Server-Side Kotlinで必要なJavaの知識

2021年5月12日

竹端 尚人

Server-Side Kotlinで必要なJavaの知識

2021年5月12日

竹端 尚人

自己紹介

概要

竹端 尚人
フリーランスエンジニア

Twitter: @n_takehata
職種:バックエンドエンジニア
好きな言語:Kotlin

- Server-Side Kotlin、Java、etc…
- (少し前まで)スマートフォンゲーム開発
- 昨年12月からフリーランスに

登壇、執筆など

- CEDEC2018、2019登壇
- Software Design 2019年2〜4月号で短期連載
- **書籍「Kotlin サーバーサイドプログラミング実践開発」**を2021年4月に発売

書籍の一節

またJavaとの相互互換という特徴があり、世の中にすでに多くあるJavaの資産を活用するこ ともでき、モダンかつ資産も豊富にあるという素晴らしい言語です。

しかし、その特徴ゆえ「Javaがわかる人じゃないと難しいんじゃないか」と思われてしまうことも多い です。

Kotlinを学ぶのにJavaの知識はどれだけ必要なのか？

アジェンダ

1.Javaの知識は絶対必要なのか？

2.よく使う知識

3.まとめ

アジェンダ

1.Javaの知識は絶対必要なのか？

2.よく使う知識

3.まとめ

1.Javaの知識は絶対必要なのか？

結論

Java自体の知識はなくても学べる

書籍では「Javaとの相互互換」について書いた3章以外は、Javaのコードは出てきません

- Javaの知識があることを前提としなくても、Kotlinを学ぶことに問題はない
- Java経験がない場合の学習コストは、他の言語に比べて特別高いということはない
- ただし、知っていると学ぶのが圧倒的に楽になる

知っていると有利な点

- 構文や機能で似たものを有している
- フレームワークやライブラリはJava製のものを活用することも多い
- 世の中にある既存のJavaの資産を活用できる

Javaを知っていると楽だが、なくても大丈夫

(KotlinやるためにJavaを先に学ばなくては・・・ということはない)

アジェンダ

1.Javaの知識は絶対必要なのか？

2.よく使う知識

3.まとめ

2.よく使う知識

- オブジェクト思考の知識
- フレームワーク、ライブラリの知識
- Javaの言語自体の知識

オブジェクト思考の知識

- クラス、インターフェースなどの基本的なオブジェクト思考の知識はあった方が学びやすい
- C#など他のオブジェクト思考言語をやっていると理解はしやすい

(似たような話で関数型の知識なども)

広義の意味でJavaの知識

フレームワーク、ライブラリの知識

- Spring Bootが無難な選択肢として上がる
- 各種ORMもJava製が使われることが多い(Exposedがなかなか正式版にならないので)
- テスティングフレームワークはJUnitが多い

例えばSpring Bootを知っていた場合

以下のようなメリットがある

- Javaと同じようなアーキテクチャのアプリケーションなので実装が理解しやすい
- フレームワークの機能を把握している
- 運用ノウハウがある

コード例

Spring Bootのアノテーション

```
@RestController
@RequestMapping("greeter")
class GreeterController(
    private val greeter: Greeter
) {
    @GetMapping("/hello")
    fun hello(@RequestParam("name") name: String): HelloResponse {
        return HelloResponse("Hello ${name}")
    }
}
```

Spring Securityのインターフェース実装

```
class MyAuthenticationFailureHandler : AuthenticationFailureHandler {
    override fun onAuthenticationFailure(
        request: HttpServletRequest,
        response: HttpServletResponse,
        exception: AuthenticationException
    ) {
        response.status = HttpServletResponse.SC_UNAUTHORIZED
    }
}
```

Javaのライブラリを使う場合

KotlinからJavaのライブラリ呼び出し

```
fun main() {
    val uuid = UUID.randomUUID()
    println(uuid.toString())

    val now = LocalDateTime.now()
    println(now)
}
```

いずれもJavaの知識は使うが、Javaのコードは書かない

Javaの言語自体の知識

- フレームワークでJavaのコードを生成する部分
- Javaとの互換のための機能を使う時
- 既存のJavaの資産を活用したい場合

フレームワークでJavaのコードを生成する部分

- gRPC、各種ORMなどで自動生成するコードでKotlin対応してないもの
- 初めて扱う際などは読んでみた方が良い

gRPCの実装

```
@RestController
class GreeterClientController {
    @GetMapping("/greeter/hello/{name}")
    fun hello(@PathVariable name: String): String = runBlocking {
        val channel = ManagedChannelBuilder.forAddress(HOST, PORT)
            .usePlaintext()
            .build()

        val request = HelloRequest.newBuilder().setName(name).build()
        val stub = GreeterGrpcKt.GreeterCoroutineStub(channel)

        val response = async { stub.hello(request) }
        response.await().text
    }
}
```

生成されたコード

```
public final class HelloRequest extends
    com.google.protobuf.GeneratedMessageV3 implements
    // @@protoc_insertion_point(message_implements:example.greeter.HelloRequest)
    HelloRequestOrBuilder {

// . . .

  public static Builder newBuilder() {
    return DEFAULT_INSTANCE.toBuilder();
  }

// . . .

    public Builder setName(
        java.lang.String value) {
      if (value == null) {
    throw new NullPointerException();

// . . .
```

自動生成なので書くことはないが、
必要に応じて読むことはある

Javaとの互換のための機能を使う時

- @JvmStaticなどの各種アノテーション
- SAM変換
- etc…

@JvmStatic

```
class CompanyConstants {
    companion object {
        @JvmStatic
        val maxEmployeeCount = 100
    }
}
```

アノテーションがなかった場合

```
public static void main(String[] args) {
    System.out.println(CompanyConstants.Companion.getMaxEmployeeCount());
}
```

Javaからどう呼ばれるかの理解が必要

SAM変換

```kotlin
fun main() {
    val function = CalcJava { num1, num2 -> num1 + num2 }
    println(function.calc(1, 3))
}
```

```java
@FunctionalInterface
public interface CalcJava {
    Integer calc(Integer num1, Integer num2);
}
```

Javaの何を呼んでいるかの理解が必要
(今はKotlin同士でも使えるようになったのでJava独自のものではないが)

Javaからどう呼ばれているか、Javaをどう呼んでいるかを
理解する必要はある

既存のJavaの資産を活用したい場合

- Javaにしかないオープンソースを使いたい場合
- 組織でJavaの資産を持っていて活用したい場合

この場合は多分組織内に有識者がいるので問題ない

アジェンダ

1.Javaの知識は絶対必要なのか？

2.よく使う知識

3.まとめ

3.まとめ

- Javaを知らなくてもServer-Side Kotlinは始められる
- ただし、Javaを知っていると学習コストは圧倒的に下がる
- Javaを書く必要はないが、多少読んだり理解する必要がある場面は存在する(でもそんなに怖がらなくていいレベル)