Introduction

Kotlin is a statically-typed, cross platform, general purpose programming knowledge that was designed to be fully interoperable with Java.

History

Kotlin was developed by JetBrains, the company behind popular integrated environments (IDEs) like IntelliJ IDEA.

It was first announced in 2011 but was officially released in 2016.

Kotlin gained prominence as an official language for Android App development in 2017 when Google endorsed, alongside Java.

Difference from Java:

1. Conciseness: Kotlin code is often more concise than equivalent Java code. It reduce boilerplate code.

making development faster and easier to read.

2. Null safety: kotlin type system includes nullable and non-nullable types, which help prevent null pointer exceptions. a common issue in java.

3. Extension functions: kotlin allow you to add new functions to existing classes without modifying their source code. The feature is not present in java.

4. Smart Casts: kotlin's type system allows for automatic casting of types in certain situations, reducing the need for explicit type casting.

5. Data classes: kotlin provides data classes that automatically generate useful methods like 'equals()', 'hashCode()' and 'toString()'.

6. Coroutines: Kotlin offers native support for Coroutines, simplifying asynchronous programming compared to Java's thread-based model.

Purpose

- Kotlin was created to address several pain points in Java development and improve overall productivity.
- It aimed to maintain compatibility with Java, making it easier for Java developers to transition to Kotlin and allowing both languages to exist in the same Code Base.

Kotlin was designed to be a more modern, expressive, and safer alternative to Java while still leveraging the vast Java ecosystem and libraries.

Key Features

Interoperability:
Kotlin can seamlessly interoperable with Java, allowing developers to use both languages in the same project.

Null safety:
Kotlin type system distinguishes between nullable and non-nullable types, reducing null pointer Exceptions.

Extension functions:
This feature allow you to Extend existing classes with new functionality without altering their source code.

Data classes:
Data classes in Kotlin automatically generate useful methods for working with data, such as 'equals()' 'hashCode()' and 'toString()'.

Coroutines:

Kotlin provides native support for asynchronous programming through Coroutines simplifying concurrent and asynchronous code.

Functional Programming:

Kotlin support functional programming concepts like lambdas, higher-order functions and immutable data structures.

Conclusion

Kotlin was developed to improve upon Java by offering a more concise, expressive and safer language while maintaining compability with Java. Its modern features and emphasis on null safety make it popular choice for android development and other application domain.

print Helloworld in kotlin

Step1:-

Create a file Hello.kt

Step2:-

write following code

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

Step 3:-

Run the code with following command

Hello.kt -include-runtime -d Hello.jar

Step 4:-

Hello world.