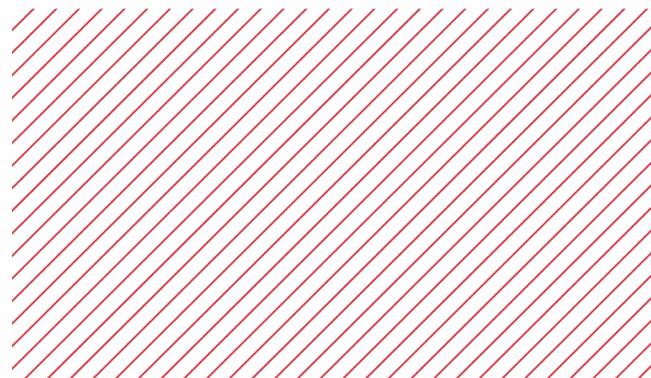


академия
больших
данных



Advanced Scala

Корольков Алексей



Обо мне



Более 5 лет работаю над Scala проектами



Прежде работал в большом банковском апи Тинькофф



Сейчас работаю в одном из проектов Тинькофф Инвестиций

Структура курса

1. Введение в Scala. Основные конструкции языка



2. ООП в Scala. Pattern matching. Функциональные конструкции. Adt

3. Библиотека коллекций в Scala



4. Асинхронные операции, обработка исключений, неявные параметры

5. Параметрический полиморфизм. Имплиситы



6. Основы функционального программирования. Часть 1

7. Основы функционального программирования. Часть 2

8. ZIO, TF и основной ФП-стэк в Scala

9. Функциональные стримы на примере fs2, работа с бд

10. Акторы в Scala



Functional vs Object Oriented

Scala - это ООП язык, т.к.
каждое значение в Scala -
это объект

Scala - это ФП язык, т.к.
каждая функция в Scala -
это значение



Functional vs Object Oriented

Scala != Haskell

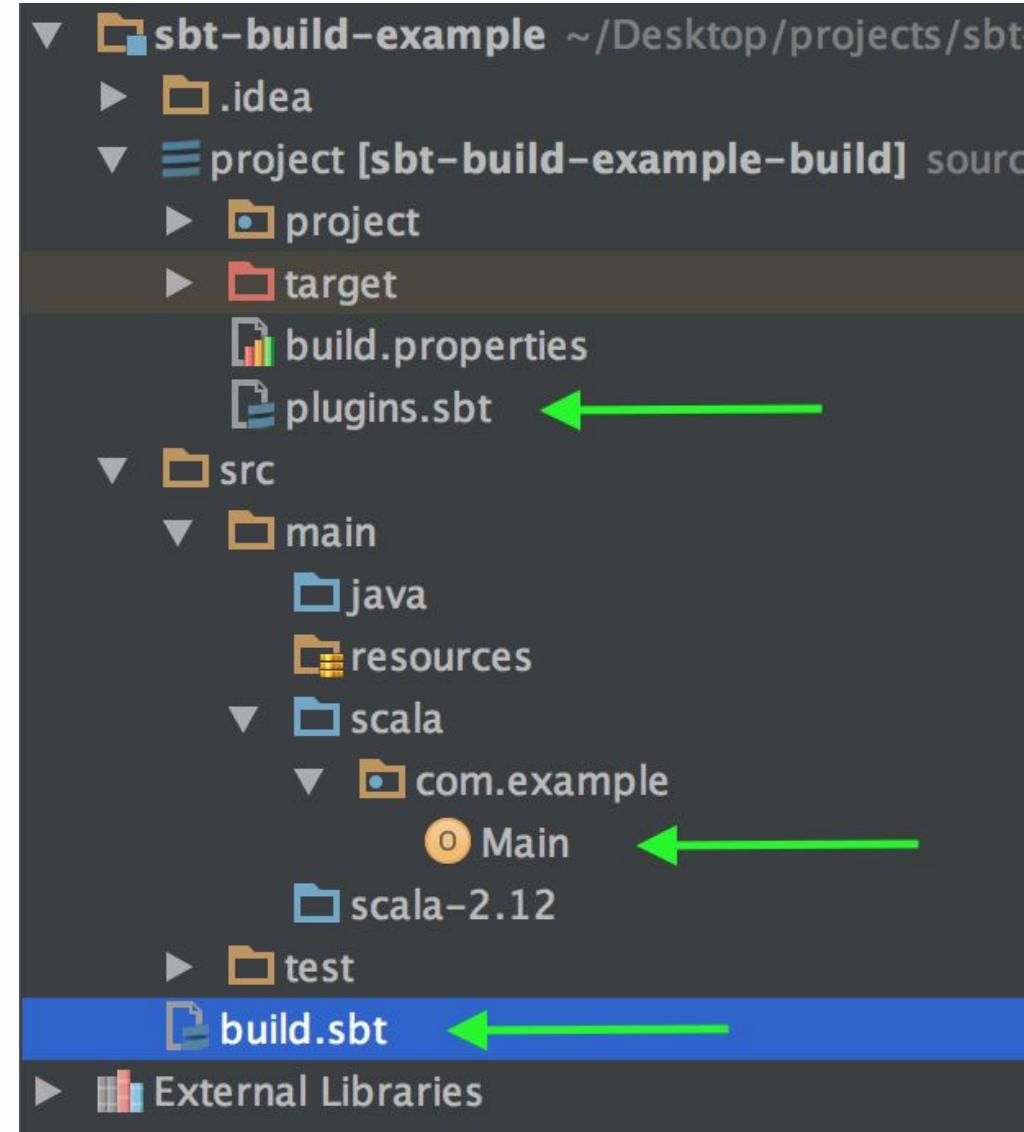
Scala != Better Java

Структура sbt проекта

sbt - scala build tool

<https://www.scala-sbt.org/>

ОСНОВНОЙ ИНСТРУМЕНТ
сборки scala проектов





Переменные
Блоки кода
Функции



Переменные

val

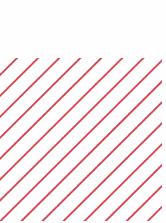
-immutable

-can be lazy

var

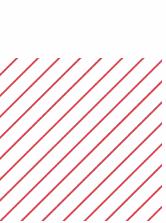
-mutable

-not lazy



Управляющие конструкции

- if/then/else
- for loops
- while loops
- try/catch/finally
- for comprehension
- match expressions



Функции vs методы

def Функции методы

```
def foo(x: Int): Int
```

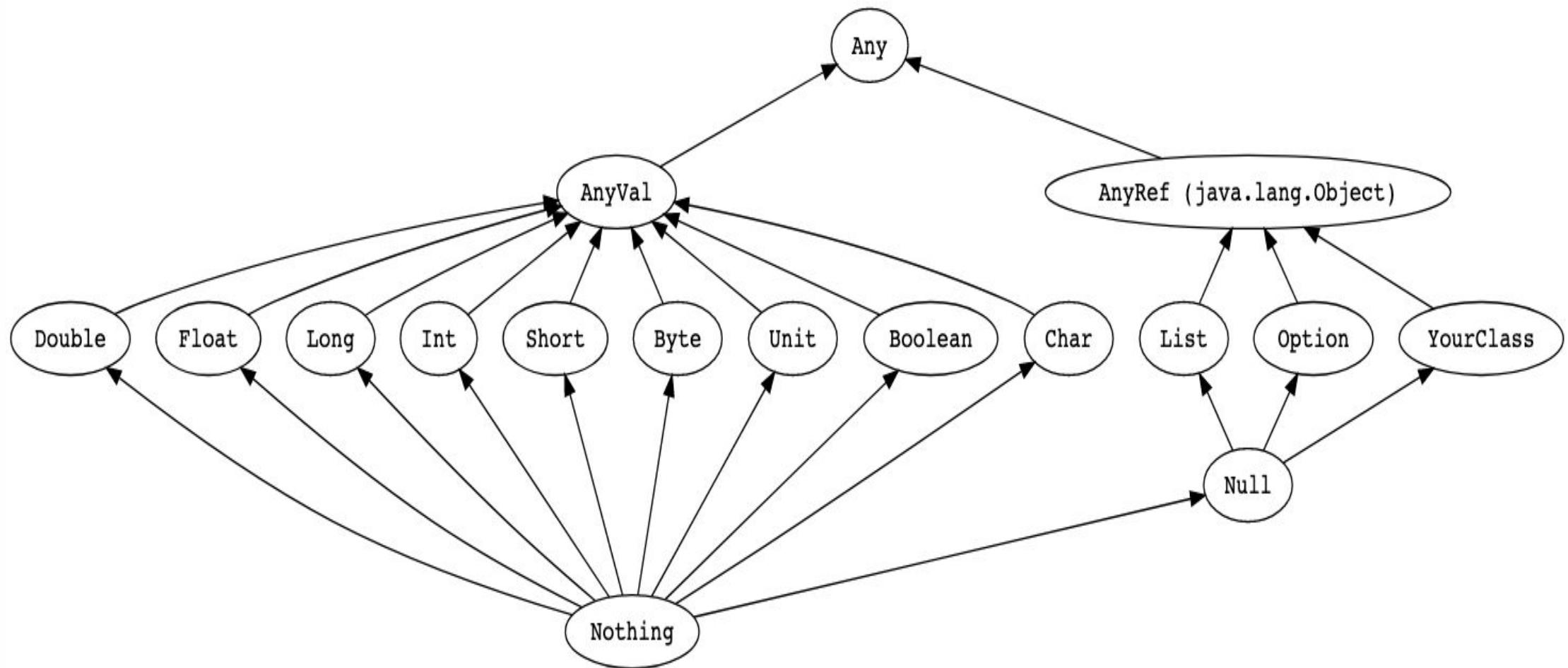
val Функции значения

```
val foo: Int => Int
```



Scala Type System

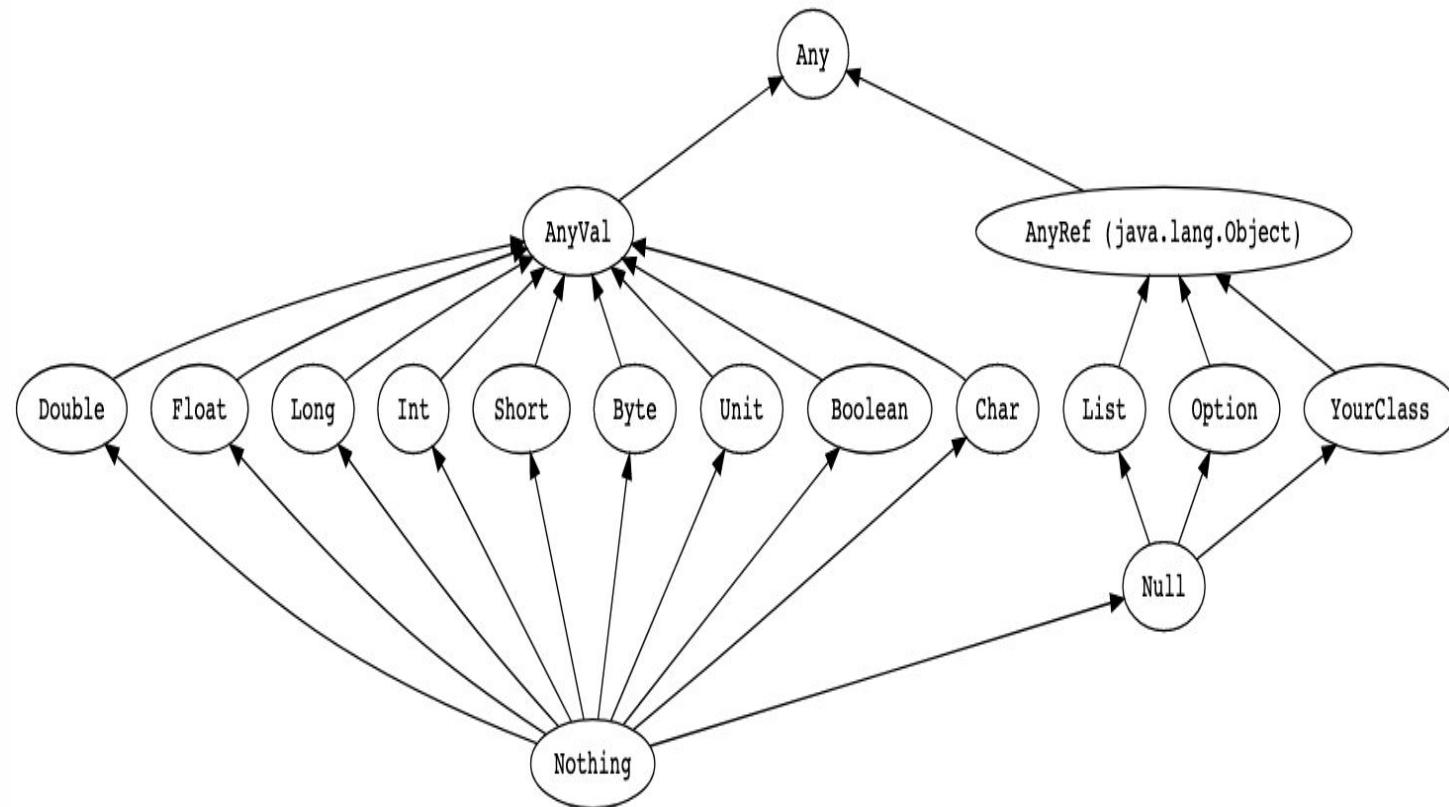
Scala Type System



Scala Type System: Unit

Unit

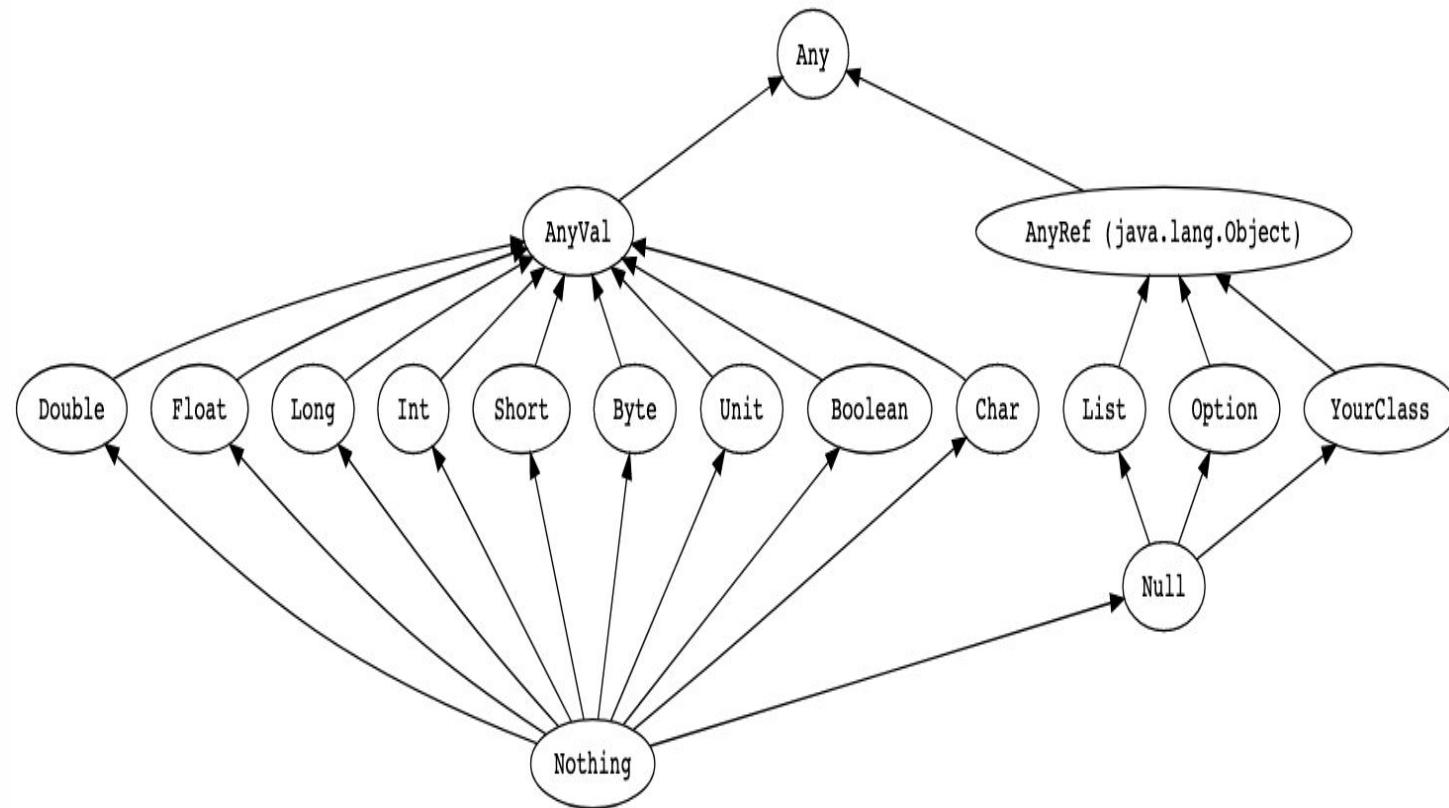
- Имеет один единственный экземпляр
- () - литерал



Scala Type System: Null

Null

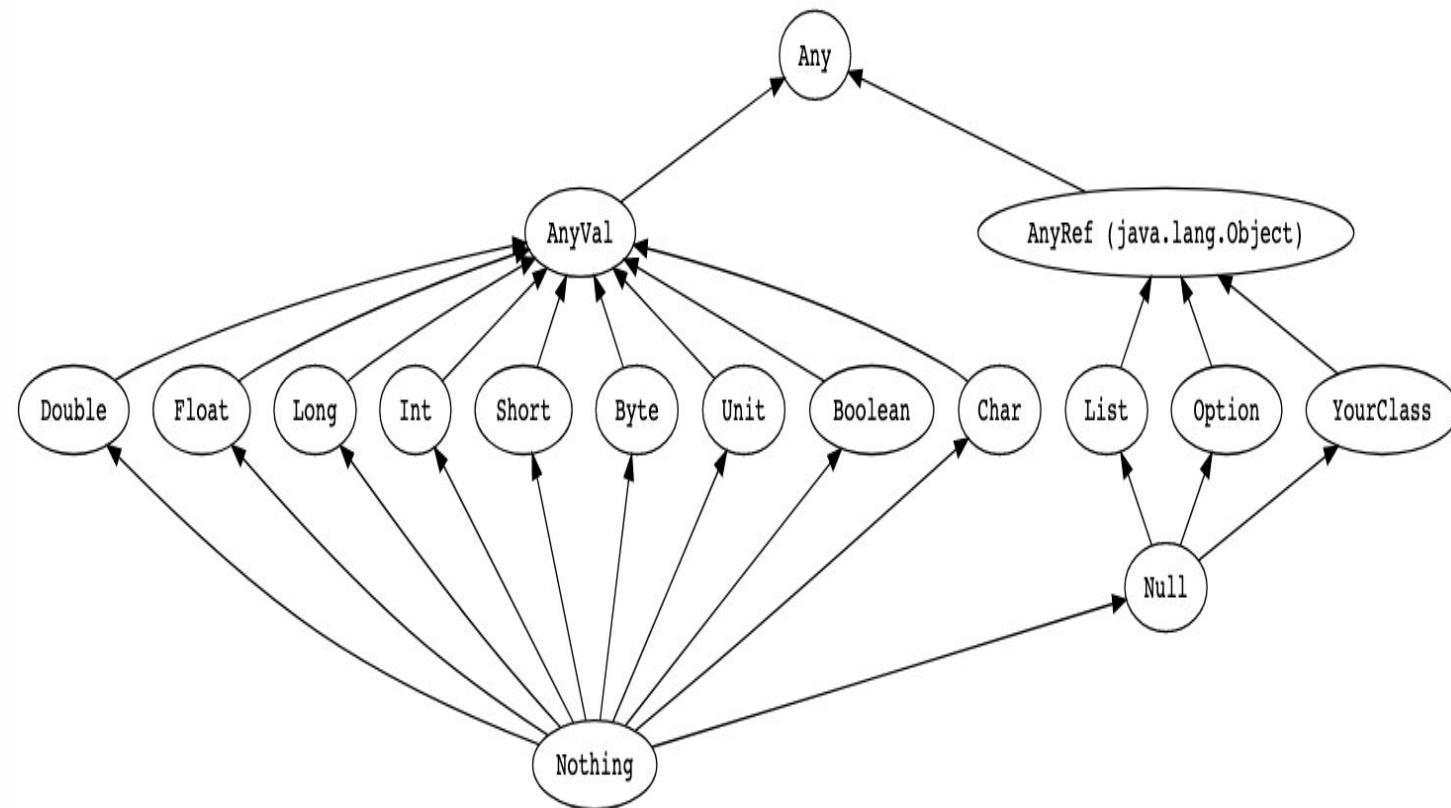
- Имеет один единственный экземпляр
- null - литерал
- jvm interop



Scala Type System: Nothing

Nothing

- Не имеет экземпляра
- Означает расходимость
- jvm interop





generics



Generics

```
def sum(x: Int): Int
```



Generics

```
def sum[T](x: T): T
```



Generics

```
def foo[T <: A](x: T) = ???
```

```
def foo[T >: A](x: T) = ???
```



traits
classes
objects



Traits

- конкретные и абстрактные методы
- нет параметров конструктора
- не можем создавать экземпляры
- поддерживают множественное подмешивание
- можно подмешивать при создании экземпляра



Classes and Objects

- классы являются шаблонами для создания объектов
- классы могут иметь поля, методы, конструкторы
- классы могут наследовать другие классы
- экземпляры классов называются объектами



case Classes and Objects

- иммутабельная структура данных
- объект companion
- copy
- hashCode
- equals
- toString



Итоги

Спасибо за внимание