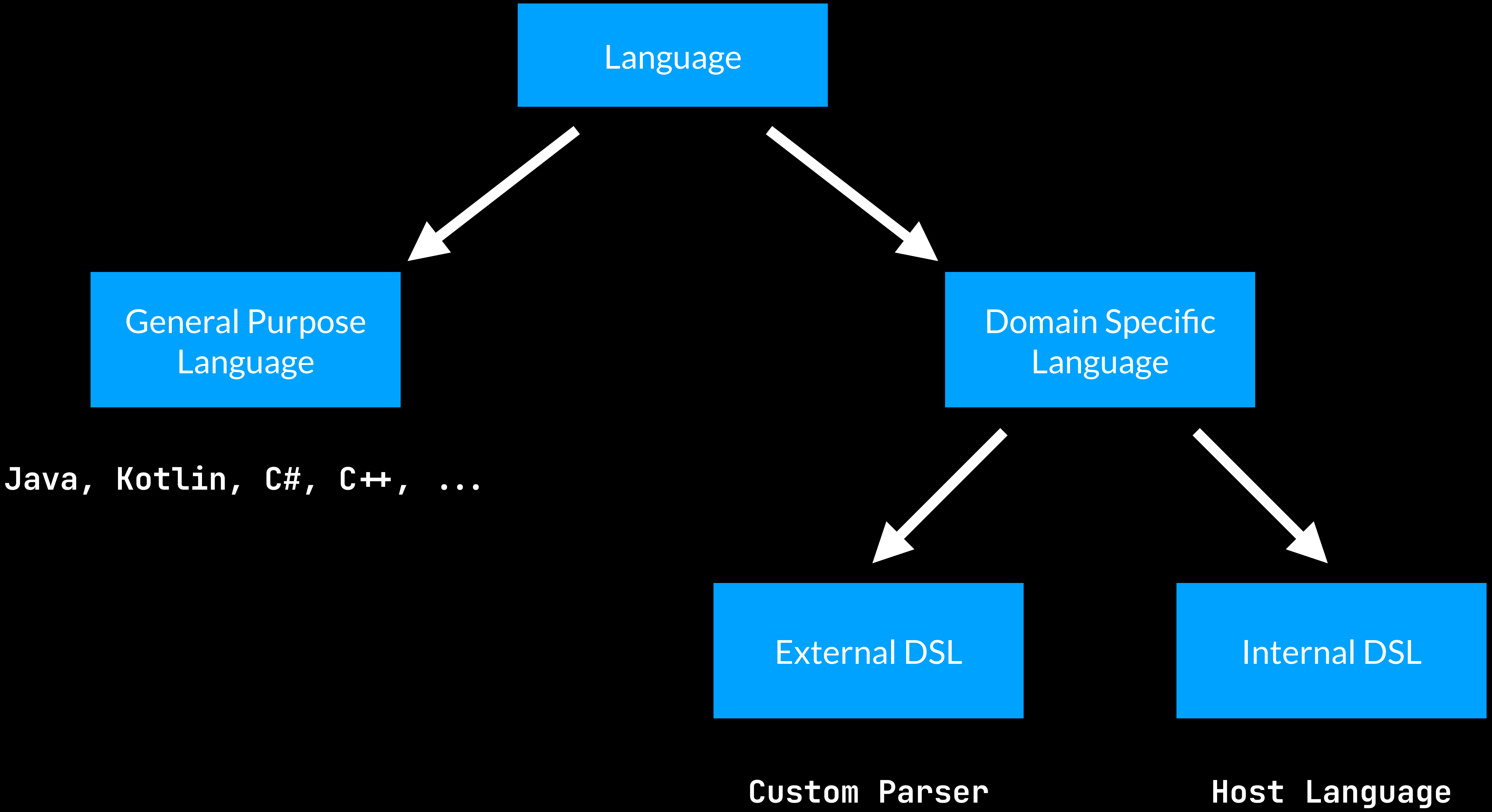
Algorithmic Photo Books

Kotlin DSLs demystified

Stefan Schöberl
schoeberl.dev
stefanschoeberl

Overview


Language
General Purpose Language
Domain Specific Language
Java, Kotlin, C#, C++, ...
External DSL
Internal DSL
Custom Parser
Host Language

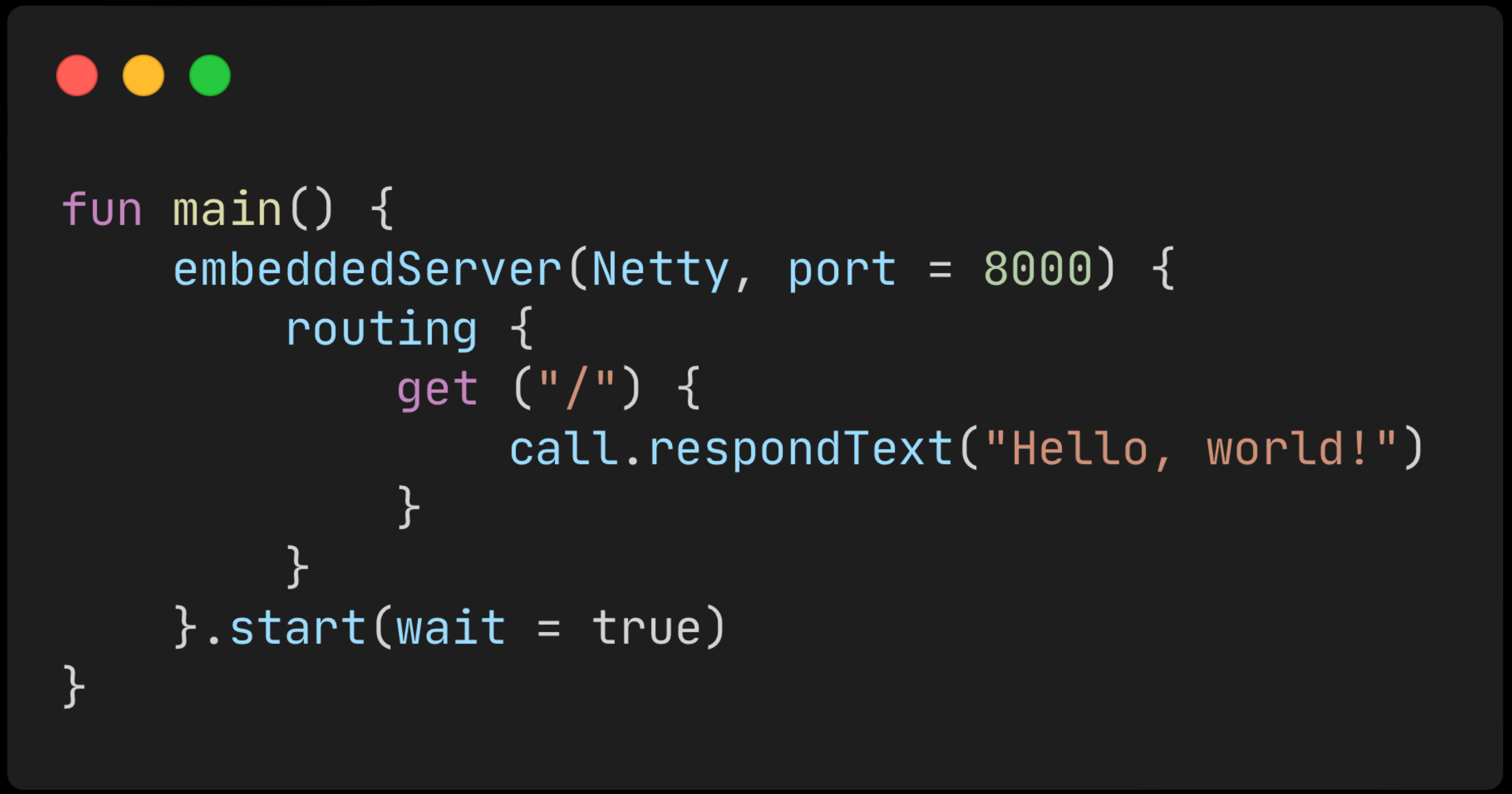
Kotlin DSLs

kotlinx.html

Ktor

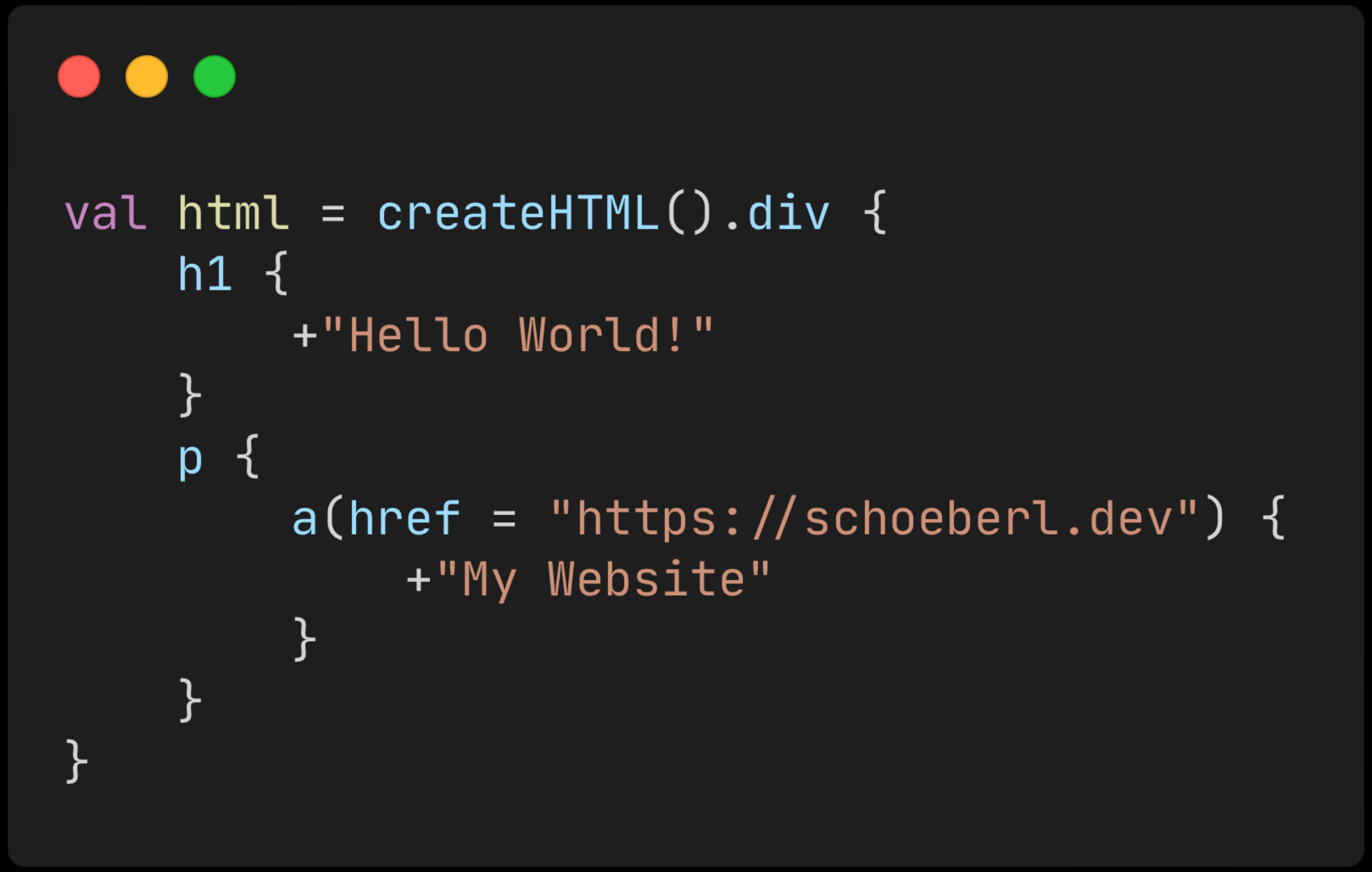
Gradle

Kotlin DSLs



```kotlin
fun main() {
    embeddedServer(Netty, port = 8000) {
        routing {
            get ("/") {
                call.respondText("Hello, world!")
            }
        }
    }.start(wait = true)
}
```

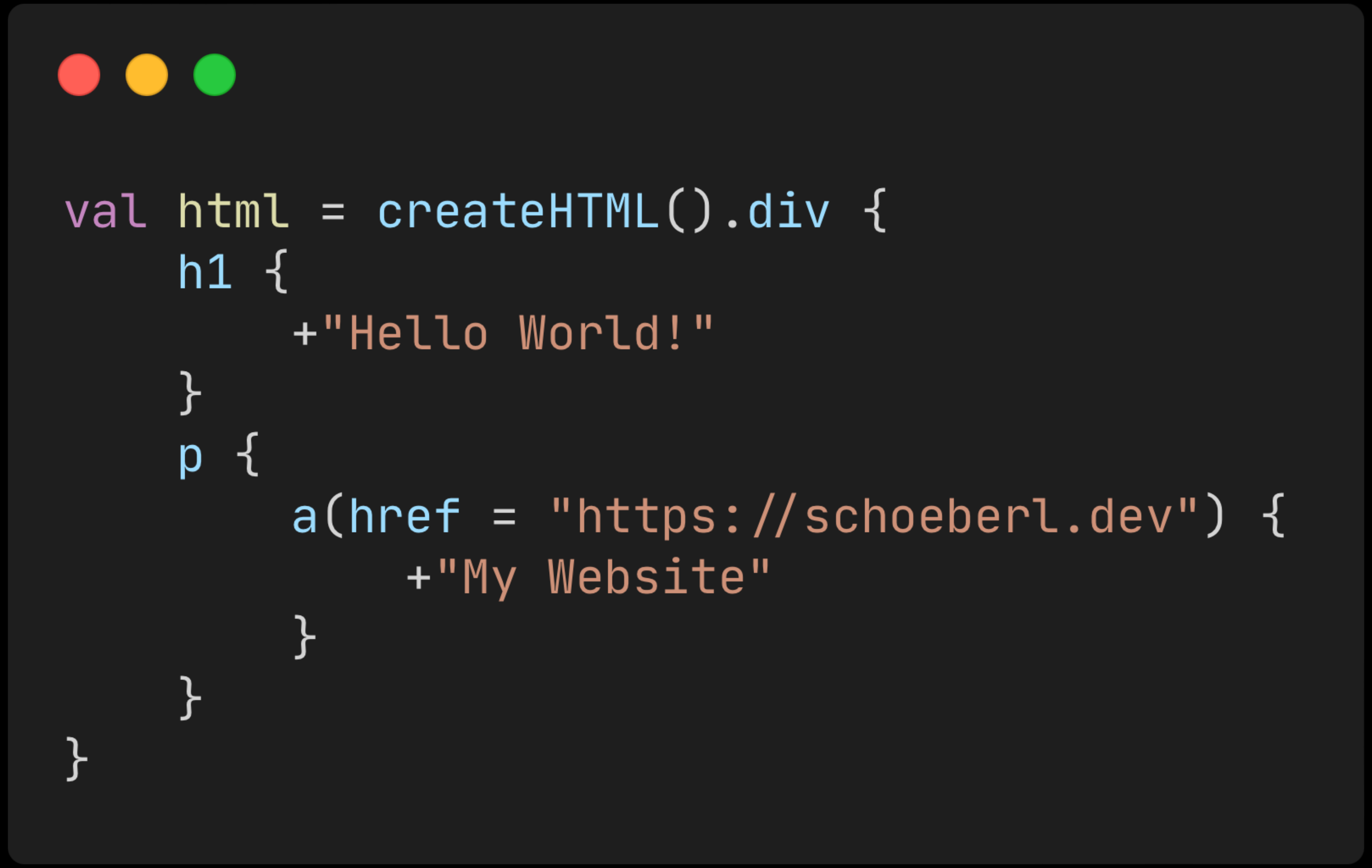
Ktor



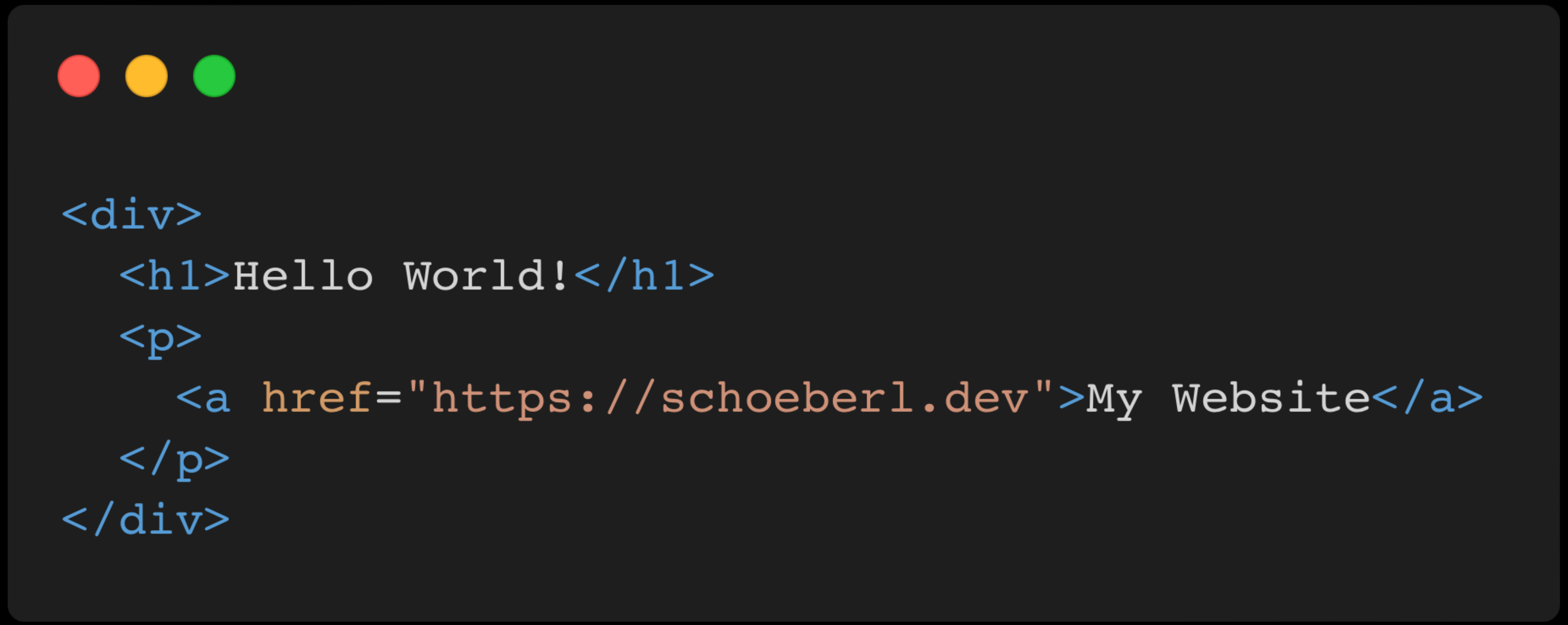
```kotlin
val html = createHTML().div {
    h1 {
        +"Hello World!"
    }
    p {
        a(href = "https://schoeberl.dev") {
            +"My Website"
        }
    }
}
```

kotlinx.html

Kotlin DSLs



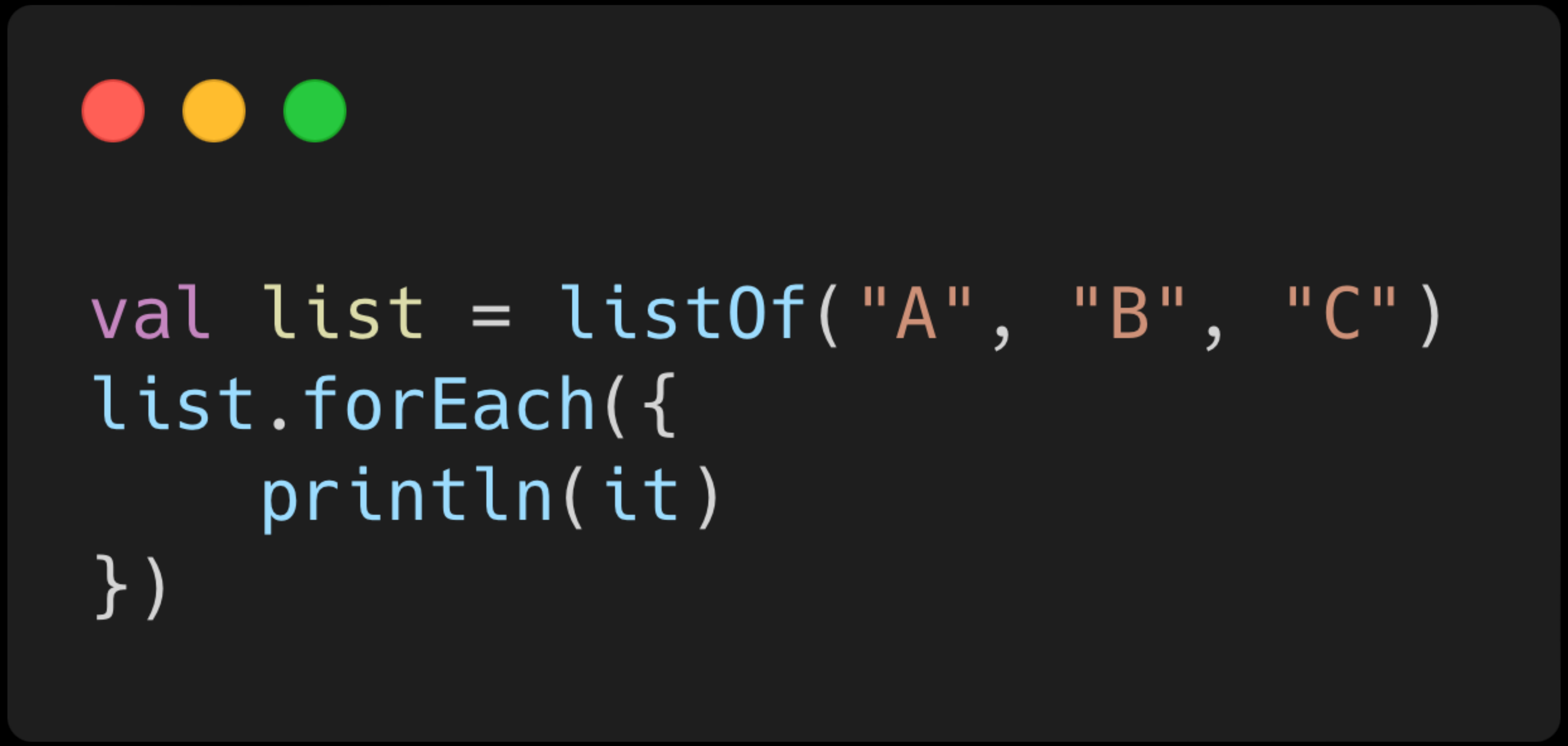
```kotlin
val html = createHTML().div {
    h1 {
        +"Hello World!"
    }
    p {
        a(href = "https://schoeberl.dev") {
            +"My Website"
        }
    }
}
```



```html
<div>
  <h1>Hello World!</h1>
  <p>
    <a href="https://schoeberl.dev">My Website</a>
  </p>
</div>
```

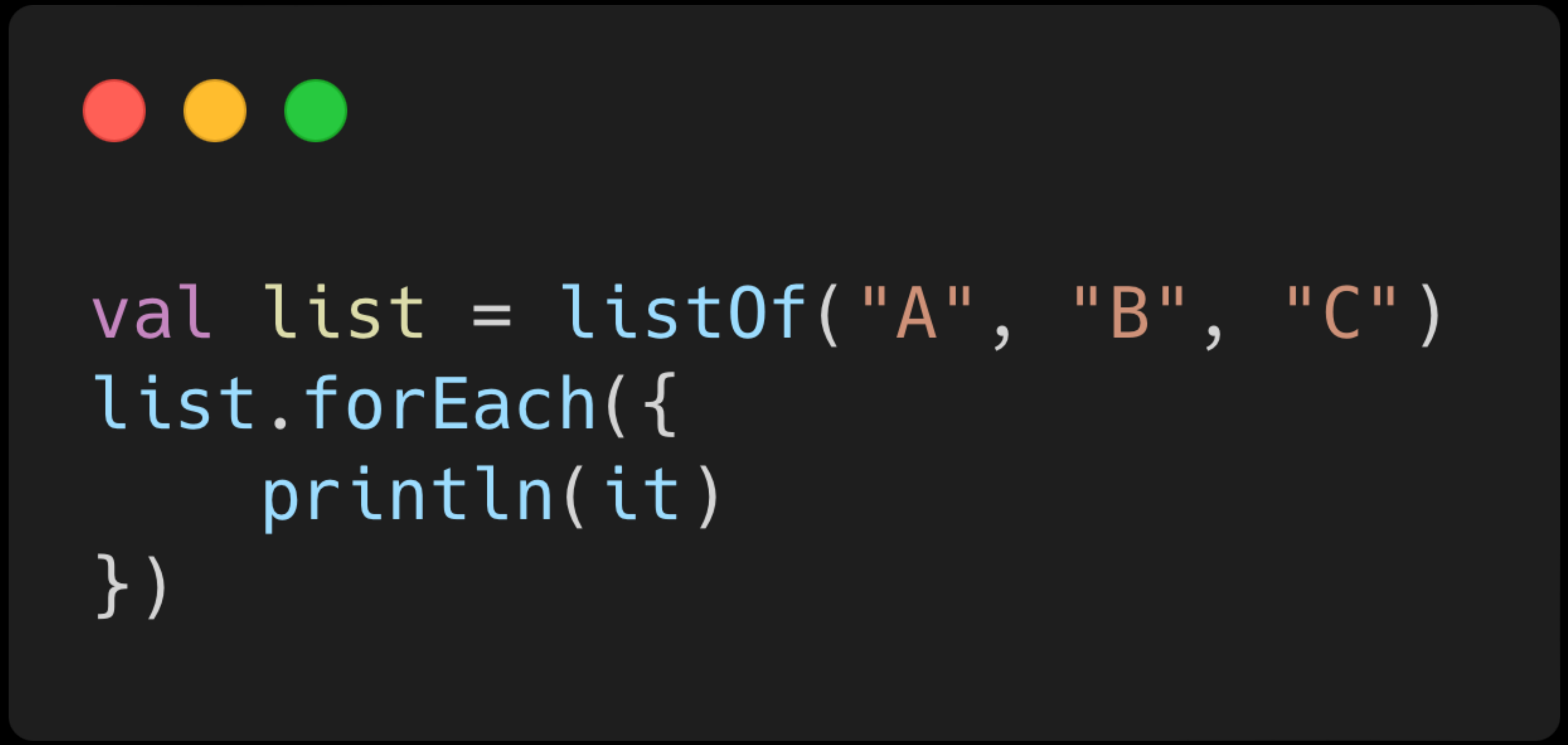
`kotlinx.html`

#1 Lambda as last parameter

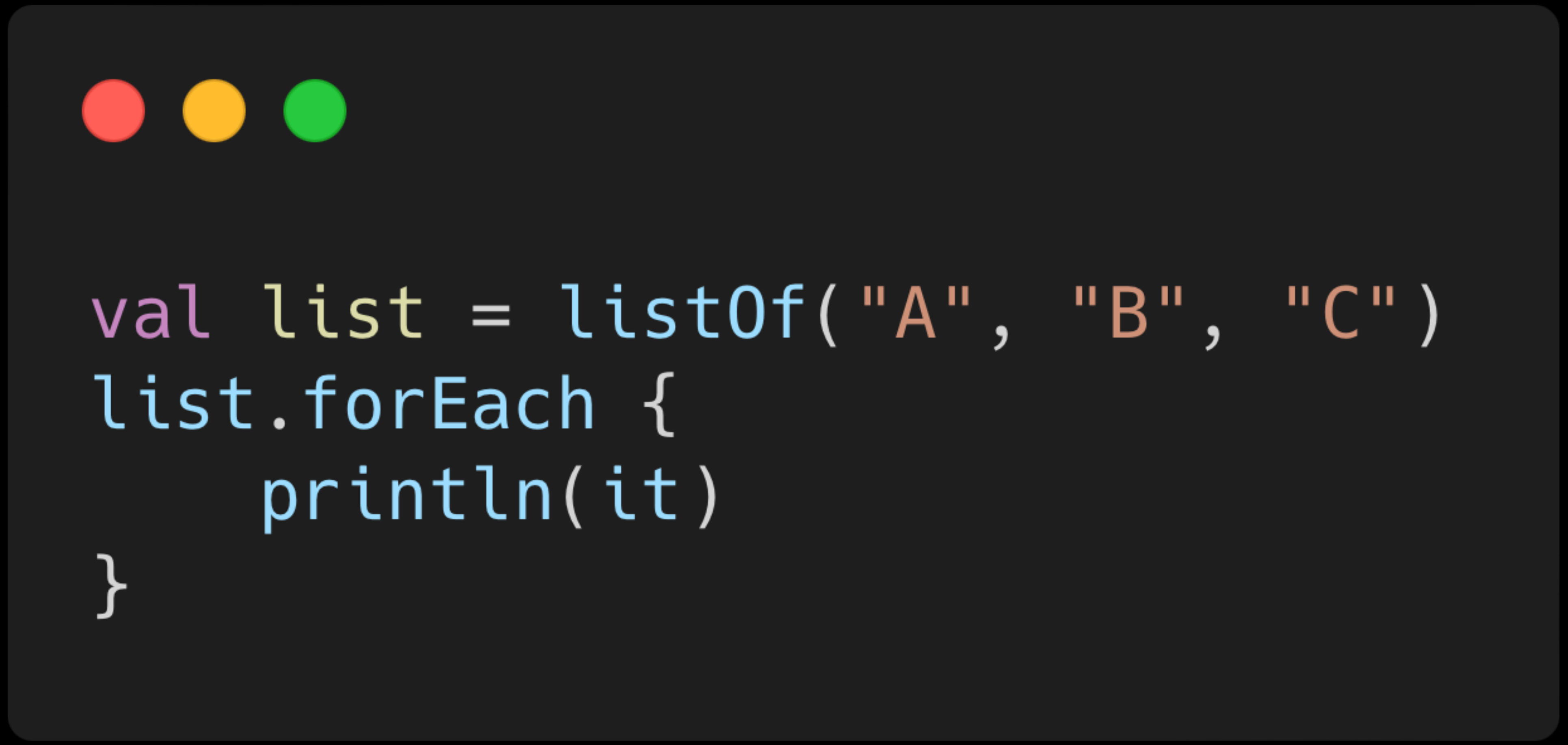


```kotlin
val list = listOf("A", "B", "C")
list.forEach({
    println(it)
})
```

#1 Lambda as last parameter

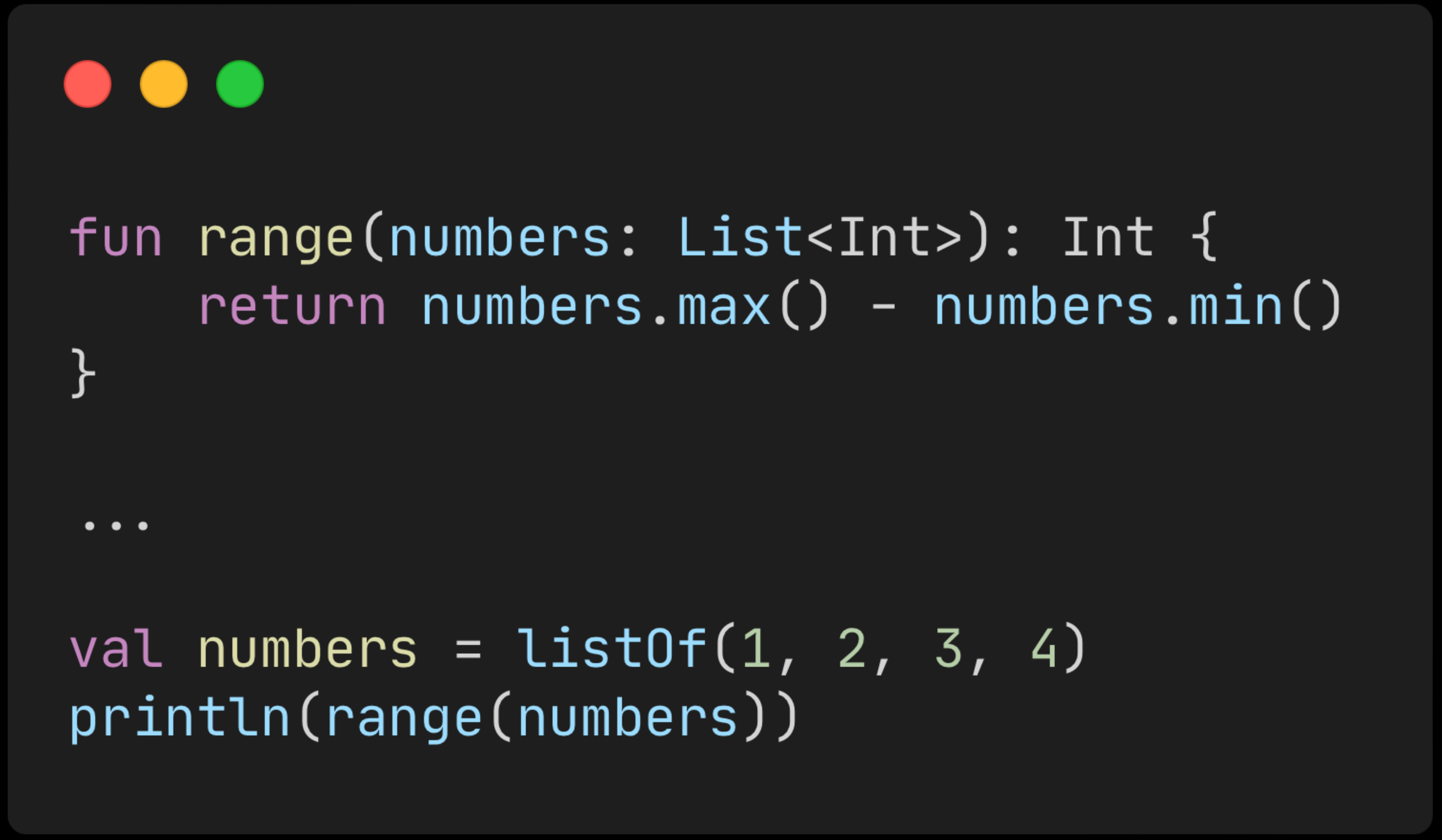


```kotlin
val list = listOf("A", "B", "C")
list.forEach({
    println(it)
})
```



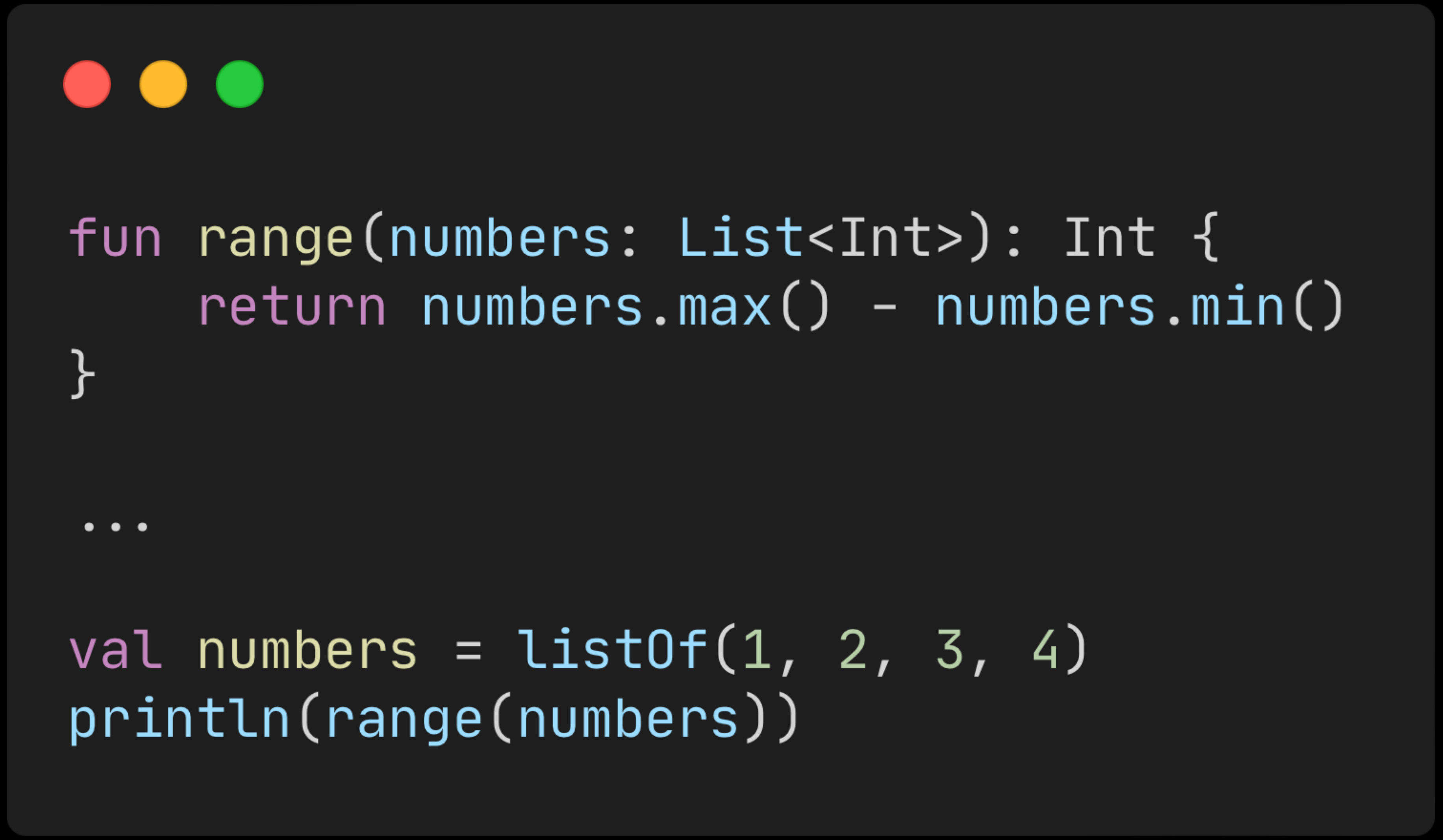
```kotlin
val list = listOf("A", "B", "C")
list.forEach {
    println(it)
}
```

#2 Extension functions

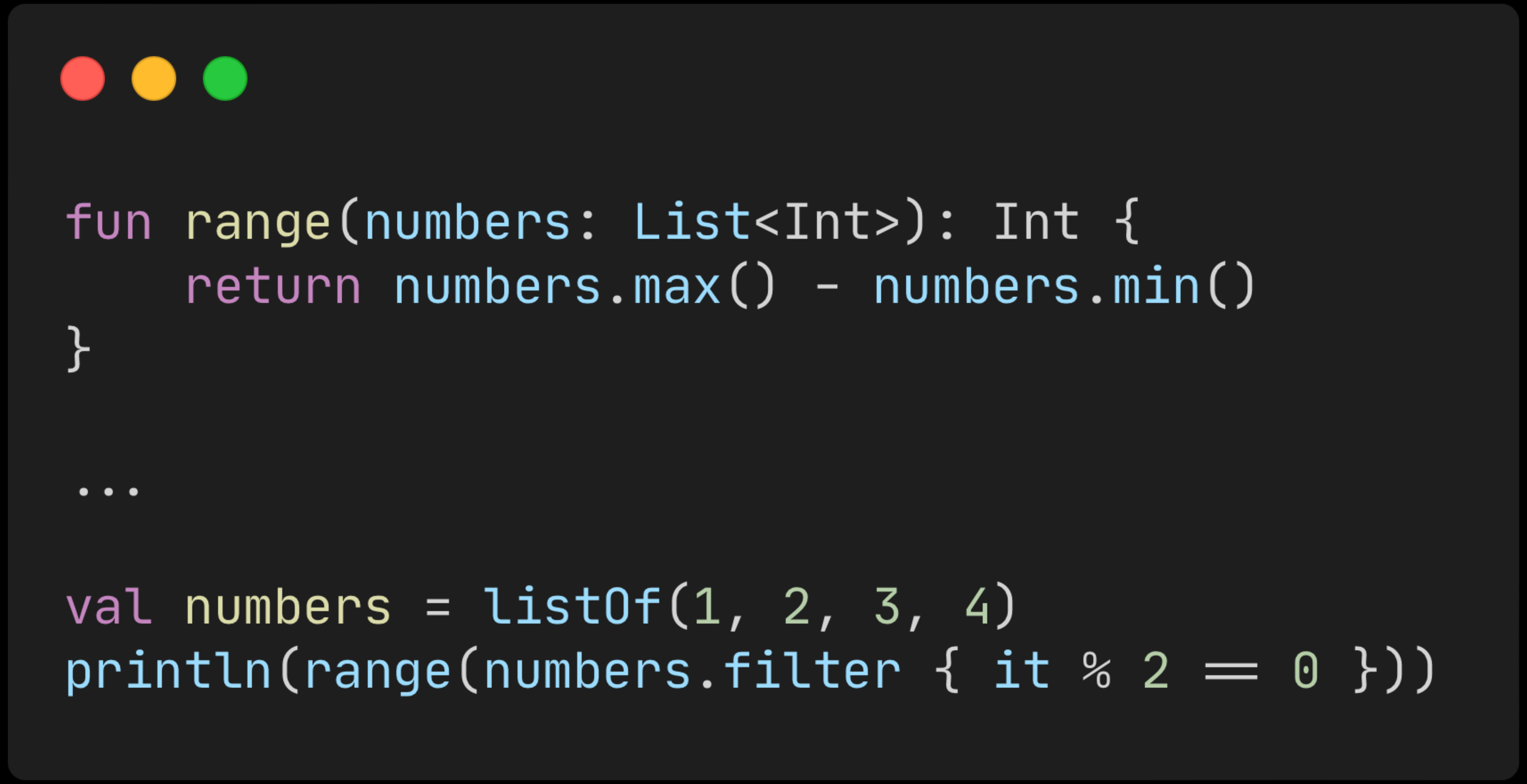


```
fun range(numbers: List<Int>): Int {
    return numbers.max() - numbers.min()
}

...

val numbers = listOf(1, 2, 3, 4)
println(range(numbers))
```

#2 Extension functions

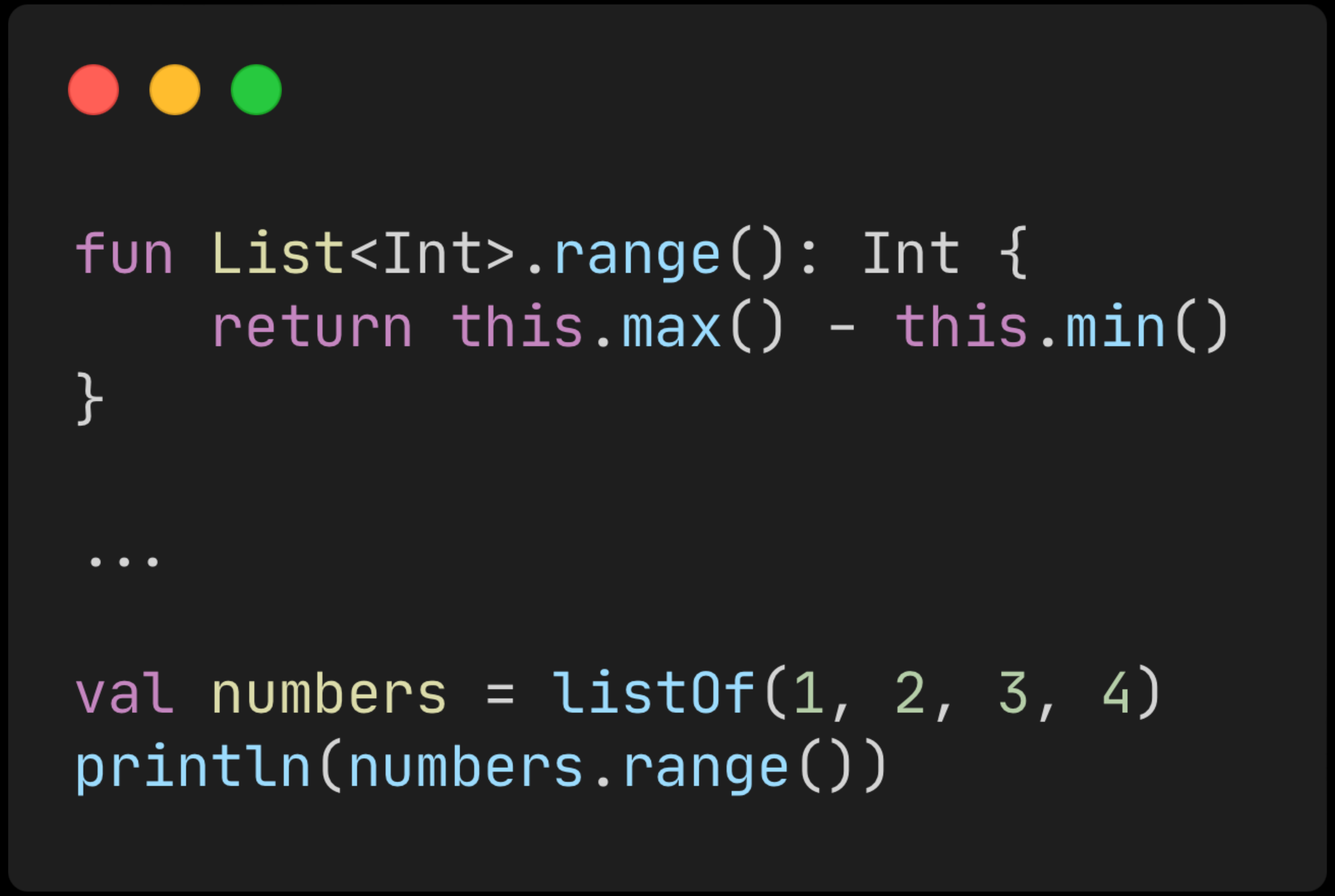


```kotlin
fun range(numbers: List<Int>): Int {
    return numbers.max() - numbers.min()
}

...

val numbers = listOf(1, 2, 3, 4)
println(range(numbers))
```

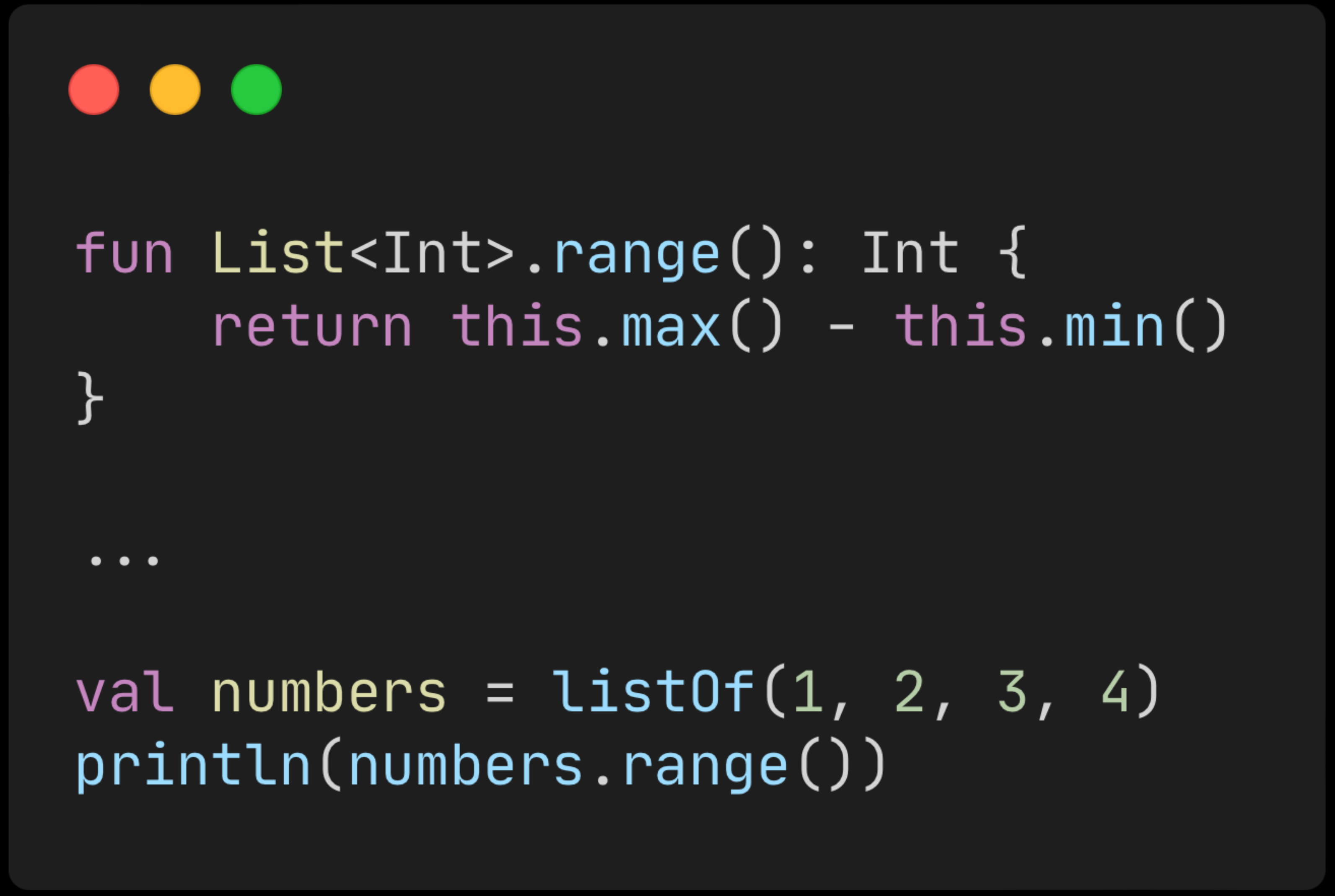


```kotlin
fun range(numbers: List<Int>): Int {
    return numbers.max() - numbers.min()
}

...

val numbers = listOf(1, 2, 3, 4)
println(range(numbers.filter { it % 2 == 0 }))
```

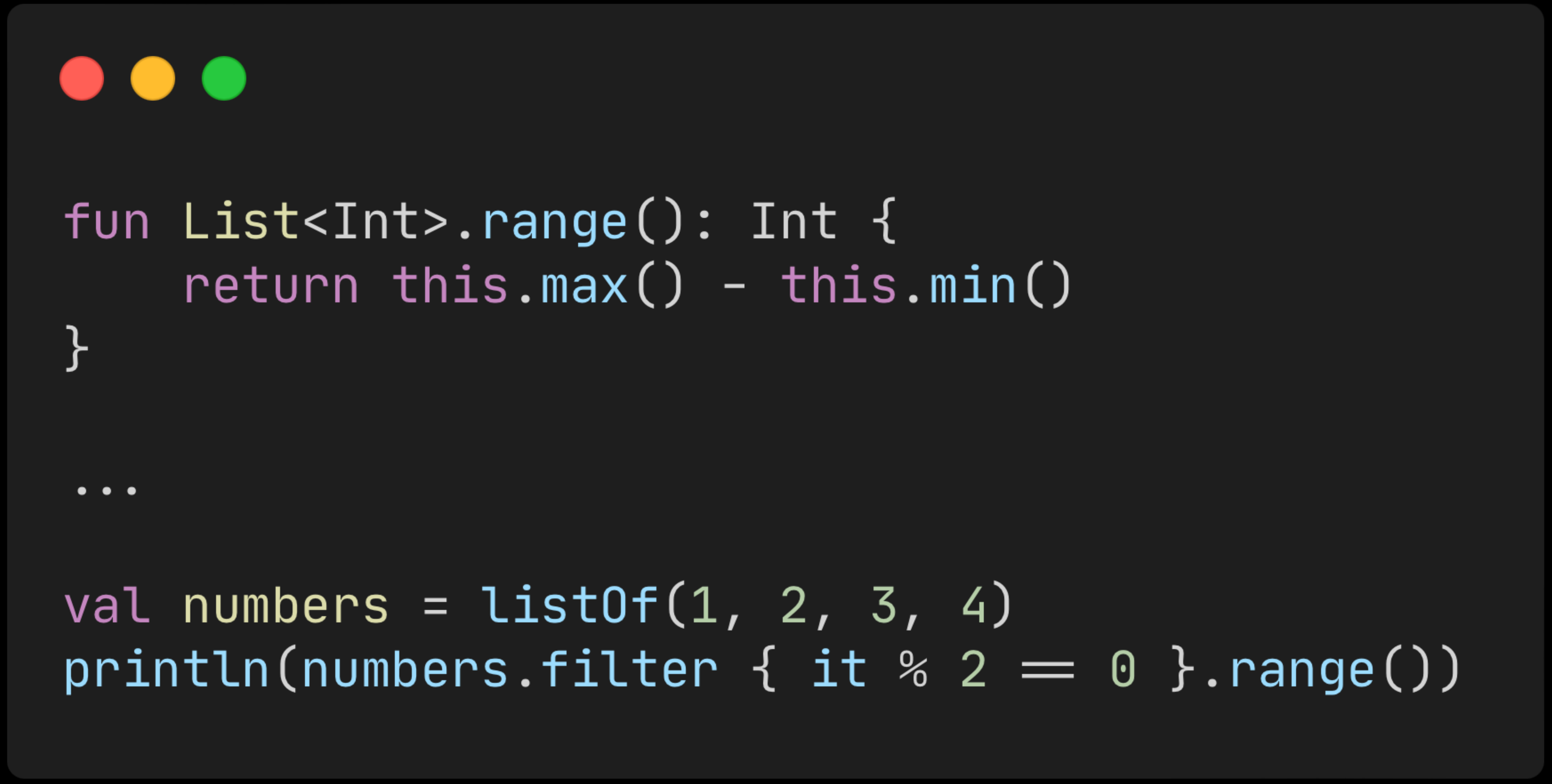
#2 Extension functions


```kotlin
fun List<Int>.range(): Int {
    return this.max() - this.min()
}

...

val numbers = listOf(1, 2, 3, 4)
println(numbers.range())
```

#2 Extension functions

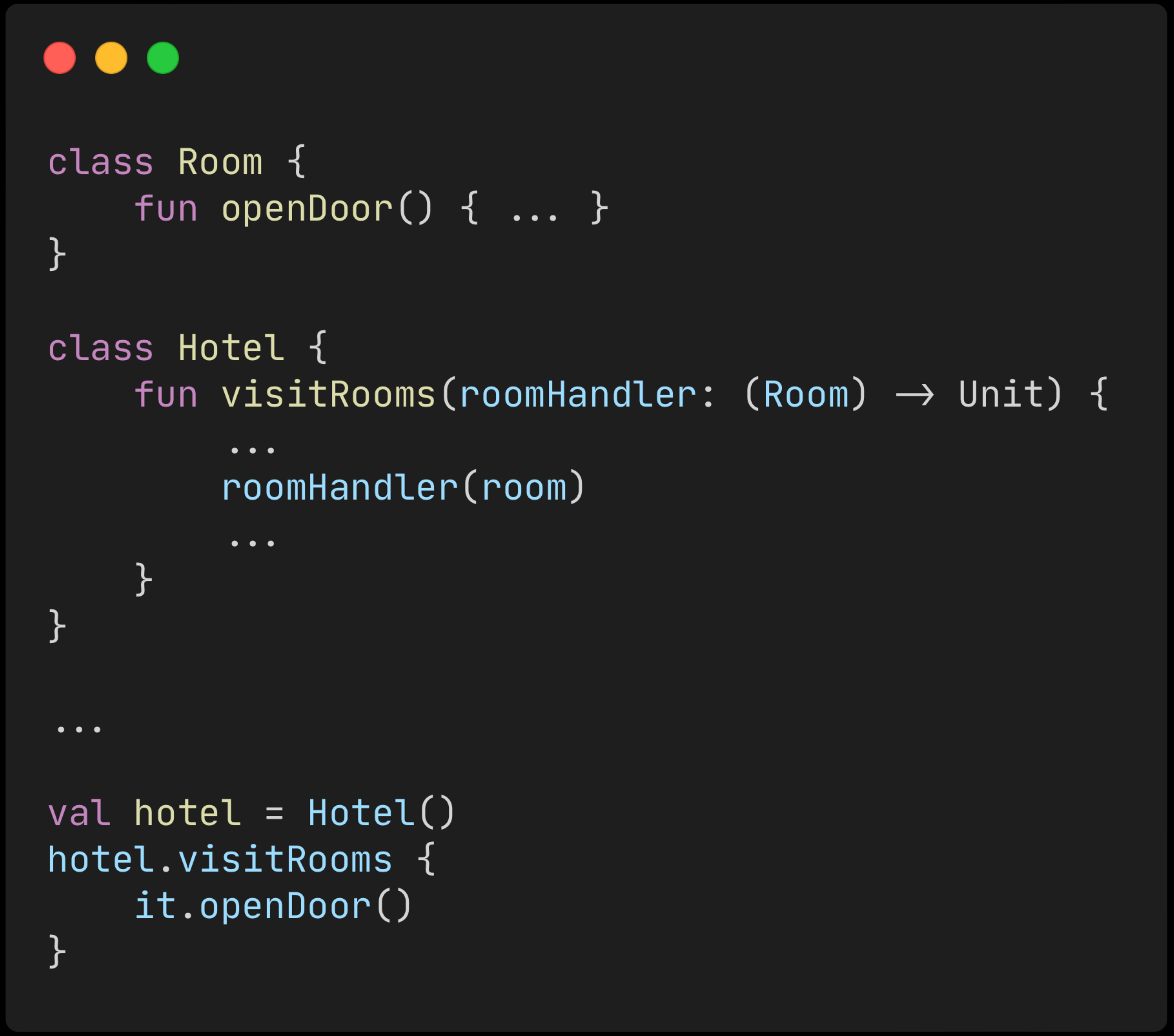


```kotlin
fun List<Int>.range(): Int {
    return this.max() - this.min()
}

...

val numbers = listOf(1, 2, 3, 4)
println(numbers.range())
```



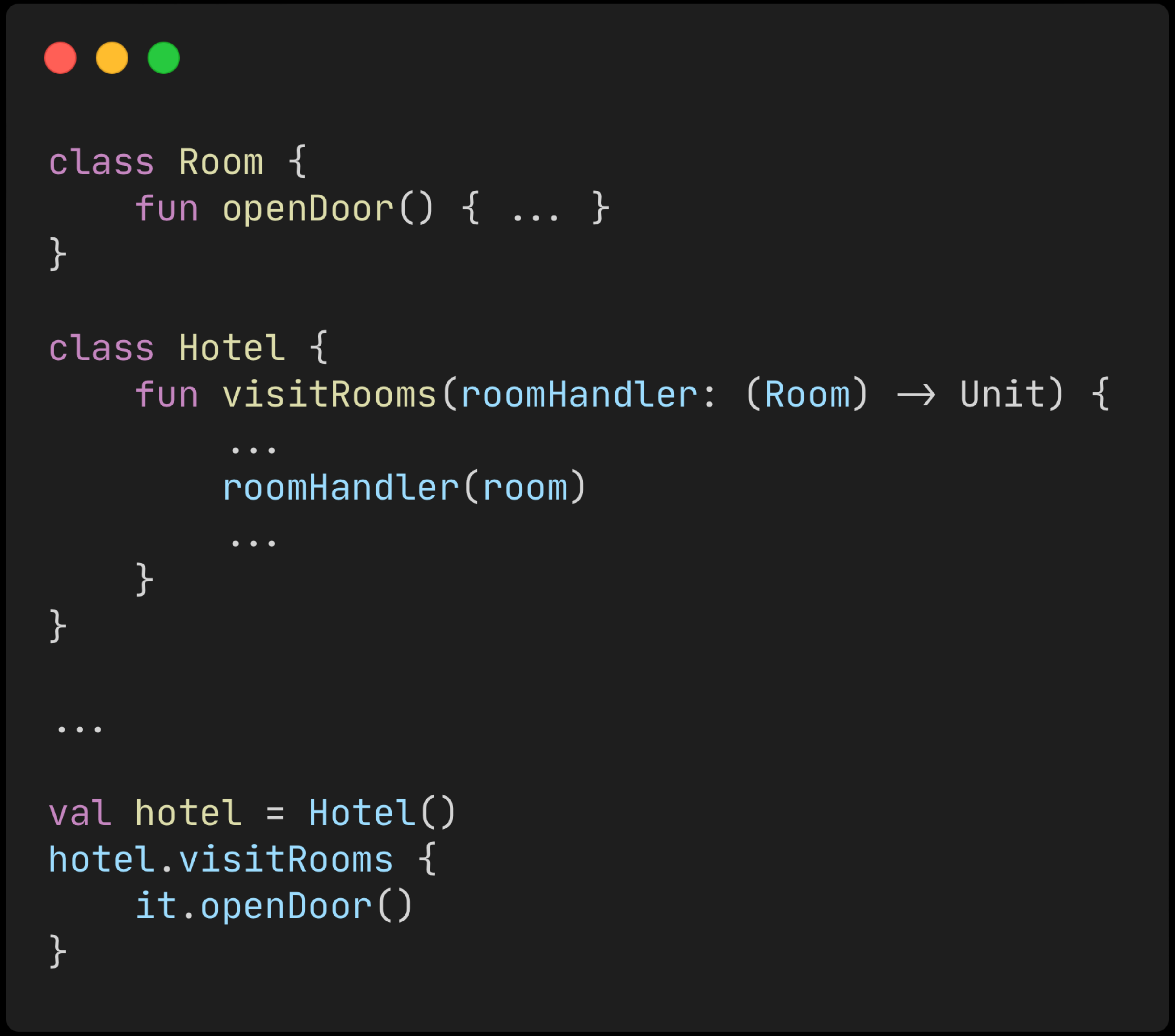
```kotlin
fun List<Int>.range(): Int {
    return this.max() - this.min()
}

...

val numbers = listOf(1, 2, 3, 4)
println(numbers.filter { it % 2 == 0 }.range())
```

#2 Extension functions

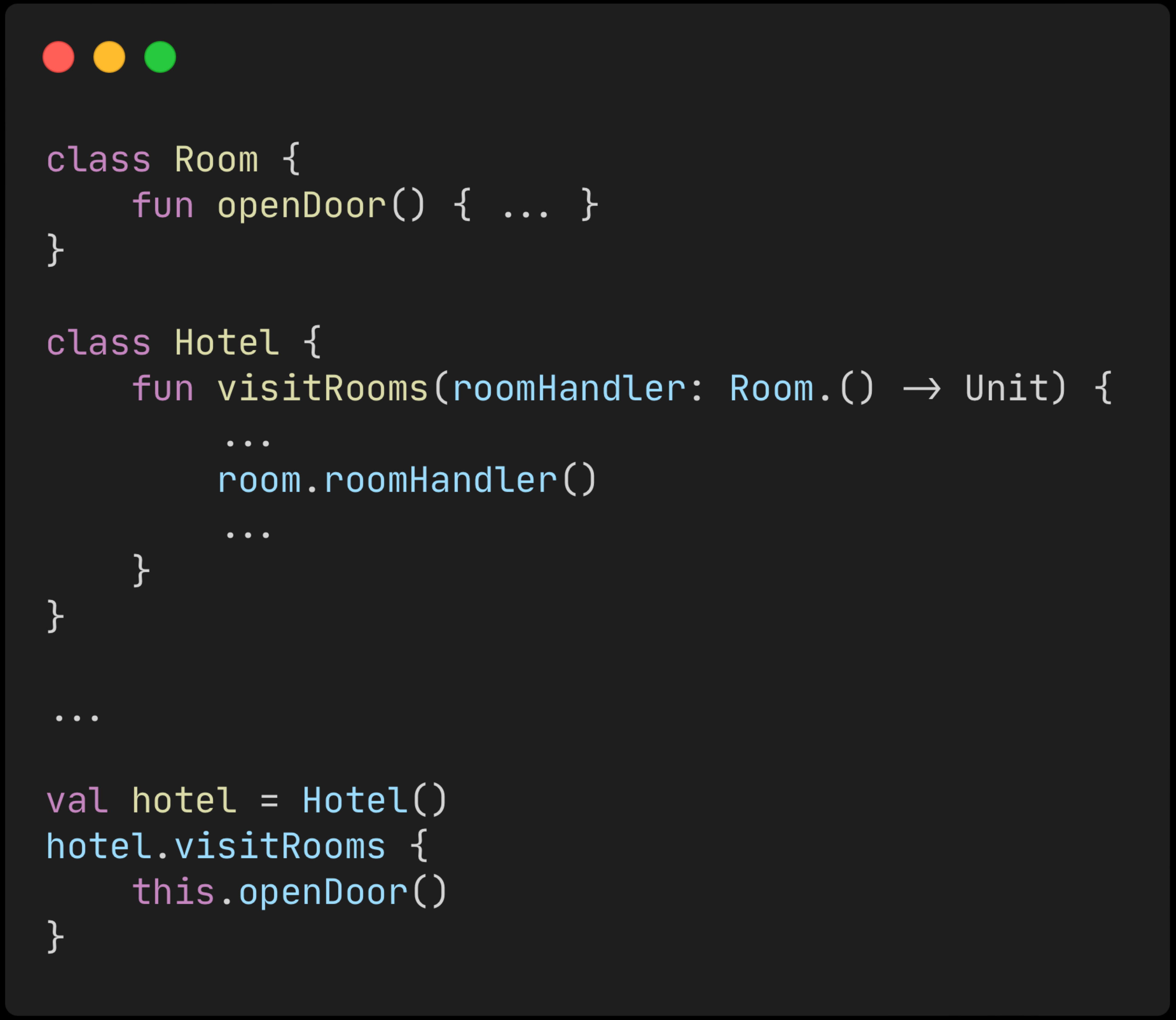


```kotlin
class Room {
    fun openDoor() { ... }
}

class Hotel {
    fun visitRooms(roomHandler: (Room) -> Unit) {
        ...
        roomHandler(room)
        ...
    }
}

...

val hotel = Hotel()
hotel.visitRooms {
    it.openDoor()
}
```

#2 Extension functions

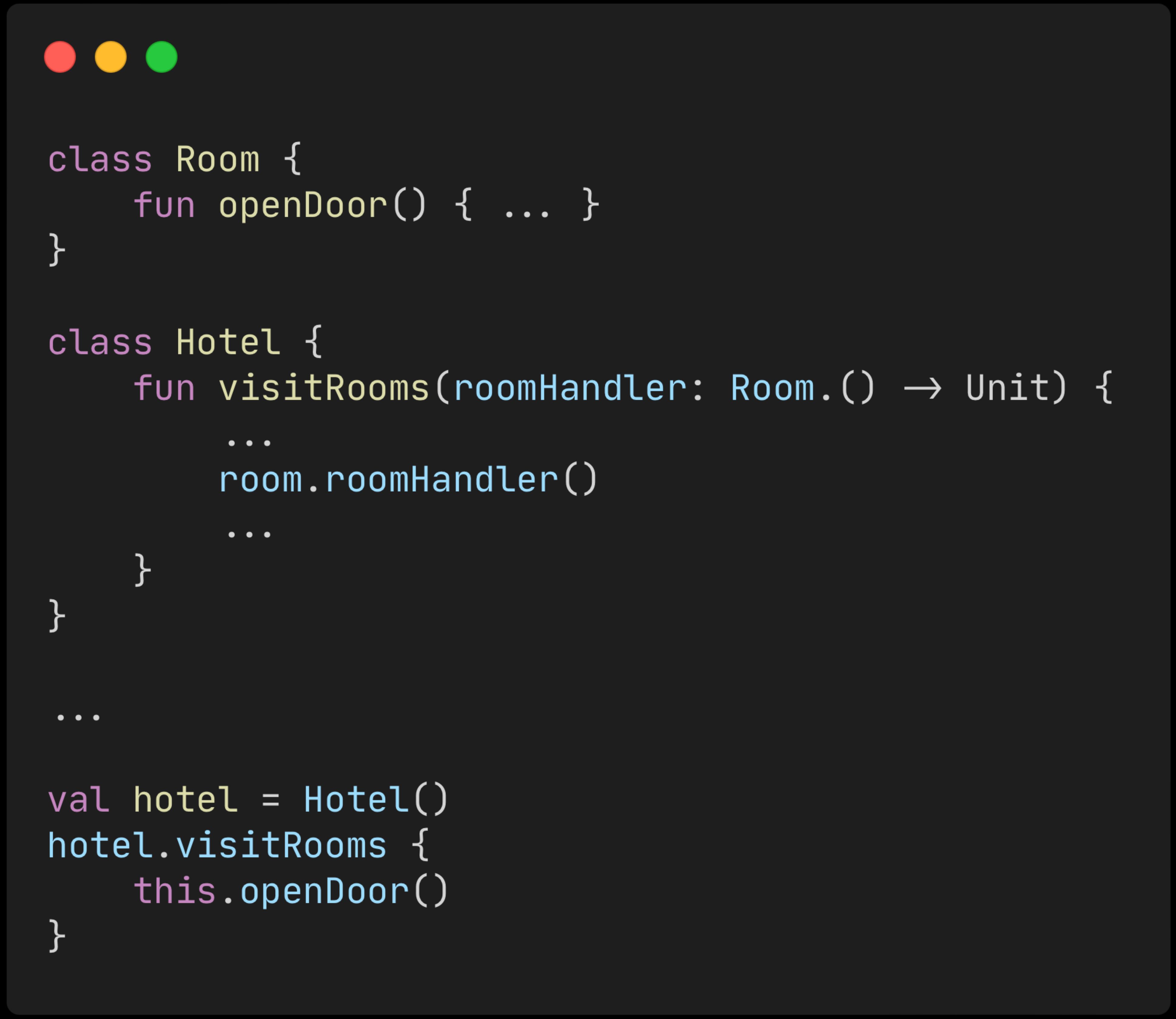


```kotlin
class Room {
    fun openDoor() { ... }
}

class Hotel {
    fun visitRooms(roomHandler: (Room) → Unit) {
        ...
        roomHandler(room)
        ...
    }
}

...

val hotel = Hotel()
hotel.visitRooms {
    it.openDoor()
}
```



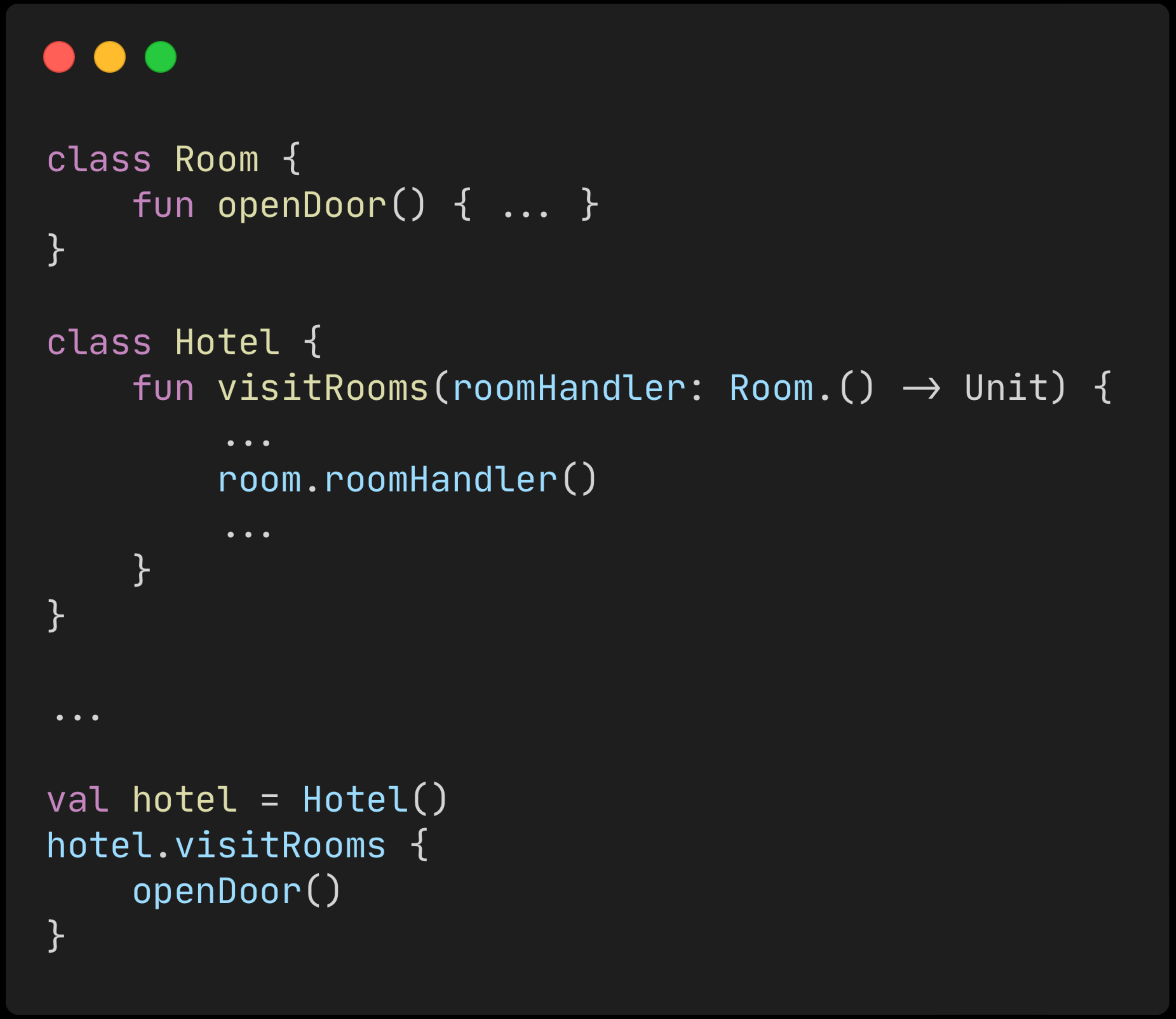
```kotlin
class Room {
    fun openDoor() { ... }
}

class Hotel {
    fun visitRooms(roomHandler: Room.() → Unit) {
        ...
        room.roomHandler()
        ...
    }
}

...

val hotel = Hotel()
hotel.visitRooms {
    this.openDoor()
}
```

#2 Extension functions



```
class Room {
    fun openDoor() { ... }
}

class Hotel {
    fun visitRooms(roomHandler: Room.() → Unit) {
        ...
        room.roomHandler()
        ...
    }
}

...

val hotel = Hotel()
hotel.visitRooms {
    this.openDoor()
}
```

→

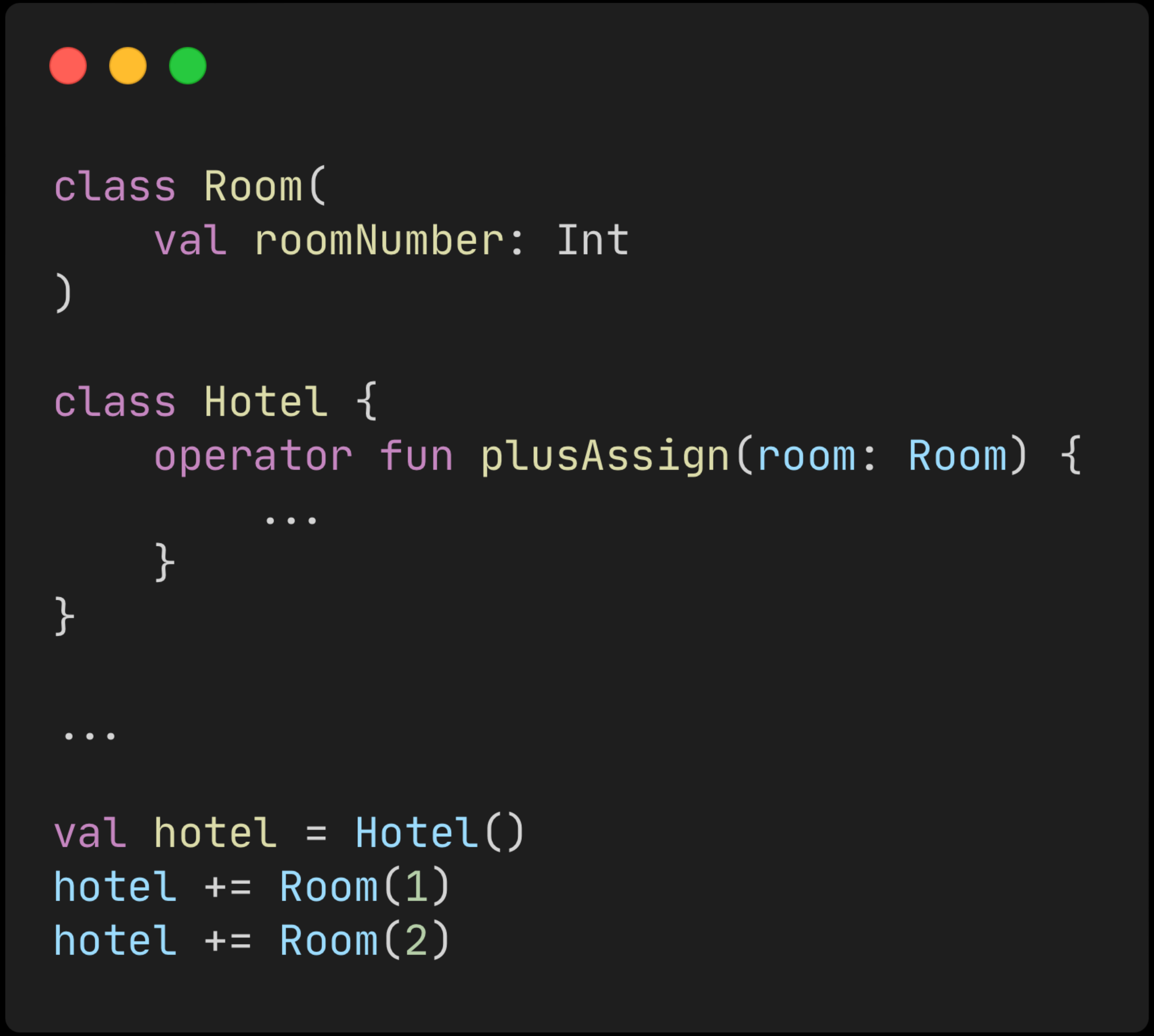


```
class Room {
    fun openDoor() { ... }
}

class Hotel {
    fun visitRooms(roomHandler: Room.() → Unit) {
        ...
        room.roomHandler()
        ...
    }
}

...

val hotel = Hotel()
hotel.visitRooms {
    openDoor()
}
```

#3 Operator overloading

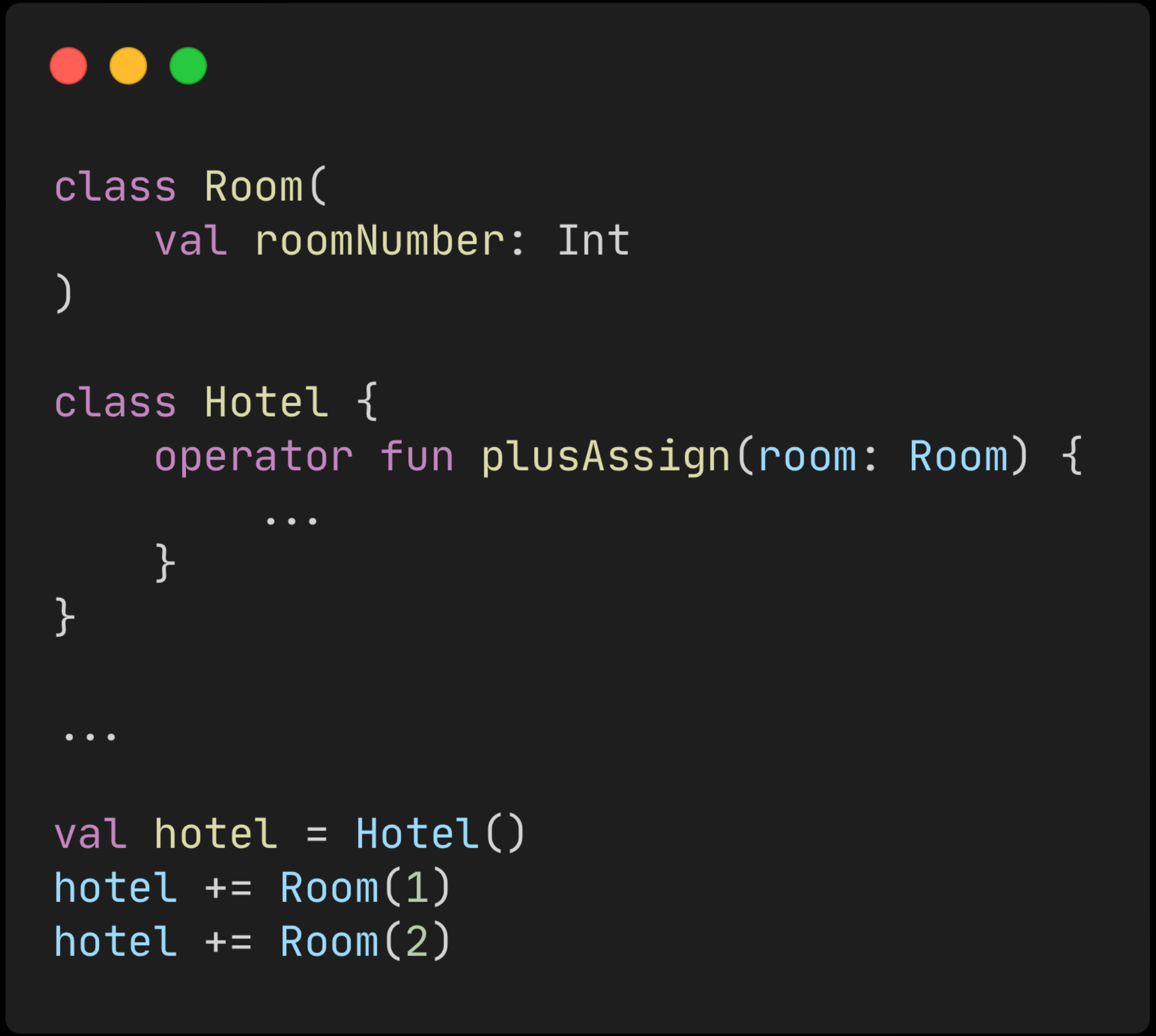
`a + b`	→	`a.plus(b)`
`a - b`	→	`a.minus(b)`
`a..b`	→	`a.rangeTo(b)`
`a[i]`	→	`a.get(i)`
`+a`	→	`a.unaryPlus()`
`a > b`	→	`a.compareTo(b) > 0`

#3 Operator overloading

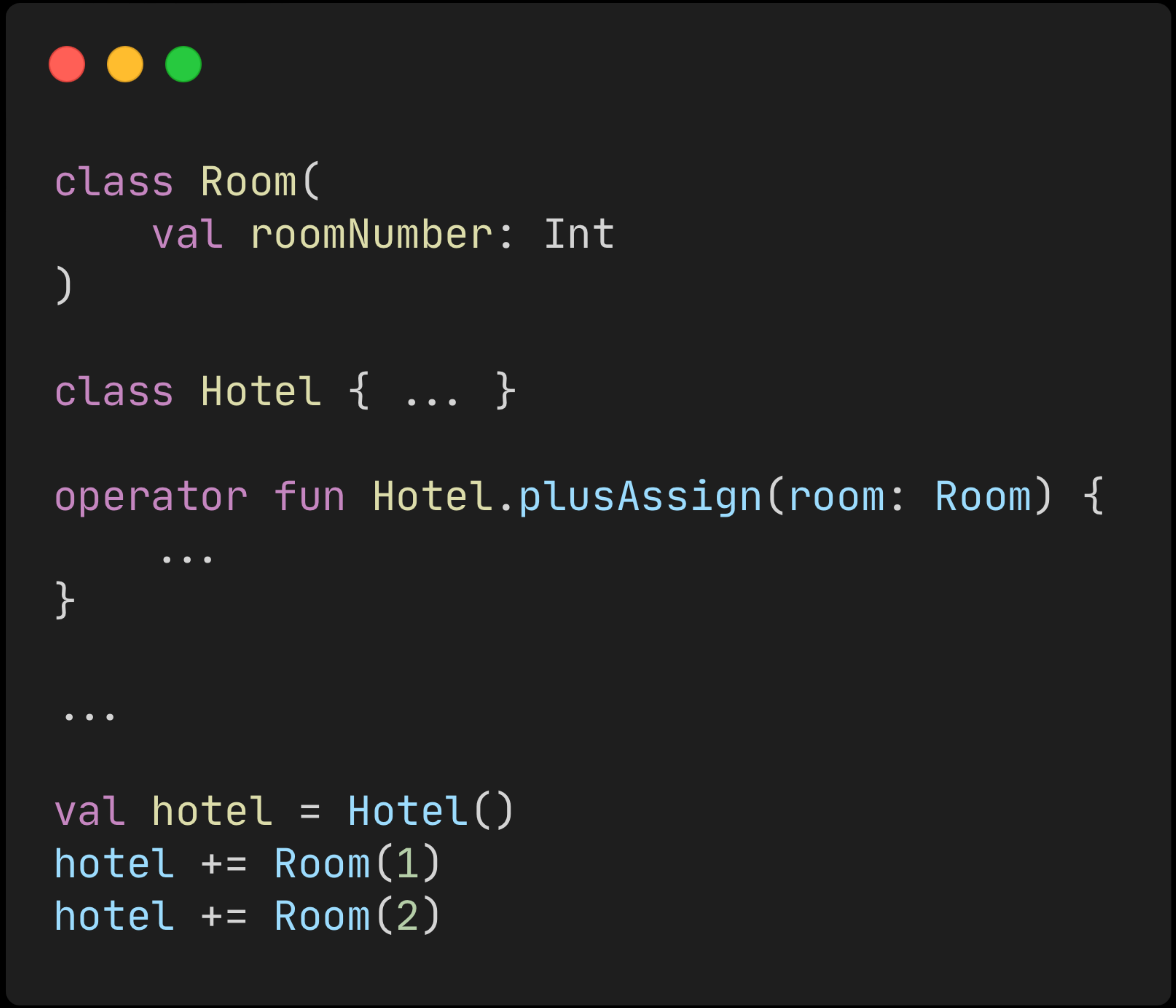


```
class Room(
    val roomNumber: Int
)

class Hotel {
    operator fun plusAssign(room: Room) {
        ...
    }
}

...

val hotel = Hotel()
hotel += Room(1)
hotel += Room(2)
```

#3 Operator overloading



```
class Room(
    val roomNumber: Int
)

class Hotel {
    operator fun plusAssign(room: Room) {
        ...
    }
}

...

val hotel = Hotel()
hotel += Room(1)
hotel += Room(2)
```



```
class Room(
    val roomNumber: Int
)

class Hotel { ... }

operator fun Hotel.plusAssign(room: Room) {
    ...
}

...

val hotel = Hotel()
hotel += Room(1)
hotel += Room(2)
```

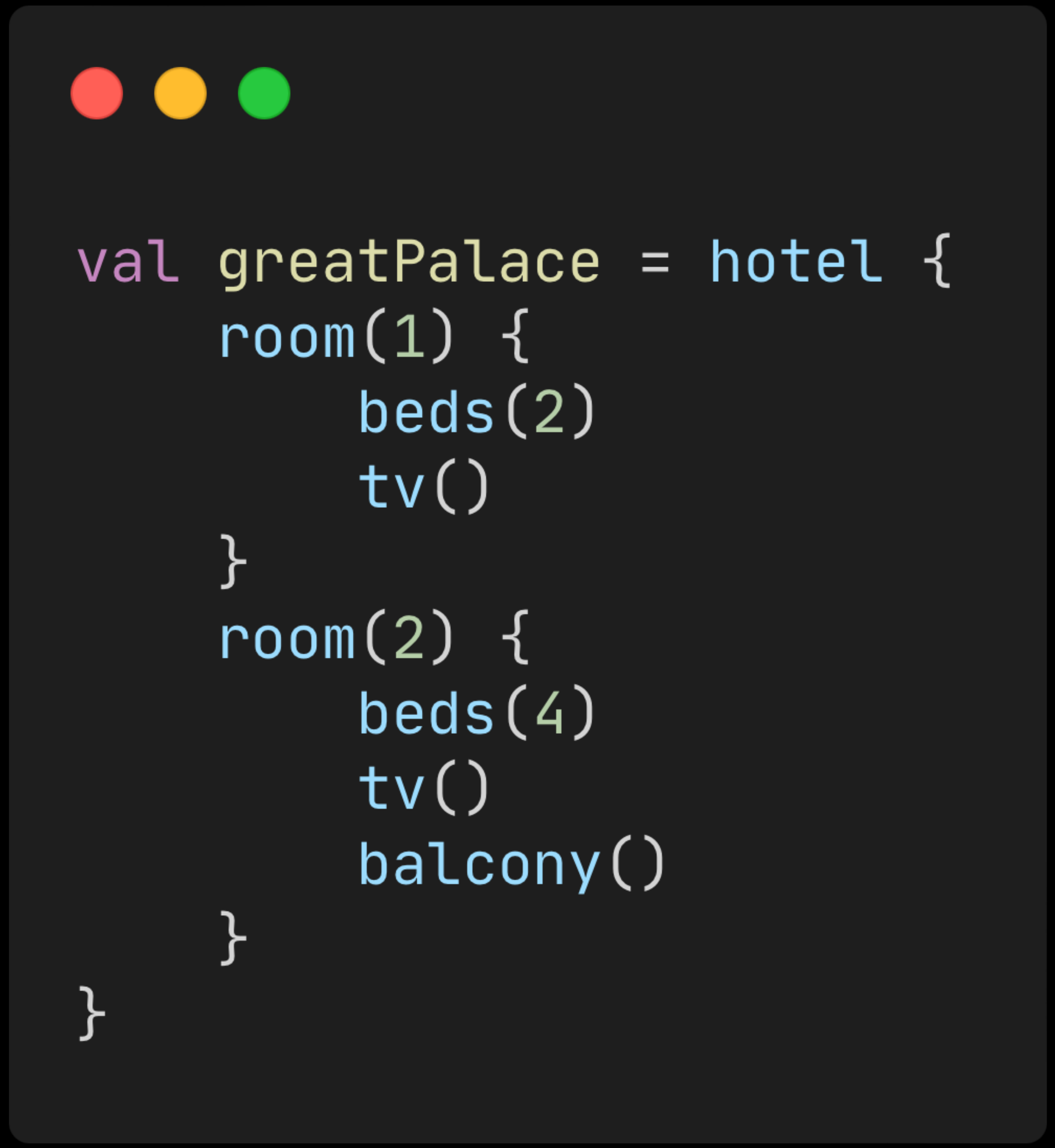
#4 @DslMarker

```kotlin
class Room(
    val roomNumber: Int
) {
    fun beds(amount: Int) { ... }
    fun tv() { ... }
    fun balcony() { ... }
}

class Hotel {
    private fun addRoom(room: Room) { ... }

    fun room(number: Int, init: Room.() -> Unit) {
        val room = Room(number)
        room.init()
        addRoom(room)
    }
}
```

```kotlin
val greatPalace = hotel {
    room(1) {
        beds(2)
        tv()
    }
    room(2) {
        beds(4)
        tv()
        balcony()
    }
}
```

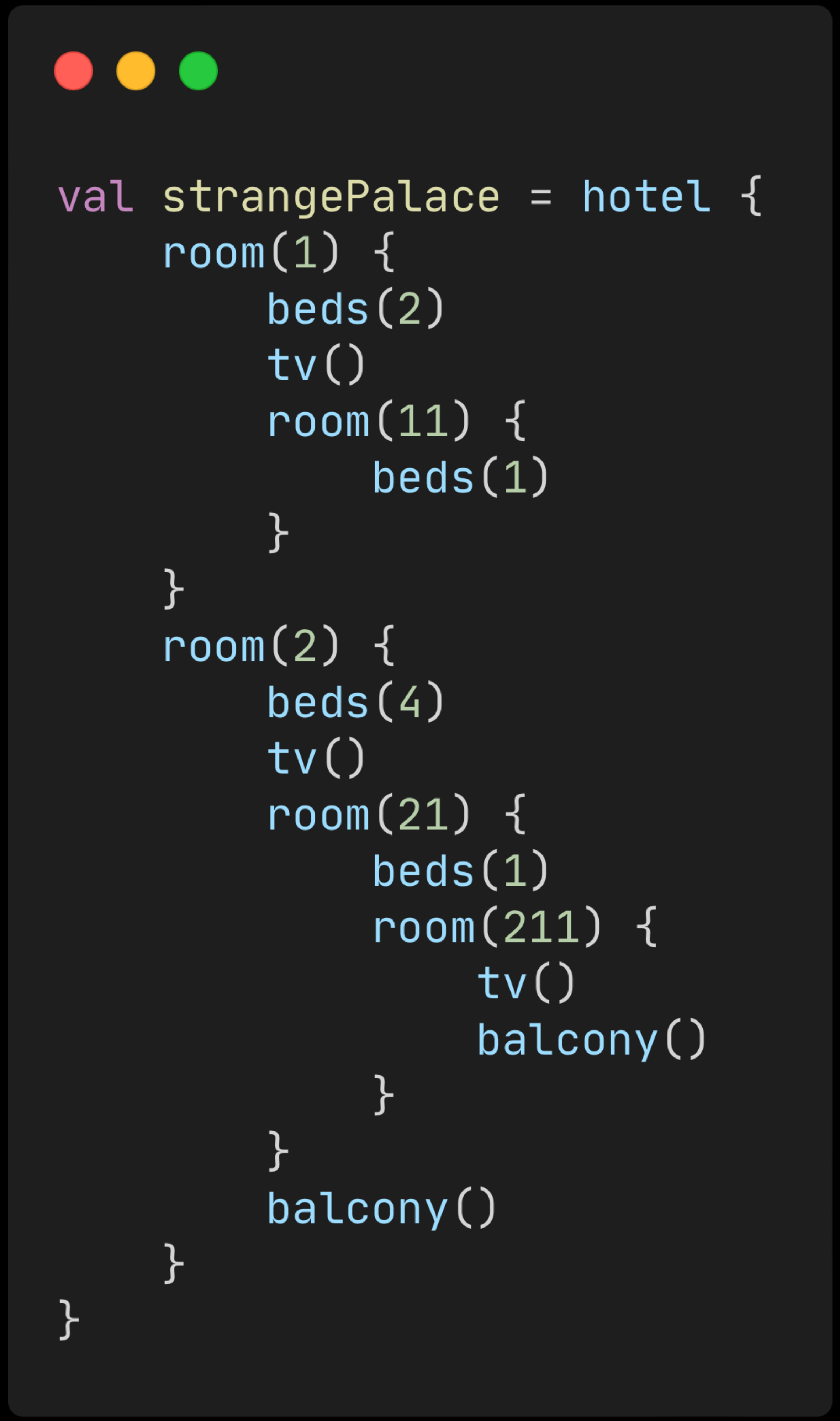
```kotlin
fun hotel(init: Hotel.() -> Unit): Hotel {
    val hotel = Hotel()
    hotel.init()
    return hotel
}
```

#4 @DslMarker



```
val greatPalace = hotel {
    room(1) {
        beds(2)
        tv()
    }
    room(2) {
        beds(4)
        tv()
        balcony()
    }
}
```

$\longrightarrow$



```
val strangePalace = hotel {
    room(1) {
        beds(2)
        tv()
        room(11) {
            beds(1)
        }
    }
    room(2) {
        beds(4)
        tv()
        room(21) {
            beds(1)
            room(211) {
                tv()
                balcony()
            }
        }
        balcony()
    }
}
```

#4 @DslMarker

```kotlin
@DslMarker
annotation class HotelDsl
```

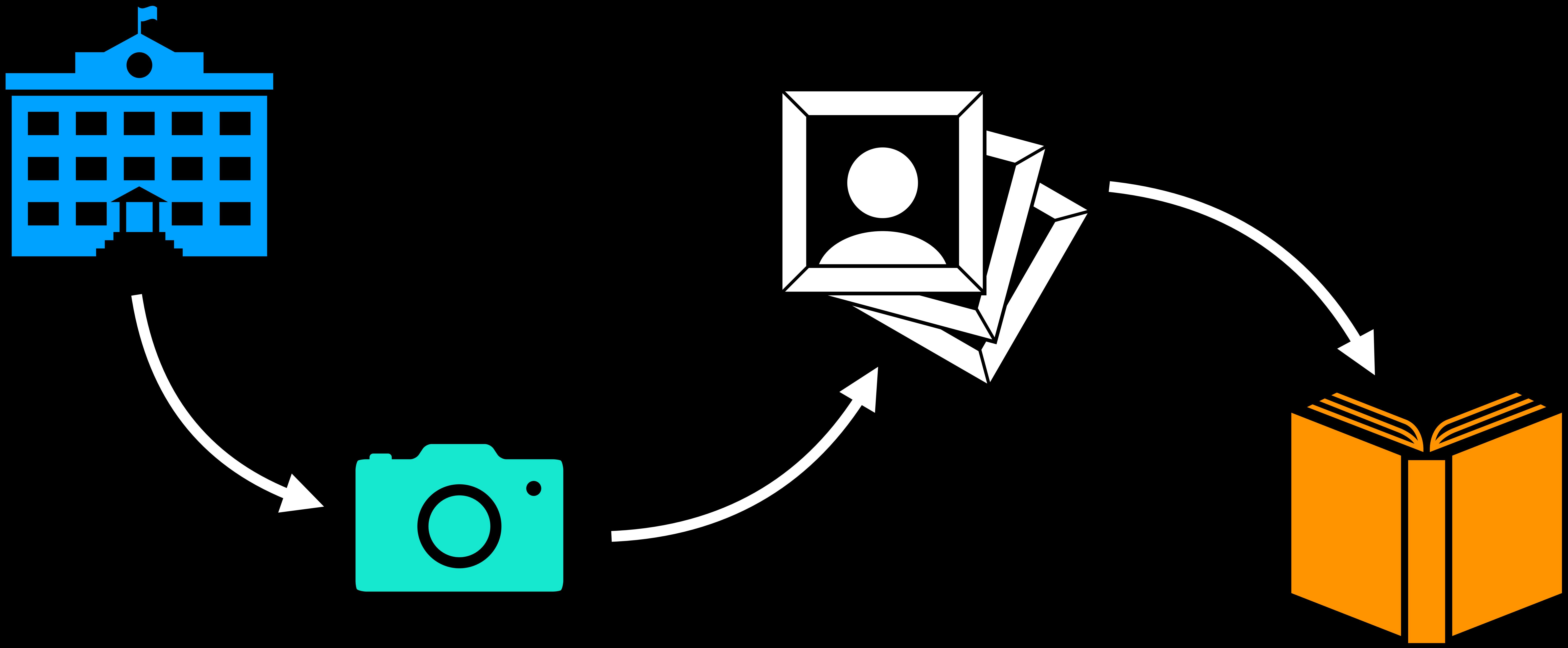
```kotlin
@HotelDsl
class Room(...) { ... }

@HotelDsl
class Hotel(...) { ... }
```

```kotlin
val strangePalace = hotel {
    room(1) {
        beds(2)
        tv()

        room(11) {
            beds(1)
        }
    }
}
```

Compiler Error

```kotlin
val strangePalace = hotel {
    room(1) {
        beds(2)
        tv()

        this@hotel.room(11) {
            beds(1)
        }
    }
}
```

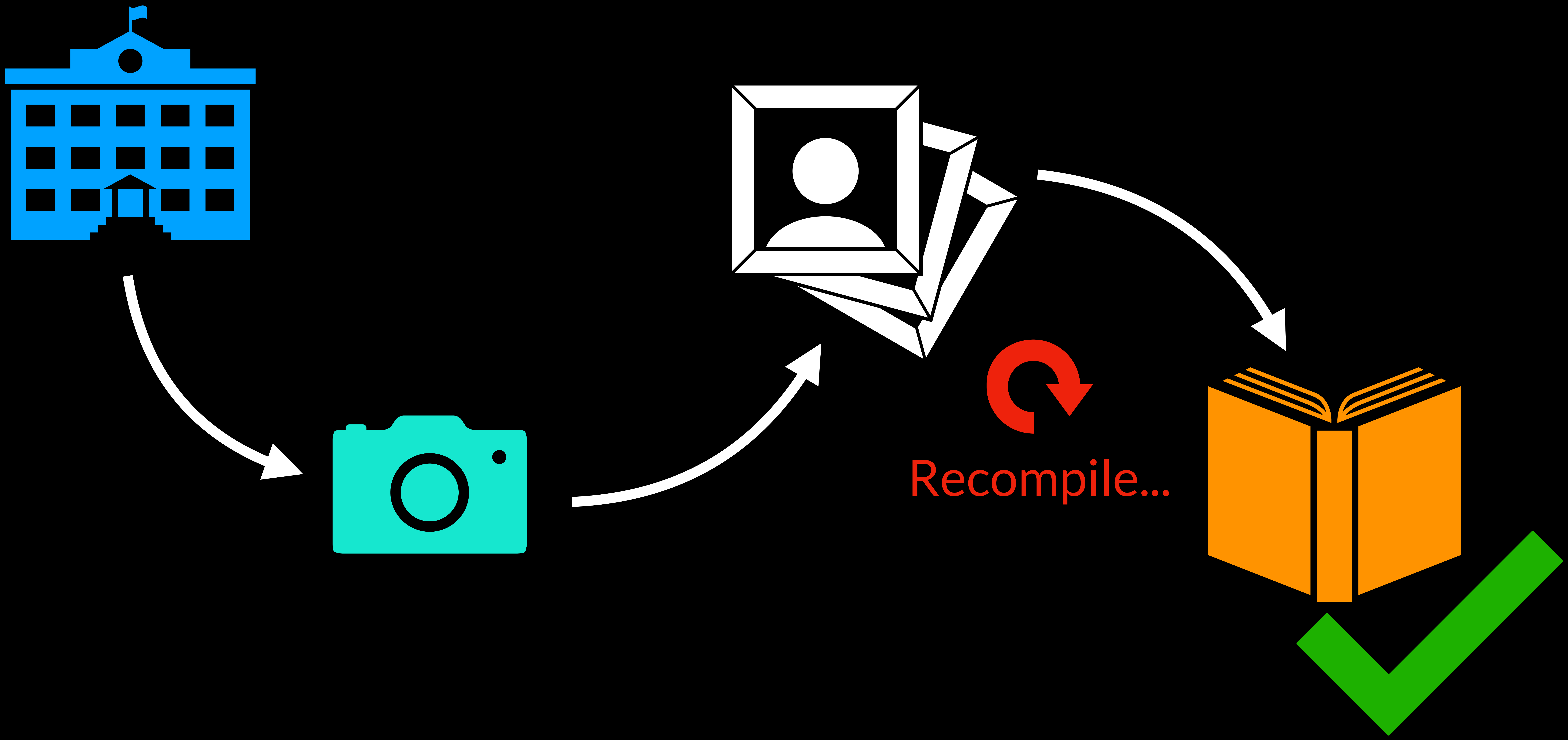
Photo Books



Live Coding

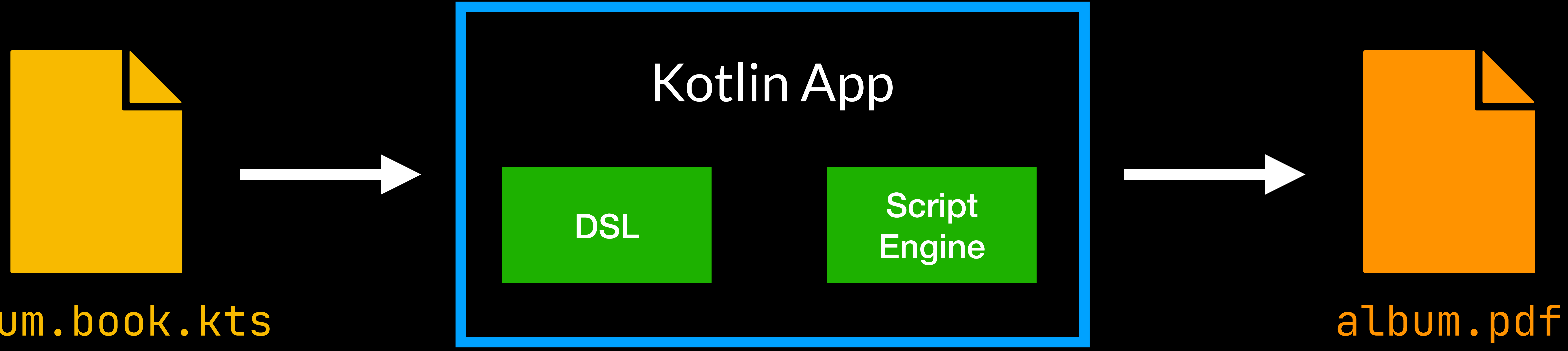
PhotoBook in Kotlin

Photo Books



Experimental

Kotlin Scripting

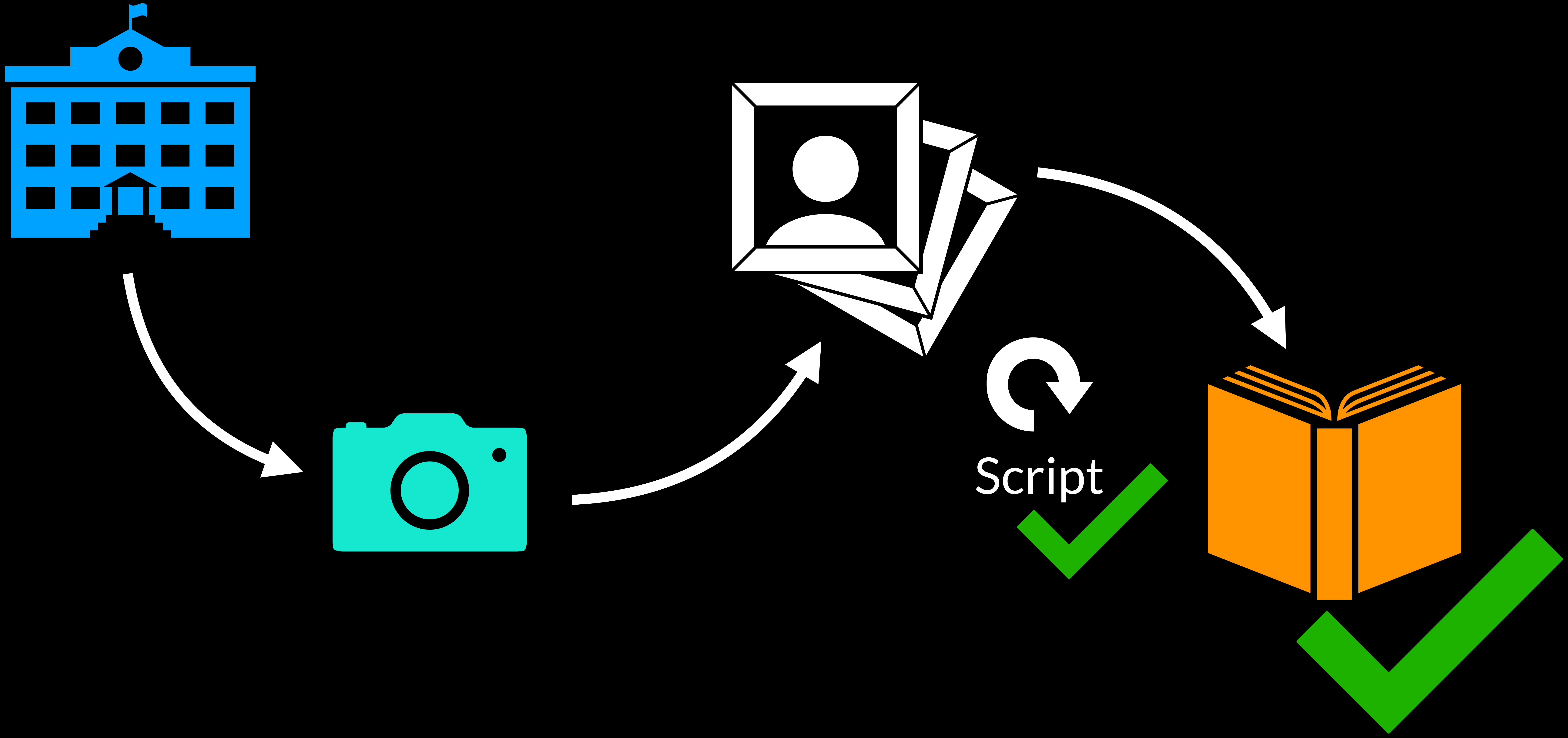


Gradle + Kotlin

Live Coding

PhotoBook Script

Photo Books



Algorithmic Photo Books

Kotlin DSLs demystified

Stefan Schöberl
schoeberl.dev
stefanschoeberl

Images

- Kotlin: https://kotlinlang.org/docs/kotlin-brand-assets.html
- Ktor: https://github.com/ktorio/ktor/blob/main/.github/images/ktor-logo-for-dark.svg
- Gradle: https://gradle.com/brand/
- Code formatting: https://carbon.now.sh