

Guía completa: Convertir una app web a APK usando Android Studio y Capacitor

Esta guía explica cómo transformar una aplicación web (hecha con Ionic, Angular, React o Vue) en un archivo APK instalable para Android, utilizando Capacitor y Android Studio.

Requisitos previos

- Tener Node.js instalado
- Tener Android Studio instalado y configurado (con SDKs)
- Tener instalado Ionic (si tu app lo usa):

```
npm install -g @ionic/cli
```

1. Construcción de la app web

Primero, asegurate de que tu app web esté lista para producción. Si estás usando Ionic:

```
ionic build
```

¿Qué hace este comando?

Genera los archivos estáticos finales (`index.html`, `main.js`, `styles.css`, etc.) dentro de la carpeta `/www`. Esta carpeta será la que Android use para mostrar tu app.

2. Agregar Capacitor al proyecto

Si aún no lo hiciste, inicializa Capacitor en tu proyecto:

```
npm install @capacitor/core @capacitor/cli  
npx cap init
```

¿Qué hace esto?

- Instala Capacitor, que sirve como puente entre tu app web y funciones nativas de Android/iOS.
 - El comando `init` configura Capacitor con:
 - Un nombre para la app (lo que el usuario verá en el dispositivo)
 - Un ID de paquete (por ejemplo: `com.miempresa.miapp`)
-

3. Agregar la plataforma Android

```
npx cap add android
```

¿Qué hace esto?

- Crea una carpeta `/android` con un proyecto de Android Studio completamente funcional.
- Aquí es donde se integrará tu app web con una "cáscara" nativa de Android.

4. Sincronizar cambios de la web con Android

Cada vez que cambies algo en tu app web, hacé:

```
ionic build      # Recompila los archivos web en /www
npx cap copy     # Copia los archivos de /www a Android
npx cap sync     # (Opcional) Sincroniza plugins nativos y configuraciones
```

¿Para qué sirve cada comando?

- `ionic build`: crea los archivos de producción.
- `npx cap copy`: lleva esos archivos a la carpeta `android/app/src/main/assets/public`.
- `npx cap sync`: también actualiza configuraciones de Capacitor y plugins nativos (si usás).

5. Abrir en Android Studio


```
npx cap open android
```

¿Qué hace esto?

Abre automáticamente Android Studio con el proyecto nativo ya preparado.

6. Probar la app en un dispositivo o emulador

Desde Android Studio:

1. Conectá tu teléfono (activar "Depuración USB") o usá un emulador.
2. Hacé clic en el botón **Run** .
3. Android Studio compilará y ejecutará tu app en el dispositivo.

7. Generar el APK final

En Android Studio:

1. Ir a **Build > Build Bundle(s) / APK(s) > Build APK(s)**
2. Espera a que termine el proceso.
3. Android Studio te mostrará un link con el archivo APK, por ejemplo:

```
/android/app/build/outputs/apk/debug/app-debug.apk
```

☒ Notas finales

- Si quieres firmar tu APK para subirlo a Play Store, usá **Build > Generate Signed Bundle/APK** en Android Studio.
- Siempre que cambies tu app web, no te olvides de correr:

```
ionic build && npx cap copy
```

⚙️ ¿Por qué se usa `npx cap` y no `ionic cap`?

Aunque podrías usar:

```
ionic cap copy
```

Eso depende del CLI de Ionic, y a veces **no está tan actualizado** o puede introducir comportamientos extra.

En cambio, usar:

```
npx cap copy
```

☒ Ventajas:

- Ejecuta directamente el comando del CLI de Capacitor.
- Asegura que estás usando la versión local del CLI (la instalada en `node_modules`), lo cual es más confiable y coherente con tu proyecto.