What's your favorite tool or library for Android? Why is it so useful?

I am of course a fan of some rather common tools/libraries, like the constraint-layout and recyclerview tools, and various firebase libraries. But my favorite right now is Amit Shekhar's Debug-DB tool. It is a simple tool which allows a developer to examine the actual database and shared preferences an app uses. This has been INCREDIBLY useful to me in my SQLite coding. By being able to actually see what's in the database, I am much more quickly able to pin down errors and bugs. And it's so easy to use: include a debugCompile line in gradle build script, and it works. A link is sent out to the log as soon as the app starts, and clicking the link opens a page that shows all databases and shared pref files. This is my favorite tool right now, because I just finished setting up a couple small databases in my Ms. Count metronome.

You want to open a map app from an app that you're building. The address, city, state, and ZIP code are provided by the user. What steps are involved in sending that data to a map app?

First you make a Uri of the address using a straight forward "geo:" query. That Uri is used to create an ACTION_VIEW intent. Using setPackage, you can ensure that Google Maps handles the intent, or just leave it, and let the system suggest possibilities to the user. Then you start the activity.

Implement a method to perform basic string compression using the counts of repeated characters.

```
public static String compress(String input) {
    int length = input.length();
    int charCount = 1;
    char lastChar = input.charAt(0);
    StringBuilder builder = new StringBuilder();

    for(int i = 1; i < length; i++) {
        char current = input.charAt(i);
        if(current == lastChar) {
            charCount++;
        } else {
            builder.append(lastChar);
            builder.append(charCount);
            charCount = 1;
        }
        lastChar = current;
    }
    builder.append(lastChar);
    builder.append(charCount);

    String output = builder.toString();
    return output.length() < length ? output : input;
}
```

List and explain the differences between four different options you have for saving data while making an Android app. Pick one, and explain (without code) how you would implement it.

The options are: SharedPreferences, local (internal or external) memory, SQLite database(s), and networked data storage (various possibilities, including Firebase).

SharedPreferences allow you to store primitive datatypes and strings in simple key-value pairs. Local memory allows you to store all sorts of files and file types. SQLite is a straight forward database, with columns for each characteristic of the data, and a row for each instance of that data. Great for saving lots of a particular thing and having fast searchability. Online or cloud storage can be many things. Firebase is a great, easy system, and allows a user to store any Object that can be created in Java.

SharedPreferences are probably the most common, and also the simplest approach to saving data. Data is saved by acquiring an instance of SharedPreferences.Editor, 'putting' key-value pairs into the editor, and finally 'applying' or 'commiting' the preferences. Retrieval is very much the same, though you start by acquiring an instance of SharedPreferences (not an Editor). Individual pieces of data are retrieved via their keys. (A default value must be provided, in case the key is not present.)

SharedPreferences are great for storing bits and pieces of information that might be needed across various activities, such as a user's name; or various Settings options that an app might use. They are also good for saving something similar to an instance state across longer term, like between restarts of an app.

What are your thoughts about Fragments? Do you like or hate them? Why?

I have an up and down relationship with Fragments. When I am reading about organizing my app, about Android best practices, etc., Fragments make a great deal of sense. They are good for keeping a larger app organized around the different things it might do. I like Fragments for this.

And then, during actual implementation, they are a bit frustrating. Numerous things appear to become more difficult, or at least more cumbersome, than it feels like they should need to be; particularly transitions! Passing data around is tricky, but then I usually figure something out that makes it seem okay in the end afterall. One can never, of course, get away from the fact that with Fragments, you have to manage both sets of lifecycles, in the Activity and the Fragment. That is confusing at times.

My biggest thought about Fragments is that I would like to have the opportunity to build an app using Fragments, together with someone who REALLY understands Fragments.

If you were to start your Android position today, what would be your goals a year from now?

My primary goal a year from now is to have my family in a place where we are living happily, not fretting over health insurance, and not living paycheck to paycheck. Professionally, my goals include having gained enough varied experience to be confident when assigned a project, that I will be able to pull it together quickly, efficiently, and elegantly. I expect to still be challenged by my work, but to have a strong handle on the numerous resources available to be able to find my own answers, learn my own lessons, and move on to the next piece of the project's puzzle. In less than a year, I hope to be fulfilling

my technical/administrative responsibilities smoothly enough, that I am able to seamlessly draw on my many years of education experience to improve the project, and support the overall mission in ways only a programmer with my background possibly can.