5. Kotlin

사용 목적

- Java 6.0과 완전 호환.
- Java와 성능차이를 비교했을 때 전혀없음. 코틀린 컴파일러가 생성한 바이트코드는 일반적인 자바 코도와 똑같이 실행됨.
- Android Java 1.8 Lambda를 사용하기 위해서는 min Sdk 24 필요. (안드로이드 7.0 누가 version)
    - Kotlin은 min Sdk 상관없이 Lambda식 사용가능. (함수형 프로그래밍 가능함)
- 타 언어의 장점들의 집약체
- nullable, non-nullable 타입으로 Null safety
    - 초기화되지 않는 사용을 컴파일러에서 막는다.
- Extention functions
    - Decorator pattern 이나 클래스 상속없이 기능을 추가할 수 있다.
    - Decorator pattern (https://gmlwjd9405.github.io/2018/07/09/decorator-pattern.html)

    - Extention functions

```
fun String.makePretty(): String {
    //… make the string pretty and return it
}
```

- Higher-order functions (고차함수)
    - Lambda 식
- Data Class
    - 내부적으로(컴파일러가) Getter, Setter 만들어서 명시적으로 만들필요 없음.
    - 간단한 Data Class 선언

    - data class

```
data class User(var name: String, var age: Int)
```

- Immutability (불변성)
    - val (value), var (variable) 변수 선언으로 mutable 가능한지 아닌지 선택할 수 있음.
        - java 의 final과 비슷하지만 추가적으로 val (value) 사용시 추가적으로 private setter, getter로 정의됨.
    - immutable collection interface를 지원한다.
        - listOf() 읽기 전용 빈 리스트 생성.
- Coroutines (코루틴)
    - 전형적으로 롱러닝 태스크를 수행하기 위해 스레드를 호출하는데 Coroutines는 스레드를 블로킹하지 않고 실행할 수 있다.

      겉으로는 동기적이면서 실제론 비동기적인 코드를 쓸 수 있다. 컴파일하는 동안 그 코드는 비동기적으로 바뀐다. 즉, 이 기술은 가상 머신이나 OS에 의존하지 않는다.
    - https://terms.naver.com/entry.nhn?docId=819147&cid=50376&categoryId=50376
- Type aliases
    - Type에 대해서 별칭으로 코드 가독성을 높일 수 있다.

```
typealias MapOfLists = Map<String, List>

fun useMap(map: MapOfList) {
  //...
}
```

도입한 회사

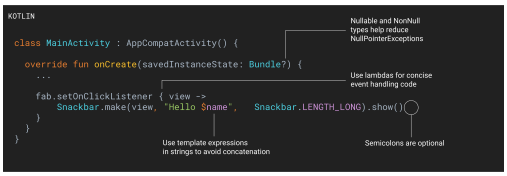
- 야놀자 노현석 개발자 (2017년 GDG 자료 중)
    - 발표 동영상
        - https://www.youtube.com/watch?v=IfSUPYVTz8c&feature=youtu.be
    - 발표 자료
        - https://speakerdeck.com/pluu/the-basic-kotlin
    - 현재 Java 내용들 Kotlin으로 변경 진행 중
- 배달의 민족 우아한형제들 최윤석 개발자
    - 베민프레시 기술에 도입
        - http://woowabros.github.io/experience/2017/07/18/introduction-to-kotlin-in-baeminfresh.html
        - NullPointerException
        - 안드로이드 보일러 플레이트와 다소 부족한 자바 언어의 표현력
        - 보일러 플레이트 코드 제거
        - 스트림 대안

Property

- lateinit
    - Property 초기화를 미루는 방법임. 변수 선언 후에 늦은 초기화가 가능함.
    - 제약 사항
        - lateinit는 var로 선언한 프로퍼티에만 사용할 수 있다.
        - lateinit는 클래스 몸체, Top-Level, 함수 내부에 선언한 프로퍼티에 사용할 수 있다. 주 생성자에서는 사용할 수 없다.
        - lateinit는 사용자 정의 getter/setter를 사용하지 않은 프로퍼티에만 사용할 수 있다.
        - null 허용 프로퍼티에는 사용할 수 없다.
        - 기초 타입 프로퍼티에는 사용할 수 없다.

        출처: https://kkangsnote.tistory.com/67 [깡샘의 토마토]

문법



- **간결한 문법**
    - 세미콜론 넣지않아도 됨.
    - new 키워드 쓰지 않고 객체 생성.
    - 타입 추론 지원으로 타입을 명시적으로 안써도됨.
- **널 안전성**
    - 널 값의 허용여부를 변수선언 시 명시적으로 써야함. (컴파일 단계에서 체크됨)
- **가변/불변 구분**
    - var : variable의 약어로 가변이 가능한 변수.
    - val : value의 약어로 불변인 변수.
    - 컬렉션 자료형에서 가변/불변 구분

▪ 가변 자료형

```
//
val immutable: List<String> = listOf("foo", "bar", "baz")
//
val mutable: MutableList<String> = mutableListOf("foo","
bar","baz")
```

- 람다 표현식 (Lambda)
  - 자바로 작성된 인터페이스에 한해 SAM(Single Abstract Method) 변환 지원.

▪ 람다 표현식

```
view.setOnClickListener {
        Toast.makeText(it.context, "Click", Toast.
LENGTH_SHORT).show()
}
```

- 스트림 API 지원
  - 자바8에서 사용하는 stream API를 자바로 사용하면 안드로이드 버전 6.0(마시멜로) 이상의 플랫폼에서만 사용가능함.
  - kotlin 표준 라이브러리를 통해 제공함.

▪ stream API

```
val items = listOf(10, 2, 3, 5, 6)


//     .
val sumOfEvens = items.filter { it % 2 == 0 }.sum()
```

Extensions

- findViewById 사용하지 않고, layout id 직접 접근. (view instance 선언, 초기화 삭제)
  - https://www.androidhuman.com/lecture/kotlin/2016/07/25/kotlin_android_extensions/
- recyclerView, viewHolder 사용시 주의해야할 점
  - https://www.androidhuman.com/lecture/kotlin/2017/11/26/kotlin_android_extensions_on_viewholder/

참고문헌

- 커니 코틀린 github
  - https://github.com/kunny/kunny-kotlin-book
- bug & issue tracker
  - https://youtrack.jetbrains.com/issues/KT
- 우아한형제들 기술블로그
  - http://woowabros.github.io/experience/2017/07/18/introduction-to-kotlin-in-baeminfresh.html
- web 코드 테스트
  - https://try.kotlinlang.org/#/Examples/Hello,%20world!/Simplest%20version/Simplest%20version.kt
- Kotlin labda
  - https://tourspace.tistory.com/110?category=797357
  - https://tourspace.tistory.com/111
- 코틀린의 장단점
  - https://imcreator.tistory.com/113
- 코틀린 키워드 연산자
  - https://medium.com/@joongwon/kotlin-kotlin-%ED%82%A4%EC%9B%8C%EB%93%9C-%EB%B0%8F-%EC%97%B0%EC%82%B0%EC%9E%90-%ED%95%B4%EB%B6%80-1-hard-keywords-3062f5fe2d11
  - https://medium.com/@joongwon/kotlin-kotlin-%ED%82%A4%EC%9B%8C%EB%93%9C-%EB%B0%8F-%EC%97%B0%EC%82%B0%EC%9E%90-%ED%95%B4%EB%B6%80-2-soft-keywords-b3a70cc16955
  - https://medium.com/@joongwon/kotlin-kotlin-%ED%82%A4%EC%9B%8C%EB%93%9C-%EB%B0%8F-%EC%97%B0%EC%82%B0%EC%9E%90-%ED%95%B4%EB%B6%80-part-3-59ff3ed736be

- Learn Spring for Android Application Development
    - https://books.google.co.kr/books?id=y4KGDwAAQBAJ&pg=PA301&lpg=PA301&dq=@SerializedName+:%5B%7B&source=bl&ots=Bm4vW5aaM4&sig=ACfU3U0ttgPG8fofrHZPdTFWrjzQK36AeA&hl=ko&sa=X&ved=2ahUKEwi6_ZKUzbbkAhViyosBHb-KBBkQ6AEwBHoECAgQAQ#v=onepage&q=%40SerializedName%20%3A%5B%7B&f=false
- 학습 리소스
    - https://developer.android.com/kotlin/resources?hl=ko
- https://androidstudio.googleblog.com/2017/04/android-studio-24-preview-4-is-now.html
- https://developer.android.com/studio/preview/features/java8-support.html
- http://talkingkotlin.com/extensions-with-jake-wharton/
- https://youtu.be/mDpnc45WwII
- https://youtu.be/A2LukgT2mKc
- https://realm.io/news/oredev-jake-wharton-kotlin-advancing-android-dev/
- http://qiita.com/omochimetaru/items/98e015b0b694dd97f323
- http://d2.naver.com/helloworld/8725603
- https://medium.com/@goinhacker/in-android-convert-java-to-kotlin-afe532fa7151