So Many Things to Explore in Android

As we near the end of this course, we sincerely hope you've enjoyed this wild, crazy ride in app development with us. You're probably already noticing that we've just covering the tip of the iceberg.

Where to go next

Android Basics: Multi Screen Apps is currently our next Android Course (and the next course in the **Android Basics Nanodegree by Google**). This course covers how to build more complex android apps and also teaches more object-oriented programming concepts.

Another resource is the **Java Language tutorial** which we've referenced throughout this course. Also, share you tips for good introductory and intermediate Java material **in the forums**.

You can find the final code for Just Java on **this Github web-page**. **Github** is a website that facilitates sharing code online. Code on Github is organized via the **Git** version control system. To learn more about Git and Github, consider taking our **version control course**, which is another good introductory offering.

Advanced Resources and The Possibilities

In addition to all that we'll cover here, if you're wondering what other *possibilities* are for Android Apps, there is a lot of material explore. Much of it will require knowledge from the intermediate Developing Android Apps course, so these are considered pretty advanced resources.

What follow are some ideas for deeper dives:

Build for tablets - We've be focusing on designs for phones, but with resource files, you can actually create custom layouts for your larger devices, such as tablets. Here's some **documentation** to get you started.

Do background operations - Perhaps you want to create an app that does something, even when it's not visibly on the screen. For example, a music player application, that plays music as you so other things on your phone. Or a messaging application that pulls down the latest chats sent to you. For this, you'll need to learn more about **creating services.**

Store data on the device - Does your app need any information saved between launches? Maybe you want to keep a history of all coffees bought with JustJava. To do this, you would need to save some data associated with the app. Check out the **google documentation** on data storage for your options.

Create lists - Many android apps that have long lists of items a user could click on - think of messages in an inbox or stories in a news feed. Learn to optimize the creation of these lists by using views such as the **RecyclerView**.

Play sound - Add sounds to your apps, whether is be short clips for a game or playback capabilities for a podcast app. Check out Google's guide to **Media Playback**.

Load up things from the internet - Want to load text or images from the internet? Check out the **volley library**, for general purpose loading of web data. **Glide** is a great choice for loading

images from the web. You can check out **this lesson** from the Libraries lesson of Advanced Android Development to see how to incorporate libraries like these.

Persist data - The ability to store and retrieve data is a core function of any mobile application. Android Framework provides several **data storage options**. In particular, learn how to leverage the SQLite Database by learning the SQL programming language and table design.

Build for tablets - We've be focusing on designs for phones, but with resource files, you can actually create custom layouts for your larger devices, such as tablets. Here's some **documentation** to get you started.

Create cards - Cards, not to be confused with your awesome Birthday Card, are a special kind of Material Design inspired view that has rounded corners and a slight shadow. If you've ever used Google Now, you've seen a card. Check out this **documentation** to see how to add card elements to your app.

Post notifications - **Notifications** are messages that the user sees outside of your app. They appear in the status bar. You can then pull the status bar down to see more details about the notification. When you get a text message, for example, many times your phone will beep and show you a notification so that you can easily view the text. To learn more about designing notifications, check out this **google guide**.

Use Google Play services - Using Google Play Services is a library of Google code that gives you access to popular functionality, such as phone location, authentication (information and ability to “log in") and even fitness data. Check out some of the recent updates **here**.

Use location - Access to location is one of the many capabilities offered by Google Play services. By figuring out where a user is physically located, your app can give them more data about their surrounding and customize itself to their specific needs. Google Maps, for example, can find nearby restaurants in this manner. Check out our class on **Location and Context**, which is part of a larger series of mini courses on all of the Google Play Services.

Add analytics - Ever wondered who is using your app? Or how they are using it? By collecting this data you can make informed decisions about what features to add, update or fix. Analytics is also included in the Google Play Services libraries. Check out our course on **Analytics**.

Build for other form factors - It doesn't stop at tablets; the android operating system (and therefore your apps) run on watches, TVs, even in cars. If it makes sense, you can have different versions of your app for these different “form factors". The Maps App, for example, can give you directions on your phone, or your watch, or while you're driving around. The base functionality is the same, but it's been re-designed and tweaked for each interface. Check out our **Ubiquitious Design** course.

Animations - Subtle animations that give your views a real sense of space and physical existence are an integral part of material design. Check out the **documentation** on animations to add things like cross fading, zooming and flipping animations to your layouts.

Create server backend for your app - A stand alone app is great, but what makes an app really powerful is if it is connected to your own web server. Web servers can help organize and process data from across the world and send it to your user's phone. Any Google App more complicated than the alarm is accessing some sort of google server. Servers store everything from your Calendar

events, to your friend's G+ photos. Learn more about building a scalable web server in **Java** or **Python**.

Fitness - Ever wanted to make an app that will help users take control of their health and fitness? Google Fit is google's open platform with user fitness data. Check out the **documentation**.