Kotlin para el desarrollo de aplicaciones

El entorno Kotlin

{desafío} latam_

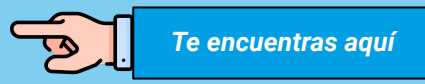
¿Qué aprenderemos en este módulo?

En este módulo, el estudiante conocerá las principales características del lenguaje Kotlin para el desarrollo de aplicaciones móviles Android Nativo. Utilizará elementos del ciclo de vida para la implementación de un aplicativo Android, además de patrones de arquitectura escalables para la construcción de una aplicación Android. Por último, empleará elementos de la programación asíncrona acorde al lenguaje Kotlin.

{desafío} latam_

Reconocer las principales características del lenguaje Kotlin para el desarrollo de aplicaciones móviles Android Nativo.

- Unidad 1: Kotlin para el desarrollo de aplicaciones.



- Unidad 2: Ciclo de vida de componentes.
- Unidad 3: Arquitectura en Android.
- Unidad 4: Programación asíncrona en Android.

{desafío} latam_

¿Qué aprenderás en esta sesión?

- *Reconocer las principales características del lenguaje Kotlin para el desarrollo de aplicaciones Android*

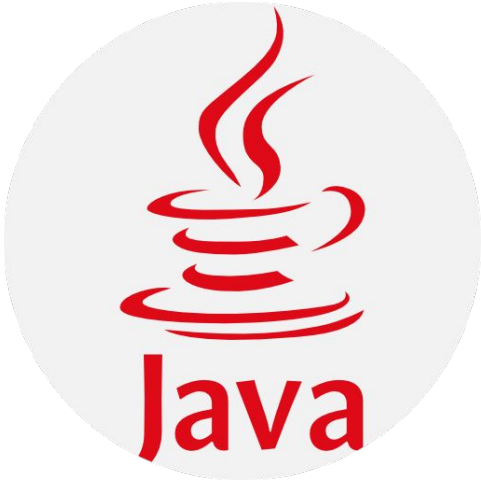
{desafío} latam_

/* Kotlin y Java, la interoperabilidad */

¿Qué es Java?

Java es un lenguaje de programación multiplataforma orientado a objetos que se ejecuta en miles de millones de dispositivos de todo el mundo. Impulsa aplicaciones, sistemas operativos de smartphones, software empresarial y muchos programas conocidos.

A pesar de haber sido eliminado hace más de 20 años, Java es actualmente el lenguaje de programación más popular para los desarrolladores de aplicaciones.



Fuente:
¿Qué es Java?. Disponible en:
https://azure.microsoft.com/es-es/resources/cloud-computing-dictionary/what-is-java-programming-language/

¿Qué es Kotlin?

Por otro lado, Kotlin es un lenguaje de programación multiplataforma, tipificado estáticamente y de propósito general, con inferencia de tipos. Está diseñado para interoperar completamente con Java, lo que lo hace el lenguaje ideal para trabajar con Android, ya sea en nuevos proyectos o existentes

¿Qué significa que Java y Kotlin sean interoperables?

De forma resumida significa que:

- Puedes utilizar Koltin y Java en un mismo proyecto, lo que hace posible que la migración de un proyecto originalmente escrito en Java, sea más amigable.
- Puedes usar librerías externas, desarrolladas en Java, en un proyecto Kotlin y viceversa.

Tip: Android Studio provee las herramientas necesarias para la conversión de código Java a Kotlin

/* Stackoverflow */

Stackoverflow será tu mejor aliado

- Stackoverflow es uno de los mejores sitios web existentes.
- En él puedes encontrar la mayor cantidad de información relacionada a problemas de diversa complejidad, en formato pregunta/respuesta.

Stackoverflow será tu mejor aliado

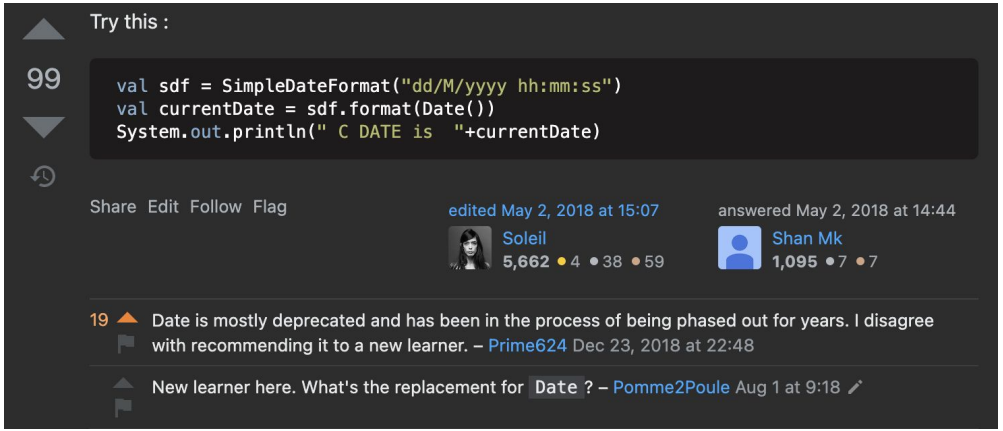
¿Qué tan confiables son las respuestas?

- Muchas veces se pueden encontrar respuestas de los ingenieros de Google que trabajan con Android (AOSP).
- Dependiendo de cuán antigua sea la pregunta, la respuesta podría estar pensada en código Java.
- Usualmente las respuestas son actualizadas, por lo que podrás encontrar la solución en Kotlin.

Tip: se recomienda encarecidamente buscar respuestas y hacer preguntas en inglés. La cantidad y la calidad de las respuestas son mucho mejores.

Ejemplo de pregunta y respuesta en Stackoverflow

El siguiente ejemplo es una pregunta muy común. A pesar de haber varias respuestas, hay una en particular que se destaca por la cantidad de puntos otorgados por los otros usuarios.



{desafío} latam_

Tip: Fíjate siempre en cuál año se hizo la pregunta. No siempre la respuesta marcada como “Aceptada” es la mejor.

/* Diferencias de Kotlin con Java */

El código

Una de las diferencias clave entre Kotlin y Java, es que Kotlin requiere mucho menos código. Es un lenguaje muy conciso, lo que reduce las posibilidades de cometer errores de código y simplifica el trabajo de los desarrolladores.

En general, la brevedad de Kotlin hace que sea más manejable escribir proyectos grandes, pues requiere menos líneas de código para escribir exactamente las mismas funciones. Además, sabe cómo hacerlo breve y directo sin comprometer la legibilidad de la sintaxis.

Diferencias de Kotlin con Java

Ejemplo

En este ejemplo se puede ver cómo ocupar la clase “**User**” escrita en Java desde “**MainActivity**” escrito en Kotlin

Java class

```java
public class User {

    private String firstName;

    public String getFirstName() {
        return firstName;
    }

    public void setFirstName(String firstName) {
        this.firstName = firstName;
    }
}
```

Kotlin class

```kotlin
class MainActivity : AppCompatActivity() {
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_main)

        val user = User()
        user.firstName = "Nombre"
    }
}
```

Comparemos...

	Kotlin	Java
Null Safety	No es posible atribuir valores nulos a variables u objetos, si no son definidos como nullable	**NullPointerExceptions**: permite atribuir un valor nulo a cualquier variable.
Extension Functions	Permite ampliar la funcionalidad de las clases sin tener que heredar necesariamente de una clase.	Para ampliar la funcionalidad de una clase existente, se debe crear una nueva clase y heredar las funciones de la clase principal.
Smart Casts	El compilador inteligente de Kotlin administra automáticamente las conversiones redundantes.	El desarrollador debe verificar el tipo de variables en consonancia con la operación.

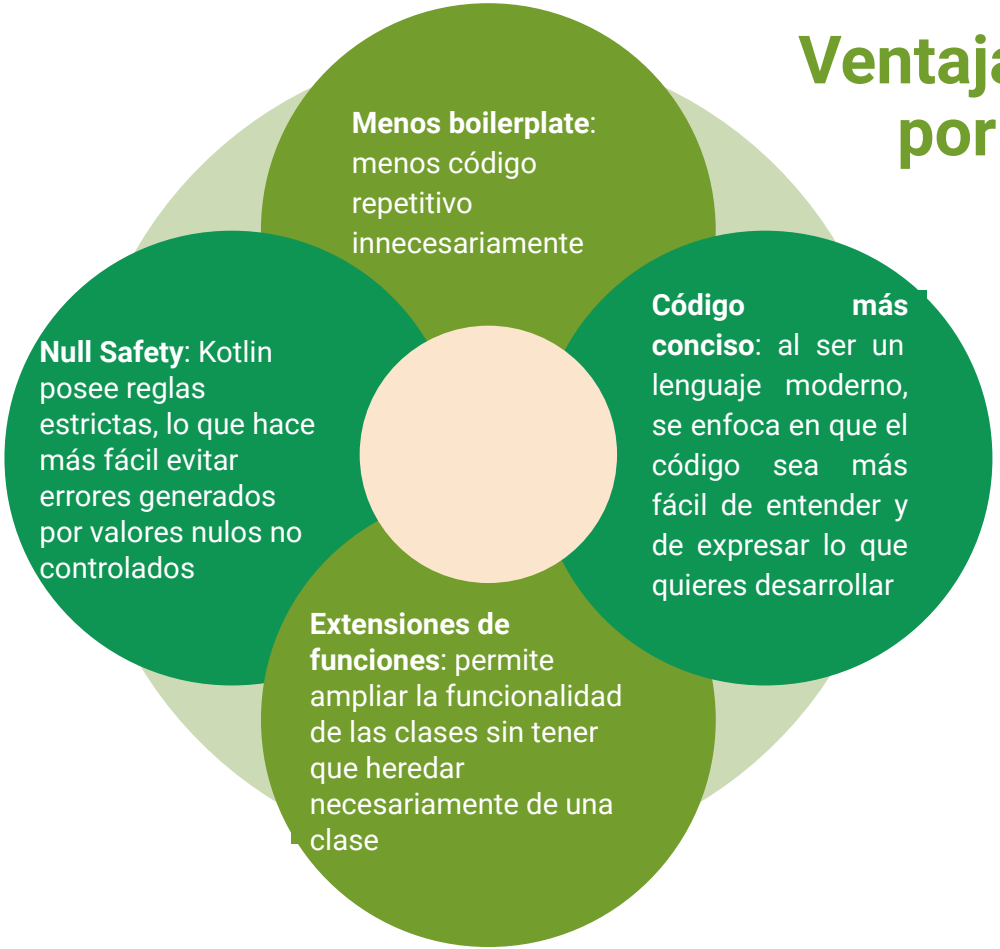
Comparemos…

	Kotlin	Java
Functional Programming	Kotlin es una mezcla de programación orientada a objetos y funcional. Puede usar expresiones lambda y funciones de alto orden.	Java está más limitado a la programación orientada a objetos.
Data Classes	Kotlin proporciona una forma más sencilla de crear clases para almacenar datos, simplemente incluyendo la palabra clave “datos” en la definición de la clase.	Los desarrolladores deben establecer los campos (o variables) para almacenar los datos, el constructor y las funciones getter y setter para los campos/variables, así como otras funciones, como hashCode(), equals() y toString( ).

/* Ventajas de Kotlin por sobre Java */

Ventajas de Kotlin por sobre Java

Como ya pudiste ver en una tabla anterior, se mostraron algunas de las diferencias entre Java y Kotlin, ahora veremos en más detalle a que se refiere eso:



Menos boilerplate

Clase “**User**” en Java:

```
public class User {

    private String firstName;

    public String getFirstName() {
        return firstName;
    }

    public void setFirstName(String firstName) {
        this.firstName = firstName;
    }
}

User user = new User();
user.setFirstName("Nombre");
printLn(user.getFirstName);
```

Clase “**User**” en Kotlin:

```
data class User(val firstName: String)

val user = User("Nombre")
println(user)
```

Código más conciso

```
val list = arrayListOf<KPet>()
    list.add(KPet(name = "Lucky", type = PetType.OTHER))
    list.add(KPet(name = "Luna", type = PetType.CAT))
    list.add(KPet(name = "Bodoque", type = PetType.DOG))
    list.add(KPet(name = "Cuchito", type = PetType.CAT))

val filteredList = list.asSequence()
    .filter { it.type == PetType.CAT }
    .sortedWith(compareBy {
        it.name == "Cuchito"
    }).toMutableList()
println("lista de mascotas: $filteredList")


resultado:
lista de mascotas: [KPet(id=0, type=CAT, name=Cuchito, nickName=null)]
```

Crea datos de prueba

Toma los datos de prueba y los transforma en una secuencia, luego los filtra por “PetType.CAT”, luego los ordena y compara con el nombre de mascota “Cuchito”, finalmente, imprime el resultado

Null Safety

En Kotlin no está permitido asignar valores nulos por defecto:

```
val firstName: String = null
```

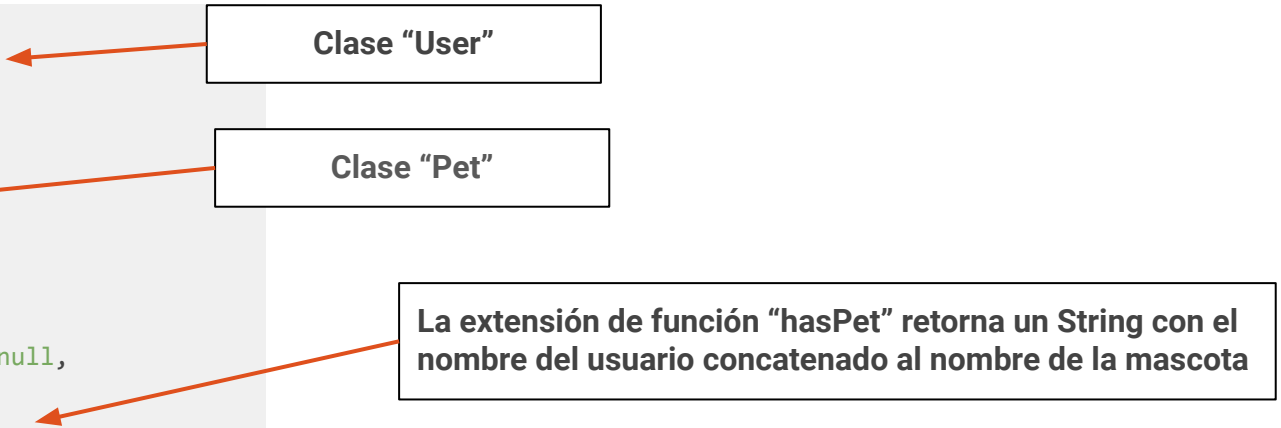
Android Studio mostrará un error indicando "Null can not be a value of a non-null type String"

Sin embargo, cuando es necesario asignar un valor nulo, la variable se debe crear tipo nullable, para esto basta con agregar **"?"** al tipo de variable, en este caso String. Esto permitirá asignar a firstName un valor string o null.

```
val firstName: String? = null
```

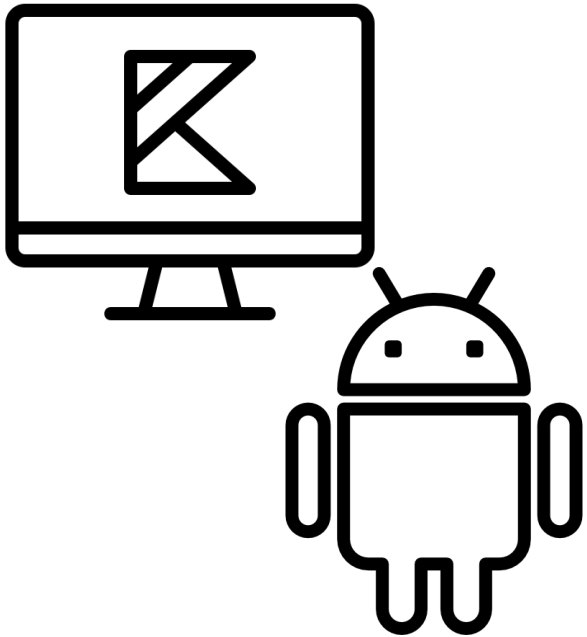
Extension Functions

```
data class User(
    val firstName: String,
    val pet: Pet,
)

data class Pet(
    val id: Int? = 0,
    val type: PetType,
    val name: String,
    val nickName: String? = null,
)

fun User.hasPet(): String {
    return "${this.firstName} tiene una
mascota que se llama ${this.pet.name}"
}
```



/* Integrando Kotlin a un proyecto en Android */

Kotlin en un proyecto Android



El 7 de mayo de 2019, Google anunció que Kotlin sería el lenguaje de programación preferido para desarrollo de aplicaciones Android.

A partir de esa fecha, se sugirió empezar a utilizar Kotlin como lenguaje principal en cualquier nuevo proyecto, y migrar proyectos existentes.

Kotlin en un proyecto Android

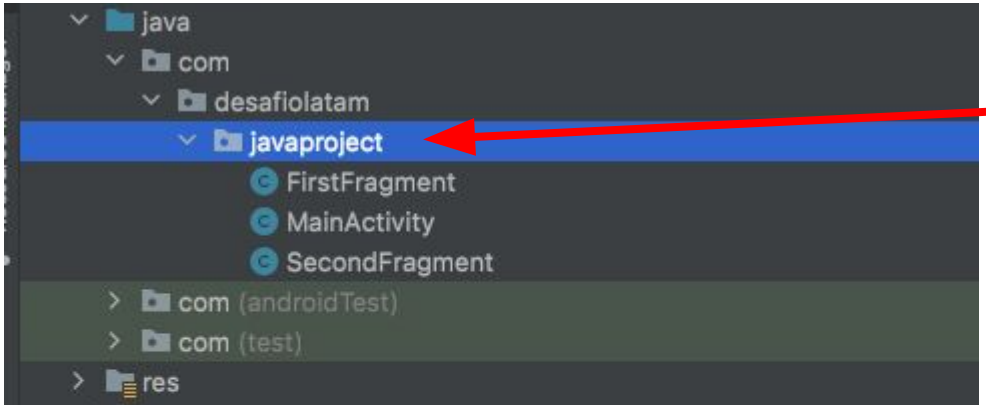
- Migrar un proyecto desarrollado en Java hacia Kotlin puede tomar tiempo. Sin embargo, debido a la interoperabilidad entre Java y Kotlin, se pueden ejecutar ambos lenguajes al mismo tiempo, y realizar una migración paulatina.

- En las últimas versiones de Android Studio, un nuevo proyecto, por defecto, es creado en Kotlin. Sin embargo, aún es posible seleccionar Java como lenguaje principal.

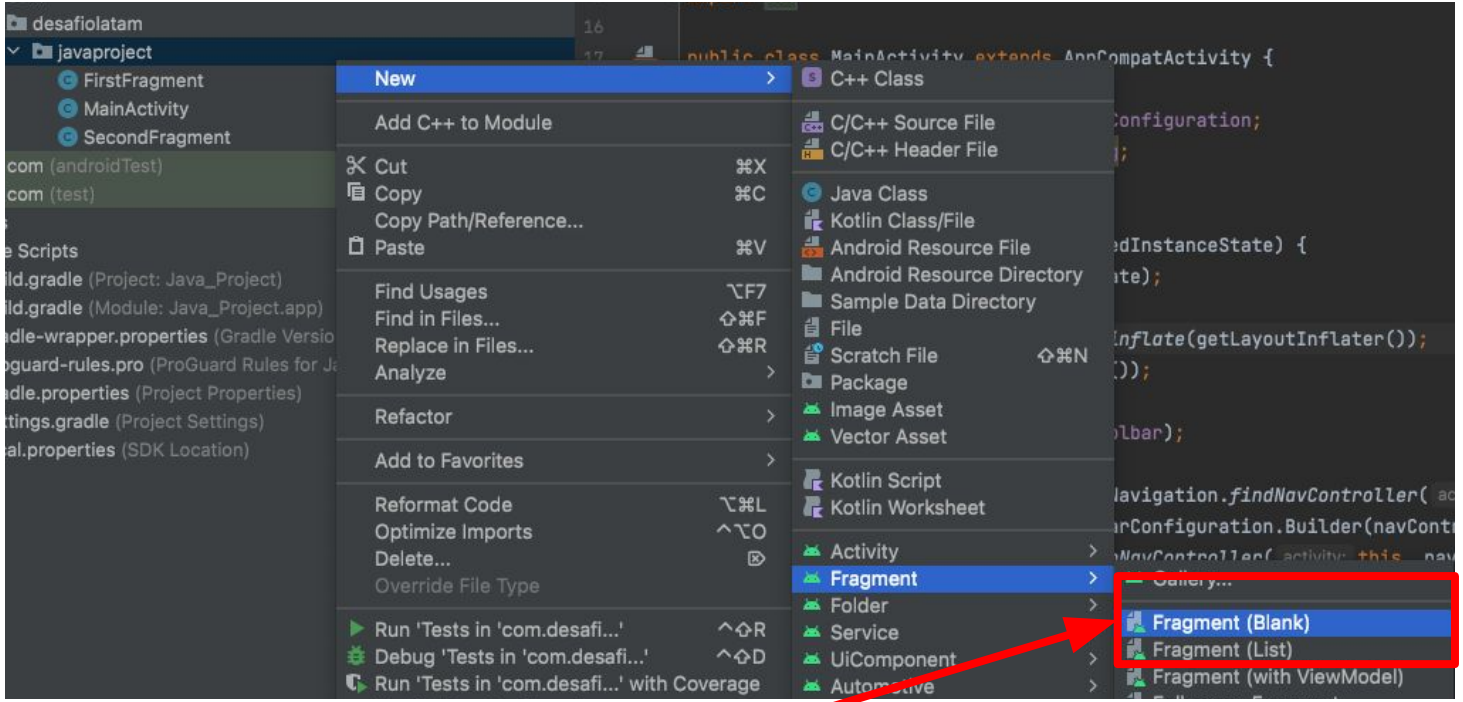
Tip: si estás utilizando Android Studio, copia y pega un código hecho en Java, en un archivo Kotlin, verás lo fácil y amigable que es Android Studio para migrar el código.

Integrando Kotlin a un proyecto Java

Una de las formas más simples de habilitar Kotlin en un proyecto Java, es simplemente creando un nuevo activity o fragment y seleccionando el lenguaje en el cual se desea desarrollar.
Por ejemplo:

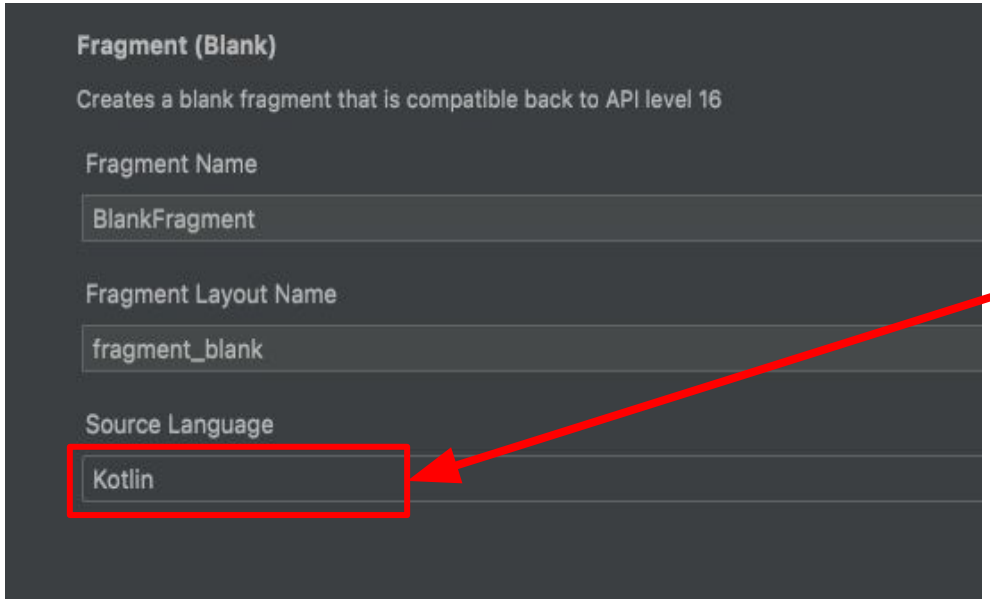


Click derecho sobre el paquete al cual se le agrega el nuevo fragment/activity



Podemos seleccionar un “**fragment (blank)**”

{desafío} latam_



Seleccionamos como “Source language” **Kotlin,** y listo, ahora Android Studio descargará y configurará el proyecto para hacerlo compatible con Kotlin

Si tuvieras que crear un proyecto Android desde cero, ¿utilizarías Java o Kotlin?

Después de haber visto algunas de las características de Kotlin, ¿cuál llamó más tu atención? ¿Por qué?

¿Te fijaste que en Kotlin no es necesario escribir un “;” al final de cada línea? ¿Por qué piensas que ocurre eso?

Próxima sesión...

- *Utilizar la sintaxis del lenguaje Kotlin para la definición de variables, clases y funciones*

{desafío} latam_

{desafío}
latam_

Academia de
talentos digitales