SUNG KYUN KWAN UNIVERSITY

ARCS

Kotlin + ANTLR

PA#2 Overview

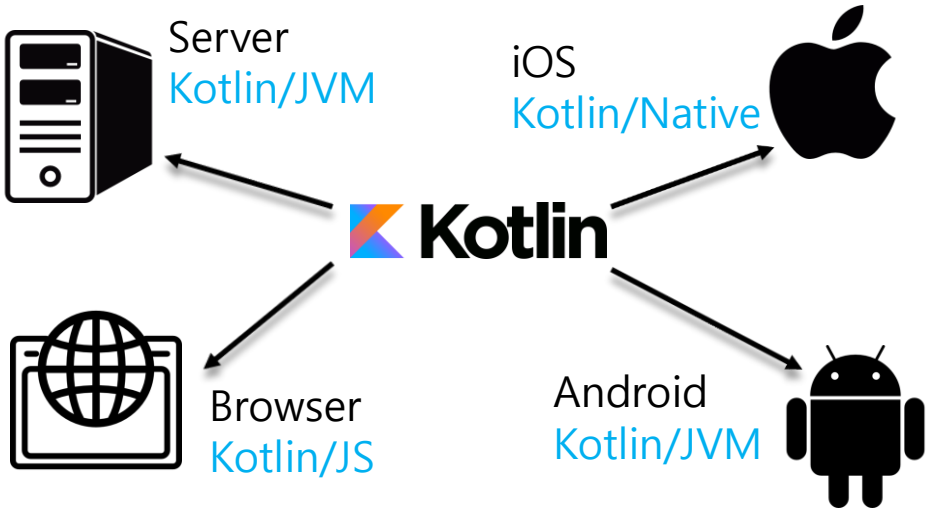
Overview: Programming Assignments

- PA #2
  - ANTLR based mini-Kotlin
  - implement Kotlin.g4
  - parsing basic syntax of Kotlin
    - https://kotlinlang.org/docs/reference/basic-syntax.html

- PA #3
  - Kotlin-to-Java (Source code-to-Source code) compiler
  - Type Inference

Kotlin

- General-purpose language
- OOP + FP
- Static typing
  - However, do not need type keywords



Kotlin Basic Syntax

- Defining functions

Kotlin	Java
1. Function with *return*	

`fun sum(a: Int, b: Int): Int {`
`  return a + b`
`}`
	`int sum(int a, int b) {`
	`  return a + b;`
	`}`

2. Function with an expression and inferred return type	
`fun sum(a: Int, b: Int) = a + b`	

Kotlin Basic Syntax

- Defining variables

Kotlin	Java
val: Read-only local variables	final variables

`val a: Int = 1`
`val b = 2`
`val c: Int`
`c = 3`
	`final int a = 1;`
	`final int b = 2;`
	`final int c;`
	`c = 3;`

var: Reassign-available variables	
`var x = 5` `x += 1`	`int x = 5;` `x += 1;`

Kotlin Basic Syntax

- Nullable values

Kotlin

```kotlin
fun StringLength(obj: Any): Int? {
    if (obj is String)
        return obj.length
    return null
}
fun main(){
    println(StringLength("String"))
    println(StringLength(123))
}
```

Java

```java
class Main{
  static Integer StringLength(Object obj){
    if (obj instanceof String)
      return ((String) obj).length();
    return null;
  }
  public static void main(String[] args) {
    System.out.println(StringLength("String"));
    System.out.println(StringLength(123));
  }
}
```

Result

6
null

Kotlin Basic Syntax

- Nested functions(methods)

Kotlin

```kotlin
fun main(){
    fun StringLength(obj: Any): Int? {
        if (obj is String)
            return obj.length
        return null
    }
    println(StringLength("String"))
    println(StringLength(123))
}
```

Result

6
null

Java

```java
class Main{
  public static void main(String[] args) {
    class Inner{
      Integer StringLength(Object obj){
        if (obj instanceof String)
          return ((String) obj).length();
        return null;
      }
    }
    System.out.println(new Inner().
                                StringLength("String"));
    System.out.println(new Inner().
                                StringLength(123));
  }
}
```

Kotlin Basic Syntax

- Iterating over a range

Kotlin

```kotlin
fun main(){
    for (x in 1..5) {
        print(x)
    }
}
```

Result
12345

```kotlin
fun main(){
    for (x in 1..10 step 2) {
        print(x)
    }
    println()
    for (x in 9 down 0 step 3) {
        print(x)
    }
}
```

Result
13579
9630

Kotlin Basic Syntax

- Using collections

Kotlin

```
fun main(){
    val items = listOf("apple", "banana",
                                    "kiwifruit")
    for (item in items) {
        println(item)
    }
}
```

Result

apple
banana
kiwifruit

```
fun main(){
    val items = setOf("apple", "banana",
                                    "kiwifruit")
    when {
        "orange" in items -> println("juicy")
        "apple" in items -> println("apple is
                                    fine too")
    }
}
```

Result

apple is fine too

PA#2 Kotlin Parser

- Submit a Kotlin g4 file
  - Kotlin.g4
  - Test your g4 file with kotlin code

```
$ make
$ ./kt_parser input.kt
```

- Should handle all Kotlin code in basic syntax page, https://kotlinlang.org/docs/reference/basic-syntax.html
  - Including surrounding code (code appears when click +)

```
fun sum(a: Int, b: Int): Int {
    return a + b
}
```

Target platform: JVM Running on kotlin v. 1.3.50

```
fun sum(a: Int, b: Int): Int {
    return a + b
}

fun main() {
    print("sum of 3 and 5 is ")
    println(sum(3, 5))
}
```

Target platform: JVM Running on kotlin v. 1.3.50

Note!

- Kotlin Basic Syntax
  - https://kotlinlang.org/docs/reference/basic-syntax.html
  - skip:
    - String template
      - but you **need to handle** normal string constants: "apple is fruit"
    - Nested comments
      - but you **need to handle** non-nested comments:
      - /* multi-line comment1
        * multi-line comment2
        */
        // rest of the line comment3
    - When expression
    - Lambda expressions in Collections
  - Equality
    - Use != for ≠