Liberating Code with

Dev Talks

Muh Isfhani Ghiath 🌱

Senior Software Engineer Android @ **Tokopedia**
Co-Organizer @ **GDG Jakarta**
Former Lead @ **Google DSC**

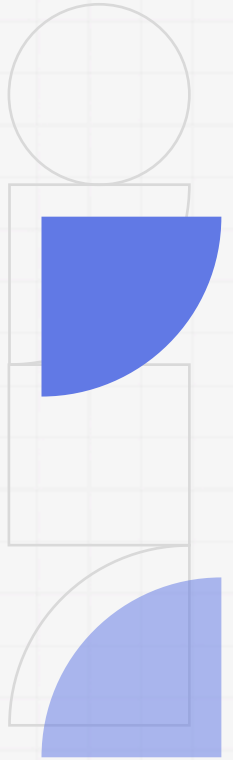
@isfaaghyth

Opinions expressed here are my own

~~Liberating Code~~ with 

How kotlin makes life easier

Part 1



Tips in general

Kotlin delegation 🏄‍♂️

Delegation is one of design pattern in which an object handles a request by delegating to a helper object, called the delegate.

```
class Label @JvmOverloads constructor(
    context: Context,
    attrs: AttributeSet? = null,
    defStyleAttr: Int = 0
) : FrameLayout(context, attrs, defStyleAttr) {
    var inverted by uniqueObservable(false) { refreshDrawableState() }
    ...
}
```

Kotlin delegation 🏄

Main point is to reduce boilerplate of code,
with following a rule based.

```
// This automatically creates and clears the binding in a lifecycle-aware way.
private val binding: WifiNetworksFragmentBinding by viewBinding()

// This does the inflate too.
private val binding: WifiNetworkViewBinding by viewBinding()
```

Extension! 🧤

Kotlin provides an ability to extend a class with new functionality without having to inherit from the class or use design patterns such as *Decorator*.

One of example, converting dp to px.

```
val Number.dp get() = toFloat() * (Resources.getSystem().displayMetrics
                        .densityDpi.toFloat() / DisplayMetrics.DENSITY_DEFAULT)


recyclerView.updatePadding(top = 14.dp.toInt())
```

Structured Concurrency! ⚡

Kotlin Coroutines make asynchronous code as easy to work with as blocking code. Coroutines dramatically simplify background task management for everything from network calls to accessing local data.

Structured Concurrency! ⚡

```
private class DownloadFilesTask extends AsyncTask<URL, Integer, Long> {
    protected Long doInBackground(URL... urls) {
        int count = urls.length;
        long totalSize = 0;
        // TODO
        return totalSize;
    }

    protected void onProgressUpdate(Integer... progress) {}

    protected void onPostExecute(Long result) {}
}
```

Structured Concurrency! ⚡

```
fun makeLoginRequest(username: String, token: String) {
    viewModelScope.launch {
         when (result) {
            is Result.Success<LoginResponse> -> // Happy path
            else -> // Show error in UI
        }
    }
}
```

Testing! 


```
val car = mockk<Car>()

every { car.drive(Direction.NORTH) } returns Outcome.OK

car.drive(Direction.NORTH) // returns OK

verify { car.drive(Direction.NORTH) }

confirmVerified(car)
```

Testing! 

Final by default

In contrast to Java, Kotlin classes (and functions) are final by default. This means Mockito requires extra configuration to enable it to work (and only in some cases). Whereas, Mockk can work around this using inline class transformation.

Testing! 

Chained mocking

With Mockk you can chain your mocking, meaning you can provide concise and clear tests.

```
val mockedClass = mockk<MyClass>()
every { mockedClass.someMethod().someOtherMethod() } returns
"Something"
```

Testing! 

Object mocking

Kotlin objects map to Java statics. Mockito alone doesn't support mocking of statics. There are other frameworks you can combine with Mockito, but Mockk provides this out of the box.

```
mockObject(MyObject)
every { MyObject.someMethod() } returns "Something"
```

🚀 Live Templates

makes it easier for you to write
the same code over and over.

✅ Green Gutter Culture

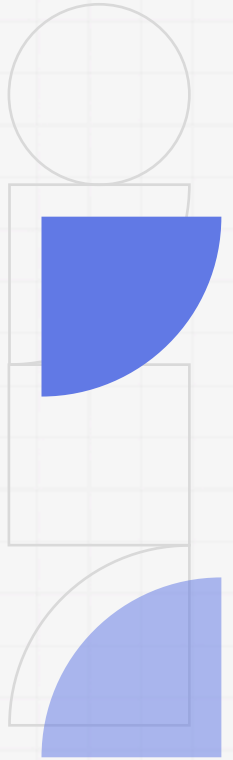
Implement a culture based on
Android Studio's built-in inspections.

Treat all Kotlin warnings as errors 🧙🏻

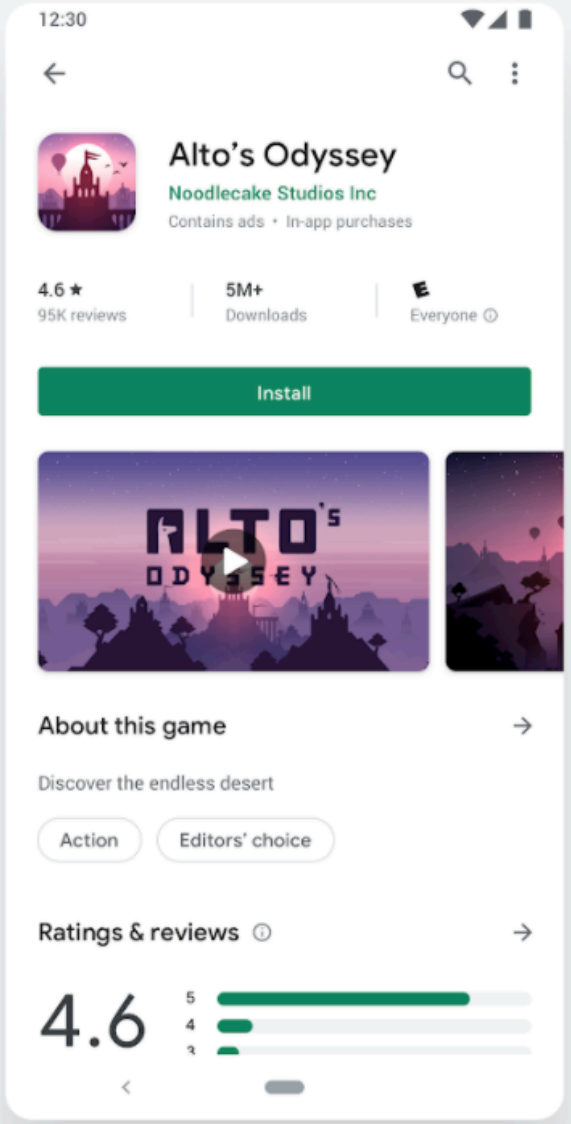
Treat all of the *warnings* as *error* so we can fix it
and explicit suppressed.

```
allprojects {
    gradle.projectsEvaluated {
        tasks.withType(org.jetbrains.kotlin.gradle.tasks.KotlinCompile).configureEach {
            kotlinOptions {
                allWarningsAsErrors = true
            }
        }
    }
}
```

Part 2



Study Case


12:30
Alto's Odyssey
Noodlecake Studios Inc
Contains ads • In-app purchases
4.6 ★
95K reviews
5M+
Downloads
Everyone
Install
About this game
Discover the endless desert
Action
Editors' choice
Ratings & reviews
4.6
5
4

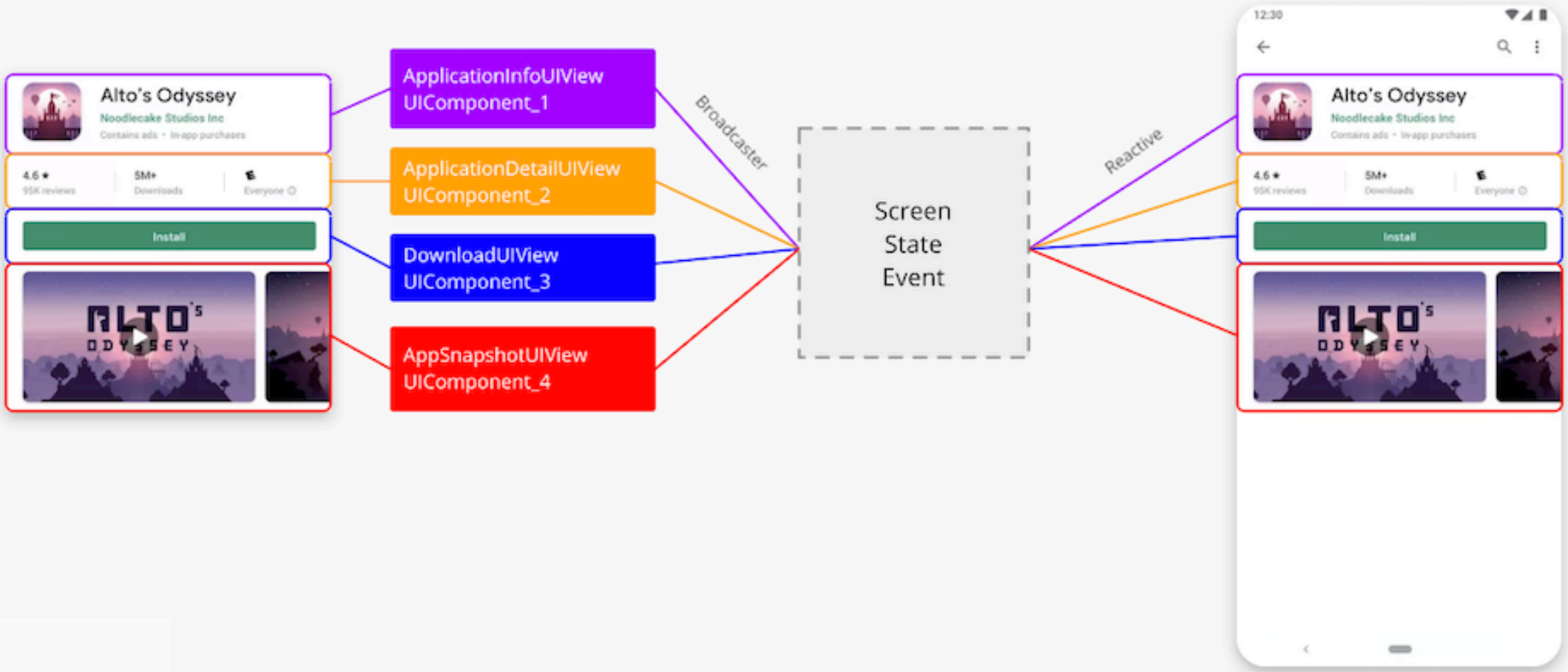
A few solution:

- Using RecyclerView with Multiple View Type
  - With Visitable (Visitor Pattern)
  - With Adapter Delegate

- <include layout="...">

Is it better?

No, of course not. you have to define
a ton of code in one place (unmaintainable)


Alto's Odyssey
Noodlecake Studios Inc
Contains ads • In-app purchases
4.6 ★
95K reviews
5M+
Downloads
Everyone
Install
ApplicationInfoUIView
UIComponent_1
ApplicationDetailUIView
UIComponent_2
DownloadUIView
UIComponent_3
AppSnapshotUIView
UIComponent_4
Broadcaster
Screen
State
Event
Reactive
12:30
Alto's Odyssey
Noodlecake Studios Inc
Contains ads • In-app purchases
4.6 ★
95K reviews
5M+
Downloads
Everyone
Install

How it works?

Let’s take a look in here:
https://github.com/isfaaghyth/floppy

Kotlin based?

Jetpack Compose! 

Jetpack Compose is Android's modern toolkit
for building native UI. It simplifies and accelerates UI
development on Android. Quickly bring your app to life
with less code, powerful tools, and intuitive Kotlin APIs.

Jetpack Compose! 

Less code

Do more with less code and avoid entire classes of bugs, so code is simple and easy to maintain.

Intuitive

Just describe your UI, and Compose takes care of the rest. As app state changes, your UI automatically updates.

Accelerate Development

Compatible with all your existing code so you can adopt when and where you want. Iterate fast with live previews and full Android Studio support.

Powerful

Create beautiful apps with direct access to the Android platform APIs and built-in support for Material Design, Dark theme, animations, and more.

```
@Composable
fun JetpackCompose() {
    Card {
        var expanded by remember { mutableStateOf(false) }
        Column(Modifier.clickable { expanded = !expanded }) {
            Image(painterResource(R.drawable.jetpack_compose))
            AnimatedVisibility(expanded) {
                Text(
                    text = "Jetpack Compose",
                    style = MaterialTheme.typography.h2,
                )
            }
        }
    }
}
```

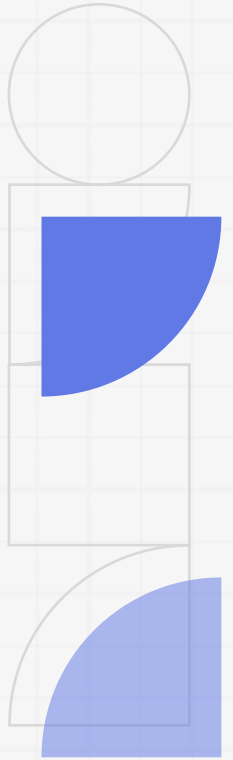
Jetpack Compose! 

Sample Project with Modularization:

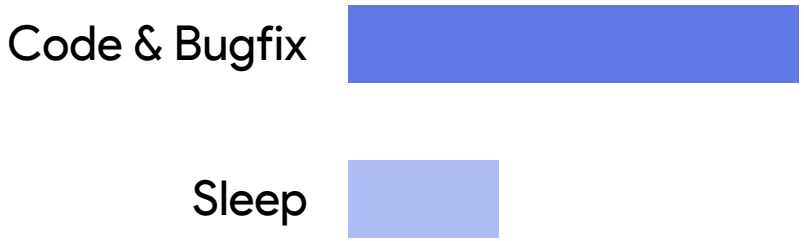
https://github.com/isfaaghyth/unofficial-pd

(~~unmaintain well~~) (~~will be treated soon~~)

TL;DR



Before and After using Kotlin 🛌



Thankyou!