



# TALLER 1

## Evaluación Unidad N°1

UNIDAD DE APRENDIZAJE I  
Fundamentos de Kotlin y Android

PROGRAMACIÓN II

## Instrucciones generales

Una vez realizada la lectura comprensiva de la unidad I del material de estudio, es fundamental la realización de este taller, el que tiene por objetivo medir la correcta internalización y aplicación de los conceptos abordados en las mencionadas unidades.

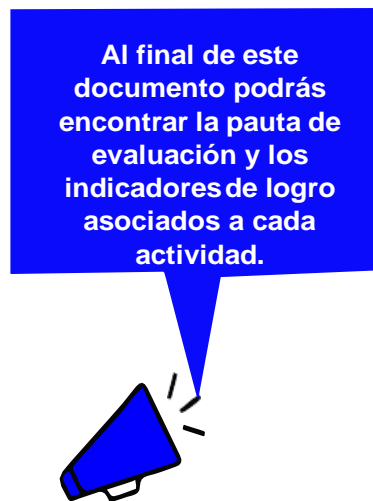
El desarrollo de este trabajo es individual, no se permitirán entregas de talleres en duplas o en grupos.

## Contenidos asociados al taller:

1. Lenguaje de programación Kotlin: variables, tipos de datos, funciones, programación orientada a objetos
2. Vistas XML: EditText, TextView, Switch e ImageView
3. Layouts: ConstraintLayout

## Instrucciones específicas

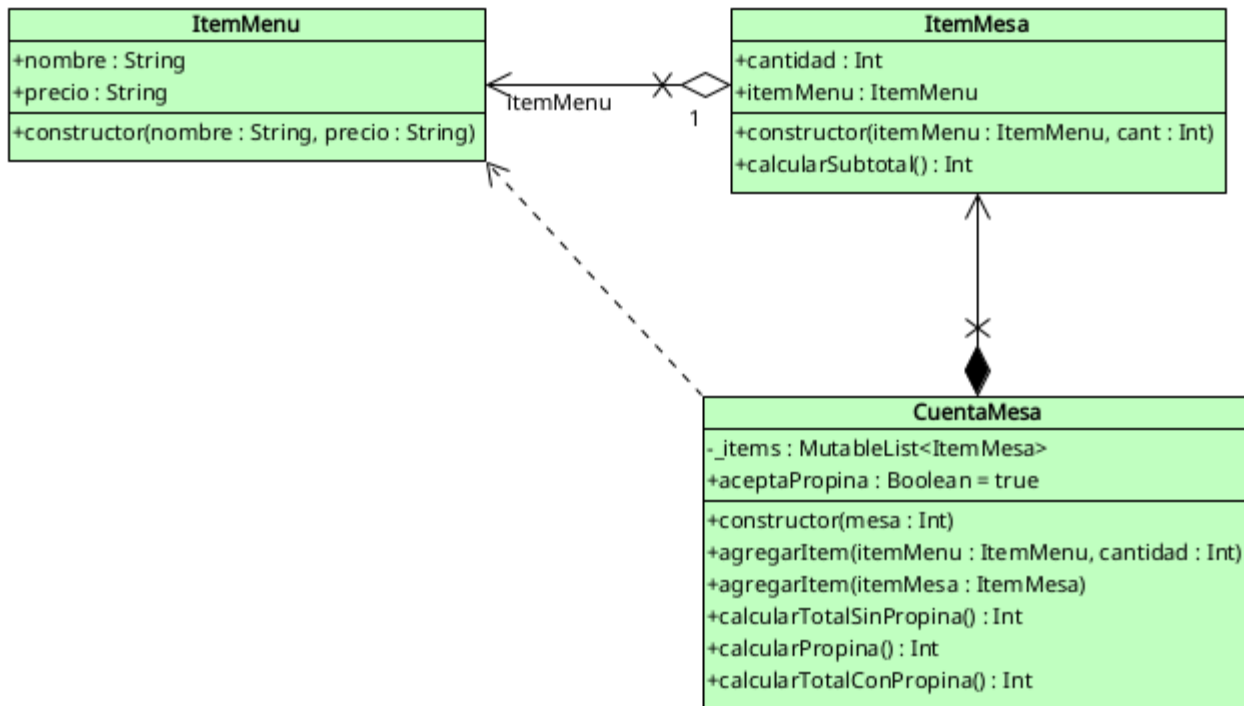
El taller propuesto requiere articular contenidos abordados en la Unidad I , a través de las siguientes acciones:



# I. Actividad 1



Figura 1 Sugerencia solución taller



*Figura 2 Diagrama de clases sugerido para la solución*

**Siga las siguientes instrucciones para la construcción de la solución:**

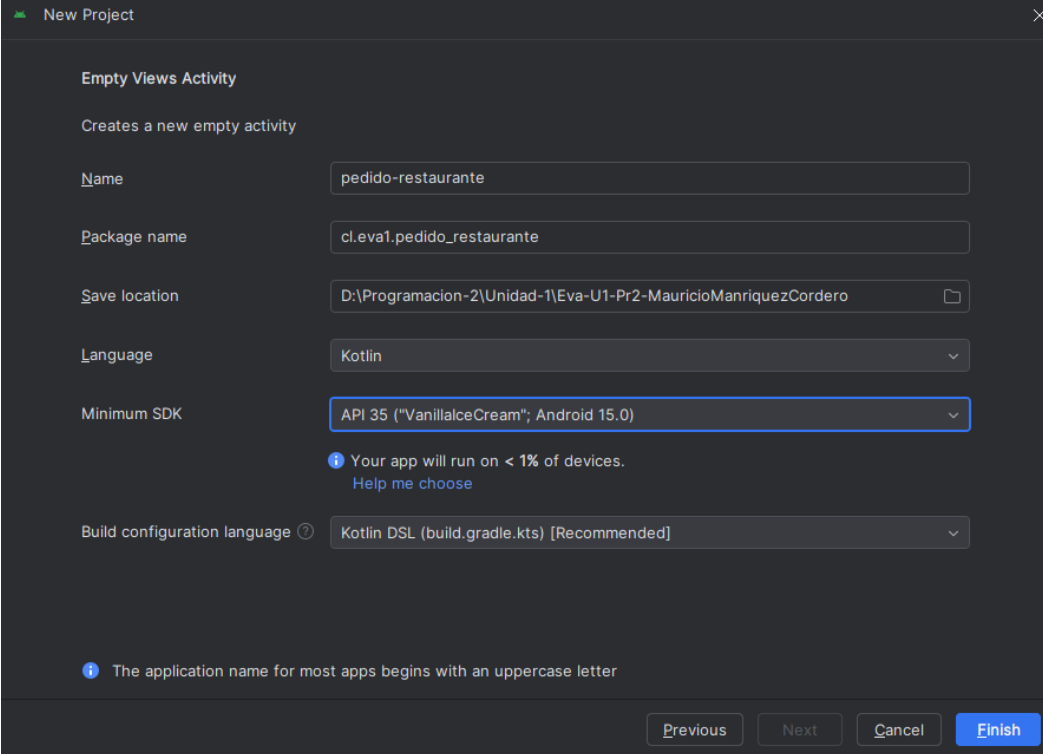
1. Emplee `ConstraintLayout` para organizar las vistas en la pantalla.
2. Utilice a lo menos las siguientes vistas: `TextView`, `EditText`, `Switch` e `ImageView` para diseñar una pantalla que se asemeje a la sugerida. La interfaz debe incluir, como mínimo, la presentación de los dos platillos ofrecidos por el restaurante, la posibilidad de ajustar la cantidad deseada para cada comida, mostrar el subtotal para cada especialidad (cantidad por el valor unitario del plato), y presentar en la parte inferior los totales sin incluir la propina, el monto de la propina y el total final con la propina incluida.
3. Implemente las clases que resuelven la problemática en un paquete independiente al de la `Activity`. Se sugiere estructurar las clases siguiendo un patrón similar al mostrado en la Figura 2 para mantener una organización coherente y facilitar la comprensión del código.
4. Actualice los montos de la pantalla inmediatamente se cambien las cantidades o si se desea incluir propina. Se recomienda investigar y utilizar eventos como: `TextChanged` y `CheckedChange`.
5. Formatee los valores monetarios como pesos chilenos usando un objeto del tipo `NumberFormat`.

**Se solicitan los siguientes entregables para poder evaluar el taller:**

1. Informe redactado utilizando sus propias palabras para explicar el trabajo realizado. Se recomienda seguir una estructura similar a la presentada más abajo en el documento. La claridad en la exposición de su trabajo contribuirá a una mejor evaluación y retroalimentación.
  - a. Debe estar en formato Word o PDF
  - b. Se requiere cargar el trabajo en la plataforma sin comprimir, de manera que pueda someterse a las herramientas de detección de plagio. De lo contrario, se aplicarán penalizaciones en forma de puntos restados. Este procedimiento es crucial para asegurar la integridad académica y garantizar una evaluación justa y precisa del trabajo.
2. Adjuntar el proyecto directamente o proporcionar un enlace accesible en GitHub, Google Drive u otra plataforma semejante.
  - a. En caso de adjuntar el código fuente comprimido, debe realizar una limpieza previa del proyecto utilizando la función "Clean Project" de Android Studio, la cual se encuentra en el menú "Build". Esta práctica garantiza la eliminación de archivos innecesarios.
  - b. En el caso de utilizar GitHub u otra plataforma similar, es su responsabilidad asegurarse de que el enlace proporcionado sea correcto y que el repositorio sea visible públicamente.
  - c. En el caso de utilizar alguna plataforma de almacenamiento, es su responsabilidad compartir un enlace que permita la visibilidad pública de los archivos. Asegúrese de configurar adecuadamente los permisos para garantizar el acceso público.

## Estructura sugerida para el informe

### 1. Construcción de proyecto



New Project

Empty Views Activity

Creates a new empty activity

Name: pedido-restaurante

Package name: cl.eva1.pedido\_restaurante

Save location: D:\Programacion-2\Unidad-1\Eva-U1-Pr2-MauricioManriquezCordero

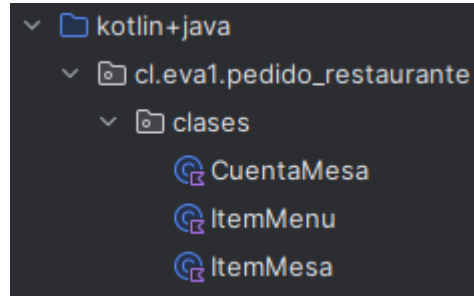
Language: Kotlin

Minimum SDK: API 35 ("VanillalceCream"; Android 15.0)

Build configuration language: Kotlin DSL (build.gradle.kts) [Recommended]

Previous Next Cancel Finish

## 2. Construcción de las clases modelo que solucionan el problema del Restaurant



```
CuentaMesa.kt
1 package cl.eva1.pedido_restaurante.clases
2
3 class CuentaMesa(val mesa: Int) {
4     private val _items = mutableListOf<ItemMesa>()
5     var aceptaPropina: Boolean = true
6
7     fun agregarItem(itemMesa: ItemMesa) {
8         _items.add(itemMesa)
9     }
10
11     fun calcularTotalSinPropina(): Int {
12         return _items.sumOf { it.calcularSubtotal() }
13     }
14
15     fun calcularPropina(): Int {
16         val totalSinPropina = calcularTotalSinPropina()
17         return if (aceptaPropina) (totalSinPropina * 0.10).toInt() else 0
18     }
19
20     fun calcularTotalConPropina(): Int {
21         return calcularTotalSinPropina() + calcularPropina()
22     }
23 }
```

```
ItemMenu.kt
1 package cl.eva1.pedido_restaurante.clases
2
3 data class ItemMenu (val nombre: String, val precio: Int)
```

```
ItemMesa.kt
1 package cl.eva1.pedido_restaurante.clases
2
3 class ItemMesa(val itemMenu: ItemMenu, var cantidad: Int) {
4     fun calcularSubtotal(): Int {
5         return cantidad * itemMenu.precio
6     }
7 }
```

- Se genera la construcción de las tres clases para dar desarrollo al proyecto de pedidos platillos restaurante, siendo estos CuentaMesa, ItemMenu e ItemMesa. Con esto tendremos los primeros pasos para poder dar inicio a nuestra actividad solicitada.



### 3. Construcción de la pantalla con ConstraintLayout

```

1  <?xml version="1.0" encoding="utf-8"?>
2  <androidx.constraintlayout.widget.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/android"
3      xmlns:app="http://schemas.android.com/apk/res-auto"
4      xmlns:tools="http://schemas.android.com/tools"
5      android:layout_width="match_parent"
6      android:layout_height="match_parent"
7      tools:context=".MainActivity">
8
9
10     <ImageView
11         android:id="@+id/imageView"
12         android:layout_width="210dp"
13         android:layout_height="89dp"
14         android:layout_marginTop="56dp"
15         app:layout_constraintEnd_toEndOf="parent"
16         app:layout_constraintHorizontal_bias="0.497"
17         app:layout_constraintStart_toStartOf="parent"
18         app:layout_constraintTop_toTopOf="parent"
19         app:srcCompat="@drawable/restaurant"
20         tools:ignore="ContentDescription,MissingConstraints" />
21
22     <TextView
23         android:id="@+id/tvSubtotal"
24         android:layout_width="wrap_content"
25         android:layout_height="wrap_content"
26         android:layout_marginTop="52dp"
27         android:text="Comida $0"
28         app:layout_constraintEnd_toEndOf="parent"
29         app:layout_constraintStart_toStartOf="parent"
30         app:layout_constraintTop_toBottomOf="@+id/editTextNumberDecimal2"
31         tools:ignore="HardcodedText" />
32
33     <ImageView
34         android:id="@+id/imgCazuela"
35         android:layout_width="140dp"
36         android:layout_height="135dp"
37         android:layout_marginStart="28dp"
38         android:layout_marginTop="64dp"
39         android:contentDescription="Cazuela"
40         android:src="@drawable/cazuela_chilena"
41         app:layout_constraintHeight_percent="0.3"
42         app:layout_constraintStart_toStartOf="parent"

```

- Construcción de esquema utilizando ConstraintLayout, según lo establecido en el documento.

#### 4. Construcción de la Activity

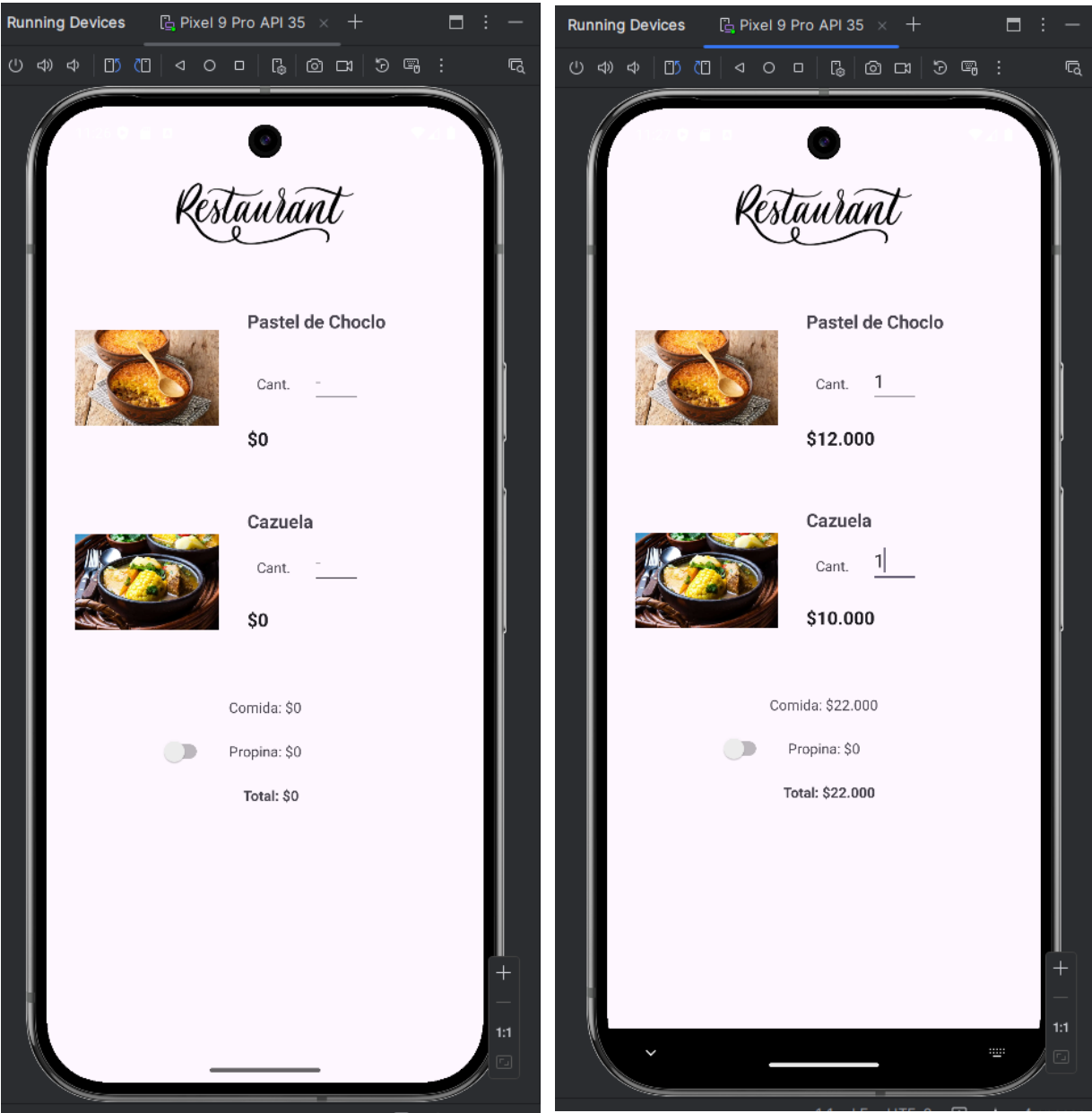
- a. Integración de la Activity con las clases modelo
- b. Gestión de eventos

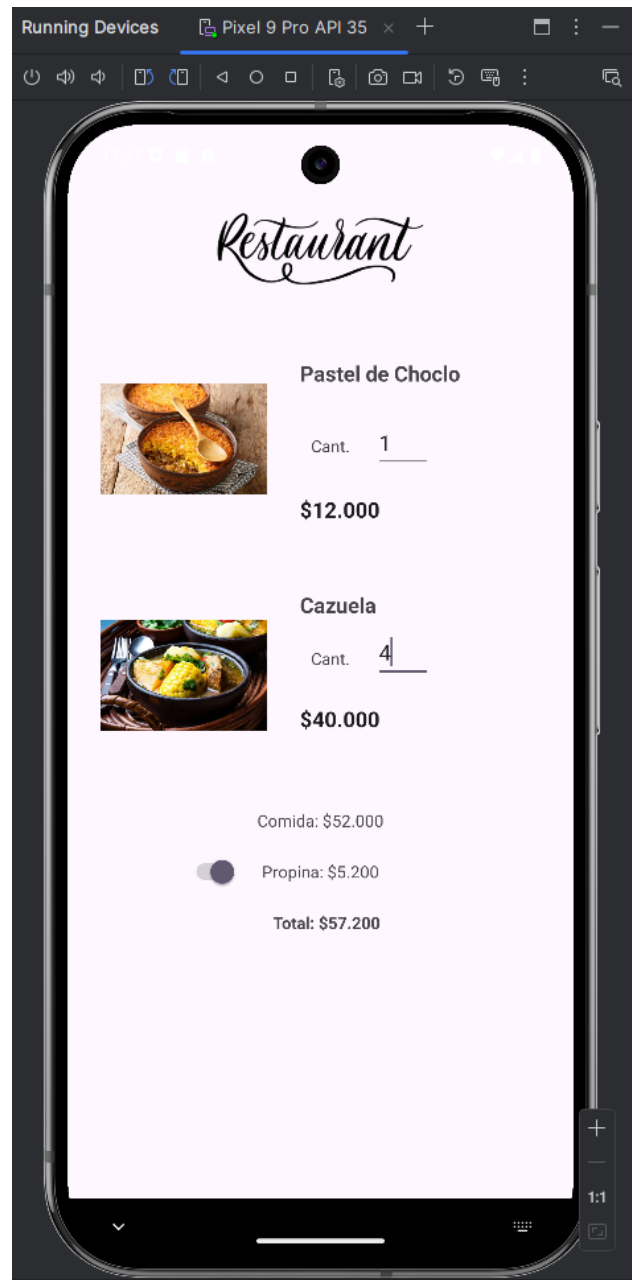
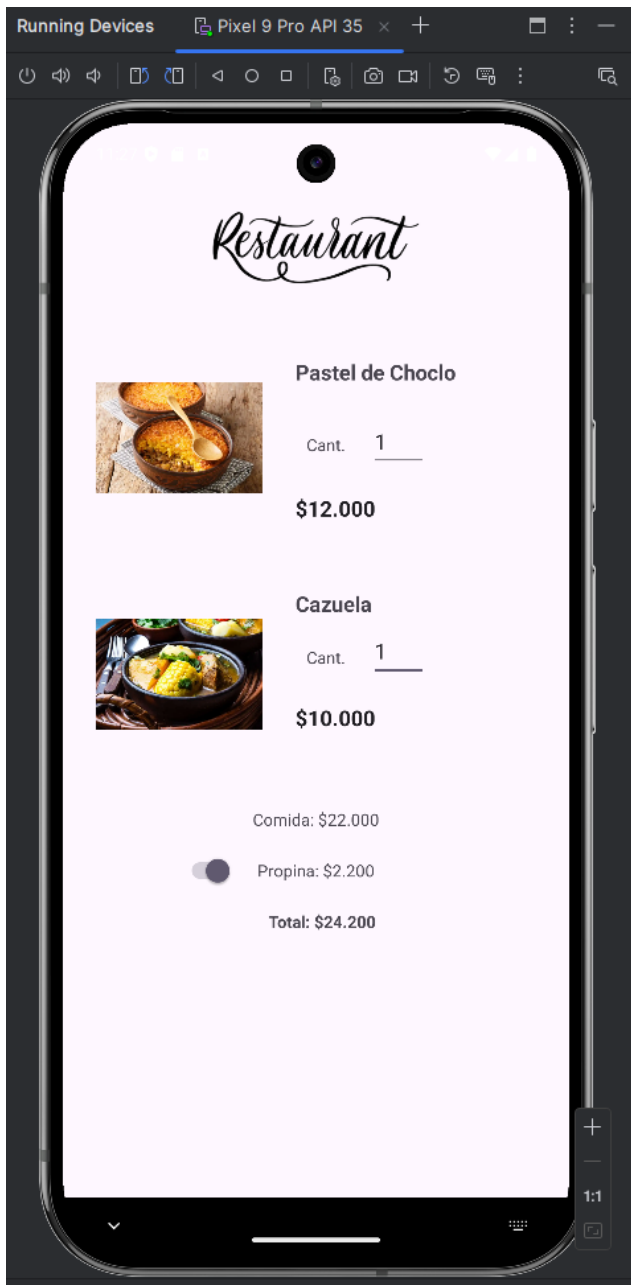
```

1 package cl.eval.pedido_restaurante
2
3 import android.os.Bundle
4 import android.text.Editable
5 import android.text.TextWatcher
6 import android.widget.Switch
7 import android.widget.TextView
8 import android.widget.EditText
9 import androidx.appcompat.app.AppCompatActivity
10 import java.text.NumberFormat
11 import java.util.Locale
12
13 class MainActivity : AppCompatActivity() {
14
15     // Referencias de los elementos del layout
16     private lateinit var etPastelDeChocloCantidad: EditText
17     private lateinit var etCazuelaCantidad: EditText
18     private lateinit var etPastelDeChocloValor: TextView
19     private lateinit var etCazuelaValor: TextView
20     private lateinit var switchPropina: Switch
21
22     private lateinit var tvSubtotal: TextView
23     private lateinit var tvPropina: TextView
24     private lateinit var tvTotal: TextView
25
26     private var valorUnitarioPastelDeChoclo = 12000.0
27     private var valorUnitarioCazuela = 10000.0
28     private var cantidadPastelDeChoclo = 0
29     private var cantidadCazuela = 0
30     private var propina = 0.0
31
32     override fun onCreate(savedInstanceState: Bundle?) {
33         super.onCreate(savedInstanceState)
34         setContentView(R.layout.activity_main)
35
36         // Inicializar las vistas
37         etPastelDeChocloCantidad = findViewById(R.id.etPastelDeChocloCantidad)
38         etCazuelaCantidad = findViewById(R.id.etCazuelaCantidad)
39         etPastelDeChocloValor = findViewById(R.id.editTextNumberDecimal)
40         etCazuelaValor = findViewById(R.id.editTextNumberDecimal2)
41         switchPropina = findViewById(R.id.switchPropina)
    
```

- Desarrollo de estructura MainActivity.kt, para la ejecución de la app.

5. Evidencia de la solución funcionando (pantallazos)





- Imágenes referenciales de la ejecución de la app. Utilización de emulador en Android Studio.

## 6. Desafíos encontrados

Uno de los puntos encontrados, es la creación de los tipos de clases mencionados en el segundo punto del desarrollo del presente documento, no obstante, estos no fueron requeridos al momento de realizar el MainActivity.kt, ya que este tuvo una ejecución sin la necesidad de estas clases, teniendo ya incorporado en la estructura un llamado de ejecución para el inicio de la app. Por lo demás el funcionamiento se encuentra normalizado

## Recomendaciones generales

1. Se recomienda una lectura minuciosa de las instrucciones proporcionadas y una revisión exhaustiva de la pauta de evaluación. Asegúrese de abordar cada uno de los puntos indicados en ambas fuentes, ya que estas contienen detalles cruciales para el desarrollo y la evaluación exitosa de su proyecto.
2. Se sugiere que el informe se mantenga conciso, centrándose específicamente en los puntos relevantes para la evaluación. Dirija su atención a los aspectos clave del desarrollo que se están evaluando, evitando incluir información superflua. Para cualquier requisito adicional, es posible revisar el código fuente. Este enfoque permite una mejor retroalimentación.
3. Cuando escriba su código, es recomendable emplear nombres significativos tanto para variables, atributos, funciones, métodos como para clases. Opte por nombres descriptivos que reflejen claramente el propósito y contenido de la entidad que representan. Por ejemplo, en lugar de utilizar nombres genéricos como "a1", considere nombrar sus variables de manera explícita, por ejemplo, "montoAPagar". Esto no solo mejora la claridad y la legibilidad del código, sino que también facilita la comprensión tanto para usted como para otros colaboradores que puedan revisar o trabajar en el proyecto en el futuro. Además, incluir comentarios explicativos puede proporcionar una guía adicional sobre la función y la lógica subyacente.

## Información Importante

Esta actividad será evaluada por el profesor de la asignatura, por lo que se sugiere, para su realización, considerar los Criterios de Evaluación detallados al final de este documento. El puntaje total de este taller es de 100 puntos, y el puntaje mínimo de aprobación es de 60 puntos.

# Formalidades

- Tipo de letra: Arial 12
- Interlineado: 1,15
- Formato - Entrega del taller en archivo Word o PDF, que lleve como nombre: SuNombre\_SuApellido.
- Subir archivo a la plataforma

## Pauta de evaluación

Criterios de Evaluación	Indicadores de Logro	Puntaje
<i>Comprende los conceptos fundamentales de Kotlin para el desarrollo de aplicaciones</i>	Utiliza correctamente variables mutables e inmutables (val y var).	10
<i>Utiliza colecciones y arrays en Kotlin para el almacenamiento y manipulación de datos.</i>	Utiliza correctamente listas para almacenar los datos de los platillos consumidos por la mesa	10
<i>Diseña y desarrolla funciones en Kotlin para dividir y reutilizar funcionalidades</i>	Codifica funciones que calculan con precisión los montos de la mesa, tanto con como sin propina, evitando redundancias innecesarias en el código.	10
<i>Aplica los principios de la programación orientada a objetos (POO) en la escritura de código Kotlin para resolver los requerimientos del usuario.</i>	Estructura y distribuye las funcionalidades necesarias para la aplicación a través de clases. Se destaca por una organización coherente y eficiente, donde cada clase desempeña su función específica de manera clara y contribuye al conjunto general de la solución del problema.	10
<i>Desarrolla aplicaciones básicas en Android utilizando Vistas XML.</i>	Construye la pantalla utilizando al menos las siguientes vistas: TextView, EditText, Switch e ImageView	10
<i>Diseña interfaces de usuario utilizando LinearLayout, ConstraintLayout y TableLayout para organizar los componentes que interactúan con el usuario</i>	Organiza la interfaz utilizando ConstraintLayout. Crea un diseño ordenado, aplicando espacios y alineaciones entre elementos. Se evidencia un diseño limpio, donde cada vista en la pantalla tiene, al menos, una restricción horizontal y una vertical.	10

<i>Gestiona los eventos de usuario en aplicaciones Android para agregar funcionalidad.</i>	Gestiona eventos de manera eficiente, posibilitando la actualización automática de los valores en pantalla. Garantiza que los subtotales de los platillos y los totales, ya sea con o sin propina, se actualicen de forma dinámica cuando hay modificaciones en algún valor.	10
	Evidencia un código ordenado mediante una correcta indentación. Usa nomenclaturas apropiadas para nombrar clases, variables y funciones, contribuyendo así a la claridad y legibilidad del código. Además, escribe comentarios explicativos, y emplea nombres significativos que facilitan la interpretación de clases, variables y funciones.	10
	Adjunta el proyecto o proporciona un enlace en el informe que dirige al código fuente. El proyecto está limpio, libre de archivos innecesarios.	10
	Cumple con éxito el plazo de entrega, presentando un informe en formato Word o PDF sin comprimir. El informe está completo, redactado con buena ortografía y explica con las propias palabras del estudiante el proceso seguido. Se enfoca en los puntos esenciales de la unidad a evaluar.	10
Puntaje Total		100



**4** INSTITUCION  
**ACREDITADA**  
NIVEL AVANZADO  
AÑOS Hasta octubre 2025



GESTIÓN INSTITUCIONAL Y DOCENTE DE PREGRADO