

# Kotlin Structure

---



**Kotlin**





" Most people talk about Java the language, and this may sound odd coming from me, but I could hardly care less. At the core of the Java ecosystem is the JVM. "

**James Gosling,**

Creator of the Java Programming Language(2011, TheServerSide)



# JVM Language History





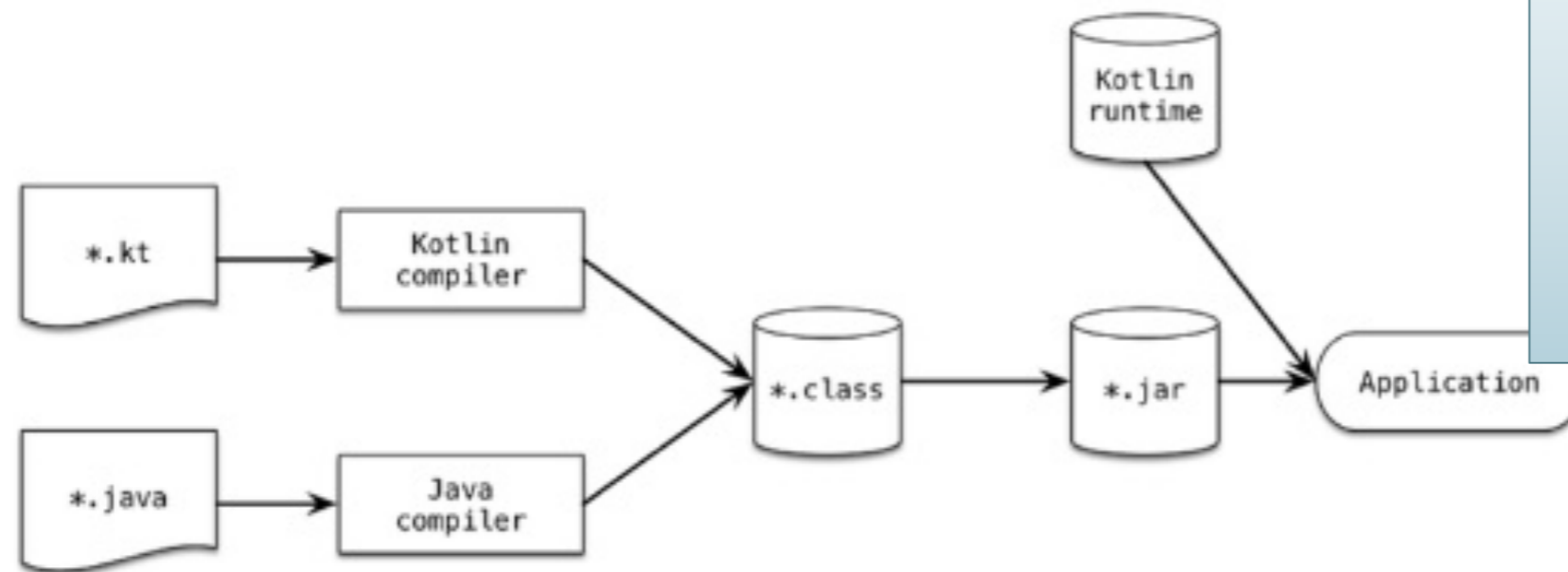
# Kotlin

---

- runs on Java Virtual Machine.
- is an evolution of the Java syntax but is more concise and has cleaner syntax.
- is not syntax compatible with Java; but is interoperable with Java.
- relies on some Java Class Libraries e.g. Collections framework.
- is a statically-typed programming language.
- offers null safety.

# Runs on Java Virtual Machine

## Kotlin and the JVM

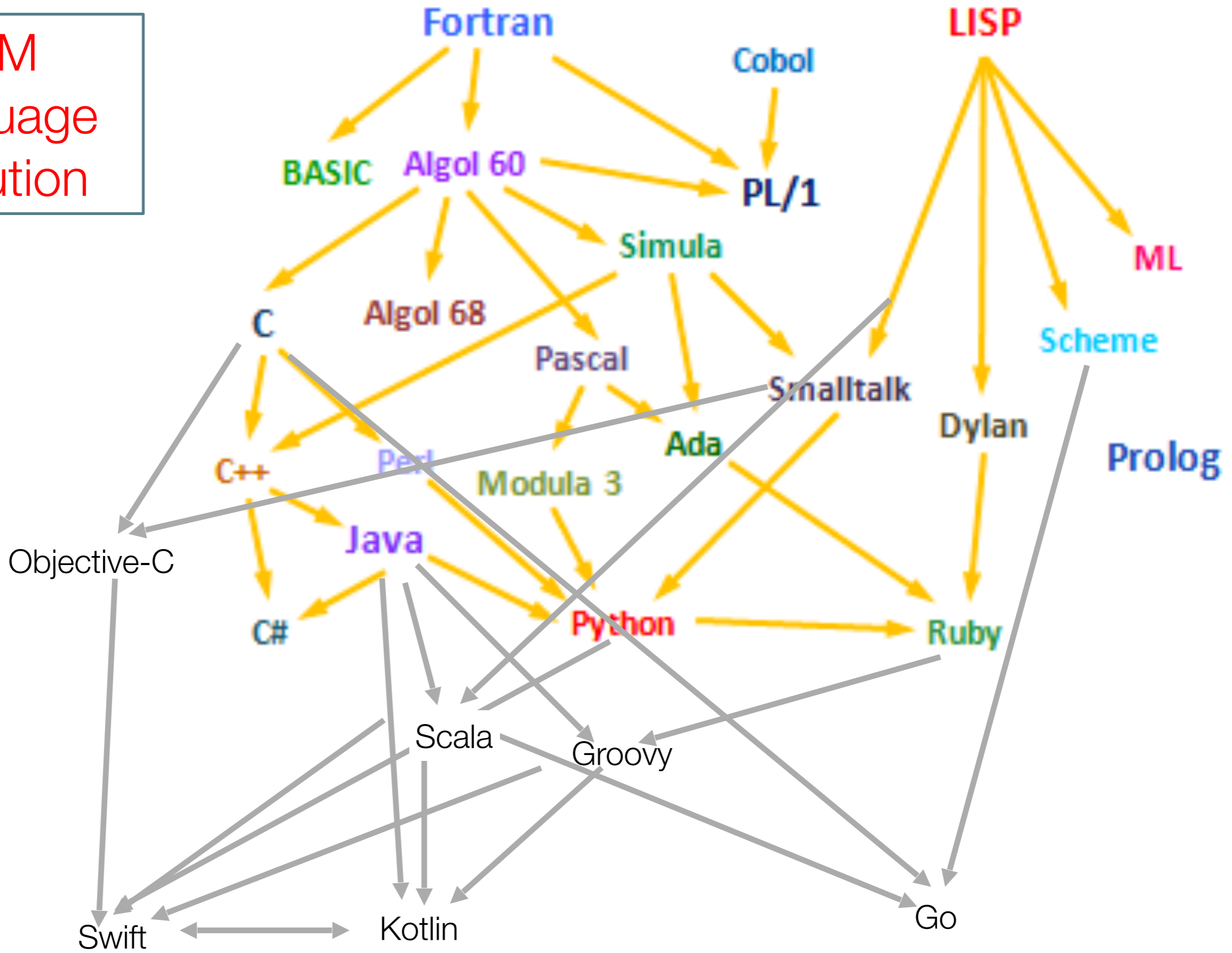


\* Interoperable 100%

Kotlin compiles to JVM ByteCode (like Java)

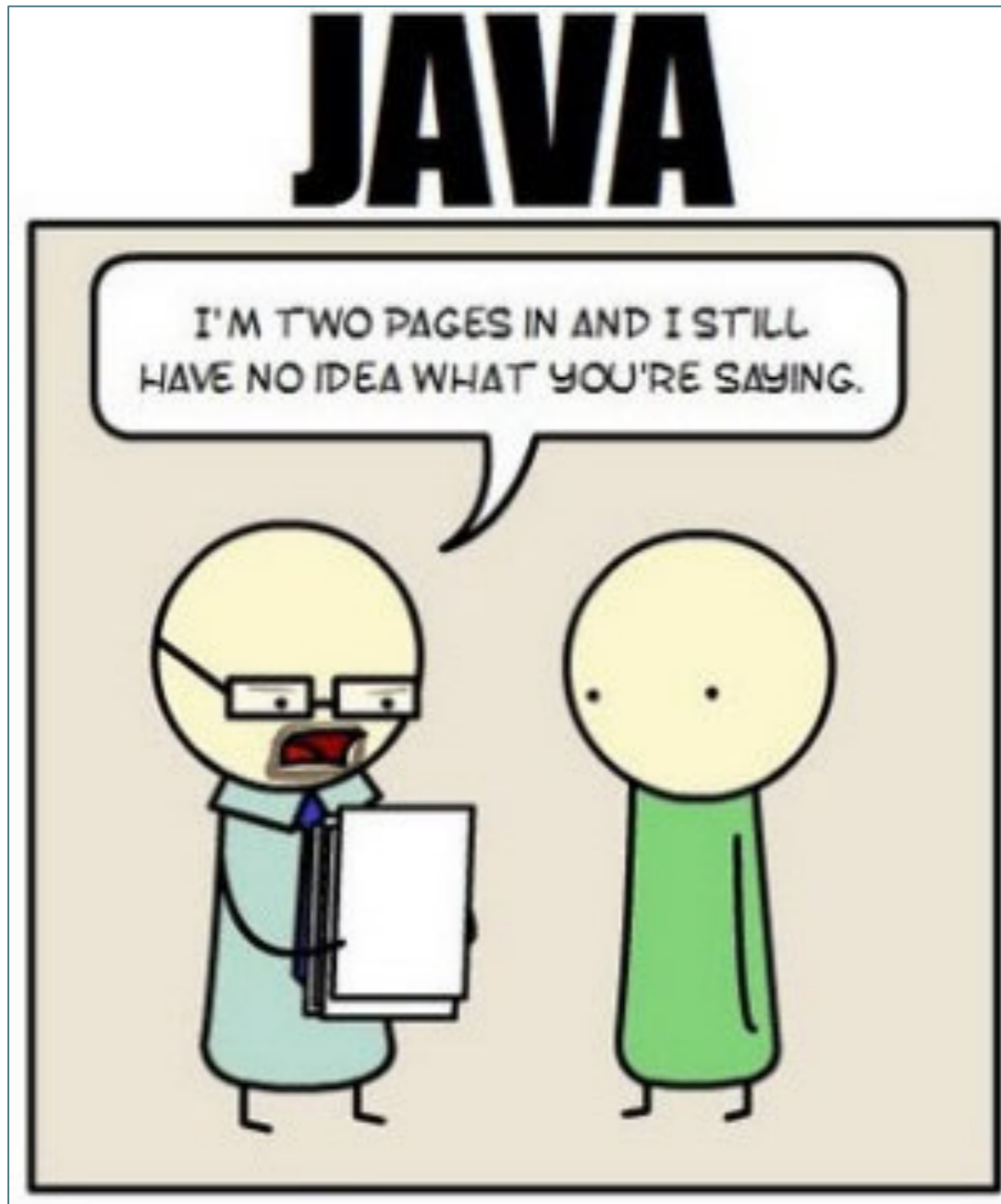
*Note: Kotlin also compiles to JavaScript*

JVM  
Language  
Evolution



# Kotlin (Concise) Vs Java (Overly Verbose)

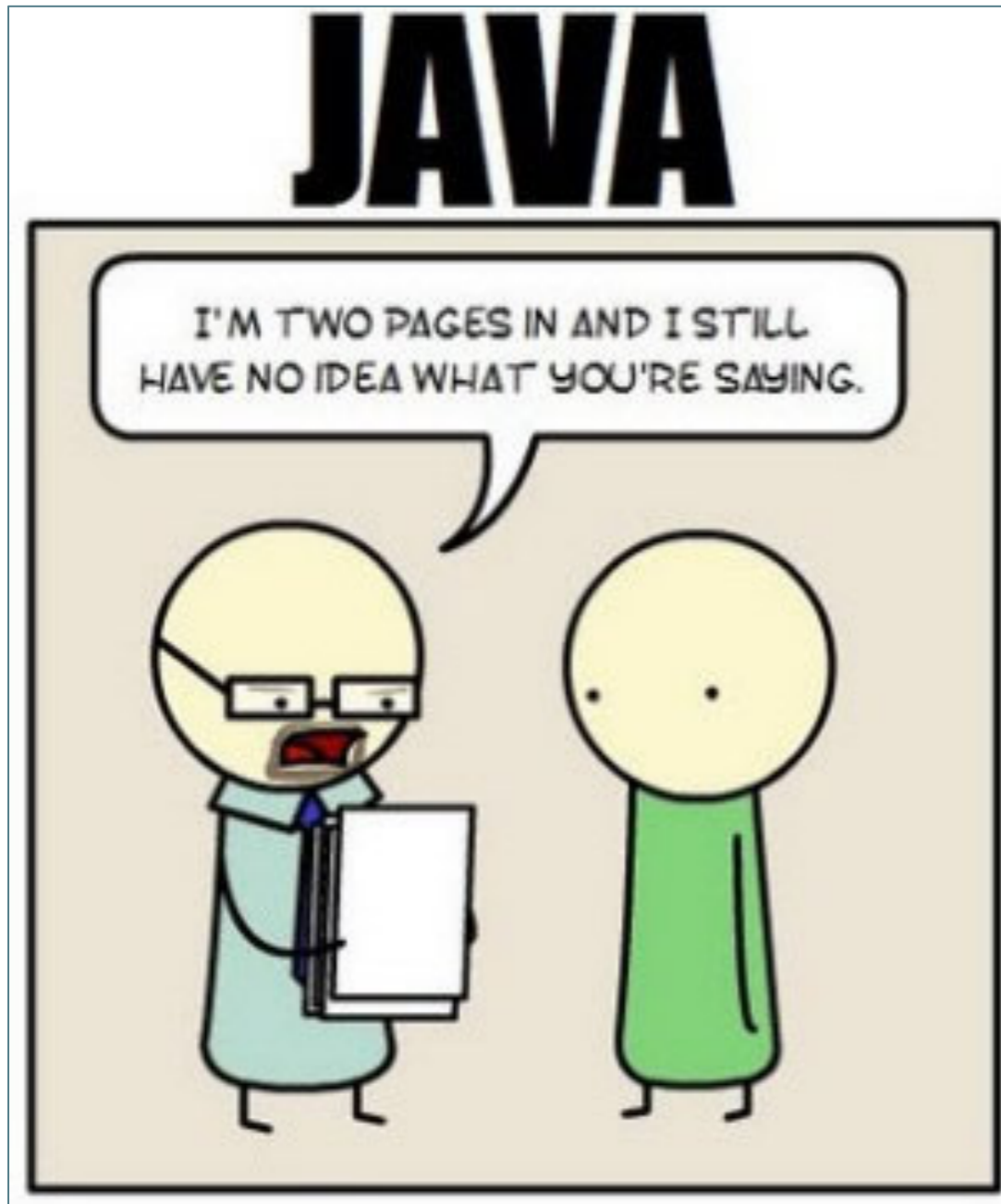
---



*“Java is extremely verbose and ceremonious. Programmers need to write reams of code to get a simple task done. There's a great deal of ‘ceremony’ in Java APIs, and Android aggravates this by forcing developers to go through many steps, in a specific order, to get things done.”*

# Kotlin (Concise) Vs Java (Overly Verbose)

---



*“Java code is verbose...”*

*“Kotlin provides a well thought-out syntax and extensive standard library that removes many of the pain points that exist in Java.”*

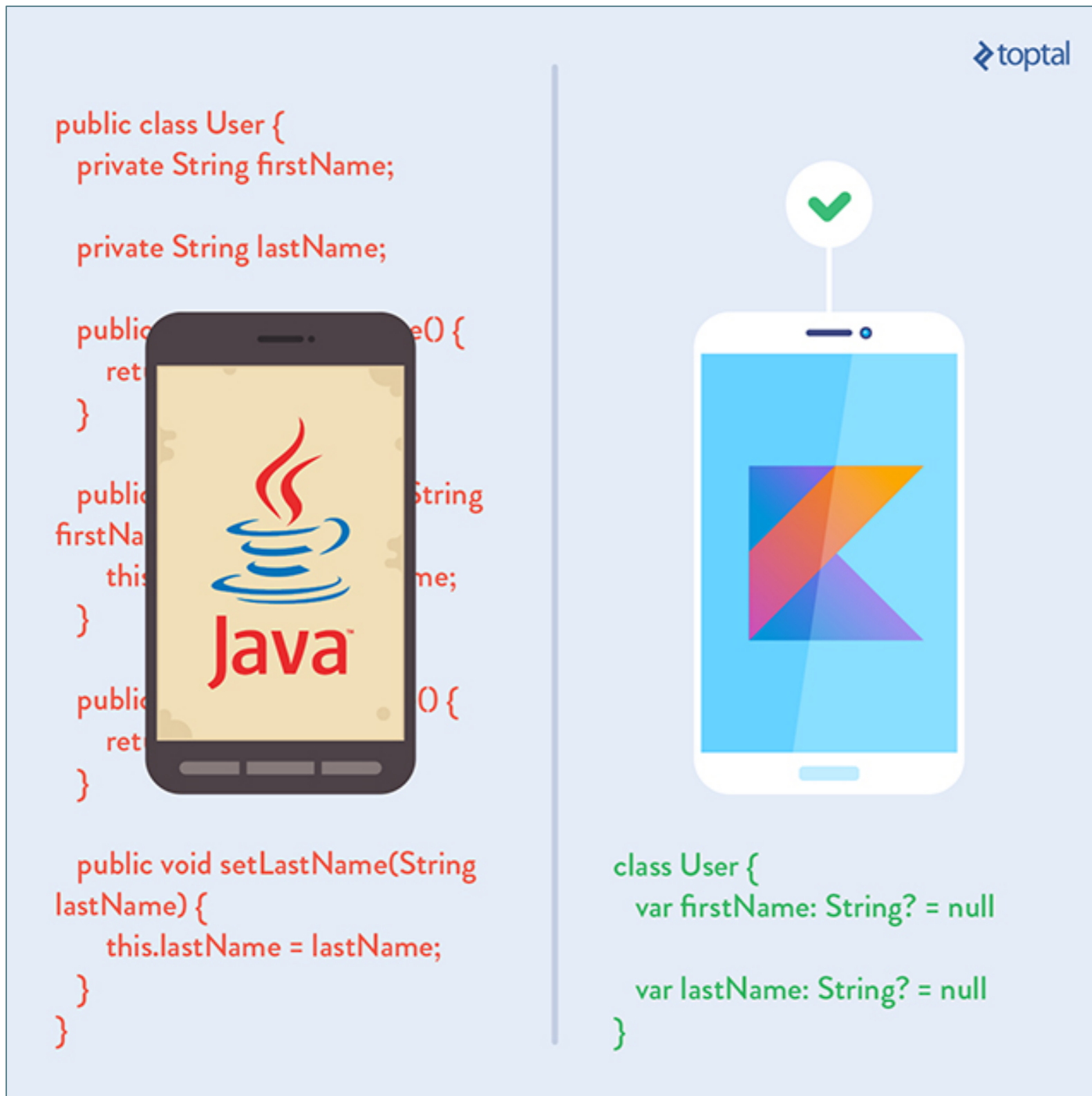


# Kotlin (Concise) Vs Java (Overly Verbose)

---

*Rough estimates  
indicate approximately  
a 40% cut in the  
number of lines of  
code.*

# Kotlin (Concise) Vs Java (Overly Verbose)



toptal

```
public class User {  
    private String firstName;  
  
    private String lastName;  
  
    public User() {  
        return this;  
    }  
  
    public String  
    firstName  
    this  
    }  
  
    public  
    return  
    }  
  
    public void setLastName(String  
    lastName) {  
        this.lastName = lastName;  
    }  
}
```

```
class User {  
    var firstName: String? = null  
  
    var lastName: String? = null  
}
```

Kotlin drastically reduces the amount of boilerplate code you have to write.

The less code you write, the fewer mistakes you make, the less to test, the better the execution.

# Easy(ish) learning curve for Java Developers

---

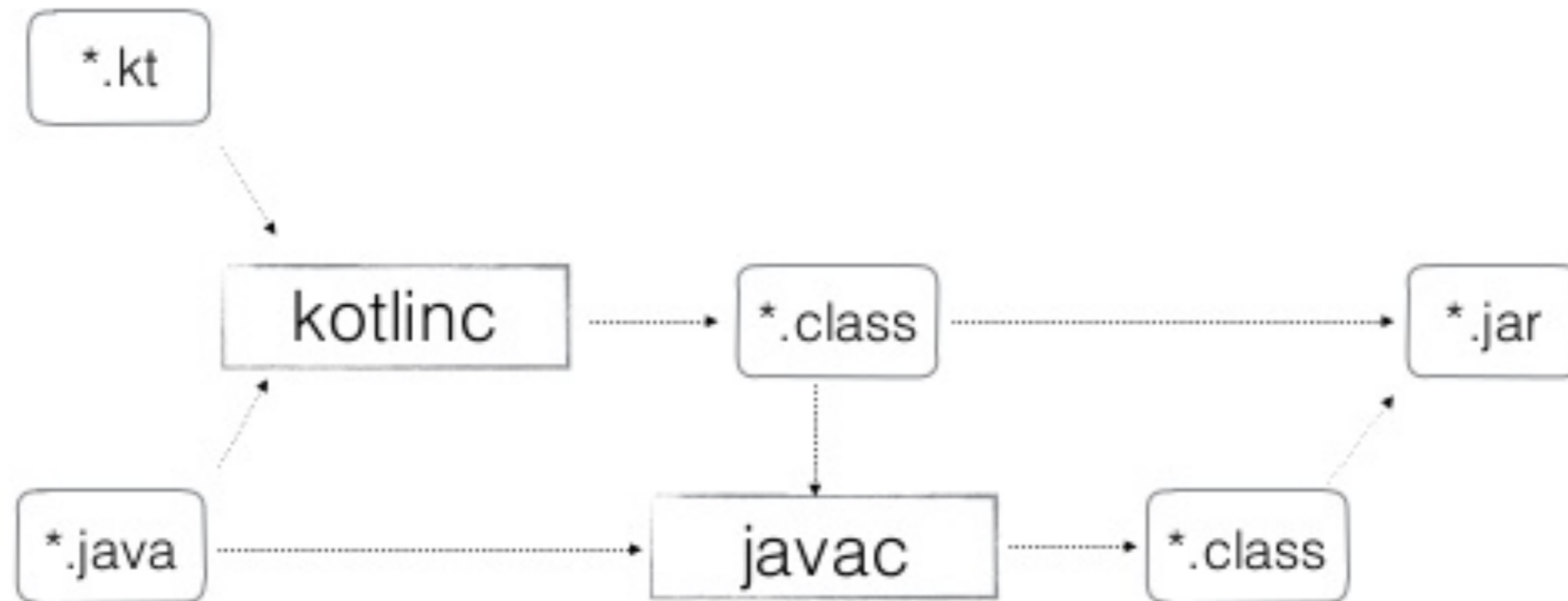
*“Kotlin is approachable and can be acquired in a few hours by simply reading the language reference. It has a lean and intuitive syntax.”*

*“Kotlin is also designed to have a gentle learning path for Java developers. Java programmers will find that most of the Kotlin syntax feels familiar.”*

# Kotlin / Java Interoperability

---

## Compilation of a mixed project



# Kotlin / Java Interoperability

---

- Kotlin and Java are 100% interoperable; Java and Kotlin code can co-exist very well in the same project and compile perfectly.
- Kotlin can be called from Java and Java from Kotlin.
- Both .java and .kt files are compiled to .class bytecode.
- When a project containing both Java and Kotlin is compiled, it would be difficult to tell which parts were created in Java and which in Kotlin.
- You can start using Kotlin in an existing Java project, without having to convert the project to Kotlin.

# Kotlin and the Java Class Libraries

---

Interoperability advantages:

- you can use any of the vast number of Java Libraries and Frameworks in a Kotlin project.
- Kotlin can also easily integrate with Maven, Gradle and other build systems.