp. z o.o.
4.6 ★
FREE

Shopping list voice input
TK Solution
4.6 ★
FREE

Shopping List

Users can review!

- Make sure your app is ready before publishing to the world!
- People will give you bad reviews if they are unhappy with it
- The more they like it, the more they will want more features!



4.1 ★★★★★ 147

★★★★★ 8/1/16

Love it

I love this app just wish it had like a tutorial or something so I didn't have to figure out how to work everything on my own

★★★★★ 7/22/16

Would be nice if it allowed online booking with the breed of dog.

★★★★★ 7/24/16

Steps to publish

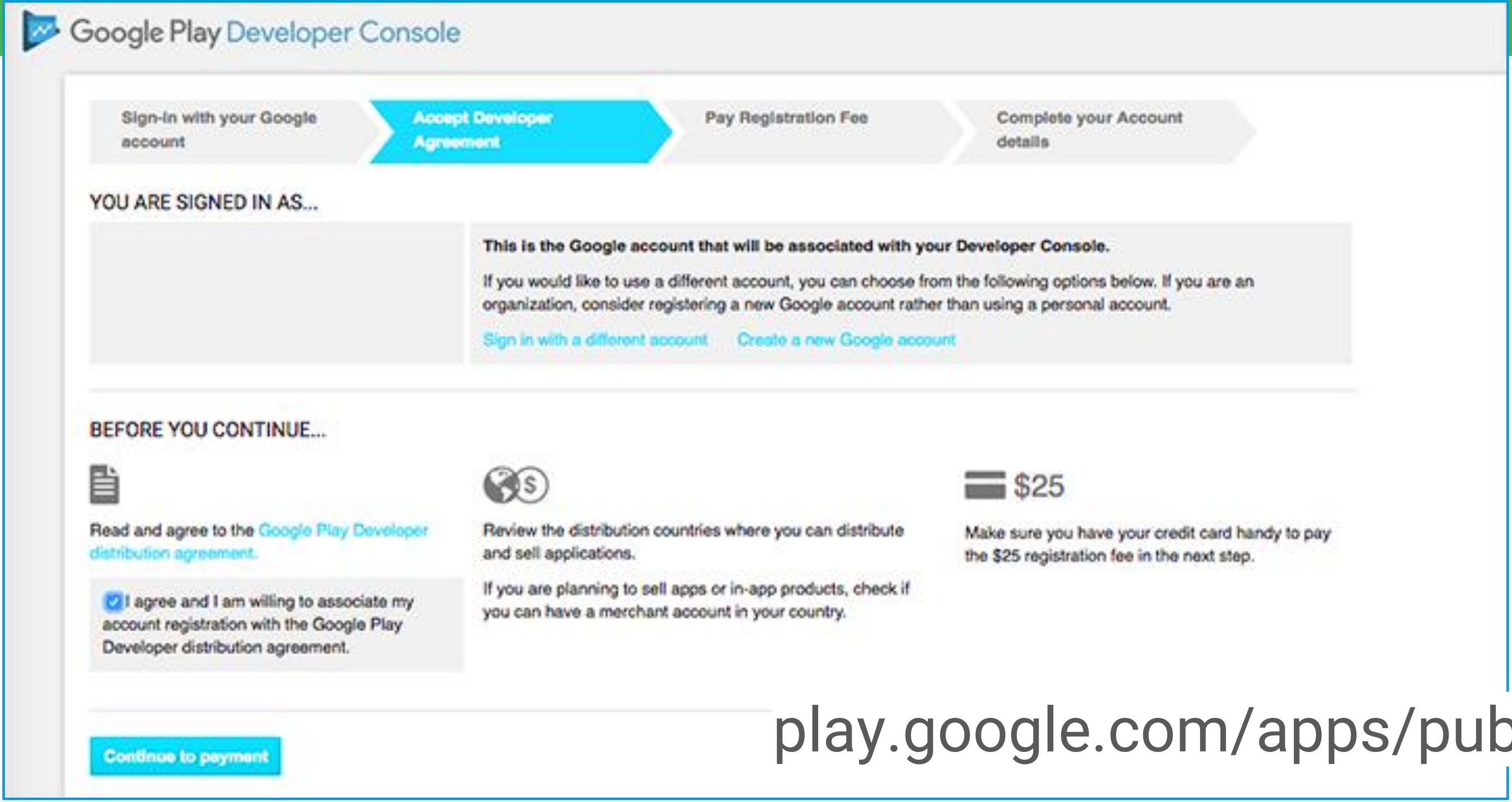
- Create an account on Google Play developer console
- Create an entry for your app
- Add the required assets and information
- Run alpha and beta tests
- Run pre-launch reports
- Publish to the world!

Google Play Developer Console

Sign up for Google Play

1. Go to play.google.com/apps/publish/
2. Accept agreement
3. Pay the registration fee
4. Enter your details

Google Play Developer Console



Google Play Developer Console

Sign-in with your Google account | Accept Developer Agreement | Pay Registration Fee | Complete your Account details

YOU ARE SIGNED IN AS...

This is the Google account that will be associated with your Developer Console.

If you would like to use a different account, you can choose from the following options below. If you are an organization, consider registering a new Google account rather than using a personal account.

Sign in with a different account   Create a new Google account

BEFORE YOU CONTINUE...

Read and agree to the Google Play Developer distribution agreement.

☑ I agree and I am willing to associate my account registration with the Google Play Developer distribution agreement.

Review the distribution countries where you can distribute and sell applications.

If you are planning to sell apps or in-app products, check if you can have a merchant account in your country.

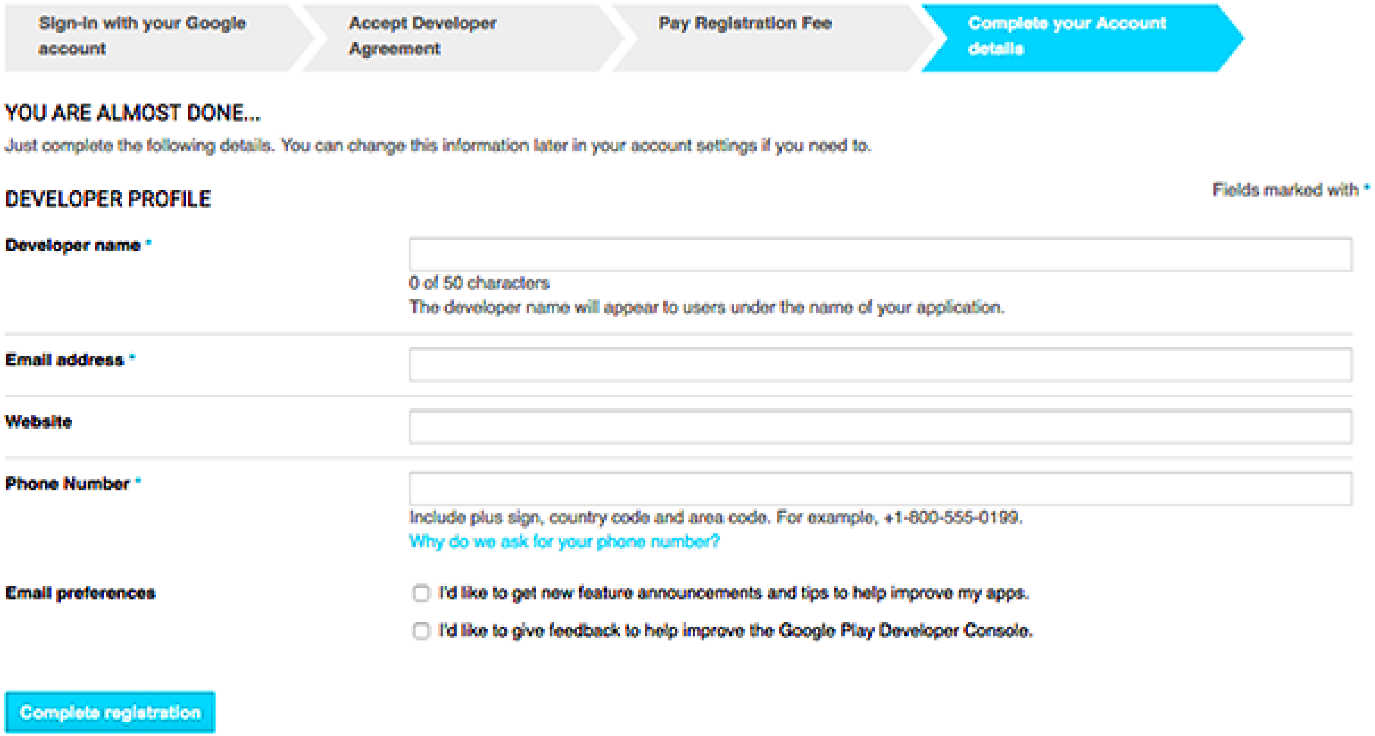
$25

Make sure you have your credit card handy to pay the $25 registration fee in the next step.

Continue to payment

play.google.com/apps/publish/

Enter your details



Sign-in with your Google account
Accept Developer Agreement
Pay Registration Fee
Complete your Account details

YOU ARE ALMOST DONE...

Just complete the following details. You can change this information later in your account settings if you need to.

DEVELOPER PROFILE

Fields marked with *

Developer name *

0 of 50 characters

The developer name will appear to users under the name of your application.

Email address *

Website

Phone Number *

Include plus sign, country code and area code. For example, +1-800-555-0199.

Why do we ask for your phone number?

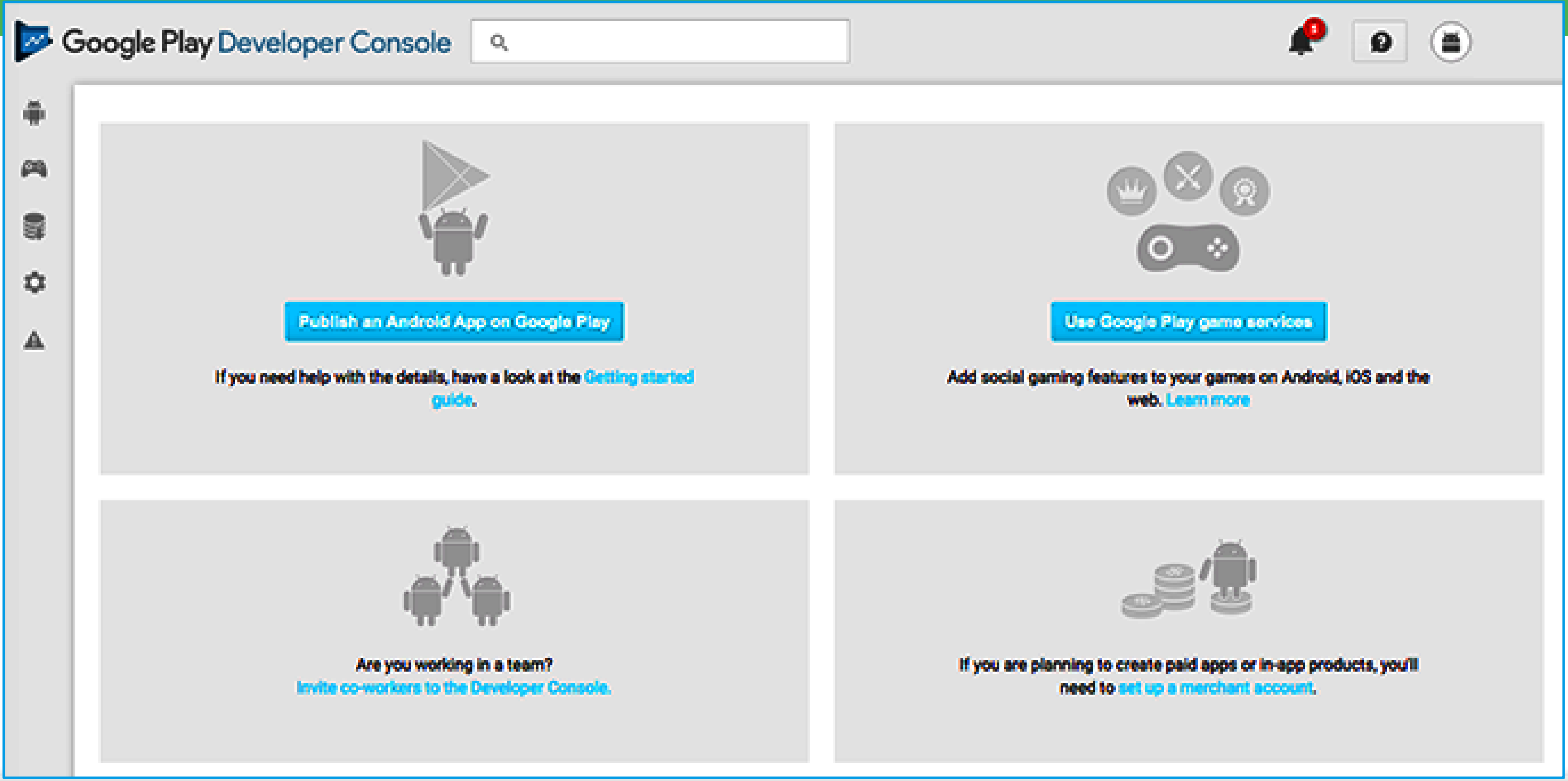
Email preferences

□ I'd like to get new feature announcements and tips to help improve my apps.

□ I'd like to give feedback to help improve the Google Play Developer Console.

- Name
- Address
- Website
- Phone
- Email preferences

You're in!



Google Play Developer Console

Publish an Android App on Google Play

If you need help with the details, have a look at the Getting started guide.

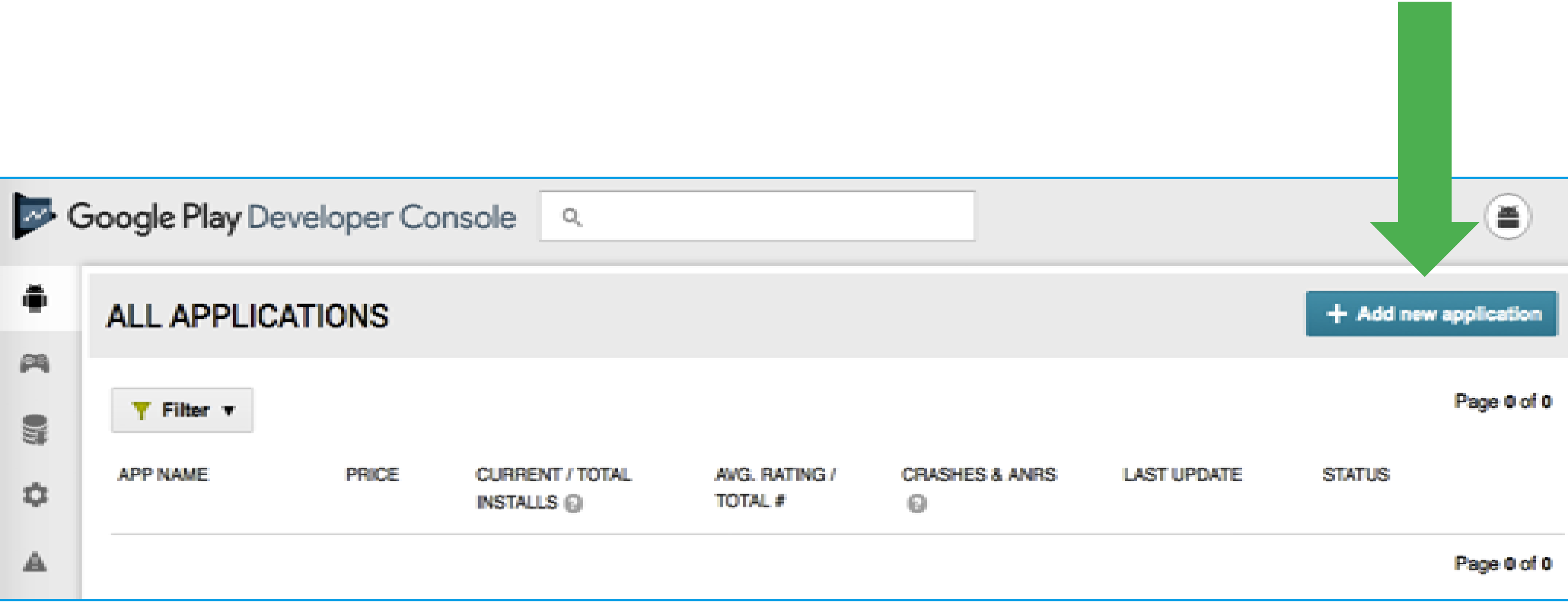
Use Google Play game services

Add social gaming features to your games on Android, iOS and the web. Learn more

Are you working in a team?
Invite co-workers to the Developer Console.

If you are planning to create paid apps or in-app products, you'll need to set up a merchant account.

Add an application



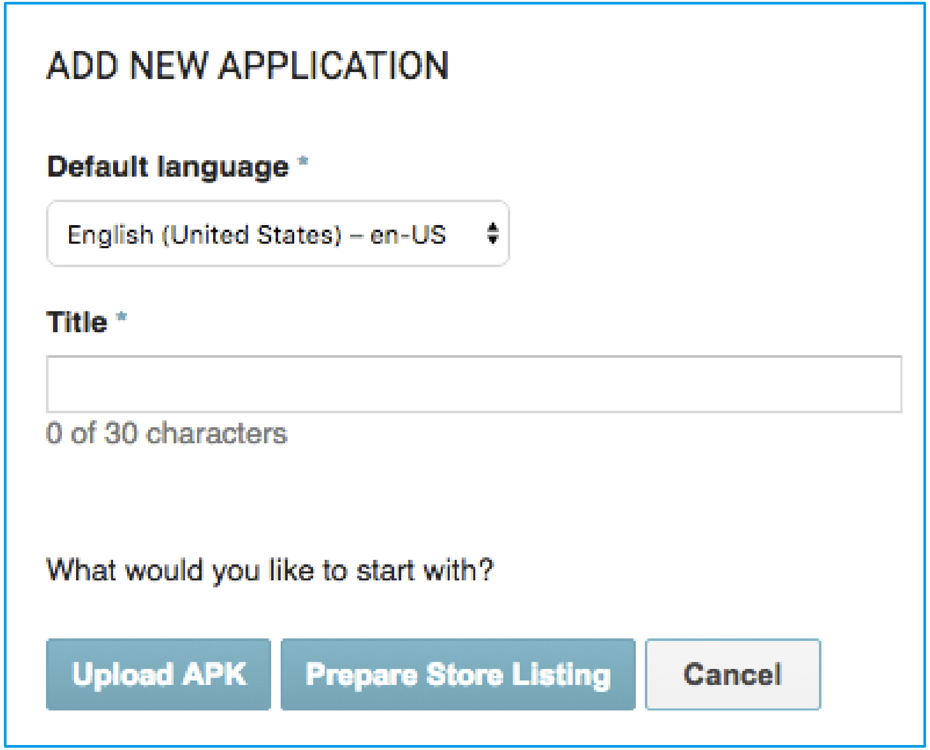
Add an application

- Upload APK

  or

- Prepare Store Listing

To run alpha and beta tests, upload the APK



ADD NEW APPLICATION

Default language *

English (United States) – en-US

Title *

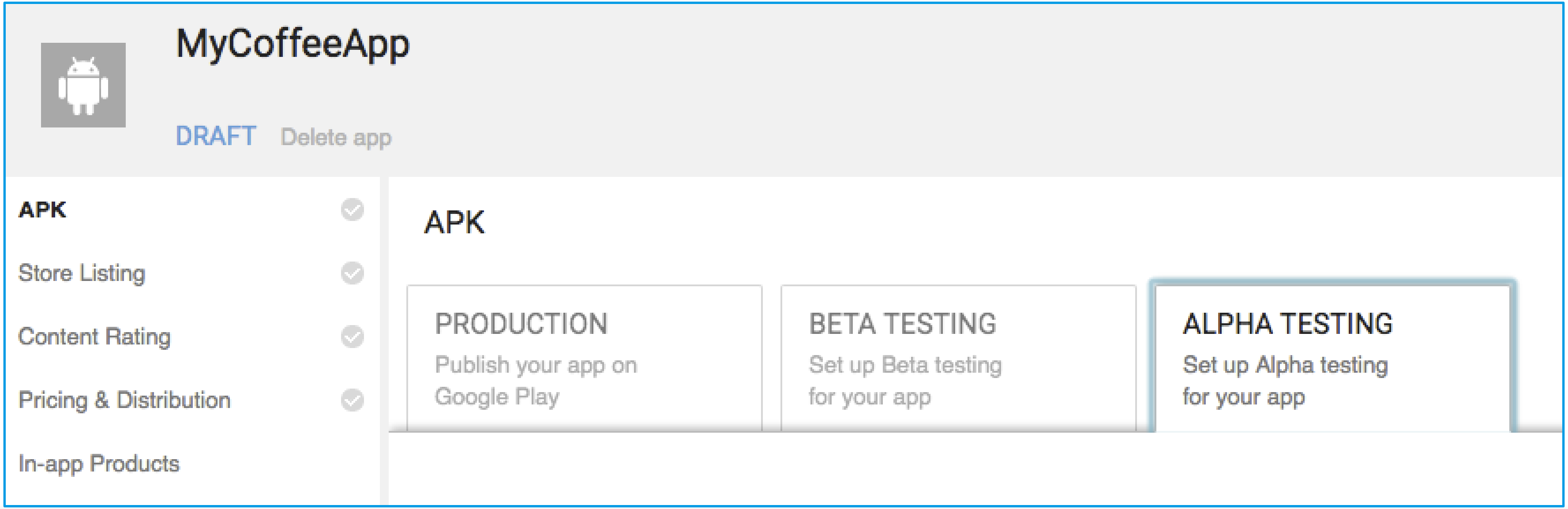
0 of 30 characters

What would you like to start with?

Alpha and Beta Tests

Run alpha and beta tests

Run alpha and beta tests in Google Play developer console



MyCoffeeApp

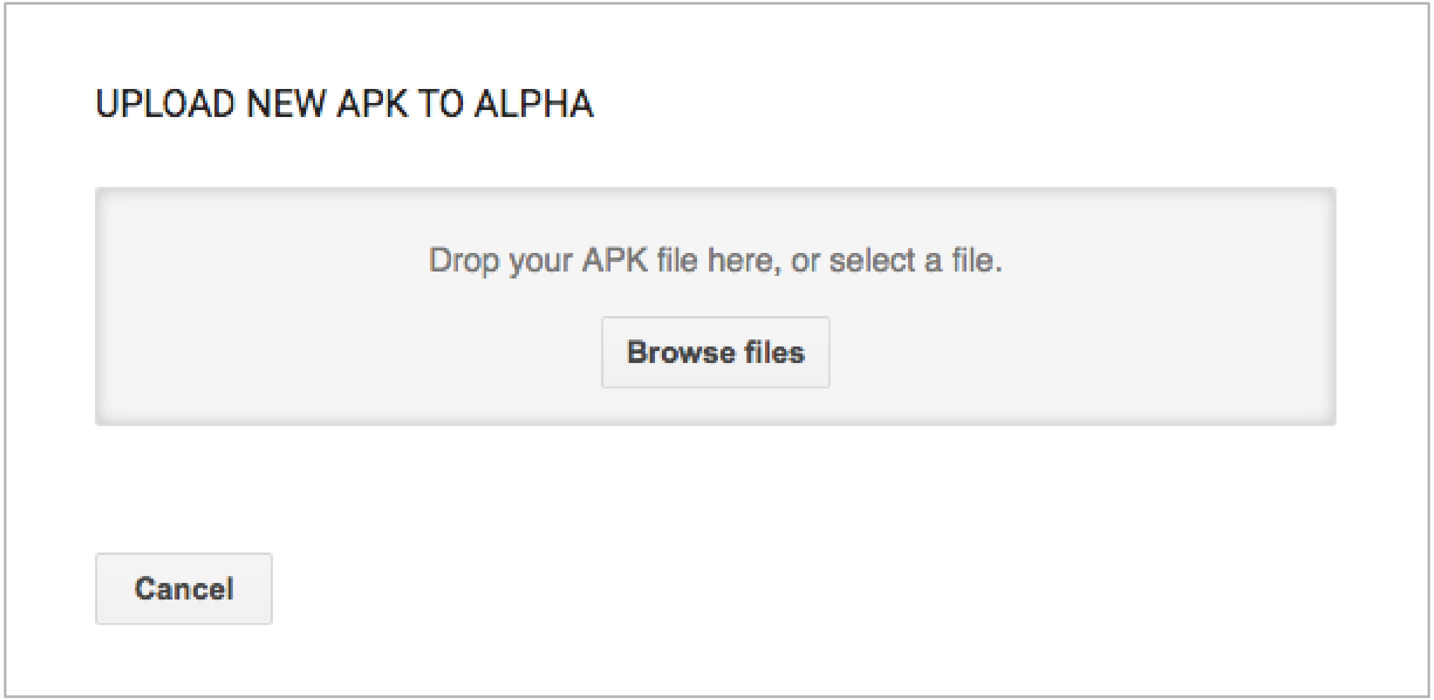
APK

PRODUCTION
Publish your app on Google Play

BETA TESTING
Set up Beta testing for your app

ALPHA TESTING
Set up Alpha testing for your app

Choose file to upload



UPLOAD NEW APK TO ALPHA

Drop your APK file here, or select a file.

Why run alpha and beta tests?

- Test your app before public release to help find and fix any technical or user experience issues
- Get the bugs out before you release your app to the world!
- Feedback from your test users does not affect your app’s public rating

Share with testers before publishing

- Alpha test during development

  Use alpha tests for early experimental versions of your app that might contain incomplete or unstable functionality

- Beta test with limited number of real users

  Use beta tests for apps that should be complete and stable

Alpha and beta tests

During alpha and beta tests

- Your app is not listed in Google Play
- Testers must have a link to get it
- Testers cannot give reviews in Google Play

Closed beta

People have to be invited to join closed alpha and beta tests

- Closed beta using email addresses
  - use lists of individual email addresses which you can add individually or upload as a .csv file
- Closed beta with Google+ community or Google Group

You can move closed betas to an open beta while maintaining your existing testers

Open beta

- Anyone with the link can join
- Can scale to a larger group
- You can limit max number of testers

Get feedback

- Closed tests:

  Provide a way for users to give feedback on your app, such as by email, website, or a message forum

- Open tests:

  Your testers can give private feedback in Google Play Store

  support.google.com/googleplay/android-developer/answer/138230#browse_reviews

Get private feedback for open betas

- For open tests (alpha or beta) program, you can see and reply to user feedback in the Developer Console
- Alpha and beta feedback from users is only visible to you and cannot be seen in the Google Play store

Pre-launch Report

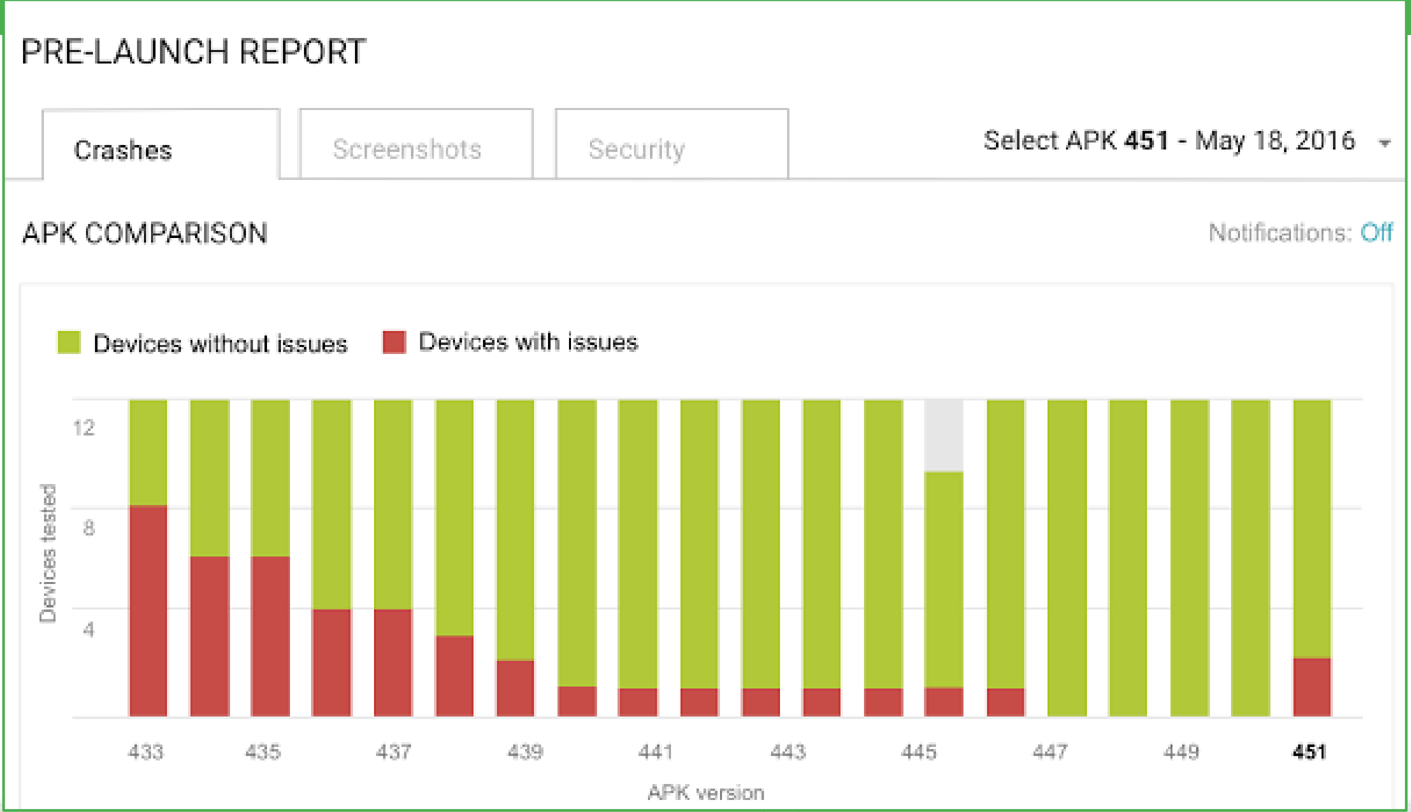
Test before you publish

Opt in for pre-launch reporting

- Pre-launch reports identify crashes, display issues, and security vulnerabilities
- During pre-launch check, test devices automatically launch and crawl your app for several minutes
- The crawl performs basic actions every few seconds on your app, such as typing, tapping, and swiping

support.google.com/googleplay/android-developer/answer/7002270

Example pre-launch report



PRE-LAUNCH REPORT

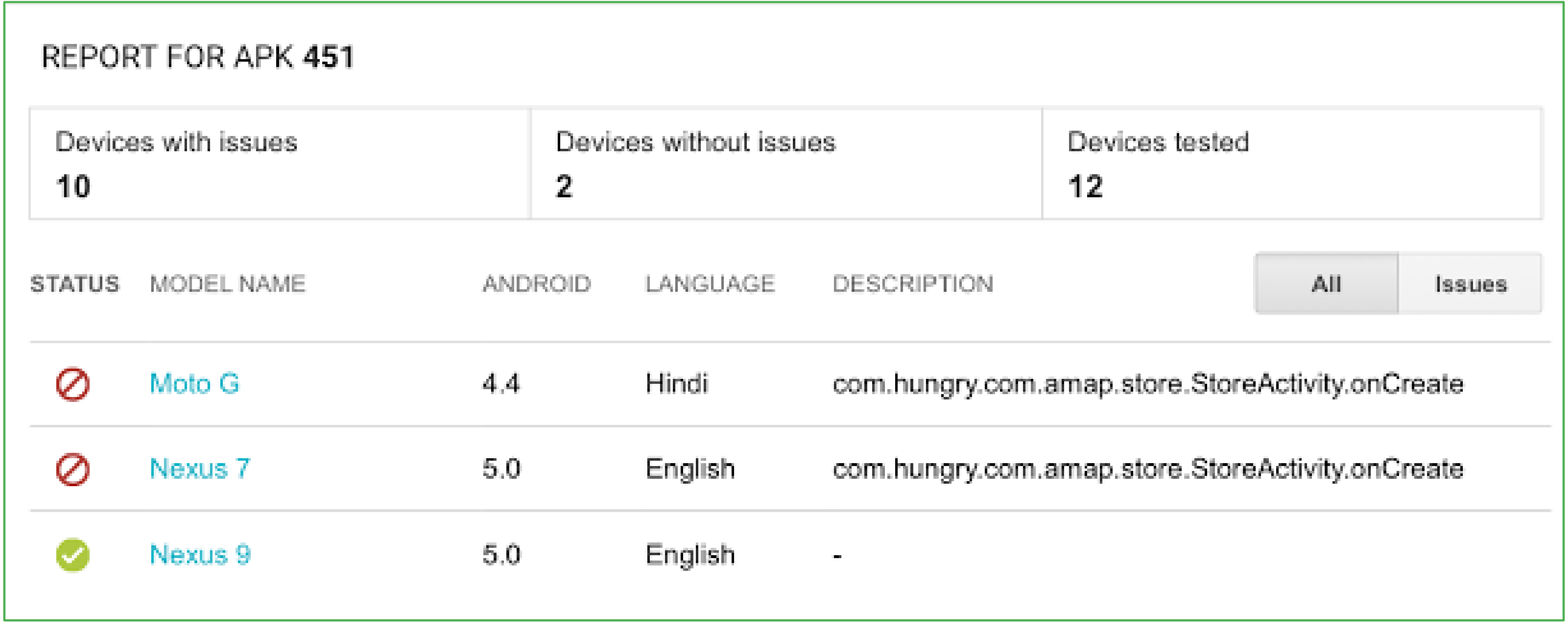
Crashes | Screenshots | Security

Select APK **451** - May 18, 2016

APK COMPARISON

Notifications: Off

Example pre-launch report



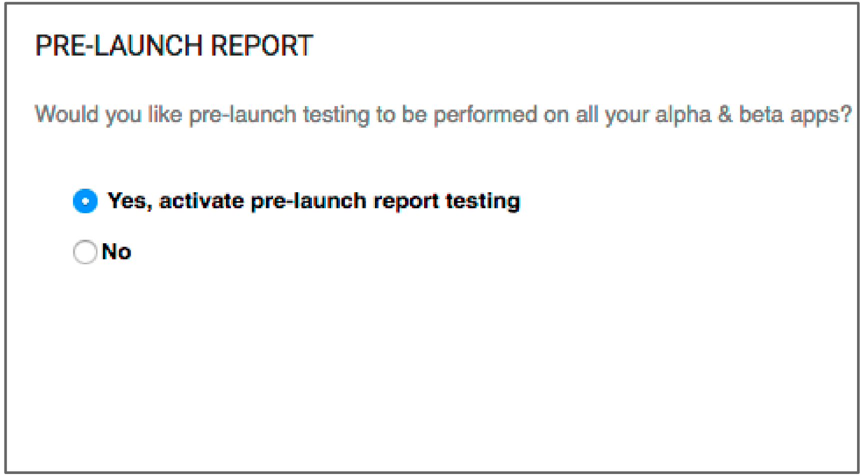
REPORT FOR APK **451**

Devices with issues	Devices without issues	Devices tested
10	**2**	**12**

STATUS	MODEL NAME	ANDROID	LANGUAGE	DESCRIPTION
⊘	Moto G	4.4	Hindi	com.hungry.com.amap.store.StoreActivity.onCreate
⊘	Nexus 7	5.0	English	com.hungry.com.amap.store.StoreActivity.onCreate
✓	Nexus 9	5.0	English	-

Running pre-launch reports

- Opt-in to pre-launch reports in the console
- Then when you upload or publish an alpha or beta APK, the pre-launch test runs automatically



PRE-LAUNCH REPORT

Would you like pre-launch testing to be performed on all your alpha & beta apps?

● Yes, activate pre-launch report testing

○ No

Criteria for Publishing

Only publish high quality apps

Your app must meet core app quality requirements

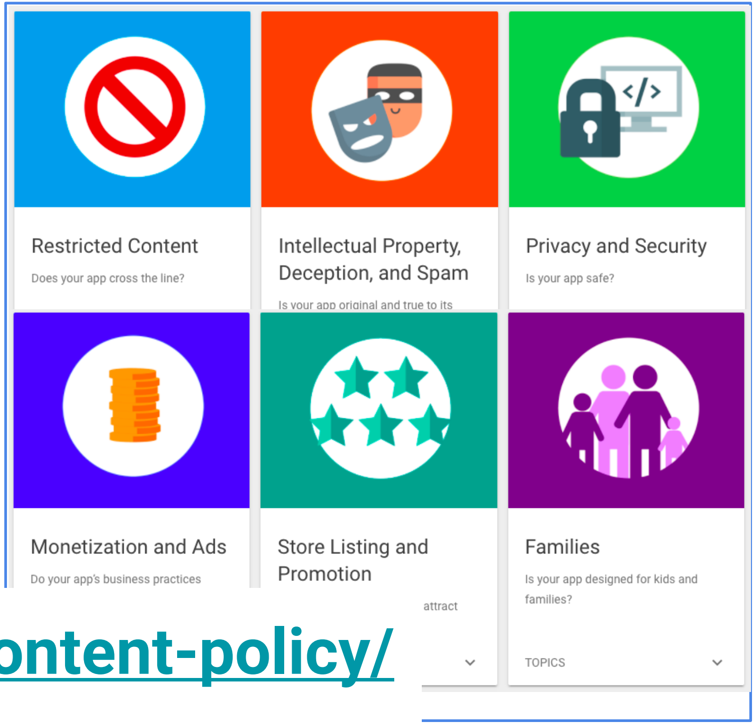
- Visual design and user interaction
- Functionality
- Performance and stability

Android users expect high-quality apps!

developer.android.com/distribute/essentials/quality/core.html

Comply with Google Play policies

Google Play policies ensure that all apps on Google Play provide a safe experience for everyone



Restricted Content

Does your app cross the line?

Intellectual Property, Deception, and Spam

Is your app original and true to its

Privacy and Security

Is your app safe?

Monetization and Ads

Do your app's business practices

Store Listing and Promotion

attract

Families

Is your app designed for kids and families?

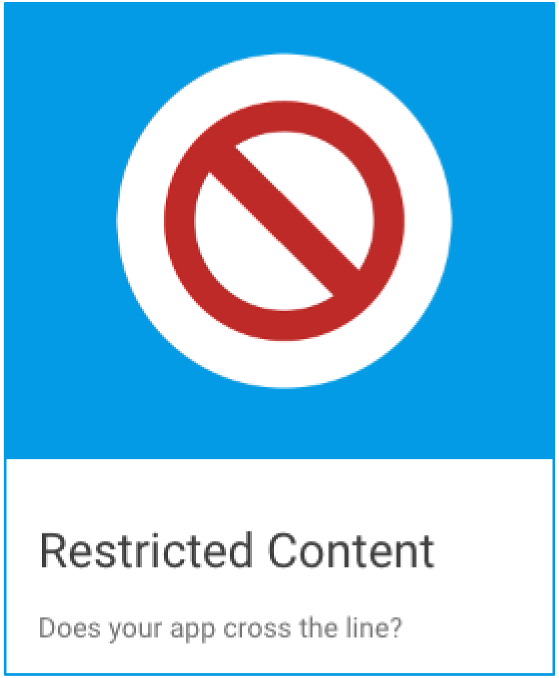
TOPICS

play.google.com/about/developer-content-policy/

Restricted Content

- Sexually Explicit Content
- Child Endangerment
- Violence
- Bullying & Harassment
- Hate Speech
- Sensitive Events
- Gambling
- Illegal Activities

play.google.com/about/restricted-content

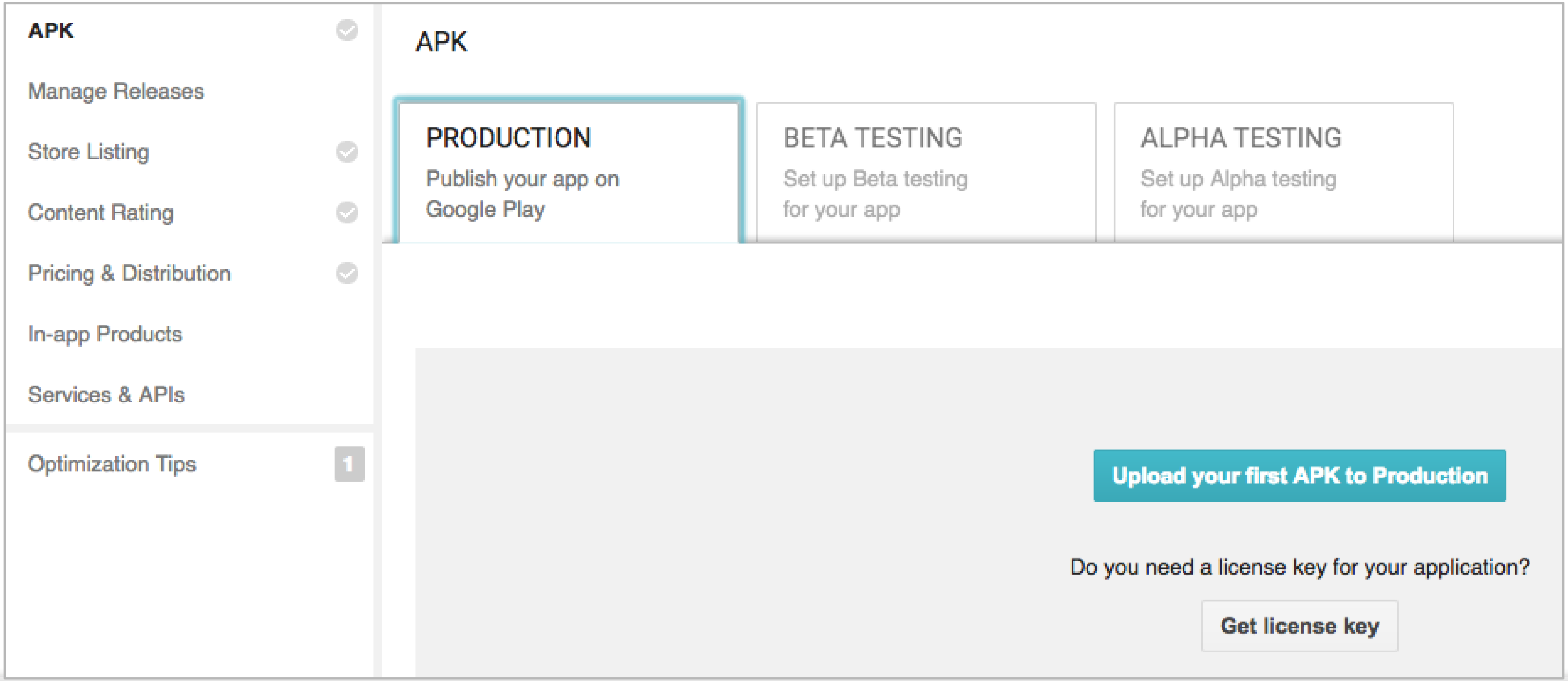


Restricted Content

Does your app cross the line?

Publish to the world!

Upload your APK to Production

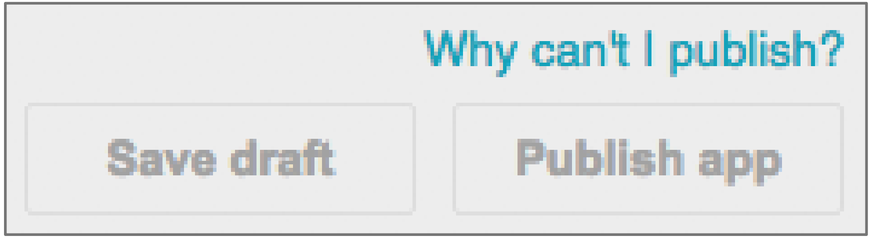


APK ✓
Manage Releases
Store Listing ✓
Content Rating ✓
Pricing & Distribution ✓
In-app Products
Services & APIs
Optimization Tips 1
APK
PRODUCTION
Publish your app on Google Play
BETA TESTING
Set up Beta testing for your app
ALPHA TESTING
Set up Alpha testing for your app
Upload your first APK to Production
Do you need a license key for your application?
Get license key

Check what's missing

- add a high-res icon
- add a feature graphic
- add 2 non-Android TV screenshots
- select a category
- select a content rating
- target at least one country
- enter a privacy policy URL
- make your app free or set a price for it
- declare if your app contains ads
- add a required content rating

Google Play tells you what is missing



Google checks your app

- Automatic and manual checking
- Your app can be rejected for violating any requirement
- If you get rejected, fix the problem and try again

Learn more

Learn more about prepping your app

- Review the launch checklist
- Core app quality checklist
  developer.android.com/distribute/essentials/quality/core.html
- Handling user data
  play.google.com/about/privacy-security/user-data/
- Design for tablets and handsets
  developer.android.com/guide/practices/tablets-and-handsets.html
- Min, max, and target API levels
  developer.android.com/guide/topics/manifest/uses-sdk-element.html

Learn more about prepping your app

- Preparing for release
  https://developer.android.com/studio/publish/preparing.html
- Google Play filters
  developer.android.com/google/play/filters.html
- Reduce app size
  developer.android.com/topic/performance/reduce-apk-size.html
- Sign your app
  developer.android.com/studio/publish/app-signing.html

Learn more about publishing your app

Google Play Developer Console:

- Go to the console:
  play.google.com/apps/publish/
- Dev guide:
  developer.android.com/distribute/googleplay/developer-console.html
- Help center:
  support.google.com/googleplay/android-developer/#topic=3450769

Learn more about publishing your app

- Get started publishing
  developer.android.com/distribute/googleplay/start.html
- Alpha and Beta testing
  Dev guide:
  developer.android.com/distribute/engage/beta.html

  Help center:
  support.google.com/googleplay/android-developer/answer/3131213

What's next?

- This is the end of the course!
- Build great apps and publish them

END