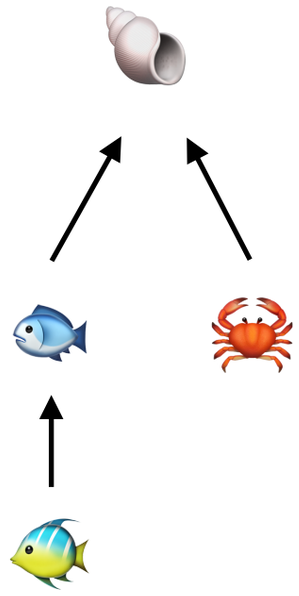


Kotlin

Macha DA COSTA

Kotlin : Délégation, Générique

- 📎 Concept de délégation
- 📎 Délégation de fonctions, propriétés
- 📎 Concept de Generics
- 📎 Covariance, Contravariance, Invariance

Kotlin : Délégation, Générique

Concept de délégation

- Principe : patron de conception Delegation
- Mécanisme : une classe délègue l'implémentation d'une fonction à une classe déléguée.

Kotlin : Délégation, Générique

Délégation de fonction

```
interface Base {
    fun print()
}

class BaseImpl(val x: Int) : Base {
    override fun print() { print(x) }
}

class Derived(b: Base) : Base by b

fun main() {
    val b = BaseImpl(10)
    Derived(b).print()
}
```

 Référence [18] : https://kotlinlang.org/docs/delegation.html

Kotlin : Délégation, Générique

Délégation de propriété

```
class Example {
    var p: String by Delegate()
}
```

ℹ `Delegate` fournit `setValue()` / `getValue()`

 Référence [26] : kotlinlang.org: delegated-properties.html

Kotlin : Délégation, Générique

Délégation de propriété standard

```
val mStringArray: Array<String> by lazy
{ resources.getStringArray(R.array.motivation_quote) }
```

ℹ Initialisation quand la variable est prête, c’est une constante !

 Référence [15] : https://antonioleiva.com/property-delegat/

Kotlin : Délégation, Générique

Concept de générique

- Principe : la classe, fonction peut convenir à différents type
- Mécanisme : une classe peut avoir des paramètres de type

Kotlin : Délégation, Générique

Concept de générique

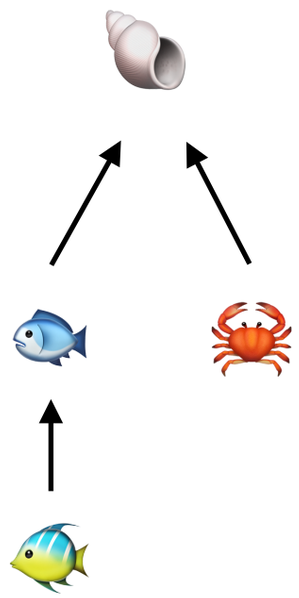
```
class Box<T>(t: T) {
    var value = t
}

fun main() {
    val box = Box(1) // 1 has type Int,
    //so the compiler figures out that it is Box<Int>
    print(box.value)
}
```


Référence [18] : https://kotlinlang.org/docs/generics.html

Kotlin : Délégation, Générique

Covariance, Contravariance, Invariance

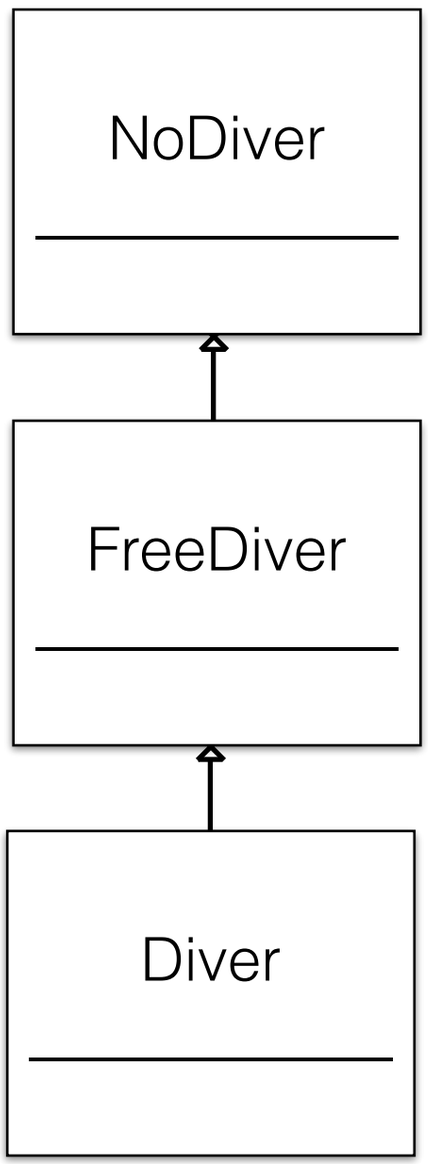


Kotlin : Délégation, Générique

Covariance, Contravariance, Invariance

Contravariance:

```
MountainPeople<in T>
goMountain
(mountainPeople:
MountainPeople<FreeDiver>)
```



Invariance:

```
ChillPeople<T>
goMeditate
(chillPeople:
ChillPeople<FreeDiver>)
```

Covariance:

```
DiverPeople<out T>
goDeeper
(diverPeople:
DiverPeople<FreeDiver>)
```

Kotlin : Délégation, Générique

Covariance

- Principe : classe générique où le sous type est préservé

- Mécanisme :
  - passer des valeurs comme argument de fonction même si c’est pas exactement le même type
  - un seul sens, serviette et parasol sont des sous type de matériel de plage

Kotlin : Délégation, Générique

Covariance

```
public interface List<out E> : Collection<E>
```

Kotlin : Délégation, Générique

Contravariance

- Principe : reflexion de la covariance

 Référence [27] : medium.com: kotlin co-contra-in-variance

Kotlin : Délégation, Générique

Contravariance

```
interface Comparator<in T>
```

Kotlin : Délégation, Générique

Invariance

- Principe : la classe, fonction peut convenir à un type T
- Mécanisme : cela convient seulement au type T

Kotlin : Délégation, Générique

</> by

</> T: in out where

THE EXPERT
AT ANYTHING
WAS ONCE
A BEGINNER.

_ Helen Hayes

Références

Kotlin for Android

- TRY Kotlin
- Kotlin Workshop on Github: Slides and Questions
- https://antonioleiva.com/free-kotlin-android-course/
- ChillCoding.com : Introduction à Kotlin
- ChillCoding.com : Configurer Kotlin dans un projet Android Studio

Library

- ChillCoding.com : Utiliser des bibliothèques graphiques Kotlin dans un projet Android

Fonction d’extension

- Odelia Technologies : Les fonctions d'extension de Kotlin

Kotlin in videos

- Jake Wharton and Kotlin (DEC 2015)
- Tue Dao & Christina Lee on The Road to Kotlintown (KotlinConf 2017)
- Introduction to Kotlin Google I/O '17