

MÉTODO DE PROYECTOS DE ENSEÑANZA - APRENDIZAJE



**PLAN DE TRABAJO DEL
ESTUDIANTE**

Desarrollo de mi Primera App Android en Java

Objetivo:

Desarrollar una aplicación en Android Studio utilizando Java, donde los estudiantes instalen y configuren el entorno, exploren la estructura del proyecto y revisen estructuras de control en Java.

Parte 1: Instalación y Configuración

1. Descargar Android Studio desde la página oficial de Google.
2. Instalar Android Studio siguiendo los pasos del asistente.
3. Configurar un Emulador en el AVD Manager.
4. Crear un Nuevo Proyecto con la plantilla "Empty Activity".

Parte 2: Estructura del Proyecto en Android Studio

Explica a los alumnos la organización del proyecto:

- Manifests/ → Archivo `AndroidManifest.xml`, donde se definen permisos y configuración básica.
- java/ → Código fuente, incluyendo la actividad principal.
- res/ → Recursos gráficos, layouts y strings.
- Gradle Scripts/ → Archivos de configuración del proyecto.

Parte 3: Estructuras de Control en Java

Antes de modificar la app, repasen estructuras como:

- Condicionales (`if`, `switch`)
- Bucles (`for`, `while`)
- Manejo de eventos (`onClickListener`)

Parte 4: Proyecto Base (Ejemplo a Mejorar)

Se proporciona un ejemplo de app con un botón que cambia el texto de un `TextView` al ser presionado.

Código Base (MainActivity.java)

```
java

package com.example.miapp;

import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.TextView;
import androidx.appcompat.app.AppCompatActivity;
```

```

public class MainActivity extends AppCompatActivity {
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        TextView textView = findViewById(R.id.texto);
        Button button = findViewById(R.id.boton);

        button.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                textView.setText("¡Hola, Android!");
            }
        });
    }
}

```

activity_main.xml (Diseño de la interfaz)

xml Copiar Editar

```

<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical"
    android:gravity="center">

    <TextView
        android:id="@+id/texto"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Texto original"
        android:textSize="24sp"/>

    <Button
        android:id="@+id/boton"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Presionar"/>

</LinearLayout>

```

Entregable 01:


- Cambiar colores y estilos.
- Agregar un nuevo botón que cambie el color del TextView.
- Utilizar una estructura de control (if) para alternar entre dos textos.

Entregable 02:

- Cambio de colores y estilos dinámicos.
- Uso de SharedPreferences para guardar datos entre sesiones.
- Uso de Random para cambiar colores de fondo.
- Agregar un EditText para que el usuario pueda ingresar su nombre.
- Mostrar un Toast con el nombre ingresado.
- Alternar texto con estructuras de control (if-else).
- Implementar un Snackbar en lugar de un Toast.

Entrega Final

- Cada alumno sube su código a un repositorio en GitHub y comparte el enlace.
- Incluir un video corto demostrando la funcionalidad.
- Agregar comentarios en el código explicando cada mejora realizada.

 SENATI	INGENIERÍA DE SOFTWARE CON INTELIGENCIA ARTIFICIAL	CÓDIGO DEL PROYECTO: HT-01	
	<i>HT-01: Utiliza el entorno Android Studio y Java</i>	TIEMPO: 4h 45min	Nº PAG: 2
		FECHA: 15/02/2025	

2. INFORMACIÓN PARA EL DESARROLLO DEL PROYECTO

Carrera:	Ingeniería de Software con Inteligencia Artificial		
Módulo Ocupacional:	Ingeniería de Software		
Módulo Formativo:	SEM DE COMPLEMENT PRÁCTICA III		
Proyecto:	Desarrollo de un App Android en Java	Código:	HT-1
Duración:	7h 00 min		
Instructor:	VARGAS MORALES, FRANZ DENIS		

2.2. OBJETIVOS

Considerando la información de los enlaces, videos y los recursos adicionales, el aprendiz estará en la capacidad de desarrollar un chatbot básico utilizando Google Colab para entrenar el modelo, implementar una interfaz de usuario atractiva con Streamlit donde los usuarios puedan interactuar con el chatbot, y utilizar ngrok para exponer la aplicación localmente para que pueda ser accesible desde cualquier parte del mundo a través de Internet.

2.3 CONTENIDO SINTÉTICO A DESARROLLAR

Modelo de Lenguaje Preentrenado para Chatbots.
Implementación de la Interfaz de Usuario con Streamlit.
Exposición Local de la Aplicación con Ngrok.
Desafíos y Mejoras en la Personalización del Chatbot.

2.4. CUESTIONARIO GUIA

Nº	PREGUNTAS
1	¿Cuales son las principales ventajas de usar kotlin sobre Java en el desarrollo de aplicaciones Android con Android Studio?
2	¿Que es una coroutine en kotlin y como se utiliza para manejar operaciones asíncronas en una aplicación Android?
3	¿Como puedes implementar un RecyclerView en Android Studio utilizando kotlin y cuál es la función del adapter en este contexto?

2.5. BIBLIOGRAFÍA

Nº	BIBLIOGRAFIA – SITIOS WEB
2.5.1	Tu primer programa en Kotlin Android Developers. (s. f.). Android Developers. https://developer.android.com/codelabs/basic-android-kotlin-compose-first-program?hl=es-419&utm_source=Guía de interoperabilidad de Kotlin-Java . (s. f.). Android Developers. https://developer.android.com/kotlin/interop?hl=es-419&utm_source=Coroutines Kotlin
2.5.2	Coroutines Kotlin. (s. f.). Kotlin Help. https://kotlinlang.org/docs/coroutines-overview.html
2.5.3	Corrutinas de Kotlin en Android. (s. f.). Android Developers. https://developer.android.com/kotlin/coroutines?hl=es-419 <i>Kotlin for Android / Kotlin</i> . (s. f.). Kotlin Help. https://kotlinlang.org/docs/android-overview.html

2.5.4	Zhang, Y., & Zhao, L. (2021). A Survey of Chatbot Development Technologies. International Journal of Computer Applications, 37(3), 46-55. Recuperado de: https://www.ijcaonline.org/



MÉTODO DE PROYECTOS DE ENSEÑANZA – APRENDIZAJE

3. HOJA DE RESPUESTAS AL CUESTIONARIO GUÍA

¿Cuáles son las principales ventajas de usar Kotlin sobre Java en el desarrollo de aplicaciones Android con Android Studio?

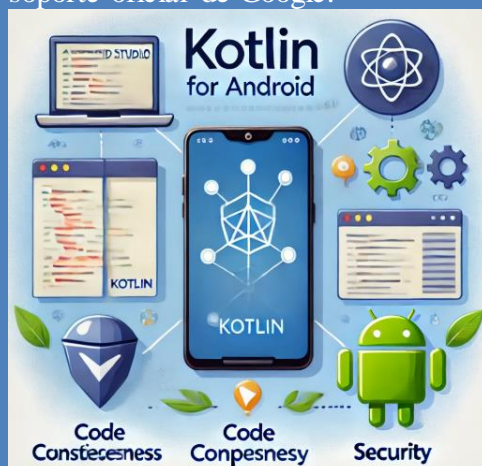
Kotlin es un lenguaje de programación moderno que ofrece varias ventajas sobre Java en el desarrollo de aplicaciones Android. Una de las principales es su concisión; con Kotlin, los desarrolladores pueden escribir menos líneas de código para lograr la misma funcionalidad, lo que mejora la legibilidad y el mantenimiento del código.

Además, Kotlin aborda de manera efectiva los problemas relacionados con las referencias nulas, una causa común de errores en Java conocidos como NullPointerException. Al incorporar un sistema de tipos que distingue entre referencias que pueden ser nulas y las que no, Kotlin ayuda a reducir estos errores en tiempo de ejecución, aumentando la estabilidad de las aplicaciones.

Ventajas:

La interoperabilidad es otra ventaja significativa. Kotlin está diseñado para integrarse perfectamente con el código Java existente, permitiendo a los desarrolladores adoptar Kotlin de manera gradual sin necesidad de reescribir todo el proyecto. Esta compatibilidad facilita la transición y el aprovechamiento de las bibliotecas y frameworks ya desarrollados en Java.

En resumen, la elección de Kotlin para el desarrollo de aplicaciones Android en Android Studio ofrece beneficios claros en términos de concisión, seguridad y compatibilidad, respaldados por el soporte oficial de Google.



¿Que es una coroutine en kottin y como se utiliza para manejar operaciones asíncronas en una aplicación Android?

Una coroutine en Kotlin es una técnica para manejar la concurrencia de manera más eficiente y sencilla. Estas funciones permiten pausar y reanudar su ejecución sin bloquear el hilo principal. Esto mejora el rendimiento de las aplicaciones, especialmente al realizar tareas asíncronas como solicitudes de red o acceso a bases de datos.

En Android, las corrutinas se utilizan para manejar operaciones asíncronas como solicitudes de red, lectura/escritura de archivos, y otras tareas que pueden tardar tiempo en completarse. Se pueden ejecutar en diferentes hilos utilizando los dispatchers (principal, IO, etc.).

Para usar coroutines en una aplicación Android:

CoroutineScope: Define el ámbito en el que se ejecutan las coroutines, controlando su ciclo de vida.

Dispatchers: Determina en qué hilo se ejecuta la coroutine, como Dispatchers.Main para el hilo principal y Dispatchers.IO para operaciones de entrada/salida.

Funciones de suspensión: Pausan y reanudan la ejecución de coroutines sin bloquear hilos.

Builders de coroutines: Funciones como launch y async que inician coroutines dentro del CoroutineScope.


```

import android.os.Bundle
import android.widget.TextView
import androidx.appcompat.app.AppCompatActivity
import androidx.lifecycle.LifecycleScope
import kotlinx.coroutines.*

class MainActivity : AppCompatActivity() {
    private lateinit var textView: TextView

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_main)
        textView = findViewById(R.id.textview)

        lifecycleScope.launch {
            val result = fetchUserData()
            textView.text = result
        }

        suspend fun fetchUserData(): String {
            delay(2000) // Simula una operación de red que tarda 2 segundos
            return "Usuario encontrado"
        }
    }
}

```

Para este ejemplo, utilice lifecycleScope para lanzar una coroutine que realiza una solicitud de red simulada (fetchUserData). La coroutine se ejecuta en el hilo principal, pero gracias a la suspensión, no bloquea la interfaz de usuario.

¿Como puedes implementar un RecyclerView en Android Studio utilizando kottin y cuál es la función del adapter en este contexto?

Un RecyclerView es un componente de la interfaz de usuario que permite mostrar grandes cantidades de datos de manera eficiente. Es una versión mejorada del ListView y proporciona más flexibilidad y rendimiento.

El Adapter es una clase que actúa como un puente entre el RecyclerView y la fuente de datos que quieres mostrar. Proporciona las vistas necesarias para cada elemento de la lista y gestiona la vinculación de datos con estas vistas.

Añadir las dependencias necesarias:

Primero, asegúrate de tener las dependencias de RecyclerView en tu archivo build.gradle:

Gradle

```

dependencies {
    implementation 'androidx.recyclerview:recyclerview:1.2.1'
}

```

Creamos un archivo XML (res/layout/item_layout.xml) que define la estructura visual de cada elemento de la lista.

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="vertical"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:padding="16dp">

    <TextView
        android:id="@+id/textView"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="Item"/>

</LinearLayout>
```

Creamos una clase Adapter (MyAdapter.kt) que actúa como un puente entre el RecyclerView y los datos. El Adapter infla el layout de los ítems y vincula los datos a las vistas.

```
import android.view.LayoutInflater
import android.view.View
import android.view.ViewGroup
import android.widget.TextView
import androidx.recyclerview.widget.RecyclerView

class MyAdapter(private val dataList: List<String>) : RecyclerView.Adapter<ViewHolder>() {

    class ViewHolder(itemView: View) : RecyclerView.ViewHolder(itemView) {
        val textView: TextView = itemView.findViewById(R.id.textView)
    }

    override fun onCreateViewHolder(parent: ViewGroup, viewType: Int): ViewHolder {
        val view = LayoutInflater.from(parent.context).inflate(R.layout.item, parent, false)
        return ViewHolder(view)
    }

    override fun onBindViewHolder(holder: ViewHolder, position: Int) {
        holder.textView.text = dataList[position]
    }

    override fun getItemCount(): Int {
        return dataList.size
    }
}
```

Finalmente, configuramos el RecyclerView en nuestra actividad principal (MainActivity.kt). Le asignamos un LayoutManager y el Adapter que creamos previamente.

```
import android.os.Bundle
import androidx.appcompat.app.AppCompatActivity
import androidx.recyclerview.widget.LinearLayoutManager
import androidx.recyclerview.widget.RecyclerView

class MainActivity : AppCompatActivity() {

    private lateinit var recyclerView: RecyclerView
    private lateinit var myAdapter: MyAdapter

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

        recyclerView = findViewById(R.id.recyclerView)
        recyclerView.layoutManager = LinearLayoutManager(this)

        val dataList = listOf("Item 1", "Item 2", "Item 3")
        myAdapter = MyAdapter(dataList)
        recyclerView.adapter = myAdapter
    }
}
```

La aplicación Android se vería como una lista vertical que contiene elementos de texto como "Item 1", "Item 2" y "Item 3". Cada uno de estos elementos se mostraría en un diseño sencillo, definido por el archivo de layout item_layout.xml.

```
Item 1
Item 2
Item 3
```

```
val dataList = listOf("Item 1", "Item 2", "Item 3", "Item 4", "Item 5")
```

OBSERVACIONES _____

 SENATI	INGENIERÍA DE SOFTWARE CON INTELIGENCIA ARTIFICIAL	ESCALA:
	<i>HT-01: Utiliza el entorno Android Studio y</i>	CÓDIGO: HT-01
	<i>Java</i>	FECHA: 15/02/2025



MÉTODO DE PROYECTOS DE ENSEÑANZA – APRENDIZAJE

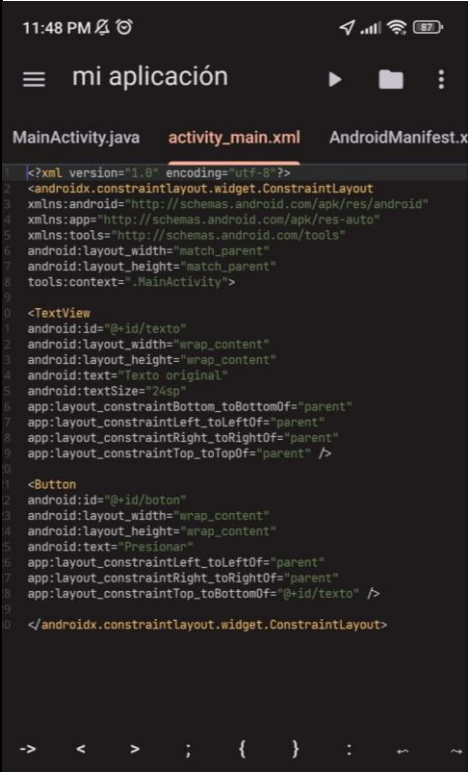
4. HOJA DE PLANIFICACIÓN DIBUJO / ESQUEMA

PROCESO DE EJECUCIÓN

DIBUJOS / ESQUEMAS	OPERACIONES /PASOS – SUBPASOS / SEGURIDAD / MEDIO AMBIENTE / NORMAS - ESTANDARES
<p>objetivo principal de este proyecto fue crear una aplicación Android funcional que permitiera al usuario interactuar con la interfaz de usuario y realizar acciones básicas, como cambiar el texto de un TextView al hacer clic en un Button. Además, se buscó mejorar la apariencia visual de la aplicación y agregar funcionalidades adicionales.</p>	
<p>Mejoras Implementadas Diseño de la interfaz de usuario:</p> <p>Se utilizó un diseño LinearLayout para organizar los elementos de la interfaz de usuario de forma vertical y centrada.</p> <p>Se agregó un TextView para mostrar un texto inicial y un Button para activar el cambio de texto.</p> <p>Se definieron atributos como android:layout_width, android:layout_height, android:text y android:textSize para configurar el tamaño, el texto y la apariencia de los elementos.</p>	
<p>Código Final A continuación, se muestra el código final de los archivos activity_main.xml y MainActivity.java con las mejoras implementadas:</p>	

A continuación, se muestra el código final de los archivos activity_main.xml y MainActivity.java con las mejoras implementadas:

13

	
<p>Se logró crear una aplicación Android funcional que permite al usuario interactuar con la interfaz de usuario y realizar acciones básicas. Se mejoró la apariencia visual de la aplicación utilizando un diseño LinearLayout y se agregó la funcionalidad de cambiar el texto de un TextView al hacer clic en un Button.</p>	

RESULTADO FINAL:

OBSERVACIONES _____		
	INGENIERÍA DE SOFTWARE CON INTELIGENCIA ARTIFICIAL	ESCALA:
	<i>HT-01: Utiliza el entorno Android Studio y</i>	CÓDIGO: HT-01
	<i>Java</i>	FECHA: 15/02/2025



MÉTODO DE PROYECTOS DE ENSEÑANZA – APRENDIZAJE

5. HOJA DE PREVENCIÓN DE RECURSOS

Para la ejecución del proyecto se requiere de recursos, listen lo que se necesite:

6.1. MATERIALES <hr/> <hr/> <hr/> <hr/> <hr/> <hr/> <hr/> <hr/>	6.2. INSUMOS <hr/> <hr/> <hr/> <hr/> <hr/> <hr/> <hr/> <hr/>
6.3. HERRAMIENTAS <hr/> <hr/> <hr/> <hr/> <hr/> <hr/> <hr/> <hr/>	6.4. INSTRUMENTOS <hr/> <hr/> <hr/> <hr/> <hr/> <hr/> <hr/> <hr/>
6.5. MAQUINAS <hr/> <hr/> <hr/> <hr/> <hr/> <hr/> <hr/> <hr/>	6.6. EQUIPOS <hr/> <hr/> <hr/> <hr/> <hr/> <hr/> <hr/> <hr/>
6.7. RECURSOS <hr/> <hr/> <hr/> <hr/> <hr/> <hr/> <hr/> <hr/>	6.8. OTROS REQUERIMIENTOS <hr/> <hr/> <hr/> <hr/> <hr/> <hr/> <hr/> <hr/>

3. PLAN DE TRABAJO

ETAPA (TIEMPO)	ACTIVIDADES DE LOS PARTICIPANTES	ACTIVIDADES DEL FACILITADOR	EVIDENCIA DE DESEMPEÑO
INFORMAR (30 min)	<p>Se organizan en grupos de estudio y trabajo.</p> <p>Analizan los objetivos específicos.</p> <p>Recopilan la información para resolver lo referente al planteamiento del problema.</p> <p>Responden el cuestionario guía en el plan de trabajo del participante.</p> <p>Socializan las respuestas del cuestionario guía.</p>	<p>Expone breve y didácticamente sobre el tema a desarrollar.</p> <p>Informa sobre el método de proyectos.</p> <p>Plantea el problema– Objetivo del proyecto.</p> <p>Entrega el plan de trabajo del participante.</p> <p>Entrega la rúbrica de evaluación.</p> <p>Indica tiempos de las fases del proyecto y organiza los grupos.</p> <p>Evalúa la presentación del cuestionario guía desarrollado de los participantes.</p>	<p>Cuestionario guía desarrollado.</p> <p>Evaluación de la presentación del cuestionario guía desarrollado.</p>
PLANIFICAR (20 min)	<p>Elaboran los pasos a seguir en el proceso de ejecución de acuerdo con el problema planteado.</p> <p>Redactan las hojas de planificación y prevención de recursos en el plan de trabajo del participante.</p> <p>Listan materiales y equipos en el plan de trabajo del participante.</p>	<p>Observa el trabajo de grupo durante la planificación y retroalimenta.</p> <p>Modera a los grupos.</p> <p>Incentiva la interacción de los participantes que no se integran.</p>	<p>Hoja de planificación (proceso de ejecución).</p> <p>Hoja de prevención de recursos.</p>
DECIDIR (10 min)	<p>Deciden la secuencia de pasos a seguir.</p> <p>Fundamentan al facilitador la secuencia de los pasos a seguir.</p> <p>Solicitan autorización para pasar a la siguiente fase.</p>	<p>Hace reflexionar a cada grupo sobre las características finales del resultado.</p> <p>Propone eventuales cambios.</p> <p>Autoriza pasar a la ejecución y/o revisar el planteamiento.</p>	<p>Hoja de planificación (proceso de ejecución) validada por el instructor.</p> <p>Hoja de prevención de recursos validada por el instructor.</p>
EJECUTAR (310 min)	<p>Organizan la ejecución de las actividades a realizar por cada miembro del equipo.</p> <p>Ejecutan las actividades según los pasos descritos en el plan de trabajo del participante, recomendaciones, puntos clave, normas de seguridad, etc.</p> <p>Anotan las observaciones recibidas por parte del facilitador.</p>	<p>Verifica que los materiales, medios didácticos, especificaciones técnicas y otras ayudas estén al alcance de los participantes.</p> <p>Observa el trabajo e interviene en caso de riesgos.</p> <p>Evalúa las habilidades y destrezas aplicando instrumentos de evaluación.</p>	<p>Hoja de planificación (proceso de ejecución)</p> <p>Material informativo elaborado.</p>
CONTROLAR (20 min)	<p>Auto controlan el cumplimiento de las especificaciones técnicas en el material informativo elaborado según la rúbrica de evaluación.</p> <p>Verifican su desempeño en el proceso realizado.</p> <p>Cotejan que todo lo solicitado en el plan del trabajo del participante esté desarrollado.</p>	<p>Controla el cumplimiento de las especificaciones técnicas en el material informativo elaborado contrastándolo con la rúbrica de evaluación.</p> <p>Promueve la discusión de los resultados obtenidos.</p> <p>Motiva a los equipos para la presentación de sus resultados al plenario.</p>	<p>Plan del trabajo del participante desarrollado.</p> <p>Material informativo elaborado contrastado con la rúbrica de evaluación.</p>
VALORAR (30 min)	<p>Reflexionan acerca de la experiencia vivida en el modelo de acción completa.</p> <p>Valoran el proceso realizado a través de las seis fases (Auto evaluación).</p> <p>Socializan los proyectos realizados por los grupos de trabajo.</p>	<p>Modera la socialización de los proyectos terminados.</p> <p>Retroalimenta según los resultados presentados.</p> <p>Evalúa los logros obtenidos.</p>	<p>Plan del trabajo del participante desarrollado.</p> <p>Material informativo elaborado.</p>
Tiempo total de desarrollo del proyecto			420 min.