



QA In Kotlin: From a Compiler Backend To the Entire Ecosystem

Alexander Zakharenko, Artur Parpibaev (JetBrains)



Artur Parpibaev

Kotlin Automation / Performance QA Lead

Cat owner 🐱 / Love mountains 🏔️ / Fan of Korean culture 🇰🇷





Alexander Zakharenko

Kotlin Compiler QA Lead

Wine lover 🍷 / Love singing 🎤 / Fan of Paul McCartney 🎸



Talk Structure

- What is Kotlin and its ecosystem
- From users to machine code
- Levels of testing
- How we measure quality
- Conclusions

What is Kotlin

What is Kotlin

- Announced in 2011 (JVM)
- Version 1.0 released in 2016
- Kotlin Multiplatform released in 2017
- Preferred language for Android app developers since 2019
- Used by Google, Amazon, Meta, Netflix, Uber, etc.

What is Kotlin and its ecosystem

What is Kotlin and its ecosystem

Compiler

What is Kotlin and its ecosystem

Compiler

Code editor

What is Kotlin and its ecosystem

Compiler

Code editor

Libraries

What is Kotlin and its ecosystem

Compiler

Code editor

Build system

Libraries

What is Kotlin and its ecosystem

Compiler

Kotlin Multiplatform

Code editor

Libraries

Build system

What is Kotlin and its ecosystem

Compiler

Kotlin Multiplatform

Code editor

Libraries

Build system

Documentation

What is Kotlin and its ecosystem

Compiler

Kotlin Multiplatform

Code editor

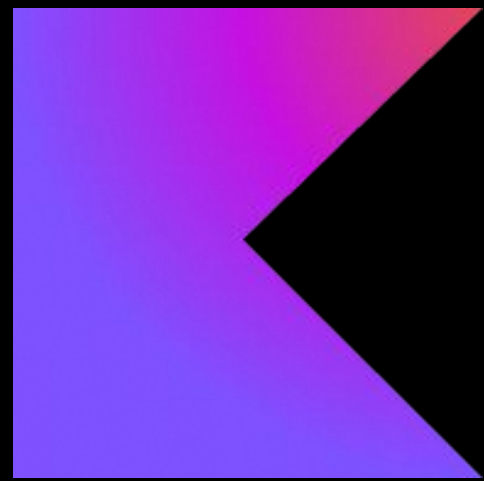
Libraries

Build system

Support

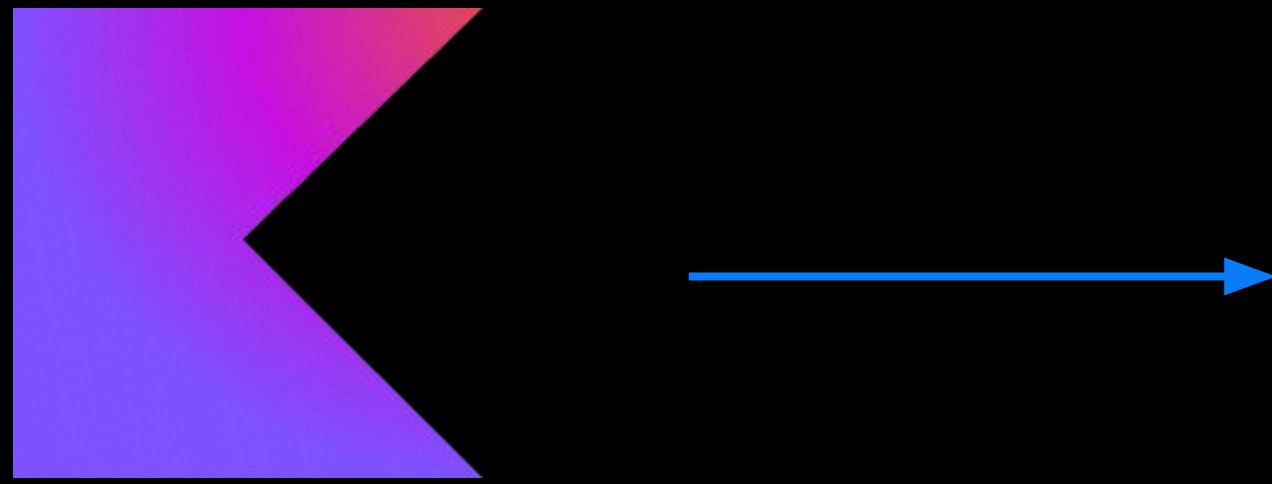
Documentation

What is Kotlin and its ecosystem



Compiler

What is Kotlin and its ecosystem



Compiler

- Kotlin / JVM
- Kotlin / JS
- Kotlin / Native
- Kotlin / Wasm

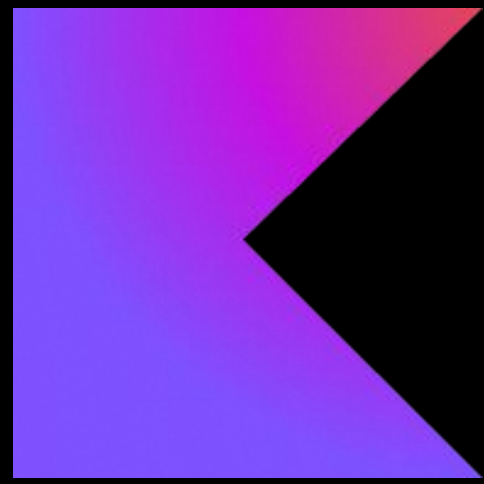
→ `workdir cat main.kt`

```
fun main() {  
    println("Hello, Stockholm!")  
    stockholm()  
}
```

→ `workdir kotlinc-native -o main.kexe main.kt`

```
main.kt:3:5: error: unresolved reference: stockholm  
    stockholm()  
    ^
```

What is Kotlin and its ecosystem

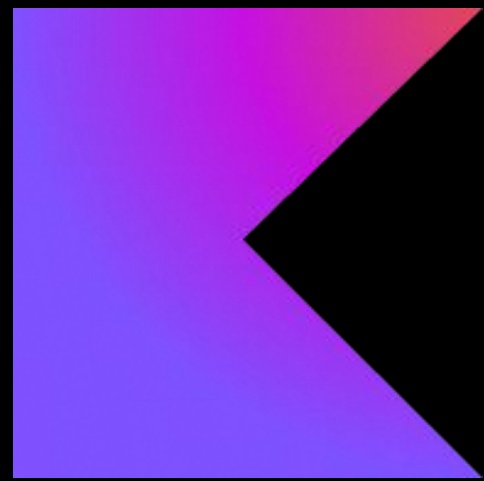


Compiler

- Kotlin / JVM
- Kotlin / JS
- Kotlin / Native
- Kotlin / Wasm

Core Ecosystem

What is Kotlin and its ecosystem



Compiler

- Kotlin / JVM
- Kotlin / JS
- Kotlin / Native
- Kotlin / Wasm

Core Ecosystem

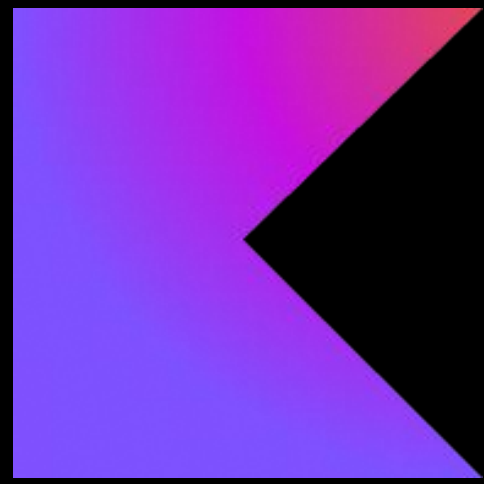
- Gradle / Maven support

```
→ workdir kotlinc hello.kt -include-runtime -d  
hello.jar
```

```
→ workdir kotlinc-native -g -enable-assertions -module-name  
org.example:enums -no-endorsed-libs -output enums.klib -produce library  
-Xshort-module-name=enums -target macos_arm64 -Xmulti-platform hello.kt  
→ workdir kotlinc-native -g -enable-assertions -Xinclude=enums.klib  
-no-endorsed-libs -output Enums.framework -produce framework -target  
macos_arm64 -Xmulti-platform  
→ workdir clang -framework Foundation,Enums -F . -Xlinker -rpath  
-Xlinker . -Xlinker Enums.framework/Versions/A/Enums main.m -o  
enums_objc.kexe
```

→ `workdir ./gradlew build`

What is Kotlin and its ecosystem



Compiler

- Kotlin / JVM
- Kotlin / JS
- Kotlin / Native
- Kotlin / Wasm

Core Ecosystem

- Gradle / Maven support
- Libraries

- All
- Public
- Sources
- Forks
- Archived
- Templates

All

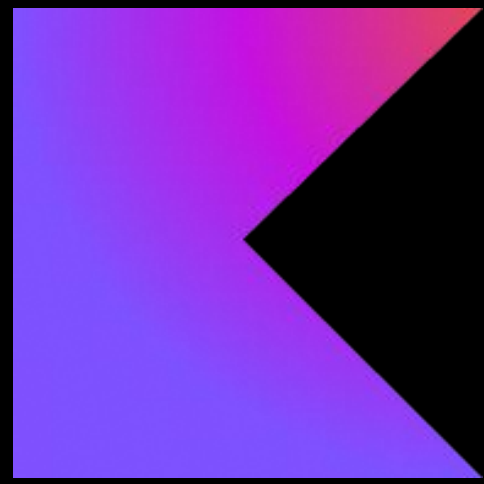
kotlinx sort:stars

18 repositories

Stars

 kotlinx.coroutines Library support for Kotlin coroutines	Kotlin	1.9k	14k
 kotlinx.serialization Kotlin multiplatform / multi-format serialization	Kotlin	660	5.8k
 kotlinx-datetime KotlinX multiplatform date/time library	Kotlin	121	2.7k
 kotlinx.html Kotlin DSL for HTML	Kotlin	134	1.7k
 kotlinx-kover	Kotlin	54	1.5k
 kotlinx-io Kotlin multiplatform I/O library	Kotlin	69	1.5k
 kotlinx.collections.immutable Immutable persistent collections for Kotlin	Kotlin	64	1.3k
 kotlinx-atomicfu The idiomatic way to use atomic operations in Kotlin	Kotlin	69	1k
 kotlinx-rpc Add asynchronous RPC services to your multiplatform applications.	Kotlin	37	956
 kotlinx-cli Pure Kotlin implementation of a generic CLI parser.	Kotlin	71	928
 kotlinx-benchmark Kotlin multiplatform benchmarking toolkit	Kotlin	43	605
 kotlinx-knit Kotlin source code documentation management tool	Kotlin	19	310
 kotlinx-nodejs Kotlin external declarations for using the Node.js API from Kotlin code targeting JavaScript	Kotlin	19	219
 kotlinx.reflect.lite Lightweight library allowing to introspect basic stuff about Kotlin symbols	Kotlin	13	179
 kotlinx-browser Kotlin browser API	Kotlin	17	170

What is Kotlin and its ecosystem



Compiler

- Kotlin / JVM
- Kotlin / JS
- Kotlin / Native
- Kotlin / Wasm

Core Ecosystem

- Gradle / Maven support
- Libraries
- Dokka

- ▶ [kotlin-reflect](#)
- ▶ [kotlin-stdlib](#)
- ▶ [kotlin-test](#)

kotlin-stdlib

CommonJSJVMNativeWasm-JSWasm-WASI

Kotlin Standard Library

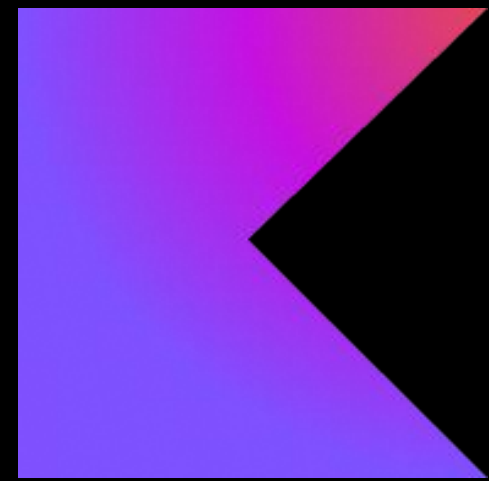
The Kotlin Standard Library provides living essentials for everyday work with Kotlin. These include:

- Higher-order functions implementing idiomatic patterns ([let](#), [apply](#), [kotlin.io.use](#), [kotlin.synchronized](#), etc).
- Extension functions providing querying operations for collections (eager) and sequences (lazy).
- Various utilities for working with strings and char sequences.
- Extensions for JDK classes making it convenient to work with files, IO, and threading.

Packages

kotlin	● Common	● JS	● JVM	● Native	● WASM-JS	● WASM-WASI
Core functions and types, available on all supported platforms.						
kotlin.annotation						● Common
Library support for the Kotlin annotation facility.						
kotlin.collections	● Common	● JS	● JVM	● Native	● WASM-JS	● WASM-WASI
Collection types, such as Iterable , Collection , List , Set , Map and related top-level and extension functions.						

What is Kotlin and its ecosystem



Compiler

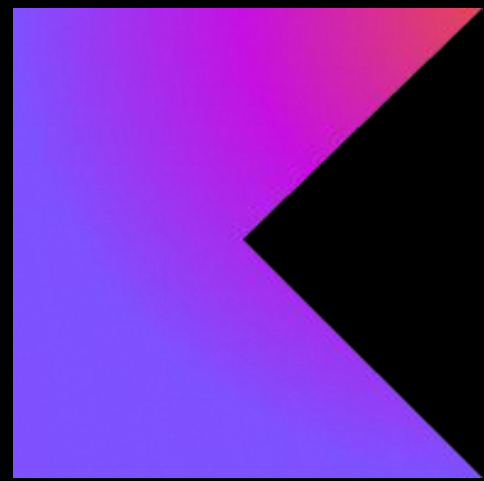
- Kotlin / JVM
- Kotlin / JS
- Kotlin / Native
- Kotlin / Wasm

Core Ecosystem

- Gradle / Maven support
- Libraries
- Dokka

IDE Plugin

What is Kotlin and its ecosystem



Compiler

- Kotlin / JVM
- Kotlin / JS
- Kotlin / Native
- Kotlin / Wasm

Core Ecosystem

- Gradle / Maven support
- Libraries
- Dokka

IDE Plugin

- Endless number of features
 - Project wizards
 - Navigation
 - Quick fixes
 - Refactorings
 - ...

New Project

Kotlin

Groovy

Python

Empty Project

Generators

Maven Archetype

Jakarta EE

Spring Boot

JavaFX

Quarkus

Micronaut

Ktor

Compose Multiplatform

HTML

React

Express

Angular CLI

IDE Plugin

Vue.js

Vite

More via plugins...

Name:

kotlin_demo

Location:

~/IdeaProjects

Project will be created in: ~/IdeaProjects/kotlin_demo

☐ Create Git repository

Build system:

IntelliJ

Maven

Gradle

JDK:

17 java version "17.0.6"

Gradle DSL:

Kotlin

Groovy

☒ Add sample code

☐ Generate code with onboarding tips

☐ Generate single module build

?

To create a Kotlin Multiplatform project, [click here](#)

> Advanced Settings

?

Cancel

Create

Project wizard

App.kt × String.kt README.md

1

package org.example.app

2

3

import org.example.utils.Printer

4

5

// This is the main entrypoint of the application.

6

// It uses the Printer class, from the `:utils` subproject.

7 ▶

fun main() {

8

val message = "Hello JetBrains!"

9

val printer = Printer(message)

10

printer.printMessage()

11

}

12

✓

Refactoring

What is Kotlin and its ecosystem



Kotlin as a Platform

What is Kotlin and its ecosystem



Kotlin as a Platform

Kotlin Ecosystem

- Kotlin for Data Science

```
weather.ipynb x
+ | ✂ | 📄 | 📄 | ⬆ | ⬇ | 🏠 | 🔄 | 🏠 | 🗑 | Code v | ⚙
Managed: v Kotlin v Trusted v

Filter records from the year 2023

[16] 1 val lastYear = amsWeather.filter { datetime.year == 2023 }
2 lastYear.head(3)
Executed at 2024.04.04 14:24:58 in 262ms

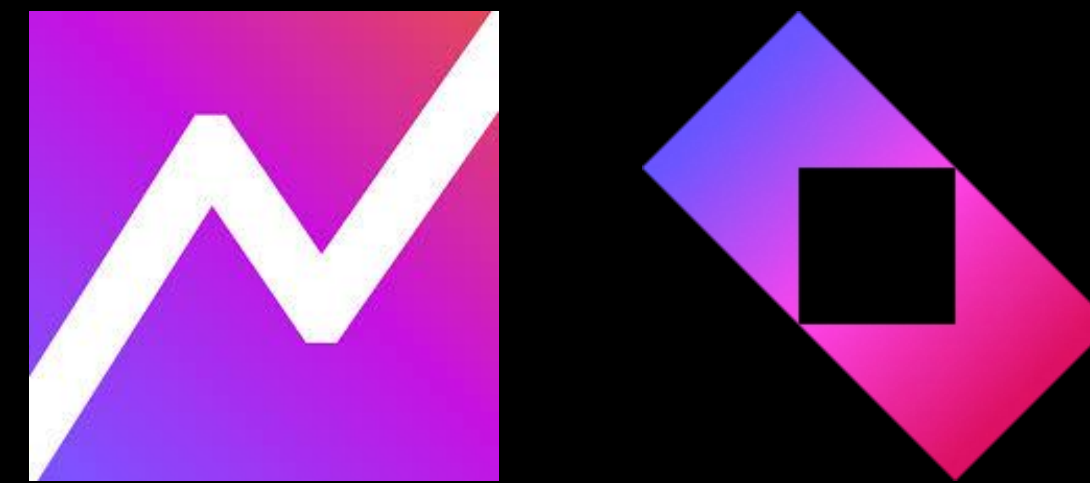
[16] v |< < 3 rows v > >| 3 rows x 24 columns | ↗ | ⬇ | @ | ⋮
name | datetime | temp | feelslike | dew | humidity | precip |
Amsterdam | 2023-01-01T00:00 | 14 | 14 | 85 | 6919 | 0 |
Amsterdam | 2023-01-01T01:00 | 14 | 14 | 87 | 7017 | 14 |
Amsterdam | 2023-01-01T02:00 | 137 | 137 | 9 | 731 | 0 |
📄

- Add a column for the date and move it to the left
- Remove the name column

[15] 1 val withDate = lastYear
2 .add("date") { datetime.date }
3 .moveToLeft { takeLast(1) }
4 .remove { name }
5 withDate.head(3)

[15] v |< < 3 rows v > >| 3 rows x 24 columns | ↗ | ⬇ | @ | ⋮
date | datetime | temp | feelslike | dew | humidity | precip |
2023-01-01 | 2023-01-01T00:00 | 14 | 14 | 85 | 6919 | 0 |
2023-01-01 | 2023-01-01T01:00 | 14 | 14 | 87 | 7017 | 14 |
2023-01-01 | 2023-01-01T02:00 | 137 | 137 | 9 | 731 | 0 |
📄
```

What is Kotlin and its ecosystem



Kotlin as a Platform

Kotlin Ecosystem

- Kotlin for Data Science
- Frameworks (Ktor, etc.)



The screenshot shows an IDE window with three tabs: Application.kt, Routing.kt (selected), and build.gradle.kts (ktor-sample). The code in Application.kt is as follows:

```
1 package com.example
2
3 > import ...
7
8 fun Application.configureRouting() {
9     routing {
10         get("/") {
11             call.respondText(text: "Hello World!")
12         }
13     }
14 }
15
```

A tooltip is visible over the `call.respondText` call on line 11, showing the parameter `text: "Hello World!"`. On the right side of the editor, there is a warning icon and the number 1, indicating a single warning.

Ktor features in IDE

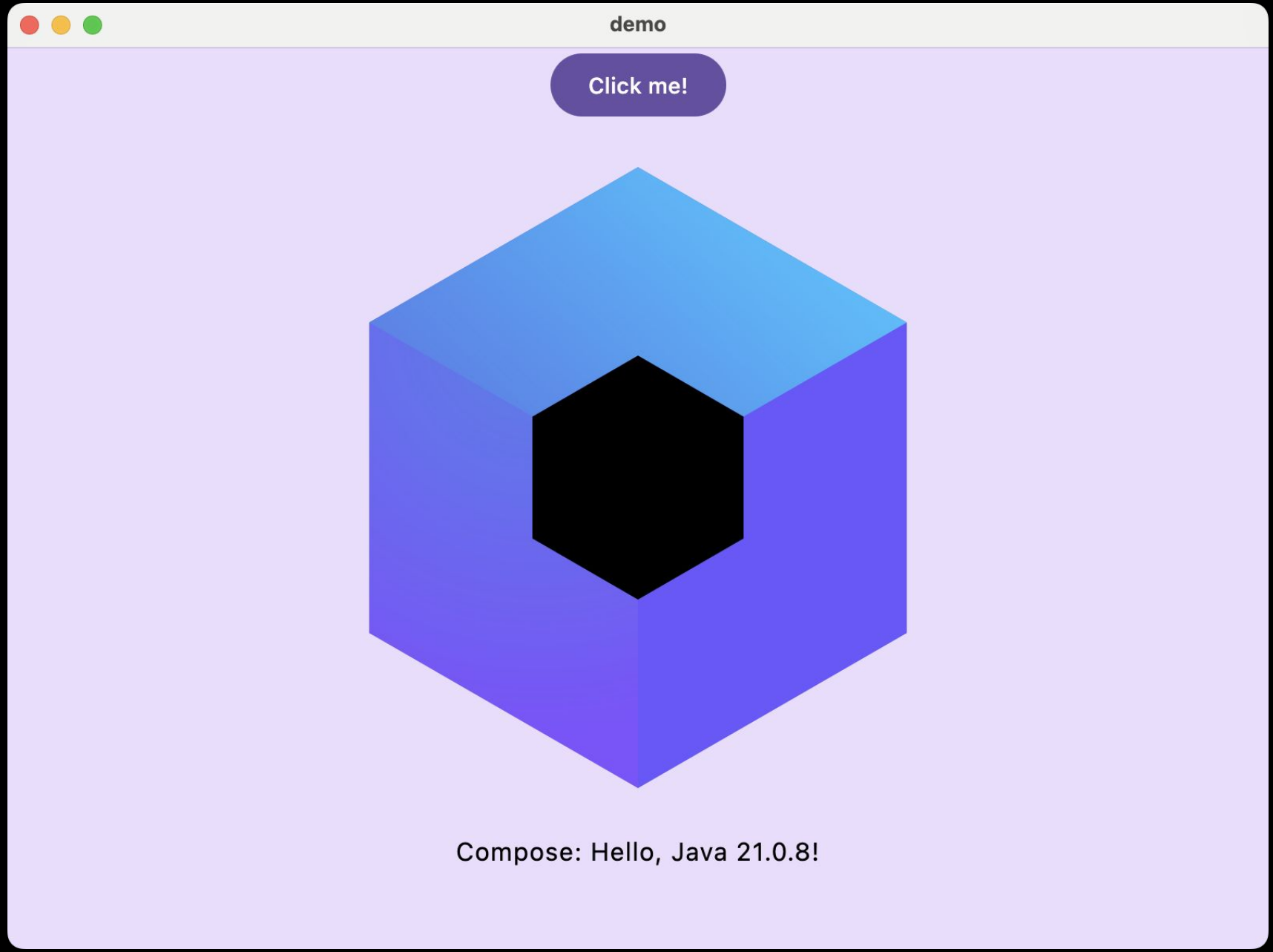
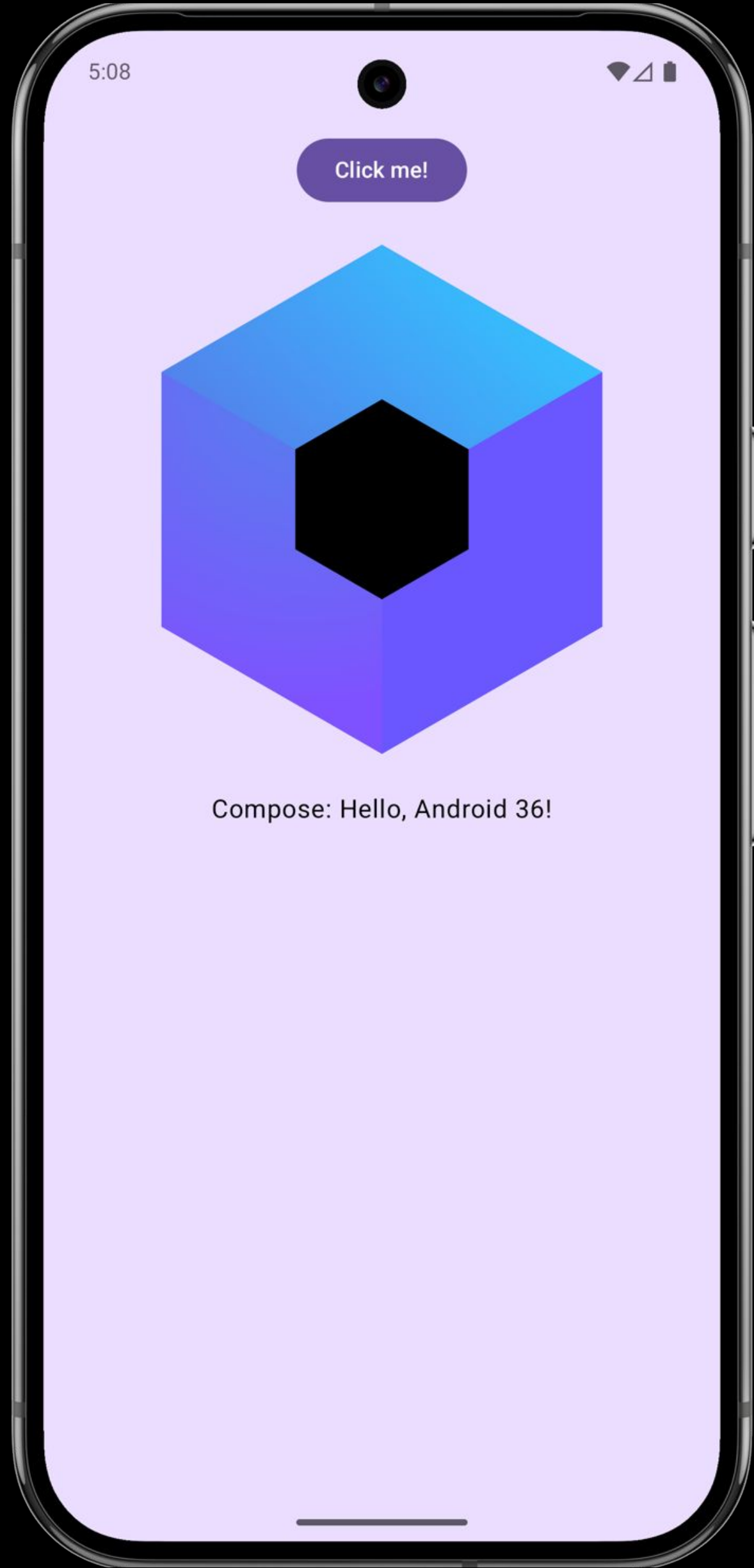
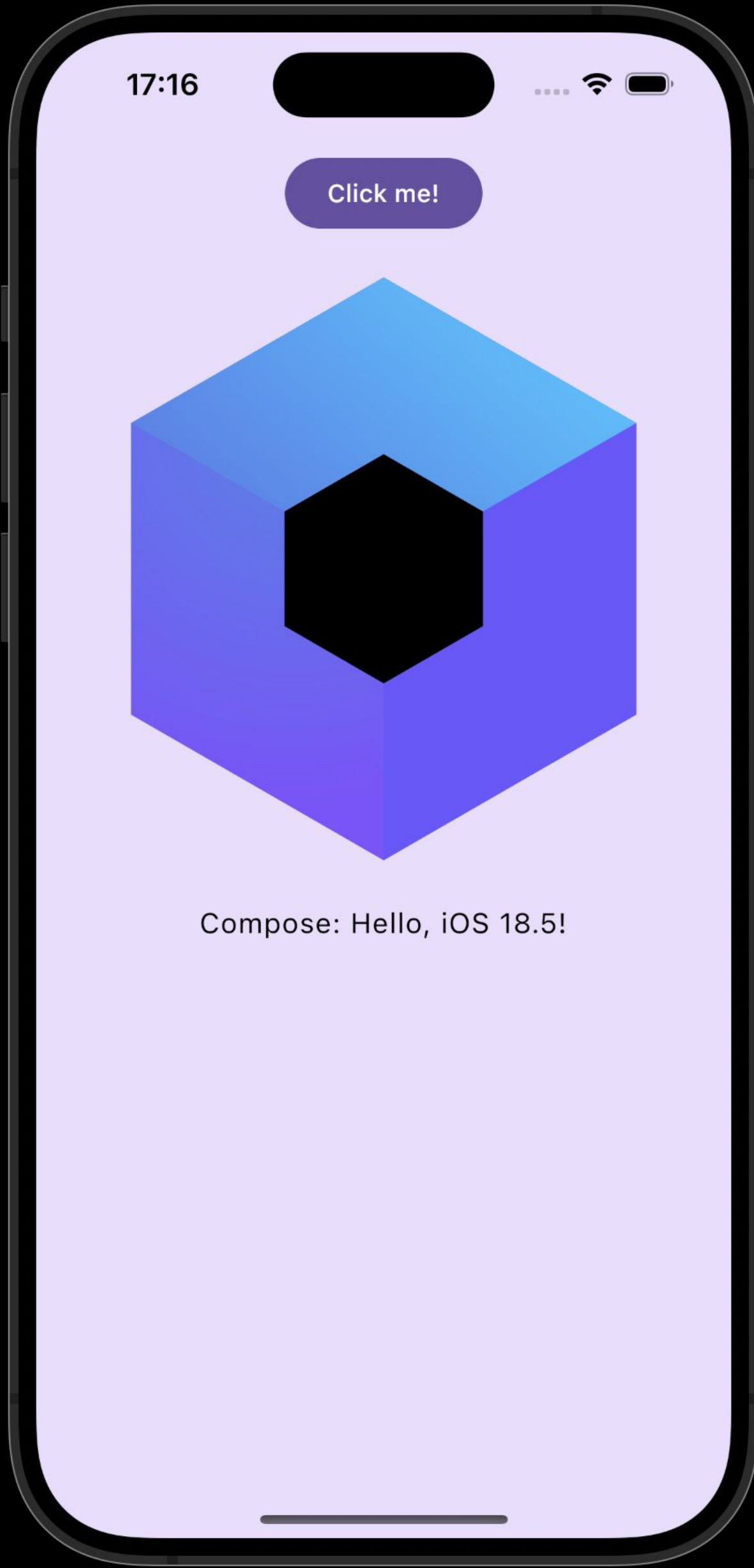
What is Kotlin and its ecosystem



Kotlin as a Platform

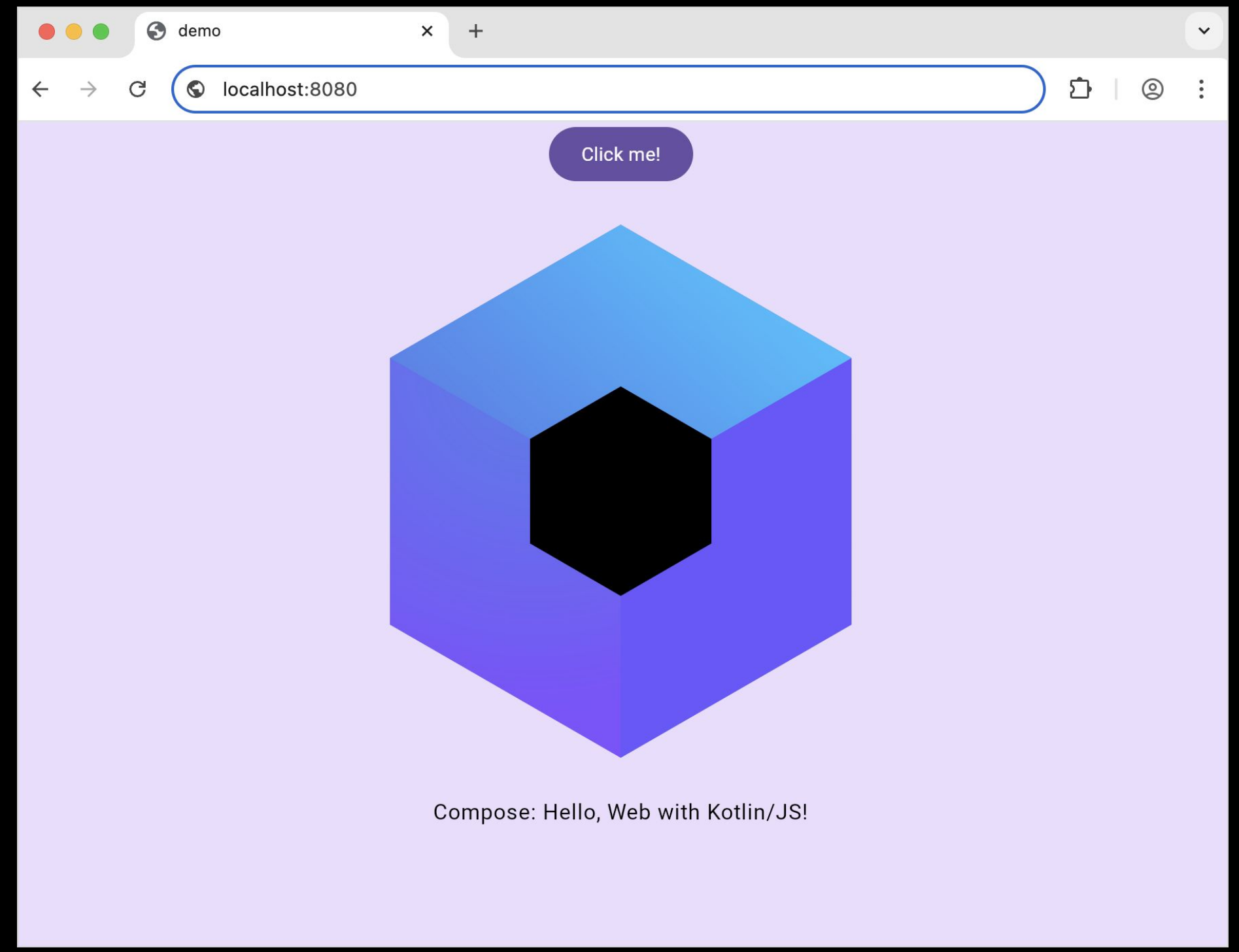
Kotlin Ecosystem

- Kotlin for Data Science
- Frameworks (Ktor, etc.)
- Compose Multiplatform



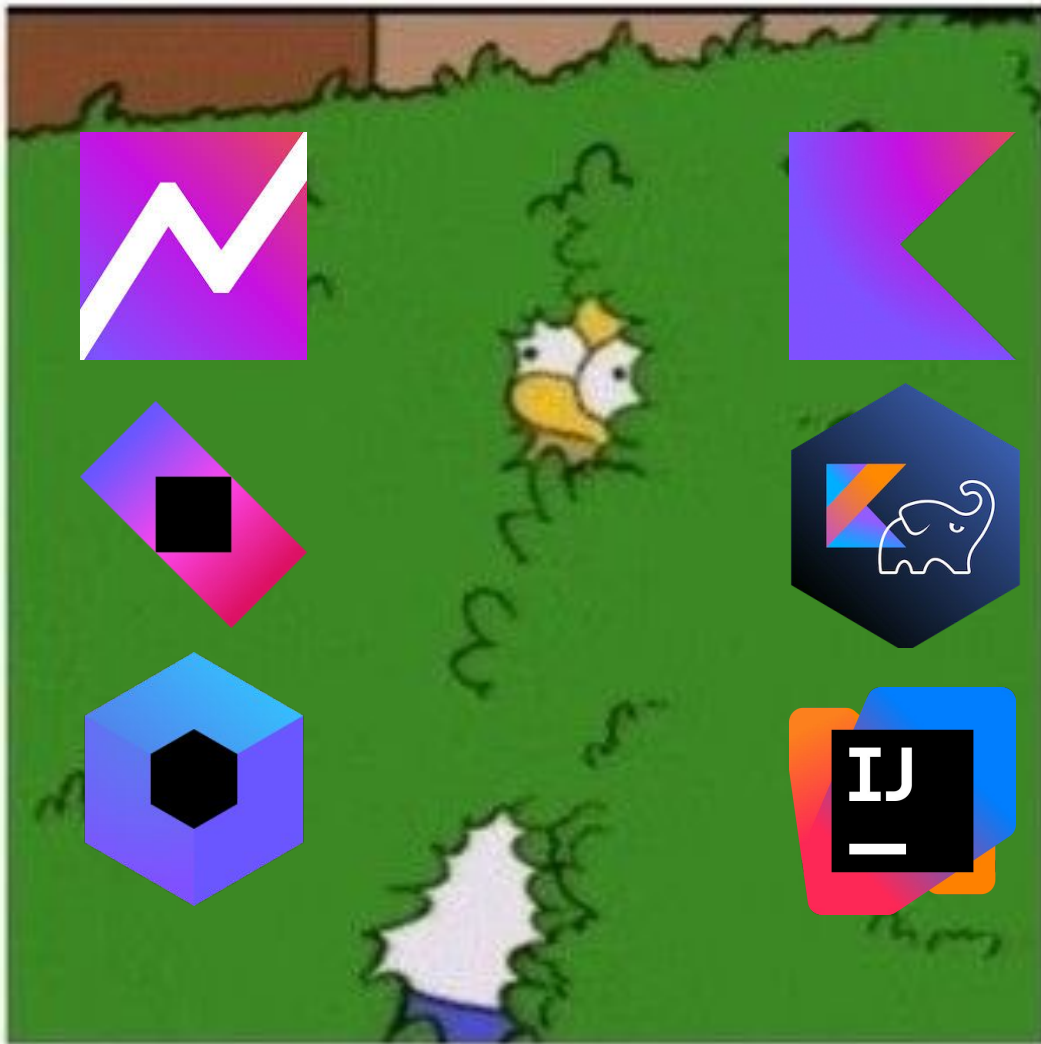
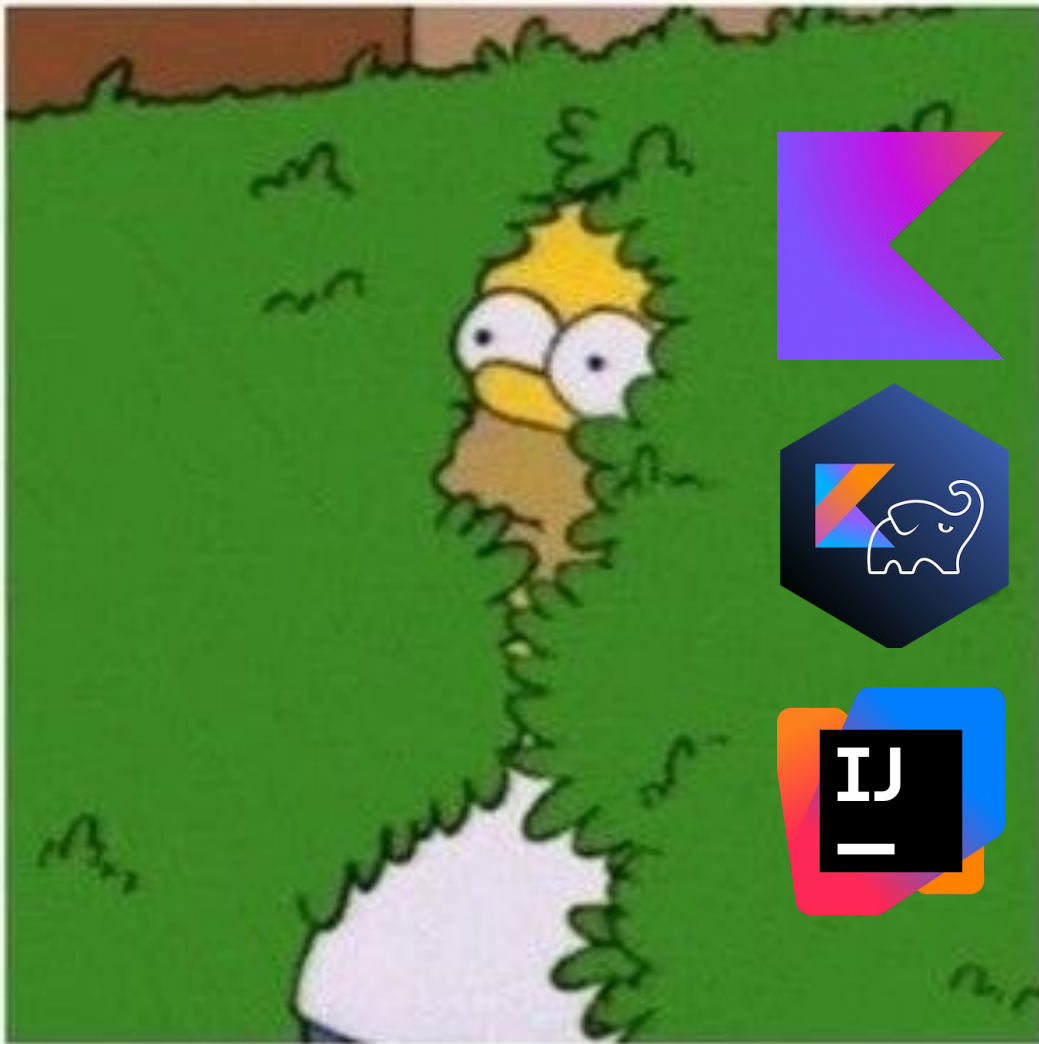
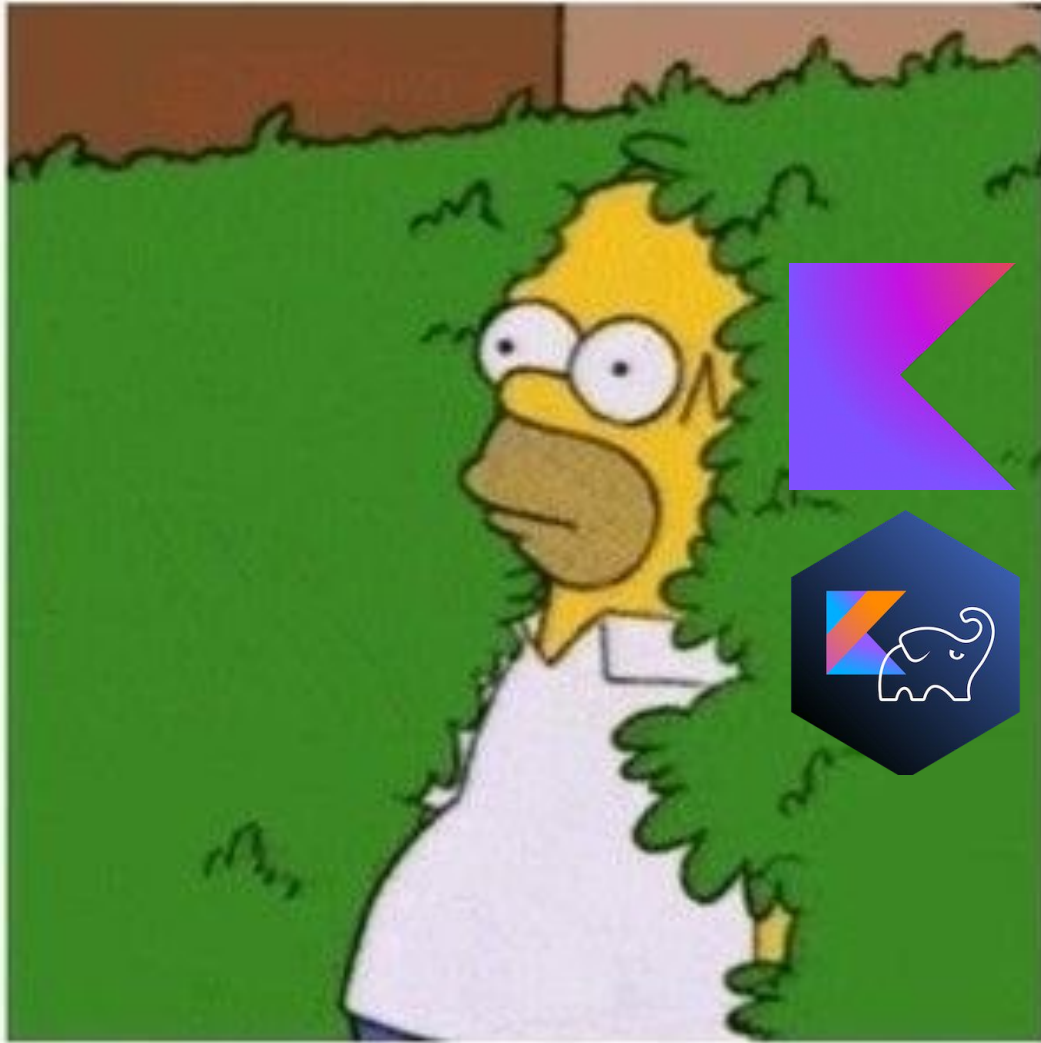
Desktop

Web

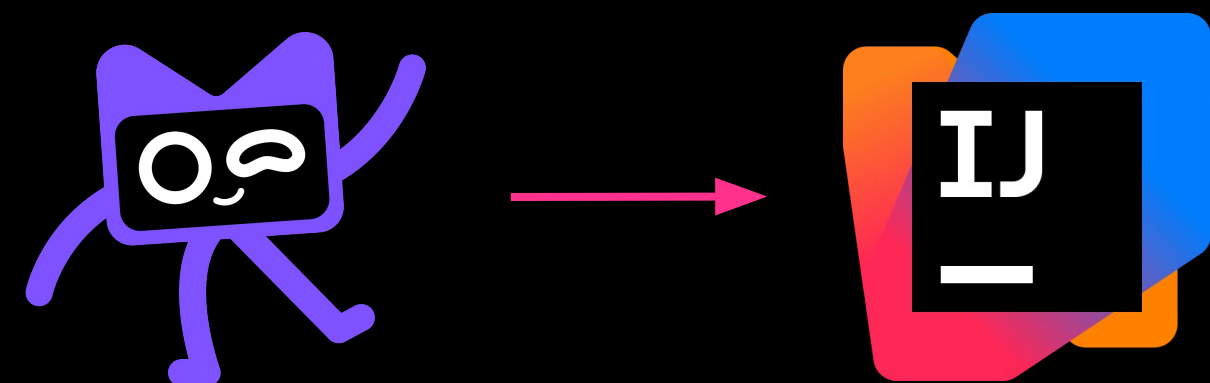


Mobile

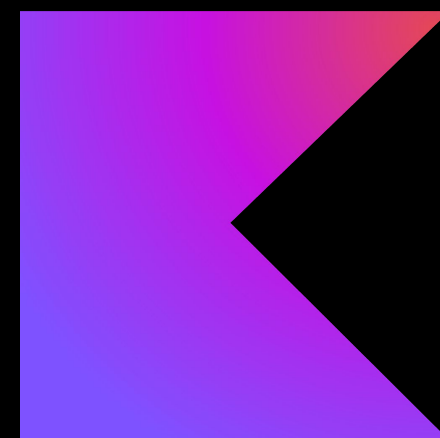
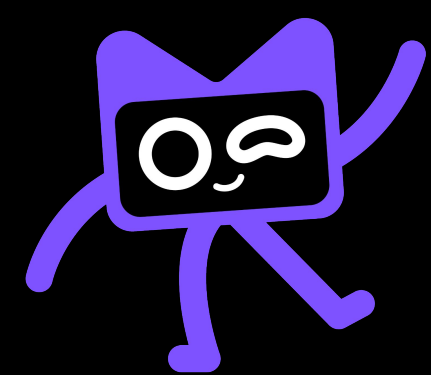
Compose Multiplatform

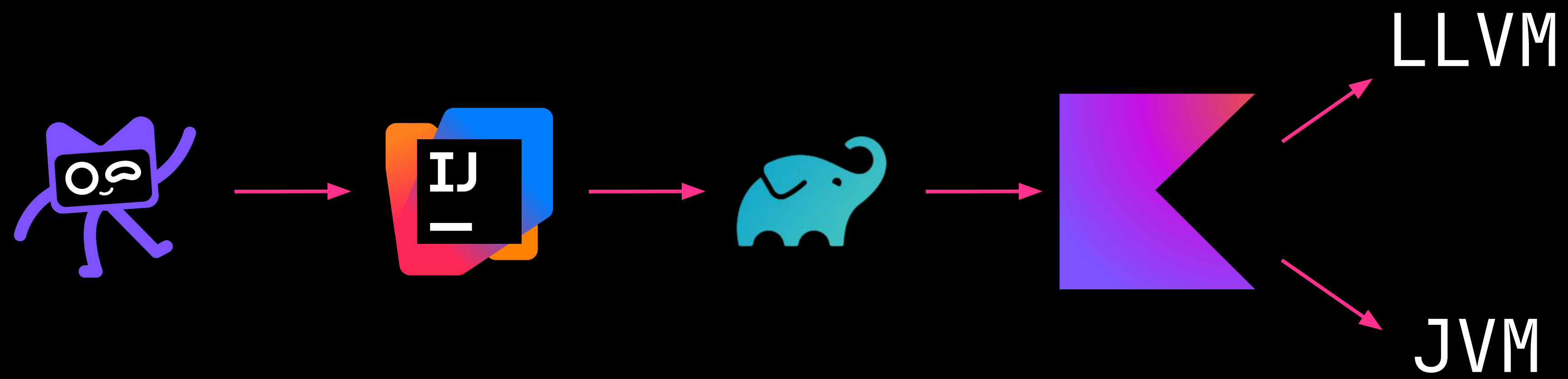


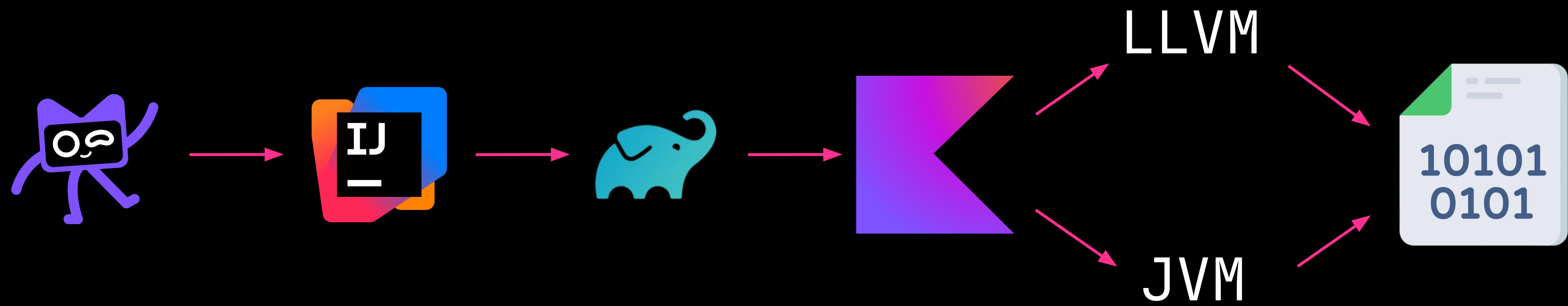
**From users to machine
code**









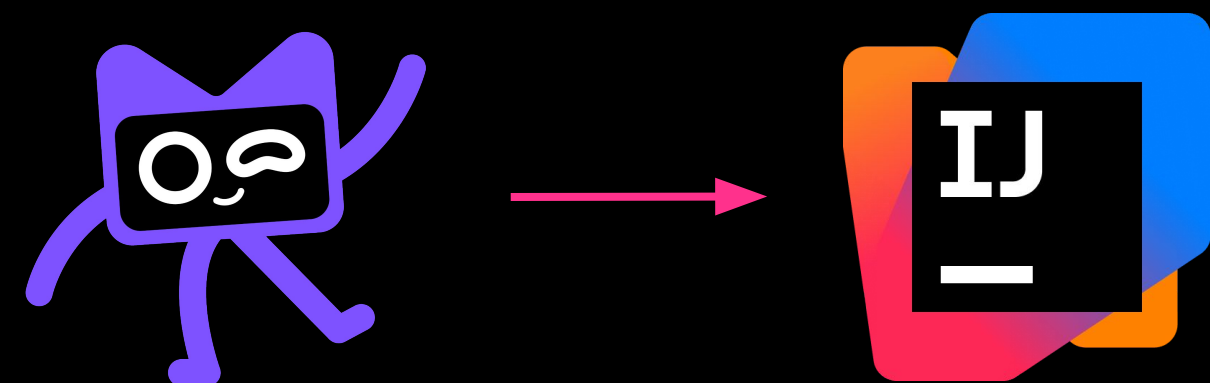


The screenshot shows the IntelliJ IDEA interface for a project named 'kotlin_demo_project'. The top toolbar includes icons for running code, debugging, and other IDE functions. The code editor displays a Kotlin file with the following content:

```
1 package org.example
2
3 To Run code, press ^ R or click the ▶ icon in the
4 gutter.
5 ▶ fun main() {
6     val name = "Kotlin"
7
8     Press \n ↵ Return with your caret at the
9     highlighted text to see how IntelliJ IDEA suggests
10    fixing it.
11
12    println("Hello, " + name + "!")
13
14    for (i in 1..5) {
15        Press ^ D to start debugging your code. We
16        have set one ● breakpoint for you, but you can
17        always add more by pressing ⌘ F8.
18
19        println("i = $i")
20    }
21 }
```

Line 14 has a red breakpoint icon (a solid red circle) in the gutter. Line 15 has a yellow lightbulb icon in the gutter, indicating an IDE suggestion. The code is color-coded: keywords like 'package', 'fun', 'val', 'println', and 'for' are in orange, while identifiers and literals are in green or blue.

Entry point to the ecosystem



Kotlin IDE Plugin

Perception of quality

Kotlin IDE Plugin

Perception of quality

- Project opens **without any red code**

Current File ▾



Main.kt ×



⚠ Kotlin not configured

Configure

Ignore



To Run code, press or click the icon in the gutter.

! 9 ⚠ 1 ^ ▾

```
3 fun main() {  
4     val name = "Kotlin"
```

Press with your caret at the highlighted text to see how IntelliJ IDEA suggests fixing it.

```
7     println("Hello, " + name + "!!")  
8  
9     for (i in 1..5) {
```

Press to start debugging your code. We have set one ● breakpoint for you, but you can always add more by pressing .

```
● 12     println("i = $i")  
13  
14 }
```

JetBrains

Kotlin IntelliJ Plugin

Public

12 projects, 2 members

Search projects

View members

Insights

Ultimate teamcity config

master 21h ago

No problems

Coverage not enabled

License audit not enabled

firebase-kotlin-sdk_mpp

master 31 Aug 23:00

No problems

Coverage not enabled

License audit not enabled

SQLite

master 20h ago

No problems

Coverage not enabled

License audit not enabled

tbe-server kts files

master 21h ago

No problems

Coverage not enabled

License audit not enabled

tbe-server

master 21h ago

2 problems

Coverage not enabled

License audit not enabled

ktor kts files

master 20h ago

No problems

Coverage not enabled

License audit not enabled

ktor

master 20h ago

5 problems

Coverage not enabled

License audit not enabled

kotlinx.serialization

master 20h ago

No problems

Coverage not enabled

License audit not enabled

kotlinx.coroutines kts files

master 20h ago

No problems

Coverage not enabled

License audit not enabled

kotlinx.coroutines

master 20h ago

5 problems

Coverage not enabled

License audit not enabled

kotlin repo kts files

master 21h ago

No problems

Coverage not enabled

License audit not enabled

intellij ultimate master

master 20h ago

31 problems

Coverage not enabled

License audit not enabled

Red Code tests

Kotlin IDE Plugin

Perception of quality

- Project opens **without any red code** ✓
⇒ **Red Code** tests

Kotlin IDE Plugin

Perception of quality

- Project opens **without any red code** 
 - ⇒ **Red Code** tests
- Most important Kotlin **user scenarios** work in a live IDE

Kotlin IDE Plugin

User Workflow scenarios

Scenarios

- Kotlin + Java developer
- Kotlin Android developer
- Kotlin Spring developer
- Kotlin Multiplatform developer
- Kotlin Fullstack (JVM + JS) developer
- and around 20 more

Kotlin IDE Plugin

User Workflow scenarios

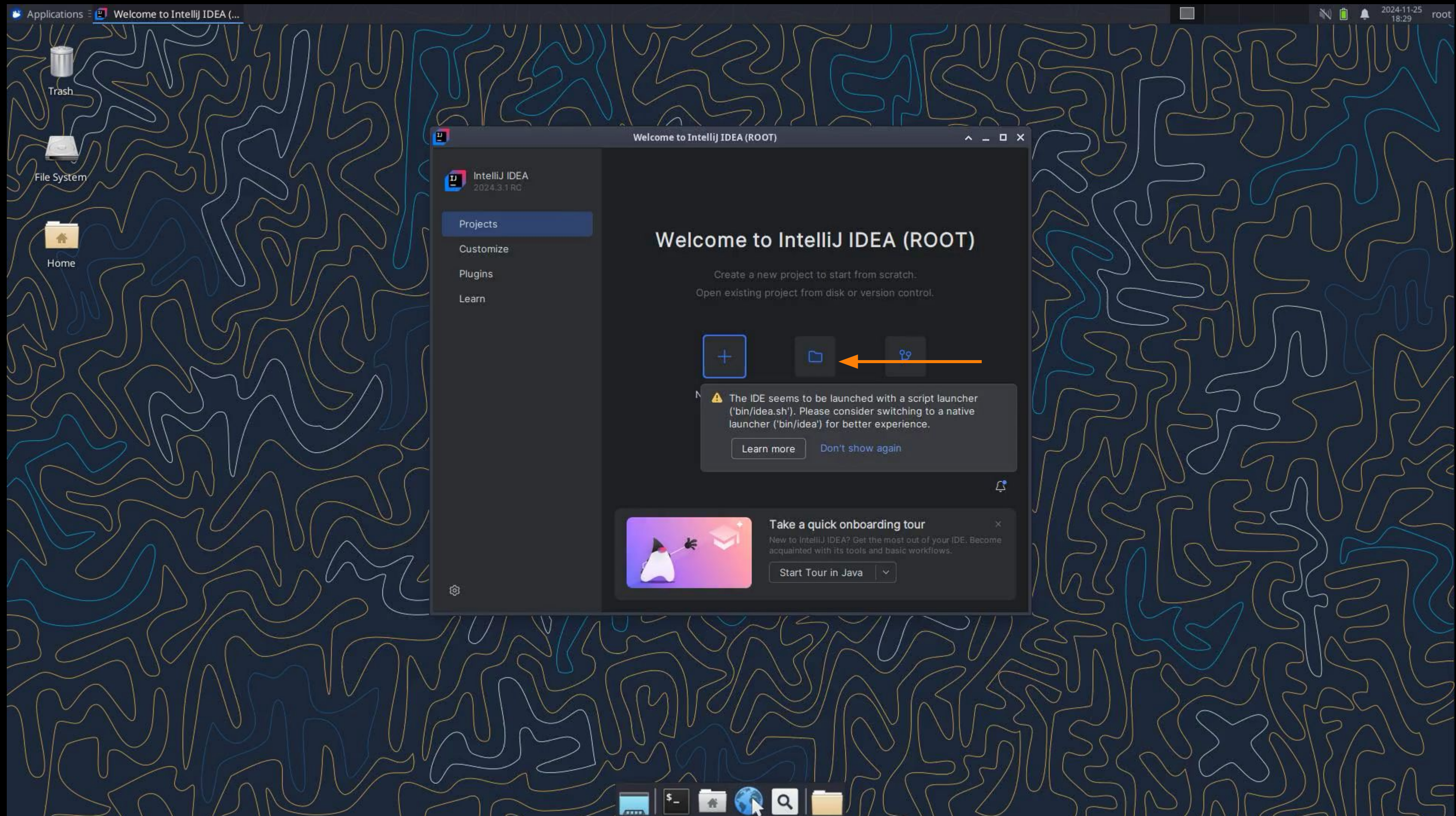
Scenarios

- Kotlin + Java developer
- Kotlin Android developer
- Kotlin Spring developer
- Kotlin Multiplatform developer
- Kotlin Fullstack (JVM + JS) developer
- and around 20 more

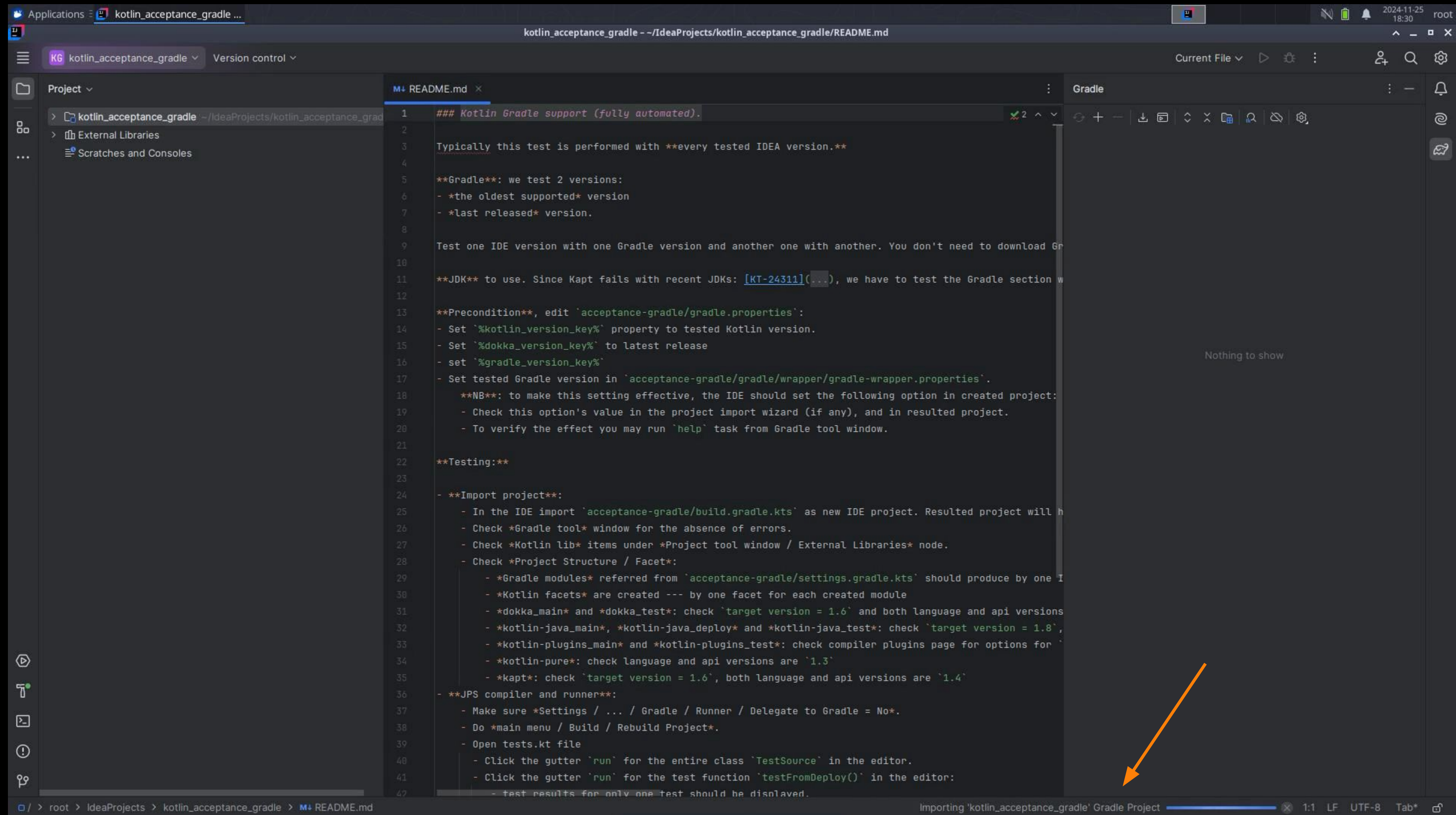
Use Cases

Typical developer session

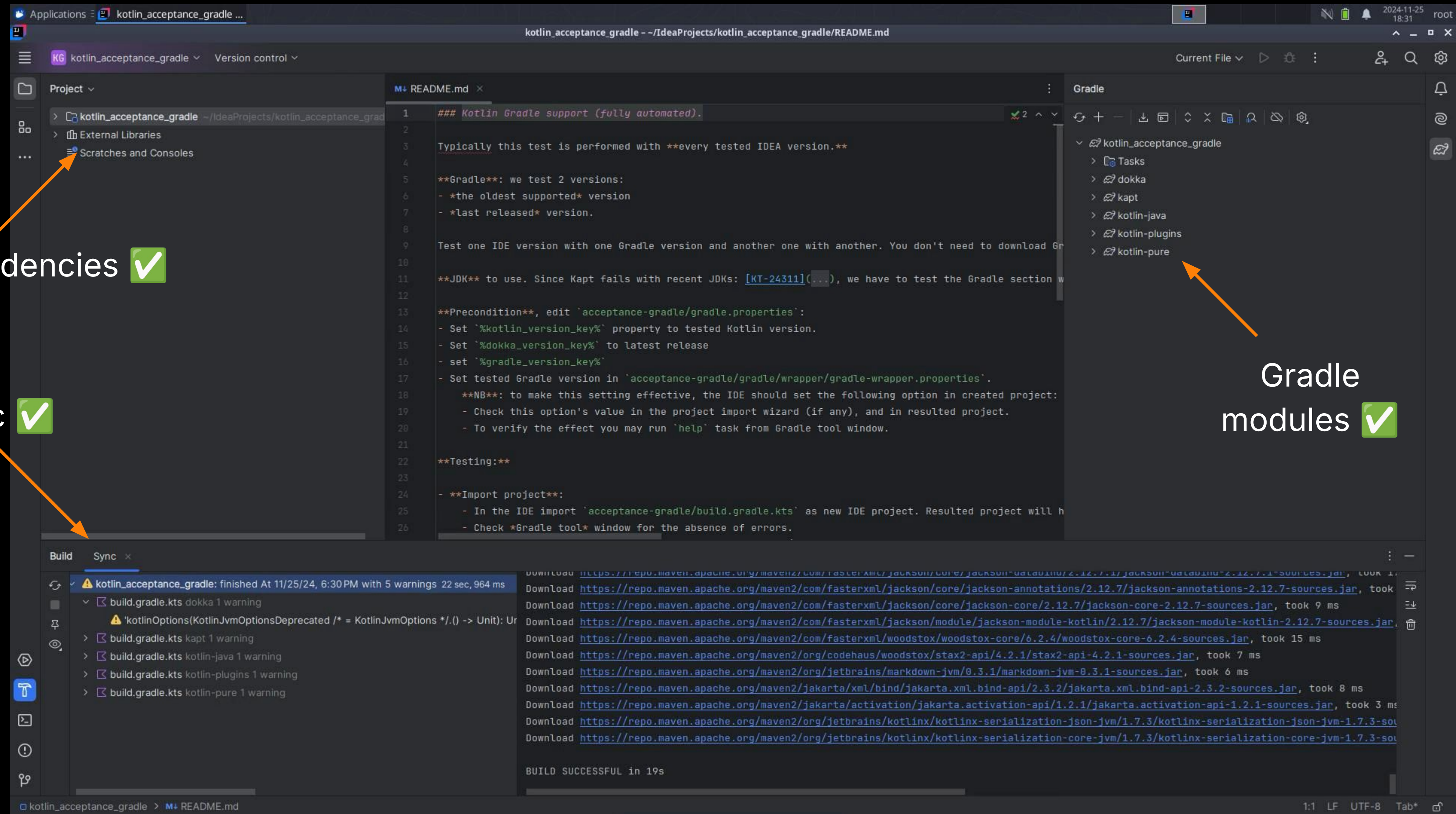
- Open project → assert import
- Open .kt file → assert analysis
- Navigate / refactor → assert K/IDE features
- Run project → assert run configurations
- Debug project → assert debugging features
- Run tests → assert Test Source features
- + unique cases for platforms



Developer opens a project



Let's wait for the project import



Project dependencies ✓

Gradle sync ✓

Gradle modules ✓

We assert import success

```

/**
 * https://jetbrains.team/p/ivua/code/ui-tests-data/files/projects/kotlin_acceptance_maven/README.md
 * Kotlin Maven support.
 */
@TestFactory @ alexander.khlopotov +5
@Execution(ExecutionMode.SAME_THREAD)
fun kotlinMavenAcceptance() = kotlinScenario("Kotlin Maven Acceptance") {
    startIdea

    kotlinTest("Open project test") {
        openProject(PROJECT_NAME)
    }

    kotlinTest("Check facets") {
        kotlinProject {
            mavenAcceptanceFacets
        }
    }

    kotlinTest("Check dependencies") {
        kotlinProject {
            mavenAcceptanceDependencies
        }
    }
}

```

How it looks in the framework code

Created by Alexander Khlopotov over 1 year ago Updated by Sergey Moryahin over 1 year ago

☆ K/N libraries don't load and index after project import

kotlin-regression ✕

multiplatform ✕

KED-qa-autotest-covered ✕

```
22
23 @OptIn(ExperimentalForeignApi::class)
24 fun main() {
25     println("Hello from ${SystemParameters().getOS()} target!")
26     println("Native library call: ${pthread_self()}")
27     val file = platform.posix.fopen("run_results", "a")
28     try {
29         platform.posix.fputs("mac.main\n", file)
30     } finally {
31         platform.posix.fclose(file)
32     }
33 }
```

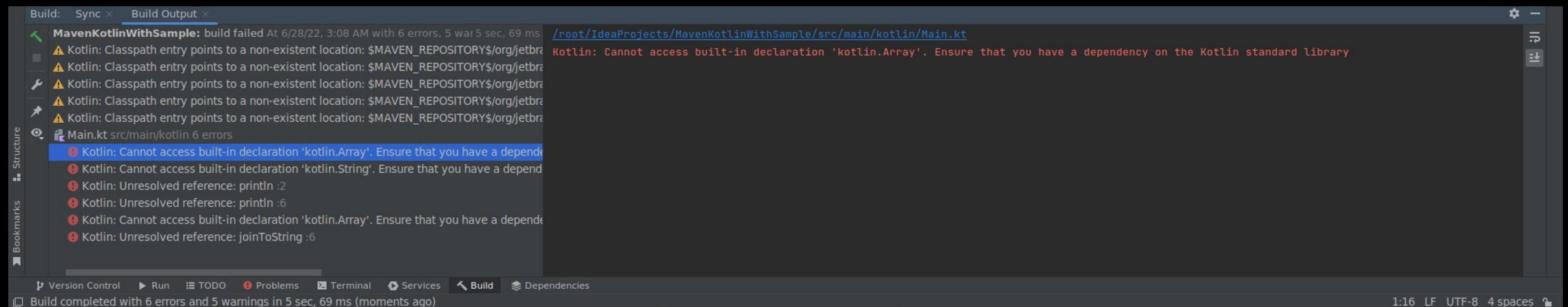
Found Bugs Examples

Created by Artur Parpibaev about 3 years ago Updated by Vladimir Naumenko 9 months ago

★ Maven / JPS wizard project without any IDEA / Maven caches fails to build

kotlin-internal-user ✕

KED-qa-autotest-covered ✕



Found Bugs Examples

Created by Alexander Khlopotov over 1 year ago Updated by Sergey Moryahin over 1 year ago

☆ **K/N libraries don't load and index after project import**  

kotlin-regression ✕

multiplatform ✕

KED-qa-autotest-covered ✕

Created by Artur Parpibaev about 3 years ago Updated by Vladimir Naumenko 9 months ago

★ **Maven / JPS wizard project without any IDEA / Maven caches fails to build**

kotlin-internal-user ✕

KED-qa-autotest-covered ✕

Created by Sergey Moryahin 3 months ago Updated by Sergey Moryahin 25 days ago

☆ **JS MPP. Breakpoints are not hit on debugging**     

kotlin-regression ✕

multiplatform ✕

k2 ✕

KED-qa-autotest-covered ✕

Found Bugs Examples

Kotlin IDE Plugin

Perception of quality

- Project opens **without any red code** ✓
 - ⇒ **Red Code** tests
- Most important Kotlin **user scenarios** work in a live IDE ✓
 - ⇒ **User Workflow** tests

Kotlin IDE Plugin

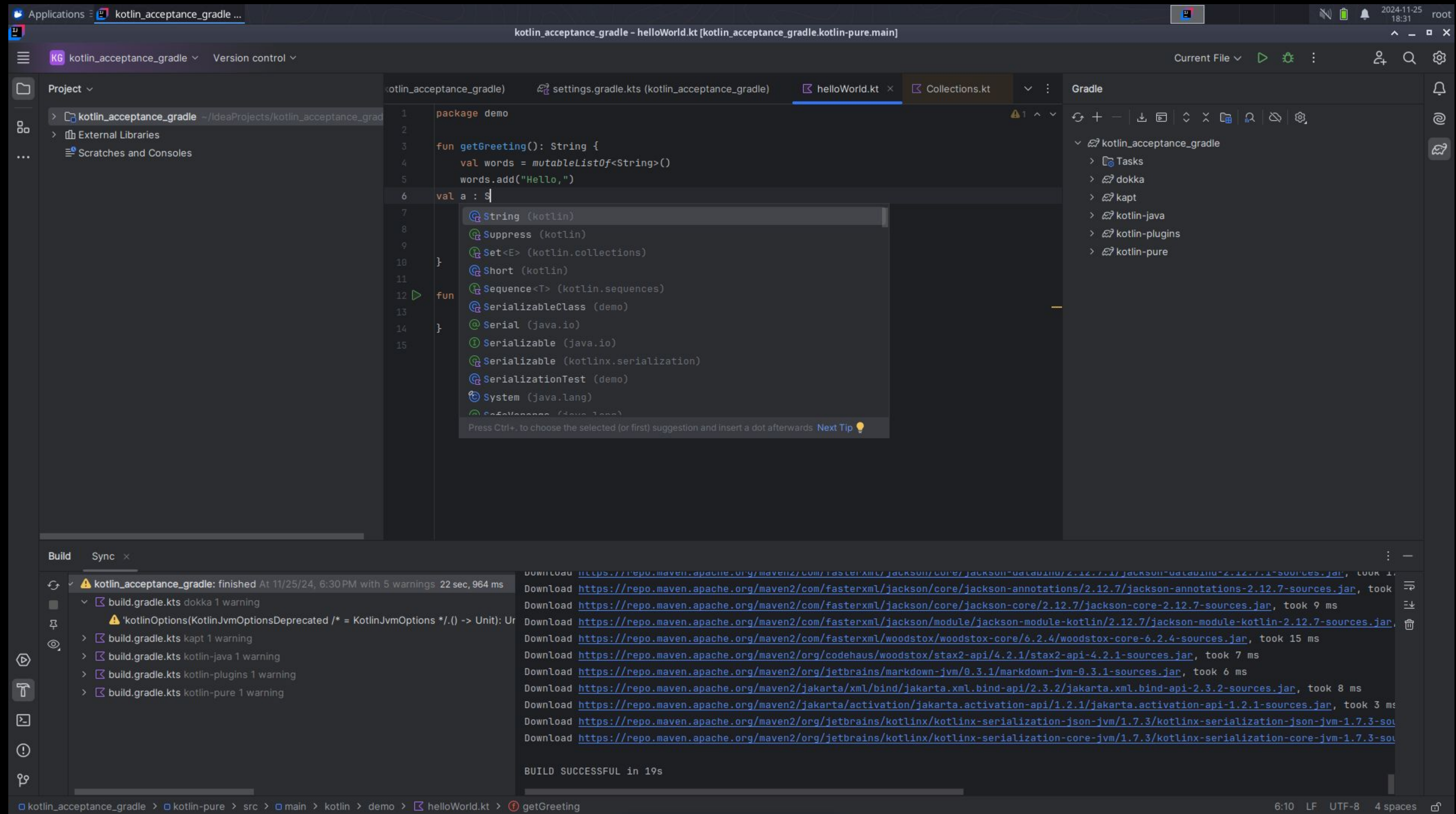
Perception of quality

- Project opens **without any red code** ✓
 - ⇒ **Red Code** tests
- Most important Kotlin **user scenarios** work in a live IDE ✓
 - ⇒ **User Workflow** tests
- **Features with UI** work in a live IDE

Kotlin IDE Plugin

UI testing of the UI-rich features

- More than 1000 UI-tests
 - Navigation
 - Code Completion
 - Quick Documentation
 - Find Usages
 - Refactorings
 - Code Analysis
 - Project Wizards
 - Convert Java to Kotlin
 - Debuggers (JVM, JS, Wasm, Native, Coroutines)
 - Kotlin Bytecode viewer
 - Add Kotlin module to project
 - ...



Code Completion check

```
fun getGreeting(): String {  
    val words = mutableListof<String>()  
    words.add("Hello,")  
}
```

```
val a : S
```

- String (kotlin)
- Suppress (kotlin)
- Set<E> (kotlin.collections)
- Short (kotlin)
- Sequence<T> (kotlin.sequences)
- SerializableClass (demo)
- @Serial (java.io)
- Serializable (java.io)
- Serializable (kotlinx.serialization)
- SerializationTest (demo)
- System (java.lang)
- SafeVersion (java.lang)

Press Ctrl+. to choose the selected (or first) suggestion and insert a dot afterwards [Next Tip](#) 

Wait for suggestions
to load 

Assert completion

```
fun getGreeting(): String {  
    val words = mutableListOf<String>()  
    words.add("Hello,")  
}  
val a : S
```

String (kotlin)
Suppress (kotlin)
Set<E> (kotlin.collections)
Short (kotlin)
Sequence<T> (kotlin.sequences)
SerializableClass (demo)
Serial (java.io)
Serializable (java.io)
Serializable (kotlinx.serialization)
SerializationTest (demo)
System (java.lang)
SafeVarargs (java.lang)

Press Ctrl+. to choose the selected (or first) suggestion and insert a dot afterwards Next Tip

← Standard library ✓

← Project class ✓

← Java platform ✓

← Wait for suggestions to load ✓

Assert completion options & success

Kotlin IDE Plugin

Perception of quality

- Project opens **without any red code** ✓
 - ⇒ **Red Code** tests
- Most important Kotlin **user scenarios** work in a live IDE ✓
 - ⇒ **User Workflow** scenarios
- **Features with UI** work in a live IDE ✓
 - ⇒ Feature-based **UI-tests**



Kotlin Build Toolchain

How can we assure a build system works well?

... probably by testing Kotlin Gradle Plugin? Or the whole build tooling for Kotlin?

Kotlin Build Toolchain

How can we assure a build system works well?

... probably by testing Kotlin Gradle Plugin? Or the whole build tooling for Kotlin?



Kotlin Build Tools QA Team

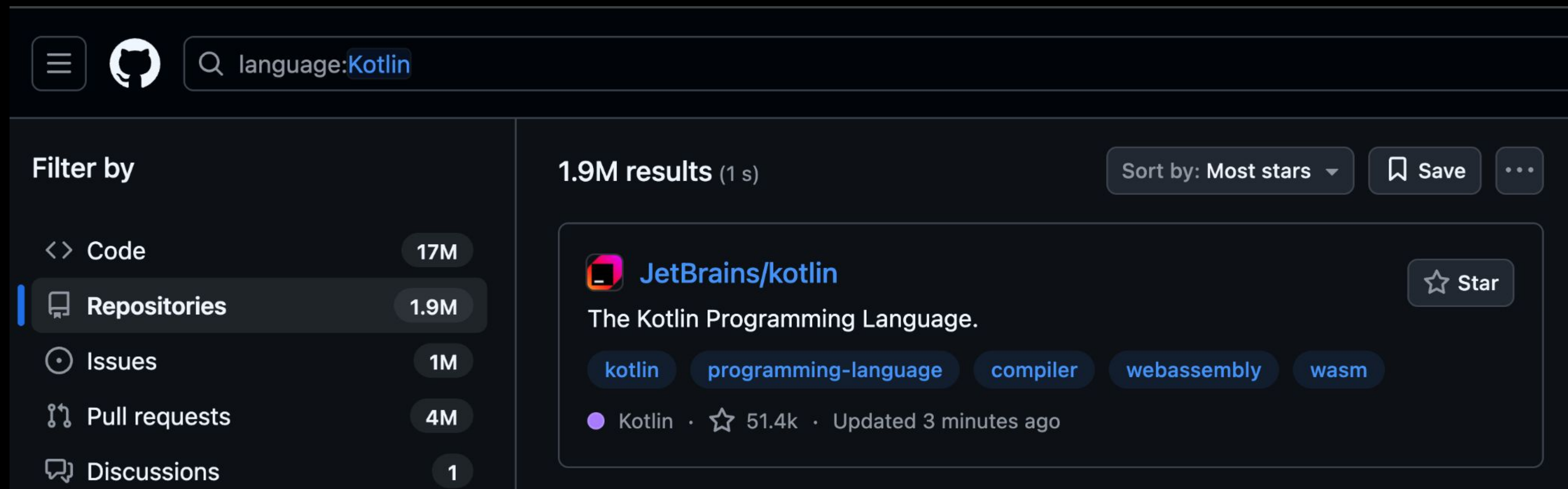
Kotlin Build Toolchain

And how can we assure we build **any** code smoothly...

... probably by building the entire world*?

*Kotlin projects in the most popular ways of using

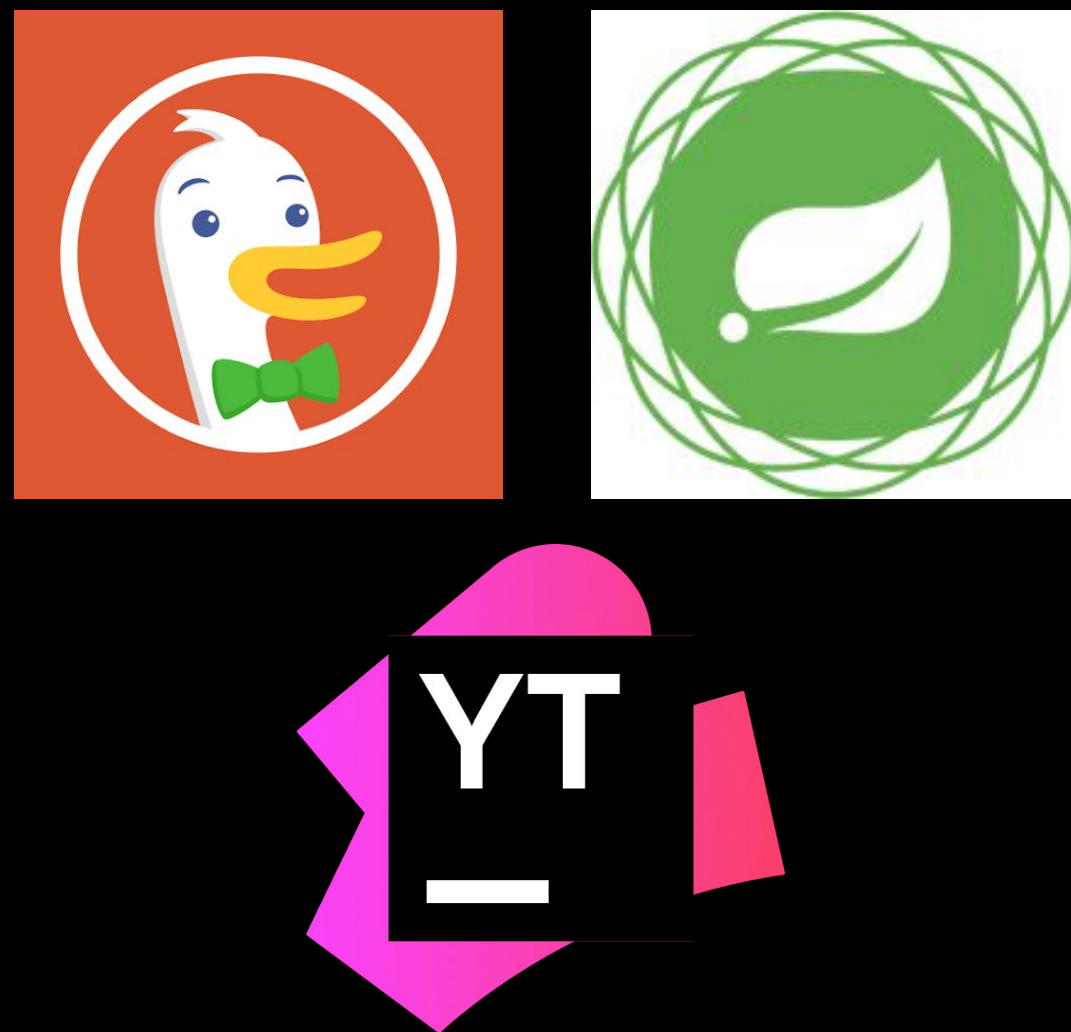
User Projects



Does not look realistic

User Projects

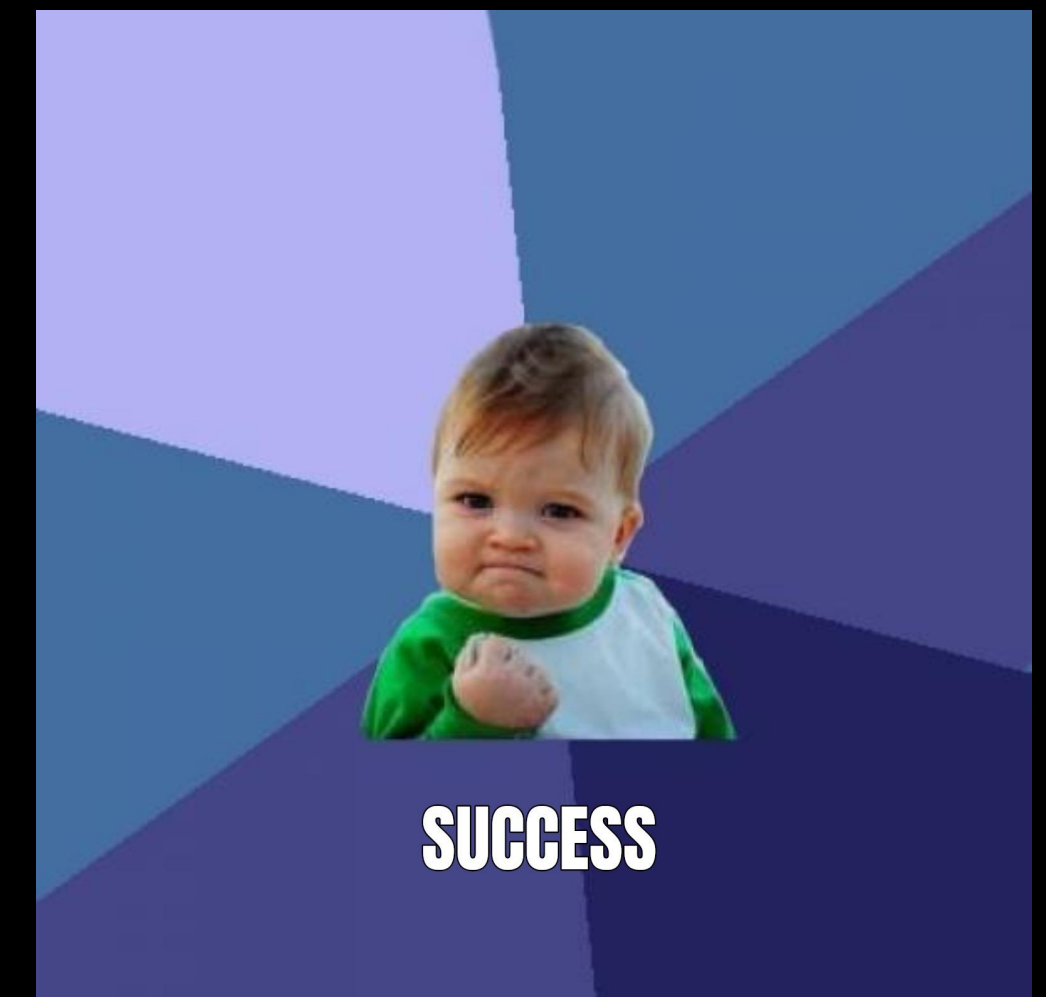
The concept



Take popular apps



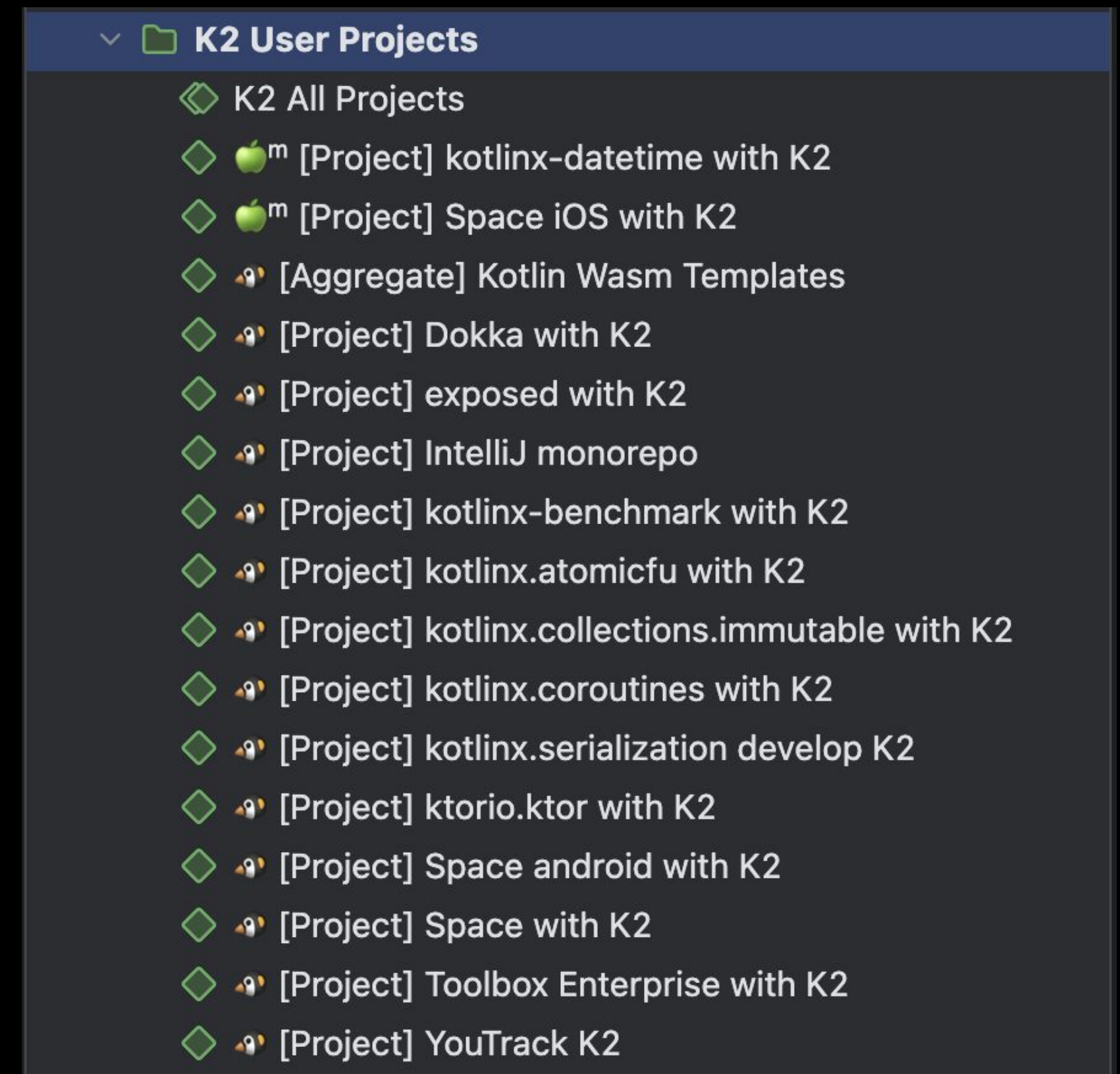
Build with the latest dev Kotlin



Catch all the bugs

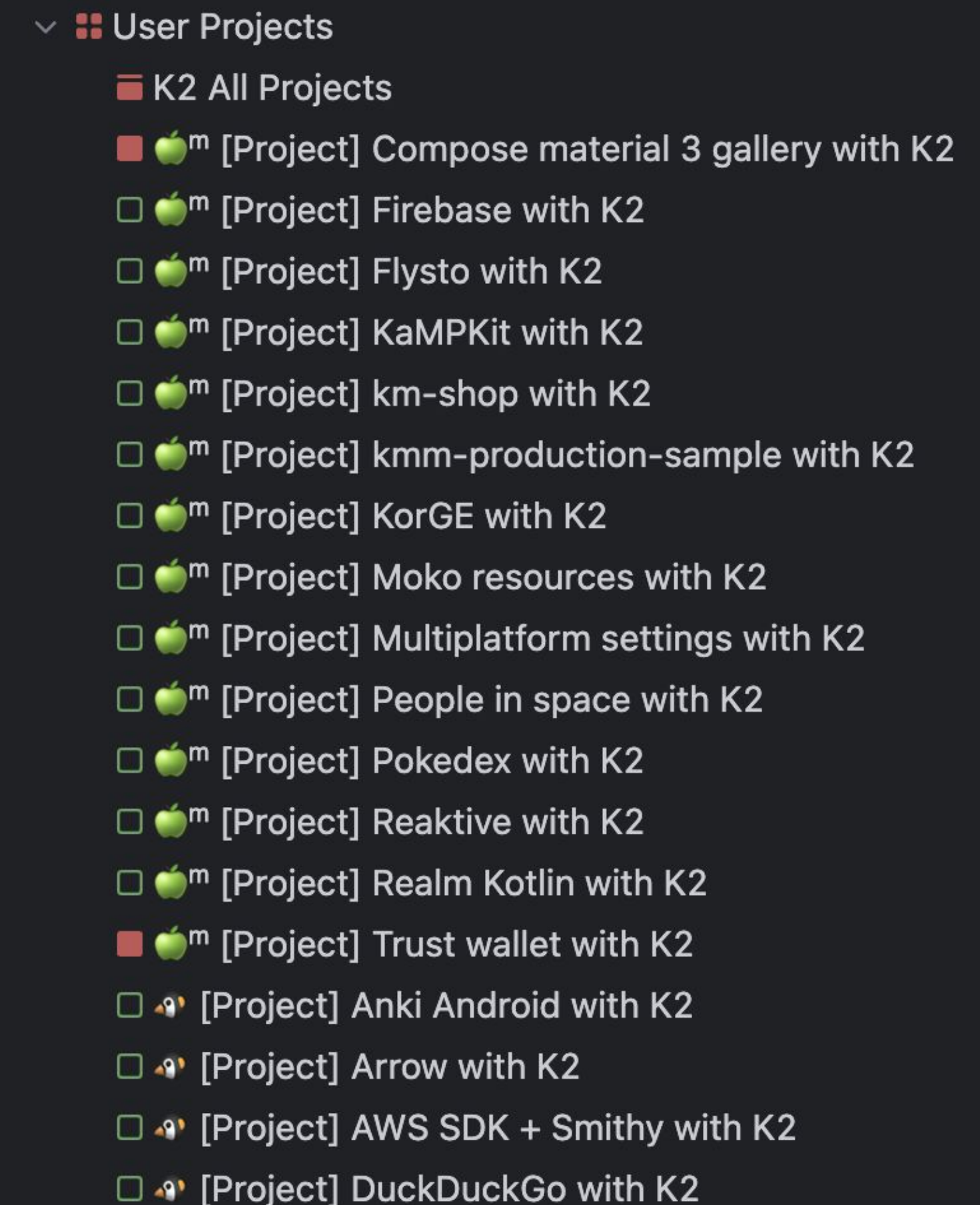
User Projects

- Important internal JetBrains Kotlin projects
 - IntelliJ IDEA
 - Kotlin
 - Kotlin Libraries
 - Teamcity
 - YouTrack
 - etc.



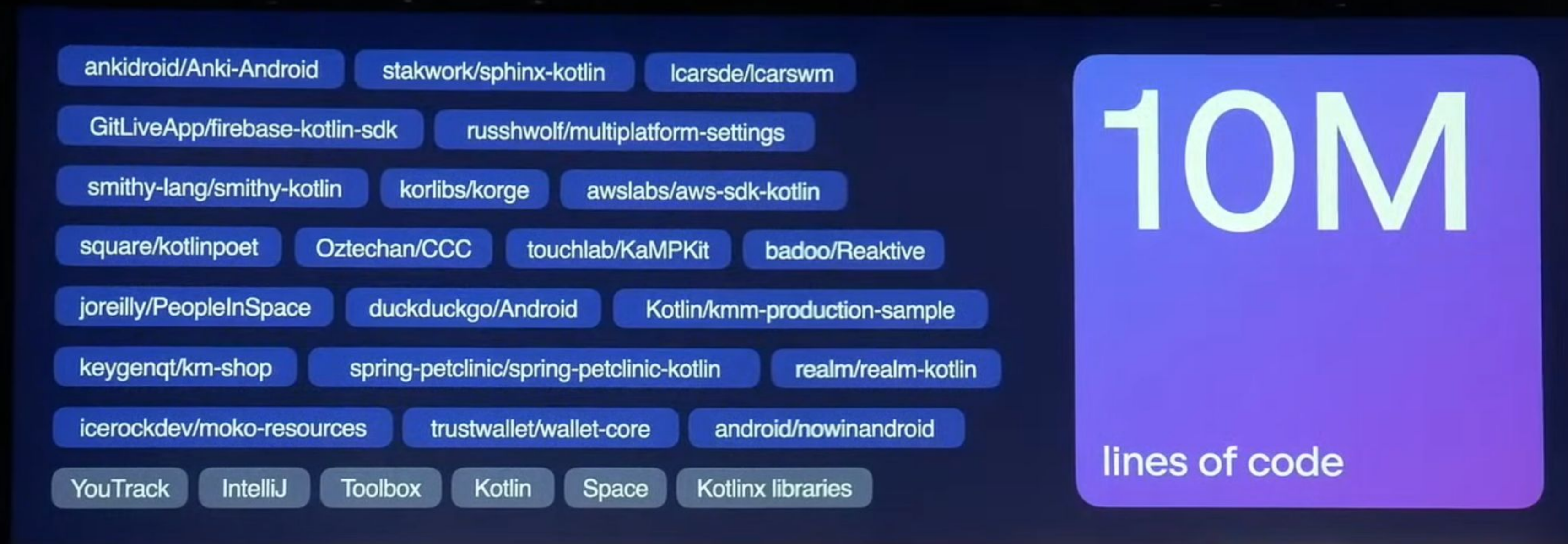
User Projects

- Important internal JetBrains Kotlin projects
- Open-source largest and diverse Kotlin projects
 - JVM
 - JS
 - Mobile Multiplatform
 - iOS
 - Android
 - Native Desktop



User Projects

- Important internal JetBrains Kotlin projects
- Open-source largest and diverse Kotlin projects



← Projects / Kotlin / Issues / ~~KT-56083~~

Enter search request

Created by Mikhail Glukhikh over 2 years ago

Updated by Aleksandr Shalygin about 1 month ago

Visible to issue readers

☆ **K2: build ktor**

k2 × Compiler Core ×

Depends on

☆ **M** ~~KTOR-7025~~ K2: Stabilize flaky tests

☆ **N** ~~KTOR-7081~~ K1/K2: io.ktor.client.tests.WebSocketTest.testImmediateReceiveAfterConnect[wasmJs, node] fails with...

☆ **P** ~~KT-56023~~ Constant operations (e.g. division) are not constant in K2 (JS, Native)

☆ **P** ~~KT-56354~~ K2/MPP: unresolved references to library entities

☆ **P** ~~KT-56368~~ K2/MPP: compiler backend crash on missing actual declaration

☆ **P** ~~KT-57341~~ K2 MPP: Overload resolution ambiguity between generic and simple functions when kotlin.mpp.enableCompatibilityMetadataVariant...

☆ ~~KT-57343~~ K2, MPP: Unresolved reference to library methods

☆ **P** ~~KT-57364~~ K2/Native requires to have override equals function in expect class if the function is presented in actual class

☆ **P** ~~KT-57384~~ K2/MPP/Native: impossible to override Any::equals with non-external function

☆ **P** ~~KT-57532~~ K2: IrActualizer doesn't handle properties overloaded by extension receiver correctly

☆ **P** ~~KT-57568~~ K2: K2, Native reports overload resolution ambiguity

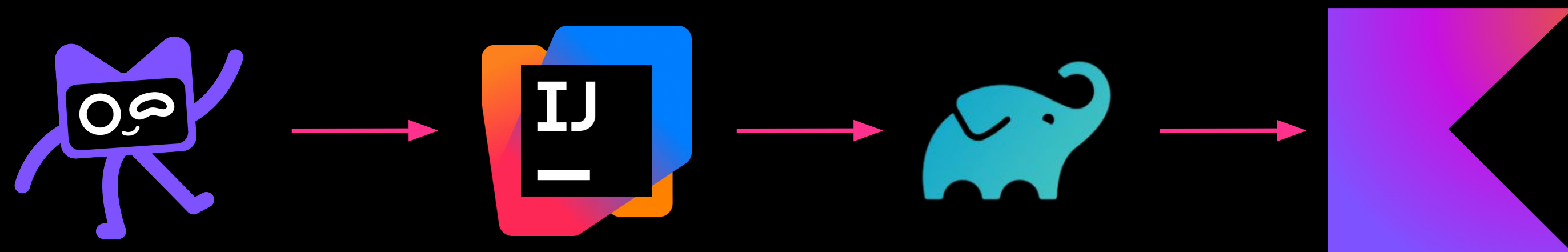
☆ **P** ~~KT-57649~~ K2 reports warning about incompatible upper bounds

☆ **P** ~~KT-57654~~ K2: Lambda with receiver deserialized as lambda without receiver during metadata compilation

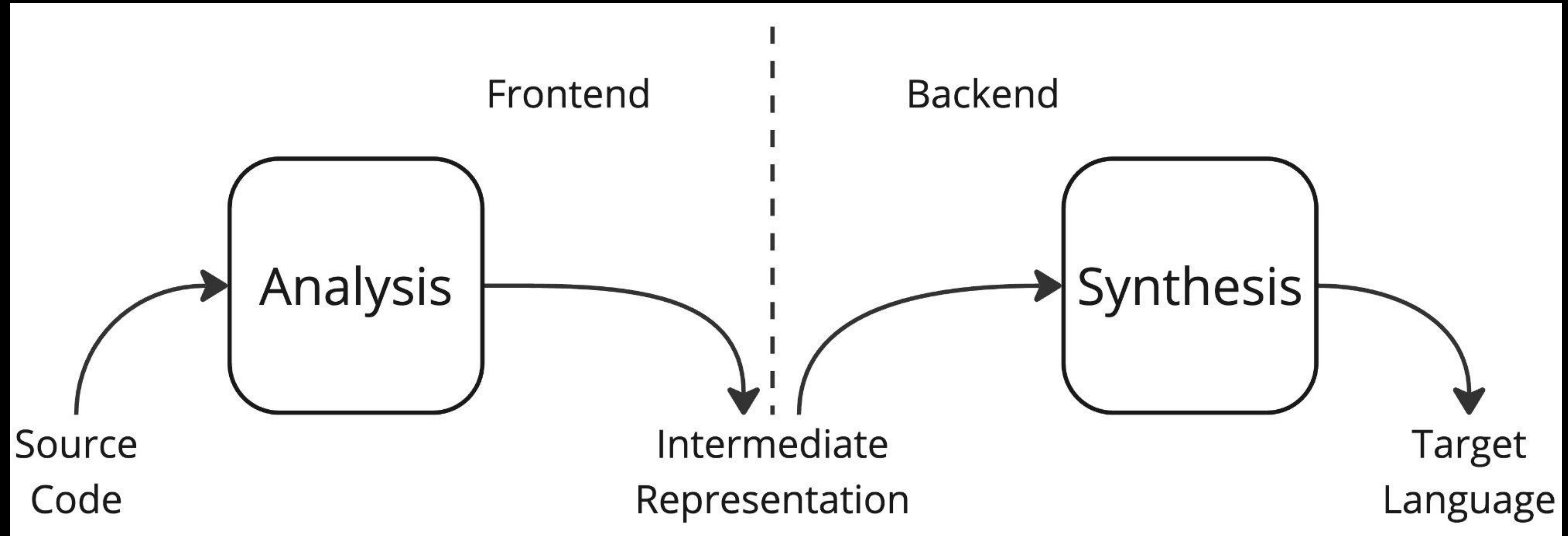
Over 40 major+ regressions was found

In 2 years

On a single project out of 50



Compiler



Compiler frontend

- Analysis (i.e. syntax highlighting)
- Language features
- Intermediate Representation (IR) generation

Created by Aleksandra Arsenteva over 1 year ago Updated by Aleksandra Arsenteva 11 months ago

👁 Visible to issue readers

Destructuring declaration shouldn't be possible in declaration in when ...

↔

The following code is green and runs successfully but does nothing:

```
1 data class Foo(val name: String)
2
3 fun main() {
4     val foo = Foo("John")
5     when (val (x) = foo) {
6         is String -> println("1")
7         is Foo -> println("2")
8         //else -> println(x)           //Unresolved reference 'x'.
9     }
10 }
```

Kotlin detected

According to the [spec](#) it should be forbidden:

The scope of this property is limited to the `when` expression, including both conditions and control structure bodies of the expression. As its form is limited to a simple "assignment-like" declaration with an initializer, this property does not allow getters, setters, delegation or destructuring.

Just a simple code snippet with a 🐛

Context parameters: "IllegalStateException: Cannot find variable a: R[kotlin/String] in local storage " when context from another local function is called

```
1 fun outerFun() {  
2     context(a: String)  
3     fun local1() { }  
4  
5     fun local2() {  
6         a  
7     }  
8 }
```

Kotlin detected

Result:

```
1 e: org.jetbrains.kotlin.util.FileAnalysisException: While analysing /Users/Aleksandra.Arsenteva/repositories/qa-materials-spac  
2     at org.jetbrains.kotlin.util.AnalysisExceptionsKt.wrapIntoFileAnalysisExceptionIfNeeded(AnalysisExceptions.kt:57)  
3     at org.jetbrains.kotlin.fir.FirCliExceptionHandler.handleExceptionOnFileAnalysis(Utils.kt:251)  
4     at org.jetbrains.kotlin.fir.backend.Fir2IrConverter.runSourcesConversion(Fir2IrConverter.kt:712)  
5     at org.jetbrains.kotlin.fir.backend.Fir2IrConverter.access$runSourcesConversion(Fir2IrConverter.kt:71)  
6     at org.jetbrains.kotlin.fir.backend.Fir2IrConverter$Companion.generateIrModuleFragment(Fir2IrConverter.kt:702)  
7     at org.jetbrains.kotlin.fir.pipeline.Fir2IrPipeline.runFir2IrConversion(convertToIr.kt:159)  
8     at org.jetbrains.kotlin.fir.pipeline.Fir2IrPipeline.convertToIrAndActualize(convertToIr.kt:126)  
9     at org.jetbrains.kotlin.fir.pipeline.ConvertToIrKt.convertToIrAndActualize(convertToIr.kt:97)  
10    at org.jetbrains.kotlin.fir.pipeline.ConvertToIrKt.convertToIrAndActualize$default(convertToIr.kt:72)  
11    at org.jetbrains.kotlin.cli.jvm.compiler.legacy.pipeline.JvmCompilerPipelineKt.convertToIrAndActualizeForJvm(jvmCompil
```

Just a simple code snippet and a compiler crash 🌟

Compiler backend

- JVM
 - Java interop
- Native
 - C, ObjC interops, Swift Export
 - Dealing with targets
 - Pict
 - It helps with the problem of unclear coverage while having a lot of flags — we trust pairwise testing
- WASM and JS
 - JS/TS interop

Compiler backend

<https://youtrack.jetbrains.com/issue/KT-56464/K-N-Allow-HiddenFromObjC-for-classes>

K/N: Allow HiddenFromObjC for classes ...

Currently, `@HiddenFromObjC` annotation can be added only to functions and properties.

We want to use that annotation in Compose to "hide" the Composable functions.

A task 💪

Compiler backend

```
1 @file:OptIn(kotlin.experimental.ExperimentalObjCRefinement::class)
2
3 @HiddenFromObjC
4 class Hidden
5
6 fun Hidden.foo() = 5
```

Bash

Still a task 💪

Compiler backend

```
1 @file:OptIn(kotlin.experimental.ExperimentalObjCRefinement::class)
2
3 @HiddenFromObjC
4 class Hidden
5
6 fun Hidden.foo() = 5
```

Bash

Compilation:

```
1 → hiddenfromobjc_extension_bug /Users/Alexander.Zakharenko/Downloads/kotlin-native-prebuilt-macos-aarch64-1.9.0-Beta-162/bin/
2 → hiddenfromobjc_extension_bug /Users/Alexander.Zakharenko/Downloads/kotlin-native-prebuilt-macos-aarch64-1.9.0-Beta-162/bin/
3 error: compilation failed: Shouldn't be exposed: deserialized class Hidden
```

A bug 🥰

Compiler backend

Autotests

```
import kotlin.test.*
```

```
@Test
```

```
fun addition() {  
    assertEquals(42, 40 + 2)  
}
```

```
@Test
```

```
fun multiplication () {  
    assertEquals(42, 21 * 2)  
}
```

Compiler backend

Autotests

```
import kotlin.test.*

@Test
fun addition() {
    assertEquals(42, 40 + 2)
}

@Test
fun multiplication () {
    assertEquals(42, 21 * 2)
}
```

Gradle tasks and arguments: `:nativeCompilerTest` --init-script /opt/buildAgent/plugins/gradle-runner/scripts/init_since_8.gradle --init-script /opt/buildAgent/temp/agentTmp/build-scan-init.gradle -Pteamcity=true --no-watch-fs -Pkotlin.build.scan.url=XXX -Dscan.tag.kotlin-dev -Pkotlin.build.testRetry.maxRetries=0 -Pkotlin.build.isObsoleteJdkOverrideEnabled=true --parallel --continue -Pkotlin.native.enabled=true -Pkotlin.internal.native.test.nativeHome=/opt/buildAgent/work/17e640964edd2053/test_dist -Pkotlin.incremental=false -Pkotlin.internal.native.test.mode=TWO_STAGE_MULTI_MODULE -Pkotlin.internal.native.test.optimizationMode=DEBUG -Pkotlin.internal.native.test.cacheMode=STATIC_EVERYWHERE -Pkotlin.internal.native.test.gcType=CMS -Pkotlin.internal.native.test.gcScheduler=ADAPTIVE -Pkotlin.internal.native.test.alloc=CUSTOM -Pkotlin.internal.native.test.target=ios_simulator_arm64 -Pbuild.number=2.1.0-dev-8366 -Pkotlin.native.tests.tags=frontend-fir -Dorg.gradle.daemon=false -s -b build.gradle.kts

Compiler backend

Autotests

- Different aggregate test configurations
- Tagged tests
- Optimized test configurations
 - Pairwise testing as we have a lot of compiler flags — for each pair of input parameters, tests all possible discrete combinations of those parameters
 - Coverage is clarified at the level of configurations
 - Less tests
 - Some “must have” configurations on top

Compiler backend

Autotests — PICT

Mode: TwoStage

Optimization Mode: Debug, Optimized

Cache Mode: No, StaticOnlyDist, StaticEverywhere, StaticPerFile

GC Type: PMCS (50), CMS

GC Scheduler: Adaptive

Allocator: Custom

Thread State Checker: Disabled

Target: IOS_SIMULATOR_ARM64 (50), MINGW_X64, LINUX_X64

Test Set: onlyK2

Parameter Name

Possible Values

Weight

Conditional Constraints

```
IF [Cache Mode] <> "No" THEN [Optimization Mode] = "Debug";
IF [Cache Mode] = "No" AND [Optimization Mode] = "Optimized" THEN [Target] <> "MINGW_X64";
IF [Target] = "MINGW_X64" THEN [Optimization Mode] <> "Optimized" AND [Cache Mode] = "No";
```

Compiler backend

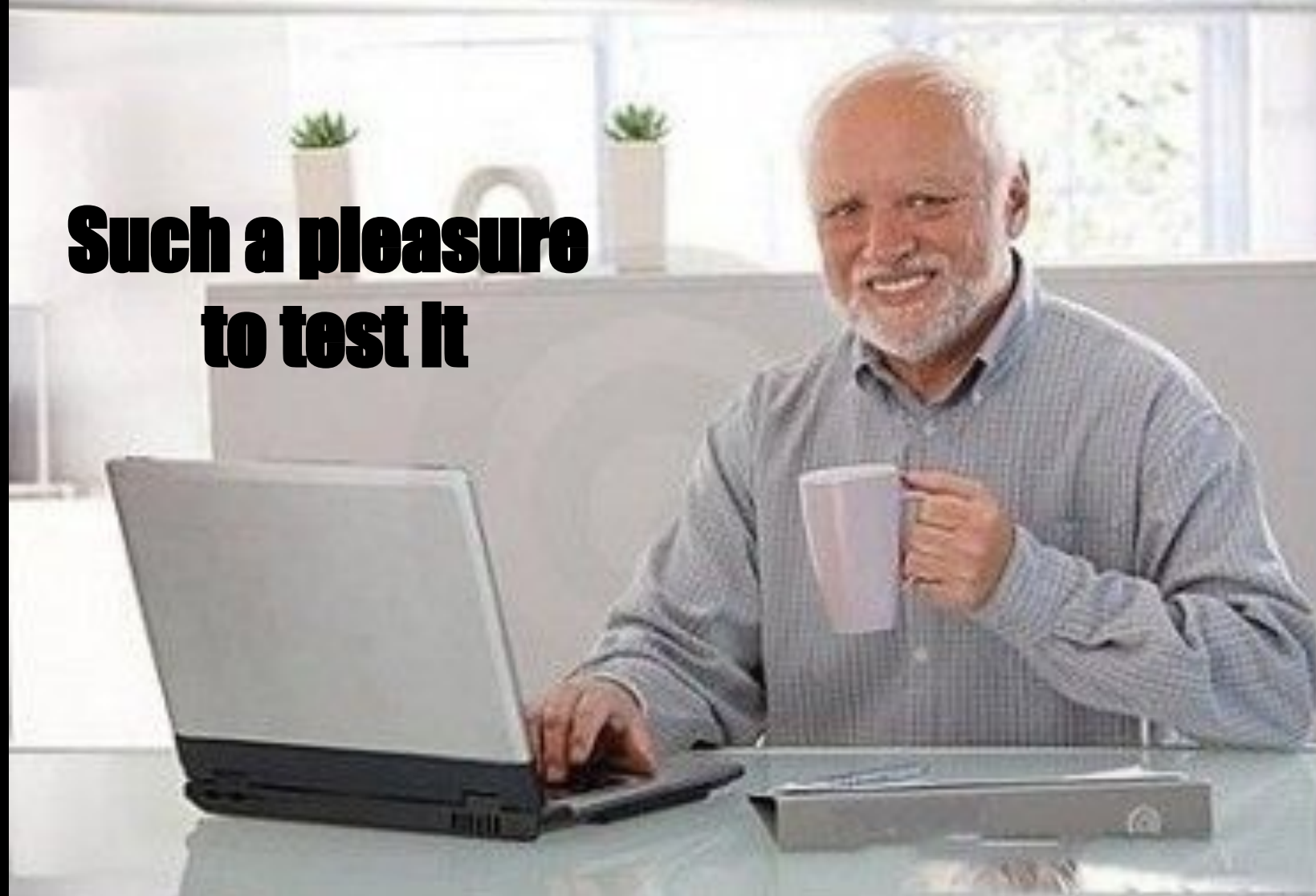
Autotests — PICT

Mode	Optimization Mode	Cache Mode	GC Type	GC Scheduler	Allocator	Thread State Checker	Target	Test Set
TwoStage	Debug	StaticOnlyDist	PMCS	Adaptive	Custom	Disabled	LINUX_X64	onlyK2
TwoStage	Debug	StaticPerFile	CMS	Adaptive	Custom	Disabled	LINUX_X64	onlyK2
TwoStage	Debug	StaticEverywhere	CMS	Adaptive	Custom	Disabled	IOS_SIMULATOR_ARM64	onlyK2
TwoStage	Optimized	No	PMCS	Adaptive	Custom	Disabled	IOS_SIMULATOR_ARM64	onlyK2
TwoStage	Debug	StaticEverywhere	PMCS	Adaptive	Custom	Disabled	LINUX_X64	onlyK2
TwoStage	Debug	StaticPerFile	PMCS	Adaptive	Custom	Disabled	IOS_SIMULATOR_ARM64	onlyK2
TwoStage	Debug	No	CMS	Adaptive	Custom	Disabled	MINGW_X64	onlyK2
TwoStage	Debug	StaticOnlyDist	CMS	Adaptive	Custom	Disabled	IOS_SIMULATOR_ARM64	onlyK2
TwoStage	Optimized	No	CMS	Adaptive	Custom	Disabled	LINUX_X64	onlyK2
TwoStage	Debug	No	PMCS	Adaptive	Custom	Disabled	MINGW_X64	onlyK2

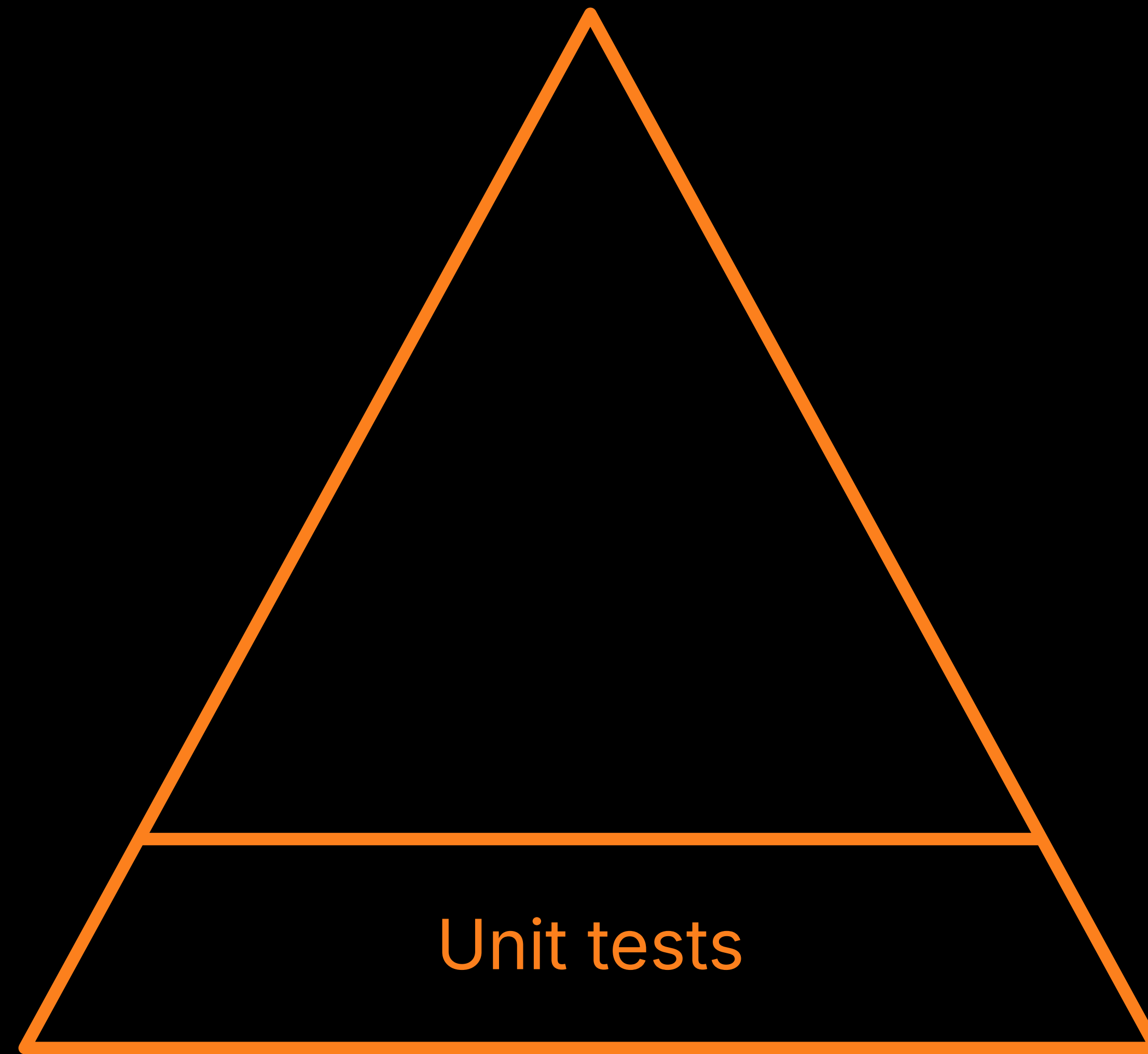
**Kotlin is a great
language**

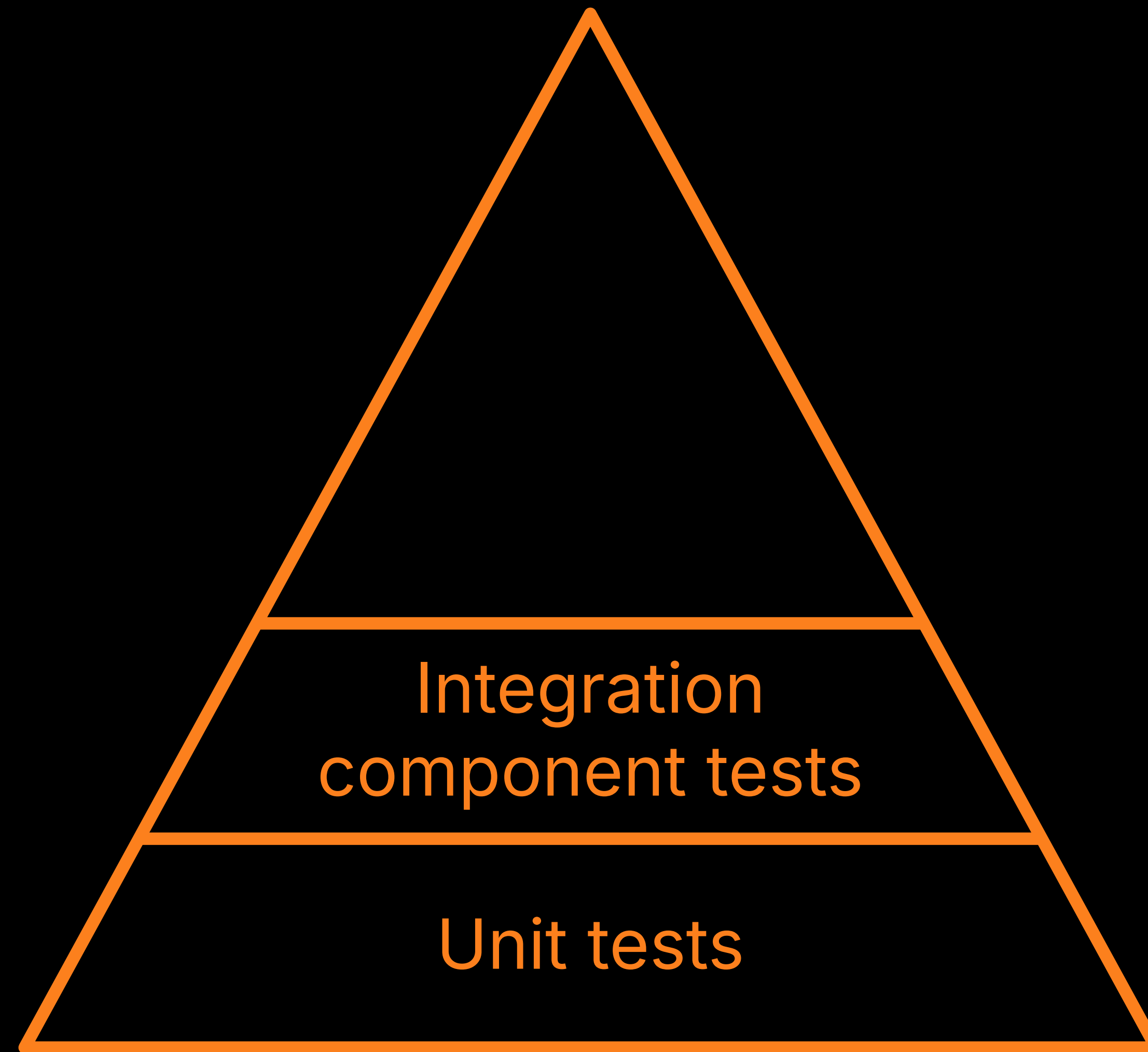


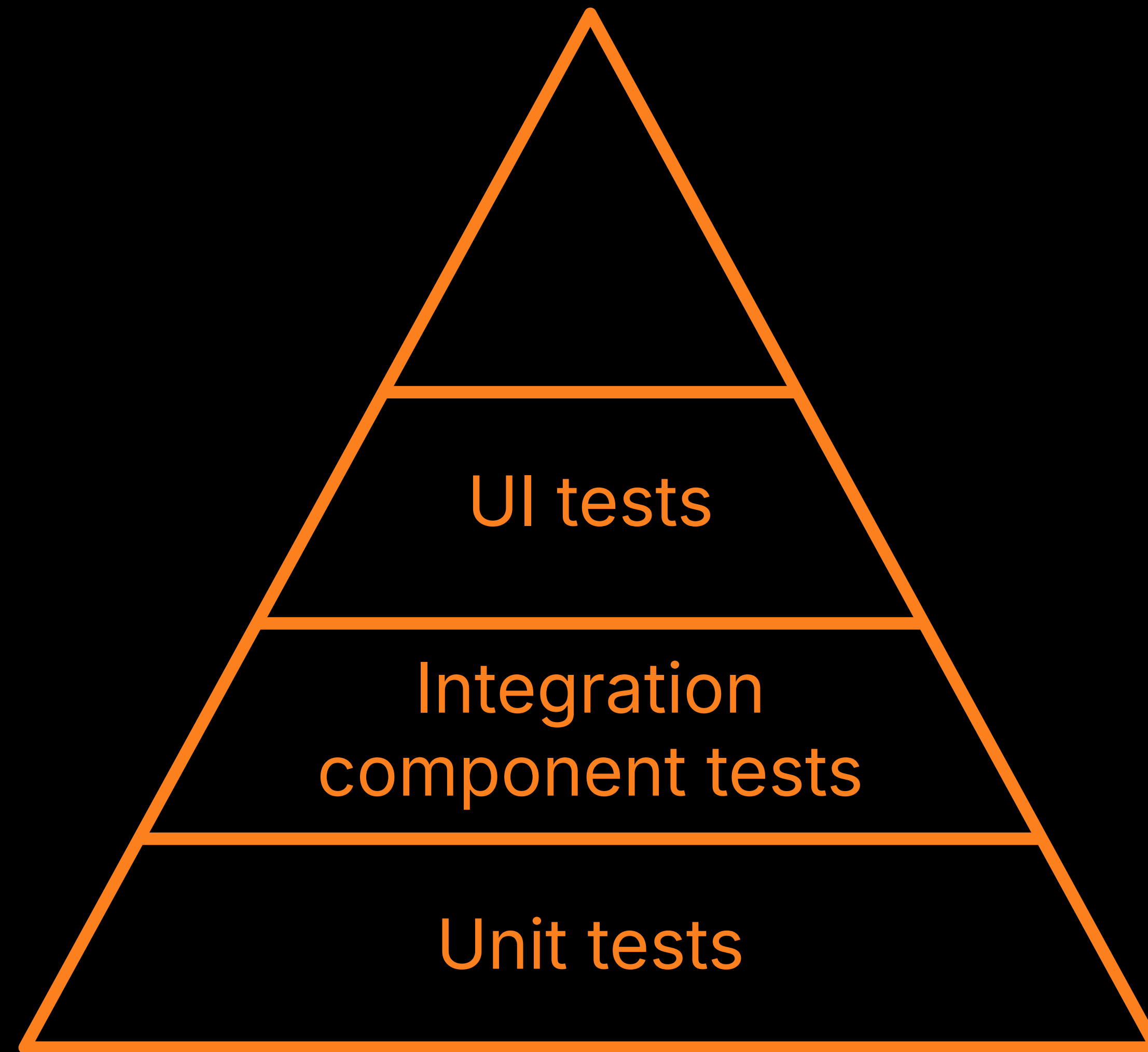
**Such a pleasure
to test it**

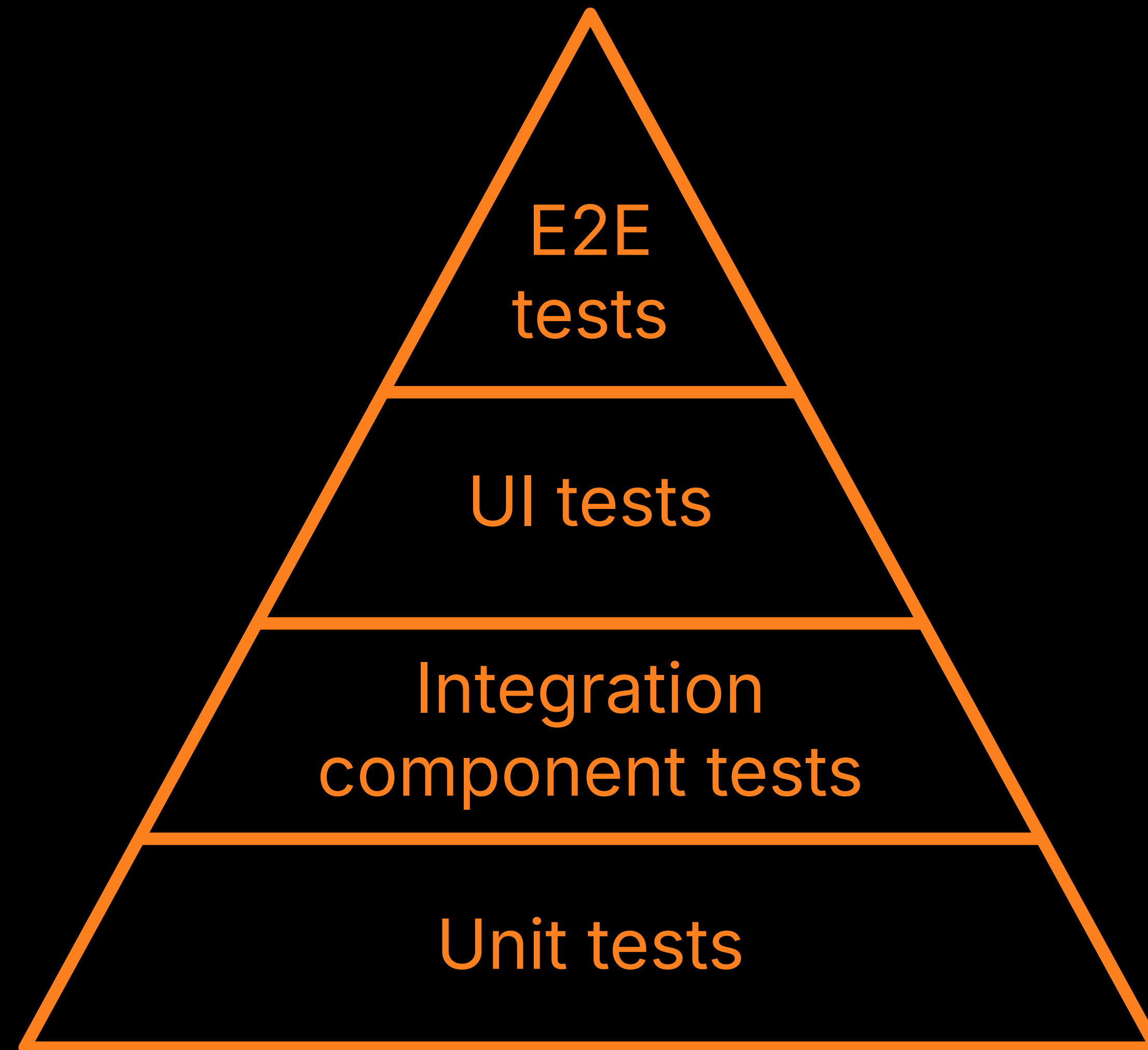


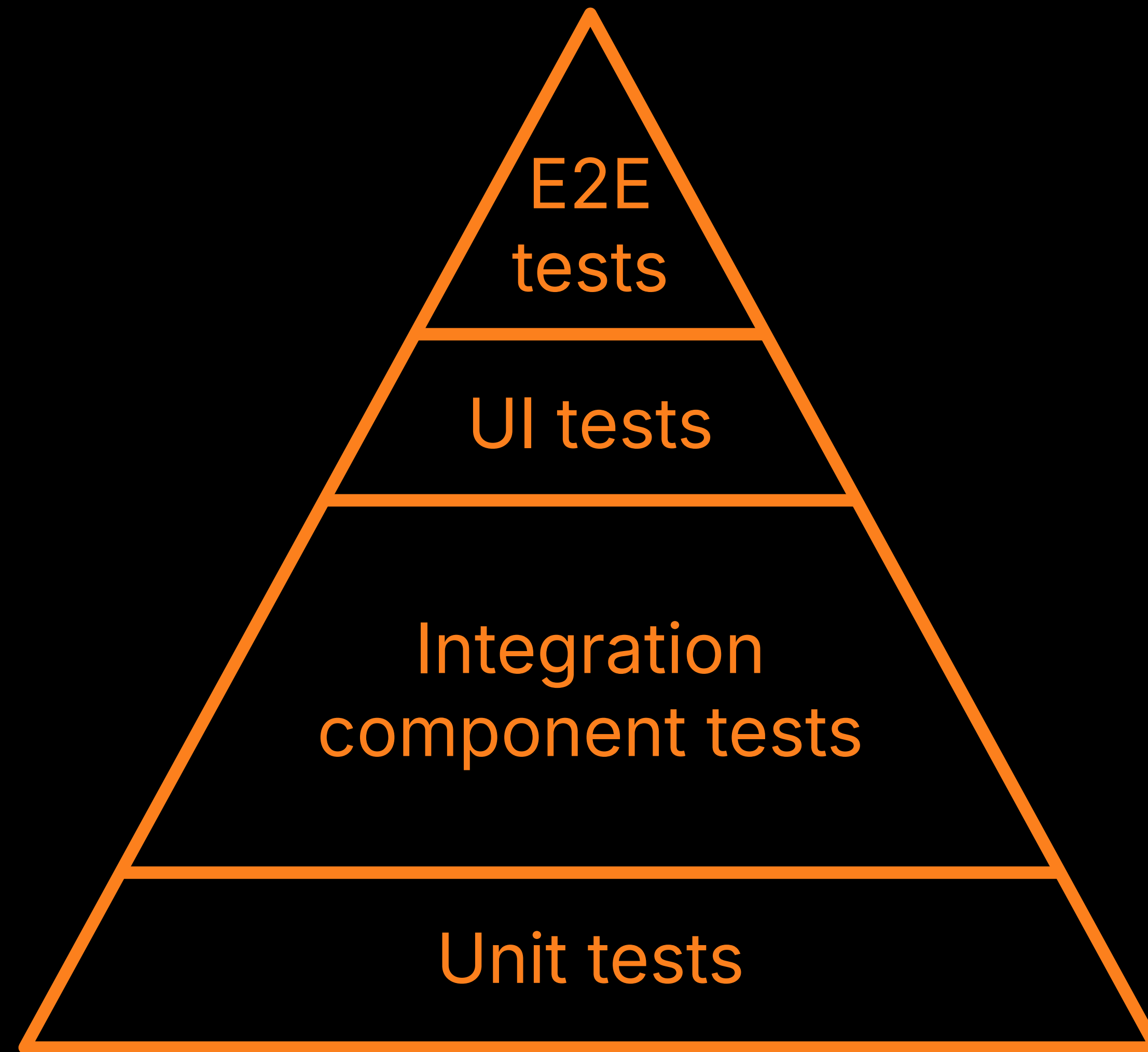
Levels of testing

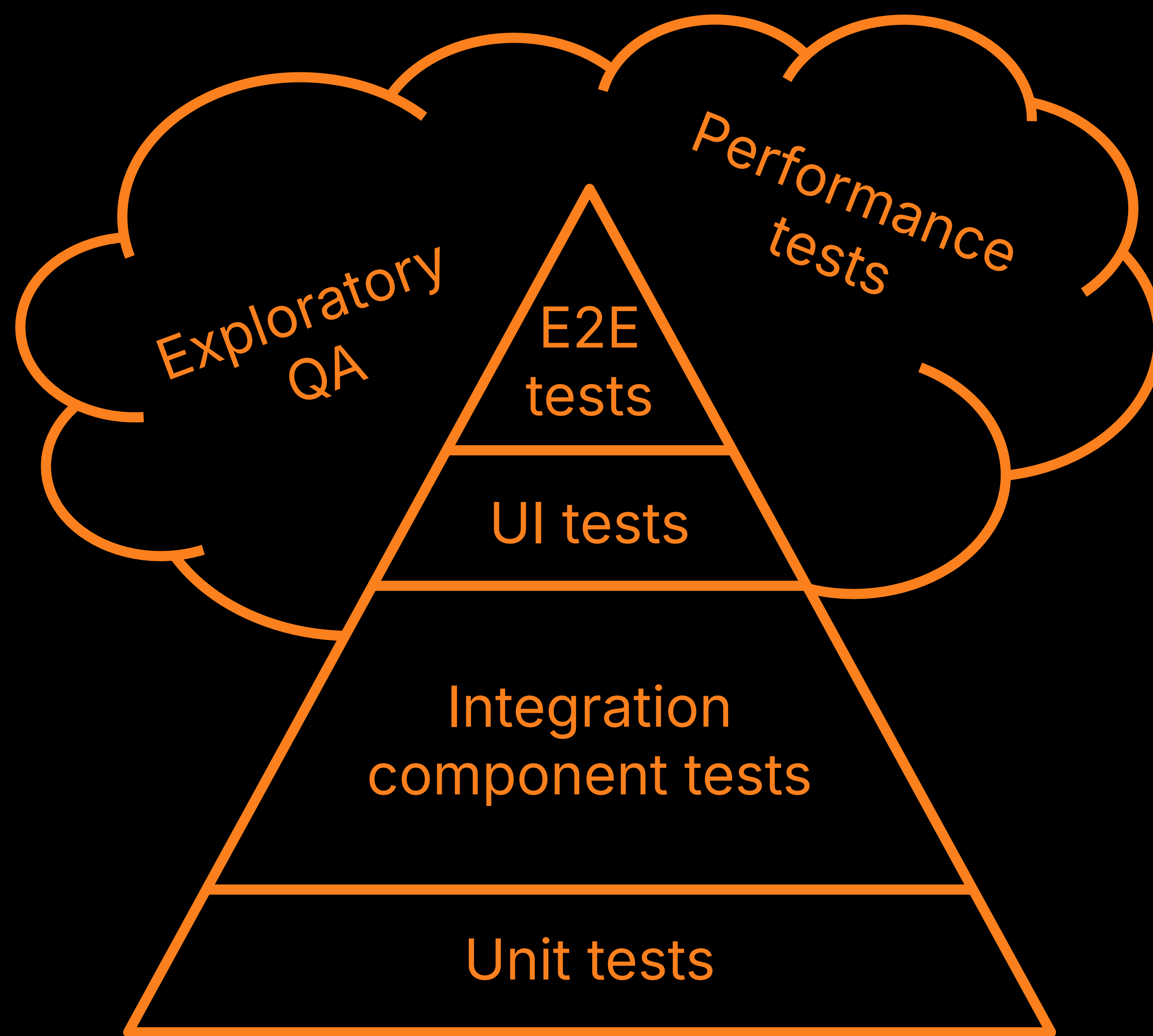












**It's all fine. But what did we
miss?**

How we measure quality

How we measure quality



Vsevolod Tolstopyatov
Kotlin Project Lead

“Hey Artur, do we have a reliable way to measure quality?”

How we measure quality



Vsevolod Tolstoplyatov
Kotlin Project Lead

“Hey Artur, do we have a reliable way to measure quality?”



How we measure quality

Or the excuses not to

- Coverage model is **very complicated** and **expensive**
 - Instead – the **quality criterias** in each QA team

How we measure quality

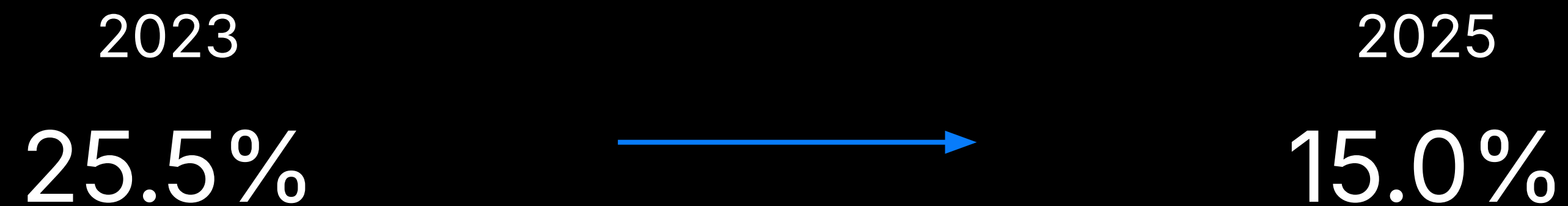
Or the excuses not to

- Coverage model is **very complicated** and **expensive**
 - Instead – the **quality criterias** in each QA team
- ⇒
- **Feedback !**
 - Monitor **regressions** reported by **users** vs **devs + QAs**
 - Closely look at the **JetBrains internal developers issues**
 - Analyze the internal and external **survey data**

How we measure quality

Or the excuses not to

According to Kotlin Developer Survey, the dissatisfaction with Quality and Stability improved a lot



How we measure quality

Or the excuses not to

We are lucky to work without coming up with
pseudo-metrics of quality :)

How we measure quality

Or the excuses not to

We are lucky to work without coming up with
pseudo-metrics of quality :)

CSAT for Kotlin ~90% 🎉

CSAT for Kotlin support in IDE ~85% 🎉

How we measure quality

Or the excuses not to

We are lucky to work without coming up with
pseudo-metrics of quality :)

CSAT for Kotlin ~90% 🎉*

CSAT for Kotlin support in IDE ~85% 🎉*

* but of course there is a room for improvements

How we measure quality

Releases

We Are Looking For EAP Champions!



Ekaterina Petrova

November 23, 2022

The Kotlin Early Access Preview (EAP) is a program where the Kotlin team ships a few Beta and Release Candidate builds before every release. Any Kotlin developer can participate in the Early Access Preview, try out the latest Kotlin features before they are released, and help the Kotlin team gather the feedback necessary to stabilize a new language version.



Square

How to install Kotlin 2.2.20

If you run into any problems:

Special thanks to our EAP champions

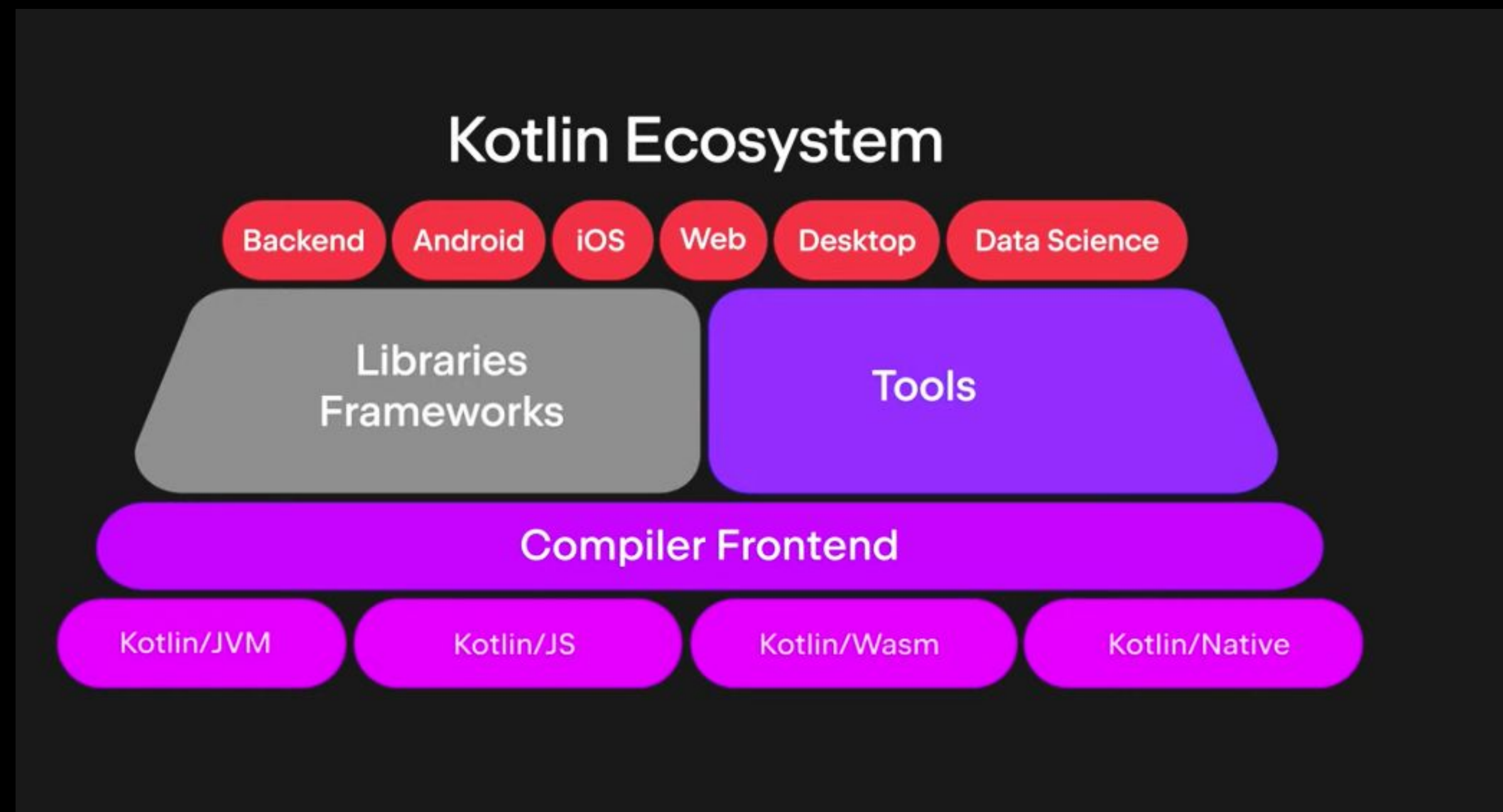


Further reading

Conclusions

Takeaways

- Programming language is an **entire ecosystem**



Takeaways

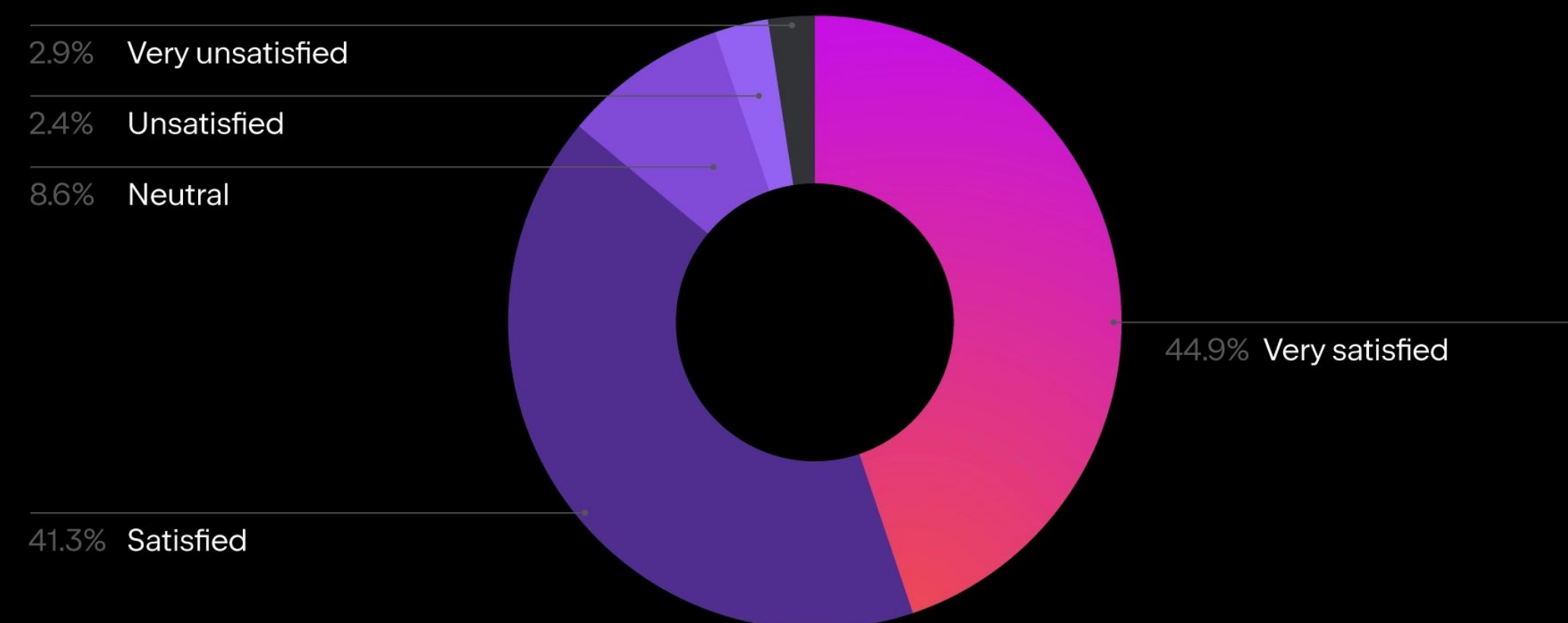
- Programming language is an **entire ecosystem**
- In such a complex system **imitation of a user** from different angles is a must



Takeaways

- Programming language is an **entire ecosystem**
- In such a complex system **imitation of a user** from different angles is a must
- We don't need to calculate coverage to **satisfy our users**

How users rate their experience with Kotlin in the last six months



Takeaways

- Programming language is an **entire ecosystem**
- In such a complex system **imitation of a user** from different angles is a must
- We don't need to calculate coverage to **satisfy our users**
- Even though it is tooling for developers, it is **still a product** that depends on the feedback

2 million
developers
regularly
write Kotlin
code





Artur



Alexander

Do connect with us on LinkedIn!

Thank you!



We are hiring!

Thank you!