Mobile Computing
Getting Started with Android

Slides: tmb.gr/mc-ndr

Source on GDocs

Overview

These slides show how to get started with Android.

How to set up a development environment (IDE).

How apps are structured and how to run them.

Prerequisites

Download AndroidStudio for Mac/Windows/Linux.

Have some basic knowledge of writing Kotlin code.

Bring your Android device* or use the emulator.

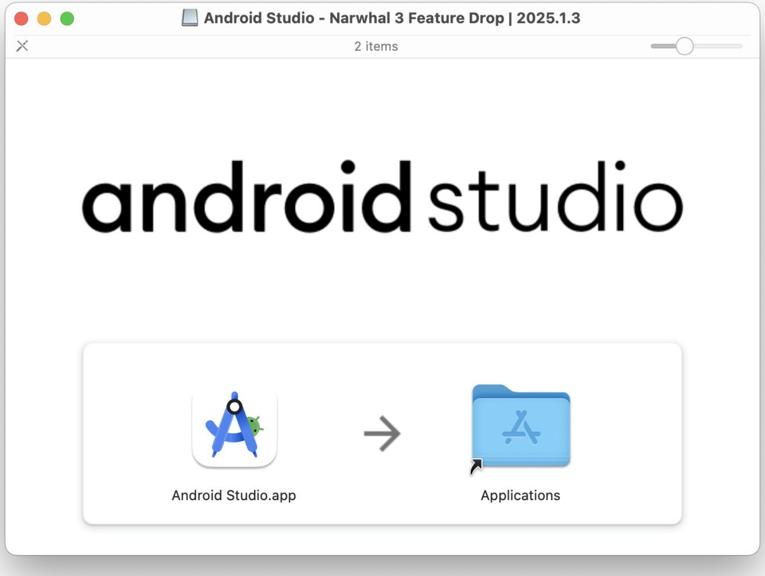
Install git and sign up for a Github account.

*Some schools lend devices to students.

Install AndroidStudio .html

Install the AndroidStudio version* you downloaded.

- Open the installer, e.g. *android-studio-2025.1.3.x-mac_arm.dmg* (on Mac)



Android Studio - Narwhal 3 Feature Drop | 2025.1.3

2 items

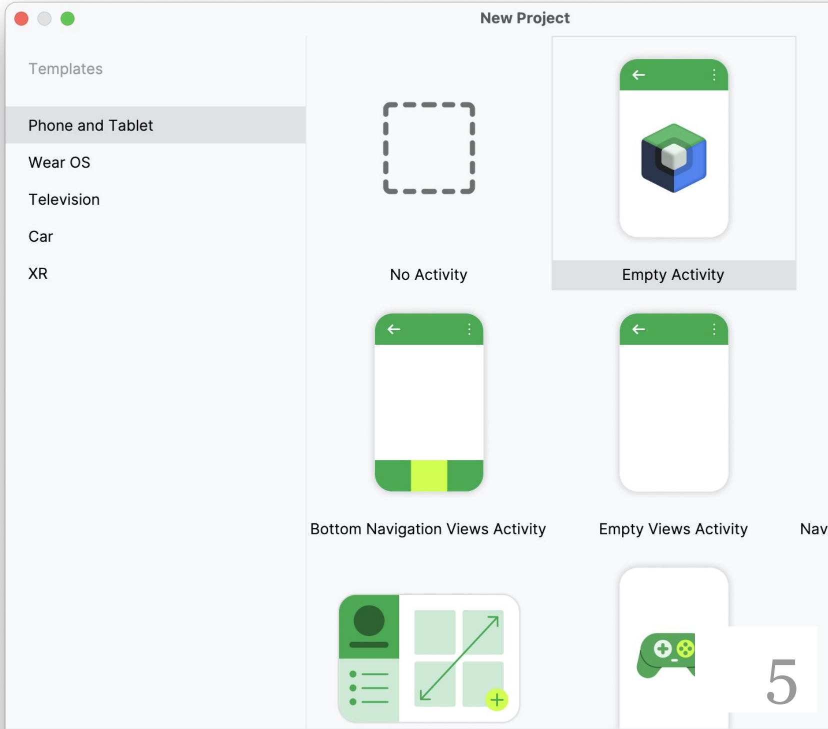
*Installed it long ago? Update, when asked.

Create a first app

.html

Create a first app from a template.

- Open *AndroidStudio*
- *Projects > New Project*
- *Empty Activity > Next*
- *Minimum SDK = 24*
- *Finish > … > Finish**



**Gradle sync* can take a while.

Minimum and target SDK

.html

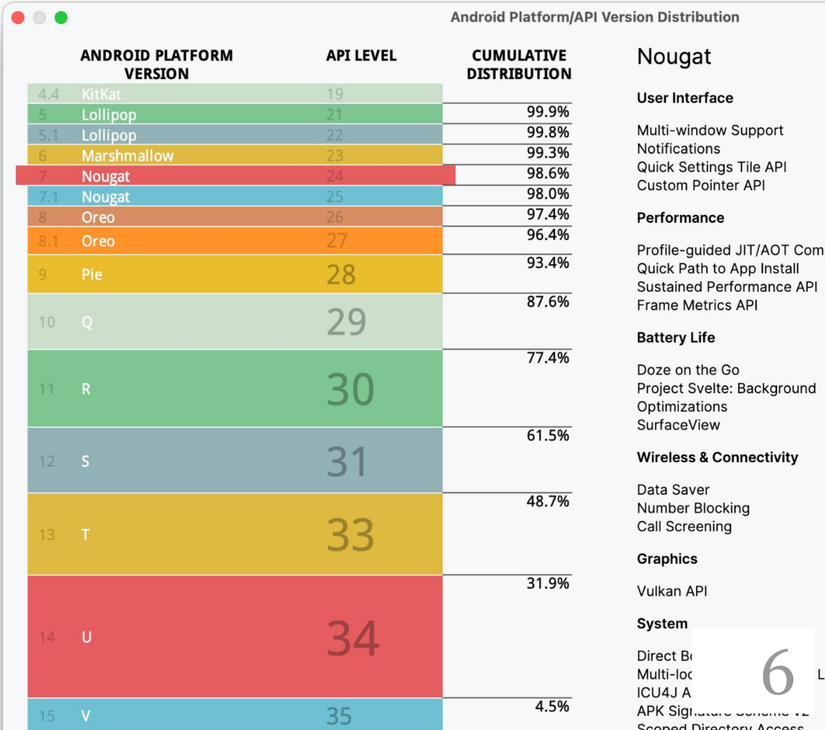
Minimum SDK is the earliest supported SDK version*.

Lower means more devices.

Target SDK is the version which you develop/test for.

Higher means more features.

*API level != Android version.



Android Platform/API Version Distribution

ANDROID PLATFORM VERSION		API LEVEL	CUMULATIVE DISTRIBUTION
4.4	KitKat	19	
5	Lollipop	21	99.9%
5.1	Lollipop	22	99.8%
6	Marshmallow	23	99.3%
7	Nougat	24	98.6%
7.1	Nougat	25	98.0%
8	Oreo	26	97.4%
8.1	Oreo	27	96.4%
9	Pie	28	93.4%
10	Q	29	87.6%
11	R	30	77.4%
12	S	31	61.5%
13	T	33	48.7%
14	U	34	31.9%
15	V	35	4.5%

Nougat

User Interface

Multi-window Support
Notifications
Quick Settings Tile API
Custom Pointer API

Performance

Profile-guided JIT/AOT Com
Quick Path to App Install
Sustained Performance API
Frame Metrics API

Battery Life

Doze on the Go
Project Svelte: Background Optimizations
SurfaceView

Wireless & Connectivity

Data Saver
Number Blocking
Call Screening

Graphics

Vulkan API

System

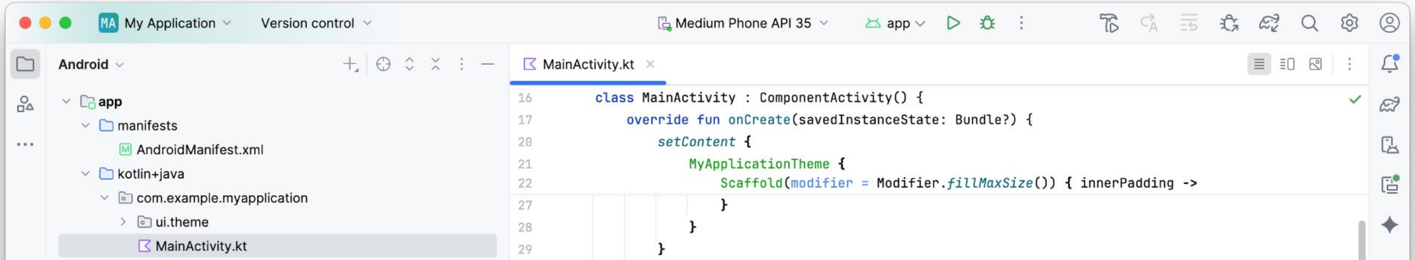
Direct B
Multi-loc
ICU4J A
APK Sig
Scoped Directory Access

Find project files

.html

Find project files in the AndroidStudio IDE.

- Open *AndroidStudio* > *Project* tab (folder icon).
- Select *Android* to see the development view.
- Select *Project files* to see the file structure.
- Double click a file in the tree to open it.



```
class MainActivity : ComponentActivity() {
    override fun onCreate(savedInstanceState: Bundle?) {
        setContent {
            MyApplicationTheme {
                Scaffold(modifier = Modifier.fillMaxSize()) { innerPadding ->
                }
            }
        }
```

AndroidManifest .xml|.html

The *AndroidManifest.xml* file contains components, permissions, hard- and software features of the app.

```
<manifest …>
  <application …
    <activity android:name=".MainActivity" …
      <intent-filter>
        <action android:name="….MAIN" />
        <category android:name="….LAUNCHER" />
```

MainActivity

.kt|.html

The *MainActivity* is the entry point* into the app.

```
class MainActivity : ComponentActivity() {
  override fun onCreate(…) { … } // **
}
```

*There is no *main()* function in an Android app.

**We will use 2-space indents, … for code snippets.

@Composable

.kt|.html

The *@Composable* annotation is part of Compose UI*.

Composable functions can be used in *setContent()*.

This allows you to create custom UI components.

```
@Composable
fun Greeting(name: String, …) { … }
```

*More on this in the next lesson.

Build.gradle

.kts|.html

Build.gradle.kts configures the Android build system.

```
plugins { … } // for the build tool itself
android { …
  compileSdk = 36 // usually same as targetSdk
  defaultConfig {
    minSdk = 24
    targetSdk = 36 … } }
dependencies { … } // to Java/Kotlin libraries
```

Run apps on the emulator .html

Run apps on the emulator, on a virtual device (AVD).

- Open *AndroidStudio > Tools > Device Manager*
- Click the plus symbol + to create a virtual device
- Select the *Phone* category, and e.g. *Pixel 6* (or other)
- Select the virtual device *API level*, e.g. *34* (or other)
- Download the *release* (if ⤓ is shown) > … > *Finish*
- Close Device Manager, select AVD, run the app.

Connect your Android device .html

- On the device, open *Settings > About phone*
- Tap *Build number* seven times (*You're now a dev!*)
- Go to *Settings > System > Advanced options*
- Tap *Developer options > USB debugging [on]*
- (On Windows, open *AndroidStudio > Tools > SDK Manager > SDK Tools > [x] Google USB driver*)
- Connect the device via USB, *Allow USB debugging*
- In *AndroidStudio*, select your device, run an app.

Hands-on, 10': Run your first app

Go through the steps described so far.

- Install AndroidStudio.
- Create a first app.
- Run the app*.

*On the emulator, on your device, or both.

Copy our course repository

Get a private copy of our repository (do not fork it).

- Click *Use this template > Create new repository*.
- Name it *mse-tsm-mobcom-YOUR_GITHUB_USER*
- Set visibility to *Private > Create repository*.
- Use *git* to *clone* your private repository (not ours)

```
$ git clone git@github.com:YOUR_GITHUB_USER/\
mse-tsm-mobcom-YOUR_GITHUB_USER.git
```

Use your private repository

Work in private, add me as a collaborator, to share.

- Open your repository directory in a terminal, e.g.

```
$ cd mse-tsm-mobcom-YOUR_GITHUB_USER
```

- Use *git* to *add* files, *commit* and *push* changes.

```
$ git status          $ git commit my.txt
$ git add my.txt      $ git push
```

- *Settings > Collab… > Add People > @tamberg*
- To share code with me, post links in MS Teams.

Update your repository

Update your private copy of our repository (weekly).

- Open your repository directory in a terminal, e.g.

```
$ cd mse-tsm-mobcom-YOUR_GITHUB_USER
```

- Use *git* to *add upstream* (once), *fetch* and *merge*.

```
$ git remote add upstream git@github.com:\
tamberg/mse-tsm-mobcom.git && git remote -v
$ git fetch upstream && git checkout main
$ git merge upstream/main --allow-un<TAB>
```

Open an existing app

Open an existing app from your repository.

- Open *AndroidStudio* > *Projects* > *Open* > … *mse-tsm-mobcom-YOUR_GITHUB_USER/01/Android/MyHelloApp* (or other *MyXYApp*).
- *Open* > *Trust Project**.

**Gradle sync* can take a while.

Hands-on, 5': Set up your repository

Go through the steps described so far.

- Copy the course repository (template).
- Use *git add upstream* as described (once).
- Use *git fetch* and *merge* later on (weekly*).
- Open and run the *MyHelloApp* project.

*As the course repository is still work in progress.

Hands-on, 5': Push a change

Make a small change, *commit* and *push* it.

- Open the *MyHelloApp* in your private repository.
- Change the greeting *name* in *MainActivity.kt*.
- Use *git status* and git *diff* to see what changed.
- Use *git commit* to store the changes locally.
- Use *git* to *push* the change back to Github.

Consider adding me to your repository.

Summary

These are the basics of getting started with Android.

Setting up *AndroidStudio* and creating a first app.

Looking at project files and running it on device.

Working with git to commit and push changes.

Next: Composing UIs for Android.

Challenge: Build a "real" 3rd-party app

Study the Android wallet project by swiyu.

- Use *git clone* to get *https://github.com/swiyu-admin-ch/eidch-android-wallet*
- Open the project in *AndroidStudio*.
- Decide if you can trust this 3rd-party.
- Take a quick look at the project structure.
- Try to build and run the app (if possible).

Feedback or questions?

Write me on Teams or email

thomas.amberg@fhnw.ch

Thanks for your time.