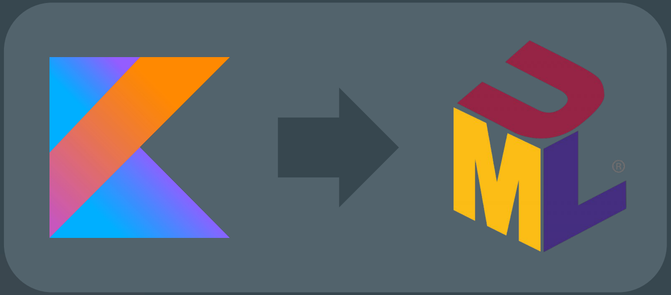
Quick Sequence Diagrams

From Kotlin Code to UML Sequence Diagrams
(With Very Little Effort)

Quick Seq



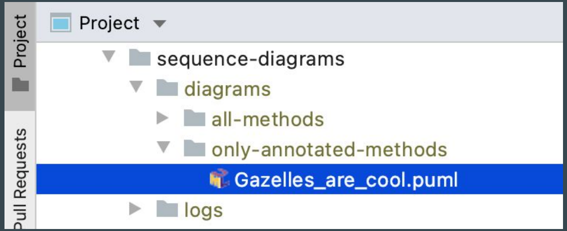
What Does the User See?

```
@SequenceDiagram
fun someMethod() {...}

@Test
@SequenceDiagramEntryPoint(“Gazelles are cool”)
fun someTest() {...}
```

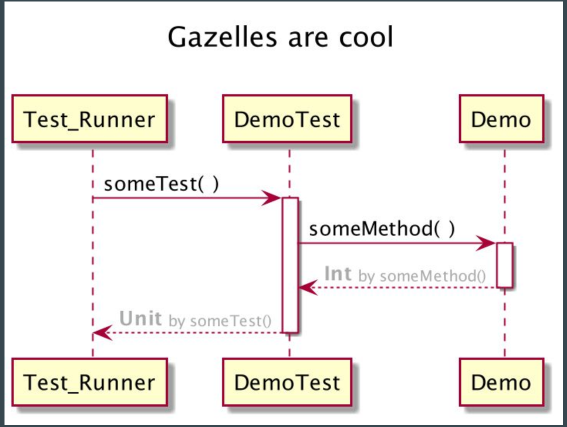
```
apply(plugin = “com.kiwi.quickseq”)
```

Run Test(s)



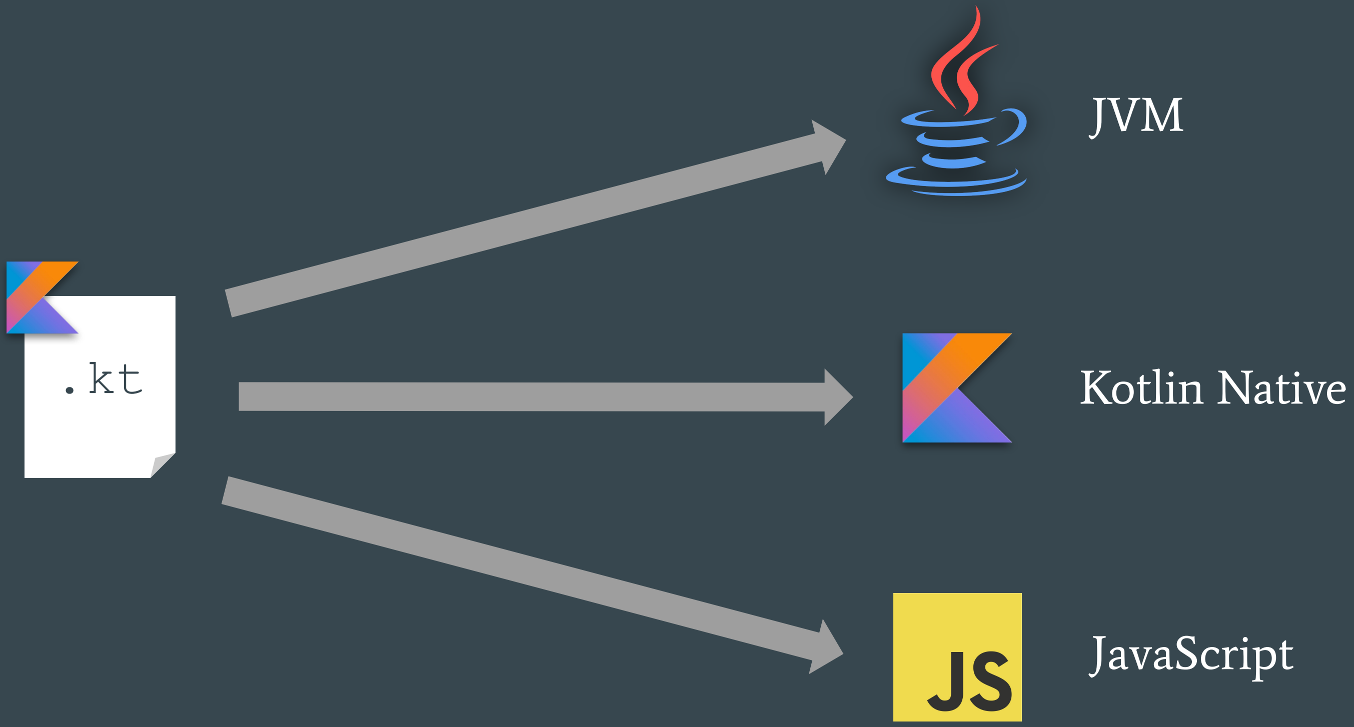
Project
Pull Requests
Project

- sequence-diagrams
  - diagrams
    - all-methods
    - only-annotated-methods
      - Gazelles_are_cool.puml
  - logs

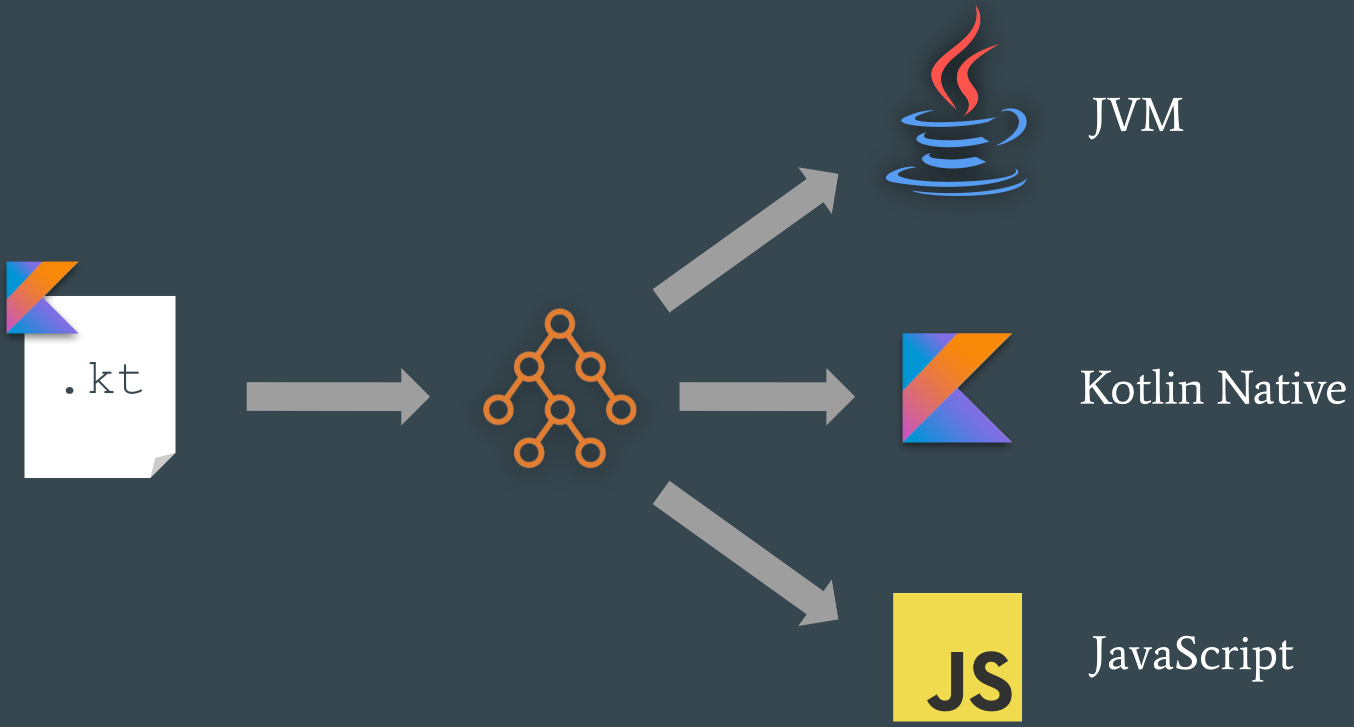


What Happens Behind the Scenes?

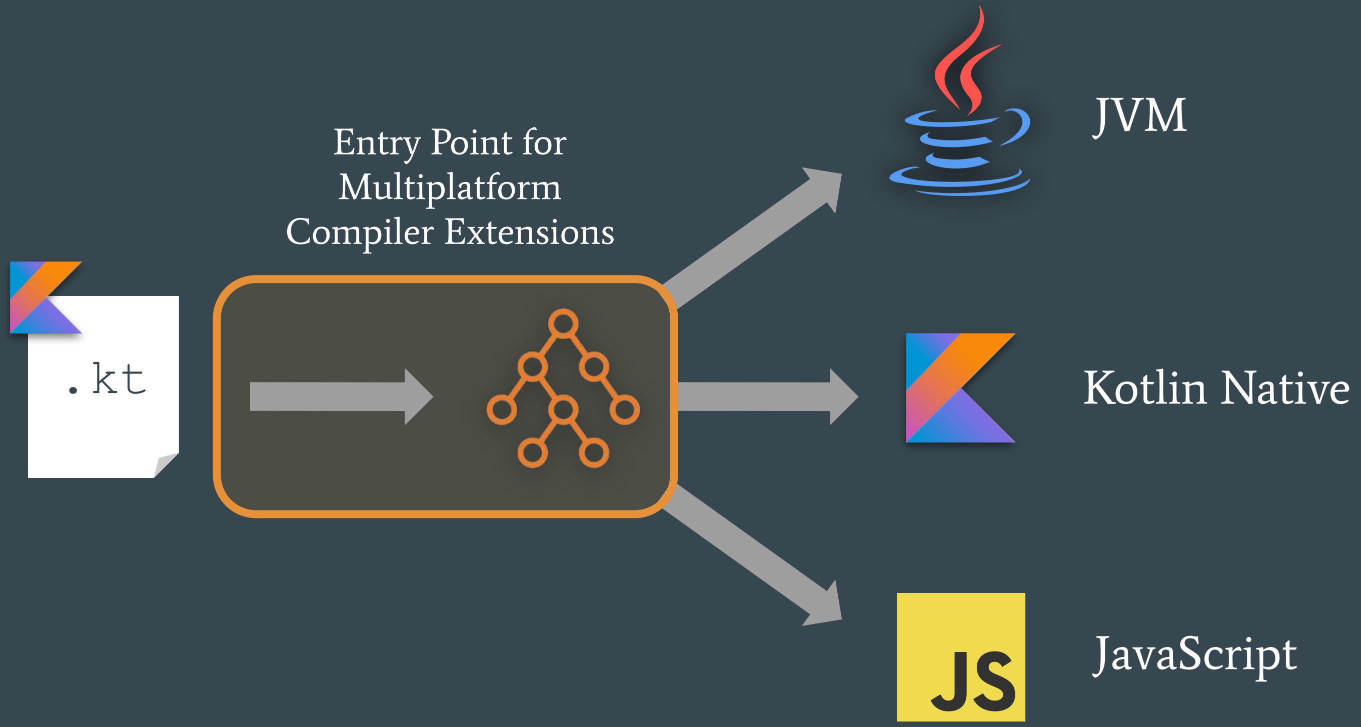
The Kotlin Compiler – once upon a time



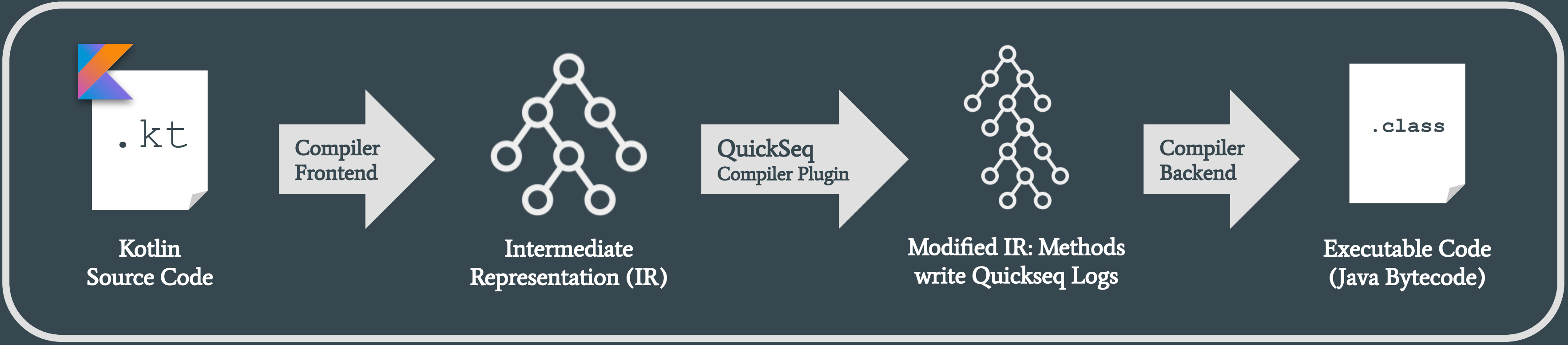
The Kotlin Compiler – since 1.4 / 1.5



The Kotlin Compiler – since 1.4 / 1.5



COMPILE TIME



.kt

Kotlin Source Code

Compiler Frontend

Intermediate Representation (IR)

QuickSeq Compiler Plugin

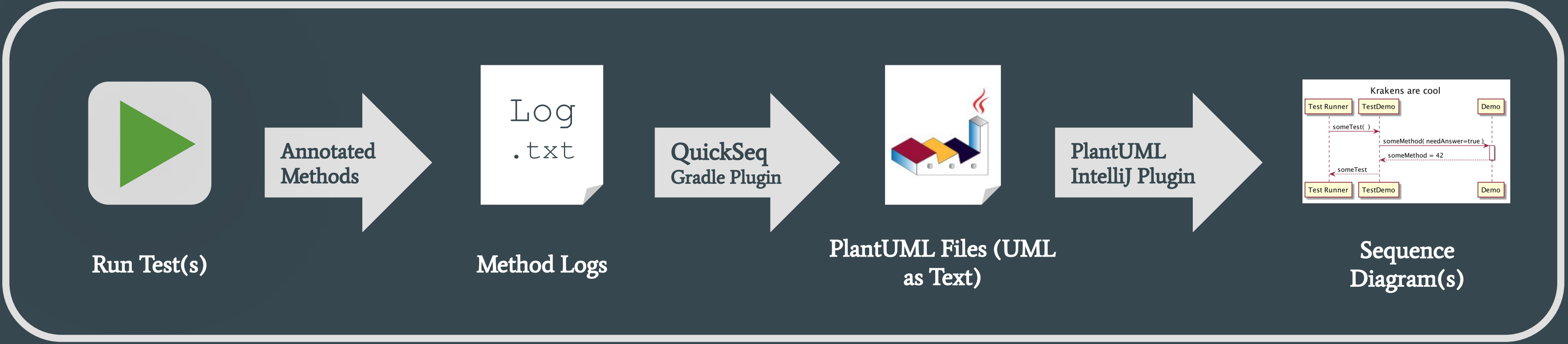
Modified IR: Methods write Quickseq Logs

Compiler Backend

.class

Executable Code (Java Bytecode)

TEST RUNTIME



Run Test(s)

Annotated Methods

Log .txt

Method Logs

QuickSeq Gradle Plugin

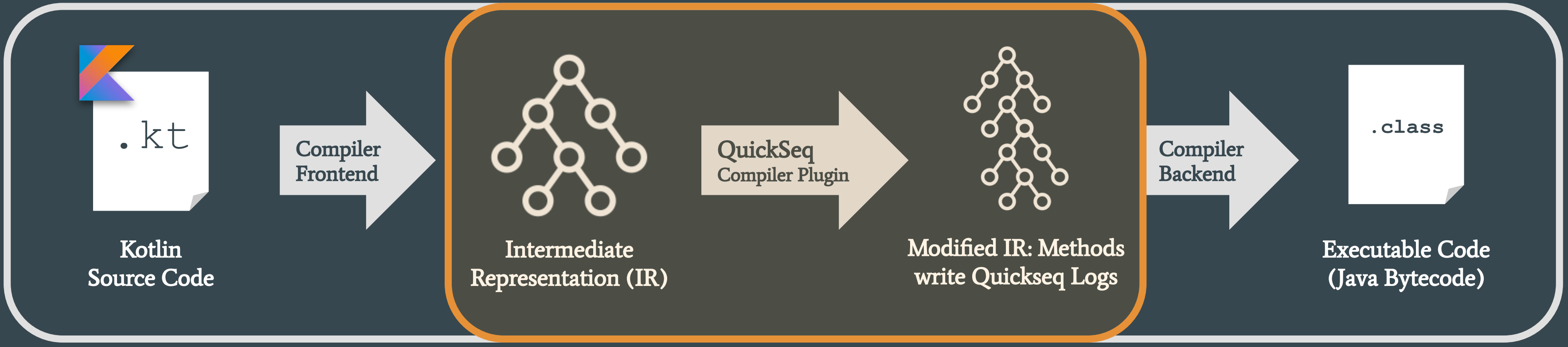
PlantUML Files (UML as Text)

PlantUML IntelliJ Plugin

Sequence Diagram(s)

COMPILE TIME

COMPILER PLUGIN



.kt

Kotlin Source Code

Compiler Frontend

Intermediate Representation (IR)

QuickSeq Compiler Plugin

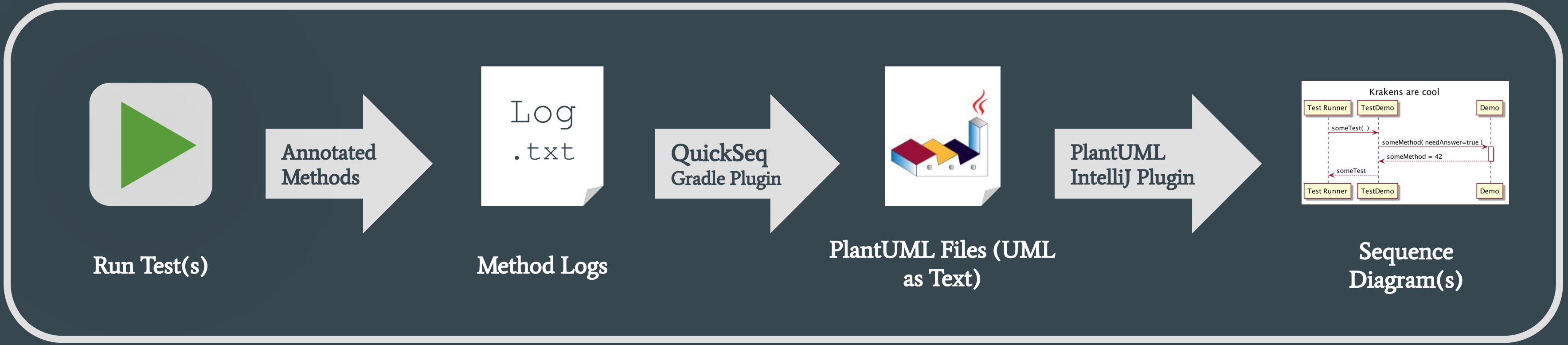
Modified IR: Methods write Quickseq Logs

Compiler Backend

.class

Executable Code (Java Bytecode)

TEST RUNTIME



Run Test(s)

Annotated Methods

Log .txt

Method Logs

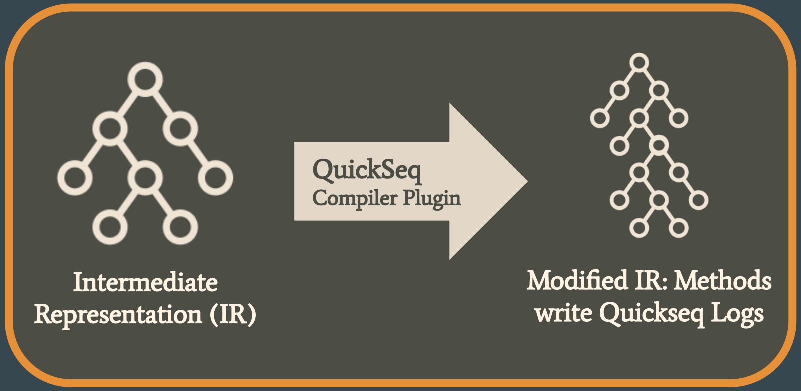
QuickSeq Gradle Plugin

PlantUML Files (UML as Text)

PlantUML IntelliJ Plugin

Sequence Diagram(s)

COMPILER PLUGIN

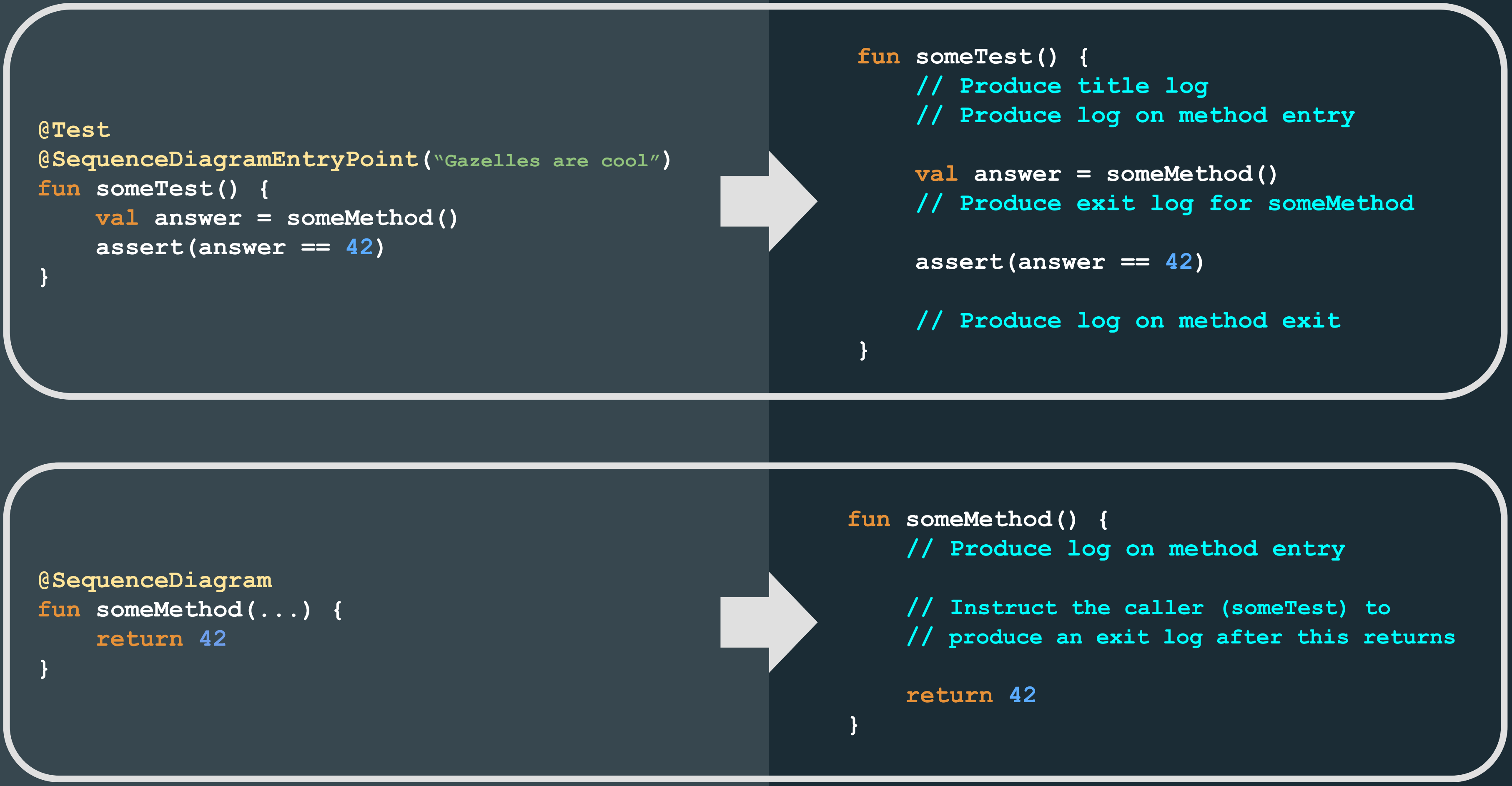


Intermediate Representation (IR)

QuickSeq
Compiler Plugin

Modified IR: Methods write Quickseq Logs

?



```
@Test
@SequenceDiagramEntryPoint(“Gazelles are cool”)
fun someTest() {
    val answer = someMethod()
    assert(answer == 42)
}
```

```
fun someTest() {
    // Produce title log
    // Produce log on method entry

    val answer = someMethod()
    // Produce exit log for someMethod

    assert(answer == 42)

    // Produce log on method exit
}
```

```
@SequenceDiagram
fun someMethod(...) {
    return 42
}
```

```
fun someMethod() {
    // Produce log on method entry

    // Instruct the caller (someTest) to
    // produce an exit log after this returns

    return 42
}
```

Demo

- Ideas to continue:
  - Support more Kotlin / JVM / Gradle versions
  - Support Kotlin Multiplatform
  - Integrate into CircleCI
  - Conversion of PlantUML files to PNG
- Want to use it?
  - Please leave us your email address
  - As soon as we know the final link to the QuickSeq repo, we will notify you

Questions?

References

- Kotlin IR compiler:
  - https://blog.jetbrains.com/kotlin/2021/02/the-jvm-backend-is-in-beta-let-s-make-it-stable-together/
  - https://bnorm.medium.com/exploring-kotlin-ir-bed8df167c23
- DebugLog (starting point for QuickSeq):
  - https://github.com/bnorm/debuglog
  - https://blog.bnorm.dev/writing-your-second-compiler-plugin-part-1
- PlantUML
  - https://plantuml.com/sequence-diagram
- Similar Projects
  - Kotlin Support is only experimental and does not allow to model a specific scenario (because of static code analysis) https://plugins.jetbrains.com/plugin/8286-sequencediagram
  - Supports class diagrams only: https://www.jetbrains.com/help/idea/class-diagram.html
- Where to find the source code
  - TBD, Navid will keep an overview

Images

- Icon of Kotlin, JetBrains, 1 August 2020, https://www.jetbrains.com/company/brand/logos/
- UML Logo, Object Management Group®, Inc. – embedded in http://www.uml.org/
- Gradle Logo, https://github.com/gradle/gradle/blob/35ed534acf7b97ff40ffabf5a3e156ea0b772437/gradle.png
- Tree Icon, https://icon-library.net/icon/family-tree-icon-13.html
- PlantUML Logo, Arnaud Rosques – http://plantuml.com/Logo_PlantUML.svg
- Telescope, made by https://www.flaticon.com/authors/srip from Flaticon
- Java Logo, https://brandslogos.com/j/java-logo-1/
- JavaScript Logo, Copyright (c) 2011 Christopher Williams chris@iterativedesigns.com