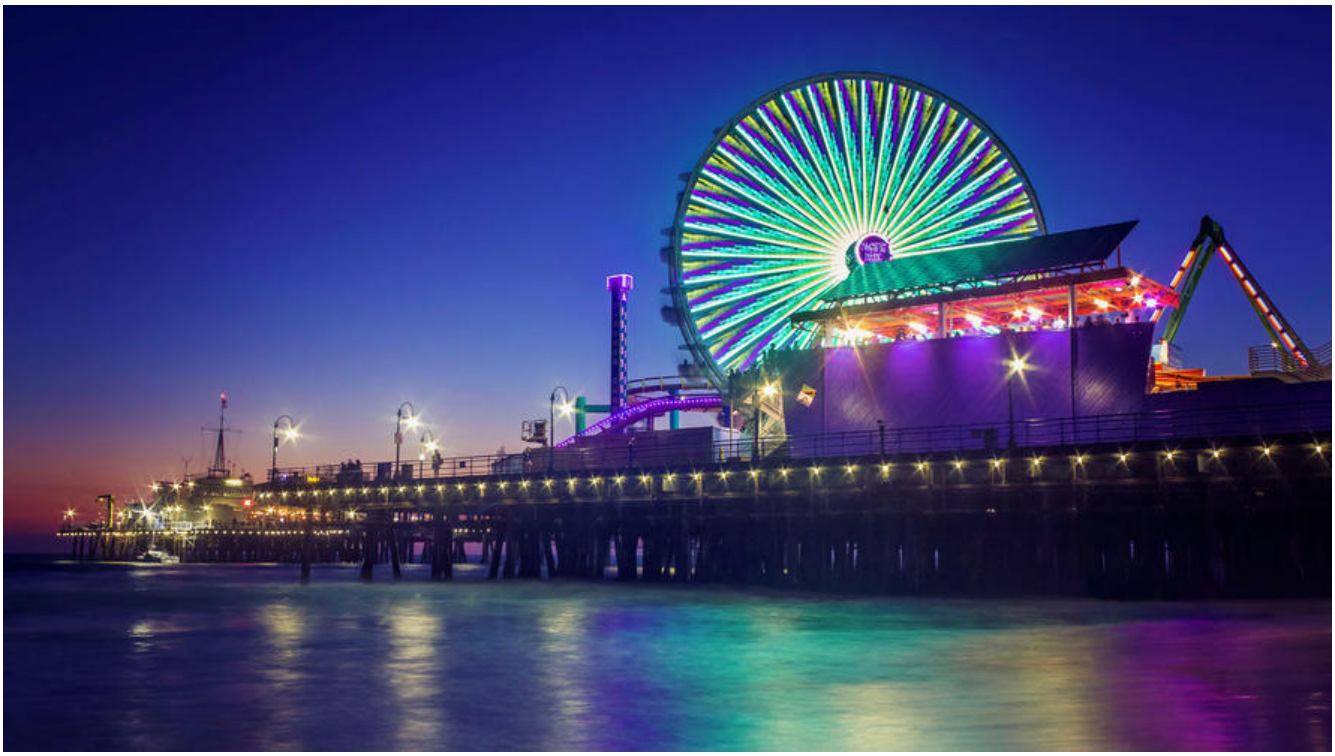
链滴

kotlin 1.3.70 中 kotlinx.serialization 升级到 0.20.0

作者：lizhongyue248
原文链接：https://ld246.com/article/1583919535787
来源网站：链滴



月初的时候 Kotlin 1.3.70 正式发布了，这不得不说是一个非常好的消息，带来的不仅是标准库新的能，同时让我们写起来也更加舒服，现在对 .gradle.kts 也更加友好了，具体可以查看它的发布文档随之而然的，就是相应的生态进行更新了，不得不提的就是 kotlinx.serialization 这个库了。他为我序列化提供了极大的方便，而他与 1.3.70 对应的版本是 0.20.0。

如果你还在使用 v0.14.0 版本，可能会报以下错误：

```
java.lang.NoSuchMethodError: No direct method (ILkotlinx/serialization/SerializationConstruc
orMarker;)V in class Lcom/xxxx/common/core/Model; or its super classes (declaration of 'com
xxxx.common.core.Model' appears in /data/app/com.xxxx.demo-PV-n86-hzEl-eyc8UqbACQ=
/base.apk!classes7.dex)
10-11 15:30:10.048 E/AndroidRuntime(25760): at com.xxxx.reg.data.User.(Unknown Source:9)
10-11 15:30:10.048 E/AndroidRuntime(25760): at com.xxxx.reg.data.User$$serializer.deserializ
(Unknown Source:624)
10-11 15:30:10.048 E/AndroidRuntime(25760): at com.xxxx.reg.data.User$$serializer.deserializ
(User.kt:17)
10-11 15:30:10.048 E/AndroidRuntime(25760): at kotlinx.serialization.json.internal.Polymorphi
Kt.decodeSerializableValuePolymorphic(Polymorphic.kt:33)
```

升级到 v0.20.0 就可以了。当然在这个版本中发生了不小的改变，将它过期的 api 总结一下。

Json.nonstric

默认情况下，他开启 nonstric 模式的，也就是解析的 key 必须完全符合实体类的要求，不能够多出他的字段。但是很多时候我们不希望这样，在以前的版本中是这样配置的：

```
private val json = Json(JsonConfiguration(strictMode = false))
```

但是在 0.20.0 中，已经不能够这样配置了，参见 github

- strictMode 分割为 ignoreUnknownKeys、isLenient和 serializeSpecialFloatingPointValues，

- unquoted 重命名为 unquotedPrint

在新的版本中我们需要如下使用：

```
private val json = Json(
  JsonConfiguration.Default.copy(
    ignoreUnknownKeys = true,
    isLenient = true,
    serializeSpecialFloatingPointValues = true,
    useArrayPolymorphism = true
  )
)
```

SerialDescriptor

我们在对某些数据结构进行序列化和反序列化的时候应该写过如下的代码：

```
@Serializer(forClass = LocalDateTime::class)
class LocalDateTimeSerializer : KSerializer<LocalDateTime> {

  override val descriptor: SerialDescriptor = StringDescriptor
  private val formatter: DateTimeFormatter =
    DateTimeFormatter.ofPattern("yyyy-MM-dd HH:mm:ss").withLocale(Locale.SIMPLIFIED_CH
NESE)

  override fun deserialize(decoder: Decoder): LocalDateTime =
    LocalDateTime.parse(decoder.decodeString(), formatter)

  override fun serialize(encoder: Encoder, obj: LocalDateTime) {
    encoder.encodeString(formatter.format(obj))
  }
}
```

通过实现 serialize 和 descriptor 方法就可以完成自定义的序列化。但是在 0.20.0 版本中 StringDescr ptor 被标记为过时了，相应的以下基本数据类型的都被标记了过时：

- IntDescriptor
- UnitDescriptor
- BooleanDescriptor
- ByteDescriptor
- ShortDescriptor
- LongDescriptor
- FloatDescriptor
- DoubleDescriptor
- CharDescriptor
- StringDescriptor

源码中的注解如下：

```
@Deprecated(message = message,
    replaceWith = ReplaceWith("PrimitiveDescriptor(\"yourSerializerUniqueName\", PrimitiveKi
d.STRING)"))
object StringDescriptor : Migration()
```

我们需要使用 PrimitiveDescriptor 来进行替代，参考github：

```
override val descriptor: SerialDescriptor = PrimitiveDescriptor("LocalDateTimeTz", PrimitiveKi
d.STRING)
```

需要注意的是这个字符串的 key 需要是唯一的。不然的话可能会造成他无法找到等问题。

bultins package

另外，一些功能已移至 bultins package 包。

例如：

```
import kotlinx.serialization.list
User.serializer().list
// 替换为
import kotlinx.serialization.builtins.list
User.serializer().list

import kotlinx.serialization.internal.StringSerializer
StringSerializer()
// 替换为
import kotlinx.serialization.builtins.serializer
String.serializer()
```