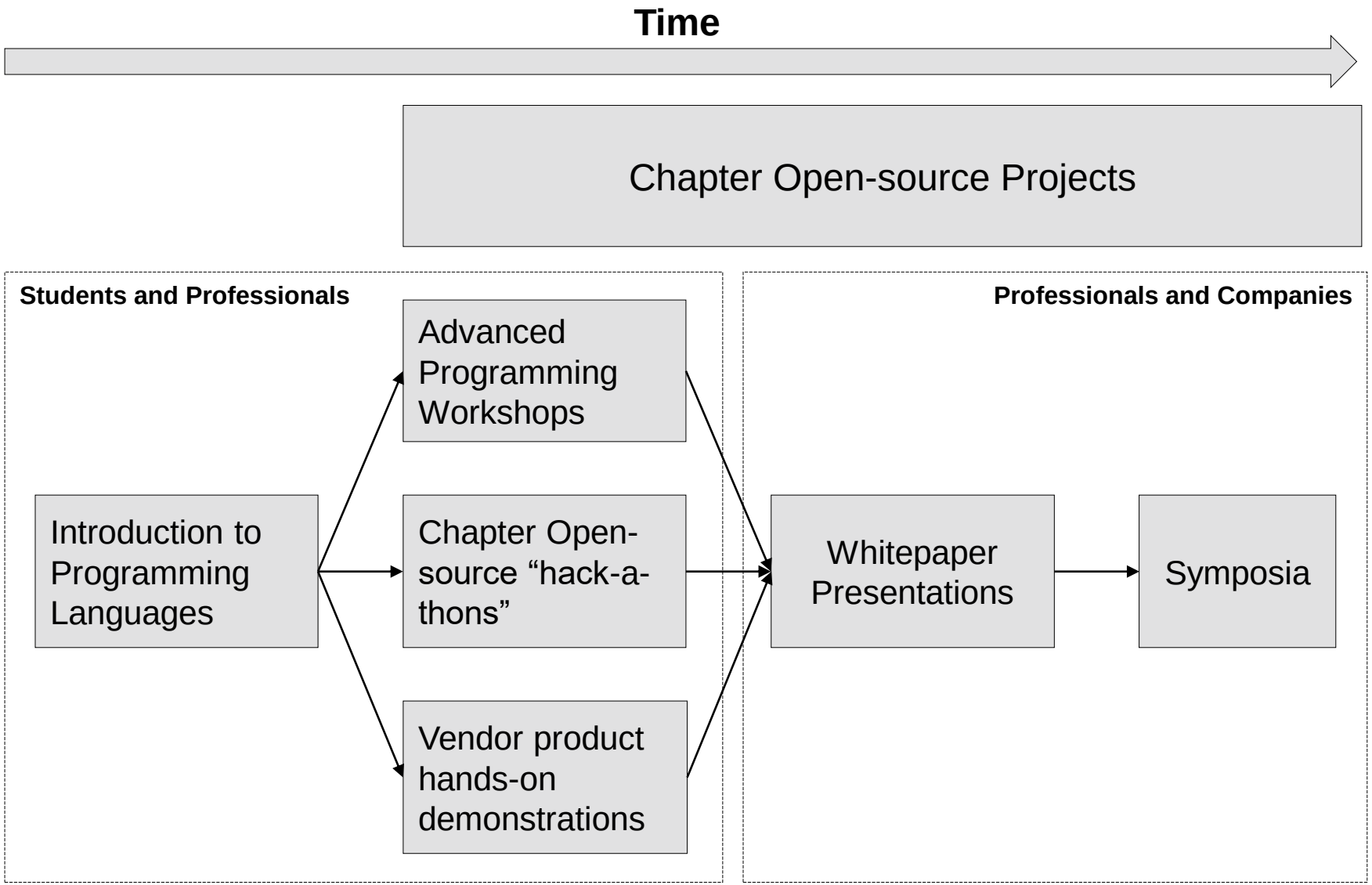
green COW coworking

sme

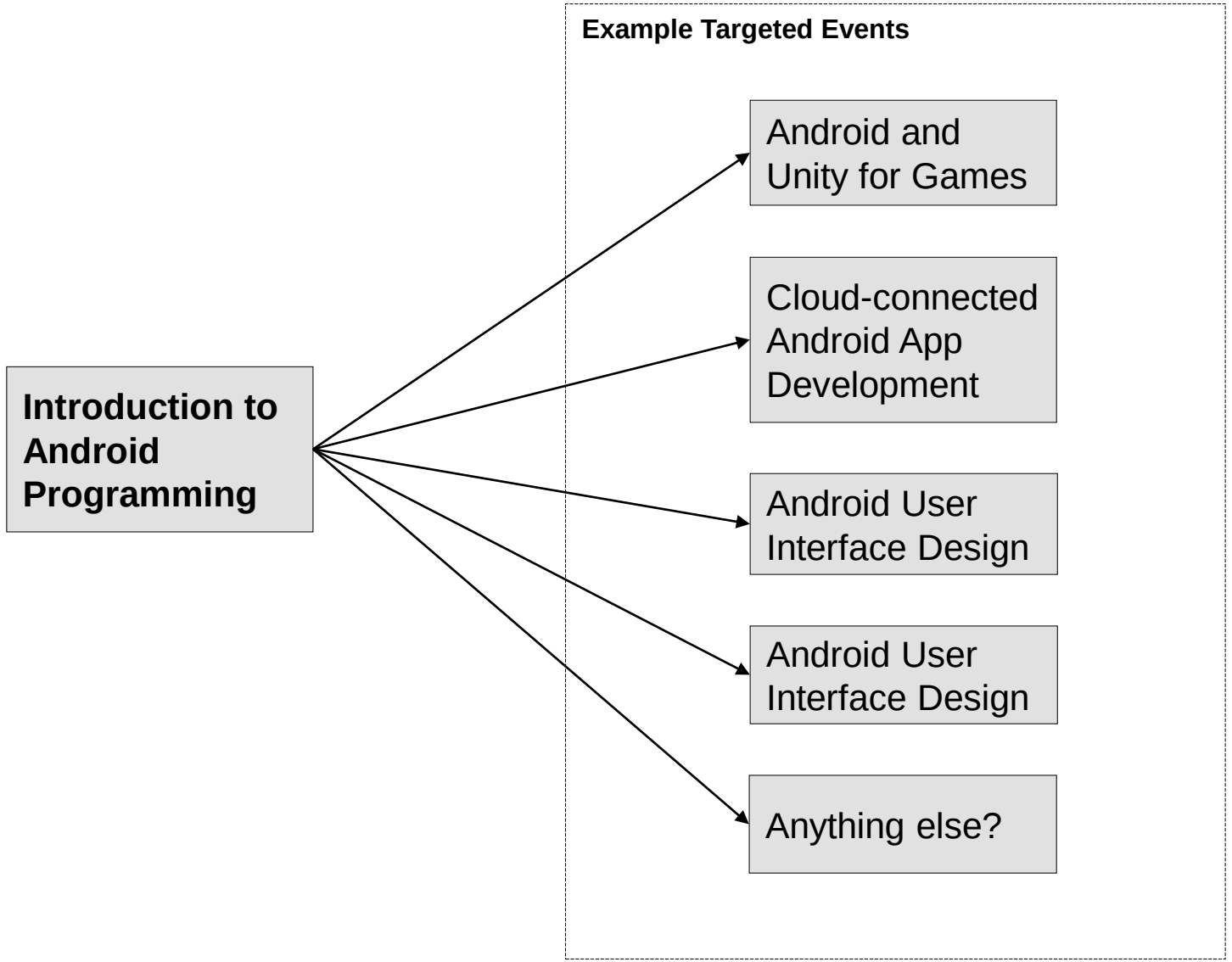
# Introduction to Android Programming

**Adam J. Cook, Chair of SME Chapter 112**

# Chapter “Digital” Initiative



# Possible Future Android Events



# Android at 10,000 Feet



- Free and **open-source** operating system and runtime platform.
- Apps written mostly in the Java (or Kotlin) programming language.
- Powerful and free development environment (Android Studio).
- Apps can be developed and simulated on any platform (Windows, MacOS or Linux).
- New Android devices sold were 81.7% globally in 2016 Q4. The rest is basically iOS. Many popular apps target both Android and iOS natively to give their user's the best experience on the devices that they prefer.
- Apps are deployed to a publicly available "app store" (Google Play).
- Apps can be free, paid or consist of various in-app payment schemes. **We will talk a little more about this.**
- Apps can be upgraded "in-place" on user devices when a new version is released.
- Android apps can run on watches, TVs, VR headsets, tablets, automobiles and phones.

# What does “open-source” mean?

- The Open Source Definition - https://opensource.org/osd-annotated
- Free redistribution, royalty-free.
- Source code availability.
- Popular open-source licenses – MIT, BSD, Apache 2.0, GPL, LGPL.
- The use of open-source components in larger applications is very common and you can use them too!
- **Always** check with your legal representation if a non-standard license is encountered or if you are unsure of your legal rights and responsibilities with the standard open-source licenses.

# Today's Agenda



- A brief walkthrough of the "mechanics" of Android app development and deployment.
- Try to do the official Android starter project (if you want).
- Discussion of some very useful resources for future study.

# Caveats and Warnings



- Programming is challenging – this event will not make you into an expert. **Practice, read code, read books, watch videos and ask questions!**
- Android app development is a combination of visual programming and Java (or Kotlin) programming. These can be complex languages at times and will require some work to become comfortable with.
- Every programming project should use a version control system. Git is **highly** recommended. See the resources at end.
- “Premature optimization is the root of all evil.” – Donald Knuth

# Ins and outs



**The Business Case**

- Why even learn Android programming at all?
- Well, two basic reasons:
  - Your primary business value will revolve around mobile applications so you need to build one.
  - One element (but not only one) of your sales channel is through your mobile application.
- Sure, but why don’t I just contract out the mobile application development? You can! But, consider which of the two cases above correspond to your business.
- Even if you do decide to contract out the development, you should be somewhat aware of the technical challenges.
- The good news? Android development has never been “easier”!

# Ins and outs



**How do you monetize Android apps?**

- There are three popular ways:
  - The app is free, but you sell your products/services through it (indirect).
  - There is an upfront fee that must be paid before someone can download your app (direct).
  - The app is initially free, however, users can “unlock” additional features and functionality while using by paying a fee (direct, in-app billing). You need API Level 8 or above for this feature.
- For the second case (an upfront fee), consider creating **two versions** of your app:
  - Free, but reduced functionality version (or a full featured time-limited trial).
  - Paid version which has the full functionality.

Sources:
https://developer.android.com/google/play/billing/index.html

# Ins and outs

**Any other ways to make money with your Android app? Sure!**

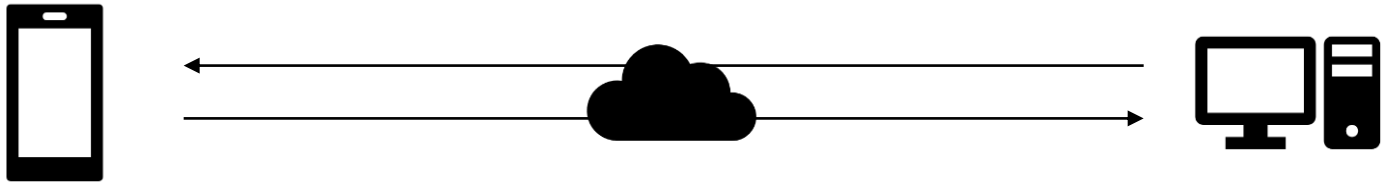
- Display third-party ads in your app.
- Sell physical goods and services through your app using Android Pay.
- Make your app subscription-based. You can choose to have your users billed on a periodic basis to use your app or certain app features.

Sources:
https://developer.android.com/distribute/google-play/about.html#monetize-your-apps

# Ins and outs

**Let’s quickly look a two basic app architectural choices. You will need to pick one.**

**First, device-server architecture. Think the Facebook or the Angry Birds app.**



Your mobile app  Internet  Your server

**Second, standalone architecture. Popular with some games and productivity apps.**

Your mobile app  Internet

# Ins and outs

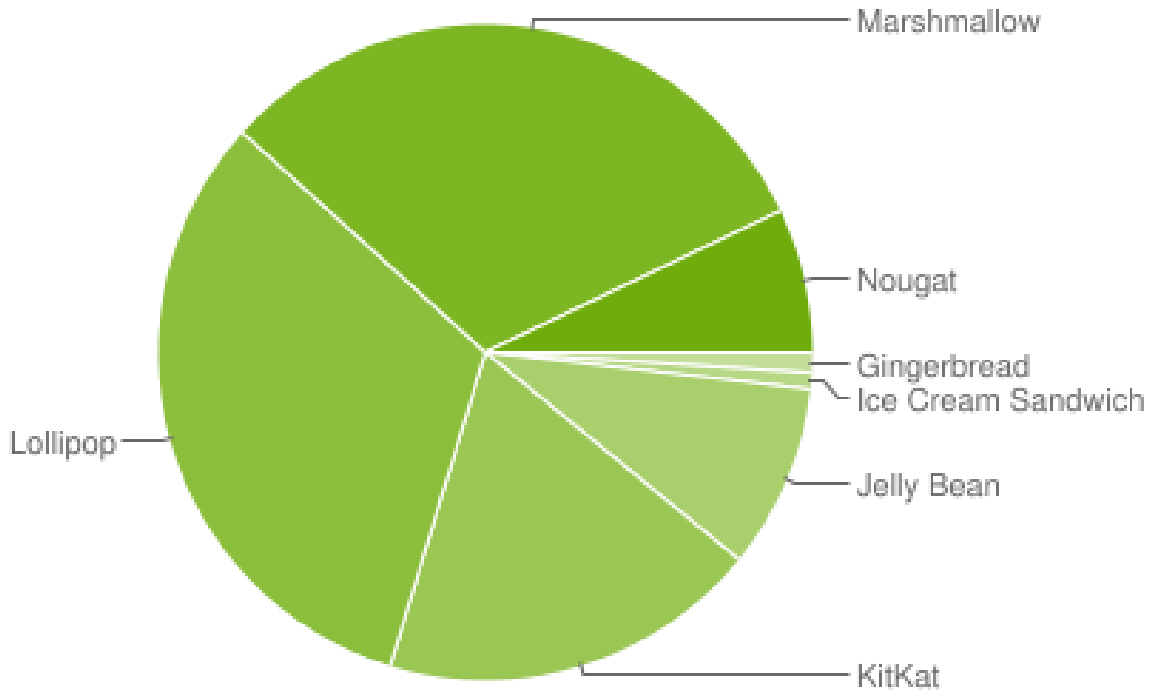


**Terminology**

- Android – the mobile operating system itself.
- Android Version – a convenient name for device users. All Android versions are desert names and three numbers separated by dots.
- API (or Application Programming Interface) – describes the **development functionality** that you can use in your mobile application. Keep that in mind for the next slides!
- SDK (or Software Development Kit) – a set of development tools that allows the creation of mobile applications. **Sometimes, SDK and API are used interchangeably in Android.**
- Android Runtime (sometimes called ART, Android or the now obsolete term “Dalvik”) – this is the environment for on top of which Android applications run.
- APK (or Android Package Kit) – package file format used by Android for distribution and installation of mobile applications. This is what you will build!

# Ins and outs

## Platform Market Share




| Version | Codename | API | Distribution |
|---|---|---|---|
| 2.3.3 - 2.3.7 | Gingerbread | 10 | 1.0% |
| 4.0.3 - 4.0.4 | Ice Cream Sandwich | 15 | 0.8% |
| 4.1.x | Jelly Bean | 16 | 3.2% |
| 4.2.x | | 17 | 4.6% |
| 4.3 | | 18 | 1.3% |
| 4.4 | KitKat | 19 | 18.8% |
| 5.0 | Lollipop | 21 | 8.7% |
| 5.1 | | 22 | 23.3% |
| 6.0 | Marshmallow | 23 | 31.2% |
| 7.0 | Nougat | 24 | 6.6% |
| 7.1 | | 25 | 0.5% |

Sources:
https://developer.android.com/about/dashboards/index.html

# Ins and outs



**Why does market share matter?**

- In short, newer versions of Android are more powerful, but, as a developer you have to decide how to balance that power with market availability.
- Android applications are generally “forward-compatible” with new versions, but are not generally “backward-compatible”.
- Code Hint! The minimum Android (SDK) version and target version with the **`android:minSdkVersion`** and **`android:targetSdkVersion`** attributes of the **`uses-sdk`** tag.
- Not recommended, but it is possible to create multiple APKs for different Android version levels. This can be a real hassle so make sure you need this! Consider using “fallbacks” instead (Support Libraries).

Sources:
https://developer.android.com/guide/topics/manifest/uses-sdk-element.html
https://developer.android.com/training/multiple-apks/api.html

# Ins and outs

**Ok. I am finished with my app. How do I get it out there?**

- Apps are primarily distributed through the Google Play Store.
- Google Play lets users find, review and download your app and any future app updates.
- All financial transactions are handled directly through Google Play so you do not have to do it.
- Remember, though, Google keeps 15% of the revenue of your app (transaction fee), so price your app accordingly.
- Google does not review your app prior to publishing it, however, there are certain types of apps which are restricted/prohibited.
- The Google Play Console is like a “launch room” for your app.

Sources:
https://developer.android.com/studio/publish/index.html

# Ins and outs

**Can you tell me more about the Google Play Console?**

- Google charges a one-time registration fee of $25 to publish unlimited amounts of apps.
- Google Play Console features:
  - Shows app performance like service usage and crashes.
  - Allows for the management of your app releases.
  - Allows edits to your pricing, distribution channels and Google Play store listing details.
  - View app analytics such as ad campaigns, promotions and user acquisition.
  - Keep an eye on user feedback and ratings.
- Let’s take a quick look at the Google Play Console.

Sources:
https://support.google.com/googleplay/android-developer/answer/6112435?hl=en

# A Brief Technical Look

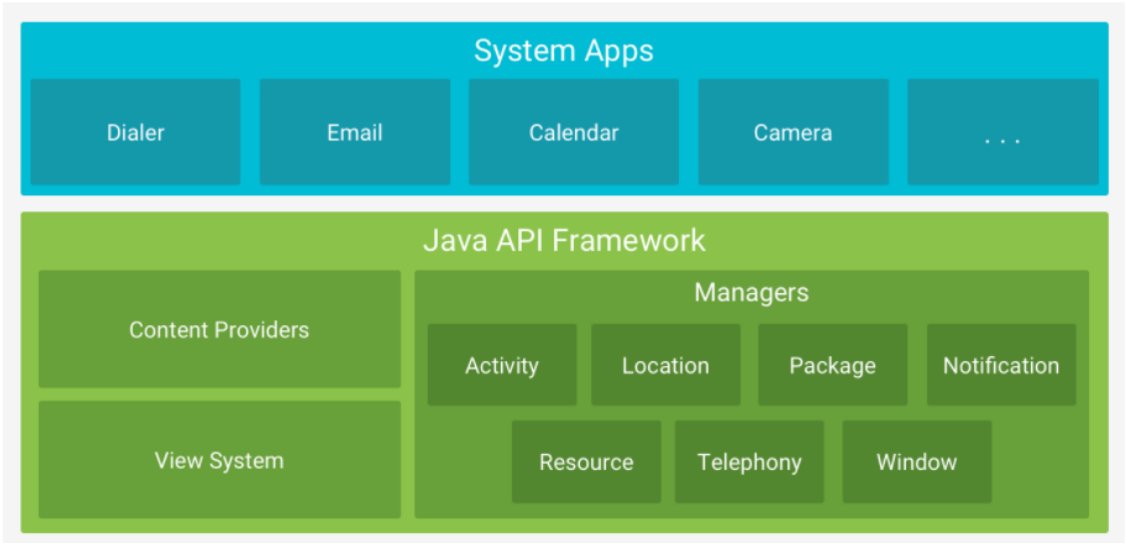
**Android Studio? What is that?**

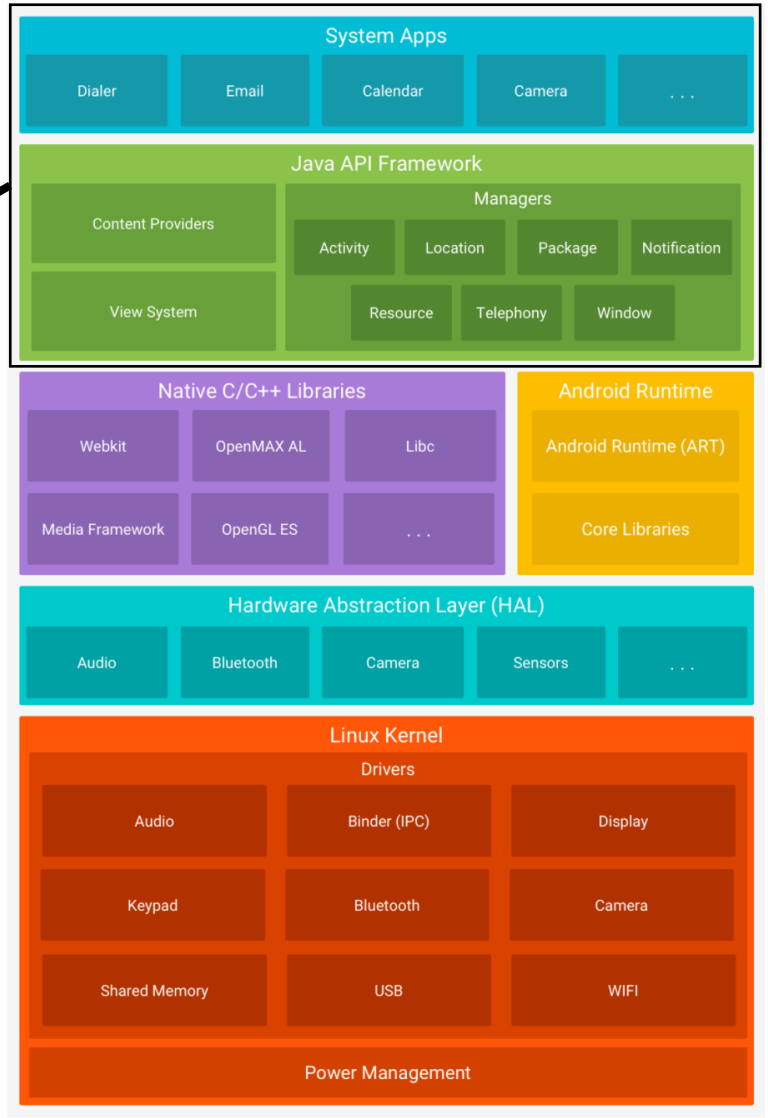
- Android Studio is an integrated development environment for Android mobile application development. It includes almost everything you need!
- Android Studio and all Android development tools are free. No catch.
  - Code editor.
  - Visual user interface layout editor.
  - Debugger.
  - Simulator.
  - Build toolchain.
  - Cloud integrations.
- Let's take a quick look at what you can do in Android Studio! We will look at the MPAndroidChart Example.

Sources:
https://developer.android.com/studio/index.html

# A Brief Technical Look

## What does Android look like architecturally?




Sources:
https://developer.android.com/guide/platform/index.html

# A Brief Technical Look

green COW coworking

sme

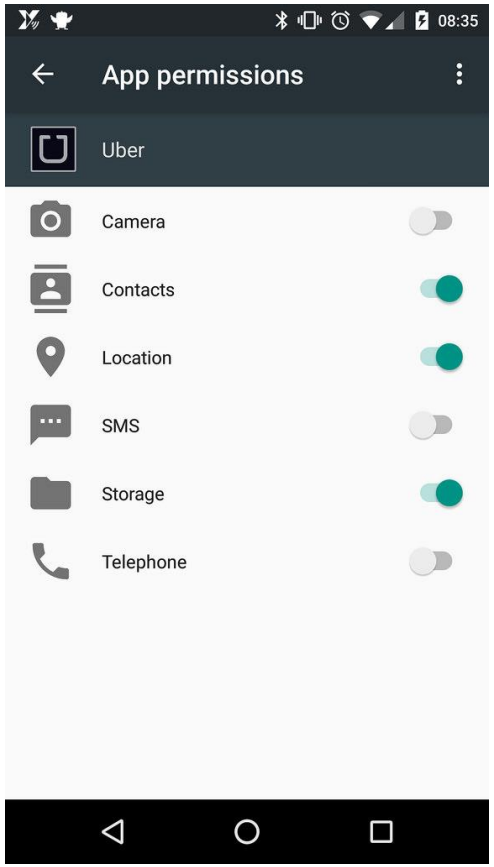
**Great! But how can the Android architecture empower my app?**

- Through APIs! APIs free up your time so that you can add business value to your app instead of the reinventing the wheel.
- Some of the popular APIs are:
  - Camera API
  - Notifications API
  - Speech Input API
  - Location Services API
  - Sensor API
  - Google APIs
- When thinking about the required functionality of your app, remember to consider the API level that you targeting.

Sources:
https://developer.android.com/develop/index.html
https://en.wikipedia.org/wiki/Android_version_history#Version_history_by_API_level

# A Brief Technical Look

**What about the sensitive data on a user's mobile device? How is that protected?**




Sources:
https://c1.staticflickr.com/6/5749/22363919146_e2aa59b953_b.jpg

# A Brief Technical Look

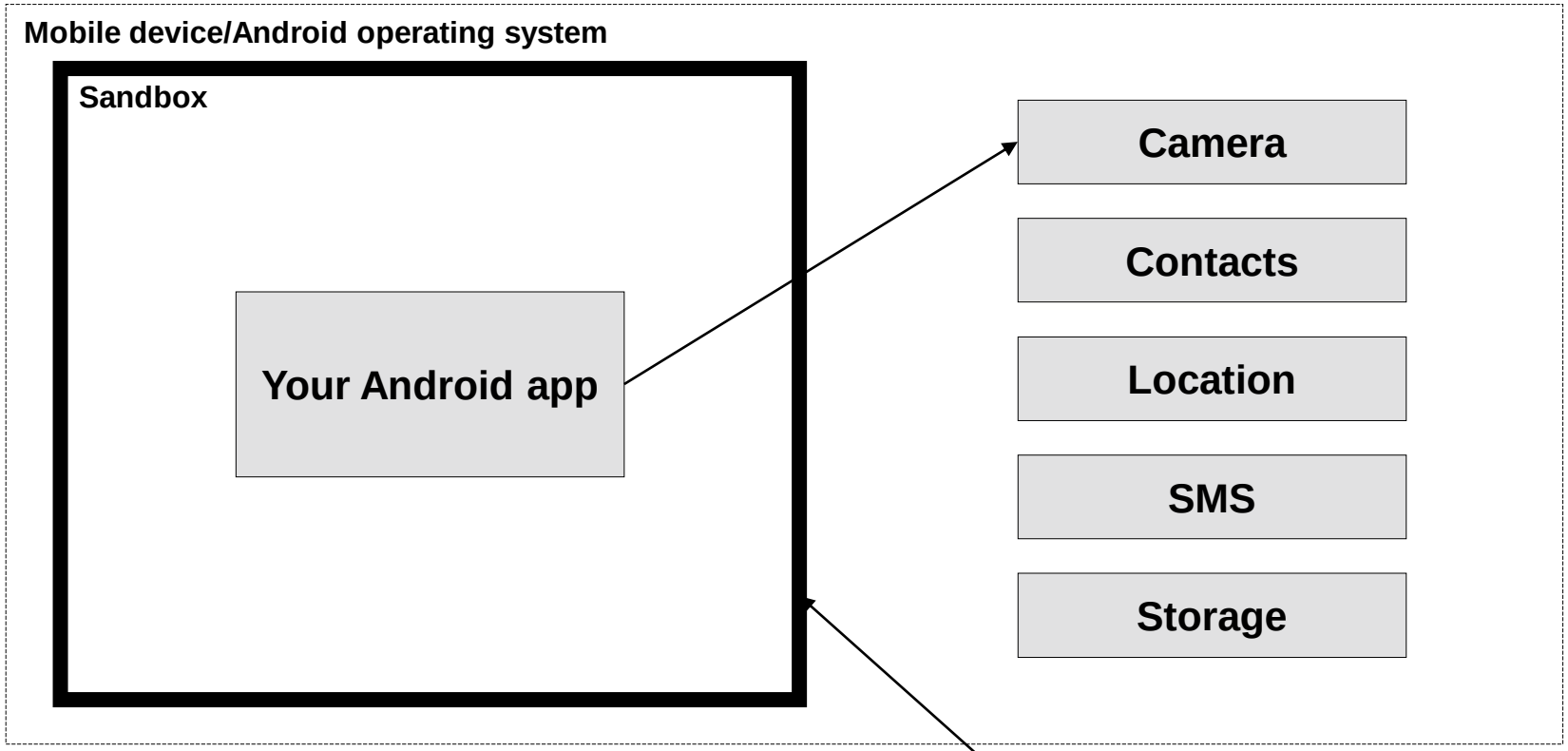
**How do System Permissions work?**

- All Android apps run in a sandbox.
- To access information outside of this sandbox (protected APIs), you need to explicitly declare what you need access to.
- The user, generally, before installing your app, has the option to allow or deny all of your requests or some of your requests.
- You can also ask the user for permission inside of the app for some specific feature but this is **not** recommended.
- Code Hint! The permissions that you app needs must be declared in the **`AndroidManifest.xml`** file in the root directory of your app source code.
- **One important rule of thumb – keep permissions use to a minimum.**

Sources:
https://developer.android.com/training/permissions/index.html
https://developer.android.com/guide/topics/manifest/manifest-intro.html#perms

# A Brief Technical Look

## Can you show me visually how this “sandbox” works?




Sources:
https://developer.android.com/training/permissions/index.html

Your app cannot get past this “wall” without asking the user for permission.

green COW coworking

sme

## Breakpoint!

## Would you rather try and work through an app tutorial or continue with the technical discussion?

# Diving Deeper



**What are the fundamental parts of an Android application?**

- App components. These are entry points through which the system or the user can enter your app.
- There are four different types:
  - Activities.
  - Services.
  - Content providers (we will not discuss this that much).
  - Broadcast receivers (we will not discuss this one in detail, either).
- Let's try to look at each of these app component types individually.

Sources:
https://developer.android.com/guide/components/fundamentals.html#Components

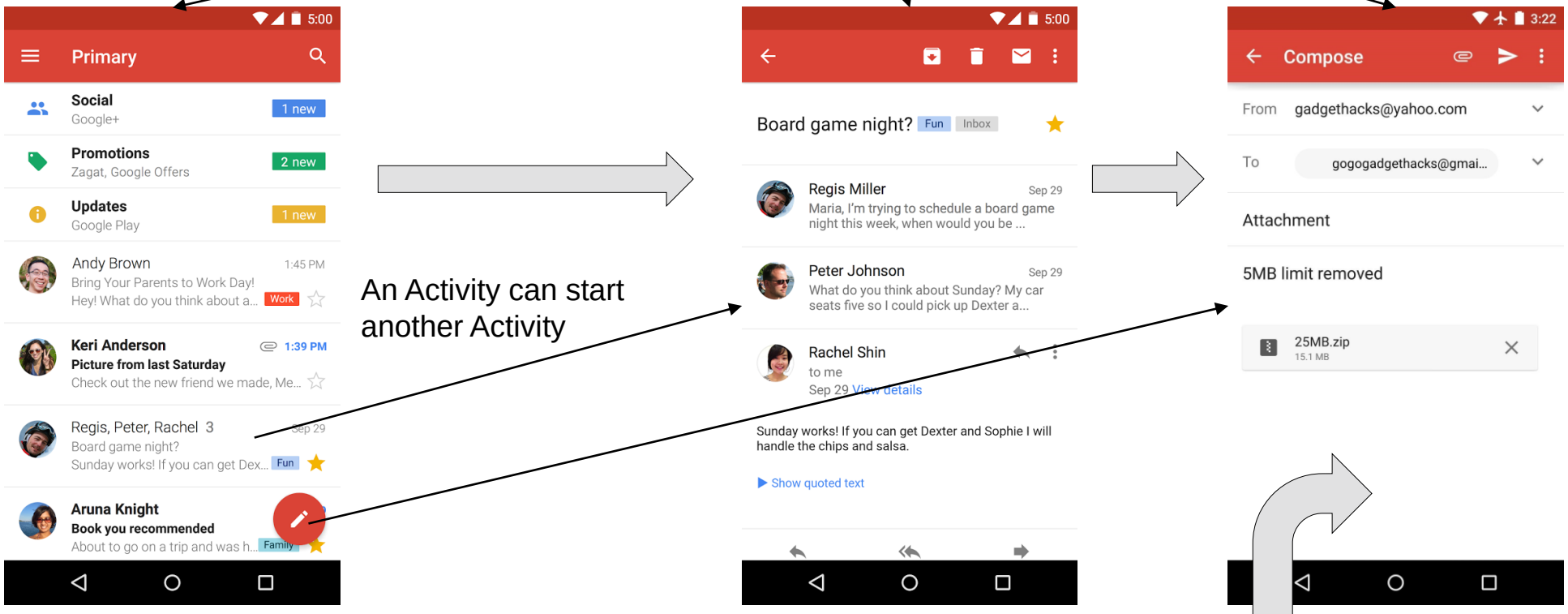
# Diving Deeper

## What is an Activity?

Each of these is a separate Activity – one app can have multiple Activities

**An Activity is an entry point for interacting with the user. The UI.**

**GMail Android app**




An Activity can start another Activity

"Gallery" app can find entry into this Activity, for example

# Diving Deeper



**What is a Service?**

- An entry point for keeping an app running in the background.
- Does not provide a user interface.
- Why would this even be useful? Several possible reasons:
  - Long running network requests.
  - Intensive computations.
  - Play music.

# Diving Deeper



**So that's it? Nope. There is so much more. Explore the following important topics after some coding.**

- **Intents – messaging object which can start an Activity or start a Service**
- **Fragments – a common, resuable portion of an Activity to use across multiple Activities**
- **App Resources – images and (possibly localized) text for your app**
- App Manifest – much of the actual content of the manifest is handled for you by Android Studio, but its still helpful to know what is going on in there.
- Activity Lifecycle
- Loaders – for "asynchronous" loading of remote resources.

# Diving Deeper



**Anything else? Yep! Get even more familiar with Java and Android and then look into these topics.**

- Building your app with Gradle
- Debugging your app
- **Testing your app**
- Profiling your app
- Publishing your app
- Debugging, testing and profiling can be tough topics. **Practice!** In time, you will get better.

# Try-it-yourself!

green COW coworking

sme

**If you want, try and follow the “Building Your First App” tutorial on the Android Developer site.**

**The tutorial is here: https://goo.gl/WhGMdo**

**If you need help, I will walk around to assist you for the remaining time of the event.**

# Learning Resources

green COW coworking

sme

- **Book - Android Programming: The Big Nerd Ranch Guide (3rd Edition)**
- **Web – Guide to App Architecture**
- **Web – Android Best Practices**
- **Web – Google Samples**
- **Book - Java: A Beginner's Guide (6th Edition)**
- Video Course – Developing Android Apps by Google (free)
- Web - Stack Overflow Android
- Web - Pro Git (free online book)
- Web - Getting Started on Android Developer site
- Web - Awesome Python on GitHub

# Popular Third-party Components



- Litho - a declarative UI framework for Android
- Unity 3D game engine (Android backend)
- React Native - a framework for building native apps with React
- MPAndroidChart - a visually-appealing and powerful charting/graphing library
- RxJava - reactive extensions for the JVM

# Thank you!

green COW coworking

sme

**Any questions?**

**Any suggestions on future events or more targeted Android events?**

**The slide deck and code for this event can be found here for future reference:** https://goo.gl/IYVxsy

# Thank you!

**Thanks for attending!**

**Special thanks to our hosting partners – greenCOW Coworking.**


**Adam J. Cook**

Chief Technical Officer of Alliedstrand

Chair of SME Chapter 112

adam.j.cook@alliedstrand.com

https://twitter.com/AdamJosephCook

https://linkedin.com/in/adam-j-cook

https://github.com/adamjcook


**SME**

www.sme.org

https://facebook.com/SMEmfg

https://twitter.com/SME_MFG

https://linkedin.com/company/sme

**SME Chapter 112**

Serving Northwest Indiana and Chicagoland

https://twitter.com/sme112

https://facebook.com/sme112

**https://github.com/sme112**