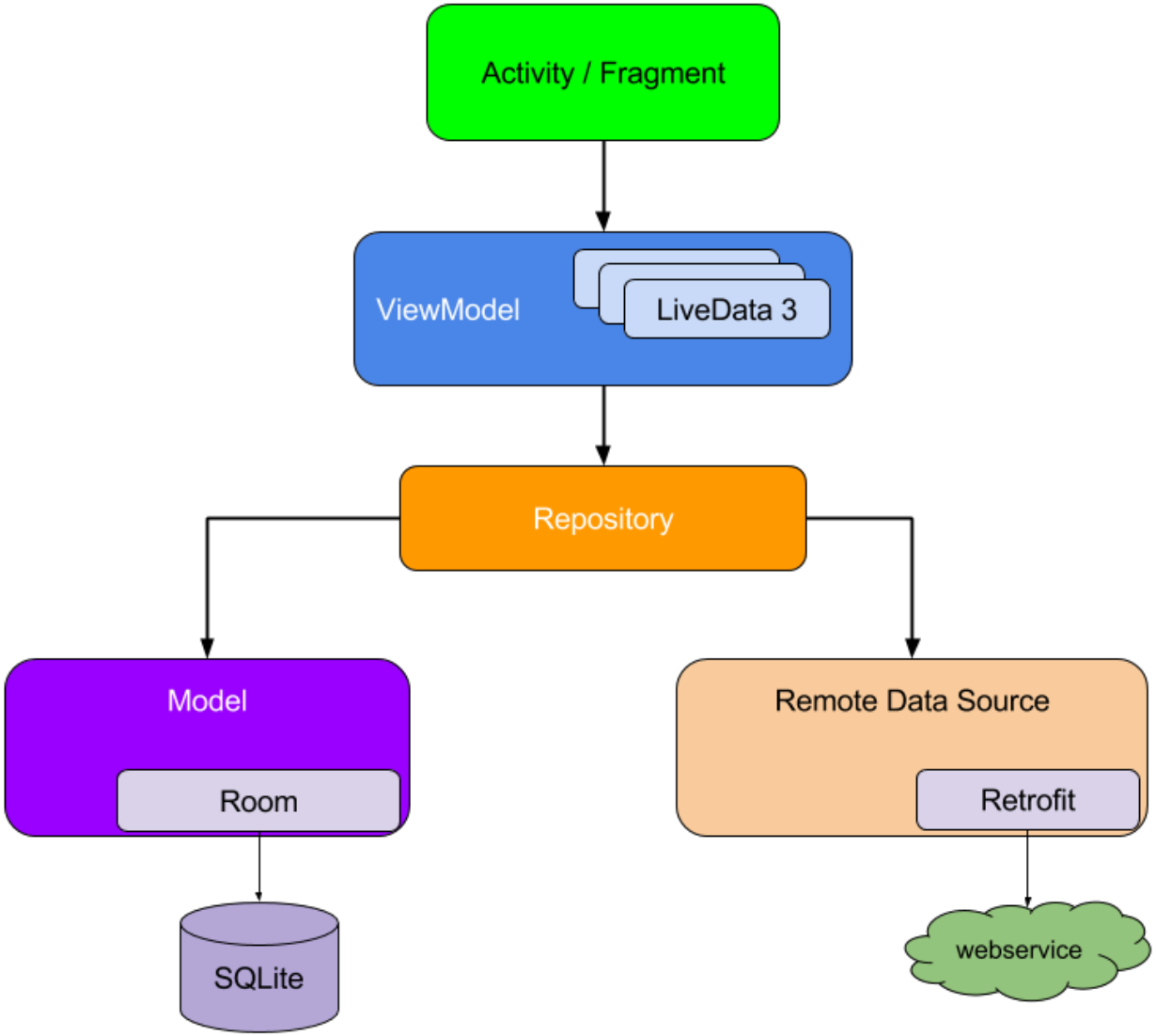
Architecture Components 3: WorkManager

Recap

- We saw the repository pattern in more detail.
- We saw how to add a Room database to our app

Repository Design Pattern



Today

- We’ll see how to use WorkManager to fetch data in a recurring fashion at a specified interval.

WorkManager

WorkManager: Introduction

- Room is awesome for writing SQLite queries, and DB read/writes.
- For your app to be reliable, you need a way to back up the local database on the cloud.
- You also need a reliable way to cache stuff into your DB periodically.
- You also need a way to do all this without draining battery.
- You need a way to keep track of whether you’re connected to the internet or not, and have your app do different things.
- WorkManager is designed to handle these different scenarios.

WorkManager: Introduction

- Under the hood, WorkManager may very well be using JobScheduler (or something else).
- WorkManager looks a lot like Volley. You have:
  - Worker: the task you want to perform.
  - WorkRequest: specify which Worker should run, conditions for the task to be run, whether it is periodic or not.
  - WorkManager: queues and manages work requests. It does magical load balancing and battery management, while honoring your constraints. Like the Volley RequestQueue.
  - WorkStatus: WorkManager provides a LiveData<WorkStatus> for each WorkRequest.

WorkManager: Typical workflow

- Create a new class extending Worker and implement doWork().
- Create either a OneTimeWorkRequest or a PeriodicWorkRequest using that Worker.
- Enqueue the task using WorkManager.
- Track the WorkStatus using an observer, do something on success, failure.
- Other features:
  - You can chain jobs with a line or two of code.
  - Backwards compatible with old SDKs (unlike JobScheduler).
  - Automatic threading.

WorkManager: Google Codelabs “Background Work with WorkManager”

- This codelabs makes you write the code to blur the image of a fish.
- We create the BlurWorker class that extends Worker.
- In the doWork() method, we blur the fish image and save it to a file.
- The BlurViewModel class enqueues the WorkRequest to run BlurWorker when we click the Go button.
- The result is in a file on the device.

Summary

- We saw how to enqueue work using WorkManager.