

Découvrir





Igor Laborie
Expert Web & Java

 [@ilaborie](https://twitter.com/ilaborie)

 igor@monkeypatch.io



- Écrire du code plus sûr
- Faciliter la maintenance
- Écrire et Tester plus rapidement
- Résoudre de nouveaux problèmes
- ...

- Expressif et pragmatique
- *null-safety* (éviter les NPE), statiquement typé
- Abordable, si on vient de Java
- Inspiré par Java, Scala, C#, Groovy, ...
- Cross-platform



JVM et Android



JavaScript



Native avec
LLVM

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

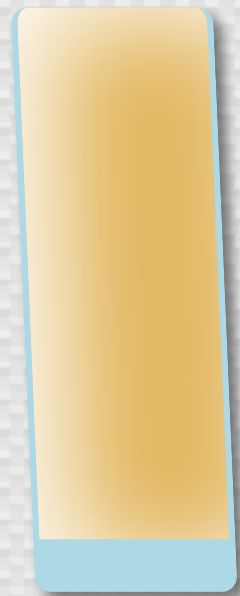


Utilisez `Alt + Shift + (Cmd|Ctrl) + K` pour convertir une classe Java en Kotlin

Ou copiez du code Java dans un fichier Kotlin



WATER POURING PROBLEM



8 / 8



0 / 6



1 / 4

Fill



1 / 4

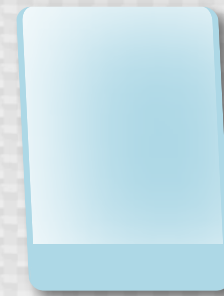


4 / 4

Empty

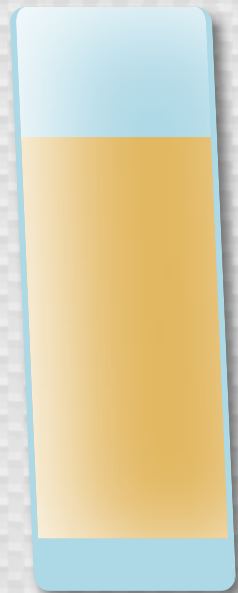


3 / 4



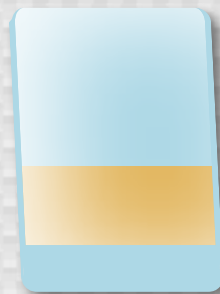
0 / 4

Pour



6 / 8

into

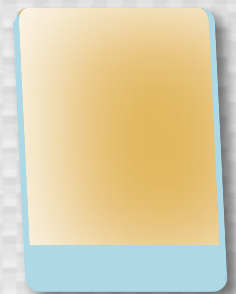


2 / 6

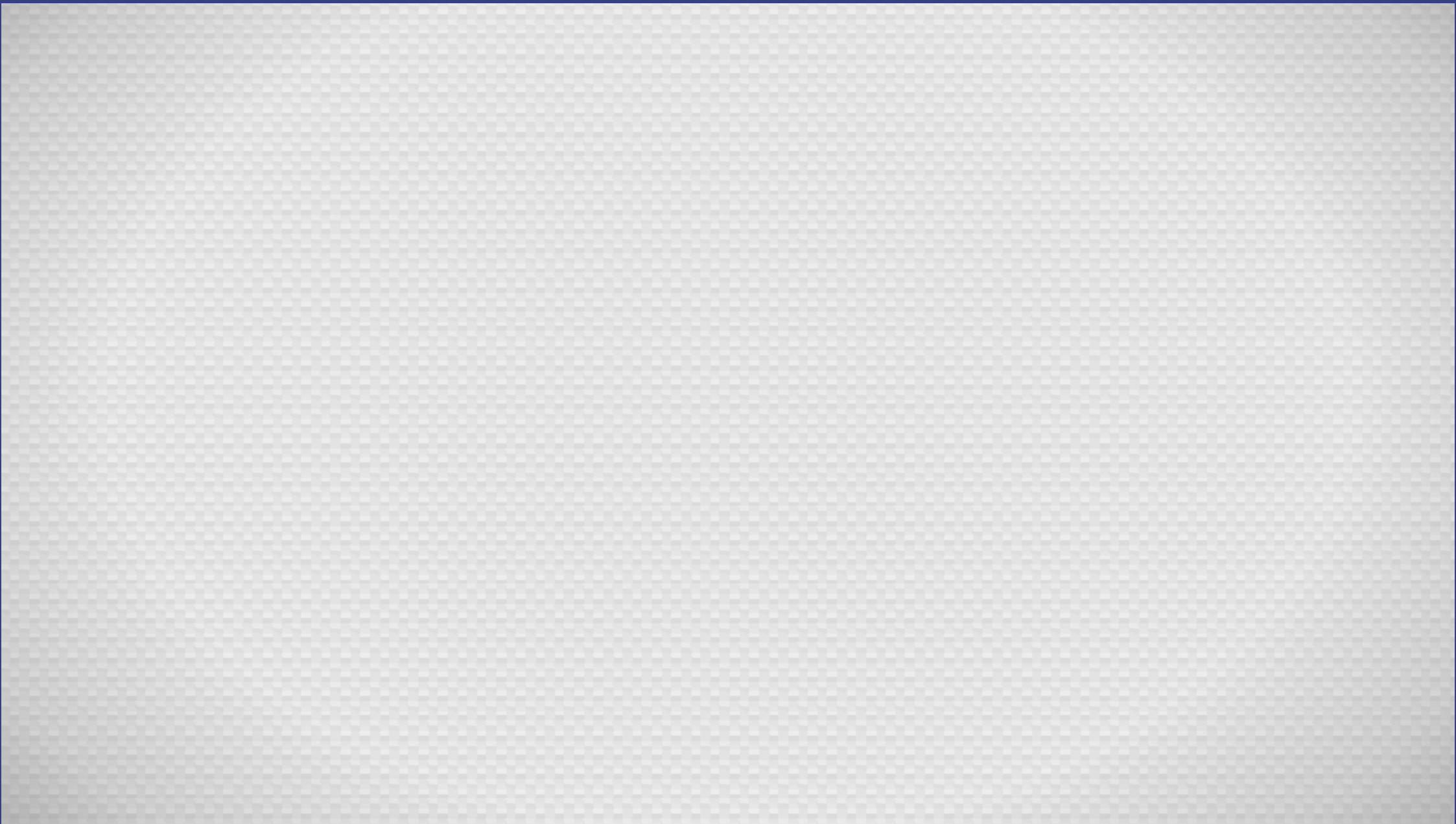


2 / 8

,



6 / 6



LIVE CODING

```
data class Glass(val capacity: Int, val current: Int = 0) {
  init {
    require(capacity > 0)
    require(current in 0..capacity)
  }

  val isEmpty: Boolean = (current == 0)
  val isFull: Boolean = (current == capacity)
  val remainingVolume: Int by lazy { capacity - current }
  fun empty(): Glass = copy(current = 0)
  fun fill(): Glass = copy(current = capacity)
  operator fun plus(value: Int): Glass =
    copy(current = (current + value).coerceAtMost(capacity))
  operator fun minus(value: Int): Glass =
    copy(current = (current - value).coerceAtLeast(0))
  override fun toString() = "$current/$capacity"
}

typealias State = List<Glass>
```

```
sealed class Move

data class Empty(val index: Int) : Move()

data class Fill(val index: Int) : Move()

data class Pour(val from: Int, val to: Int) : Move() {
    init {
        require(from != to)
    }
}
```



```
typealias StateWithHistory = Pair<State, List<Move>>

solve(from: State, to: State): List<Move> {
  @tailrec fun solveAux(states: List<StateWithHistory>, visitedStates: Set<State>): List<Move> {
    val solution: StateWithHistory? = states.find { (state, _) → state == to }
    if (solution != null) { return solution.second }

    val next = states
      .flatMap { (state, history) →
        state.availableMoves()
          .map { move → state.move(move) to (history + move) }
      }
      .filterNot { (state, _) → visitedStates.contains(state) }
    val nextVisited = visitedStates + next.map { it.first }
    return solveAux(next, nextVisited)
  }

  return solveAux(listOf(from to emptyList()), setOf(from))
}
```

```
fun State.move(move: Move): State =
  mapIndexed { index, glass →
    when (move) {
      is Empty → if (index == move.index) glass.empty() else glass
      is Fill  → if (index == move.index) glass.fill() else glass
      is Pour  → when (index) {
        move.from → glass - get(move.to).remainingVolume
        move.to    → glass + get(move.from).current
        else       → glass
      }
    }
  }
}
```

KOTLIN DÈS MAINTENANT

- Faible overhead
- Support officiel par Google
-  [Using Project Kotlin for Android](#)
-  [Kotlin Guide](#)
-  [Kotlin extensions for Android](#)

- Supporter officiellement depuis  Spring 5,  Spring Boot 2
-  SparkJava,  javalin
-  Vert.x
-  KTor
- ...

Web








- Partager du code commun
-  [Use Kotlin with npm, webpack and react](#)

Natif

- Faire des applications sans JVM
- Partager du code avec iOS
- WebAssembly

CONCLUSION

- Code plus sûr, plus simple
- Interoperable avec Java
- Outillage
- Ecosystème et communauté
- Déjà mature
- Simple à apprendre

-  [Référence](#)
-  <https://kotlin.link/>
-  [Blog](#)
-  [Forum](#),  [Slack](#)
-  [Koans](#)
-  [Kotlin by example](#)

Question(s) ?