Operators

|  |
| --- |
| Val fish =2;  fish.times(other:9);  fish.plus(other:2);  fish.div(other:5);  fish.minus(other:5); |

The Kotlin keep numbers as primitives, the same number types as your CPU uses.

*We can call methods on numbers like*

\* All the numerical types in Kotlin have a super type called numbers.

|  |
| --- |
| Using primitive ‘int’ as an object  1.toLong() |

\* We can use numbers as if they were objects, Kotlin supports object wrappers for numbers

|  |
| --- |
| Val boxed: Number =1  Boxed.toLong() |

\* **Boxing** is a term in Kotlin that refers to storing value in one variable of type number

|  |  |
| --- | --- |
| Changeable | Var |
| Unchangeable | Val |

Variables

There are two types of variables in Kotlin;

|  |
| --- |
| Val b: Byte = 1  Val i : int = b //Error: type mismatch |

Type is inferred automatically by Kotlin, even it becomes fixed at compile time, and you cannot change the type of a variable in Kotlin once its type has been determined

|  |
| --- |
| val object : Int = b.toInt() |

Casting

\* Kotlin supports underscores in numbers.

Nullability

Kotlin helps to avoid null pointer exceptions.

|  |
| --- |
| var variable: Int? = null |

When we declare a variable type explicitly by default, its value can’t be null. We use ? **Operator** to indicate that a variable can be null.

|  |
| --- |
| Var listofFish: List<String?> = ListOf(null,null) |

When we have complex data types such as a list, we can allow for the list to be null, but if it’s not null, its elements can’t be null or we can allow both list and elements to be null.

|  |
| --- |
| val l = b?.length **?:** 0 |

Elvis Operator

Strings

Strings are similar to strings in other languages

\* We can concatenate strings using **+ operator.**

|  |
| --- |
| “I have $numberOfFish fish and $numberOfPlants plants”  **Output:** *“I have 5 fish and 12 plants”* |

\* We can use ***string templating*** in order to build strings by combining them with values.

\* Kotlin has Boolean Datatype and operators == (value by comparison) != (not equal)

|  |
| --- |
| Val mix = arrayOf(“fish”,2) |

Ranges

Arrays

|  |
| --- |
| Val fish = arrayOf(“”,””,””) |

We can make array by

|  |
| --- |
| Val array = Array(size) {it \* 2}  Println(array.asList())  *{0,2,4,6,8}* |

A really cool feature of Kotlin is the ability to initialize them with dynamic code.

Loops