





EN PASANT PAR JAVA 15 (SEALED), JAVA 14 (RECORD) ET JAVA 17 (EXHAUSTIVE)

**LINK TO THE PATH**

```
sealed interface Direction {  
    record North() implements Direction {}  
    record East() implements Direction {}  
    record South() implements Direction {}  
    record West() implements Direction {}  
}  
  
public static final String label(Direction d) {  
    return switch(d){  
        case Direction.North n -> "north";  
        case Direction.East e -> "east";  
        case Direction.South s -> "south";  
        case Direction.West w -> "west";  
    }  
}
```

Enfin des interfaces scellées + record permettent de décrire correctement un type « OU »

Finalement JEP 406 <http://openjdk.java.net/jeps/8213076> Java 17 (Septembre 2021) apporte le « switch/case » pour l'exhaustivité d'un pattern matching, ainsi que c'est un nouveau « switch/case » expressif (parce que l'instruction switch java est quand même bien pourri)

Modèle similaire à Kotlin avec 10 ans de retard ou Scala avec 15 ans de retard ...

... mais incomplet (pas de support des primitives float, boolean, double), pas de pattern de déconstruction dans le case (prévu en Java 18)

... Vous n'êtes pas prêt de voir ça en production

Mettre en prod un projet JAVA < 18 est une faute professionnelle 🤦🏻

MAGNET 2-DUALS - TRANSLES

# LINK TO THE PATH

EN PASSANT PAR JAVA 15 (SEALED), JAVA 14 (RECORD) ET JAVA 17 (EXHAUSTIVITÉ)

```
sealed interface Direction {  
    record North() implements Direction {}  
    record East() implements Direction {}  
    record South() implements Direction {}  
    record West() implements Direction {}  
}  
public static final String label(Direction d) {  
    return switch(d){  
        case Direction.North n -> "north";  
        case Direction.East e -> "east";  
        case Direction.South s -> "south";  
        case Direction.West w -> "west";  
    }  
}
```

Enfin des interfaces scellées + record permettent de décrire correctement un type « OU »

Finalement JEP 406 <http://openjdk.java.net/jeps/8213076> Java 17 (Septembre 2021) apporte le « switch/case » pour l'exhaustivité d'un pattern matching, ainsi que c'est un nouveau « switch/case » expressif (parce que l'instruction switch java est quand même bien pourri)

Modèle similaire à Kotlin avec 10 ans de retard ou Scala avec 15 ans de retard ...

... mais incomplet (pas de support des primitives float, boolean, double), pas de pattern de déconstruction dans le case (prévu en Java 18)

... Vous n'êtes pas prêt de voir ça en production

Mettre en prod un projet JAVA < 18 est une faute professionnelle 🤔



# LA MINUTE HINOX

... VOUS POUVEZ DÉJÀ UTILISER

Java 20 a.k.a Kotlin

Java 30 a.k.a Scala

Java 100 a.k.a OCaml

