Kotlin + Desktop = Compose

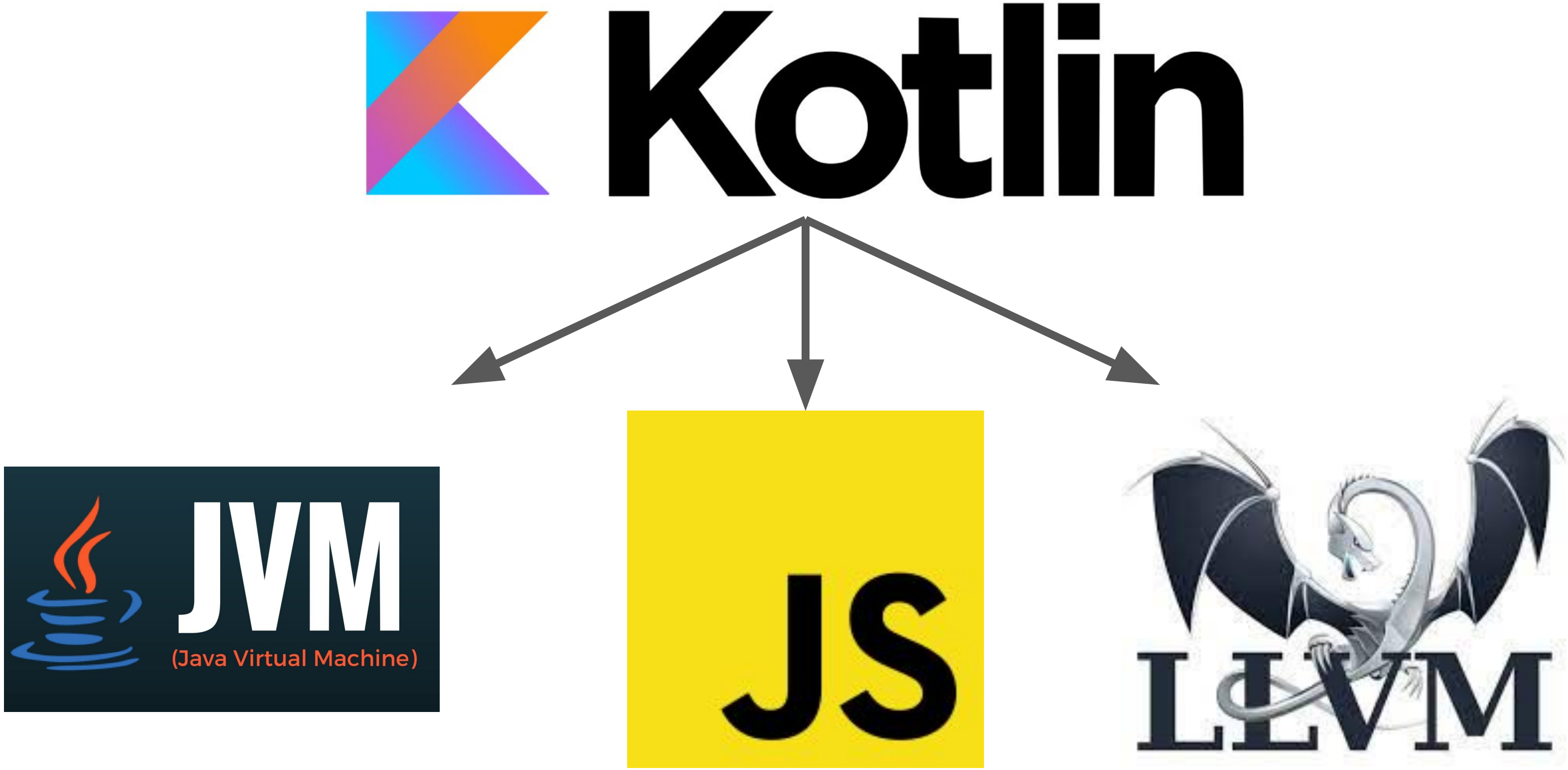
Chi sono?

- Simone Degiacomi
- Sviluppatore collaboratore di Defratech SRL
  - portali web (React, NestJS)
  - applicazioni mobile
- Ex-Microsoft Software Engineer

Kotlin

Kotlin

- Data classes
- Extension methods
- Domain Specific Language (DSL)
- Sealed classes
- Coroutines



Kotlin

JVM

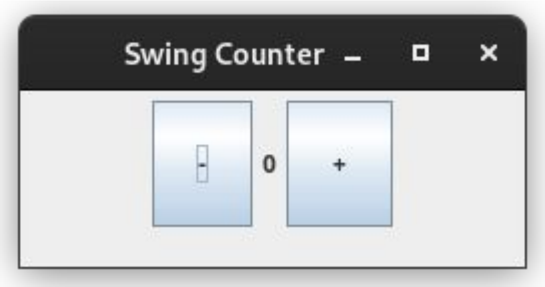
(Java Virtual Machine)

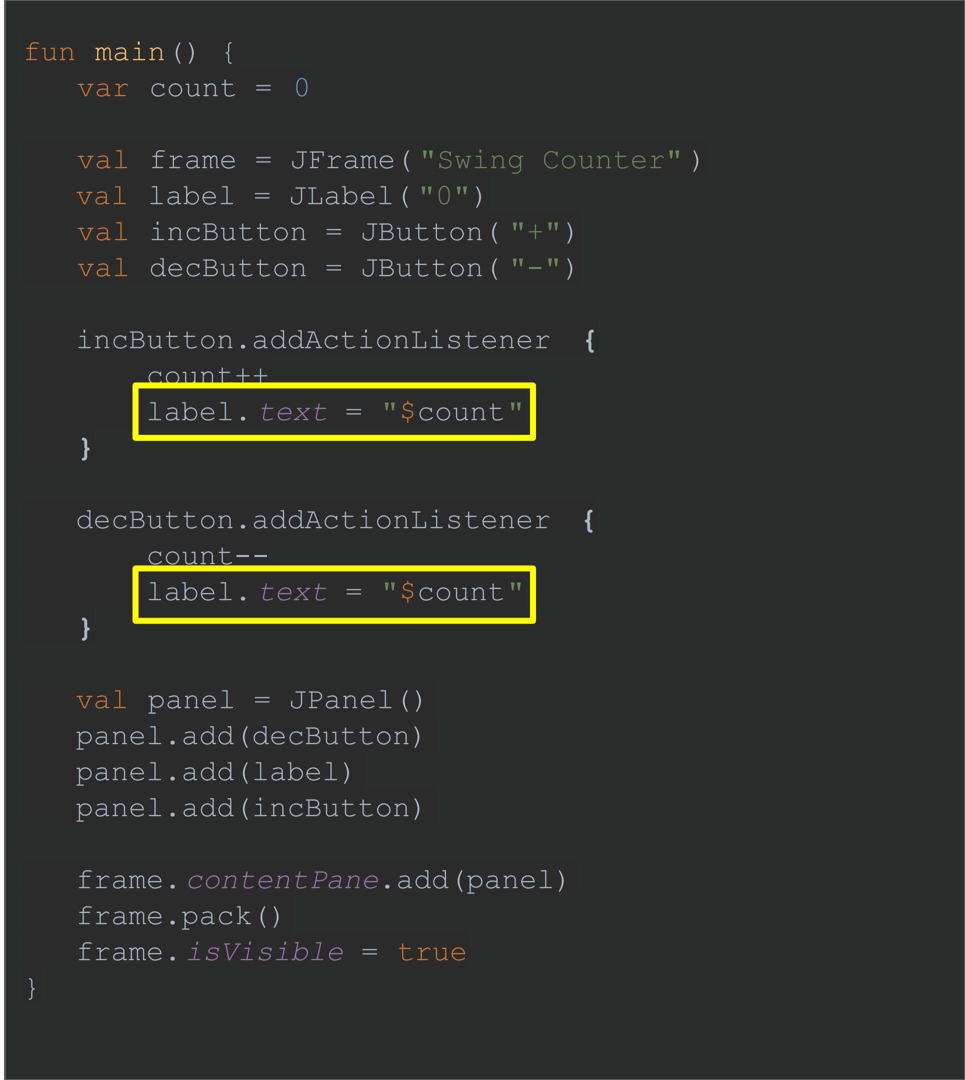
JS

LLVM

UI in Java - Swing

- Creazione verbosa delle interfacce
- Sincronizzazione degli stati

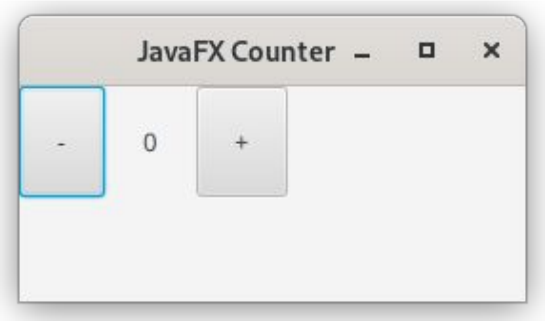


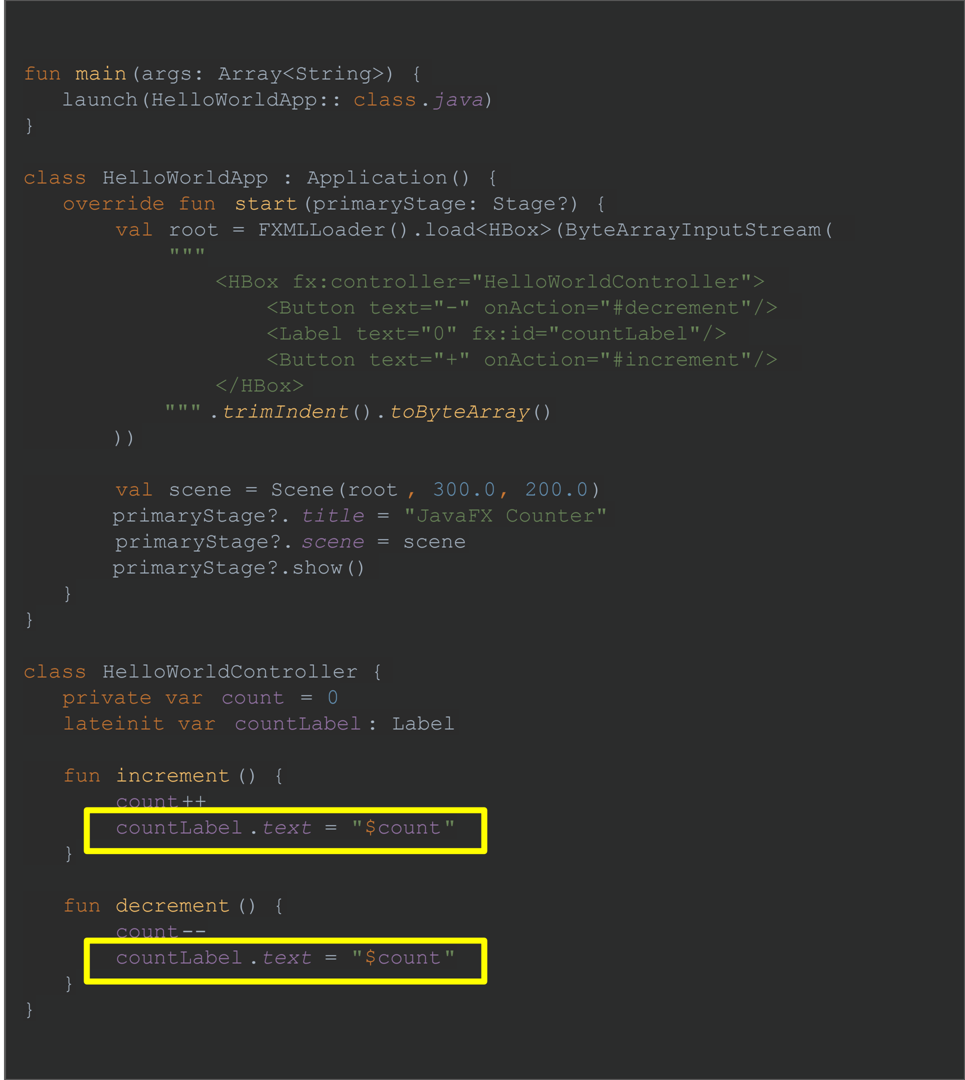


```
fun main() {
    var count = 0

    val frame = JFrame("Swing Counter")
    val label = JLabel("0")
    val incButton = JButton("+")
    val decButton = JButton("-")

    incButton.addActionListener {
        count++
        label.text = "$count"
    }

    decButton.addActionListener {
        count--
        label.text = "$count"
    }

    val panel = JPanel()
    panel.add(decButton)
    panel.add(label)
    panel.add(incButton)

    frame.contentPane.add(panel)
    frame.pack()
    frame.isVisible = true
}
```

UI in Java - Java FX

- ~~Creazione verbosa delle interfacce~~
- Sincronizzazione degli stati
- Binding tra XML e codice Java



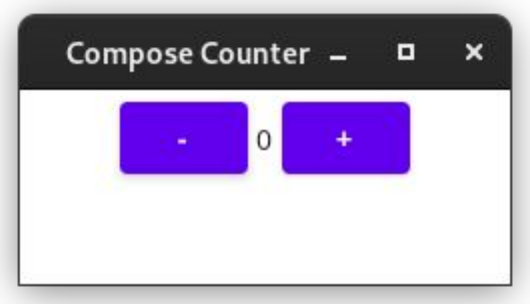


```kotlin
fun main(args: Array<String>) {
    launch(HelloWorldApp::class.java)
}

class HelloWorldApp : Application() {
    override fun start(primaryStage: Stage?) {
        val root = FXMLLoader().load<HBox>(ByteArrayInputStream(
            """
                <HBox fx:controller="HelloWorldController">
                    <Button text="-" onAction="#decrement"/>
                    <Label text="0" fx:id="countLabel"/>
                    <Button text="+" onAction="#increment"/>
                </HBox>
            """.trimIndent().toByteArray()
        ))

        val scene = Scene(root , 300.0, 200.0)
        primaryStage?.title = "JavaFX Counter"
        primaryStage?.scene = scene
        primaryStage?.show()
    }
}

class HelloWorldController {
    private var count = 0
    lateinit var countLabel: Label

    fun increment() {
        count++
        countLabel.text = "$count"
    }

    fun decrement() {
        count--
        countLabel.text = "$count"
    }
}
```

UI in Kotlin - Compose

- ~~Creazione verbosa delle interfacce~~
- ~~Sincronizzazione degli stati~~
- ~~Binding tra XML e codice Java~~

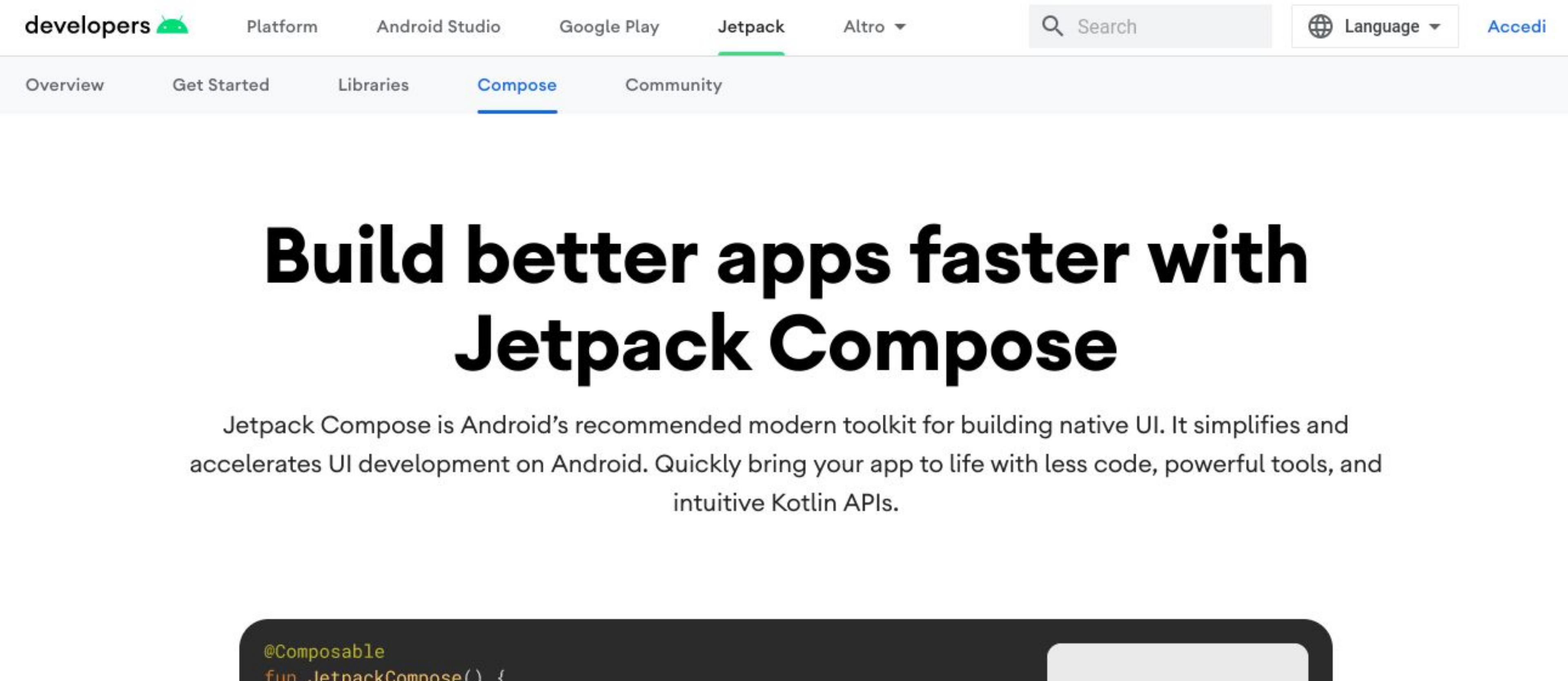
- Singola sorgente stato UI
- Codice dichiarativo



ComposeCounter.kt

```kotlin
fun main() = application {
    Window(
        onCloseRequest = ::exitApplication,
        title = "Compose Counter"
    ) {
        Counter()
    }
}

@Composable
fun Counter() {
    var counter by remember { mutableStateOf(0) }

    Row {
        Button(onClick = { counter -= 1 }) {
            Text("-")
        }

        Text("$counter")

        Button(onClick = { counter += 1 }) {
            Text("+")
        }
    }
}
```

Jetpack Compose



Build better apps faster with Jetpack Compose

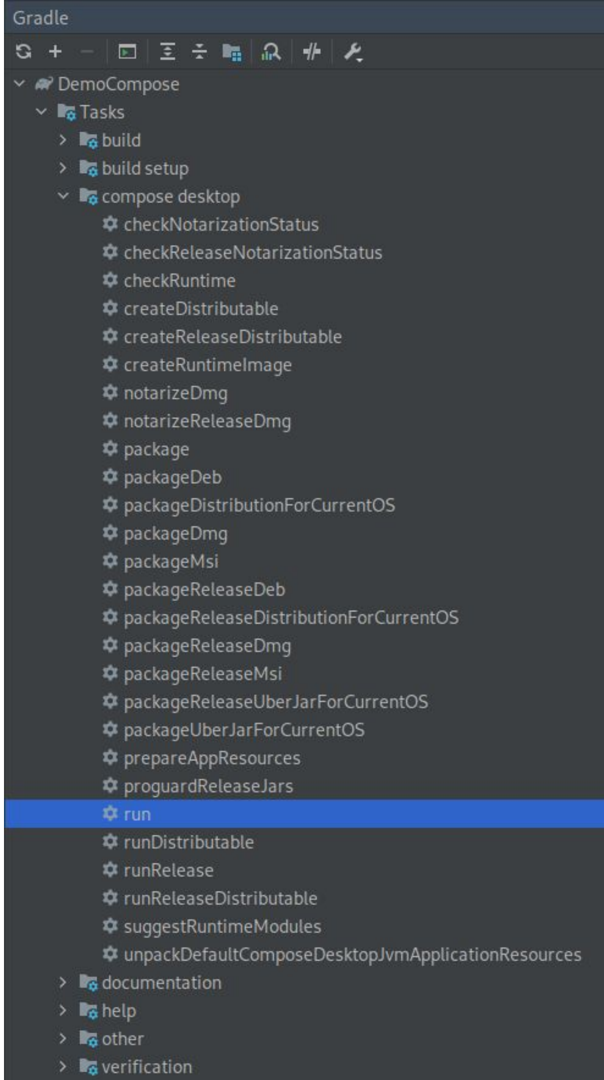
Jetpack Compose is Android’s recommended modern toolkit for building native UI. It simplifies and accelerates UI development on Android. Quickly bring your app to life with less code, powerful tools, and intuitive Kotlin APIs.

```
@Composable
fun JetpackCompose() {
```

Esempi di codice

- Gestione di più finestre
- SideEffects

Pacchettizzazione dell’applicazione



```
Gradle
DemoCompose
  Tasks
    build
    build setup
    compose desktop
      checkNotarizationStatus
      checkReleaseNotarizationStatus
      checkRuntime
      createDistributable
      createReleaseDistributable
      createRuntimeImage
      notarizeDmg
      notarizeReleaseDmg
      package
      packageDeb
      packageDistributionForCurrentOS
      packageDmg
      packageMsi
      packageReleaseDeb
      packageReleaseDistributionForCurrentOS
      packageReleaseDmg
      packageReleaseMsi
      packageReleaseUberJarForCurrentOS
      packageUberJarForCurrentOS
      prepareAppResources
      proguardReleaseJars
      run
      runDistributable
      runRelease
      runReleaseDistributable
      suggestRuntimeModules
      unpackDefaultComposeDesktopJvmApplicationResources
    documentation
    help
    other
    verification
```

Puntatori

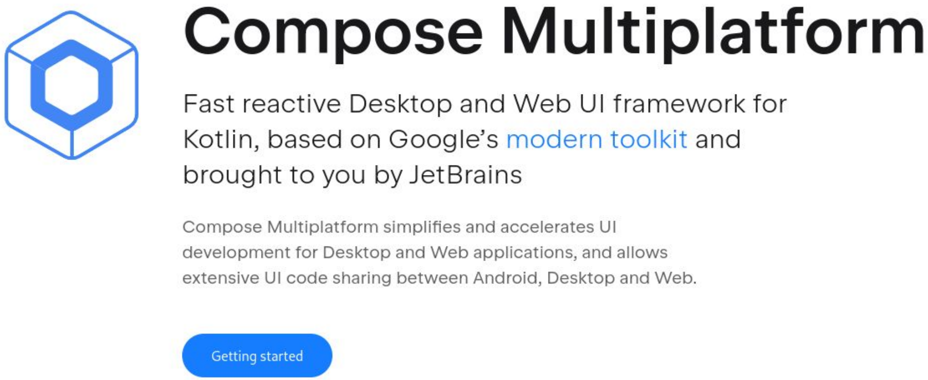
- JetBrains Compose samples: https://github.com/JetBrains/compose-jb/tree/master/examples
- Jetpack Compose Playground: https://foso.github.io/Jetpack-Compose-Playground/
- Documentazione Compose Android: https://developer.android.com/jetpack/compose/documentation
- Decompose: https://github.com/arkivanov/Decompose
- Kotlin MVI: https://github.com/arkivanov/MVIKotlin
- 7Guis: https://eugenkiss.github.io/7guis/tasks

Conclusione

- Breve storia delle interfacce grafiche per la JVM
- Concetto di UI dichiarativa
- Esempi di codice Compose Desktop

	Sviluppo Web	Compose
Layout	HTML	Kotlin DSL
Stile	CSS	Modifiers
Comportamento	JavaScript	*remember*, *mutableStateOf*, *LaunchedEffect*

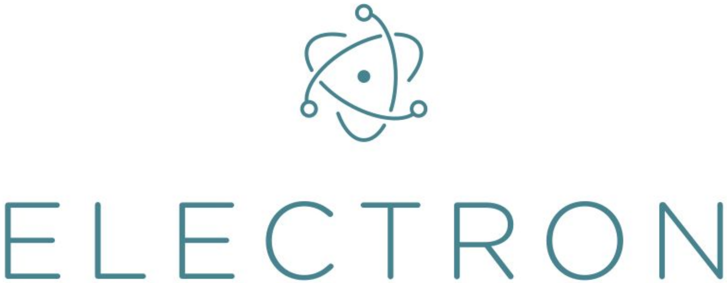
Domande?
Avete esperienze con Compose?



Compose Multiplatform

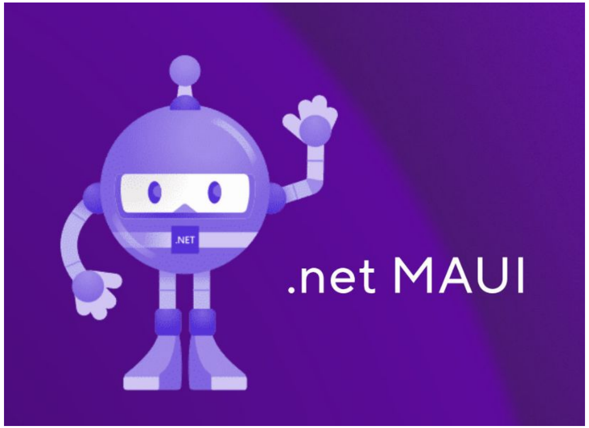
Fast reactive Desktop and Web UI framework for Kotlin, based on Google's modern toolkit and brought to you by JetBrains

Compose Multiplatform simplifies and accelerates UI development for Desktop and Web applications, and allows extensive UI code sharing between Android, Desktop and Web.


ELECTRON

Flutter

TAURI


.NET
.net MAUI