

Effective TDD on Android with Kotlin



Olmo Gallegos Hernández



@voghDev

GDG #DevFestGRX 2017

Example project

<https://github.com/voghDev/ChuckNorrisJokes>

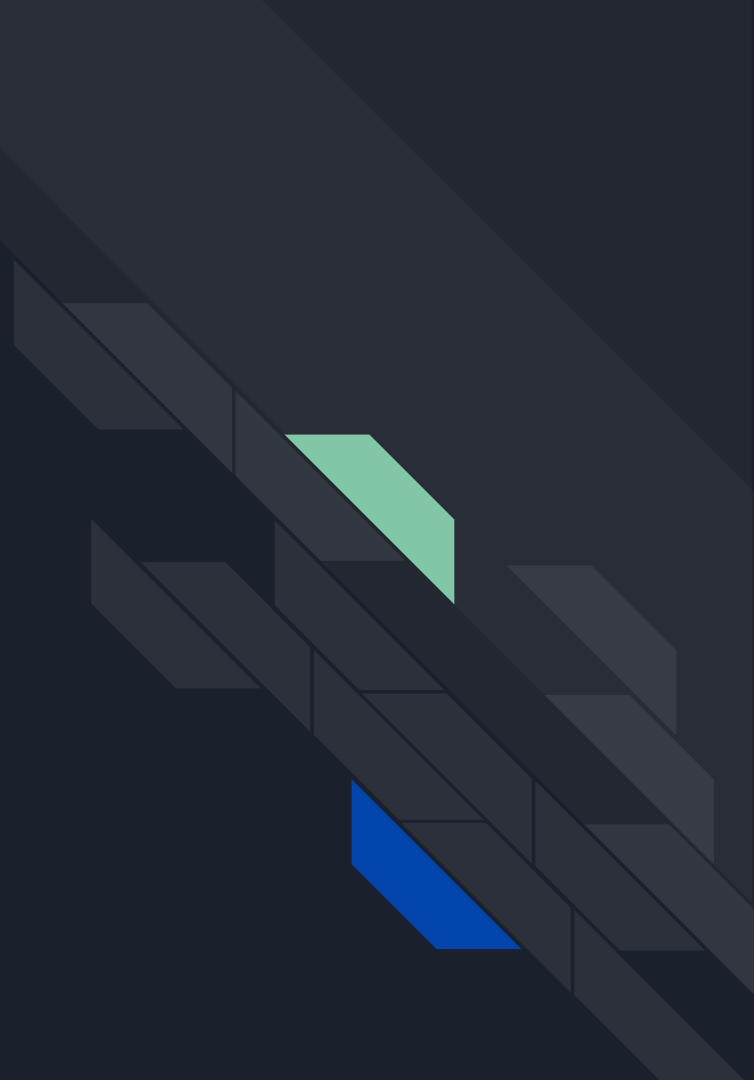




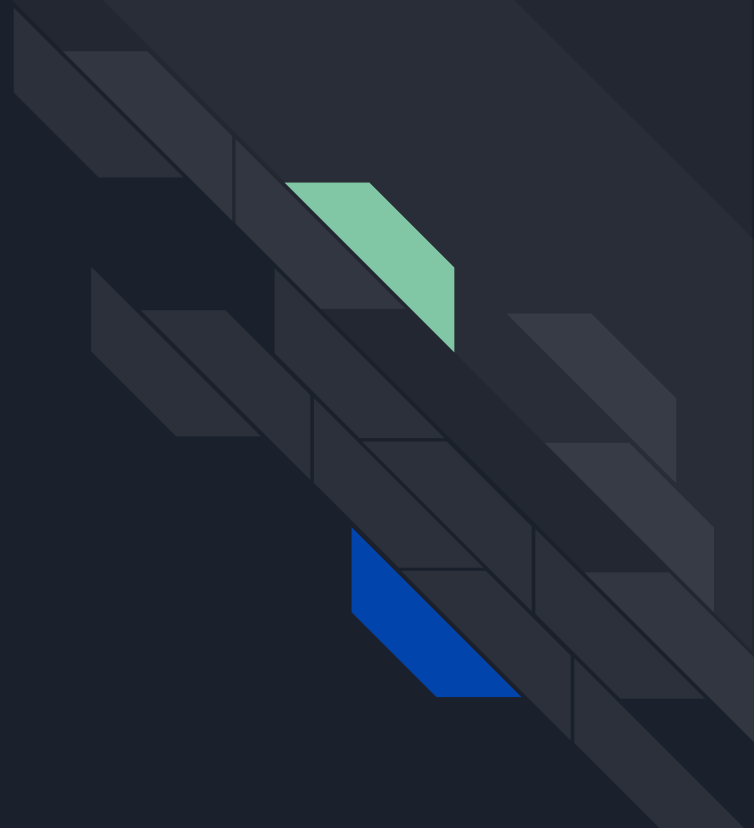
Overview - Basis of the workshop

- Custom approach to TDD
- Testable code
- Android architecture, MVP + Passive View
- Kotlin coroutines

<http://chucknorris.io> API



<http://chucknorris.io> API





API requests to <http://chucknorris.io>

- Get random joke



API requests to <http://chucknorris.io>

- Get random joke
- Get joke by category



API requests to <http://chucknorris.io>

- Get random joke
- Get joke by category
- Get categories



API requests to <http://chucknorris.io>

- Get random joke
- Get joke by category
- Get categories
- Search jokes by keyword



Example Request: Random joke by category

JSON

```
{
  "category": [
    "dev"
  ],
  "icon_url":
  "https://assets.chucknorris.host/img/avatar/chuck-norris.png",
  "id": "4rfk-ymnth28ny8q2pbrow",
  "url": "http://api.chucknorris.io/jokes/4rfk-ymnth28ny8q2pbrow",
  "value": "Chuck Norris went out of an infinite loop."
}
```



The Architecture - Android MVP

View

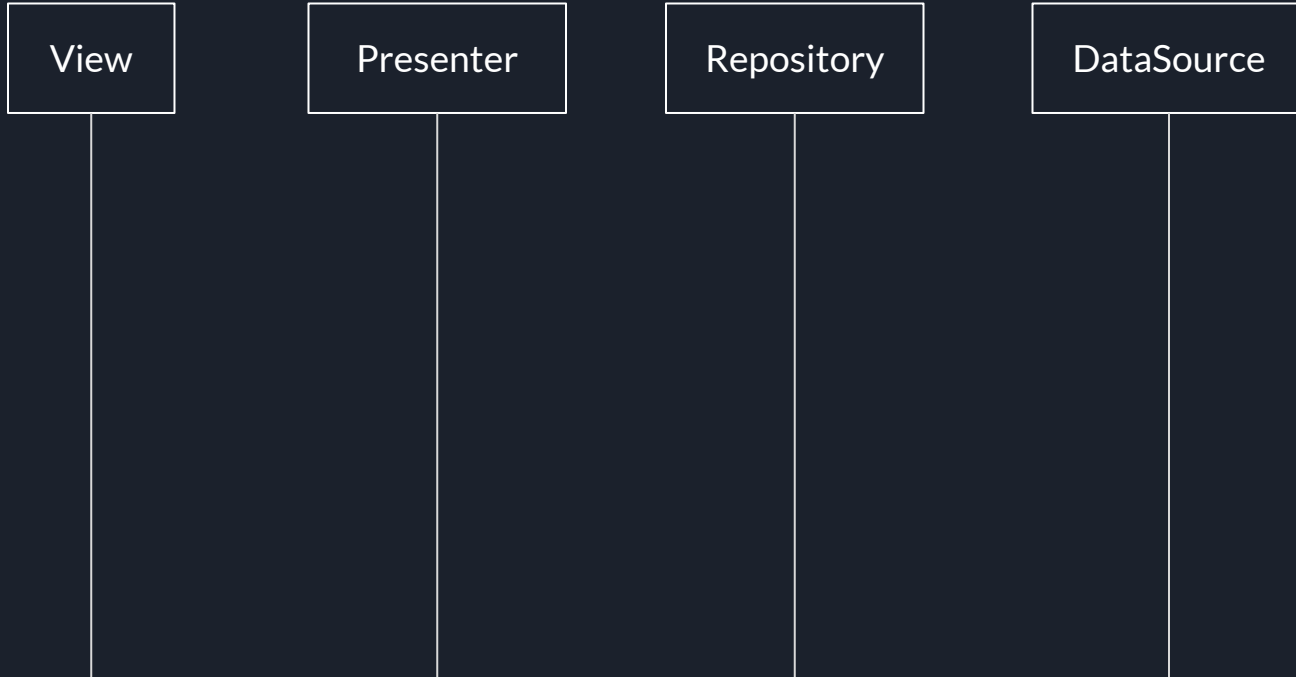
Presenter

Repository

DataSource



The Architecture - Android MVP



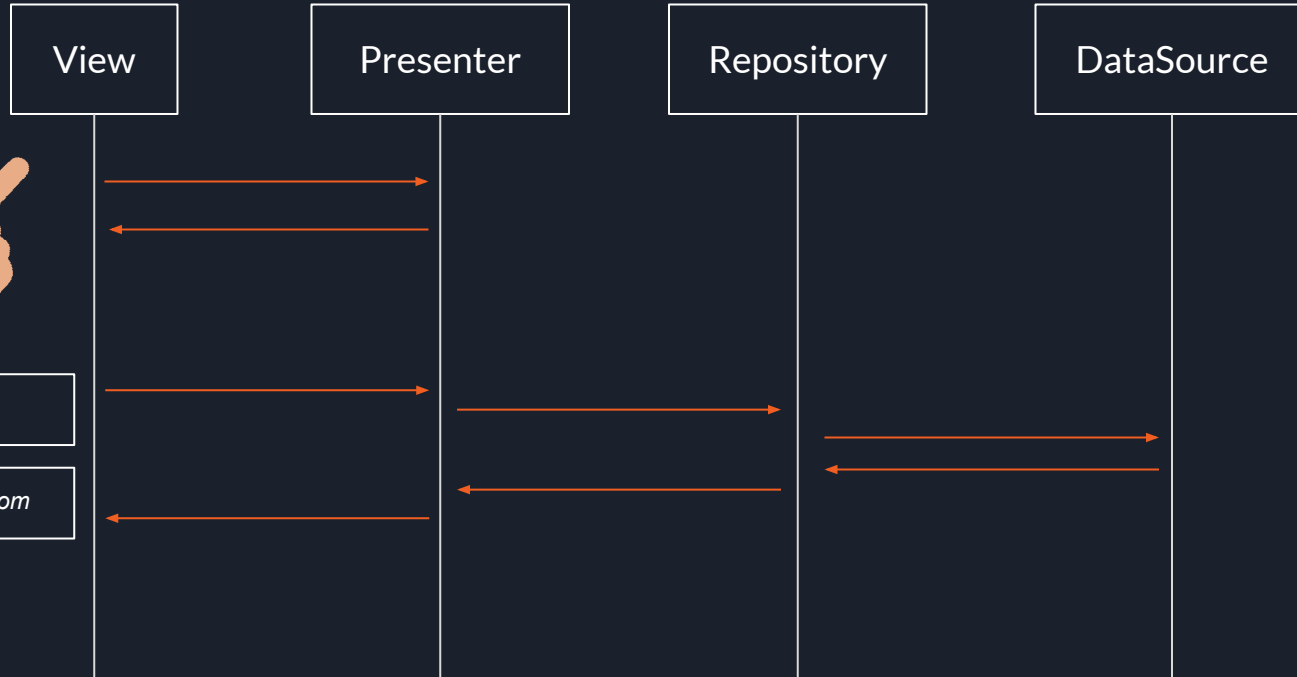
The Architecture - Android MVP



User TAP

icon by flaticon.com

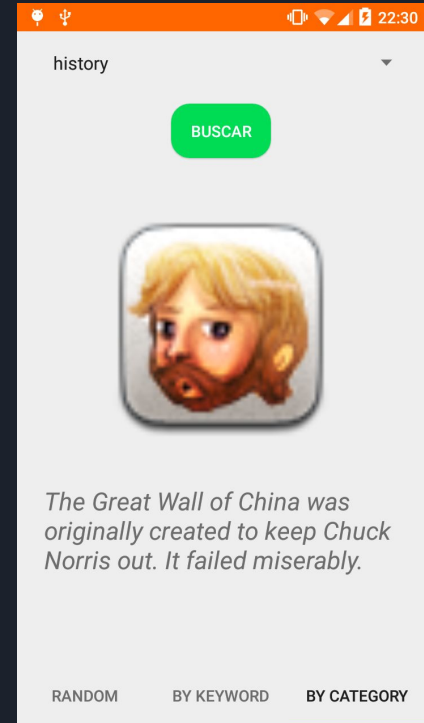
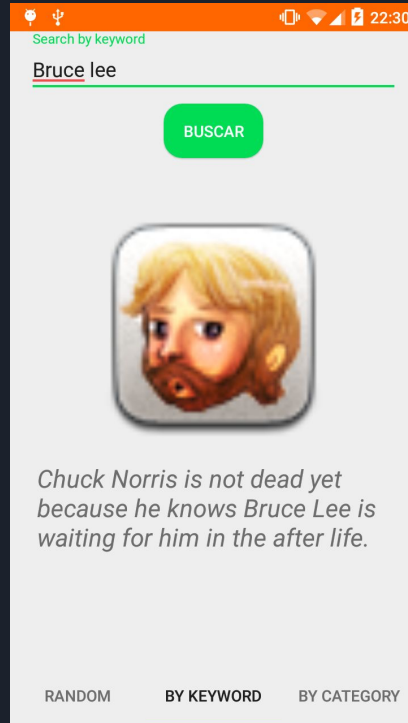
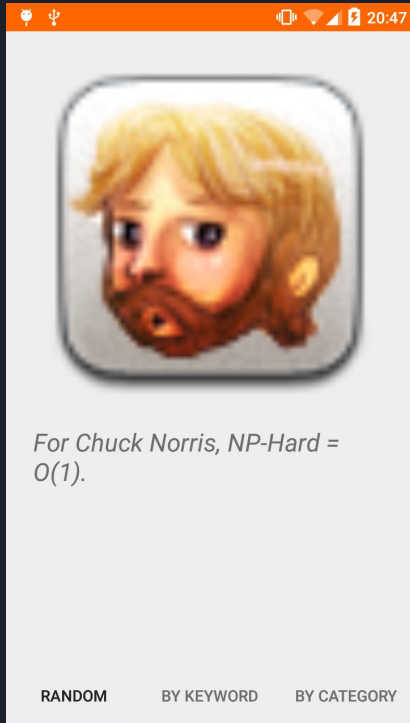
The Architecture - Android MVP




User TAP


icon by flaticon.com

Starting from empty App





Time to get some
work done!

The image features a dark blue background with several geometric shapes. In the top left, there is a large blue parallelogram and a light green parallelogram. In the bottom left, there is a smaller, multi-colored geometric shape composed of blue, orange, and pink triangles.



Credits

Bandhook Kotlin - <https://github.com/antoniolg/Bandhook-Kotlin>

OpenLibraryApp - <https://github.com/jrgonzalezg/OpenLibraryApp>

kotlinx.coroutines - <https://kotlinlang.org/docs/reference/coroutines.html>

kategory.io - <http://kategory.io>



Conclusions

- TDD is not the real challenge; **having testable code** is
- Tests need to be maintained, as requirements do
- The real win is to reproduce bugs directly in your tests
- Less executions → more productivity



That's all folks!

