Testing Our Architecture

Omri Erez

SOFTWARE ENGINEER

@innovationMaze | https://www.linkedin.com/in/omrierez/

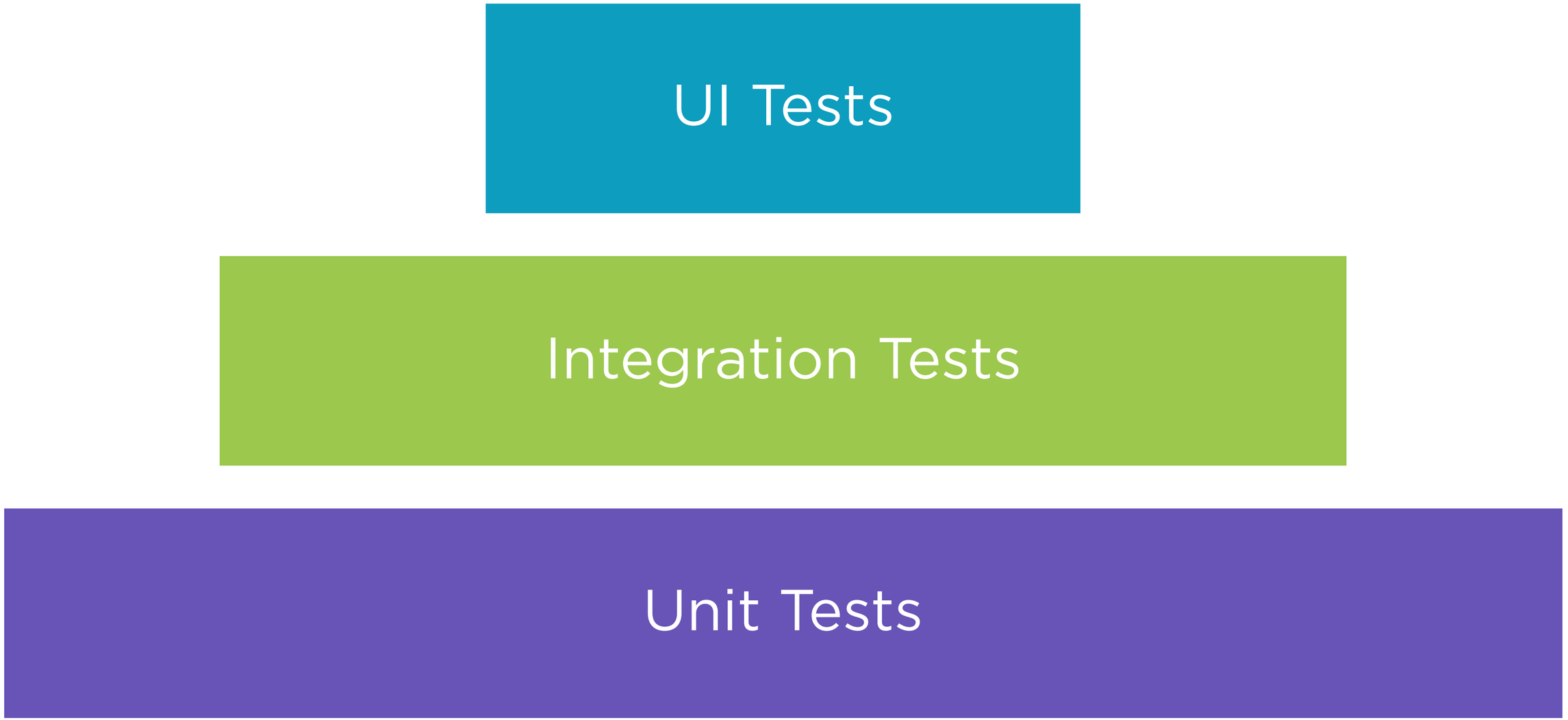
Testable Software

The ability to test each component separately

Low maintenance effort for tests' code

Efficiency in terms of testing effort and code coverage

The Testing Pyramid




SOFTWARE TESTING
SEARCH.....100%

Unit Testing

- Used to test a single component

- Mocking the OS

- Mocking other components

Integration Testing

- Used to test multiple components

- Test the interaction between them

- Usually run on an emulator or real devices

UI Testing

- Used to make sure the UI flow is working as expected

- Makes sure he user sees what he is suppose to see

- Always run on an emulator or real devices

Testing with ARCH

- We can test each component separately

- Our components are small and relatively easy to test

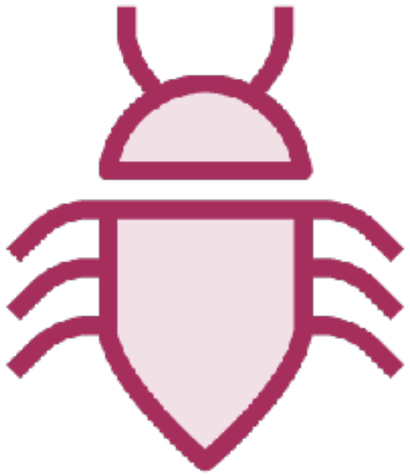
Testing with ARCH

- Our components are modular

- Simple to mock other components if needed

App’s Specific UI Testing

Android UI Testing



- Specific app’s UI testing

- Random UI testing

Test Automation Frameworks

Appium

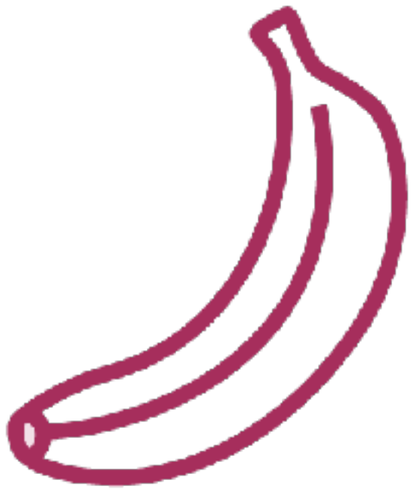
Calabash

Espresso

REFERENCE TO NORA's COURSE

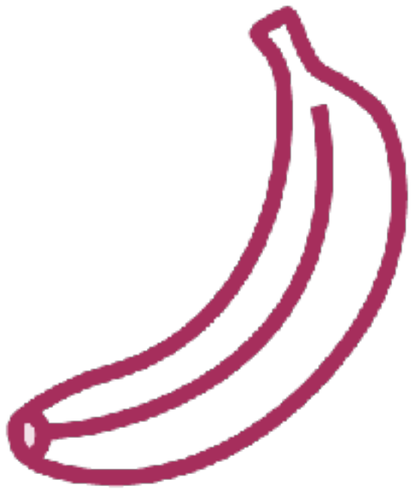
https://app.pluralsight.com/library/courses/android-ui-tests-espresso-fundamentals/table-of-contents

Random UI Testing



- UI/Application Exerciser Monkey:

  - Random UI testing command line tool
  - It can run on the emulator
  - Used to perform stress tests



- And It’s configureable:
  - Number of events
  - Frequency of events
  - Distribution of events
  - And more

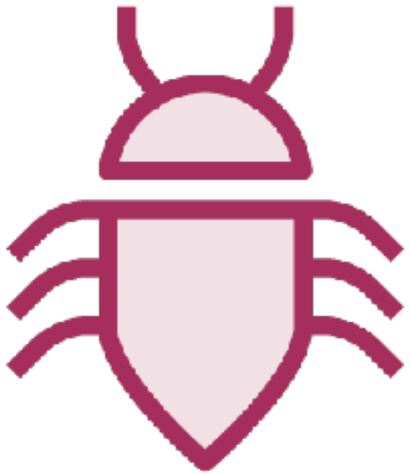
Demo

Crypto Boom App

- Run Exerciser Monkey:
  - Define a number of events
  - Run it
  - Inspect result

exerciser monkey demo

Unit Testing



Android Unit Testing

- JVM unit testing using JUnit:
    - Under “test”

- Android unit testing (Instrumented):
    - Under “androidTest”

Android Unit Testing

- For tests which need interaction with the Android framework:
  - AndroidRunner
  - Robolectric

- For object mocking:
  - Mockito

Demo

Crypto Boom App

- Write an unit for our CryptoMapper class
  - Junit apis
  - Learn about code coverage

Demo

Crypto Boom App

- Write an instrumented unit test for our CoinDao class
  - Mockito
  - Junit apis
  - Our emulator

Summary

Our Architecture Is Convenient for Testing

- Defined layers
- Short components
- Components decoupled from each other
- Simple for mocking

Summary

Various Types of Tests

- UI tests:
    - Specific UI tests (Espresso)
    - Random UI tests (Exerciser Monkey)

Summary

Various Types of Tests

- Unit tests:
    - JVM unit tests
    - Android unit tests (Instrumentation)
- Mock objects (Mockito)

Tests Types

JVM Unit Tests

Instrumented Unit Tests

Instrumented UI Tests

Runs locally

Runs on emulator or device


BUG

Congratulations!!!

Summary

Why is architecture important?

- Maintainable and extendable software
- Testable software
- Readable and easy to understand for new developers

Summary

Common Patterns

- SOLID Principles
- MVC, MVP and MVVM
- The clean architecture

The SOLID Principles for Object Oriented Design

Single responsibility principle

Open / close principle

Liskov substitution principle

Interface segregation principle

Dependency inversion principle

Summary

Common Patterns

- MVC
- MVP
- MVVM

Activity

CoinModel

CryptoCoinEntity

Tracker:
- Activity lifecycle
- Location

bindViews

MyCryptoAdapter

Network Logic for API request

Persist data to local storage

RecyclerView

EntityToModelMapperTask

Runtime permission logic

Read data from local storage

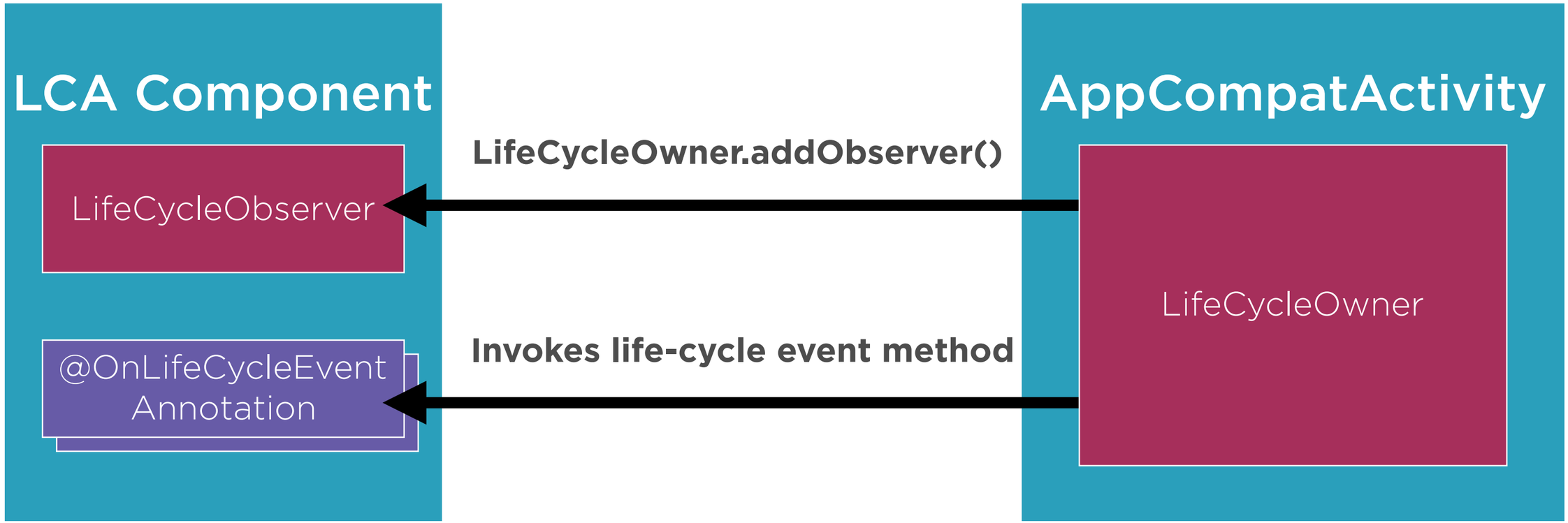
Summary

Android Architecture Components

- What are their benefits?

- How to utilize them in our application?

Lifecycle Aware Components



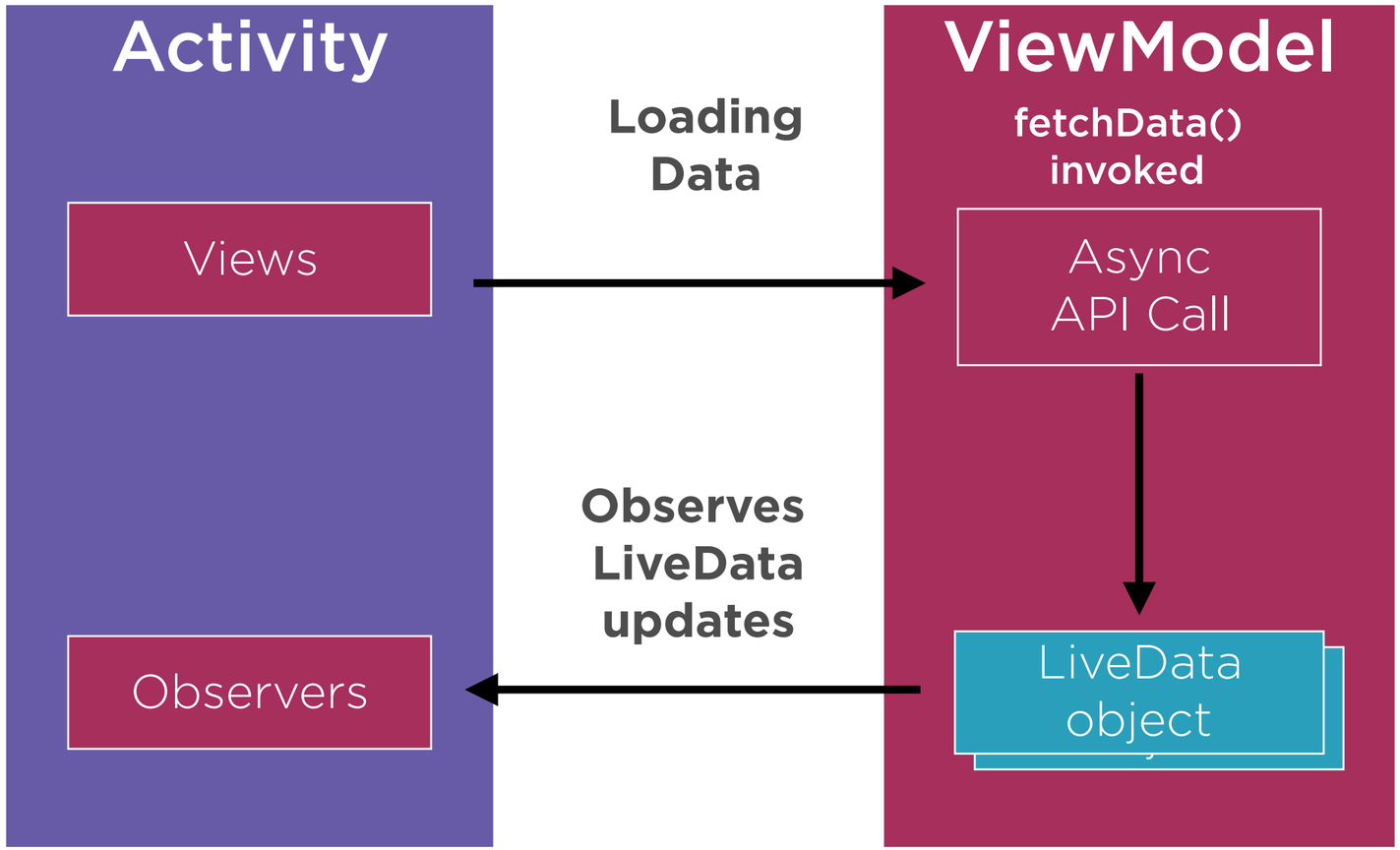
Summary

ViewModel and the LiveData Framework

- How to combine both together

- How to decouple our application logic from our activities

ViewModel and LiveData Framework

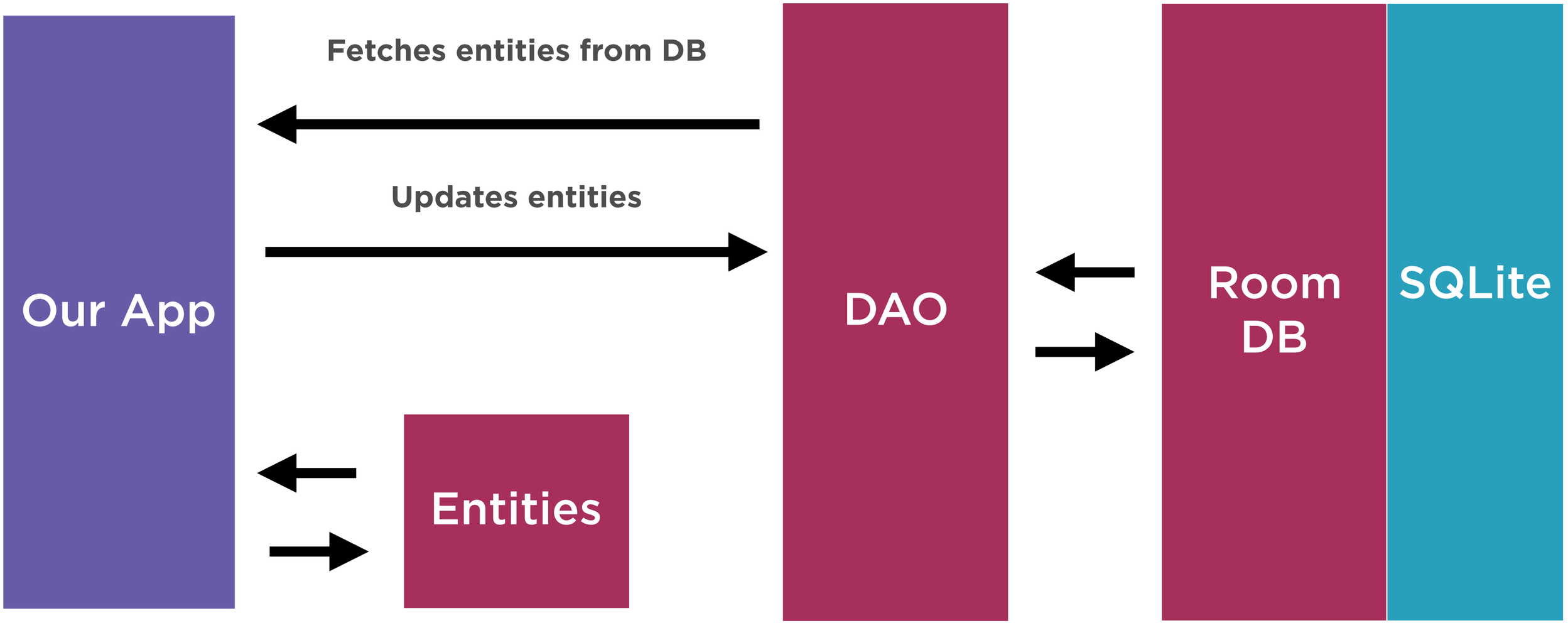


Summary

Room Persistence Solution

- Provides us with a simple abstraction layer on top of SQLite
- Uses powerful annotations for:
  - Inserting
  - Updating
  - Deleting
  - Querying

Room Persistence Solution



Summary

Android Testing

- How to use the UI Excelsior Monkey tool
- How to user Mockito for mocking objects
- How to write JVM and instrumentation Unit tests

Activity

CoinModel

CryptoCoinEntity

Tracker:
- Activity lifecycle
- Location

bindViews

MyCryptoAdapter

Network Logic for API request

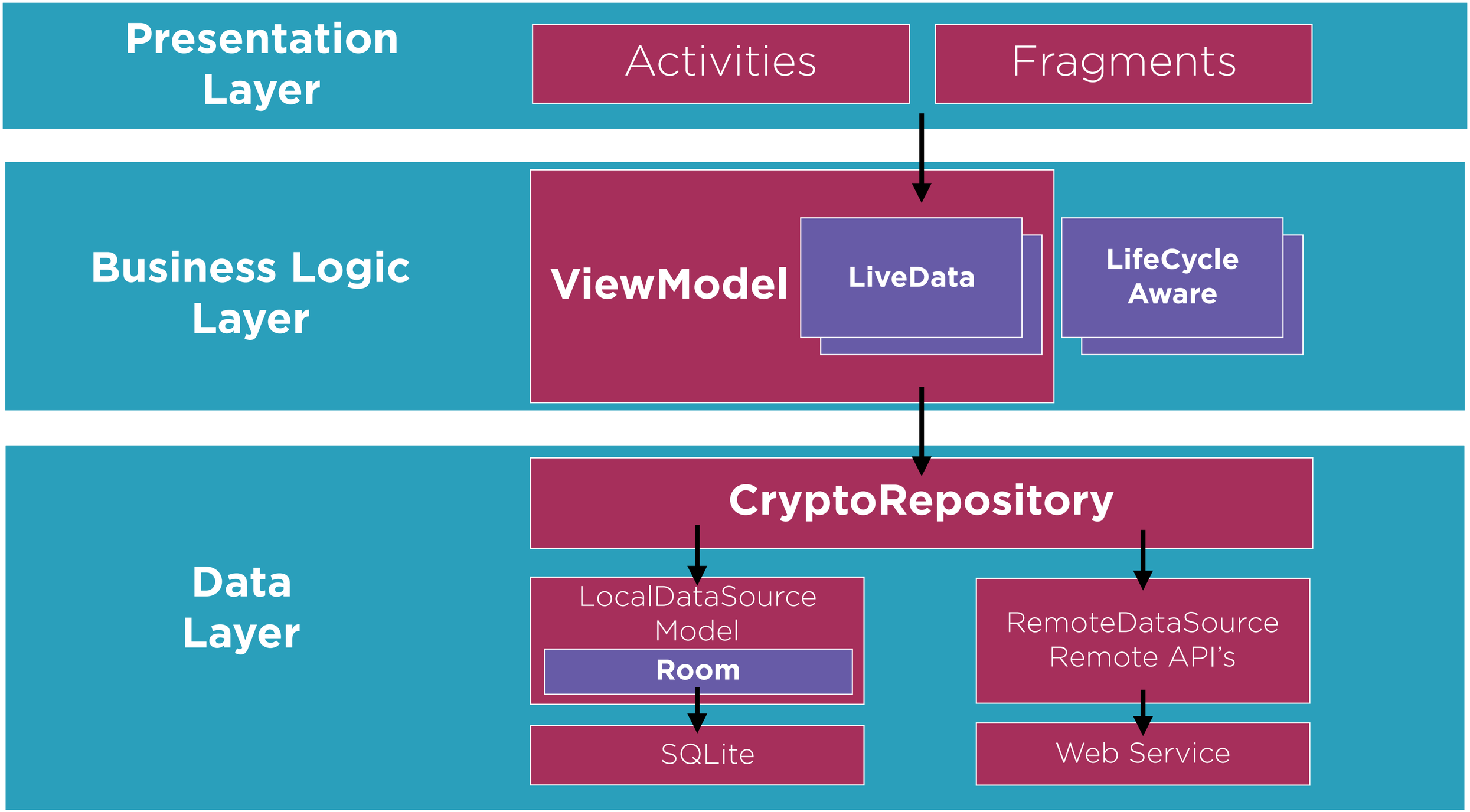
Persist data to local storage

RecyclerView

EntityToModelMapperTask

Runtime permission logic

Read data from local storage


Presentation Layer
Activities
Fragments
Business Logic Layer
ViewModel
LiveData
LifeCycle Aware
Data Layer
CryptoRepository
LocalDataSource
Model
Room
SQLite
RemoteDataSource
Remote API's
Web Service