The X Course: Android

Session 7

Agenda

- Android SDK Versions and Compatibility.
- Data Storage in Android.
- How to go to Infinity and Beyond from here ?

Android SDK Versions and Compatibility

- Android API levels, firmware versions can be found here: https://source.android.com/setup/start/build-numbers
- Distribution and the percentage of devices using each version can be found here: https://developer.android.com/about/dashboards/index.html

Android SDK Versions and Compatibility

- To reach a broad market, Android developers must create apps that perform well on devices running most of the Android versions.
- Phone screens are a variety of sizes,but the Android layout system does a good job at adapting.
- Tablets require more work, but there are techniques to do it.
- However, for Android TV and Android Wear devices, the differences in UI are large enough that you need to rethink the design of your app.

Android SDK Versions and Compatibility

- When you created the GeoQuiz project, you set a minimum SDK version within the New Application wizard.
- In addition to the minimum supported version, you can also set the target version and the build version.
- Open the build.gradle file that exists in your app module. Notice the values for compileSdkVersion , minSdkVersion , and targetSdkVersion .

Minimum SDK version

- **It is a hard floor below which the OS should refuse to install the app.**

  By setting this version to API level 16 (Jelly Bean), you give Android permission to install GeoQuiz on devices running Jelly Bean or higher. Android will refuse to install GeoQuiz on a device running, say, Froyo.

- This value should be a **SANE** minimum.

Target SDK version

- **This value tells Android which API level your app was designed to run on.**
- Most often this will be the **latest Android release.**

  If you have already designed an app, you should confirm that it works as expected on new releases. See where problems might arise.
  Then you can modify your app to work with the new behavior or lower the target SDK.

Compile SDK version

- It is a private information between you and the compiler.
- **The compile SDK version, or build target, specifies which version to use when building your own code.**
- The best choice for a build target is the **latest API level.**
- However, you can change it for an existing application if you need to. For example, you might want to update the build target when yet another version of Android is released so that you can make use of the new methods and classes introduced in that version of Android.

Adding code from later APIs safely

- The difference between GeoQuiz's minimum SDK version and build SDK version leaves you with a compatibility gap to manage.

  What happens if you call code from an SDK version that is later than the minimum SDK of Jelly Bean (API level 16)?

  - *When your app is installed and run on a Jelly Bean device,* ***it will crash.***

Android Lint to the rescue !

- Potential problems caused by calling newer code on older devices can be caught at compile time.
- If you use code from a higher version than your minimum SDK, Android Lint will report build errors.
- If you do not see a warning, you can manually trigger Lint by selecting Analyze → Inspect Code.

How do you get rid of these errors?

- One option is to raise the minimum SDK version to 21. However, raising the minimum SDK version is not really dealing with this compatibility problem.
- A better option is to wrap the higher API code in a conditional statement that checks the device's version of Android.

  if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.LOLLIPOP) { }

- The Build.VERSION.SDK_INT is the device's version of Android.

Data Storage in Android

Internal File Storage	External File Storage	Shared Preferences	Databases
Store app-private files on the device file system.	Store files on the shared external file system. For shared user files.	Store private primitive data in key-value pairs.	Store structured data in a private database.

Internal Storage

- Files that you create on internal storage are accessible only to your app. The system provides a private directory on the file system for each app where you can organize any files your app needs.
- If you want to share your data with other app processes, consider using a content provider, which offers read and write permissions to other apps and can make dynamic permission grants on a case-by-case basis.
- You should not use internal storage to save anything the user expects to persist independently of your app.

External Storage

- Files created on external storage, such as SD cards, are globally readable and writable.
- External storage can be removed by the user and also modified by any application, don't store sensitive information using external storage.
- Code and More reading in: https://developer.android.com/training/data-storage/files.html#WriteExternalStorage

External Storage

- Before you attempt to access a file in external storage in your app, you should check for the availability of the external storage directories as well as the files you are trying to access.
- You should use external storage for user data that should be accessible to other apps and saved even if the user uninstalls your app, such as captured photos or downloaded files.

Shared Preferences

- The SharedPreferences APIs allow you to read and write persistent key-value pairs of primitive data types: booleans, floats, ints, longs, and strings.
- The key-value pairs are written to XML files that persist across user sessions, even if your app is killed.
- Code and More Reading in: https://developer.android.com/training/data-storage/shared-preferences.html

Databases: SQLite

- SQLite is an open source relational database, like MySQL or Postgresql.
- Data is in the form of Structured Tables and interacted with SQL queries or using the SQlite library methods.
- Unlike other databases, though, SQLite stores its data in simple files, which you can read and write using the SQLite library.
- Android includes this SQLite library in its standard library, along with some additional Java helper classes.

Databases: SQLite

- Code and More Reading in: https://developer.android.com/training/data-storage/sqlite.html
- The SQLite APIs are fairly low-level and require a great deal of time and effort to use.
- There is no compile-time verification of raw SQL queries.
- You need to write lots of boilerplate code to convert between SQL queries and Java data objects.

Databases: Room

- Instead of using SQLite APIs directly, create and interact with databases with the Room persistence library.
- The Room library provides an object-mapping abstraction layer that allows fluent database access while harnessing the full power of SQLite.
- Code and More Reading in: https://developer.android.com/training/data-storage/room/index.html

Now, I declare you Android developers !

How to go to Infinity and Beyond from here ?

- **Idea #1:** Complete the Android Programming:Big Nerd Ranch Guide !
  - We covered most of Chapters 1,2,3,5,6,8,9.
  - There are 34 Chapters in this book !
  - And a lot more of applications to be done inside.

How to go to Infinity and Beyond from here ?

- **Idea #2:** Build Small Simple Applications step by step and upgrade your level each period of time. Some ideas I did back when i started:
  - Notepad App
  - Todo List App
  - Weather App
  - Calculator App

How to go to Infinity and Beyond from here ?

- **Idea #3:** Learn more and more always. Some advanced topics you will need to know at some point when you're ready.
  - Architectures in Android: MVP and MVVM.
  - Dependency Injection.
  - Kotlin Programming.
  - Background Work and Reactive Programming.

How to go to Infinity and Beyond from here ?

- **Idea #4:** This was a crash introductory course. How about a real 100 hours course ?
  - The Complete Android N Developer Course
  - Android's Udacity Nano Degree
  - Programming Mobile Applications for Android Handheld Systems: 2 Parts Course
  - And there a lot, lot more.

How to go to Infinity and Beyond from here ?

- **Last Advices:**
  - Whatever you do, waste no time: write code.
  - Always keep the learning spirit on.
  - Meet people and make connections.
  - Explore the open source community.
  - Stay in Contact with me, whatever you need just ask, I can always try to help.