OSC 源创会 Opensource Innovation Meetup | IT大咖说 知识共享平台

2017源创会年终盛典

与电子标准院共建开源标准

12月23日　北京万豪酒店

主办方 开源中国 oschina.net　CESI

From Java To Kotlin

By 技术小黑屋

About Me

- 技术小黑屋(droidyue.com)博主
- Flipboard 中国 研发工程师
- Droidcon beijing 演讲嘉宾
- GDG beijing 分享讲师

What is Kotlin

- It’s a statically typed language
- It’s for modern multiplatform application development
- It’s one of the official languages of Android development
- It’s powered by JetBrains

Why Kotlin

- Build applications for JVM, Android, Browser, Native
- Pragmatic
- Concise
- Safe
- Interoperable
- Tool-friendly

Target Developers

- Android developers
- Java developers
- Those who are interested in Kotlin
- Those who want to have fun programming

Let’s get it started

- Compare business implementations achieved by Java and Kotlin respectively.
- Highlight Kotlin’s cool feature and ideas
- Finally talk about the way to learn Kotlin

Annoying Null checks in Java

//((a.b.c).title())

```
public String getTitle(ClassA  a) {
    if (a != null && a.b != null && a.b.c != null){
        return a.b.c.title;
    }
    return null;
}
```

Simple But Rough

```
public String getTitleInRoughWay(ClassA  a) {
    try  {
        return a.b.c.title;
    } catch (NullPointerException e) {
        return  null;
    }
}
```

Kotlin’s Perfect way

```
fun getTitleFromA(a : ClassA): String? {
    return a?.b?.c?.title
}

//String? means nullable

//?. is null safe operator
```

From Java’s utility method

```
public static boolean isEven(int value) {
        return value % 2 == 0;
}
//the call site
println("5 is even=${JavaRoot.isEven(5)}")
```

To Kotlin’s extension functions

```
fun Int.isEven(): Boolean {
    return this % 2 == 0
}

/*
isEven seems a real function of Int
But in fact it is not.
*/
```

Sort Collections In Java

```
public static void sortBooksByPrice(List<Book> books) {
    books.sort(new Comparator<Book>() {
            @Override
            public int compare(Book o1, Book o2) {
                if (o1.getPrice() > o2.getPrice()) {
                    return 1;
                } else if (o1.getPrice() < o2.getPrice()) {
                    return -1;
                }
                return 0;
            }
    });
}
```

Sort Collection In Kotlin

```
fun sortBooksByPrice(books: List<Book>) {
    books.sortedBy { it.price }
}
/*
Simple, readable and Self-explanatory
*/
```

Telescoping constructor issues

```
Pizza(int size) { … }

Pizza(int size, boolean cheese) { … }

Pizza(int size, boolean cheese, boolean pepperoni)
{ … }

Pizza(int size, boolean cheese, boolean pepperoni,
boolean bacon) { … }

/*It will go worse if we add more parameters*/
```

Builder pattern works, but …

```
Pizza pizza =
      new Pizza.Builder(12)
     .cheese(true)
     .pepperoni(true)
     .bacon(true)
     .build();
/* But we need to write the code of Builder
manually */
```

Kotlin’s default arguments

```
//declaration
class KotlinPizza (size: Int, cheese: Boolean = false,
pepperoni: Boolean = false, bacon: Boolean = false)

//example
KotlinPizza(2, bacon = true)
```

Lots of Mutables due to Java

- We have written lots of mutable (local) variables
- The collections are mutable by default
- Mutable states would cause lots of potential problems.

Mutable local variables

```
Public void someAction(Book book) {
    String name;
    if (book != null &book.getName() != null) {
        name = book.getName();
    } else {
        name = "Not Found";
    }
         other code here
         //name would be reassigned by accident
}
```

Read-only variables in Kotlin

```
fun someAction(book: Book?){
    val name = if (book?.name != null) {
        book.name
    } else {
        "Not Found"
    }
    //some other code here
        //Any reassignment will cause compile-time errors
}
/*
Val means read-only;var means mutable and readable
 This also goes for when,try-catch
 */
```

Collections in Kotlin

```
val list1 = listOf<String>() //read-only
val list2 = mutableListOf<String>() //mutable
val list3 = arrayListOf<String>() //mutable

val map1 = mapOf<String, String>() //read-only
val map2 = hashMapOf<String, String>() //mutable
val map3 = mutableMapOf<String, String>()//mutable

val set1 = setOf<String>() //read-only
val set2 = mutableSetOf<String>() //mutable
val set3 = hashSetOf<String>() //mutable

/*Use the read-only version when possible */
```

Heavyweight callbacks in Java

```
public class MakePizzaInJava {
    public interface OnPizzaMadeListener {
        void onPizzaMade();
    }
    public OnPizzaMadeListener listener;

    private void notifyPizzaMade() {
        if (listener != null) {
            listener.onPizzaMade();
        }
    }
}
```

Lightweight callback in Kotlin

```
class MakePizzaInKotlin {
    var onPizzaMadeAction: (() -> Unit)? = null
    fun notifyPizzaMade() {
        onPizzaMadeAction?.invoke()
    }
}

/* Lambda, function type*/
```

Other cool features

- Smart cast
- Data class
- Higher order functions
- Collection Streams API
- Easy singleton implementation

How to study Kotlin

- Analyze the Bytecode(javap)
- Decompile and Analyze the Java code(Kotlin IDE plugin)

Null Safe Code Example

```
fun testNullSafe(text: String?) {
    text?.hashCode()
}
```

Null safe Bytecode

```
public final static testNullSafe(Ljava/lang/String;)V
    @Lorg/jetbrains/annotations/Nullable;() // invisible,        parameter 0
   L0
    LINENUMBER 5 L0
    ALOAD 0
    DUP
    IFNULL L1   // check if null
    INVOKEVIRTUAL java/lang/String.hashCode ()I
    POP
    GOTO L2
   L1
    POP
   L2
  …
}
```

Decompiled Java code

```
public void testNullSafe (@Nullable String text) {
  if(text != null) {
    text.hashCode();
  }
}
```

More features we could study

- Object
- Val/var
- Lambda
- Lazy initialization
- Extension methods

Thanks and Questions