

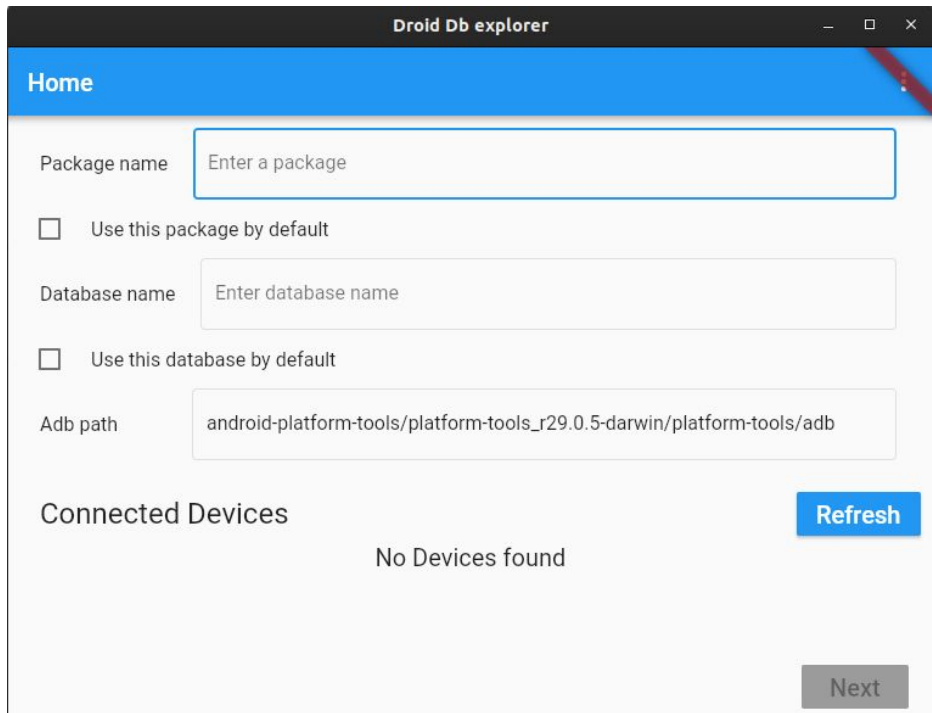
# Gql Testing tool

Desktop app - by Rahul Lohra

# Features

1. You can **add/edit** custom responses for any url
2. This application supports both **GQL** and **REST**
3. For developers - They don't have to depend on backend or manually write custom responses to see their behaviour
4. For **QA** or **PO** they can use this desktop app to see different UI behaviours based on different responses
5. This will only work for **debug apps and not on release/ playstore apps**

# How to use



The screenshot shows the 'Droid Db explorer' application window. It has a blue header bar with the text 'Home'. Below the header, there are three input fields: 'Package name' with a placeholder 'Enter a package', 'Database name' with a placeholder 'Enter database name', and 'Adb path' with the value 'android-platform-tools/platform-tools\_r29.0.5-darwin/platform-tools/adb'. Each input field has a checkbox to its left labeled 'Use this [package/database] by default'. Below these fields, there is a section titled 'Connected Devices' which currently displays 'No Devices found'. To the right of this section is a blue 'Refresh' button. At the bottom right of the window is a grey 'Next' button.

Droid Db explorer

Home

Package name Enter a package

☐ Use this package by default

Database name Enter database name

☐ Use this database by default

Adb path android-platform-tools/platform-tools\_r29.0.5-darwin/platform-tools/adb

Connected Devices

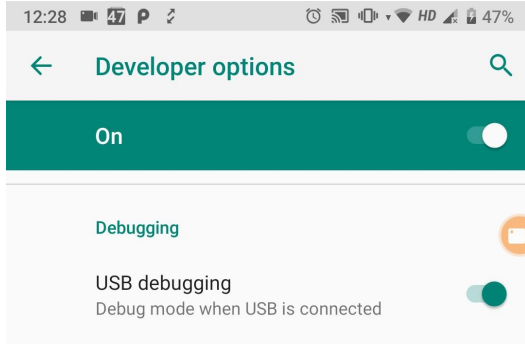
No Devices found

Refresh

Next

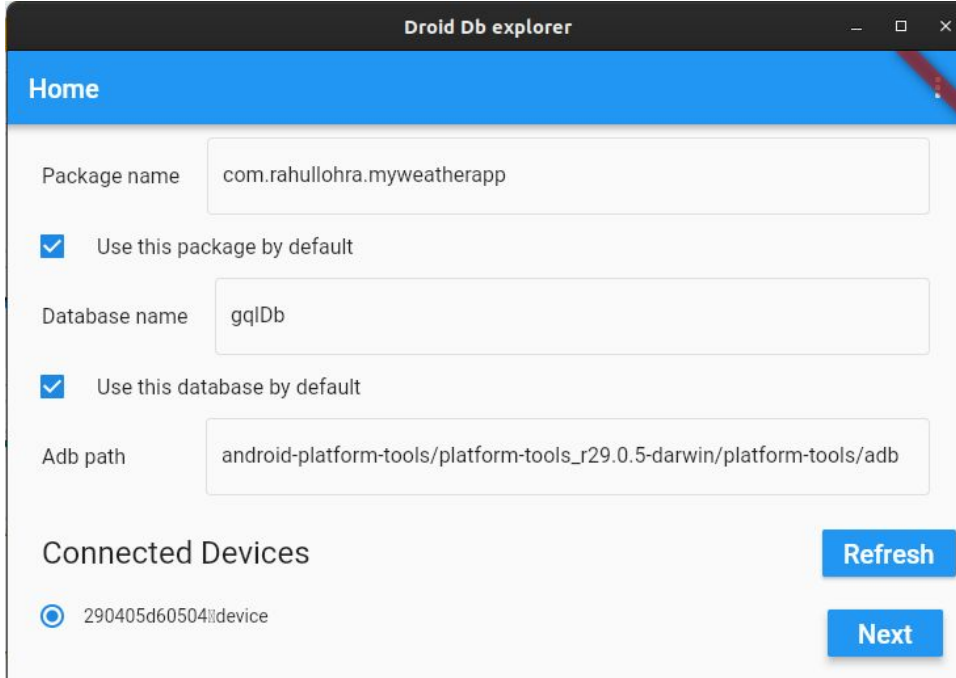
1. Open the app
2. You will see 3 fields
  - a. **Package name** - The name of the app you want to debug
  - b. **Database name** - The database where fake responses are stored
  - c. **Adb path**
3. List of **connected devices** - devices attached with usb and with debugging enabled
4. **Refresh** - will refresh list of connected devices
5. **Next** - Once you have selected the device, it will be enabled

# How to connect your device



1. Enable developer options in Android
  - a. Follow the link to see the steps
    - i. <https://drive.google.com/open?id=1wq5lNhgi6lYBjrr9mUprBfQVeGPRVK7J>
  - b. These two options must be enabled for the app to work

# After filling the details and connecting devices, ui should look like this



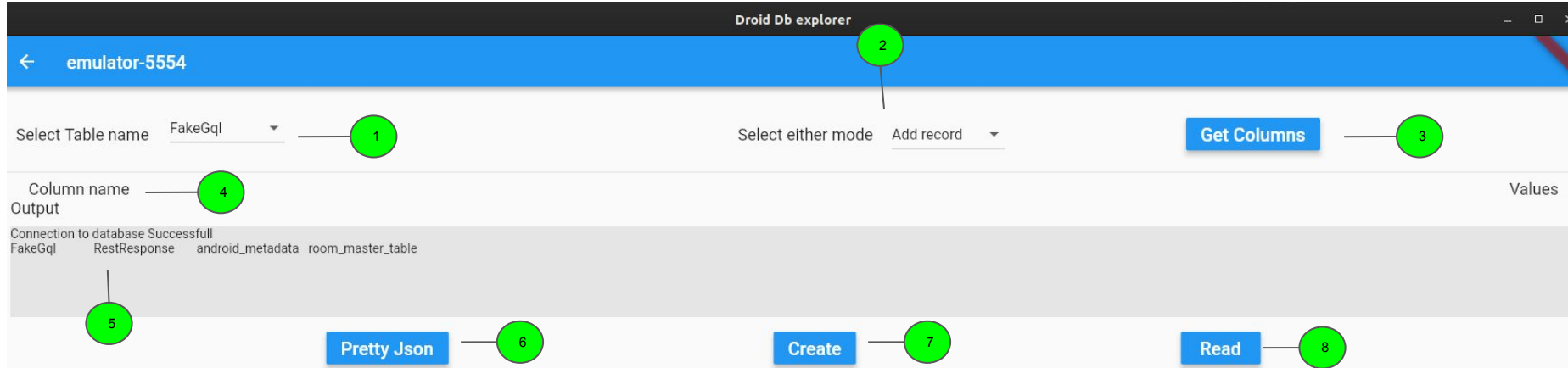
The screenshot shows the 'Droid Db explorer' application window. It has a blue header bar with the word 'Home'. Below the header, there are four input fields with labels and checkboxes:

- Package name:** com.rahullohra.myweatherapp. Below it is a checked checkbox labeled 'Use this package by default'.
- Database name:** gqlDb. Below it is a checked checkbox labeled 'Use this database by default'.
- Adb path:** android-platform-tools/platform-tools\_r29.0.5-darwin/platform-tools/adb.

At the bottom, there is a section titled 'Connected Devices'. It contains a single entry: a blue circle icon followed by the text '290405d60504@device'. To the right of this section are two blue buttons: 'Refresh' and 'Next'.

1. Select one of the connected devices and tap on **NEXT**
2. If **Connected Devices** are **empty** then please see previous page
3. Your device must be in **DEBUGGABLE MODE**

# Enter device detail screen



1. Drop down menu to select table name available in previously selected database
2. Two modes are available
  - a. Add new record
  - b. Update old record
3. Will show you all the columns for this table
4. Here we will show you the columns
5. Output window - to check for any errors
6. Will pretify the json
7. Will Execute command to create a new record
8. Will read all the data from the table name provided and will show in output window

# View all columns (Add MODE)

emulator-5554

Select Table name: FakeGql

Select either mode: Add record

Get Columns

Column name: gqlOperationName, response

Values: Enter value, Enter value

Output: Connection to database Successfull, FakeGql RestResponse android\_metadata room\_master\_table, CREATE TABLE 'FakeGql' ('id' INTEGER PRIMARY KEY AUTOINCREMENT, 'response' TEXT, 'createdAt' INTEGER NOT NULL, 'updatedAt' INTEGER NOT NULL, 'enabled' INTEGER NOT NULL, 'gqlOperationName' TEXT NOT NULL, 'javaQueryName' TEXT, 'customTag' TEXT);

Pretty Json, Create, Read

- Follow the steps in the video, to see all the columns and you can then enter the data in it
  - a. <https://drive.google.com/open?id=1umQwcwFCDDcM0gzLwRyS9-d2unnbevZ>

- These are the column names
- These are the values for respective columns
- Columns are coming of this table
- Currently it is **Add Record** mode

*ADD MODE means - You are going to insert new record*

# View all columns (UPDATE MODE)

emulator-5554

Select Table name: FakeGql

Select either mode: Update record

Get Columns

Column name

Enter values with where Query

Fill old data

Values

gqlOperationName

customTag

Enter value

Enter value

Enter values to be added (Below layout is scrollable)

gqlOperationName

response

Enter value

Enter value

Output

Connection to database Successful

FakeGql RestResponse android\_metadata room\_master\_table

CREATE TABLE 'FakeGql' ('id' INTEGER PRIMARY KEY AUTOINCREMENT, 'response' TEXT, 'createdAt' INTEGER NOT NULL, 'updatedAt' INTEGER NOT NULL, 'enabled' INTEGER NOT NULL, 'gqlOperationName' TEXT NOT NULL, 'javaQueryName' TEXT, 'customTag' TEXT);

Pretty Json

Update

Read

- Follow the steps in the video, to see all the columns and you can then enter the data in it
  - [https://drive.google.com/open?id=1M1WPkKqHLrRJ\\_tuTxvx8f4rkh7kx0rOj](https://drive.google.com/open?id=1M1WPkKqHLrRJ_tuTxvx8f4rkh7kx0rOj)

- Only mode is changed to Update record
- You can see few new columns, these are columns where the values will be updated
- This will **Fill old data** on below columns. But you need need to **provide it some values** via filling the values section in the very first 2 rows

*Update MODE means - You are going to update old record*



# Add new record (*Create your own response*)

1. Follow the steps to see how you can add your own new response
  - a. [https://drive.google.com/open?id=1tN2QO\\_z\\_Bb2x4T0awRnl2gB5Rm25kUp7](https://drive.google.com/open?id=1tN2QO_z_Bb2x4T0awRnl2gB5Rm25kUp7)

## Summary of video

1. We have an Android App named Weather app, it is pulling data from its respective backend
2. We will change the response from the desktop app. Inside the app you can also see the no **fake-response** are stored yet.
3. We have filled the required the data in the desktop app and we have change only the temperature from **31 to 61**.
4. We pressed the **CREATE** button - to inject new **fake-response** record
5. We can see in the Weather app that a new **fake-response record is added**
6. Once we re-launch the app you can see the updated temperature

# Update old record (*Edit saved responses*)

Follow the video to see the steps - [https://drive.google.com/open?id=1Q66tdKr2Nq7mU2X\\_8uowM8HMW1oRcjxX](https://drive.google.com/open?id=1Q66tdKr2Nq7mU2X_8uowM8HMW1oRcjxX)

## Summary

1. We have the same weather app, now we want to update the temperature from **61 to 100**
2. In the desktop we have to **update the either mode** as **UPDATE RECORD**
3. Tap on **Get Columns** to see all columns
4. Enter the **url** and tap on **Fill old data (ensure no extra space/line is there)**
5. You can see all the values are automatically filled in the respective places
6. We have changed the temperature from **61 to 100**
7. Tap on **UPDATE**
8. We can see the updated record in the app as-well
9. Now again launch the app you can see the updated value

# Upcoming features

1. Support for complex of queries
2. Allow to add mocked responses from your module's debug raw file
3. User can add logic for getting different mocked responses based on their strategy
  - a. For eg If you hit the API for 1st time then mock response named RESPONSE-1 should come.If you the hit same endpoint 2nd time, then RESPONSE-2 should come
  - b. Developer can also add their custom logic for their specific mock responses
4. Support for sharing the records
5. More elegant UI
6. Deploy it to web

# Feedback

Contribute here - <https://github.com/rahul-lohra/GqlDoctor>

Special thanks to Sandeep Gupta & Lavekush for giving ideas and solving issues