Kotlin 언어 소개

1. 코틀린(Kotlin) 이란?

Kotlin?

- IntelliJ IDEA를 만들 JetBrains에서 2011년에 공개한 프로그래밍 언어
- JVM(Java Virtual Mechine) 기반의 언어이며, **자바와의 상호 운용이 100% 가능**
- 안드로이드, 스프링 프레임워크, 톰캣, 자바스크립트, Java EE, HTML5, iOS, 라즈베리 파이 등 개발에 사용할 수 있음
- 2017년 구글이 안드로이드 공식 언어로 채택

왜 코틀린인가?

- 간결성 : 상용구 코드의 양을 대폭적으로 줄여줌. 자바에서는 클래스를 작성할 때, getter 및 setter 등을 만들고, equals(), hashcode(), toString() 등을 재정의해 코드가 길어지지만, 코틀린에서는 한 줄에 작성 가능
- 안전성 : ‘null pointer exceptions’과 같은 에러를 방지
- 상호 운용성 : 자바와 100% 상호 운용이 가능하며, 기존의 모든 안드로이드 라이브러리 사용 가능
- 도구 친화적 : 코틀린 개발 도구는 안드로이드 스튜디오에서 완벽히 지원되므로 안드로이드 빌드 시스템과도 완벽히 호환됨
- 위와 같은 현대적인 프로그래밍 언어의 특징과 장점을 가지고 있음
- 쉽게 배울 수 있어서 인기가 증가할 것으로 예상

2. Java와 Kotlin 비교

기본 타입

- 자바
  참조(reference)와 기본(primitive) 타입 제공
  int a = 100 // 기본 타입
  Integer a = 100 // 참조 타입

- 코틀린
  내부적으로 기본 타입의 성능을 제공하면서, 참조 타입을 쉽게 사용 가능
  var a : Int = 100 // 기본과 참조 타입을 특성을 동시에

문자열 출력

- 자바
  String str = “안녕”
  System.out.println(str + “하세요”)

- 코틀린
  var str = “안녕”
  println(str + “하세요”)
  println("$str하세요")
  println("${str}하세요")

함수

- 코틀린은 파일 수준의 함수로 정의

자바	코틀린

A.java 파일
 	
public class A {
  public void a() {
  }
  public static void main(){
  }
}
	A.kt 파일
	
	a(){
	}
	
	fun main(){
	}

Null 처리

- 코틀린은 명시적 null 타입을 이용해 NullPointerException을 방지

자바	코틀린

String drink = “맥주”
drink = null // 가능
drink = drink + “ 500cc”
// NullPointerException 발생
	var drink : String ? = "맥주"
	// null 가능 타입
	drink = null
	println(drink + " 500cc")
	// null + 500cc 출력

싱글톤

- 코틀린은 싱글톤(Singleton)을 object 키워드를 사용해서 생성할 수 있음

자바	코틀린

Class Singleton {
  private static Singleton instance;
  public static getInstance() {
  if(instance == null) {
  instance = new Singleton();
  }
  return instance;
  }
}
	object Singleton
	{
	
	}

data 클래스

- data 클래스를 이용하여 상용구 코드의 양을 줄여줌
- getters, setters, equals(), hashCode(), toString(), copy() 를 자동 생성
- 예시) 코틀린
  data class Customer(val name: String, val email: String, val company: String)

람다 vs 익명클래스

- 자바 8은 객체지향 프로그래밍과 람다 표현식을 모두 지원
- 하지만, 함수의 매개변수나 변수에 함수를 정의할 수 없음
- 대신 익명 내부 클래스를 제공
- 익명 내부 클래스를 구현할 때 람다를 정의하는 함수를 나타내기 위해 이름이 있는 타입(인터페이스나 클래스)의 정의가 필요
- 코틀린 람다는 간결한 문법을 구현하며, 자바와 기능적으로 같음

자바	코틀린

public interface Runnable {
}
 	
Runnable myRunnable = () -> {}
	fun runMyrunnable(runnable : () ->)
	= {runnable}

감사합니다^^

강사 목진혁

jhmocu@gmail.com