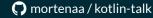
KOTLIN EN INTRODUKSJON

MORTEN NYGAARD ÅSNES

HISTORIE

- Statisk typet programmeringsspråk
 - JVM
 - JavaScript
 - Native (LLVM)
- Multi Paradigm
- JetBrains



FUNKSJONER

```
fun main(args: Array<String>) {
    println("Hello, World!")
}

fun max(a: Int, b: Int): Int {
    return if (a > b) a else b
}

fun sum(a: Int, b: Int) = a + b
```



VARIABLER

IMMUTABLE

```
val a: Int = 1
val b = 2
val c: Int
c = 3
```

MUTABLE

```
var sum: Int = 0
sum = a + b

var s = "$sum = ${a + b}"
```

NULLABLE

```
val x: String = null // Does not compile

val x: String? = null // Ok

val y: String? = if (Random().nextBoolean()) "foo" else null

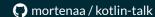
println(y.length) // Unsafe, does not compile

println(y!!.length) // Unsafe, compiles
```



NULLABLE

```
val y: String? = if (Random().nextBoolean()) "foo" else null
if (y != null) {
   println(y.length) // safe
y.let {
                  // safe
    println(it)
val l = if (y != null) y.length else -1
val m = y?.length ?: -1
```



SMART CAST

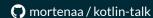
```
if (obj is Person) {
    print(obj.name)
}

if (x is Person && x.age > 20) return

val x = when (something) {
    is String -> something.length
    is Int -> something
    is List<*> -> something.size
    else -> -1
}
```

CLASSES

```
class Event { }
class Message
class Student {
    private val name: String
    constructor(name: String) {
        this.name = name
class Person(val firstName: String)
class Thing(val name: String, var age: Int, val type: Type)
```



DATA CLASSES

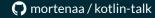


LAMBDA

```
fun doIt(value: Int, body: (Int) -> Int): Int {
    return body(value)
}

doIt(10, { v -> 16 * v})
    doIt(20, { it * 8 })

val sumIt = { x: Int, y: Int -> x + y }
```



LAMBDA



DSL

```
html {
    head {
        title { +"The Title" }
    body {
        h1 { +"Some Header" }
        p { +"Paragraph" }
            +"This is some"
            b { +"mixed" }
            +"text. For more see the"
        p { +"some text" }
```

