KOTLIN

LESS CODE, MORE FUN

Michael Fazio - @faziodev

THANK YOU, THAT CONFERENCE SPONSORS!

SKYLINE

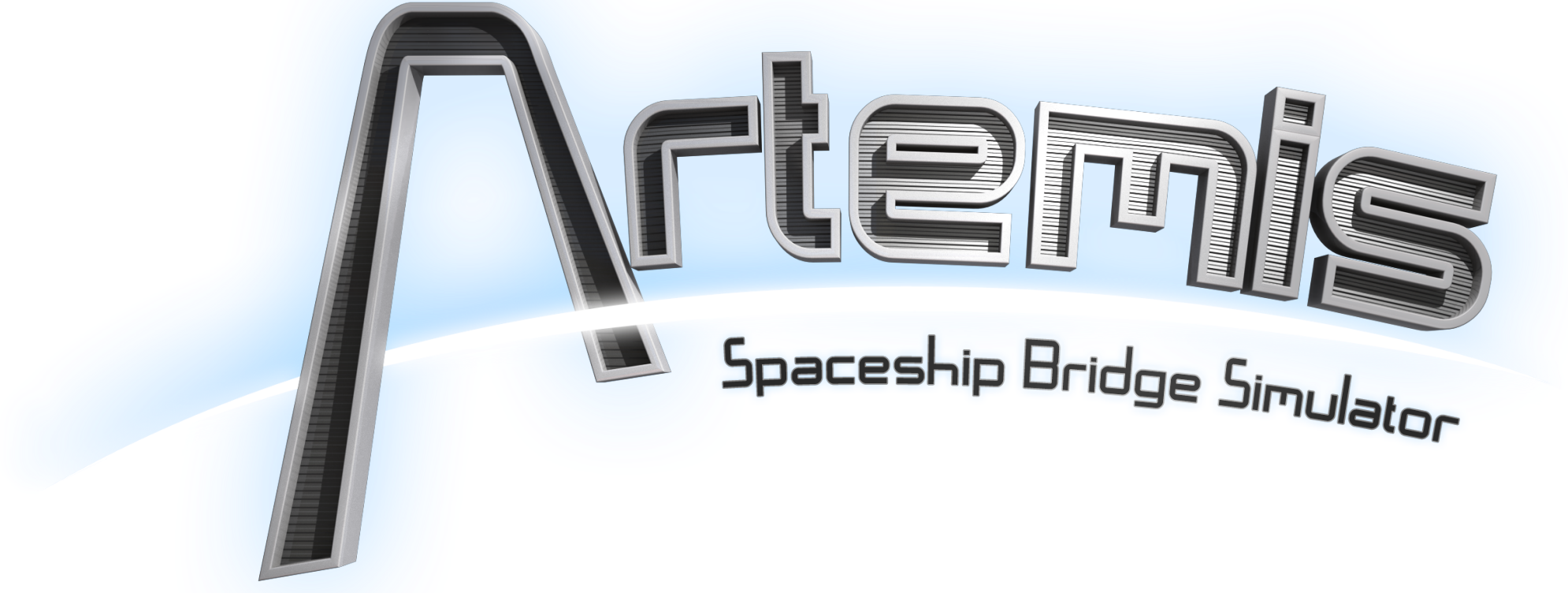
see beyond your **it**


Java™

Microsoft®

.NET


stackoverflow


Artemis
Spaceship Bridge Simulator

WHAT IS...

Kotlin

"STATICALLY TYPED PROGRAMMING LANGUAGE FOR MODERN MULTIPLATFORM APPLICATIONS"

STATICALLY TYPED

```
val numbers: List<Int> = listOf(1, 4, 7, 10, 23)
```

```
enum class Sport {
    Baseball,
    Football,
    Soccer,
    Basketball
}

data class Team(val city: String, val teamName: String, val sport: Spo

val team = Team("Milwaukee", "Brewers", Sport.Baseball)
```

MULTI-PLATFORM

Build Applications For

JVM

Android

Browser

Native

100% JAVA INTEROPERABILITY

```
import io.reactivex.Flowable
import io.reactivex.schedulers.Schedulers

Flowable
    .fromCallable {
        Thread.sleep(1000) //  imitate expensive computation
        "Done"
    }
    .subscribeOn(Schedulers.io())
    .observeOn(Schedulers.single())
    .subscribe(::println, Throwable::printStackTrace)
```

FIRST-CLASS LANGUAGE ON ANDROID

```
override fun onCreate(savedInstanceState: Bundle?) {
    super.onCreate(savedInstanceState)
    setContentView(R.layout.activity_main)
    setSupportActionBar(toolbar)
    ...

    view.SetOnClickListener { v -> handleClick(v) }
}
```

Why Kotlin?

Concise

Drastically reduce the amount of boilerplate code.

See example

Safe

Avoid entire classes of errors such as null pointer exceptions.

See example

Interoperable

Leverage existing libraries for the JVM, Android, and the browser.

See example

Tool-friendly

Choose any Java IDE or build from the command line.

See example

VAL AND VAR

```java
// Java
final String name = "Joe Smith"
```

```kotlin
// Kotlin
val name: String = "Joe Smith"
```

```java
// Java
int age = 23
```

```kotlin
// Kotlin
var age: Int = 23
```

DATA CLASSES

```java
// Java
public class Person {
    private final String first;
    private final String last;
    private int age;

    public Person(String first, String last, int age) {
        this.first = first;
        this.last = last;
        this.age = age;
    }

    public String getFirst() {
        return first;
    }
```

```kotlin
// Kotlin
data class Person(val first: String, val last: String, var age: Int)
```

DATA CLASSES

COMPONENT FUNCTIONS, DESTRUCTURING DECLARATIONS, COPY

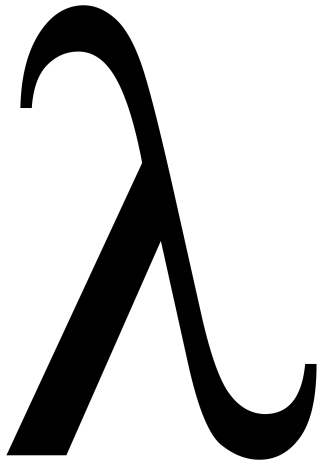
```
data class Person(val first: String, val last: String, var age: Int)
```

```
val hazel = Person(first = "Hazel", last = "Fazio", age = 4)
println(hazel) // Person(first=Hazel, last=Fazio, age=4)

val (first, last, age) = hazel
println("$first $last, $age years old.") // Hazel Fazio, 4 years old.

val hazelNY = hazel.copy(age = 5) // Next year
println("${hazelNY.component1()} - ${hazelNY.component3()}") // Hazel
```

LAMBDA FUNCTIONS



```
val sum = { x: Int, y: Int -> x + y }
```

```
val baseballTeams: List<Team> =
    teams.filter { team -> team.sport == Sport.Baseball}
```

```
strings
    .filter { it.length == 5 }
    .sortedBy { it }
    .map { it.toUpperCase() }
```

SMART CASTS

```java
// Java
private int calculateLength(Object o) {
    if(o instanceof String) {
        final String s = (String) o;
        return s.length();
    }
    return 0;
}
```

```csharp
// C#
private int CalculateLength(object o) {
    if(o is string s) return s.Length;
    return 0;
}
```

```kotlin
// Kotlin
fun calculateLength(v: Any) : Int {
    if(v is String) return v.length //v is automatically cast to a Str

    return 0
}
```

RANGES

```java
// Java
private boolean isInRange(int i) {
    return i >= 1 && i<= 10;
}
```

```kotlin
// Kotlin
fun isInRange(i: Int) : Boolean {
    return i in 1..10
}
```

```kotlin
// Kotlin
fun isInRange(i: Int) = i in 1..10
```

WHEN EXPRESSIONS

```
when (x) {
    1 -> print("x == 1")
    2 -> print("x == 2")
    else -> { // Note the block
        print("x is neither 1 nor 2")
    }
}
```

```
val result: String = when (x) {
    0, 1 -> "x == 0 or x == 1"
    else -> "otherwise"
}
```

```
when (x) {
    in 1..10 -> print("x is in the range")
    in validNumbers -> print("x is valid")
    !in 10..20 -> print("x is outside the range")
    else -> print("none of the above")
}
```

TRAVERSING MAPS

```java
// Java
public void printTeams(Map<String, Team> teamMap) {
    for(Map.Entry<String, Team> entry : teamMap.entrySet()) {
        System.out.println(entry.getKey() + " - " + entry.getValue());
    }
}
```

```kotlin
// Kotlin
fun printTeams(teamMap: Map<String, Team>) {
    for ((key, value) in teamMap) {
        println("$key - $value")
    }
}
```

NULL

THE BILLION-DOLLAR MISTAKE

```
var notNullString: String = "This can't be null"

var nullableString: String? = "This could be null"
```

NULL SAFETY

```
var a: String = "abc"
a = null // compilation error
```

```
var b: String? = "abc"
b = null // ok
```

```
val l = a.length
```

```
val l = b.length // error: variable 'b' can be null
```

NULL SCENARIOS

- throw NullPointerException();
- Usage of the `!!` operator
- Data inconsistency related to initialization
  - An uninitialized `this` available in a constructor and is used somewhere
  - A superclass constructor calls an open member whose implementation uses uninitialized state
- Java interop
  - Access a null member reference of a platform type
  - Generic types with incorrect nullability
  - "Other issues" caused by external Java code

NULL SAFETY

SAFE CALLS

```
val brewers: Team? = Team("Milwaukee", "Brewers", Sport.Baseball)
val whiteSox: Team? = Team("Chicago", "White Sox", Sport.Baseball)
val devilRays: Team? = null

val baseballTeams = listOf(brewers, whiteSox, devilRays)
```

```
val nicknames: List<String> = baseballTeams.map {
    if (it != null) it.name else ""
}
```

```
val nicknames: List<String> = baseballTeams.map {
    it?.name ?: ""
}
```

ELVIS OPERATOR

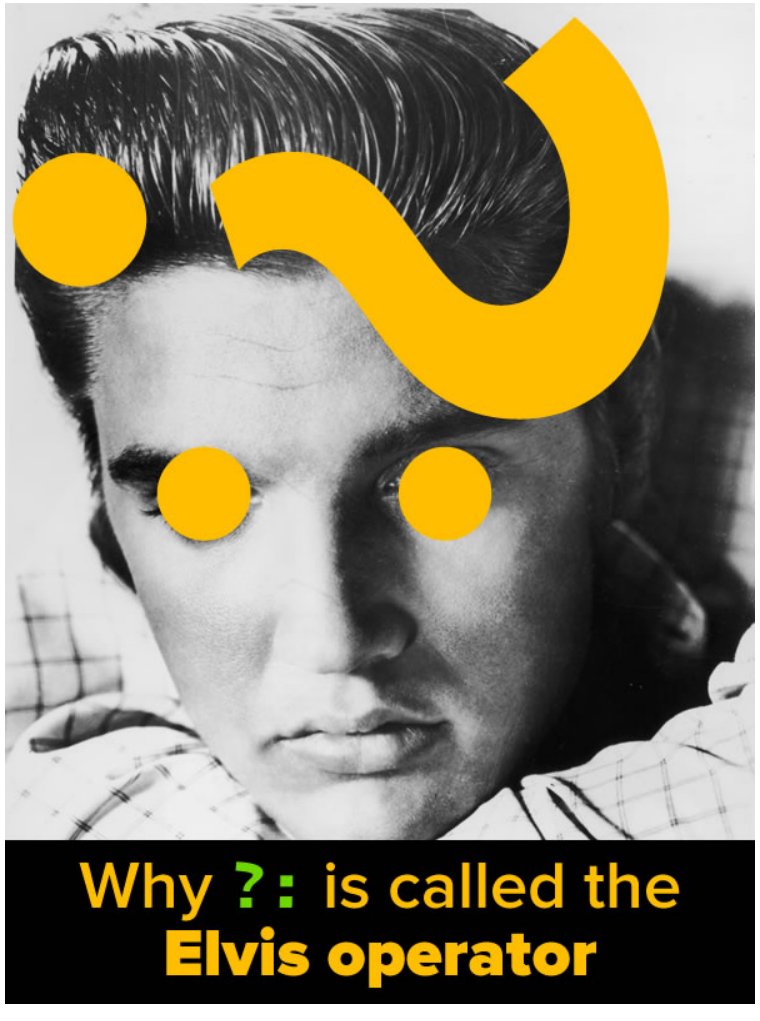
```java
// Java
public Sport getTeamSport(Team team) {
    if(team != null) {
        return team.getSport();
    }

    return Sport.Unknown;
}
```

```kotlin
// Kotlin
fun getTeamSport(team: Team?): Sport
    = if(team != null) team.sport else Sport.Unknown

fun getTeamSport(team: Team?): Sport = team?.sport ?: Sport.Unknown
```

ELVIS OPERATOR



NULL SAFE SCOPING

```
fun findTeam(city: String = "", nickname: String = ""): Team? {
    return this.teams.firstOrNull {
        team -> team?.city == city || team?.name == nickname
    }
}

findTeam(nickname = "Brewers")?.let {
    //This will only run if findTeam(...) returns a non-null value
}
```

NULL SAFE SCOPING

```
fun listTeams() {
    val brewers: Team? = Team("Milwaukee", "Brewers", Sport.Baseball)
    val whiteSox: Team? = Team("Chicago", "White Sox", Sport.Baseball)
    val devilRays: Team? = null

    println(this.getSummary(brewers))
    println(this.getSummary(whiteSox))
    println(this.getSummary(devilRays))
}
```

```
private fun getSummary(team: Team?): String {
    var fullName: String = "N/A"

    team?.let({t -> fullName = "${t.city} ${t.name} (${t.sport})"})

    return fullName
}
```

NULL SAFE SCOPING

```
private fun getSummary(team: Team?): String {
    return team?.let({ t -> "${t.city} ${t.name} (${t.sport})" }) ?: "
}
```

```
private fun getSummary(team: Team?): String {
    return team?.let { t -> "${t.city} ${t.name} (${t.sport})" } ?: "N
}
```

```
private fun getSummary(team: Team?): String {
    return team?.let { "${it.city} ${it.name} (${it.sport})" } ?: "N/A
}
```

```
private fun getSummary(team: Team?): String =
    team?.let { "${it.city} ${it.name} (${it.sport})" } ?: "N/A"
```

JAVA HAS THESE, KOTLIN DOES NOT

- Ternary-operator (`a ? b : c`)
- Checked exceptions
- Primitive types that are not classes
- Static members
- Non-private fields

INSTEAD OF STATIC

```
package org.faziodev.kotlin

// Package-level function
fun nullSafeToUpper(value: String?) : String {
    return value?.toUpperCase() ?: ""
}

// Object declaration
object TeamSource {
    fun myFunction() { ... }
}
```

```
// Import the package and call the method
val upper: String = nullSafeToUpper(brewers?.name)

// Reference the function similarly to a static method
TeamSource.myFunction()
```

COMPANION OBJECTS

```
data class Team(val city: String, val name: String, var sport: Sport)
    companion object {
        fun createBaseballTeam(city: String, name: String)
            = Team(city, name, Sport.Baseball)
    }
}
```

```
val cubs = Team.createBaseballTeam("Chicago", "Cubs")
```

ALL CLASSES ARE CLOSED

```
class Base(p: Int)

//This will not work
class Derived(p: Int) : Base(p)
```

```
open class Base(p: Int)

class Derived(p: Int) : Base(p)
```

ALL FUNCTIONS ARE CLOSED

```
open class Base {
    open fun openFun() {}
    open fun alsoOpenFun() {}
    fun notOpenFun() {}
}
```

```
class Derived() : Base() {
    //Overridden methods are open by default
    override fun openFun() {}

    // This can no longer be overridden
    final override fun alsoOpenFun() {}
}
```

COLLECTIONS ARE IMMUTABLE

```
val teams: List<Team> = this.getBaseballTeams(this.allTeams)

val brewers: Team = Team("Milwaukee", "Brewers", Sport.Baseball)

teams.add(brewers) //add doesn't exist on List
```

```
val teams: MutableList<Team> = this.getBaseballTeams(this.allTeams)

val brewers: Team = Team("Milwaukee", "Brewers", Sport.Baseball)

teams.add(brewers) //Brewers added
```

```
val teams: List<Team> = listOf(brewers, reds, cubs)

val mutableTeams: MutableList<Team> = mutableListOf(brewers, reds, cub
```

KOTLIN ON ANDROID

KOTLIN ON ANDROID

ONCLICKHANDLERS

```
// Java
loginButton.setOnClickListener(new View.OnClickListener() {
    public void onClick(View v) {
        logInUser();
    }
});
```

```
// Kotlin
loginButton.setOnClickListener {v -> logInUser() }

loginButton.setOnClickListener { logInUser() }
```

KOTLIN ON ANDROID

LATEINIT

```
private lateinit var logo: ImageView
```

```
this.logo = this.findViewById<ImageView>(R.id.logo)
```

KOTLIN ON ANDROID

ANKO

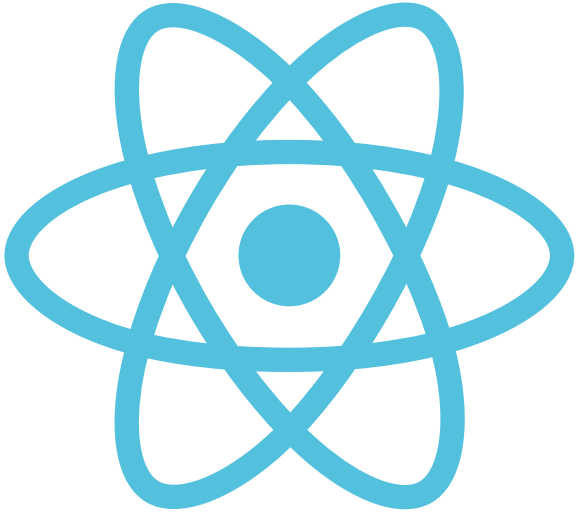
- Anko Commons
- Anko Layouts
- Anko SQLite
- Anko Coroutines

```
verticalLayout {
    val name = editText()
    button("Say Hello") {
        onClick { toast("Hello, ${name.text}!") }
    }
}
```

https://github.com/Kotlin/anko

KOTLIN AND JAVASCRIPT

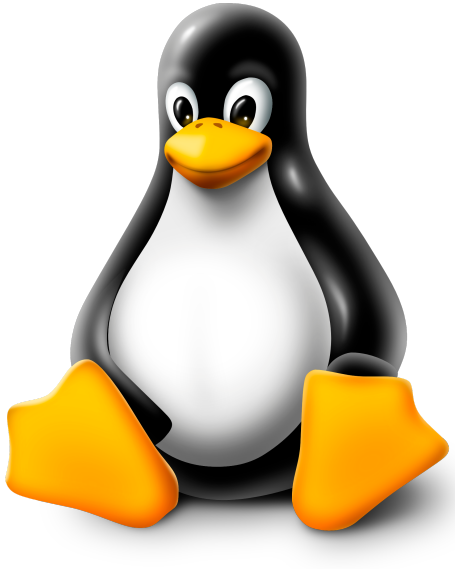
KOTLIN AND REACT



https://github.com/JetBrains/create-react-kotlin-app
https://github.com/JetBrains/kotlin-wrappers

KOTLIN NATIVE

DESKTOP



KOTLIN NATIVE

IOS

KOTLIN NATIVE

ANDROID

QUESTIONS?

THANK YOU!

Kotlin

@faziodev

https://github.com/MFazio23


THAT
CONFERENCE
SEE YOU NEXT YEAR!
AUGUST 5 - 7, 2019