PAULIEN VAN ALST

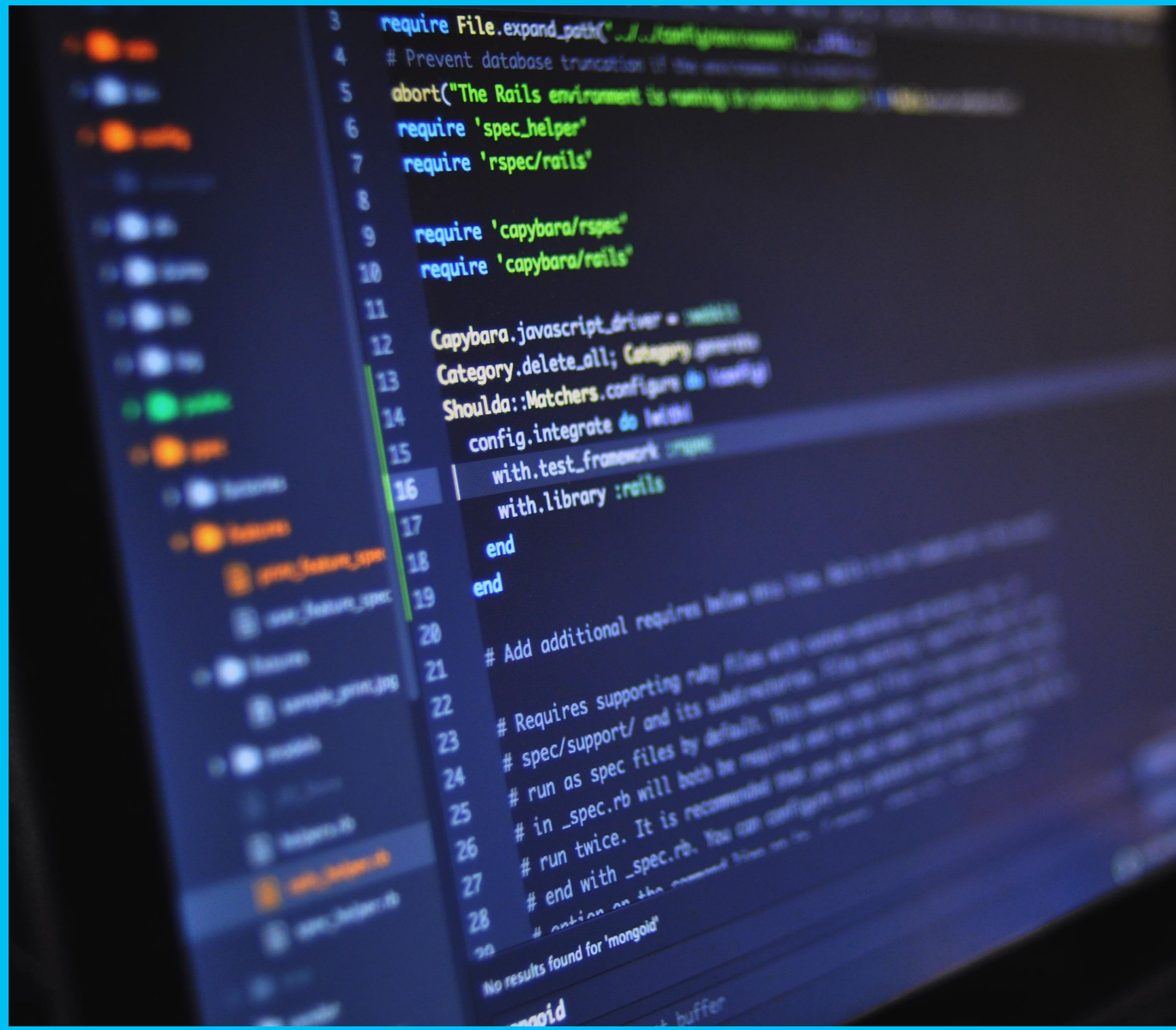
KOTLIN COROUTINES

Workshop

KOTLIN COROUTINES

WELCOME

- Who am I?
- Who are you?
- What do you expect to learn?
- Introduction
- Set-up upcoming afternoon
- Hands-on workshop



WHO AM I ?

- Paulien van Alst (@PaulienVanAlst)
- Software engineer @OpenValue
- Co-Host ~~@BarCoding~~ podcast **@coding_bar**
- Google Developer Expert Kotlin

WHO ARE YOU?

KNOWLEDGE OF KOTLIN?

WHAT DO YOU EXPECT TO LEARN?

PROGRAM OF THIS AFTERNOON

12:30 - 13:00	Small introduction and set-up
13:00 - 13:30	Introduction to coroutines (part 1)
13:30 - 13:45	Answers part 1 && Quiz time!
13:45 - 14:15	Introduction to coroutines (part 2)
14:15 - 14:30	Answers part 2 && Quiz time!

14:30 - 14:45	Break
14:45 - 15:15	Flows (part 1 and part 2)
15:30 - 15:45	Answers Flows && Quiz time!
15:45 - 16:15	Ktor (or Spring) and Coroutiunes
16:15 - 16:30	Wrap up!

COROUTINES - FIRST THINGS FIRST

```
fun start() {
    printRed("Starting up console!")
    Thread.sleep( millis: 100)
    printRed("Console started")
}
```

```
fun play() {
    printBlue("Playing a game")
     Thread.sleep( millis: 200)
    printBlue("Finished playing a game")
}
```

```
fun havingASnack() {
    printGreen("Having a snack")
    Thread.sleep( millis: 50)
    printGreen("Finished snack")
}
```

```
fun main(args: Array<String>) {
    start()
    play()
    havingASnack()
    println("Turning off console!")
}
```

```
Starting up console!
Console started
Playing a game
Finished playing a game
Having a snack
Finished snack
Turning off console!
```

COROUTINES - COROUTINE EDITION

```
suspend fun start() {
    printRed("Starting up console!")
    delay( timeMillis: 100)
    printRed("Console started")
}

suspend fun play() {
    printBlue("Playing a game")
    delay( timeMillis: 200)
    printBlue("Finished playing a game")
}

suspend fun havingASnack() {
    printGreen("Having a snack")
    delay( timeMillis: 50)
    printGreen("Finished snack")
}

suspend fun main(args: Array<String>) {

    coroutineScope { this: CoroutineScope
        launch { start() }
        launch { play() }
        launch { havingASnack() } ^coroutineScope
    }
    println("Turning off console!")
}
```

```
Starting up console!
Playing a game
Having a snack
Finished snack
Console started
Finished playing a game
Turning off console!
```

AS A PROGRAMMER YOU ARE IN CONTROL!

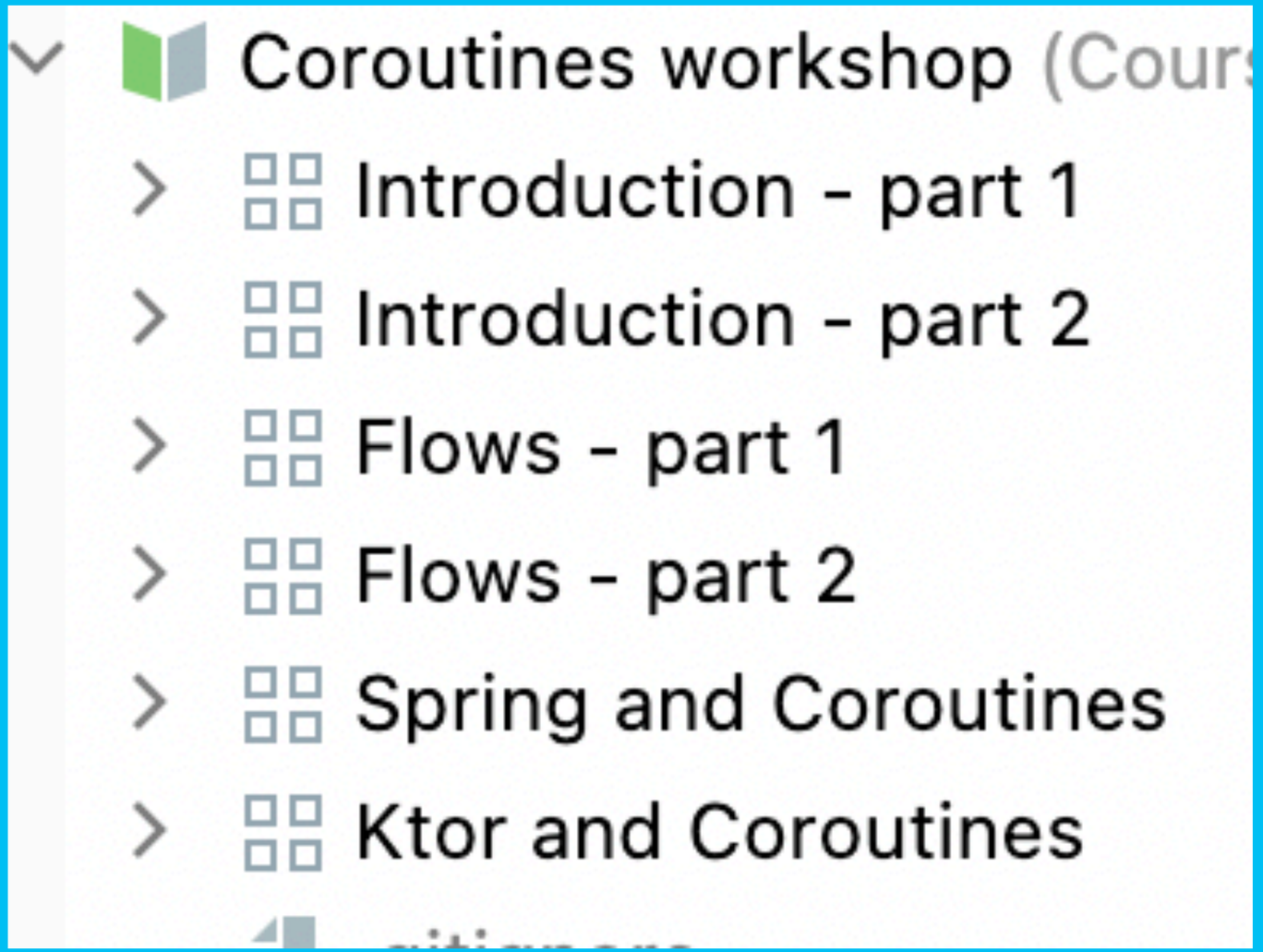
- Coroutine has several states
- Coroutine is Started
- Coroutine is Cancelled
- Coroutine is Completed
- And everything in between

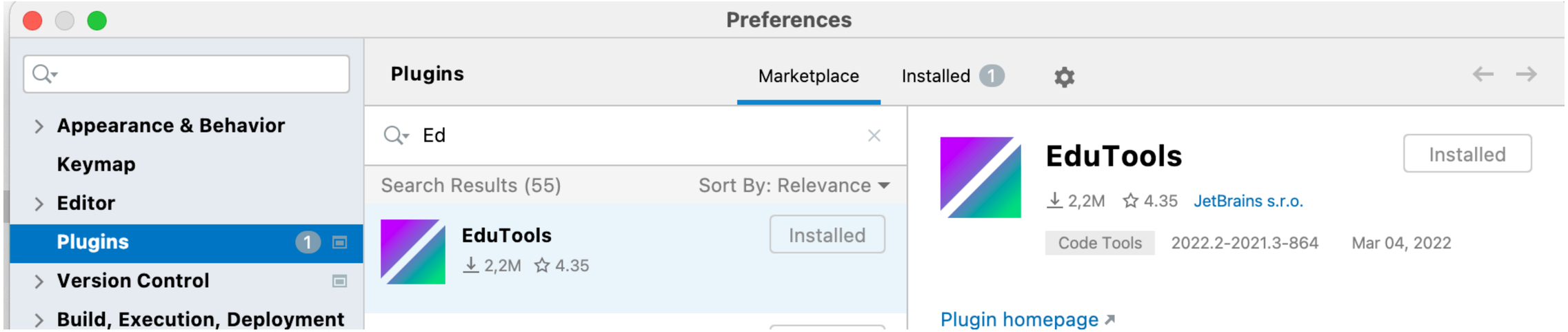
- You act on those or trigger them

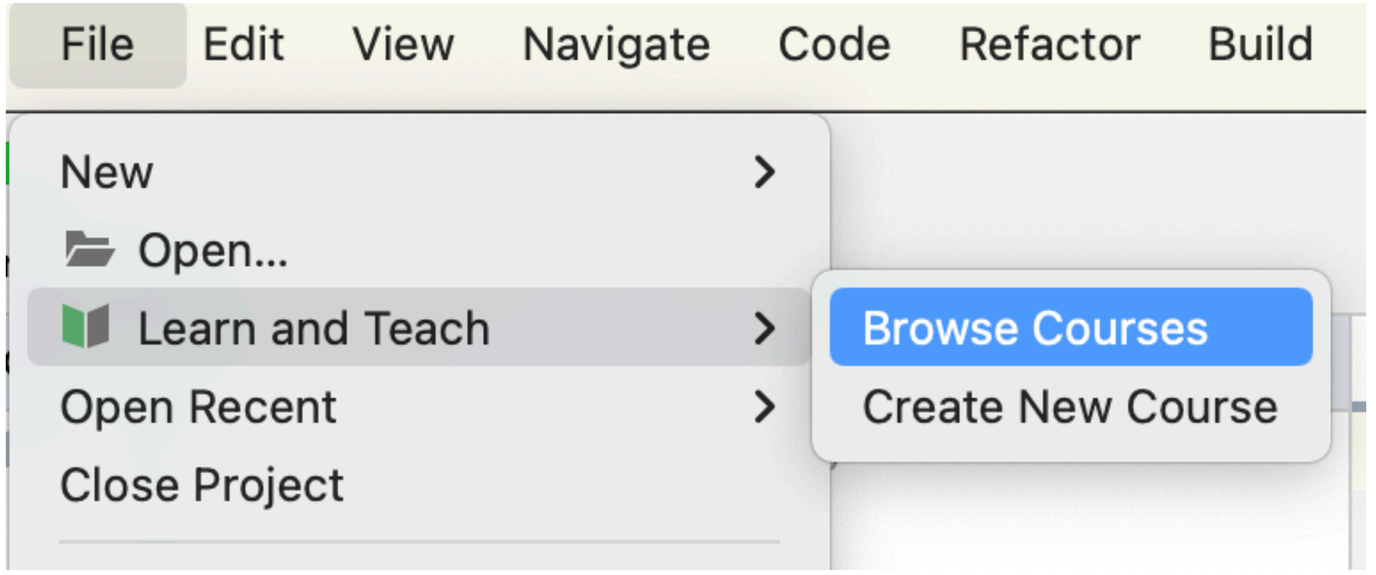
LET’S GET STARTED!

WORKSHOP SET-UP

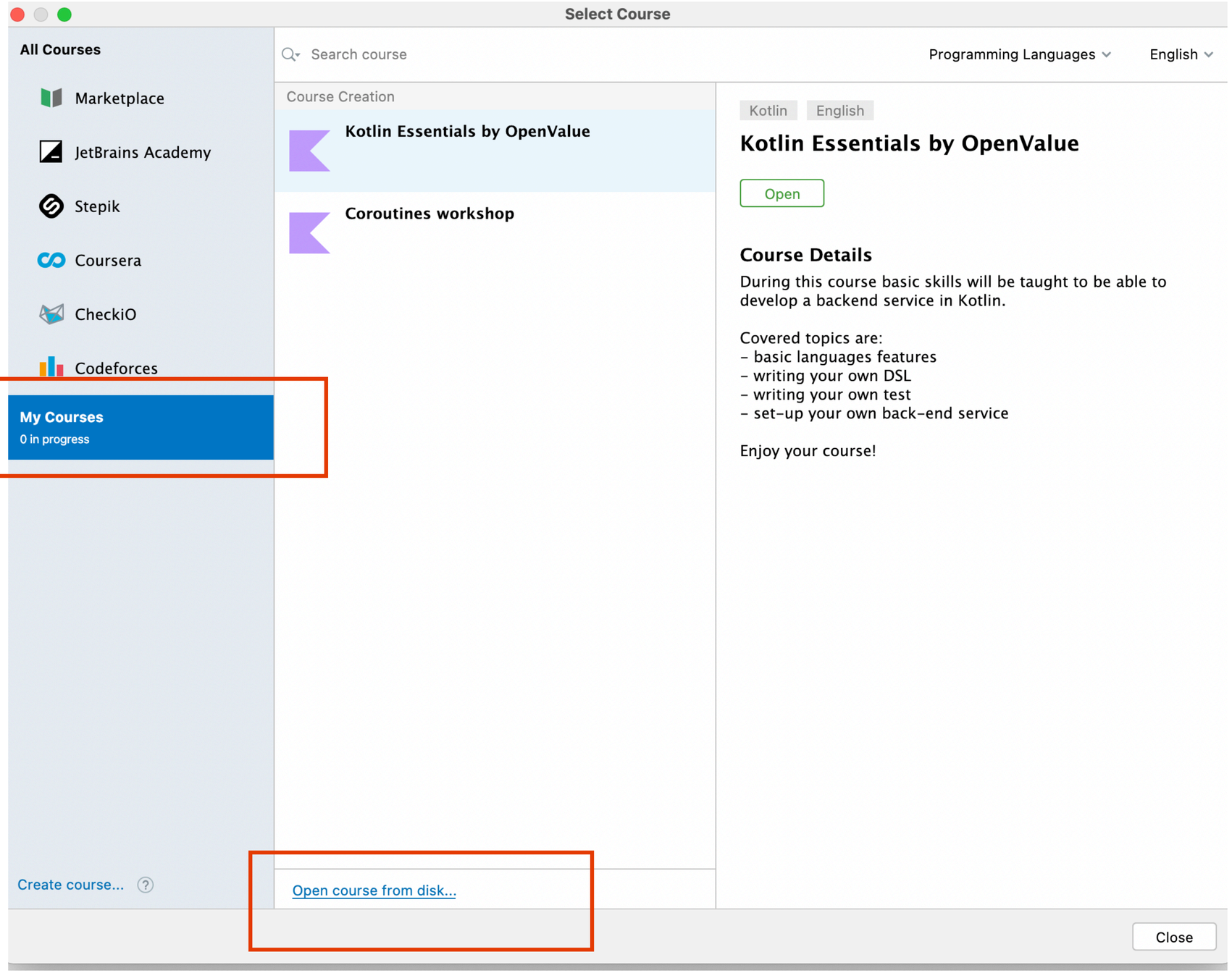
- EduTools plugin from IntelliJ
- Validation of your answers
- Keep track of progress
- Exercise independent of each other




Preferences
Appearance & Behavior
Keymap
Editor
Plugins 1
Version Control
Build, Execution, Deployment
Plugins
Marketplace
Installed 1
Ed
Search Results (55)
Sort By: Relevance
EduTools
2,2M 4.35
Installed
EduTools
Installed
2,2M 4.35 JetBrains s.r.o.
Code Tools
2022.2-2021.3-864
Mar 04, 2022
Plugin homepage


File
Edit
View
Navigate
Code
Refactor
Build
New
Open...
Learn and Teach
Open Recent
Close Project
Browse Courses
Create New Course

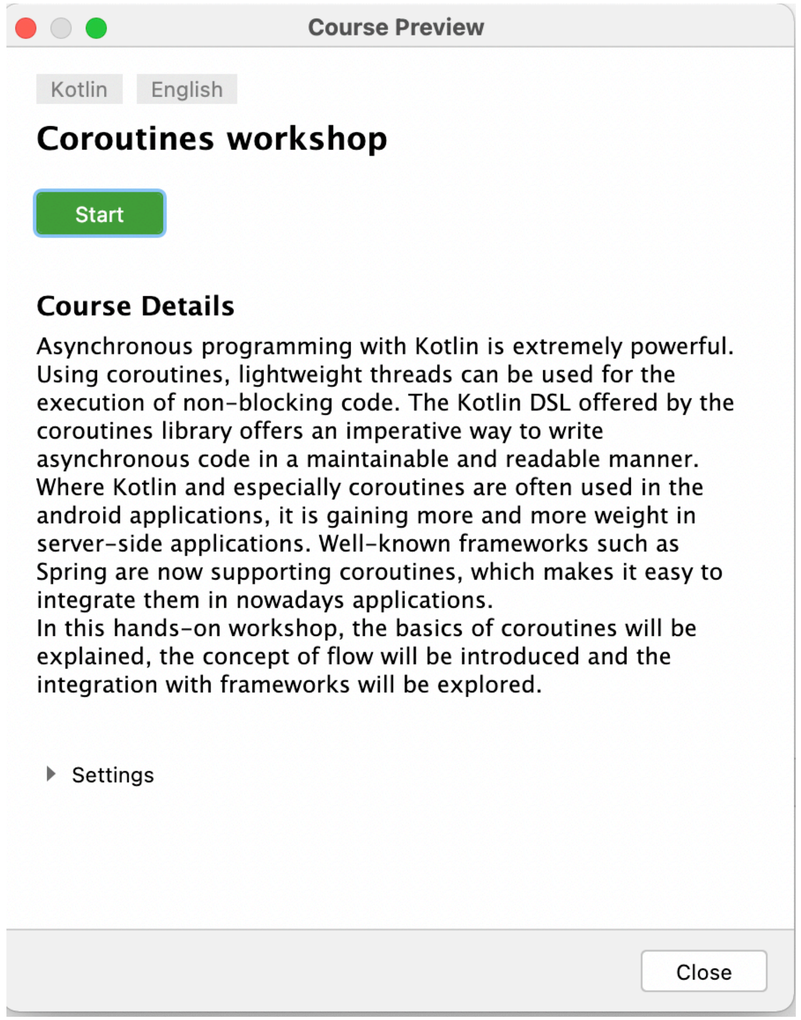
GITHUB.COM:PAULIENVA/WORKSHOPS

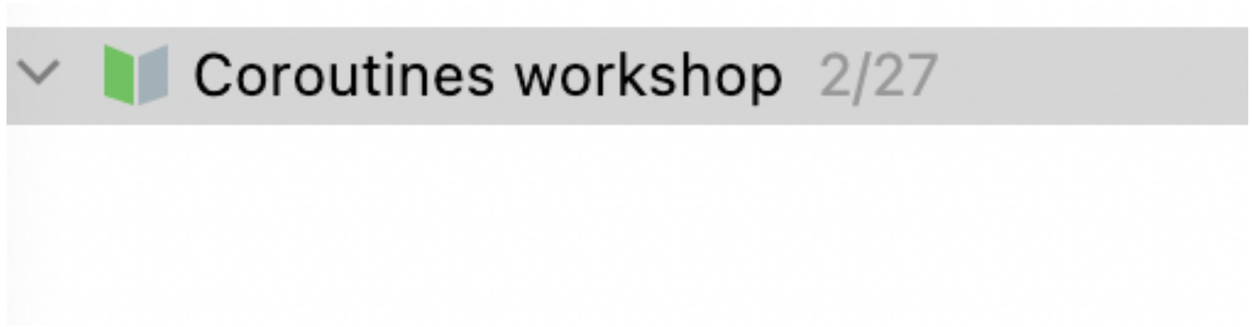

Select Course
All Courses
Marketplace
JetBrains Academy
Stepik
Coursera
CheckiO
Codeforces
My Courses
0 in progress
Create course...
Search course
Programming Languages
English
Course Creation
Kotlin Essentials by OpenValue
Coroutines workshop
Open course from disk...
Kotlin
English
Kotlin Essentials by OpenValue
Open
Course Details
During this course basic skills will be taught to be able to develop a backend service in Kotlin.
Covered topics are:
- basic languages features
- writing your own DSL
- writing your own test
- set-up your own back-end service
Enjoy your course!
Close

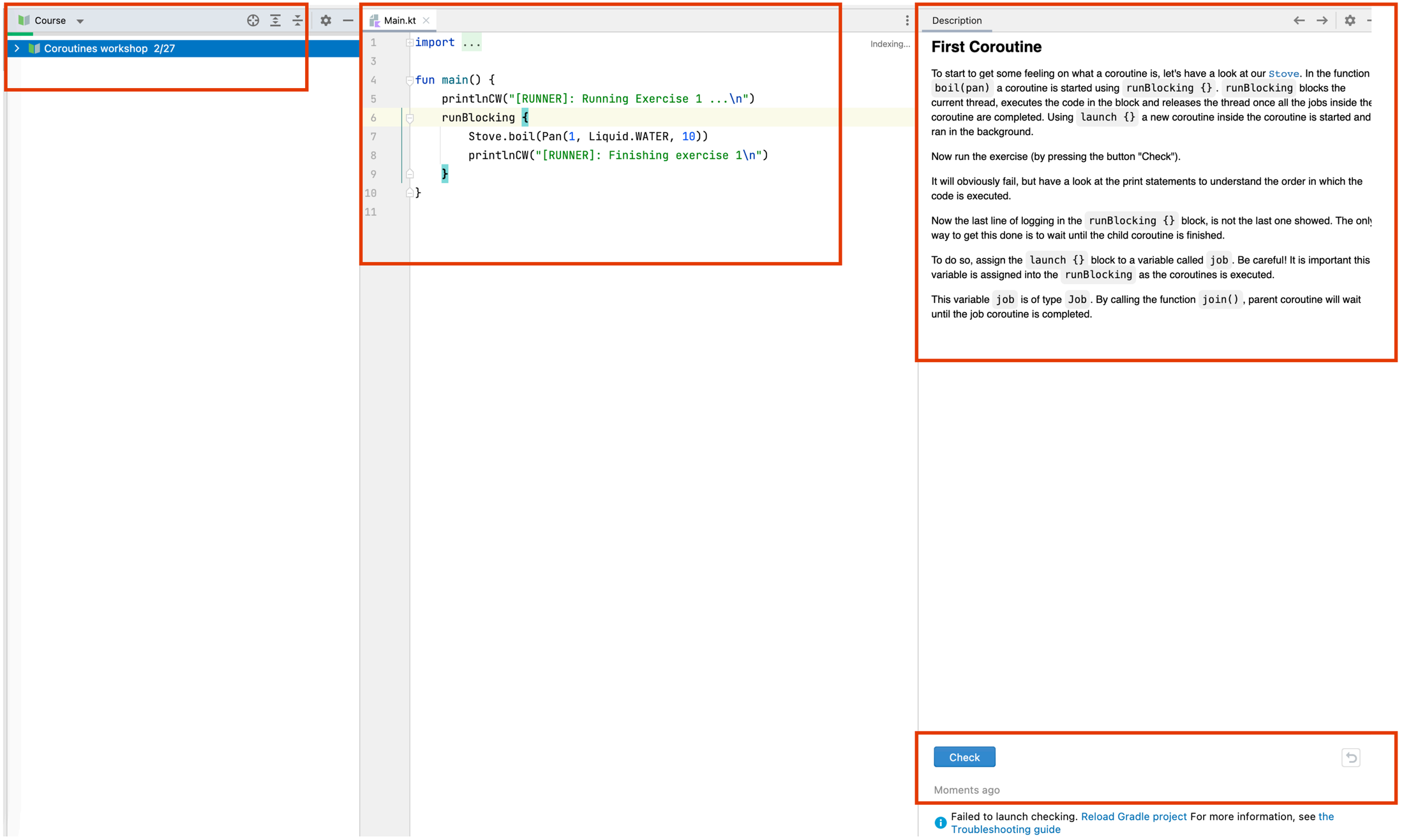
DO NOT UNZIP THE ZIP!

-

UPLOAD IT AS IS


Course Preview
Kotlin
English
Coroutines workshop
Start
Course Details
Asynchronous programming with Kotlin is extremely powerful. Using coroutines, lightweight threads can be used for the execution of non-blocking code. The Kotlin DSL offered by the coroutines library offers an imperative way to write asynchronous code in a maintainable and readable manner. Where Kotlin and especially coroutines are often used in the android applications, it is gaining more and more weight in server-side applications. Well-known frameworks such as Spring are now supporting coroutines, which makes it easy to integrate them in nowadays applications.
In this hands-on workshop, the basics of coroutines will be explained, the concept of flow will be introduced and the integration with frameworks will be explored.
Settings
Close


Coroutines workshop 2/27


Course
Coroutines workshop 2/27
Main.kt
Indexing...
1 import ...
3
4 fun main() {
5 printlnCW("[RUNNER]: Running Exercise 1 ...\n")
6 runBlocking {
7 Stove.boil(Pan(1, Liquid.WATER, 10))
8 printlnCW("[RUNNER]: Finishing exercise 1\n")
9 }
10 }
11
Description
First Coroutine
To start to get some feeling on what a coroutine is, let's have a look at our Stove. In the function boil(pan) a coroutine is started using runBlocking {}. runBlocking blocks the current thread, executes the code in the block and releases the thread once all the jobs inside the coroutine are completed. Using launch {} a new coroutine inside the coroutine is started and ran in the background.
Now run the exercise (by pressing the button "Check").
It will obviously fail, but have a look at the print statements to understand the order in which the code is executed.
Now the last line of logging in the runBlocking {} block, is not the last one showed. The only way to get this done is to wait until the child coroutine is finished.
To do so, assign the launch {} block to a variable called job. Be careful! It is important this variable is assigned into the runBlocking as the coroutines is executed.
This variable job is of type Job. By calling the function join(), parent coroutine will wait until the job coroutine is completed.
Check
Moments ago
Failed to launch checking. Reload Gradle project For more information, see the Troubleshooting guide

REQUIREMENTS

- JDK < 17
- Git
- IntelliJ with the EduTools Plugin