

**What about unit testing in Android?**

Android:

Platform

Operating system

Framework

## Unit testing in Android framework:

Superset of Java 

Adds Android specific APIs 

To keep classes unit testable:

Don't use Android static APIs

Don't instantiate any Android specific classes

Mock all Android dependencies (including data structures)

## Commonly inherited Android classes:

Application

Activity

Fragment

View

Service

Etc.

**Favor composition over inheritance**  
**[Effective Java, item 16]**

**Unit testing in Android is problematic because  
you can't test Application, Activity, Fragment,  
Service, etc.**

User Interface (UI) Layer

Application Layer

Domain Layer

Infrastructure Layer



User interface layer:

Consists primarily of Android Views which aren't unit testable

Not unit testable

UI logic isn't good fit for automated testing anyway

Application layer:

User input handling and flow control

Contains Activities, Fragments and Services which aren't unit testable

Keep logic in Activities, Fragments and Services to a minimum

Unit test the rest of the classes

Domain layer:

Three rules to keep domain logic unit testable:

1. Don't put in Activities, Fragments, etc.
2. Don't issue static calls from
3. Don't instantiate Android data structures inside

Use cases are very good abstraction for domain logic

Infrastructure layer:

Testability depends on implementation choice

Can often be replaced with third-party libraries

User Interface (UI) Layer



Application Layer



Domain Layer



Infrastructure Layer

