ThoughtWorks®

Kotlin 实战

开启高效开发之路

张亚鹏

KOTLIN 开启高效开发之路

I. 为什么选择 Kotlin

II. 用 Kotlin 快速启动项目

III. 平滑迁移至 Kotlin

IV. Kotlin 奇技赢巧

KOTLIN 开启高效开发之路

WHY KOTLIN

I. 为什么选择 Kotlin

So Java is great, just like C is great. But just like C, Java is old.

So Java is great, just like C is great. But just like C, Java is old.

所以 Java 是伟大的，正像是 C 的伟大。但正像 C 一样，Java 老了。

JAVA - TIOBE DEC 2017

Dec 2017	Dec 2016	Change	Programming Language	Ratings	Change
1	1		Java	13.268%	-4.59%
2	2		C	10.158%	+1.43%
3	3		C++	4.717%	-0.62%
4	4		Python	3.777%	-0.46%
5	6	↑	C#	2.822%	-0.35%
6	8	↑	JavaScript	2.474%	-0.39%
7	5	↓	Visual Basic .NET	2.471%	-0.83%
8	17	⇈	R	1.906%	+0.08%
9	7	↓	PHP	1.590%	-1.33%
10	18	⇈	MATLAB	1.569%	-0.25%
11	13	↑	Swift	1.566%	-0.57%
12	11	↓	Objective-C	1.497%	-0.83%
13	9	⇊	Assembly language	1.471%	-1.07%
14	10	⇊	Perl	1.437%	-0.90%
15	12	↓	Ruby	1.424%	-0.72%
16	15	↓	Delphi/Object Pascal	1.395%	-0.55%
17	16	↓	Go	1.387%	-0.55%
18	25	⇈	Scratch	1.374%	+0.19%
19	20	↑	PL/SQL	1.368%	-0.13%
20	14	⇊	Visual Basic	1.347%	-0.62%

解决 JAVA 的痛点

- 空指针处理
- List / Map 初始化
- 伪函数式编程
- 代码冗余，如POJO
- 不支持可选参数
- 不支持运算符重载
- 不支持字符串插值
- **你的吐槽 x 10,000,000+**

KOTLIN AND SCALA

KOTLIN AND SCALA

发布：2004

版本：2.12.4

平台：JVM, JavaScript

协议：BSD 3-clause

发布：2011

版本：1.2

平台：JVM, JavaScript

协议：Apache 2

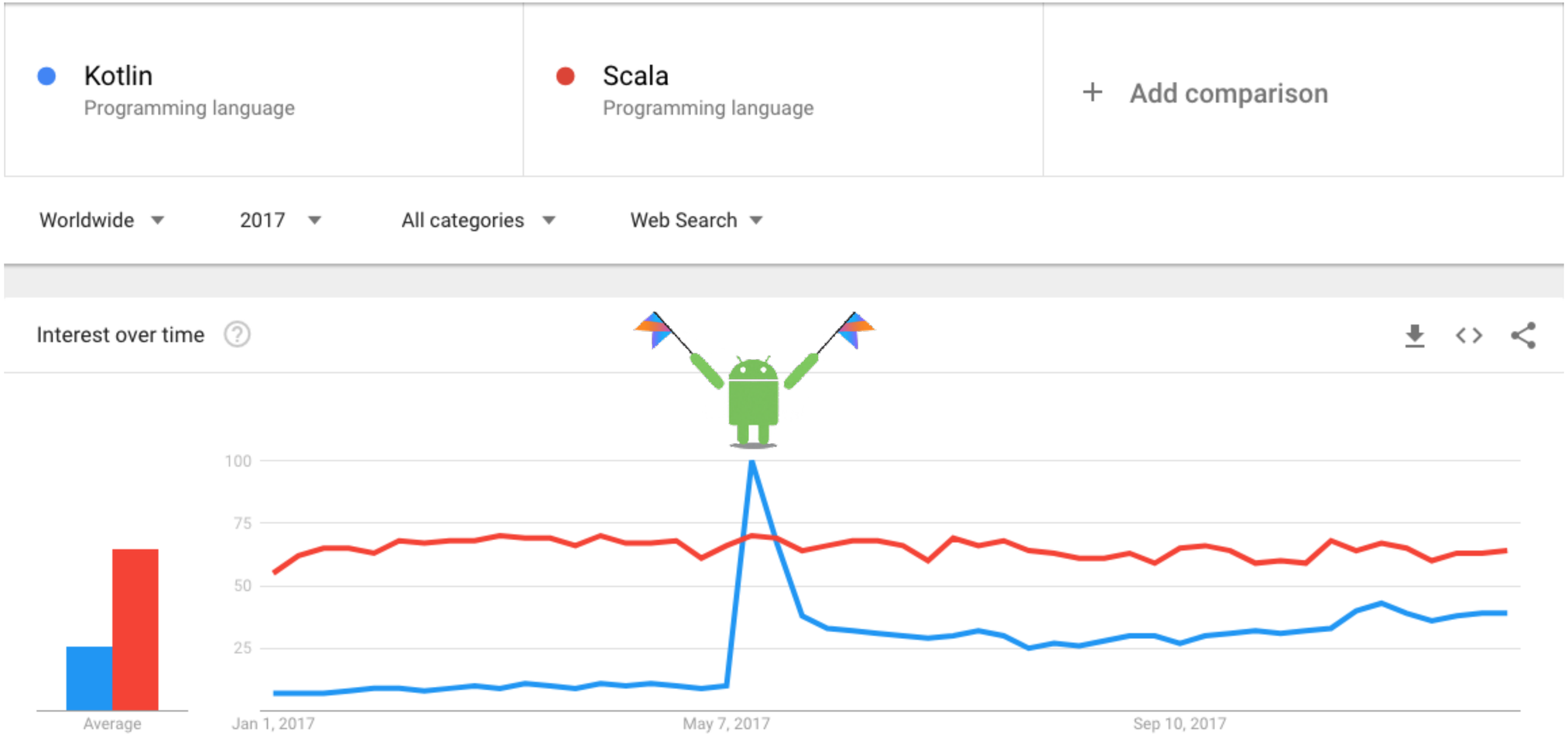
KOTLIN VS SCALA - TIOBE DEC 2017

Position	Programming Language	Ratings
21	SAS	1.346%
22	Alice	1.280%
23	Dart	1.274%
24	Scala	1.160%
25	Erlang	1.135%
26	Awk	1.050%
27	Transact-SQL	1.045%
28	Kotlin	0.994%
29	COBOL	0.961%
30	D	0.950%
31	LabVIEW	0.838%
32	VBScript	0.799%
33	Prolog	0.755%
34	ABAP	0.739%
35	Crystal	0.727%
36	VHDL	0.696%
37	Fortran	0.663%
38	Lua	0.635%
39	(Visual) FoxPro	0.576%
40	Apex	0.545%

KOTLIN VS SCALA - TIOBE DEC 2017

Position	Programming Language	Ratings
21	SAS	1.346%
22	Alice	1.280%
23	Dart	1.274%
24	Scala	1.160%
25	Erlang	1.135%
26	Awk	1.050%
27	Transact-SQL	1.045%
28	Kotlin	0.994%
29	COBOL	0.961%
30	D	0.950%
31	LabVIEW	0.838%
32	VBScript	0.799%
33	Prolog	0.755%
34	ABAP	0.739%
35	Crystal	0.727%
36	VHDL	0.696%
37	Fortran	0.663%
38	Lua	0.635%
39	(Visual) FoxPro	0.576%
40	Apex	0.545%

KOTLIN VS SCALA - 谷歌趋势 2017



KOTLIN VS SCALA - 八卦

Scala、Groovy、Clojure 、Kotlin 分别解决了 Java 的什么痛点?

补充一下，目前JVM上支持的语言光脚本就有一百多种，但是当前常见的有：Kotlin, Scala, Java, Ceylon, Clojure, Ruby(JRuby), Python(Jython), Groovy, Javascript(Nashorn)

关注问题 写回答 4 条评论 分享 邀请回答 ··· 收起

19 个回答 默认排序

呵呵一笑百媚生
Reacting to state changes is always b...

91 人赞同了该回答

随便瞎扯几句。

groovy：我不是针对java，而是所有不能愉快写DSL的都是*级

clojure：我真的不是针对java，只是关于并发，我有一个大胆的想法

scala：我也真的不是针对java，只是关于并发，我还有一种大胆的想法

kotlin：我就是处处针对java

编辑于 2017-11-01

▲ 91 ▼ 27 条评论 分享 收藏 感谢

匿名用户

43 人赞同了该回答

综合各位答主的意思，就是：

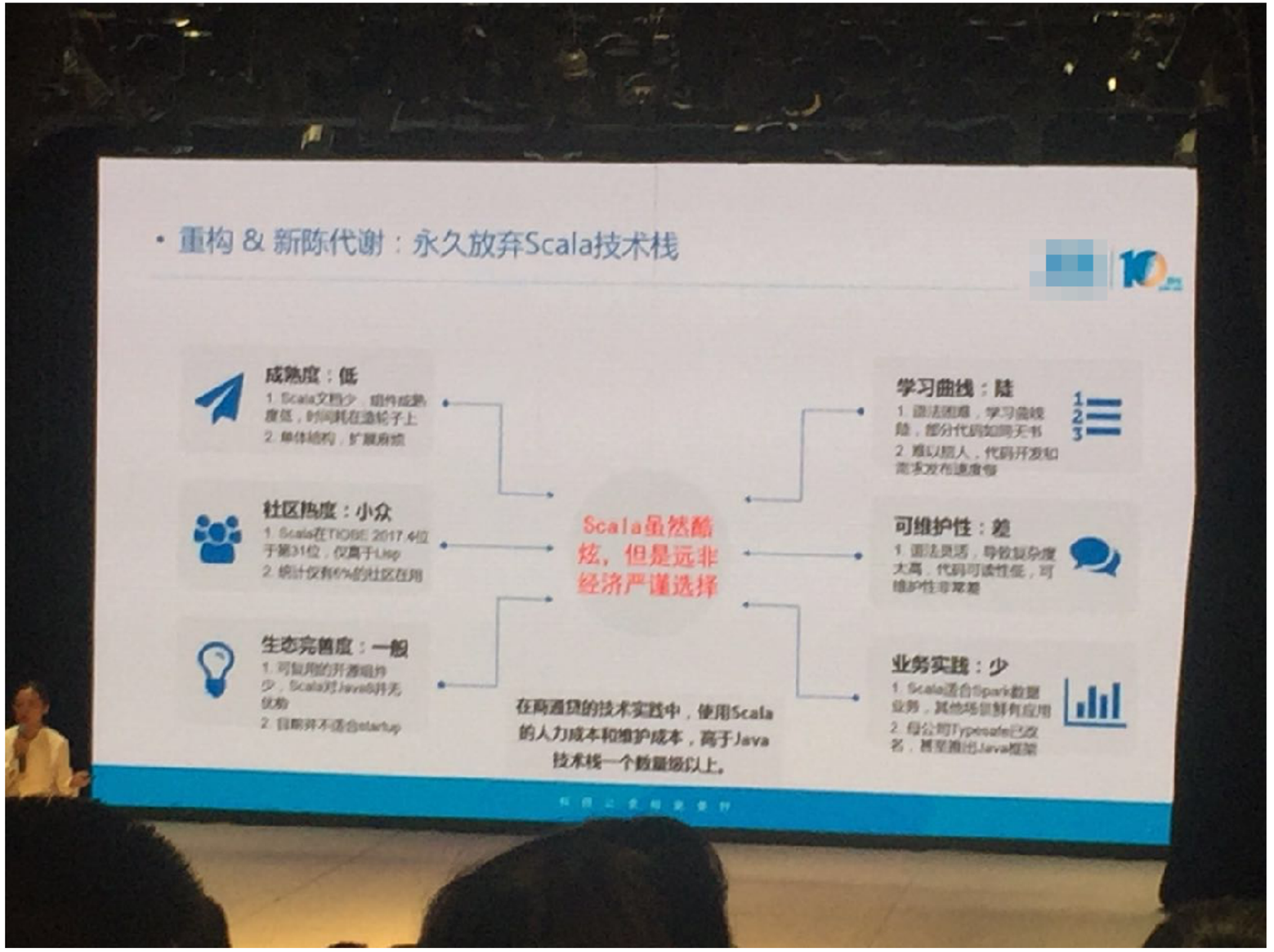
Scala：想解决Java表达能力不足的问题

Groovy：想解决Java语法过于冗长的问题

Clojure：想解决Java没有函数式编程的问题

Kotlin：想解决Java

KOTLIN VS SCALA - 八卦



KOTLIN VS SCALA - TL;DR 版总结

特点比较

	Java	Kotlin	Scala
编程范式	面向对象	面向对象 函数式	面向对象 函数式
与 Java 互操作性	N/A	100%	能
学习曲线	N/A	平缓	陡峭
从 Java 迁移难度	N/A	简单	复杂
可维护性	高	较高	低

KOTLIN VS SCALA - 结论

解决 JAVA 的痛点

空指针处理

```java
public class NullPointer {
    public static void log(String message) {
        if (message == null) {
            System.out.println(message.length());
        }
    }
}
```

```kotlin
fun main(args: Array<String>) {
    val a: Int = null // compilation error
    val b: Int? = null

    val s1: String = "abc"
    val s2: String? = null
    s1.length
    s2.length // compilation error
    s2?.length
}
```

解决 JAVA 的痛点

List / Map 初始化

```java
public class MapInitialize {
    public static void main(String[] args) {
        List<Integer> list = Arrays.asList(1, 2, 3);
        Map<String, Integer> map = new HashMap<>();
        map.put("a", 1);
        System.out.println(map);
    }
}
```

```java
public class MapInitialize {
    private static final Map<String, String> myMap;
    static
    {
        myMap = new HashMap<String, String>();
        myMap.put("a", "1");
        myMap.put("b", “2");
    }
}
```

```kotlin
fun main(args: Array<String>) {
    val list = listOf(1, 2, 3)
    val map = mapOf("a" to 1, "b" to 2)
}
```

```kotlin
class MapInitialize {
    val map = mapOf(Pair("a", 1), Pair("b", 2))
}
```

解决 JAVA 的痛点

伪函数式编程

```java
public class FakeFunctional {
    public static void main(String[] args) {
        List<Integer> list = Arrays.asList(null, 0, 1, 2);
        List<Integer> results = list.stream()
                .filter(integer -> integer != null)
                .map(integer -> integer * 2)
                .collect(Collectors.toList());
    }
}
```

```kotlin
fun main(args: Array<String>) {
    val list = listOf(null, 0, 1, 2) // List<Int?>
    val results = list
            .filter { integer -> integer != null }
            .map { integer -> integer!! * 2 }
}
```

```kotlin
fun main(args: Array<String>) {
    val list = listOf(null, 0, 1, 2)
    val results = list
            .filter { it != null }
            .map { it!! * 2 }
}
```

解决 JAVA 的痛点

代码冗余，如POJO

```java
public class Pojo {
    private String user;
    private Integer age;

    public Pojo(String user) {
        this.user = user;
    }

    public Pojo(String user, Integer age) {
        this.user = user;
        this.age = age;
    }

    public String getUser() {
        return user;
    }

    public void setUser(String user) {
        this.user = user;
    }

    public Integer getAge() {
        return age;
    }

    public void setAge(Integer age) {
        this.age = age;
    }
}
```

```kotlin
class Pojo(var user: String, var age: Int? = null)


fun main(args: Array<String>) {
    var a = Pojo("a")
    val b = Pojo("b", 18)
    a = Pojo("A") // it works
    b = Pojo("A") // compilation error
}
```

解决 JAVA 的痛点

不支持可选参数

```java
public class NoOptionalParam {

    public static Integer increase(Integer n, Integer i) {
        return n + i;
    }

    public static Integer increase(Integer n) {
        return increase(n, 1);
    }

    public static void main(String[] args) {
        System.out.println(increase(1));
    }
}
```

```kotlin
fun increase(n: Int, i: Int = 1): Int = n + i

fun main(args: Array<String>) {
    println(increase(1))
}
```

解决 JAVA 的痛点

不支持运算符重载

```java
public class Complex {
    private double real;
    private double imaginary;

    public Complex(double real, double imaginary) {
        this.real = real;
        this.imaginary = imaginary;
    }

    public Complex add(Complex c) {
        return new Complex(
                real + c.real,
                imaginary + c.imaginary
        );
    }

    public static void main(String[] args) {
        Complex a = new Complex(1, 2),
                b = new Complex(3, 4);
        System.out.println(a.add(b));
    }
}
```

```kotlin
class Complex(val real: Double, val imaginary: Double) {
    operator fun plus(c: Complex) =
            Complex(real + c.real, imaginary + c.imaginary)
}

fun main(args: Array<String>) {
    val a = Complex(1.0, 2.0)
    val b = Complex(3.0, 4.0)
    println(a + b)
}
```

解决 JAVA 的痛点

不支持字符串插值

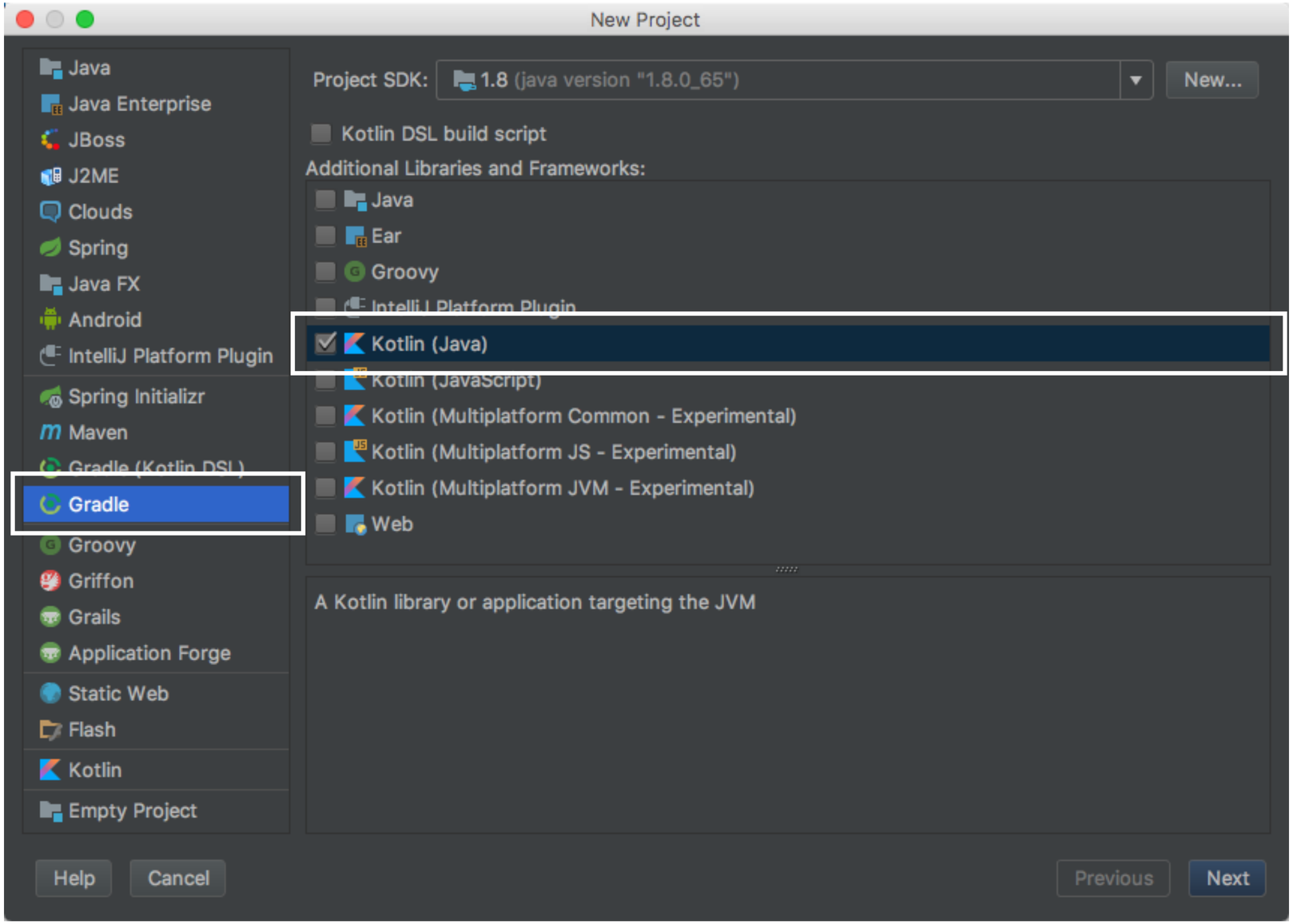
```java
public class NoStringInterpolation {

    public static void versionA() {
        Integer a = 1, b = 2;
        Integer c = a + b;
        System.out.println(a + " + " + b + " = " + c); // 1 + 2 = 3
    }

    public static void versionB() {
        Integer a = 1, b = 2;
        Integer c = a + b;
        StringBuilder sb = new StringBuilder();
        sb.append(a).append(" + ").append(b).append(" = ").append(c);
        System.out.println(sb.toString()); // 1 + 2 = 3
    }
}
```

```kotlin
fun main(args: Array<String>) {
    val a = 1
    val b = 2
    println("$a + $b = ${a + b}") // 1 + 2 = 3
}
```

START WITH KOTLIN

II. 用 Kotlin 快速启动项目

创建 KOTLIN 项目



Gradle

Kotlin (Java)

增加 SPRING BOOT 依赖

build.gradle

```
buildscript {
    ...
    dependencies {
        classpath "org.jetbrains.kotlin:kotlin-gradle-plugin:$kotlin_version"

    }
}

apply plugin: 'java'
apply plugin: 'kotlin'


dependencies {
    compile "org.jetbrains.kotlin:kotlin-stdlib-:$kotlin_version"

    testCompile group: 'junit', name: 'junit', version: '4.12'
}
```

增加 SPRING BOOT 依赖

build.gradle

```
buildscript {
    ...
    dependencies {
        classpath "org.jetbrains.kotlin:kotlin-gradle-plugin:$kotlin_version"
        classpath “org.springframework.boot:spring-boot-gradle-plugin:1.5.9.RELEASE”
    }
}

apply plugin: 'java'
apply plugin: 'kotlin'
apply plugin: 'org.springframework.boot'

dependencies {
    compile "org.jetbrains.kotlin:kotlin-stdlib-:$kotlin_version"
    compile "org.springframework.boot:spring-boot-starter-web:1.5.9.RELEASE"
    testCompile group: 'junit', name: 'junit', version: '4.12'
}
```

Spring Boot Gradle 插件依赖

Spring Boot Gradle 插件

Spring Boot 依赖

编写代码 - 应用

Application.kt

```
@SpringBootApplication
open class Application

fun main(args: Array<String>) {
    SpringApplication.run(Application::class.java, *args)
}
```

编写代码 - 应用

Application.kt

```kotlin
@SpringBootApplication
open class Application

fun main(args: Array<String>) {
    SpringApplication.run(Application::class.java, *args)
}
```

JavaApplication.java

```java
@SpringBootApplication
public class JavaApplication {
    public static void main(String[] args) {
        SpringApplication.run(JavaApplication.class, args);
    }
}
```

编写代码 - 应用

Application.kt

```kotlin
@SpringBootApplication
open class Application

fun main(args: Array<String>) {
    SpringApplication.run(Application::class.java, *args)
}
```

```
> ./gradlew bootRun
Tomcat started on port(s): 8080 (http)
Started ApplicationKt in 1.769 seconds (JVM
running for 2.028)
```

JavaApplication.java

```java
@SpringBootApplication
public class JavaApplication {
    public static void main(String[] args) {
        SpringApplication.run(JavaApplication.class, args);
    }
}
```

编写代码 - 应用

Application.kt

```kotlin
@SpringBootApplication
open class Application

fun main(args: Array<String>) {
    SpringApplication.run(Application::class.java, *args)
}
```

```
> ./gradlew bootRun
Tomcat started on port(s): 8080 (http)
Started ApplicationKt in 1.769 seconds (JVM
running for 2.028)
```

JavaApplication.java

```java
@SpringBootApplication
public class JavaApplication {
    public static void main(String[] args) {
        SpringApplication.run(JavaApplication.class, args);
    }
}
```

```
> curl -i localhost:8080
HTTP/1.1 404
Content-Type: application/json;charset=UTF-8
Transfer-Encoding: chunked
Date: Sat, 16 Dec 2017 17:06:09 GMT

{"timestamp":1513443969160,"status":
404,"error":"Not Found","message":"No message
available","path":"/"}
```

编写代码 - 控制器

GreetingController.kt

```
@RestController
class GreetingController {
    @RequestMapping("/greet")
    fun greet() = "Hello Kotlin"
}
```

编写代码 - 控制器

GreetingController.kt

```kotlin
@RestController
class GreetingController {
    @RequestMapping("/greet")
    fun greet() = "Hello Kotlin"
}
```

JavaApplication.java

```java
@RestController
public class JavaGreetingController {
    @RequestMapping("/greet")
    public String greet() {
        return "Hello Kotlin";
    }
}
```

编写代码 - 控制器

GreetingController.kt

```kotlin
@RestController
class GreetingController {
    @RequestMapping("/greet")
    fun greet() = "Hello Kotlin"
}
```

```
> ./gradlew bootRun
Tomcat started on port(s): 8080 (http)
Started ApplicationKt in 1.769 seconds (JVM
running for 2.028)
```

JavaApplication.java

```java
@RestController
public class JavaGreetingController {
    @RequestMapping("/greet")
    public String greet() {
        return "Hello Kotlin";
    }
}
```

编写代码 - 控制器

GreetingController.kt

```kotlin
@RestController
class GreetingController {
    @RequestMapping("/greet")
    fun greet() = "Hello Kotlin"
}
```

```
> ./gradlew bootRun
Tomcat started on port(s): 8080 (http)
Started ApplicationKt in 1.769 seconds (JVM
running for 2.028)
```

JavaApplication.java

```java
@RestController
public class JavaGreetingController {
    @RequestMapping("/greet")
    public String greet() {
        return "Hello Kotlin";
    }
}
```

```
> curl -i localhost:8080/greet
HTTP/1.1 200
Content-Type: text/plain;charset=UTF-8
Content-Length: 12
Date: Sat, 16 Dec 2017 17:18:30 GMT

Hello Kotlin
```

MIGRATE TO KOTLIN

III.平滑迁移至 Kotlin

平滑迁移至 KOTLIN

迁移到 Kotlin 的步骤

1. 配置构建脚本
2. 用 Kotlin 重写 Java 类

编辑构建脚本

build.gradle

```
version '1.0-SNAPSHOT'

apply plugin: 'java'


sourceCompatibility = 1.8

dependencies {

    testCompile group: 'junit', name: 'junit', version: '4.12'
}
```

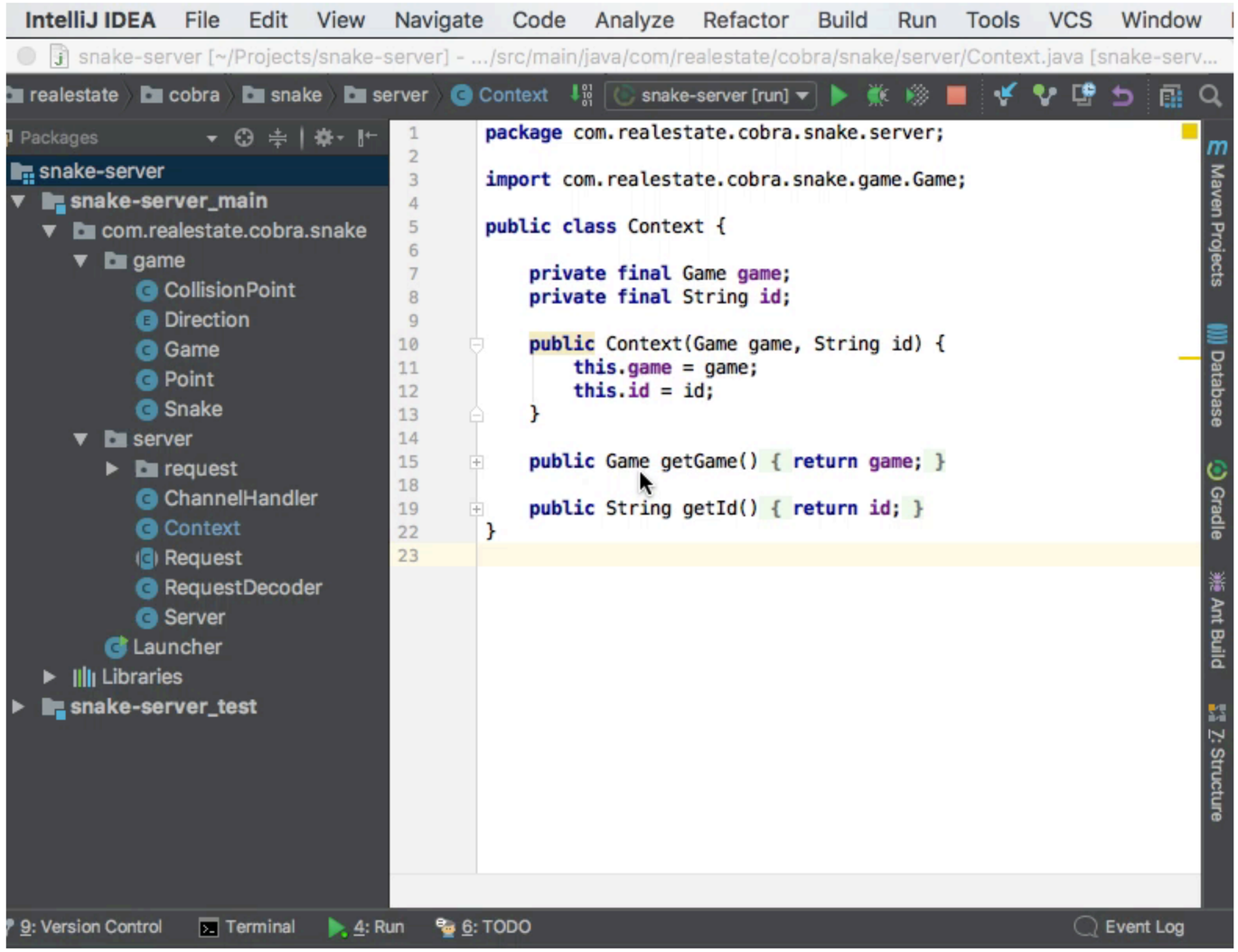
编辑构建脚本

build.gradle

```
version '1.0-SNAPSHOT'

apply plugin: 'java'
apply plugin: 'kotlin'

sourceCompatibility = 1.8

dependencies {
    compile "org.jetbrains.kotlin:kotlin-stdlib:$kotlin_version"
    testCompile group: 'junit', name: 'junit', version: '4.12'
}

buildscript {
    ext.kotlin_version = '1.2.0'

    repositories {
        mavenCentral()
    }
    dependencies {
        classpath "org.jetbrains.kotlin:kotlin-gradle-plugin:$kotlin_version"
    }
}
```

翻译代码 - 工具

Intellij IDEA 自动翻译



翻译代码 - POJO

Context.java => Context.kt

```java
public class Context {
    private final Game game;
    private final String id;

    public Context(Game game, String id) {
        this.game = game;
        this.id = id;
    }

    public Game getGame() {
        return game;
    }

    public String getId() {
        return id;
    }
}
```

```kotlin
class Context(val game: Game, val id: String)
```

翻译代码 - MAIN 方法

Launcher.java => Launcher.kt

```java
public class Launcher {

    private static final int PORT = 8080;

    public static void main(String[] args) {
        new Server(PORT).start();
    }

}
```

```kotlin
val PORT = 8080

fun main(args: Array<String>) {
    Server(PORT).start()
}
```

翻译代码 - 常规类

Point.java => Point.kt

```java
public class Point {
    private double x;
    private double y;

    public Point(Point point) {
        this.x = point.x;
        this.y = point.y;
    }

    public Point(double x, double y) {
        this.x = x;
        this.y = y;
    }

    public double getX() {
        return x;
    }

    public double getY() {
        return y;
    }

    public Point shift(double x, double y) {
        return new Point(this.x + x, this.y + y);
    }

    public boolean collide(Point point) {
        double dx = Math.abs(getX() - point.getX());
        double dy = Math.abs(getY() - point.getY());
        double distance = Math.sqrt(dx * dx + dy * dy);
        return distance < 20;
    }
}
```

```kotlin
open class Point(val x: Double, val y: Double) {

    constructor(point: Point) : this(point.x, point.y)

    fun shift(x: Double, y: Double): Point {
        return Point(this.x + x, this.y + y)
    }

    fun collide(point: Point): Boolean {
        val dx = Math.abs(x - point.x)
        val dy = Math.abs(y - point.y)
        val distance = Math.sqrt(dx * dx + dy * dy)
        return distance < 20
    }

}
```

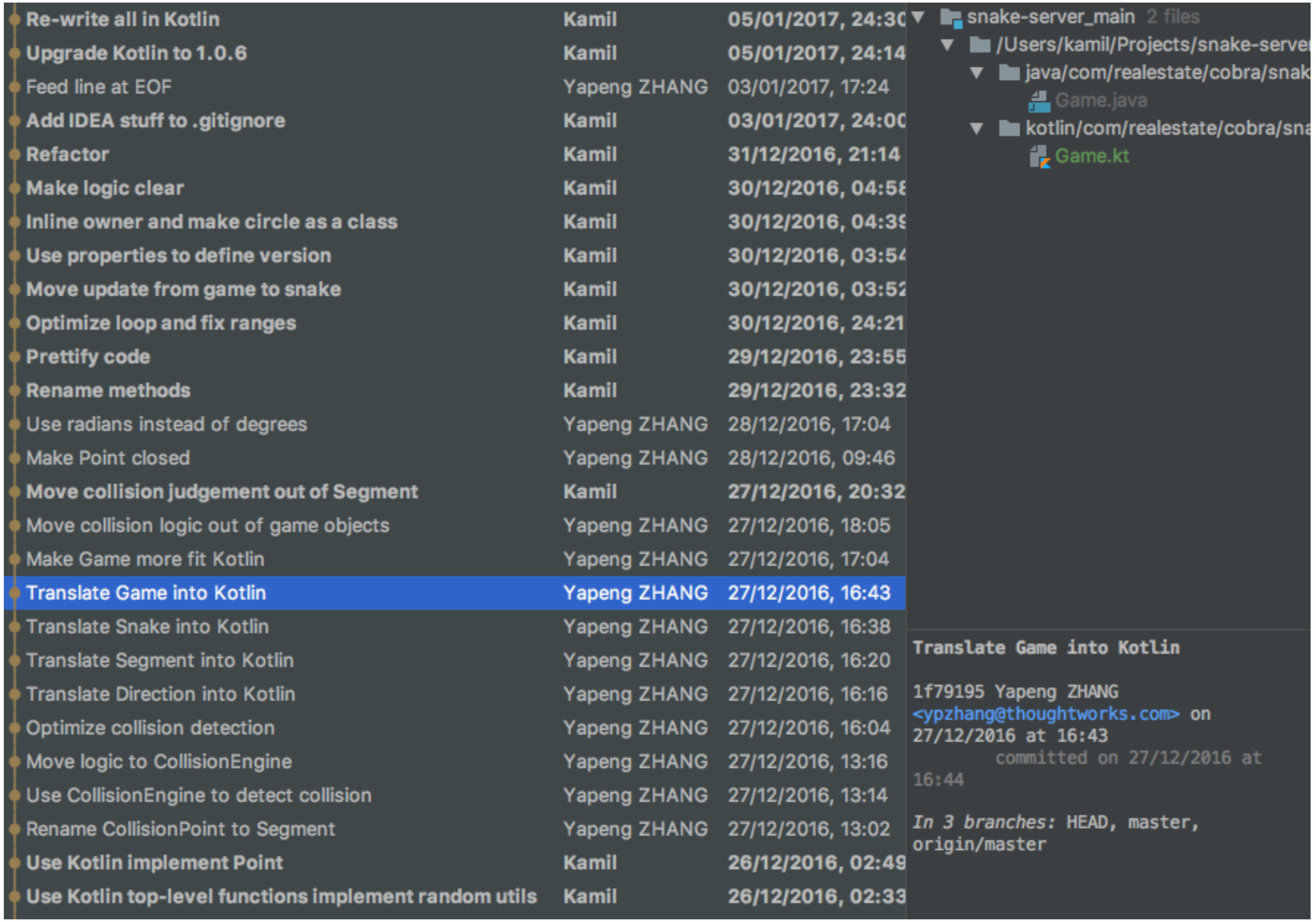
翻译代码 - 常规类

Game.java => Game.kt

```java
public class Game {

    private static final int MAX_FOODS = 100;

    private static final int LEFT_BOUND = 0;
    private static final int RIGHT_BOUND = 1600;
    private static final int TOP_BOUND = 0;
    private static final int BOTTOM_BOUND = 1200;
    private static final int SAFE_GAP = 400;

    private List<Snake> snakes = new LinkedList<>();
    private List<Point> foods = new
ArrayList<>(MAX_FOODS);

    private Map<String, Direction> directions = new
ConcurrentHashMap<>();

    public Game() {
        feed();
    }

}
```

```kotlin
class Game {

    companion object {
        private val MAX_FOODS = 100
        private val LEFT_BOUND = 0
        private val RIGHT_BOUND = 1600
        private val TOP_BOUND = 0
        private val BOTTOM_BOUND = 1200
        private val SAFE_GAP = 400
    }

    private val snakes = mutableListOf<Snake>()
    private val foods = mutableListOf<Food>()

    private val directions = mutableMapOf<String,
Command>()

    init {
        feed()
    }

}
```

迁移示例


Re-write all in Kotlin Kamil 05/01/2017, 24:30
Upgrade Kotlin to 1.0.6 Kamil 05/01/2017, 24:14
Feed line at EOF Yapeng ZHANG 03/01/2017, 17:24
Add IDEA stuff to .gitignore Kamil 03/01/2017, 24:00
Refactor Kamil 31/12/2016, 21:14
Make logic clear Kamil 30/12/2016, 04:58
Inline owner and make circle as a class Kamil 30/12/2016, 04:39
Use properties to define version Kamil 30/12/2016, 03:54
Move update from game to snake Kamil 30/12/2016, 03:52
Optimize loop and fix ranges Kamil 30/12/2016, 24:21
Prettify code Kamil 29/12/2016, 23:55
Rename methods Kamil 29/12/2016, 23:32
Use radians instead of degrees Yapeng ZHANG 28/12/2016, 17:04
Make Point closed Yapeng ZHANG 28/12/2016, 09:46
Move collision judgement out of Segment Kamil 27/12/2016, 20:32
Move collision logic out of game objects Yapeng ZHANG 27/12/2016, 18:05
Make Game more fit Kotlin Yapeng ZHANG 27/12/2016, 17:04
Translate Game into Kotlin Yapeng ZHANG 27/12/2016, 16:43
Translate Snake into Kotlin Yapeng ZHANG 27/12/2016, 16:38
Translate Segment into Kotlin Yapeng ZHANG 27/12/2016, 16:20
Translate Direction into Kotlin Yapeng ZHANG 27/12/2016, 16:16
Optimize collision detection Yapeng ZHANG 27/12/2016, 16:04
Move logic to CollisionEngine Yapeng ZHANG 27/12/2016, 13:16
Use CollisionEngine to detect collision Yapeng ZHANG 27/12/2016, 13:14
Rename CollisionPoint to Segment Yapeng ZHANG 27/12/2016, 13:02
Use Kotlin implement Point Kamil 26/12/2016, 02:49
Use Kotlin top-level functions implement random utils Kamil 26/12/2016, 02:33
snake-server_main 2 files
/Users/kamil/Projects/snake-serve
java/com/realestate/cobra/snak
Game.java
kotlin/com/realestate/cobra/sna
Game.kt
Translate Game into Kotlin
1f79195 Yapeng ZHANG
<ypzhang@thoughtworks.com> on
27/12/2016 at 16:43
committed on 27/12/2016 at
16:44
In 3 branches: HEAD, master,
origin/master

KOTLIN TRICKS

IV.Kotlin 奇技赢巧

扩展

```kotlin
fun oldGreet(who: String) = "$who: Hello"

fun String.greet() = "$this: Hello"

fun main(args: Array<String>) {
    println(oldGreet("Kotlin")) // Kotlin: Hello
    println("Kotlin".greet()) // Kotlin: Hello
}
```

APPLY

```
class StupidStringBuilder(var string: String) {
    fun append(new: String): StupidStringBuilder {
        string += new
        return this
    }

    override fun toString(): String = string
}

class StringBuilder(var string: String) {
    fun append(new: String) = this.apply { string += new }
    override fun toString(): String = string
}

fun main(args: Array<String>) {
    println(StupidStringBuilder("Kotlin").append(" is").append(" cool")) // Kotlin is cool
    println(StringBuilder("Kotlin").append(" is").append(" cool")) // Kotlin is cool
}
```

WITH

```
fun verboseDescribe(string: String) =
        "$string(${string.toLowerCase()}) has ${string.length} char(s)"

fun describe(string: String) = with (string) {
    "$this(${toLowerCase()}) has $length char(s)"
}

fun main(args: Array<String>) {
    println(verboseDescribe("Kotlin")) // Kotlin(kotlin) has 6 char(s)
    println(describe("Kotlin")) // Kotlin(kotlin) has 6 char(s)
}
```

ALL

```
const val MIN_WIDTH = 0
const val MIN_HEIGHT = 0
const val MAX_WIDTH = 1920
const val MAX_HEIGHT = 1080

fun badValidate(x: Int, y: Int): Boolean {
    return (x >= MIN_WIDTH
            && x <= MAX_WIDTH
            && y >= MIN_HEIGHT
            && y <= MAX_HEIGHT)
}

fun goodValidate(x: Int, y: Int): Boolean =
        (x in MIN_WIDTH..MAX_WIDTH && y in MIN_HEIGHT..MAX_HEIGHT)

fun validate(x: Int, y: Int): Boolean =
        listOf(x in MIN_WIDTH..MAX_WIDTH, y in MIN_HEIGHT..MAX_HEIGHT).all { it }

fun main(args: Array<String>) {
    println(badValidate(-1, 256))
    println(goodValidate(-1, 256))
    println(validate(-1, 256))
}
```

KOTLIN 开启高效开发之路

Q&A

问题解答

涉及代码

kotlin-playground Github

Java 痛点代码 src/main/java/painpoints/

Kotlin 解决痛点代码 src/main/kotlin/solve/

Kotlin 技巧代码 src/main/kotlin/tricks/

spring-boot-kotlin-template Github

最小 Kotlin 与 Spring Boot 模版

IT大咖说
知识共享平台

THANK YOU

张亚鹏

ThoughtWorks®