

# Desarrollo de Aplicaciones Móviles para Android

Dr. Marco Aurelio Nuño-Maganda

Universidad Politecnica de Victoria

[https://github.com/mnunom-upv/Curso\\_Desarrollo\\_Aplicaciones\\_Moviles\\_2023](https://github.com/mnunom-upv/Curso_Desarrollo_Aplicaciones_Moviles_2023)

Noviembre 2025



- 1 Tic-Tac-Toe
- 2 Pong
- 3 Etapa 1: Definir Apariencia de la Aplicacion (Interfaz de usuario)

- 1 Tic-Tac-Toe
- 2 Pong
- 3 Etapa 1: Definir Apariencia de la Aplicacion (Interfaz de usuario)

# Tic-Tac-Toe: La Estrategia de la Sencillez

## Mecánica y Tablero

- Es un juego para **dos jugadores** que se desarrolla en un tablero de  $3 \times 3$  casillas.
- Un jugador marca con "equis" y el otro con "círculo".

## Objetivo del Juego

- El objetivo es ser el primero en colocar tres de sus símbolos de forma consecutiva.
- Las combinaciones ganadoras pueden ser **horizontales, verticales o diagonales**.

## Desarrollo de la Partida

- Los jugadores se turnan para colocar un símbolo en una casilla vacía.
- Si el tablero se llena y nadie logra la línea de tres, la partida se declara **empate** (o \*gato\*).

## Análisis Estratégico

Se usa a menudo como una herramienta pedagógica para introducir conceptos de **teoría de juegos** y búsqueda de árboles de decisión en programación.

- 1 Tic-Tac-Toe
- 2 Pong
- 3 Etapa 1: Definir Apariencia de la Aplicacion (Interfaz de usuario)

## Creador y Compañía

- Desarrollado para la empresa **Atari** en 1972, convirtiéndose en el **primer videojuego de éxito comercial masivo**.
- El concepto se inspira en el deporte del tenis de mesa (ping pong).

## Revolución Doméstica (1975)

- Atari lanzó una versión casera, *Home Pong*, vendida exclusivamente a través de las tiendas Sears durante la temporada navideña de 1975.
- Esto introdujo los videojuegos en el **hogar** y provocó el nacimiento del mercado de consolas domésticas.

## Legado Técnico y Cultural

- A pesar de su simplicidad gráfica (dos paletas y una pelota), sentó las bases para mecánicas de juego fundamentales y estableció a los videojuegos como una **forma de entretenimiento viable y lucrativa**.

## 1 Configuración de Pantalla Completa

- Se asegura una experiencia inmersiva eliminando la barra de título (FEATURE\_NO\_TITLE) y la barra de acción (supportActionBar.hide()), y estableciendo el modo de **pantalla completa** (FLAG\_FULLSCREEN).

## 2 Motor de Juego Principal (GameView)

- La lógica del juego (dibujo, física, bucle de actualización) reside en una vista personalizada (GameView), la cual se añade dinámicamente al contenedor (FrameLayout) de la actividad.

## 3 Comunicación y Actualización de Puntaje

- Se establece un **Listener** (ScoreUpdateListener) en la GameView para recibir los nuevos puntajes.
- La actualización de los TextView de los jugadores se realiza de forma segura en el **Hilo Principal** de Android (runOnUiThread) para evitar fallos en la interfaz de usuario.

## 4 Manejo del Ciclo de Vida del Juego

- La actividad maneja el estado del juego mediante los métodos onPause() y onResume(), llamando a gameView.pauseGame() y gameView.resumeGame() respectivamente. Esto es esencial para **conservar recursos** al cambiar de aplicación.

La clase GameView es la superficie de dibujo principal y el motor de lógica de la aplicación.

## 1 Gestión de Hilos (Threading)

- Contiene un **GameThread** que ejecuta el bucle de juego para **actualizar la lógica** (`update()`) y (`draw()`) en un ciclo constante.
- Esto asegura que las animaciones se ejecuten fluidamente sin bloquear el Hilo Principal (UI).

## 2 Implementación de la Lógica del Juego

- Maneja la posición y el movimiento de los objetos (pelotas, paletas, etc.) y la detección de colisiones.
- Es responsable de incrementar los contadores de puntaje tras eventos clave, como anotar un punto.

## 3 Control de Estado y Comunicación

- Utiliza la interfaz `ScoreUpdateListener` para notificar a la actividad sobre los cambios de puntaje, separando la lógica del juego de la actualización de la UI.



1 Tic-Tac-Toe

2 Pong

3 Etapa 1: Definir Apariencia de la Aplicacion (Interfaz de usuario)

# Fase 1: Dividir la pantalla en contenedores verticales

## Archivo activity\_main.xml

```
1 <?xml version="1.0" encoding="utf-8"?>
2 <LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
3     android:layout_width="match_parent" android:layout_height="match_parent"
4     android:orientation="vertical" android:padding="16dp">
5     <LinearLayout android:layout_width="match_parent"
6         android:layout_weight="1" android:orientation="horizontal"
7         android:layout_height="0dp" >
8         <TextView
9             android:id="@+id/scorePlayer1" android:gravity="center"
10             android:layout_weight="1" android:layout_width="0dp"
11             android:layout_height="match_parent" android:textSize="32sp"
12             android:text="0" android:padding="8dp" />
13         <TextView
14             android:id="@+id/scorePlayer2" android:gravity="center"
15             android:layout_weight="1" android:layout_width="0dp"
16             android:layout_height="match_parent" android:textSize="32sp"
17             android:text="0" android:padding="8dp" />
18     </LinearLayout>
19     <FrameLayout
20         android:id="@+id/gameContainer" android:layout_width="match_parent"
21         android:layout_weight="9" android:layout_height="0dp"
22         android:layout_below="@id/scorePlayer1"/>
23 </LinearLayout>
```

### Archivo MainActivity.kt

```
1      // Declaración de variables usando 'lateinit' para inicialización no nula en onCreate
2      private lateinit var gameView: GameView
3      private lateinit var scorePlayer1: TextView
4      private lateinit var scorePlayer2: TextView
5      override fun onCreate(savedInstanceState: Bundle?) {
6          super.onCreate(savedInstanceState)
7          requestWindowFeature(Window.FEATURE_NO_TITLE)
8          supportActionBar?.hide()
9          window.setFlags(WindowManager.LayoutParams.FLAG_FULLSCREEN,
10              WindowManager.LayoutParams.FLAG_FULLSCREEN)
11          setContentView(R.layout.activity_main)
12          scorePlayer1 = findViewById(R.id.scorePlayer1)
13          scorePlayer2 = findViewById(R.id.scorePlayer2)
14          val gameContainer: FrameLayout = findViewById(R.id.gameContainer)
15          gameView = GameView(this)
16          gameContainer.addView(gameView)
17          gameView.setScoreUpdateListener(object : GameView.ScoreUpdateListener {
18              override fun onScoreUpdate(player1Score: Int, player2Score: Int) {
19                  runOnUiThread {
20                      scorePlayer1.text = player1Score.toString() // Convierte Int a String
21                      scorePlayer2.text = player2Score.toString() // Convierte Int a String
22                  }
23              }
24          })
25      }
26  }
27  ..
```