Milan | November 29 - 30, 2018

{codemotion}

Kotlin loves React

Davide Cerbo
Software Architect @ E.m.m. Informatica

Davide Cerbo

15+ years as developer

DevDay Community Leader

@davide_cerbo

https://medium.com/@davidecerbo

Kotlin is a statically-typed programming language that runs on the Java Virtual Machine and also can be compiled to JavaScript source code or uses the LLVM compiler infrastructure.

Since 2010 (2012): KOTLIN!

~40% less lines of code than Java

100% interoperable with Java

JetBrains

Apache 2

It's safe

Easy to learn, if you know Java

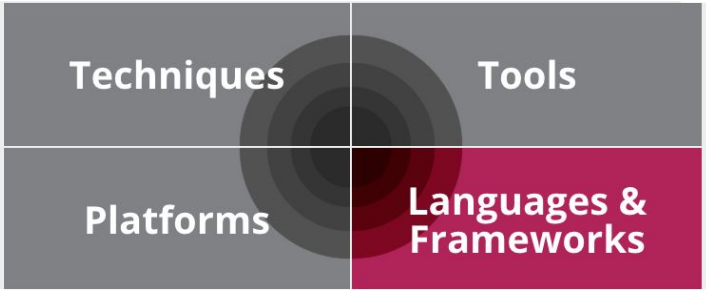
Predictability

https://dzone.com/articles/why-learn-kotlin-infographic

Predicatability not performance:https://www.youtube.com/watch?v=ExkNNsDn6Vg

ThoughtWorks®

TECHNOLOGY RADAR



Technlogy Radar

https://www.thoughtworks.com/radar/languages-and-frameworks/kotlin

My first API with Ktor

```
fun main(args: Array<String>) {
    val server = embeddedServer(Netty, 8080) {
        install(ContentNegotiation) {
            jackson { enable(SerializationFeature.INDENT_OUTPUT) // Pretty Prints the JSON }
        }
        routing {
            get("/greetings") {
                call.respond(Greetings().sayGreetings())
            }
        }
    }
    server.start(wait = true)
}
```

Greetings

```
actual class Greetings {
    actual fun sayGreetings(): String {
        return "Hello from JVM world!!!"
    }
}
```

but...

KotlinJS can!

- No longer experimental since 1.1.0
- Can run anywhere where JS runs
- Can interoperate with JS
- Can reuse code
- Can use the same Kotlin/JVM IDE

wait, React?

React

A JavaScript library for building user interfaces.

The V in M**V**C

Components, not templates.

Re-render, don’t mutate.

Virtual DOM is simple and fast

setState({a:1})

Kotlin React Wrapper

https://github.com/JetBrains/kotlin-wrappers/blob/master/kotlin-react/src/main/kotlin/react/React.kt

Kotlin (experimental) Multiplatform

https://github.com/gbaldeck/react-js-jvm-kotlin-multiplatform

Demo!!!

https://github.com/jesty/kotlin-loves-react

Main

```
import app.*
import kotlinext.js.*
import react.dom.*
import kotlin.browser.*

fun main(args: Array<String>) {
    requireAll(require.context("src", true, js("/\\.css$/")))

    render(document.getElementById("root")) {
        app()
    }
}
```

Components

```
class Welcome :
RComponent<RProps,
RState>() {
```

State & Props

```
interface WelcomeProps : RProps {
    var name: String
}

interface WelcomeState : RState {
    var color: String
}
```

Components with state & props

```
class Welcome :
RComponent<WelcomeProps,
WelcomeState>() {
```

Render

Weird! This pass this

```
override fun RBuilder.render() {
    div {
        +"Hello, ${props.name} - ${state.color}"
        button {
            +"Click me ${props.name}"
            attrs {
                onClickFunction = {
                    setState { color = "red" }
                } } } }
    }
```

https://kotlinlang.org/docs/reference/type-safe-builders.html

Style

```
import kotlinext.js.js
//...
    div {
            attr.title = “Hello”
            //Not typesafe, js body function must be a constant
            attr.style= js { color: “red” }
            //no constant issue
            jsStyle { color = “red” }
        }
//...
```

Routing

```
hashRouter {
    switch {
        route("/", IndexComponent::class, exact = true)
        route("/about", AboutComponent::class, exact = true)
    }
}
//Link
routeLink("/about") { +"About" }
```

setState {a =1}

JS interoperation

```
fun dynamicExample() {
    val a: dynamic = js("{ foo: function () { console.log(Hi!')} }")
    a.foo()
    a.bar() //Uncaught TypeError: a.bar is not a function
}
```

```
fun externalExample() {
  val e = E()
  e.foo()
  //e.bar() //Compile time error!!!
}

external class E {
  fun foo()
}

//helloworld.js
var E = function(){
  this.foo = function () {
     console.log('Hello world! (external)')
  } }
```

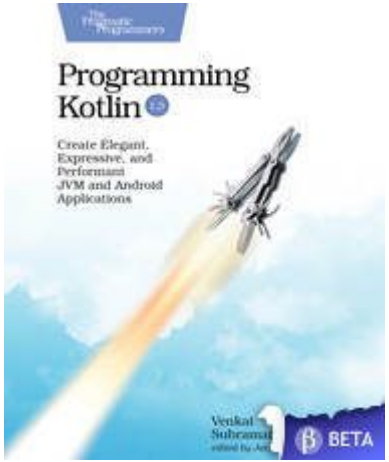
```
fun actualExpectExample() {
  val ea = EA()
  ea.foo()
}

expect class EA {
  fun foo()
}

//in a JS project
actual class EA {
  fun foo() { println("Hello from JS!") }
}
```

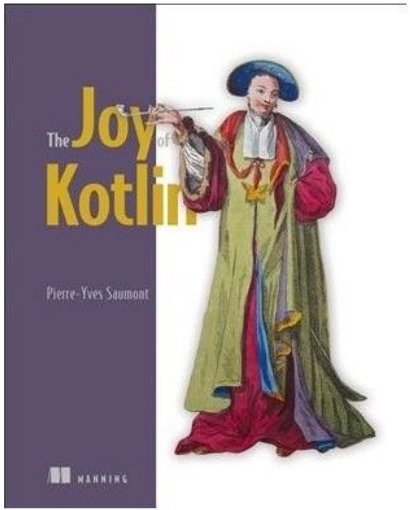
```
//in a JVM project
actual class EA {
  fun foo() { println("Hello from JVM!") }
}
```

```
if (typeof kotlin === 'undefined') {
 throw new Error("Error loading module 'codemotion'. Its dependency 'kotlin'
was not found. Please, check whether 'kotlin' is loaded prior to 'github'.");
}
var github = function (_, Kotlin) {
 'use strict';
 var println = Kotlin.kotlin.io.println_s8jyv4$;
 var Kind_CLASS = Kotlin.Kind.CLASS;
 function dynamicExample() {
   var a = {foo: function () {
     console.log('Hello world! (dynamic)');
   }};
   a.foo();
 }
 function externalExample() {
   var e = new E();
   e.foo();
 }
```

(Auto)references

- https://github.com/jesty/kotlin-fossavotabona
- https://github.com/jesty/reactiveredis



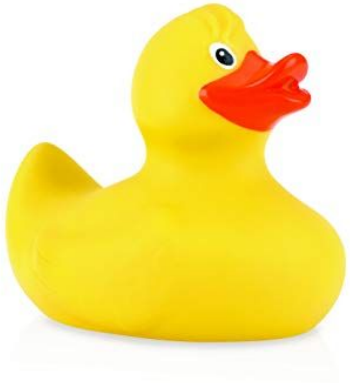


At the end...

Dynamic Vs. Static typed

Static: Types checked before run-time

Dynamic: Types checked on the fly, during execution



Typescript Vs. KotlinJs

TypeScript is a superset of Javascript.

Kotlin aims to be full-stack for creating apps.

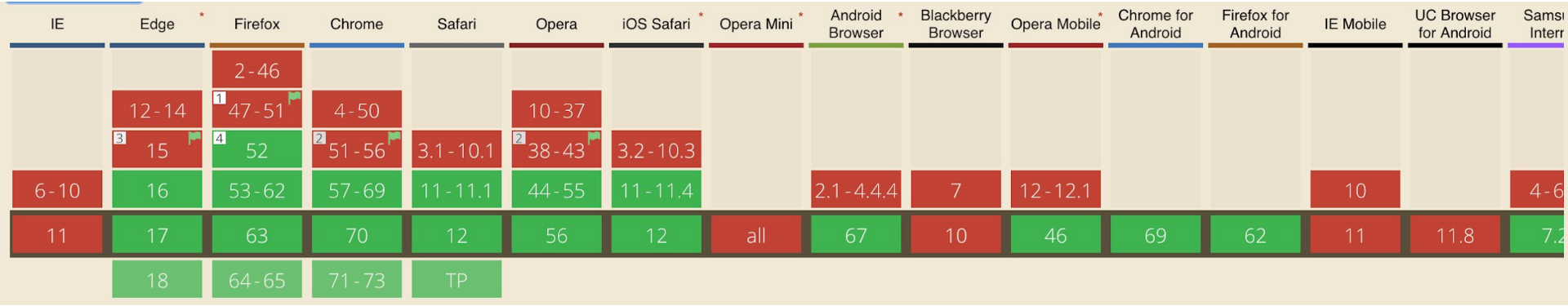
https://www.slant.co/versus/378/1543/~typescript_vs_kotlin

https://discuss.kotlinlang.org/t/feedback-on-our-migration-from-typescript-to-kotlin/2578

but...

...suspance...

Webassembly?


IE
Edge *
Firefox
Chrome
Safari
Opera
iOS Safari *
Opera Mini *
Android Browser *
Blackberry Browser
Opera Mobile *
Chrome for Android
Firefox for Android
IE Mobile
UC Browser for Android
Samsu Interr
2-46
12-14
1 47-51
4-50
10-37
3 15
4 52
2 51-56
3.1-10.1
2 38-43
3.2-10.3
6-10
16
53-62
57-69
11-11.1
44-55
11-11.4
2.1-4.4.4
7
12-12.1
10
4-6
11
17
63
70
12
56
12
all
67
10
46
69
62
11
11.8
7.2
18
64-65
71-73
TP

The end!

https://twitter.com/davide_cerbo

let's talk?

http://devday.it