



# Kotlin (Introduction)

RONALD JEAN-JULIEN



# Kotlin

- ▶ Un langage de programmation statiquement typé exécuté dans une machine virtuelle Java
- ▶ Développé par JetBrains
- ▶ Peut interagir avec du code Java ou des bibliothèques Java existantes
- ▶ Site de référence: [kotlinlang.org](https://kotlinlang.org)

# Premier exemple avec Kotlin

- ▶ REPL (Read-Evaluate-Print-Loop) disponible pour des tests rapides

- ▶ `//Java`

- ▶ `System.out.print("Hello");`

- ▶ `System.out.println("Hello GG")`

- ▶ `//Kotlin`

- ▶ `print("Hello")`

- ▶ `println("Hello GG")`

# Les variables

- ▶ //Java

- ▶ String firstName = "Gérald"
- ▶ String lastName = "Godin"
- ▶ final String nomComplet = firstName + lastName //Constante

- ▶ //Kotlin

- ▶ var firstName = "Gérard" //var : la variable est modifiable
- ▶ var lastName: String = "Godin" //Le type est optionnel en Kotlin comme en JavaScript)
- ▶ val nomComplet = "\$firstName \$lastName" //val : constante
- ▶ Les trois guillemets vous permettent d'interpoler une chaine de caractères.



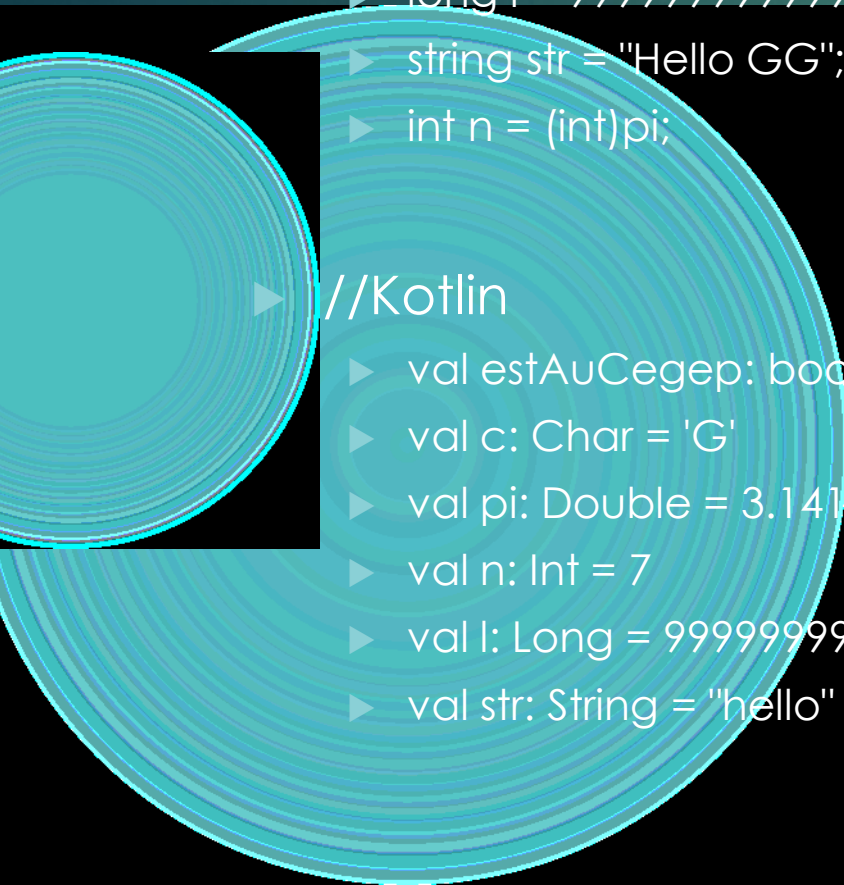
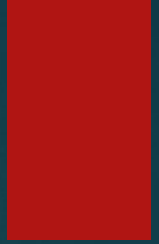
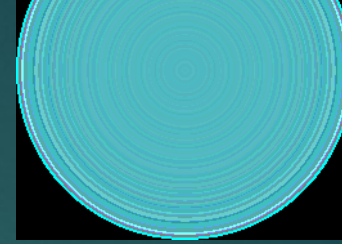
# Les types

## ▶ //Java

- ▶ `boolean estAuCegep = true;`
- ▶ `char c = 'G';`
- ▶ `double pi = 3.1415;`
- ▶ `int n = 7;`
- ▶ `long l = 9999999999999999;`
- ▶ `String str = "Hello GG";`
- ▶ `int n = (int)pi;`

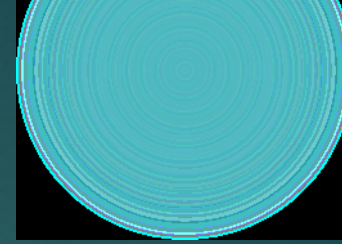
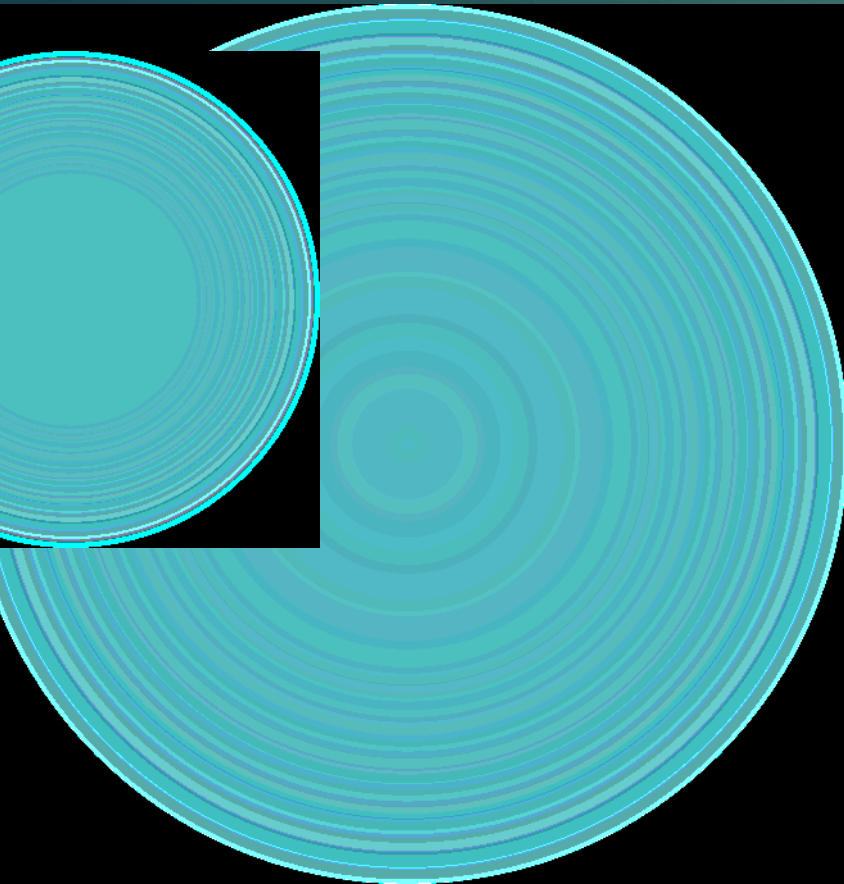
## ▶ //Kotlin

- ▶ `val estAuCegep: boolean = true`
- ▶ `val c: Char = 'G'`
- ▶ `val pi: Double = 3.1415`
- ▶ `val n: Int = 7`
- ▶ `val l: Long = 9999999999999999`
- ▶ `val str: String = "hello"`



# Conversion de types

- ▶ `val partieEntiereDePi: Int = pi.toInt()`
- ▶ `Val nEnDouble: Double = n.toDouble()`



# Chaines de caractères

▶ //Kotlin

▶ val firstName = "Gérald"

▶ val lastName = "Godin"

▶ val nomCompleet = "Ton nom complet est \$firstName \$lastName"

▶ Val ouSuisJe = ""Je suis  
au Cégep  
Gérald Godin""

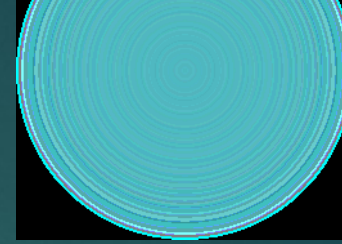
# Les boucles

## ▶ //Java

- ▶ for(int i = 1; i < 7; i++) {...}
- ▶ for(int i = 1; i < 7; i += 3){...}
- ▶ for(int i = 10; i >= 1; i--){...}
- ▶ for(String s : collection){...}
- ▶ for(String key: map.keySet()){...}

## ▶ //Kotlin

- ▶ repeat(7){...}
- ▶ for(i in 1..10){...}
- ▶ for(in in 1..10 step 3){...}
- ▶ for(s in collection){...}
- ▶ for((i, s) in collection.withIndex()){...}
- ▶ for((key, value) in map){...}





# Les listes

## ► //Java

- `int[] nums = {15, 25, 50};`
- `List<String> wordList = new ArrayList<>();`
- `wordList.add("Gérald");`
- `wordList.add("Godin");`

- `int num = nums[0];`
- `String word = wordList.get(1);`
- `If(wordList.contains("Hello")){...}`

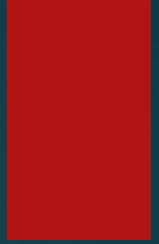
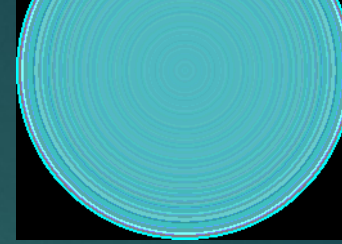
## ► //Kotlin

- `val nums: List<Int> listOf(15, 25, 50); //non modifiable (val)`
- `var wordList = mutableListOf("Gérald", "Godin") //modifiable`
- `wordList.add("Steven")`

- `val num = nums[0]`
- `val word = wordList[1]`
- `If("Hello" in wordList){...}`

# Fonctions

- ▶ //Java
- ▶ `int foo(String text, int n){`
  - ▶ `if(text == null){`
    - ▶ `Return 0;`
  - ▶ `}else{`
    - ▶ `Return text.length() / n;`
  - ▶ `}`
- ▶ `public int foo(String text){`
  - ▶ `foo(text, 2);`
  - ▶ `}`
- ▶ `int result1 = foo("Hello World", 3);`
- ▶ `int result2 = foo("Hello World");`



# Fonctions en Kotlin

```
//Ici, on peut pas passes des valeurs "null" à la fonction  
fun foo(s: String, n: Int = 2): Int{  
    return s.length() / n  
}  
...  
val result = foo("Hello GG")  
val result = foo(s: "Hello GG", n: 3) //paramètres de nom et n sont optionnels
```