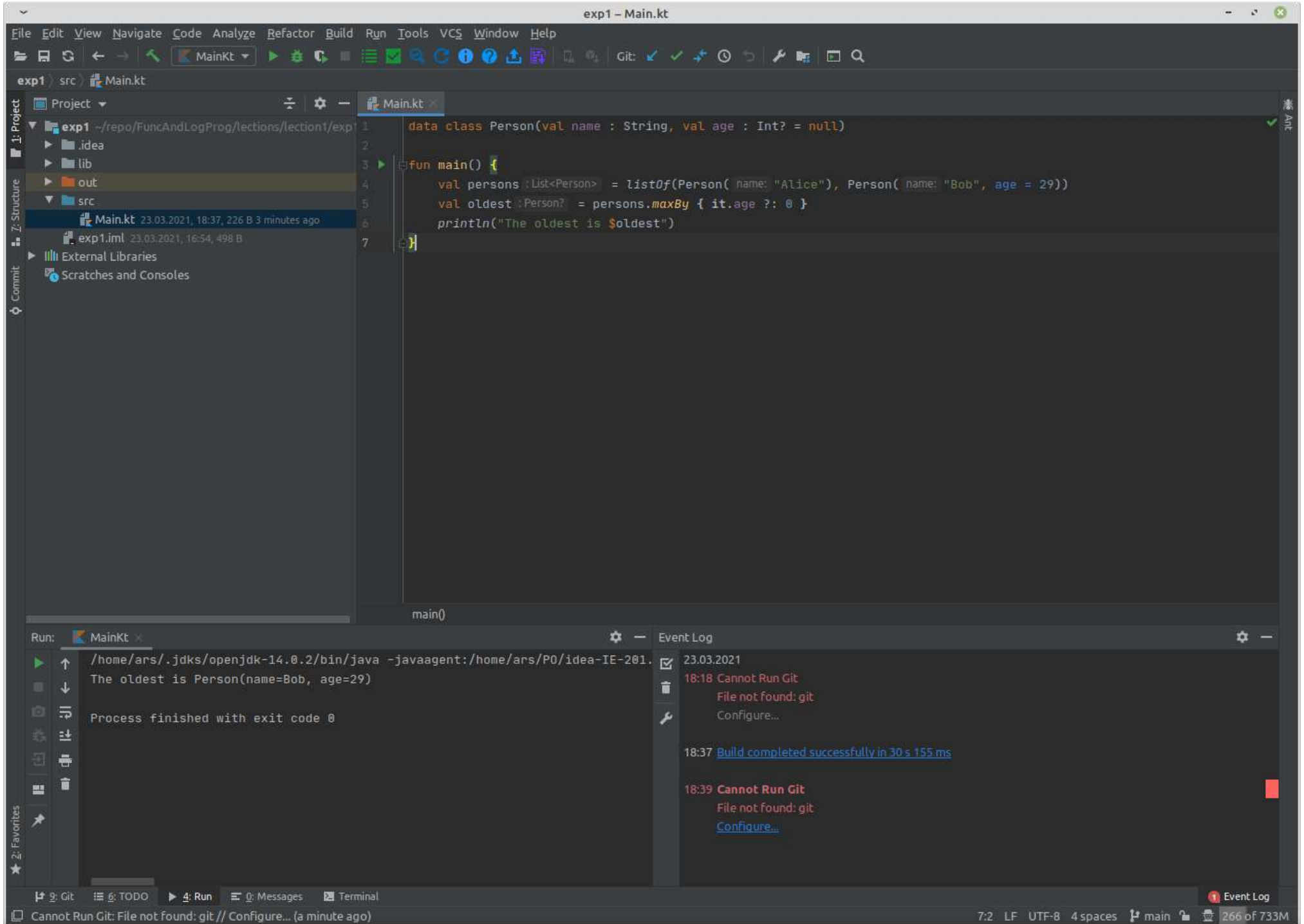
Лекция 8. Знакомство с Kotlin

Лекция 8. Знакомство с Kotlin

1. Язык
2. Философия и инструментарий
3. Основные элементы: переменные и функции.
4. Классы и свойства


exp1 – Main.kt
File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help
MainKt
Git:
exp1 〉 src 〉 Main.kt
Project
exp1 ~/repo/FuncAndLogProg/lections/lection1/exp1
.idea
lib
out
src
Main.kt 23.03.2021, 18:37, 226 B 3 minutes ago
exp1.iml 23.03.2021, 16:54, 498 B
External Libraries
Scratches and Consoles
Main.kt
data class Person(val name : String, val age : Int? = null)
fun main() {
val persons :List<Person> = listOf(Person(name: "Alice"), Person(name: "Bob", age = 29))
val oldest :Person? = persons.maxBy { it.age ?: 0 }
println("The oldest is $oldest")
}
main()
Run: MainKt
/home/ars/.jdks/openjdk-14.0.2/bin/java -javaagent:/home/ars/PO/idea-IE-201.
The oldest is Person(name=Bob, age=29)
Process finished with exit code 0
Event Log
23.03.2021
18:18 Cannot Run Git
File not found: git
Configure...
18:37 Build completed successfully in 30 s 155 ms
18:39 Cannot Run Git
File not found: git
Configure...
9: Git
6: TODO
4: Run
0: Messages
Terminal
Event Log
Cannot Run Git: File not found: git // Configure... (a minute ago)
7:2 LF UTF-8 4 spaces main
266 of 733M

```
data class Person(val name: String,
                  val age: Int? = null)

fun main(args: Array<String>) {
  val persons = listOf(Person("Alice"),
                       Person("Bob", age = 29))

  val oldest = persons.maxBy { it.age ?: 0 }
  println("The oldest is: $oldest")
}

// The oldest is: Person(name=Bon, age=29)
```

Класс «данных»

Тип, допускающий значение null (Int?); значение параметра по умолчанию

Функция верхнего уровня

Именованный аргумент

Лямбда-выражение; оператор «Элвис»

Строка-шаблон

Часть вывода автоматически сгенерирована методом toString

KOTLIN — компилируемый

KOTLIN — статически типизируемый

val x = 1

выведение типа (type inference)

- *Производительность* – вызов методов происходит быстрее, поскольку во время выполнения не нужно выяснять, какой метод должен быть вызван.
- *Надежность* – корректность программы проверяется компилятором, поэтому вероятность ошибок во время выполнения меньше.
- *Удобство сопровождения* – работать с незнакомым кодом проще, потому что сразу видно, с какими объектами код работает.
- *Поддержка инструментов* – статическая типизация позволяет увереннее выполнять рефакторинг, обеспечивает точное автодополнение кода и поддержку других возможностей IDE.

```
data class Person(val name : String, val age : Int? = null)
```

поддержка nullable типов

Int и Int?

KOTLIN — функционален!!!

- *Функции как полноценные объекты* – с функциями (элементами поведения) можно работать как со значениями. Их можно хранить в переменных, передавать в аргументах или возвращать из других функций.
- *Неизменяемость* – программные объекты никогда не изменяются, что гарантирует неизменность их состояния после создания.
- *Отсутствие побочных эффектов* – функции всегда возвращают один и тот же результат для тех же аргументов, не изменяют состояние других объектов и не взаимодействуют с окружающим миром.

- лаконичность — лямбда выражения

```
fun findAlice() = findPerson { it.name == "Alice" }
fun findBob() = findPerson { it.name == "Bob" }
```

Функция findPerson() описывает общую логику поиска

Блок кода в фигурных скобках задает свойства искомого элемента

- неизменяемость объектов:
безопасное многопоточное программирование,
чистые функции — удобное распраллеливание

- *функциональные типы,* позволяющие функциям принимать или возвращать другие функции;
- *лямбда-выражения,* упрощающие передачу фрагментов кода;
- *классы данных,* предоставляющие емкий синтаксис для создания неизменяемых объектов-значений;
- обширный набор средств в стандартной библиотеке для работы с объектами и коллекциями в функциональном стиле.

IntelliJ IDEA,
Android Studio
Eclipse

```
fun renderPersonList(persons: Collection<Person>) =
  createHTML().table {
    for (person in persons) {
      tr {
        td { +person.name }
        td { +person.age }
      }
    }
  }
}
```

Обычный цикл

Функции, выполняющие отображение в теги HTML

```
object CountryTable : IdTable() {
  val name = varchar("name", 250).uniqueIndex()
  val iso = varchar("iso", 2).uniqueIndex()
}

class Country(id: EntityID) : Entity(id) {
  var name: String by CountryTable.name
  var iso: String by CountryTable.iso
}

val russia = Country.find {
  CountryTable.iso.eq("ru")
}.first()

println(russia.name)
```

Описание таблицы в базе данных

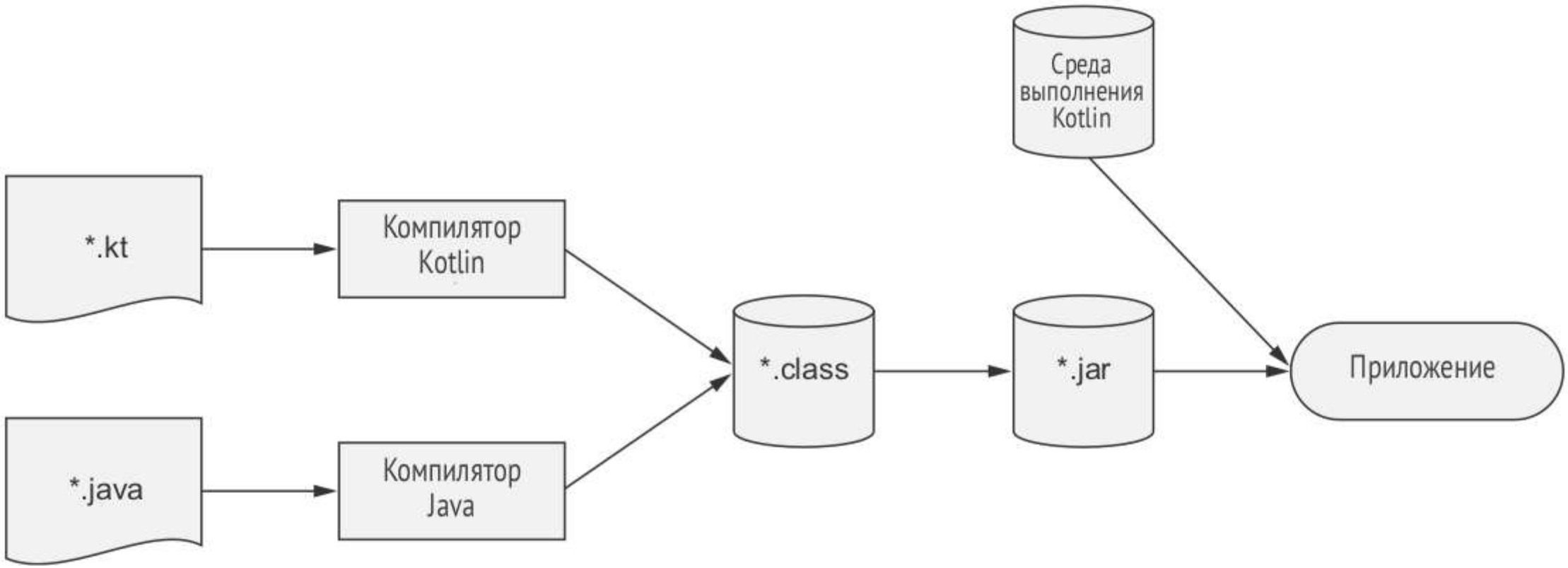
Определение класса, соответствующего сущности в базе данных

Вы можете выполнять запросы к базе данных на чистом Kotlin

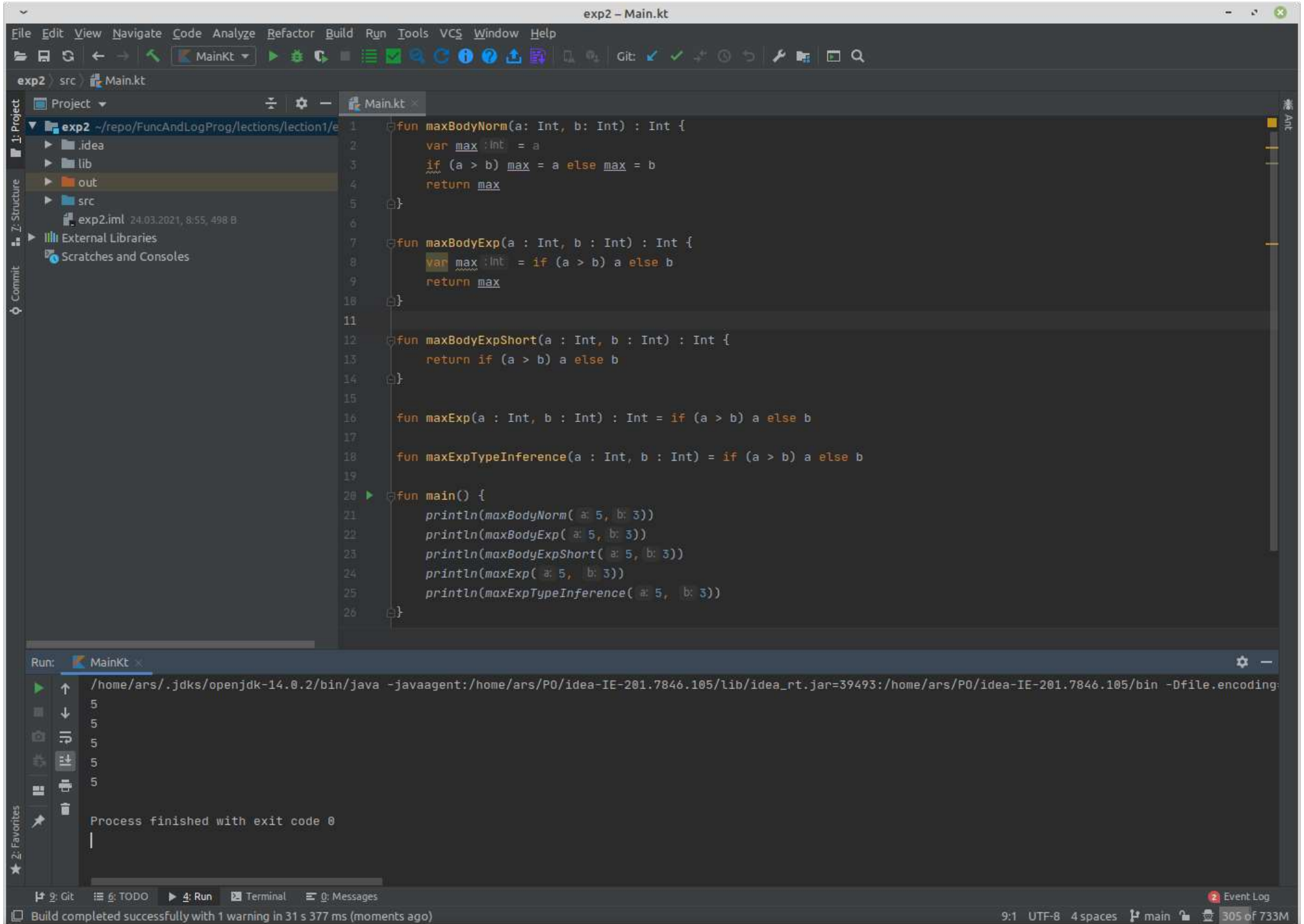
прагматичность
лаконичность
безопасность
совместимость

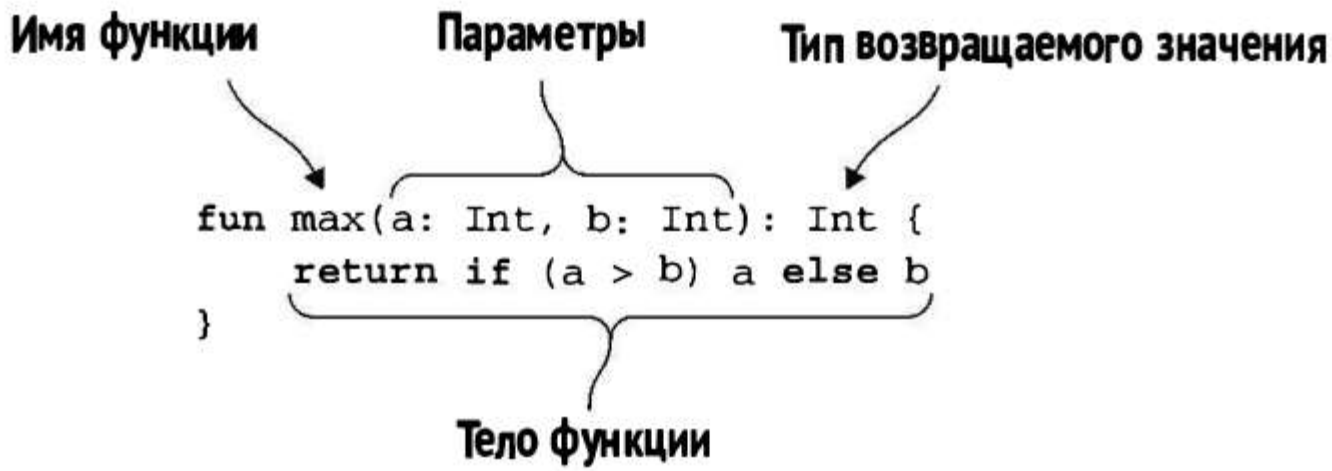
```
kotlinc <исходный файл или каталог> -include-runtime -d <имя jar-файла>
java -jar <имя jar-файла>
```

Read Eval Print
Loop


*.kt
*.java
Компилятор Kotlin
Компилятор Java
*.class
*.jar
Среда выполнения Kotlin
Приложение

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


exp2 – Main.kt
File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help
MainKt
Git:
exp2 src Main.kt
Project
exp2 ~/repo/FuncAndLogProg/lections/lection1/e
.idea
lib
out
src
exp2.iml 24.03.2021, 8:55, 498 B
External Libraries
Scratches and Consoles
Main.kt
fun maxBodyNorm(a: Int, b: Int) : Int {
var max :Int = a
if (a > b) max = a else max = b
return max
}
fun maxBodyExp(a : Int, b : Int) : Int {
var max :Int = if (a > b) a else b
return max
}
fun maxBodyExpShort(a : Int, b : Int) : Int {
return if (a > b) a else b
}
fun maxExp(a : Int, b : Int) : Int = if (a > b) a else b
fun maxExpTypeInference(a : Int, b : Int) = if (a > b) a else b
fun main() {
println(maxBodyNorm(a: 5, b: 3))
println(maxBodyExp(a: 5, b: 3))
println(maxBodyExpShort(a: 5, b: 3))
println(maxExp(a: 5, b: 3))
println(maxExpTypeInference(a: 5, b: 3))
}
Run: MainKt
/home/ars/.jdks/openjdk-14.0.2/bin/java -javaagent:/home/ars/PO/idea-IE-201.7846.105/lib/idea_rt.jar=39493:/home/ars/PO/idea-IE-201.7846.105/bin -Dfile.encoding
5
5
5
5
5
Process finished with exit code 0
9: Git 6: TODO 4: Run Terminal 0: Messages
Event Log
Build completed successfully with 1 warning in 31 s 377 ms (moments ago)
9:1 UTF-8 4 spaces main
305 of 733M


Имя функции
Параметры
Тип возвращаемого значения
fun max(a: Int, b: Int): Int {
return if (a > b) a else b
}
Тело функции

```
fun varAndValDeclaration() : Unit {
    val question = "The Ultimate Question of Life , the Universe , and Everything"
    val answer = 42 // typeInference
    val reaction : String = "AAAAAA, what????"
    val yearsToCompute = 7.5e6 // Double
    val anotherQuestion : String
    anotherQuestion = "So, what's the question?"
    question = "mmmmm" + yearsToCompute + "years to compute"

}
```

```
val languages = arrayListOf("Java")
languages.add("Kotlin")
```

Объявление неизменяемой ссылки

Изменение объекта, на который она указывает

```
var answer = 42
answer = "no answer"
```

Ошибка: несовпадение типов

```
fun main(args: Array<String>) {
  val name = if (args.size > 0) args[0] else "Kotlin"
  println("Hello, $name!")
}
```

Выведет «Hello, Kotlin!» или «Hello, Bob!», если передать аргумент со строкой «Bob»

```
fun main(args: Array<String>) {
  if (args.size > 0) {
    println("Hello, ${args[0]}!")
  }
}
```

Синтаксис ${} используется для подстановки первого элемента массива args

```
fun main(args: Array<String>) {
  println("Hello, ${if (args.size > 0) args[0] else "someone"}!")
}
```

```
class Person(val name : String, var age : Int)

fun main() {

    val vasya = Person( name: "Vasya",  age: 25)
    vasya.age += 1 // ВЫЗОВ СЕТТЕРА И ГЕТТЕРА
    vasya.name = "Petya" // попытка вызвать сеттер, но он отсутствует
```


class Person(val name : String, var age : Int) {
val canBeServed : Boolean
get () {
return age > 18
}
var mobile : String
get () {
return mobile
}
set (str : String) {
mobile = if (isMobile(str)) makeMobile(str) else ""
}
fun isMobile(str : String) : Boolean = true
fun makeMobile(str : String) : String = str
}