Kotlin 1.4 Online Event

kotlinx.serialization 1.0

Leonid Startsev

@sandwwraith

October 13, 2020

Kotlin multiplatform / multi-format reflectionless serialization

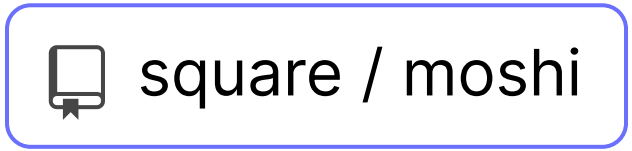
Kotlin multiplatform / multi-format reflectionless serialization

Things we'll cover today:

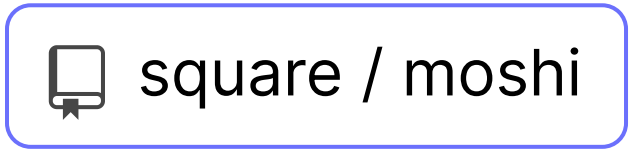
1. Introduction to kotlinx.serialization
2. Serialization APIs
3. New features in 1.0
4. Future plans
5. Setup and/or migration

A brief introduction

Why do we need yet another json parser?

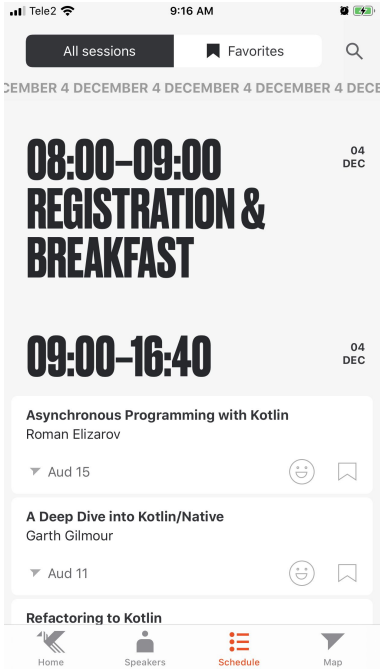


Why do we need yet another json parser?

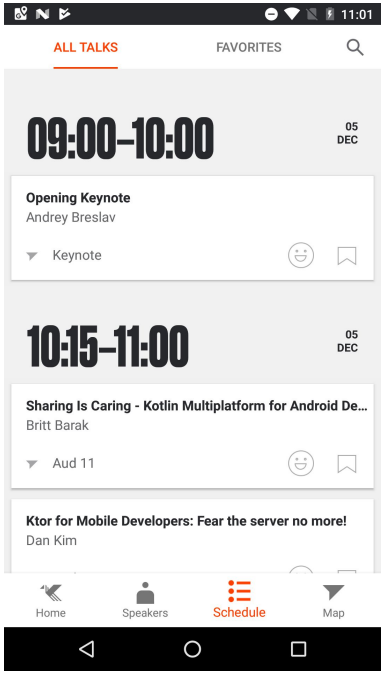


These are all Java libraries

Multiplatform




```kotlin
@Serializable
data class Session(
    val id: String,
    val isServiceSession: Boolean,
    val isPlenumSession: Boolean,
    val questionAnswers:
List<QuestionAnswer>,
    val speakers: List<String>,
    @SerialName("description")
    val descriptionText: String?,
    val startsAt: String?,
    val title: String,
    val endsAt: String?,
    val categoryItems: List<Int>,
    val roomId: Int?
)
```



Kotlin-oriented

```
@Serializable
data class Project(
    val name: String,
    val language: String = "Kotlin"
)

const val inputString =
"""{"name":"kotlinx.serialization"}"""
```

Kotlin-oriented

```
@Serializable
data class Project(
    val name: String,
    val language: String = "Kotlin"
)

println(Gson().fromJson(inputString, Project::class.java))
// Project(name=kotlinx.serialization, language=null)
```

Null for a non-nullable type, definitely an error

Kotlin-oriented

```
@Serializable
data class Project(
    val name: String,
    val language: String = "Kotlin"
)

println(Json.decodeFromString<Project>(inputString))
// Project(name=kotlinx.serialization, language=Kotlin)
```

OK!

Explicit & compile-time safe

```
@Serializable
data class Project(
    val name: String,
    val language: String = "Kotlin"
)

// not @Serializable
data class User(val userName: String)
```

Explicit & compile-time safe

```
@Serializable
data class Project(
    val name: String,
    val owner: User,
    val language: String = "Kotlin"
)

// not @Serializable
data class User(val userName: String)
```

Explicit & compile-time safe

```
@Serializable
data class Project(
    val name: String,
    val owner: User,
    val language: String = "Kotlin"
)

// not @Serializable
data class User(val userName: String)
```

Serializer for type User has not been found

Explicit & concise

```
val projectsList = Gson().fromJson<List<Project>>(
    inputStringList,
    List::class.java
)

println(projectsList.first()::class.java)
// class com.google.gson.internal.LinkedTreeMap
```

Explicit & concise

```
val projectsList = Gson().fromJson<List<Project>>(
    inputStringList,
    (object: TypeToken<List<Project>>() {}).type
)

println(projectsList.first()::class.java)
// class kotlinx.serialization.formats.json.Project
```

Explicit & concise

```
val projectsList =
 Json.decodeFromString<List<Project>>(inputStringList)


println(projectsList.first()::class.java)
// class kotlinx.serialization.formats.json.Project
```

Explicit & concise

```
public inline fun <reified T> typeOf(): KType
```

Explicit & concise

```
public inline fun <reified T> typeOf(): KType
```

```
val type = typeOf<Box<List<StringData>>>()
println(type)
```

Explicit & concise

```
public inline fun <reified T> typeOf(): KType
```

```
val type = typeOf<Box<List<StringData>>>()
println(type)
```

```
"kotlinx.serialization.Box<
  kotlin.collections.List<
    kotlinx.serialization.StringData>>"
```

Explicit & concise

```
public inline fun <reified T> serializer(): KSerializer<T>
```

Explicit & concise

```
public inline fun <reified T> serializer(): KSerializer<T>
```

```
val serial = serializer<Box<List<StringData>>>()
```

Explicit & concise

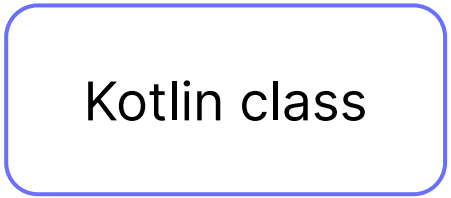
```
public inline fun <reified T> serializer(): KSerializer<T>

val serial = serializer<Box<List<StringData>>>()
// these two are equivalent
val serial = Box.serializer(StringData.serializer().list)

val box = Json.decodeFromString(serial, input)
```

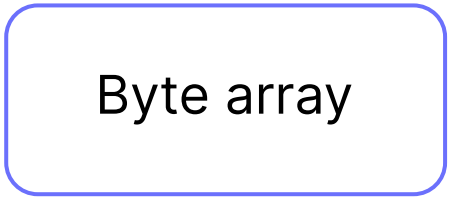
Design of kotlinx.serialization
@ KotlinConf, 2019
by Leonid Startsev

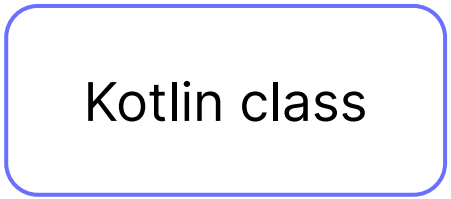
www.youtube.com

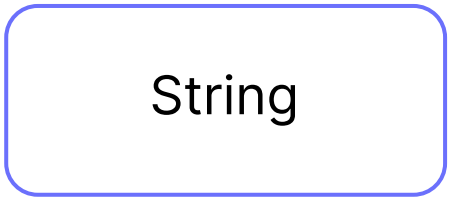
Many formats with similar APIs

- JSON
- CBOR
- Protocol buffers


Kotlin class


Byte array


Kotlin class

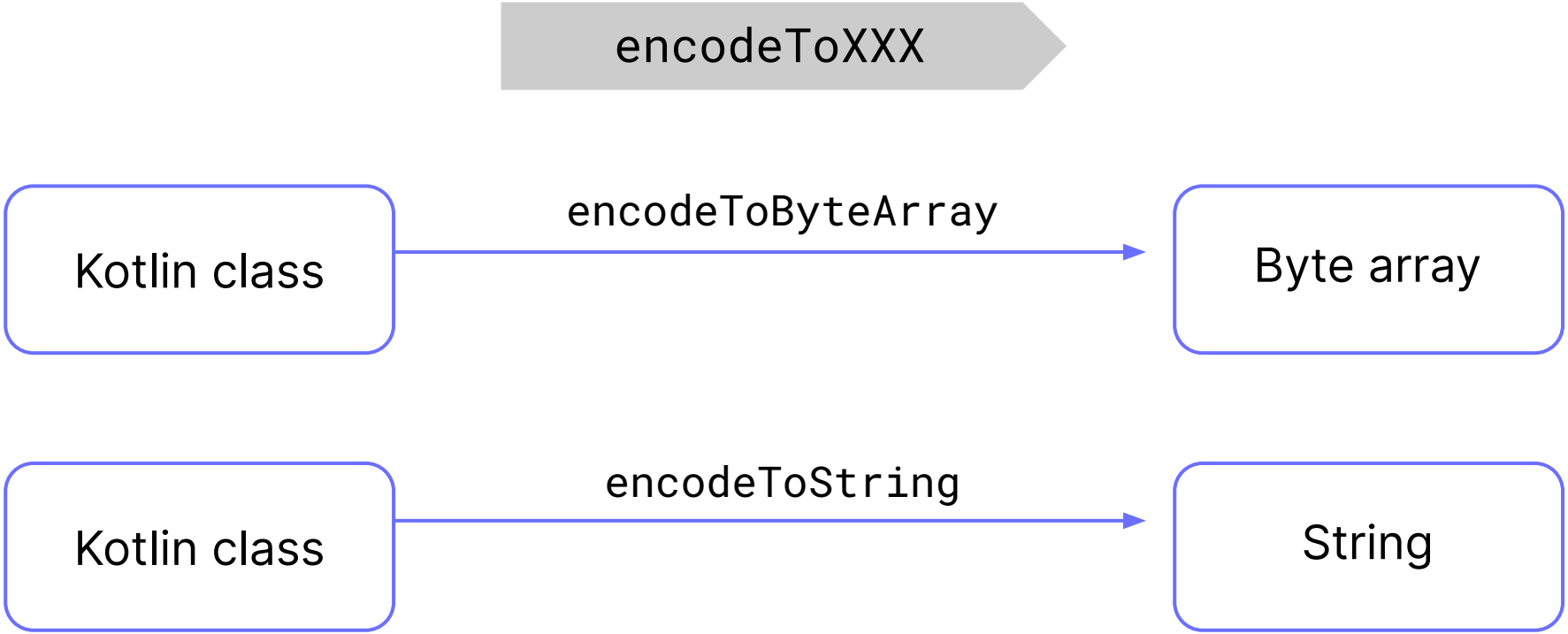

String

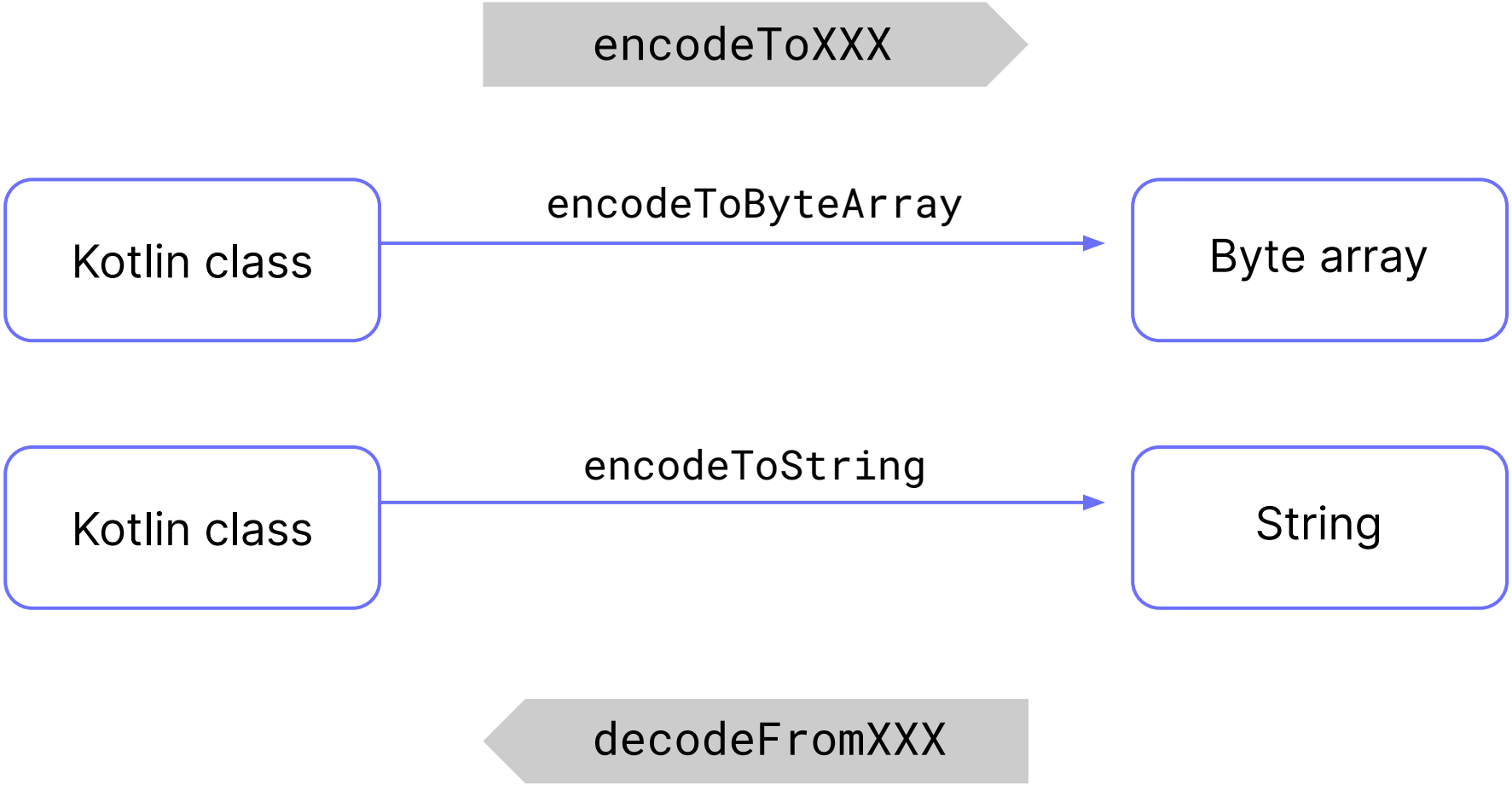
Kotlin class

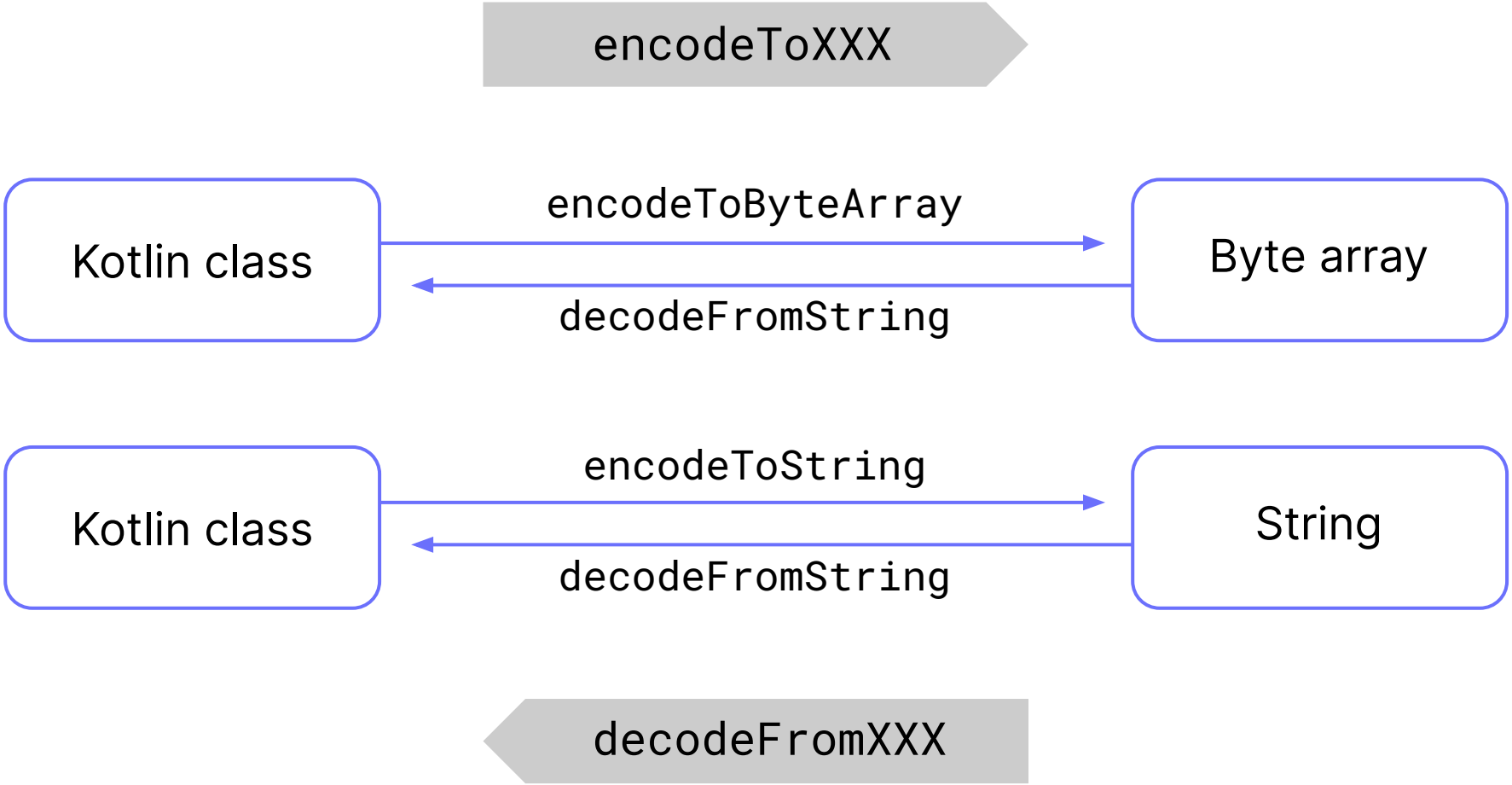
Byte array

Kotlin class

String


encodeToXXX
Kotlin class
encodeToByteArray
Byte array
Kotlin class
encodeToString
String


encodeToXXX
Kotlin class
encodeToByteArray
Byte array
Kotlin class
encodeToString
String
decodeFromXXX


encodeToXXX
Kotlin class
encodeToByteArray
decodeFromString
Byte array
Kotlin class
encodeToString
decodeFromString
String
decodeFromXXX

Future-proof configuration

Future-proof configuration

```
public data class JsonConfiguration(
    ...
    val coerceInputValues: Boolean = false,
    val useArrayPolymorphism: Boolean = false,
    ...
)
```

Future-proof configuration

```
public data class JsonConfiguration(
    ...
    val coerceInputValues: Boolean = false,
    val coolNewJsonFeature: Boolean = false,
    val useArrayPolymorphism: Boolean = false,
    ...
)
```

Future-proof configuration

modified: runtime/api/kotlinx-serialization-core.api

```
@ kotlinx-serialization-core.api:1478 @ public final class kotlinx/serialization/json/JsonBuilder {
 public class kotlinx/serialization/json/JsonConfiguration {
    public static final field Companion Lkotlinx/serialization/json/JsonConfiguration$Companion;
    public fun <init> ()V
-   public fun <init> (ZZZZZZZLjava/lang/String;ZZLjava/lang/String;)V
-   public synthetic fun <init> (ZZZZZZZLjava/lang/String;ZZLjava/lang/String;ILkotlin/jvm/internal/D
-   public final fun copy (ZZZZZZZLjava/lang/String;ZZLjava/lang/String;)Lkotlinx/serialization/json/
-   public static synthetic fun copy$default (Lkotlinx/serialization/json/JsonConfiguration;ZZZZZZZLj
inx/serialization/json/JsonConfiguration;
+   public fun <init> (ZZZZZZZLjava/lang/String;ZZZLjava/lang/String;)V
+   public synthetic fun <init> (ZZZZZZZLjava/lang/String;ZZZLjava/lang/String;ILkotlin/jvm/internal/
+   public final fun copy (ZZZZZZZLjava/lang/String;ZZZLjava/lang/String;)Lkotlinx/serialization/json
+   public static synthetic fun copy$default (Lkotlinx/serialization/json/JsonConfiguration;ZZZZZZZLj
linx/serialization/json/JsonConfiguration;
+   public final fun getCoolNewJsonFeature ()Z
    public static final fun getDefault ()Lkotlinx/serialization/json/SubtypeToDetectDefault;
    public static final fun getStable ()Lkotlinx/serialization/json/SubtypeToDetectStable;
 }
```

Exception in thread "main" java.lang.NoSuchMethodError: JsonConfiguration.<init>

Public API Challenges in Kotlin
by Jake Wharton

www.jakewharton.com

Configuring is easier

```
Json(JsonConfiguration(ignoreUnknownKeys = true))
```

Configuring is easier

```
Json(JsonConfiguration(ignoreUnknownKeys = true))
```



```
Json {
    ignoreUnknownKeys = true
}
```

Copying is easier

```
val myConfig = JsonConfiguration.Default.copy(encodeDefaults = false)
val myJson = Json(myConfig)
val myLenientJson = Json(myConfig.copy(isLenient = true))
```

Copying is easier

```
val myConfig = JsonConfiguration.Default.copy(encodeDefaults = false)
val myJson = Json(myConfig)
val myLenientJson = Json(myConfig.copy(isLenient = true))
```



```
val myJson = Json { encodeDefaults = false }
val myLenientJson = Json(myJson) { isLenient = true}
```

1.0 release APIs

Stable features

- Serializing and deserializing classes from/to Strings
- @Serializable and serialization-related annotations
- Polymorphic serialization
- JSON tree API (including custom serialization)
- Simple format-agnostic custom serializers

Stable features

- Serializing and deserializing classes from/to Strings
- @Serializable and serialization-related annotations
- Polymorphic serialization
- JSON tree API (including custom serialization)
- Simple format-agnostic custom serializers

TL;DR Serialization and deserialization of `@Serializable` classes is stable and binary backwards compatible. New compiler plugins will support old library versions down to 1.0.

Advanced usage patterns are experimental

- Custom serial formats
- Schema introspection
- CBOR, Protobuf, HOCON, and Properties
- @ExperimentalSerializationApi

Advanced usage patterns are experimental

- Custom serial formats
- Schema introspection
- CBOR, Protobuf, HOCON, and Properties
- @ExperimentalSerializationApi

TL;DR API for writing format-agnostic serializers, schema introspection, and implementing custom serial formats could be changed in the future. If they are, migration aids will be provided.

Internal functionality

- kotlinx.serialization.internal.* or @InternalSerializationApi
  - If it solves your problem, it's okay. But it is not guaranteed to be compatible between releases
  - Please tell us about your use case so we can provide a stable API.

1.0 features

More-flexible deserialization

```
val json = Json { coerceInputValues = true }
```

More-flexible deserialization

```
val json = Json { coerceInputValues = true }

@Serializable
data class Project(
    val name: String,
    val language: String = "Kotlin"
)
```

More-flexible deserialization

```
val json = Json { coerceInputValues = true }

@Serializable
data class Project(
    val name: String,
    val language: String = "Kotlin"
)

println(json.decodeFromString<Project>(
  """{"name":"Ktor","language":null}"""
))
```

More-flexible deserialization

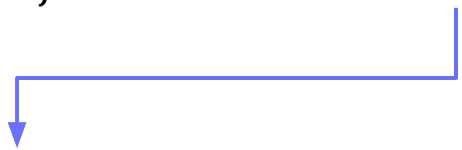
```
val json = Json { coerceInputValues = true }

@Serializable
data class Project(
    val name: String,
    val language: String = "Kotlin"
)

println(json.decodeFromString<Project>(
  """{"name":"Ktor","language":null}"""
))

=> Project(name=Ktor, language=Kotlin)
```

Polymorphic deserialization

```
@Serializable
sealed class Project {
    abstract val name: String
    abstract val language: String
}
```

Polymorphic deserialization

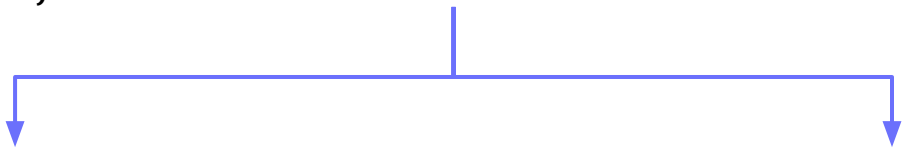
```
@Serializable
sealed class Project {
    abstract val name: String
    abstract val language: String
}
```



```
@Serializable
class OwnedProject(
    override val name: String,
    override val language: String,
    val owner: String
): Project
```

Polymorphic deserialization

```
@Serializable
sealed class Project {
    abstract val name: String
    abstract val language: String
}
```



```
@Serializable
class OwnedProject(
    override val name: String,
    override val language: String,
    val owner: String
): Project
```

```
@Serializable
class StarredProject(
    override val name: String,
    override val language: String,
    val stars: Int
): Project
```

Polymorphic deserialization

```
Json.decodeFromString<Project>(
    """{"type":"Owned","name":"Kotlin","language":"Kotlin","owner":"JetBrains"}"""
)
=> OwnedProject(name=Kotlin, language=Kotlin, owner=JetBrains)
```

Polymorphic deserialization

```
Json.decodeFromString<Project>(
    """{"type":"Owned","name":"Kotlin","language":"Kotlin","owner":"JetBrains"}"""
)
=> OwnedProject(name=Kotlin, language=Kotlin, owner=JetBrains)


Json.decodeFromString<Project>(
    """{"type":"Starred","name":"kotlinx.serialization","language":"Kotlin",
"stars":2200}"""
)
=> StarredProject(name=kotlinx.serialization, language=Kotlin, stars=2200)
```

Polymorphic deserialization

```
Json.decodeFromString<Project>(
    """{"type":"Forked","name":"Kotlin","language":"Kotlin","forks":4100}"""
)
=> Polymorphic serializer was not found for class discriminator 'Forked'
```

- Third-party party APIs you cannot control
- Independent updates

More-flexible polymorphic deserialization

```
val responseModule = SerializersModule {

}
```

More-flexible polymorphic deserialization

```
val responseModule = SerializersModule {
    polymorphic(Project::class) {

    }
}
```

More-flexible polymorphic deserialization

```
val responseModule = SerializersModule {
    polymorphic(Project::class) {
        default { className ->
            if (className != null) DefaultProject.serializer()
            else null
        }
    }
}

=> DefaultProject(name=Kotlin, language=Kotlin)
```

Future plans

IO streaming

- JVM-only API for java.io.* will be provided in 1.1.
- Integration with kotlinx-io will be provided when it’s ready.

Inline classes

Work in progress

How to set up the library

Step 1 — Compiler plugin

```
// Kotlin DSL
plugins {
    kotlin("jvm") version "1.4.0"
    kotlin("plugin.serialization") version "1.4.0"
}
```

Step 1 — Compiler plugin

```
// Kotlin DSL
plugins {
    kotlin("jvm") version "1.4.0"
    kotlin("plugin.serialization") version "1.4.0"
}


// Groovy DSL
plugins {
    id 'org.jetbrains.kotlin.multiplatform' version '1.4.0'
    id 'org.jetbrains.kotlin.plugin.serialization' version '1.4.0'
}
```

Step 2 — Runtime library

```
repositories {
    jcenter()
}

dependencies {
    implementation(
      "org.jetbrains.kotlinx:kotlinx-serialization-core:1.0.0"
    )
}
```

How to migrate

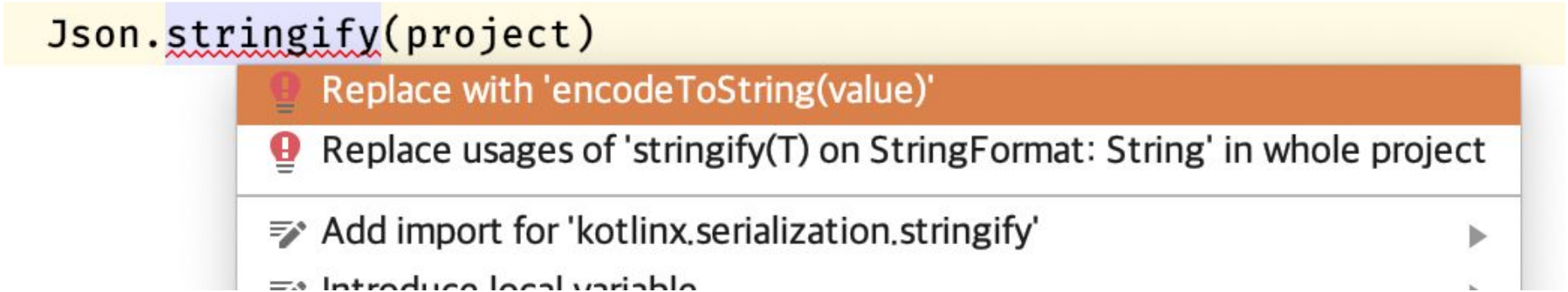
Step 0

```
Json.stringify(project)
```

[DEPRECATION_ERROR] Using 'stringify(T): String' is an error. This method was renamed to encodeToString during serialization 1.0 stabilization

Replace with 'encodeToString(value)' ⌥⇧⏎ More actions... ⌥⏎

Step 1 — Alt+Enter



Step 2 — Replace with

```
Json.encodeToString(project)
```

It’s dangerous to go it alone, take this

```
import kotlinx.serialization.*
import kotlinx.serialization.builtins.*
import kotlinx.serialization.json.*
import kotlinx.serialization.modules.*
```

Visit https://github.com/Kotlin/kotlinx.serialization
for big migration guide and changelog

Thanks!
Have a nice Kotlin

github.com/Kotlin/
kotlinx.serialization

@sandwwraith