THE KOTLIN PROGRAMMING LANGUAGE

Sergey Lukjanov, 2012
slukjanov@mirantis.com

WHAT IS KOTLIN?

MIRANTIS

WHAT IS KOTLIN?

- JVM-targeted

WHAT IS KOTLIN?

- JVM-targeted
- Statically typed

WHAT IS KOTLIN?

- JVM-targeted
- Statically typed
- Object-oriented

WHAT IS KOTLIN?

- JVM-targeted
- Statically typed
- Object-oriented
- General purpose

WHAT IS KOTLIN?

- JVM-targeted
- Statically typed
- Object-oriented
- General purpose
- Programming language

WHAT IS KOTLIN?

- JVM-targeted
- Statically typed
- Object-oriented
- General purpose
- Programming language
- Docs available today

WHAT IS KOTLIN?

- JVM-targeted
- Statically typed
- Object-oriented
- General purpose
- Programming language
- Docs available today
- Public beta in Q1 2012

OUTLINE

OUTLINE

- Motivation

OUTLINE

- Motivation
- Design goals

OUTLINE

- Motivation
- Design goals
- Feature overview

OUTLINE

- Motivation
- Design goals
- Feature overview
- Basic syntax

OUTLINE

- Motivation
- Design goals
- Feature overview
- Basic syntax
- Classes, types, inheritance

OUTLINE

- Motivation
- Design goals
- Feature overview
- Basic syntax
- Classes, types, inheritance
- Generics

OUTLINE

- Motivation
- Design goals
- Feature overview
- Basic syntax
- Classes, types, inheritance
- Generics
- High-order functions

OUTLINE

- Motivation
- Design goals
- Feature overview
- Basic syntax
- Classes, types, inheritance
- Generics
- High-order functions
- Type-safe Groovy-style builders

OUTLINE

- Motivation
- Design goals
- Feature overview
- Basic syntax
- Classes, types, inheritance
- Generics
- High-order functions
- Type-safe Groovy-style builders
- Pattern matching

MOTIVATION

MOTIVATION

- IDEA codebase ≥ 200MB of Java-code

MOTIVATION

- IDEA codebase ≥ 200MB of Java-code
- ??

MOTIVATION

- IDEA codebase ≥ 200MB of Java-code
- ??
- ??

MOTIVATION

- IDEA codebase ≥ 200MB of Java-code
- ??
- ??
- ??

DESIGN GOALS

DESIGN GOALS

- Full **Java interoperability**

DESIGN GOALS

- Full **Java interoperability**
- **Compiles** as fast as **Java**

DESIGN GOALS

- Full **Java interoperability**
- **Compiles** as fast as **Java**
- **Safer** than **Java**

DESIGN GOALS

- Full **Java interoperability**
- **Compiles** as fast as **Java**
- **Safer** than **Java**
- More **concise** than **Java**

DESIGN GOALS

MIRANTIS

- Full **Java interoperability**
- **Compiles** as fast as **Java**
- **Safer** than **Java**
- More **concise** than **Java**
- Way **simpler** than **Scala**

FEATURE OVERVIEW 1/2

FEATURE OVERVIEW 1/2

- Static null-safety guarantees

FEATURE OVERVIEW 1/2

- Static null-safety guarantees
- Traits (interfaces with default impl)

FEATURE OVERVIEW 1/2

- Static null-safety guarantees
- Traits (interfaces with default impl)
- First-class delegation

FEATURE OVERVIEW 1/2

- Static null-safety guarantees
- Traits (interfaces with default impl)
- First-class delegation
- Properties (instead of fields)

FEATURE OVERVIEW 1/2

- Static null-safety guarantees
- Traits (interfaces with default impl)
- First-class delegation
- Properties (instead of fields)
- Reified generics

FEATURE OVERVIEW 1/2

- Static null-safety guarantees
- Traits (interfaces with default impl)
- First-class delegation
- Properties (instead of fields)
- Reified generics
- Declaration-site variance & "Type projections"

FEATURE OVERVIEW 1/2

- Static null-safety guarantees
- Traits (interfaces with default impl)
- First-class delegation
- Properties (instead of fields)
- Reified generics
- Declaration-site variance & “Type projections”
- High-order functions (“closures”)

FEATURE OVERVIEW 1/2

- Static null-safety guarantees
- Traits (interfaces with default impl)
- First-class delegation
- Properties (instead of fields)
- Reified generics
- Declaration-site variance & “Type projections”
- High-order functions (“closures”)
- Extension functions

FEATURE OVERVIEW 1/2

- Static null-safety guarantees
- Traits (interfaces with default impl)
- First-class delegation
- Properties (instead of fields)
- Reified generics
- Declaration-site variance & “Type projections”
- High-order functions (“closures”)
- Extension functions
- Inline-functions (zero-overhead closures)

FEATURE OVERVIEW 2/2

FEATURE OVERVIEW 2/2

- Tuples

FEATURE OVERVIEW 2/2

- Tuples
- Modules and build infrastructure

FEATURE OVERVIEW 2/2

- Tuples
- Modules and build infrastructure
- Pattern matching

FEATURE OVERVIEW 2/2

- Tuples
- Modules and build infrastructure
- Pattern matching
- Range expressions

FEATURE OVERVIEW 2/2

- Tuples
- Modules and build infrastructure
- Pattern matching
- Range expressions
- String templates

FEATURE OVERVIEW 2/2

- Tuples
- Modules and build infrastructure
- Pattern matching
- Range expressions
- String templates
- Singletons

FEATURE OVERVIEW 2/2

- Tuples
- Modules and build infrastructure
- Pattern matching
- Range expressions
- String templates
- Singletons
- Operator overloading

FEATURE OVERVIEW 2/2

- Tuples
- Modules and build infrastructure
- Pattern matching
- Range expressions
- String templates
- Singletons
- Operator overloading
- Full-featured **IDE** by JetBrains

CODE EXAMPLES

HELLO, WORLD!

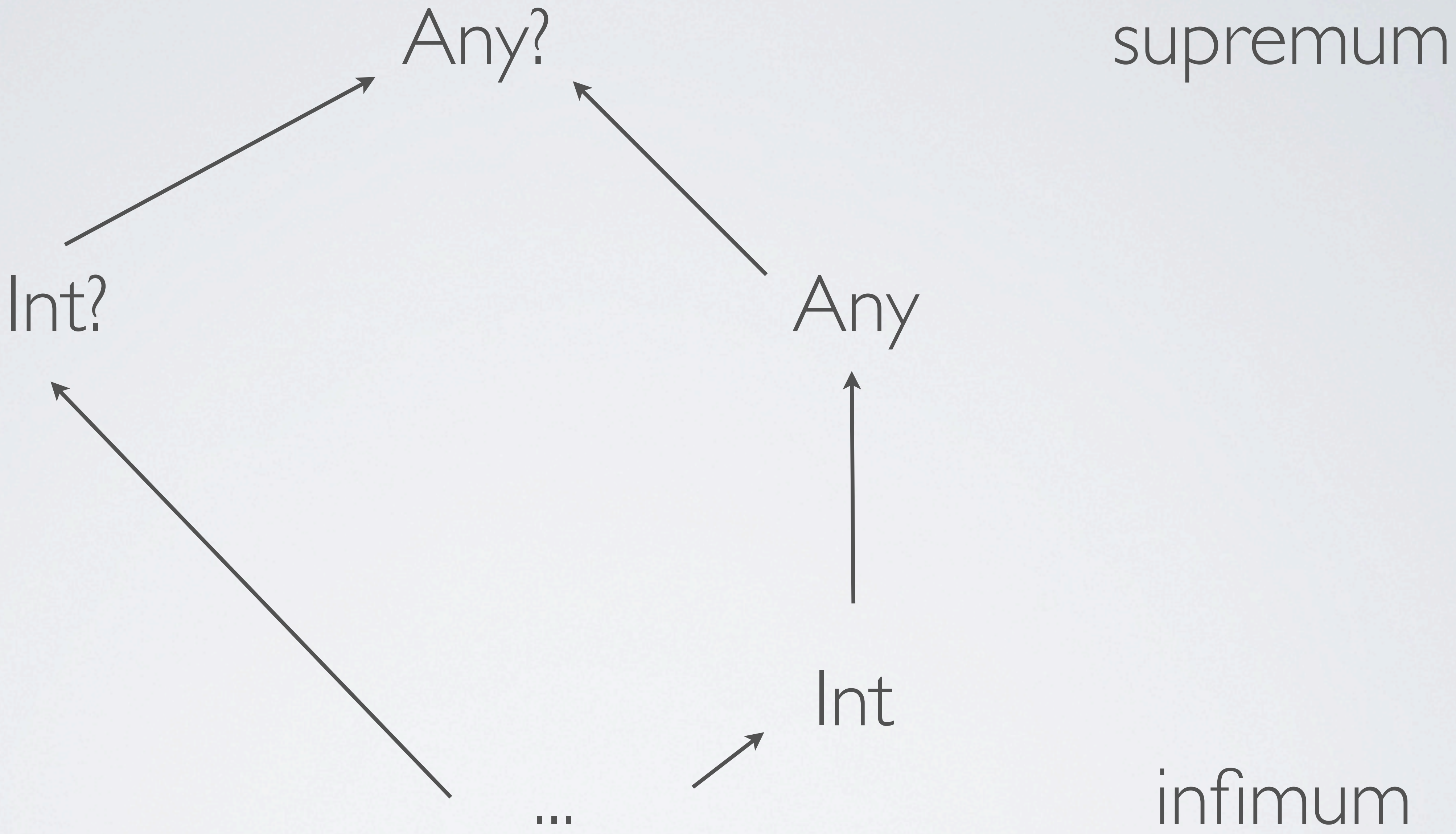
```
package hw.simple

fun main(args : Array<String>) : Unit {
    println("Hello, World!");
}

fun println(str : String) /* : Unit */ {
    System.out?.println(str);
}
```

HELLO, <NAMES>!

TYPES HIERARCHY



Complete lattice