



Paquetización en Kivy

Estudiante: Leandro Diaz Manriquez
Ramo: Desarrollo móvil
Docente: Cristian Iglesias

Fecha: 24/11/2025

Introducción

A continuación se presenta el paso a paso del proceso realizado durante el desarrollo de este proyecto. El objetivo es documentar de manera clara y ordenada las etapas que permitieron crear una aplicación en Python utilizando el framework Kivy, configurarla en un entorno Linux mediante WSL y finalmente paquetizarla en un archivo APK usando Buildozer. Este informe resume lo que se hizo, las herramientas utilizadas y el apoyo recibido de la IA para resolver dudas técnicas.

1. Instalación de herramientas necesarias

Se instalaron todas las herramientas requeridas para el desarrollo y la paquetización del proyecto, incluyendo Python, Kivy y Buildozer, junto con sus dependencias correspondientes.

2. Preparación del entorno de trabajo

El desarrollo se realizó utilizando **Ubuntu mediante WSL (Windows Subsystem for Linux)**. Se configuró el entorno inicial, se actualizaron paquetes y se habilitaron los componentes necesarios para compilar aplicaciones Android.

3. Inicio del proyecto Kivy

Se creó un nuevo repositorio y se inició el proyecto base en Python utilizando el framework **Kivy**. Esto incluyó la configuración inicial del entorno virtual y la estructura del proyecto.

4. Instalación de dependencias

Se descargaron e instalaron todas las dependencias requeridas para trabajar con Kivy.

5. Creación del archivo principal

Se desarrolló el archivo `main.py`, que inicialmente contenía un ejemplo básico de “**Hello world**” para verificar el correcto funcionamiento de Kivy y la estructura base de la aplicación.

6. Paquetización con Buildozer

Se ejecutó el comando:

```
buildozer -v android debug
```

Este proceso generó el APK de la aplicación, compilando todos los recursos, dependencias y el intérprete de Python dentro del paquete.

7. Pruebas y despliegue

Una vez generado el APK, se probó en un dispositivo Android para validar su funcionamiento. Finalmente, el proyecto completo se subió al repositorio oficial en **GitHub** para su respaldo y control de versiones.

2. Uso de Inteligencia Artificial en el Proyecto

Durante el desarrollo se utilizó IA (DeepSeek) para apoyar el proceso en las siguientes áreas:

Resolución de problemas técnicos

La IA ayudó a diagnosticar y resolver errores relacionados con dependencias faltantes, configuración del [buildozer.spec](#) y compatibilidad de herramientas.

Explicación de conceptos

Se solicitaron aclaraciones sobre conceptos de compilación, entorno Linux, Kivy, Buildozer y manejo del SDK/NDK de Android.

Asistencia en el uso de Ubuntu en WSL

La IA proporcionó guías, ejemplos de comandos y pasos para configurar correctamente el subsistema Linux.

3. Prompt utilizado (ejemplo real)

“¿Me sale este error de ‘faltan dependencias’?”

“¿Cómo instalo Buildozer correctamente en Ubuntu WSL?”

“No me detecta el SDK o el NDK, ¿qué puedo hacer?”

“¿Cómo soluciono el error de dependencia Cython al compilar?”