HTL Villach
Future Inside

Abteilung für Informatik

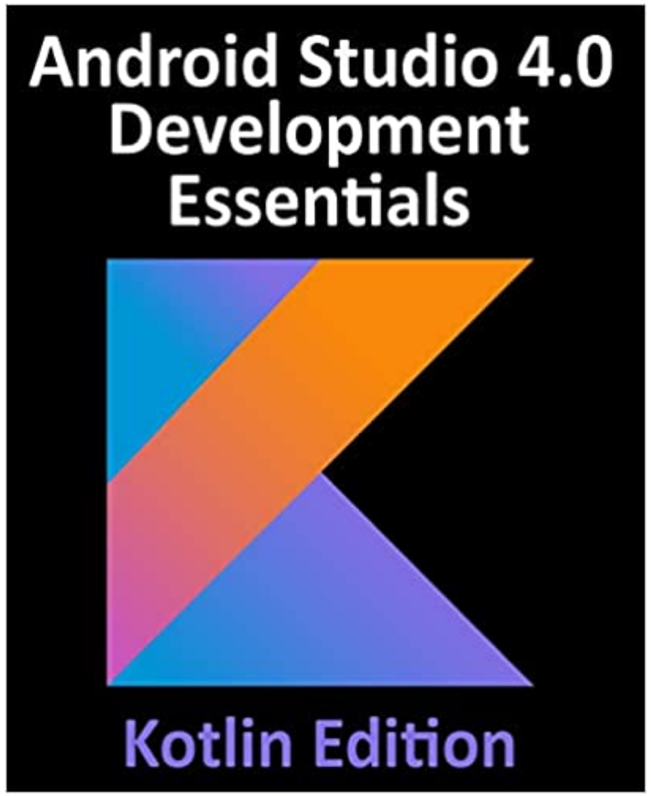
Introduction into Kotlin

software inside

EDK Feb 2021 POS

based on

Neil Smyth
2020, Payload Media Inc.

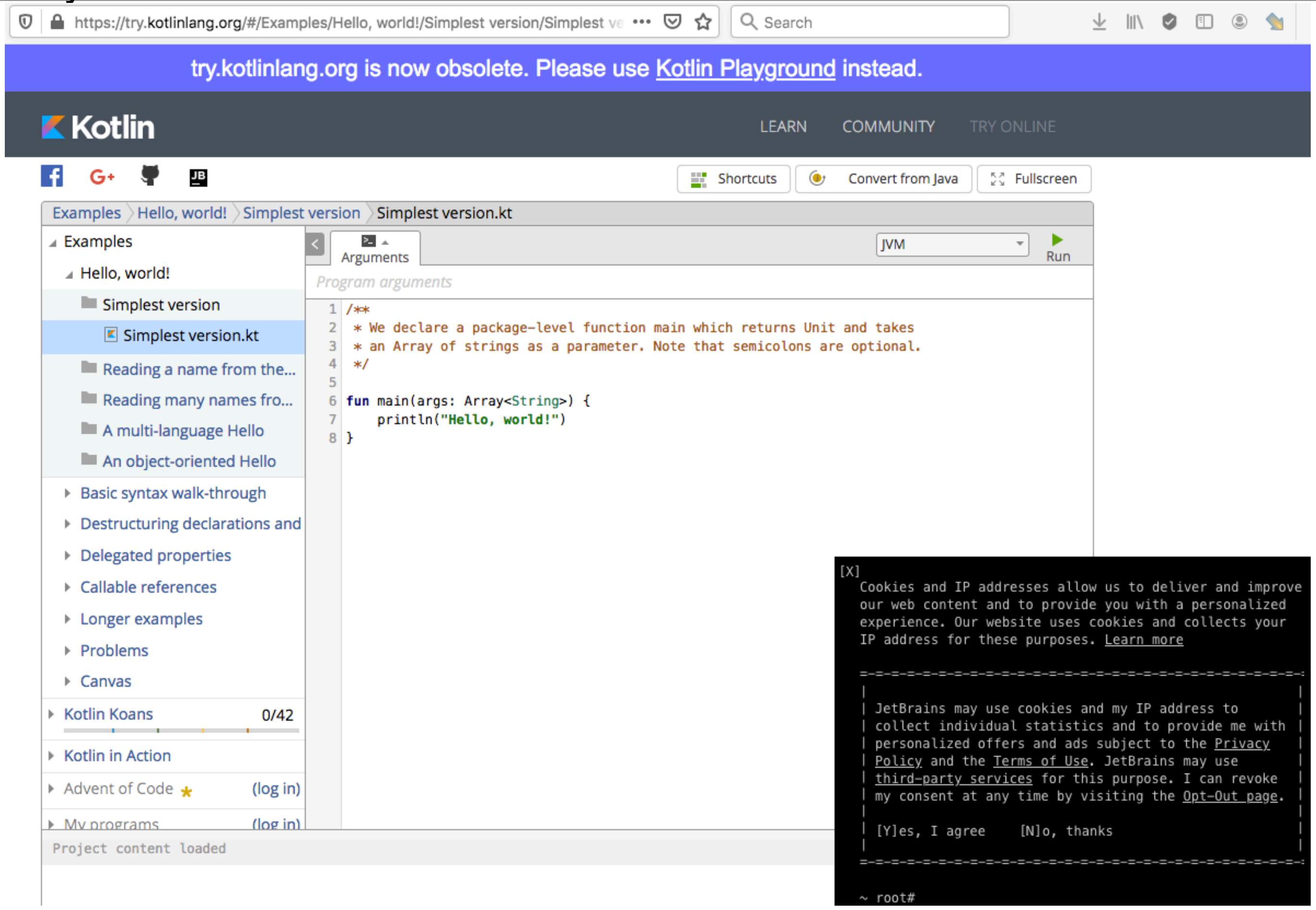


History on Kotlin which ..

- is a *f-oo ;-)* programming language (Kotlin-first since Google I/O in 2019 for Android mobile development)
- created by JetBrains https://www.jetbrains.com/ at Prague (IntelliJ, AndroidStudio)
- named after an island located in the Baltic Sea
- to make code concise (kurz & prägnant), easier to understand and safer than others
- rather than to re-invents the wheel, integrates with and work alongside Java
- generates byte code, uses JVM (like Java, Groovy, Scala, ..) and hence can apply Java frameworks seamlessly and vice versa, Java can call Kotlin code
- is a transpile language, ie source-to-source compiler to Java, JavaScript, ..

Try experimenting with Kotlin

https://try.kotl.in



```kotlin
/**
 * We declare a package-level function main which returns Unit and takes
 * an Array of strings as a parameter. Note that semicolons are optional.
 */

fun main(args: Array<String>) {
    println("Hello, world!")
}
```

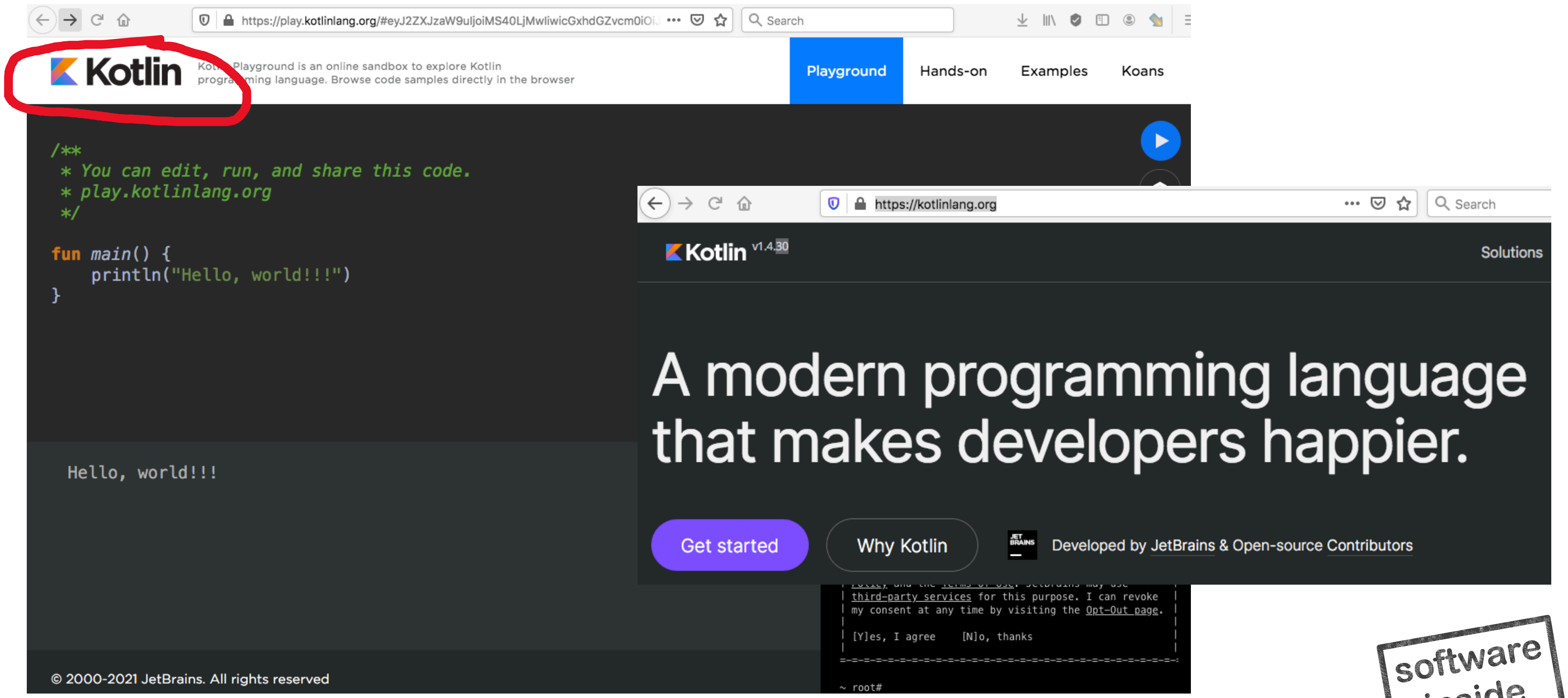
```
[X]
  Cookies and IP addresses allow us to deliver and improve
  our web content and to provide you with a personalized
  experience. Our website uses cookies and collects your
  IP address for these purposes. Learn more

  =-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-:
  |                                                        |
  | JetBrains may use cookies and my IP address to         |
  | collect individual statistics and to provide me with   |
  | personalized offers and ads subject to the Privacy     |
  | Policy and the Terms of Use. JetBrains may use         |
  | third-party services for this purpose. I can revoke    |
  | my consent at any time by visiting the Opt-Out page.   |
  |                                                        |
  | [Y]es, I agree     [N]o, thanks                        |
  |                                                        |
  =-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-:

  ~ root#
```

Play at “Kotlin Playground”

and “Get Started” in https://kotlinlang.org/


https://play.kotlinlang.org/#eyJ2ZXJzaW9uIjoiMS40LjMwIiwicGxhdGZvcm0iOi
Search
Kotlin
Kotlin Playground is an online sandbox to explore Kotlin programming language. Browse code samples directly in the browser
Playground
Hands-on
Examples
Koans
https://kotlinlang.org
Search
Kotlin v1.4.30
Solutions
Get started
Why Kotlin
JET BRAINS
Developed by JetBrains & Open-source Contributors

```
/**
 * You can edit, run, and share this code.
 * play.kotlinlang.org
 */

fun main() {
    println("Hello, world!!!")
}
```

```
Hello, world!!!
```

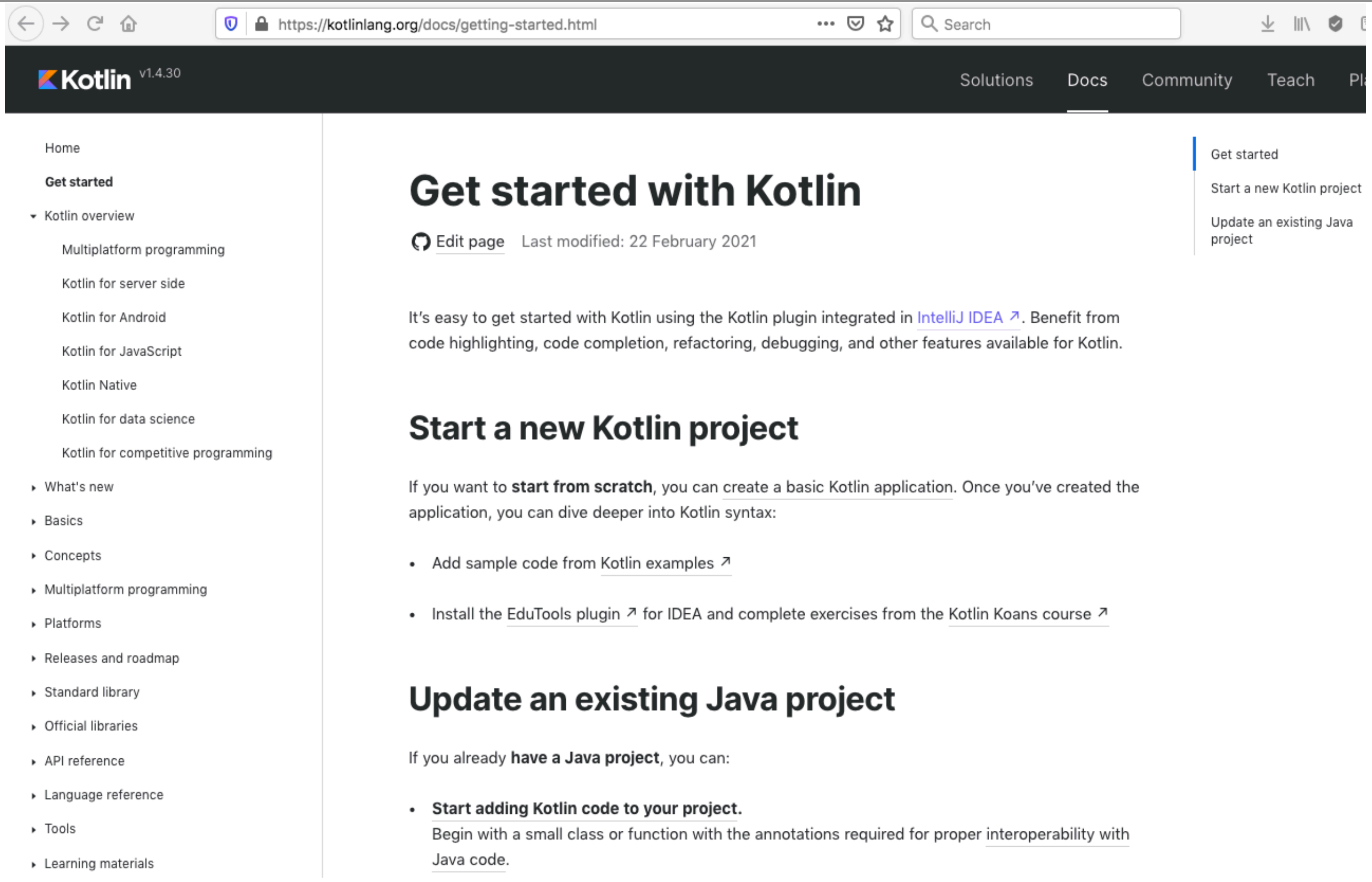
A modern programming language that makes developers happier.

```
| third-party services for this purpose. I can revoke |
| my consent at any time by visiting the Opt-Out page. |
|                                                      |
| [Y]es, I agree    [N]o, thanks                       |
|                                                      |
=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-

~ root#
```

© 2000-2021 JetBrains. All rights reserved

Get Started



Get started with Kotlin

Edit page Last modified: 22 February 2021

It's easy to get started with Kotlin using the Kotlin plugin integrated in IntelliJ IDEA ↗. Benefit from code highlighting, code completion, refactoring, debugging, and other features available for Kotlin.

Start a new Kotlin project

If you want to **start from scratch**, you can create a basic Kotlin application. Once you've created the application, you can dive deeper into Kotlin syntax:

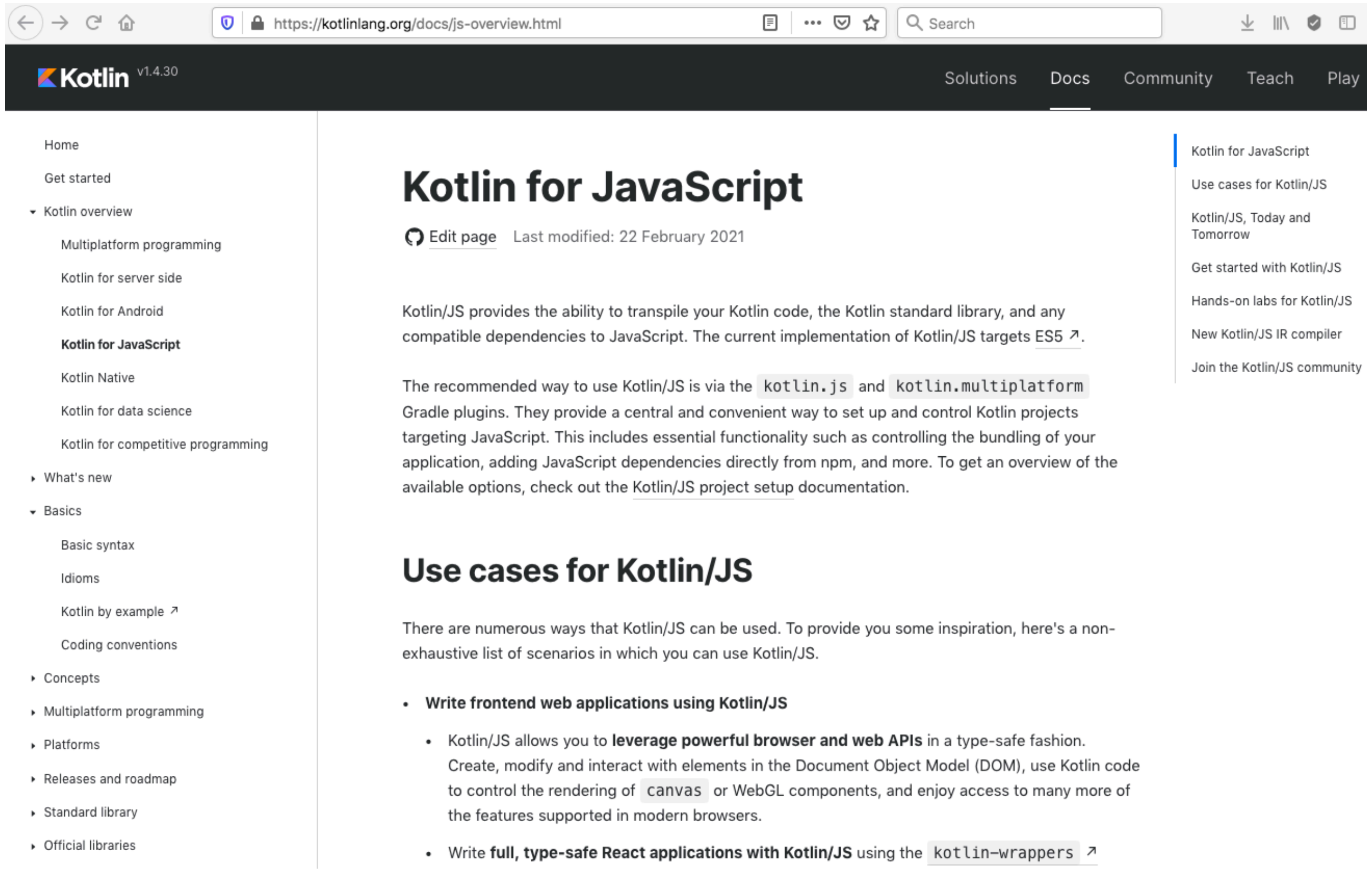
- Add sample code from Kotlin examples ↗
- Install the EduTools plugin ↗ for IDEA and complete exercises from the Kotlin Koans course ↗

Update an existing Java project

If you already **have a Java project**, you can:

- **Start adding Kotlin code to your project.**
  Begin with a small class or function with the annotations required for proper interoperability with Java code.

Multi-Platform, Transpile, Native, ..



https://kotlinlang.org/docs/js-overview.html

Kotlin for JavaScript

Edit page Last modified: 22 February 2021

Kotlin/JS provides the ability to transpile your Kotlin code, the Kotlin standard library, and any compatible dependencies to JavaScript. The current implementation of Kotlin/JS targets ES5 ↗.

The recommended way to use Kotlin/JS is via the `kotlin.js` and `kotlin.multiplatform` Gradle plugins. They provide a central and convenient way to set up and control Kotlin projects targeting JavaScript. This includes essential functionality such as controlling the bundling of your application, adding JavaScript dependencies directly from npm, and more. To get an overview of the available options, check out the Kotlin/JS project setup documentation.

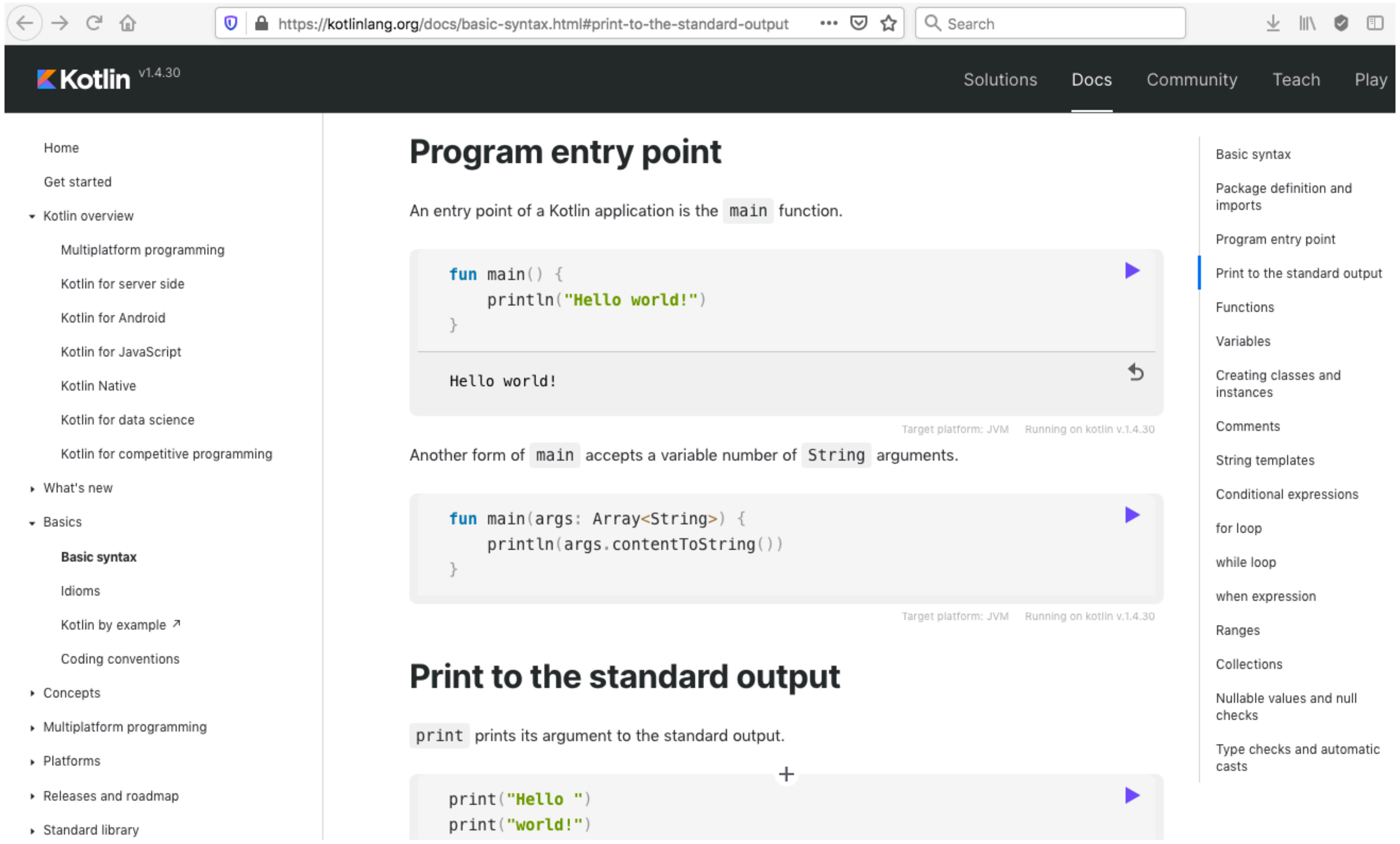
Use cases for Kotlin/JS

There are numerous ways that Kotlin/JS can be used. To provide you some inspiration, here's a non-exhaustive list of scenarios in which you can use Kotlin/JS.

- **Write frontend web applications using Kotlin/JS**
  - Kotlin/JS allows you to **leverage powerful browser and web APIs** in a type-safe fashion. Create, modify and interact with elements in the Document Object Model (DOM), use Kotlin code to control the rendering of `canvas` or WebGL components, and enjoy access to many more of the features supported in modern browsers.
  - Write **full, type-safe React applications with Kotlin/JS** using the `kotlin-wrappers` ↗

Learn by on-line Play Capabilites

HTL Villach
Future Inside



https://kotlinlang.org/docs/basic-syntax.html#print-to-the-standard-output

Program entry point

An entry point of a Kotlin application is the `main` function.

```
fun main() {
    println("Hello world!")
}
```

```
Hello world!
```

Target platform: JVM   Running on kotlin v.1.4.30

Another form of `main` accepts a variable number of `String` arguments.

```
fun main(args: Array<String>) {
    println(args.contentToString())
}
```

Target platform: JVM   Running on kotlin v.1.4.30

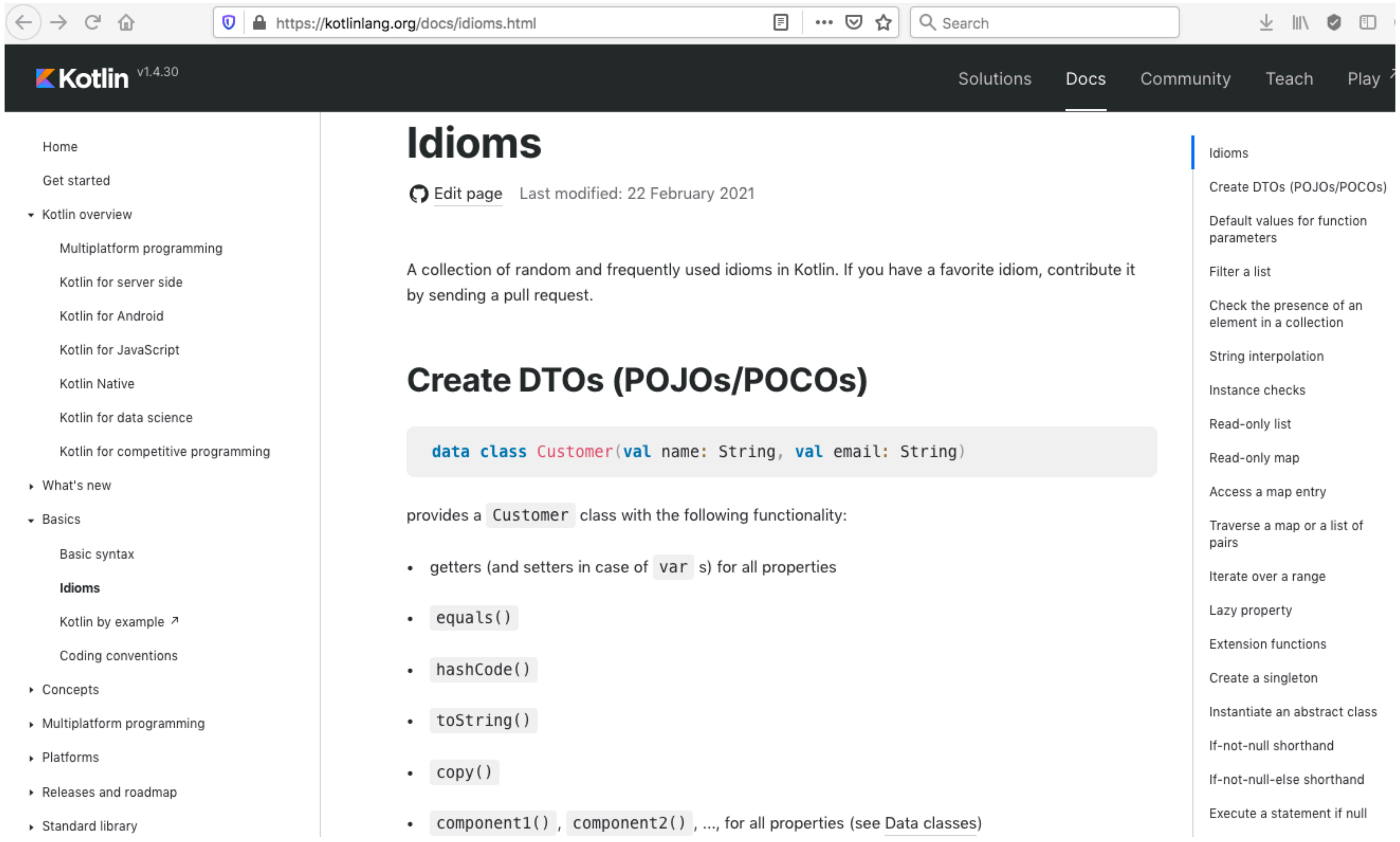
Print to the standard output

`print` prints its argument to the standard output.

```
print("Hello ")
print("world!")
```

software inside

.. on basic, favorite “(s)elected” Idioms



https://kotlinlang.org/docs/idioms.html

Idioms

Edit page Last modified: 22 February 2021

A collection of random and frequently used idioms in Kotlin. If you have a favorite idiom, contribute it by sending a pull request.

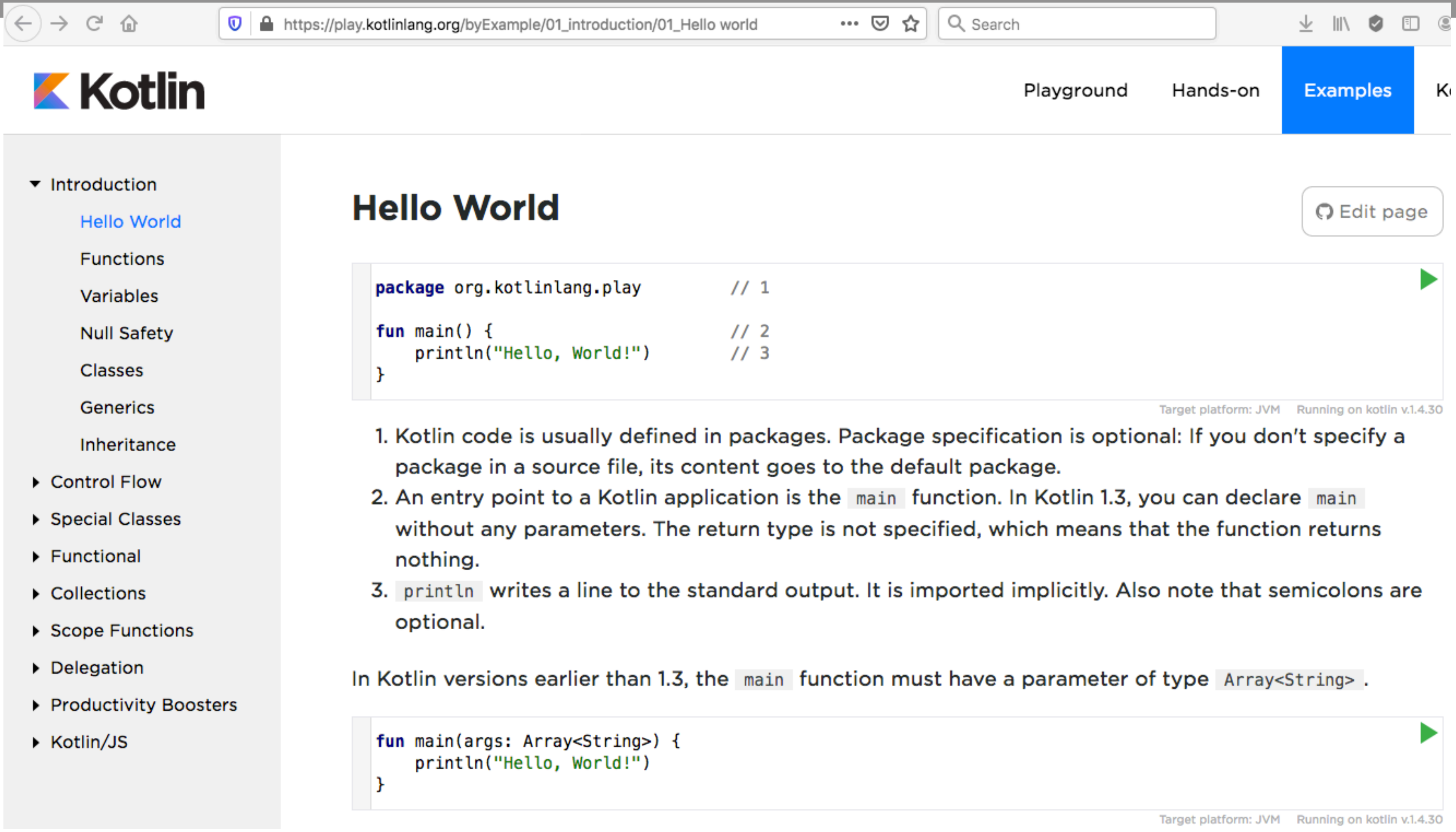
Create DTOs (POJOs/POCOs)

```
data class Customer(val name: String, val email: String)
```

provides a `Customer` class with the following functionality:

- getters (and setters in case of `var` s) for all properties
- `equals()`
- `hashCode()`
- `toString()`
- `copy()`
- `component1()` , `component2()` , ..., for all properties (see Data classes)

.. to play in a world of Examples



Hello World

```
package org.kotlinlang.play         // 1

fun main() {                        // 2
    println("Hello, World!")        // 3
}
```

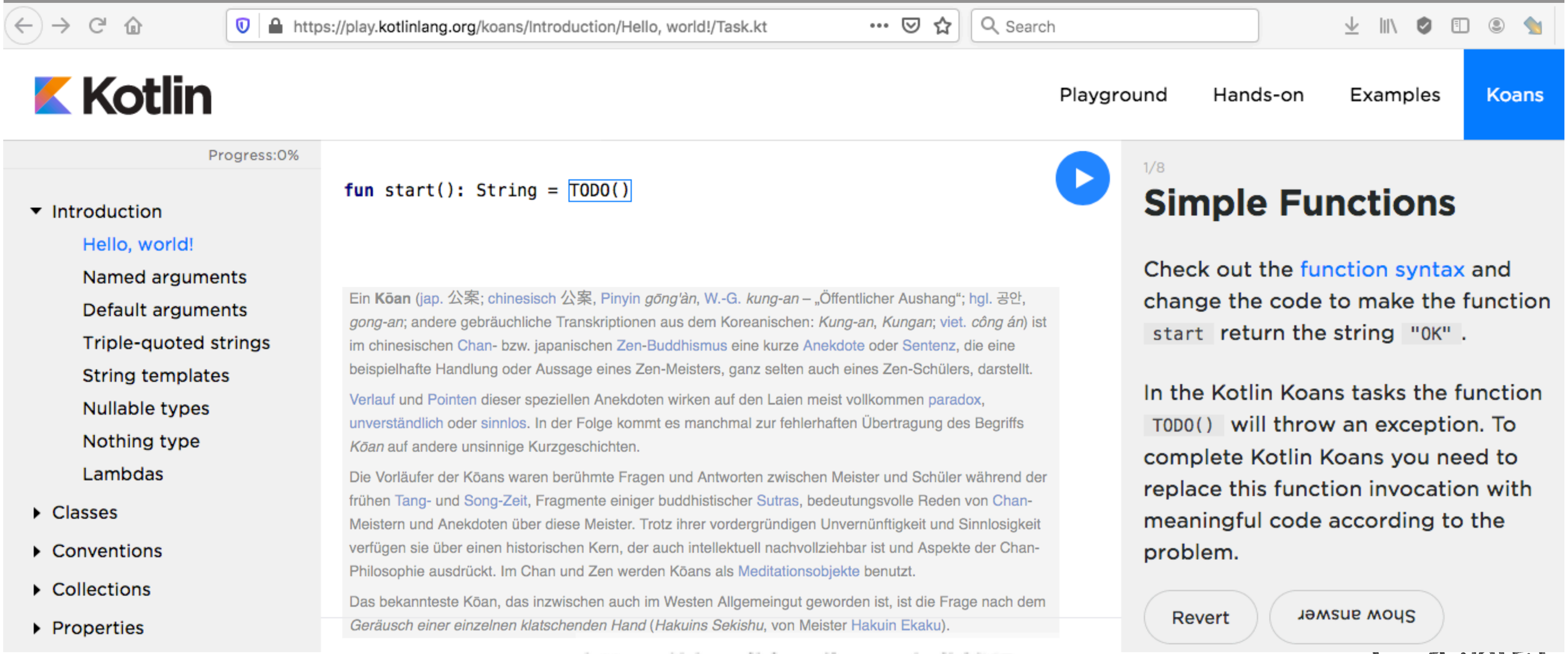
1. Kotlin code is usually defined in packages. Package specification is optional: If you don't specify a package in a source file, its content goes to the default package.
2. An entry point to a Kotlin application is the `main` function. In Kotlin 1.3, you can declare `main` without any parameters. The return type is not specified, which means that the function returns nothing.
3. `println` writes a line to the standard output. It is imported implicitly. Also note that semicolons are optional.

In Kotlin versions earlier than 1.3, the `main` function must have a parameter of type `Array<String>`.

```
fun main(args: Array<String>) {
    println("Hello, World!")
}
```

.. and test yourself by koans

HTL Villach
Future Inside



```kotlin
fun start(): String = TODO()
```

Simple Functions

Check out the function syntax and change the code to make the function `start` return the string `"OK"`.

In the Kotlin Koans tasks the function `TODO()` will throw an exception. To complete Kotlin Koans you need to replace this function invocation with meaningful code according to the problem.

Ein **Kōan** (jap. 公案; chinesisch 公案, Pinyin *gōng'àn*, W.-G. *kung-an* – „Öffentlicher Aushang"; hgl. 공안, *gong-an*; andere gebräuchliche Transkriptionen aus dem Koreanischen: *Kung-an*, *Kungan*; viet. *công án*) ist im chinesischen Chan- bzw. japanischen Zen-Buddhismus eine kurze Anekdote oder Sentenz, die eine beispielhafte Handlung oder Aussage eines Zen-Meisters, ganz selten auch eines Zen-Schülers, darstellt.

Verlauf und Pointen dieser speziellen Anekdoten wirken auf den Laien meist vollkommen paradox, unverständlich oder sinnlos. In der Folge kommt es manchmal zur fehlerhaften Übertragung des Begriffs *Kōan* auf andere unsinnige Kurzgeschichten.

Die Vorläufer der Kōans waren berühmte Fragen und Antworten zwischen Meister und Schüler während der frühen Tang- und Song-Zeit, Fragmente einiger buddhistischer Sutras, bedeutungsvolle Reden von Chan-Meistern und Anekdoten über diese Meister. Trotz ihrer vordergründigen Unvernünftigkeit und Sinnlosigkeit verfügen sie über einen historischen Kern, der auch intellektuell nachvollziehbar ist und Aspekte der Chan-Philosophie ausdrückt. Im Chan und Zen werden Kōans als Meditationsobjekte benutzt.

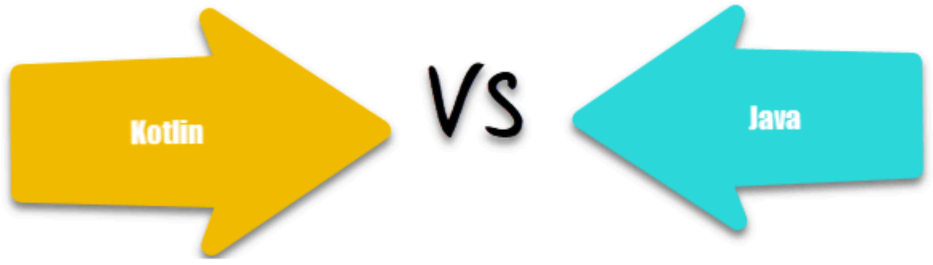
Das bekannteste Kōan, das inzwischen auch im Westen Allgemeingut geworden ist, ist die Frage nach dem *Geräusch einer einzelnen klatschenden Hand* (*Hakuins Sekishu*, von Meister Hakuin Ekaku).

https://de.wikipedia.org/wiki/Kōan

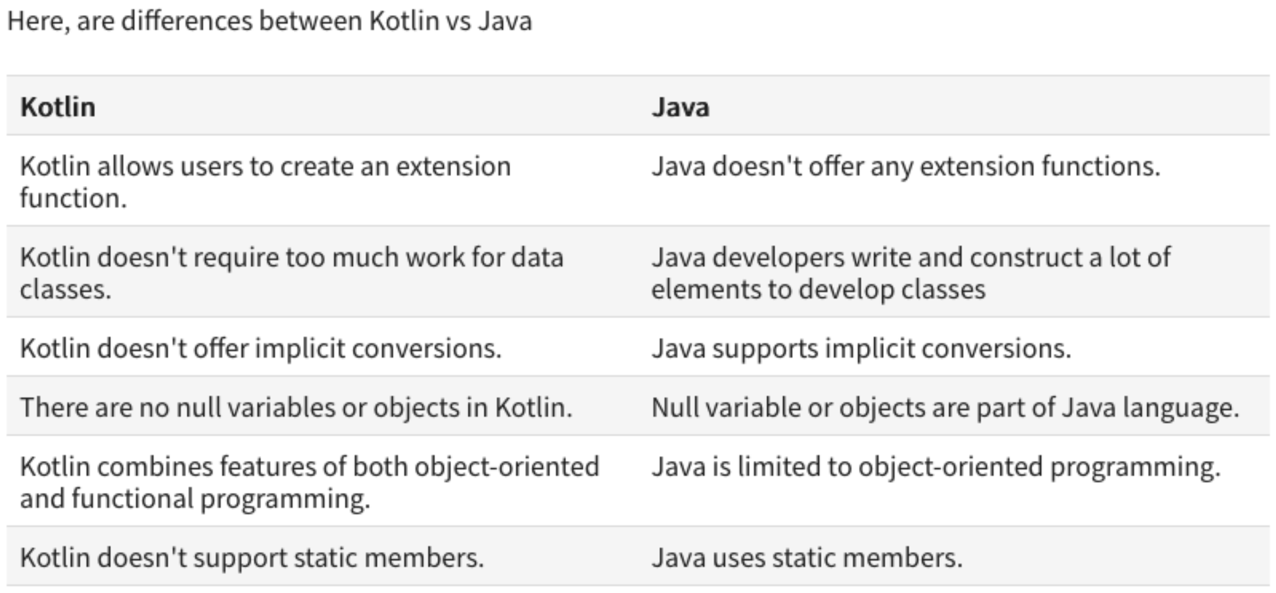
software inside

Exercise: (im)prove comparision

at https://www.guru99.com/kotlin-vs-java-difference.html



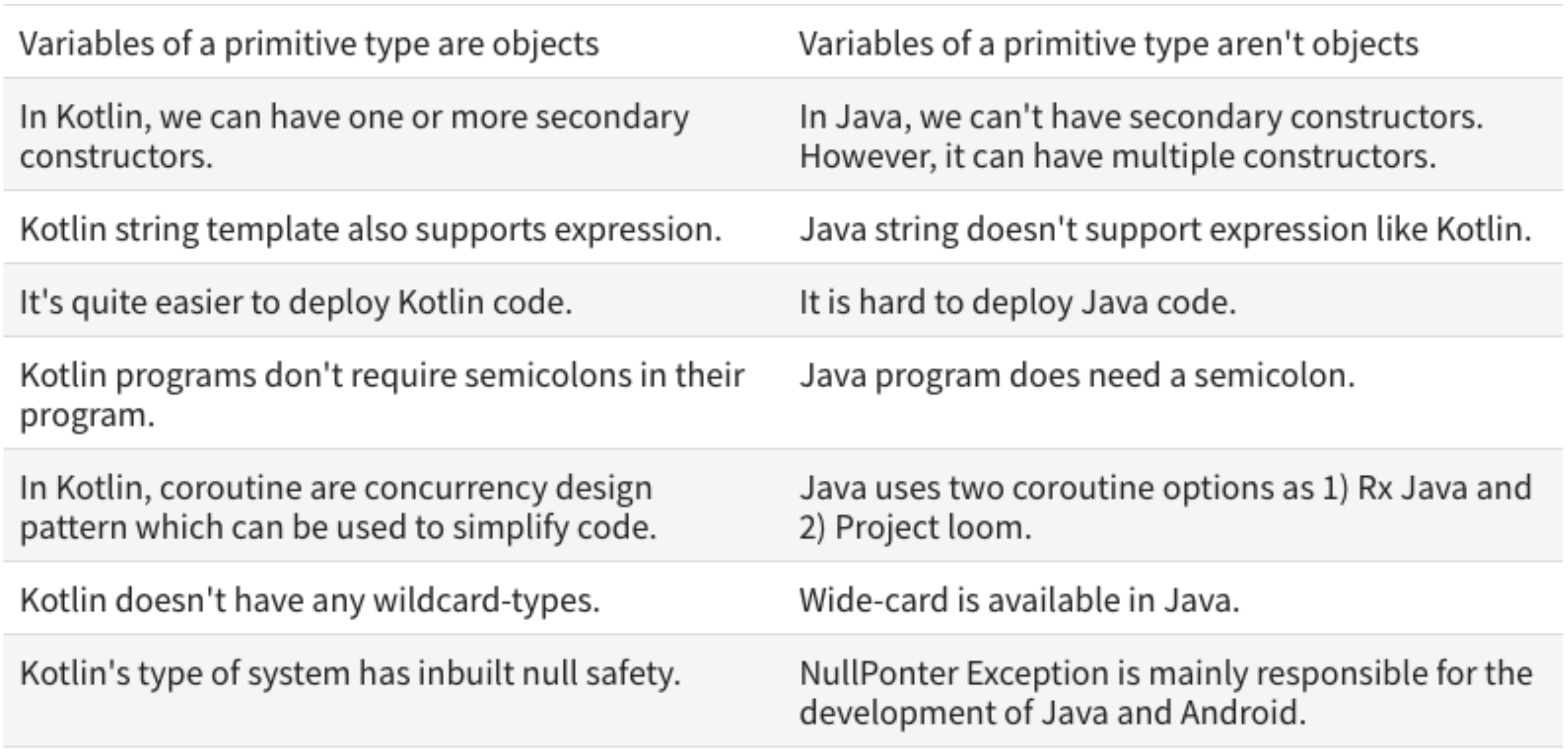
Kotlin Vs Java



Here, are differences between Kotlin vs Java

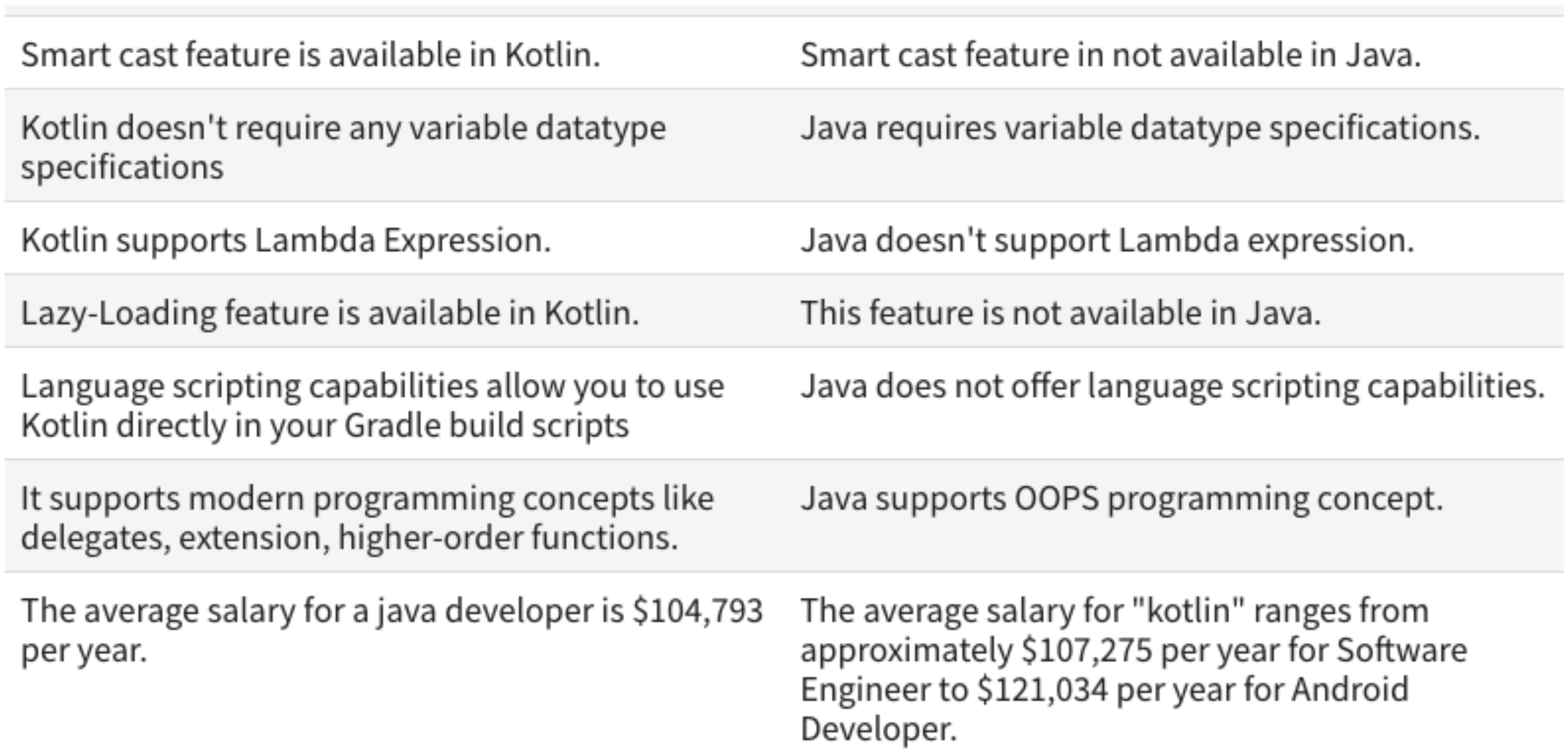
Kotlin	Java
Kotlin allows users to create an extension function.	Java doesn't offer any extension functions.
Kotlin doesn't require too much work for data classes.	Java developers write and construct a lot of elements to develop classes
Kotlin doesn't offer implicit conversions.	Java supports implicit conversions.
There are no null variables or objects in Kotlin.	Null variable or objects are part of Java language.
Kotlin combines features of both object-oriented and functional programming.	Java is limited to object-oriented programming.
Kotlin doesn't support static members.	Java uses static members.

Comparision cont'd.



Variables of a primitive type are objects	Variables of a primitive type aren't objects
In Kotlin, we can have one or more secondary constructors.	In Java, we can't have secondary constructors. However, it can have multiple constructors.
Kotlin string template also supports expression.	Java string doesn't support expression like Kotlin.
It's quite easier to deploy Kotlin code.	It is hard to deploy Java code.
Kotlin programs don't require semicolons in their program.	Java program does need a semicolon.
In Kotlin, coroutine are concurrency design pattern which can be used to simplify code.	Java uses two coroutine options as 1) Rx Java and 2) Project loom.
Kotlin doesn't have any wildcard-types.	Wide-card is available in Java.
Kotlin's type of system has inbuilt null safety.	NullPonter Exception is mainly responsible for the development of Java and Android.

Comparision cont’d. – still true?



Smart cast feature is available in Kotlin.	Smart cast feature in not available in Java.
Kotlin doesn't require any variable datatype specifications	Java requires variable datatype specifications.
Kotlin supports Lambda Expression.	Java doesn't support Lambda expression.
Lazy-Loading feature is available in Kotlin.	This feature is not available in Java.
Language scripting capabilities allow you to use Kotlin directly in your Gradle build scripts	Java does not offer language scripting capabilities.
It supports modern programming concepts like delegates, extension, higher-order functions.	Java supports OOPS programming concept.
The average salary for a java developer is $104,793 per year.	The average salary for "kotlin" ranges from approximately $107,275 per year for Software Engineer to $121,034 per year for Android Developer.