

React Native + Tailwind (NativeWind)

Diego Muñoz

20 de noviembre de 2025

- Usar una plantilla con **NativeWind + Tailwind** ya configurados.
- Trabajar el código en estilo **JavaScript** (sin preocuparnos de tipos).
- Ver conceptos básicos de **diseño móvil**: layout, scroll, safe area.

- En vez de configurar todo a mano, usaremos la plantilla oficial:

```
npx rn-new --nativewind --yarn
```

- Elegir un nombre de proyecto (ej: `hello-rn-nativewind`).
- El proyecto viene con:
 - Expo
 - NativeWind
 - Tailwind v3
 - Metro + Babel ya configurados

Aunque los archivos sean `.tsx`, podemos escribir código como si fuera JS.

Archivos importantes que ya vienen listos:

- `App.tsx` — punto de entrada.
- `tailwind.config.js` — config de Tailwind + NativeWind.
- `babel.config.js` — plugin de NativeWind.
- `metro.config.js` — integra `global.css`.
- `global.css` — importa `@tailwind base, components, utilities`.
No hay que tocar la configuración para esta clase, solo el código de la app.

- Biblioteca para usar **clases tipo Tailwind** en React Native.
- Traduce **bg-blue-500**, **text-xl**, **flex-1**, etc. a estilos nativos.
- Funciona sobre Expo con el setup generado por **rn-new**.

Primer test con NativeWind

En `App.tsx` (usamos `className` en lugar de `Styles`):

```
import "./global.css";
import { Text, View } from "react-native";
export default function App() {
  return (
    <View className="flex-1 items-center justify-center bg-slate-900">
      <View className="px-6 py-4 rounded-2xl bg-slate-800">
        <Text className="text-2xl font-semibold text-white">React</Text>
        <Text className="text-base text-slate-300 mt-2">Native</Text>
      </View>
    </View>
  );
}
```

Desde la carpeta del proyecto:

`yarn start`

- Se abre el panel de Expo Dev Tools.
- Escanear el código QR con **Expo Go**.
- Si algo se rompe tras editar, hacer **Reload** desde la app o reiniciar `yarn start`.

- Pantalla pequeña y orientación variable.
- No hay **hover**, solo toques/gestos.
- Texto debe ser legible en pantallas con DPI alto.
- Considerar **notch**, barra de estado y zonas no utilizables.
- Usuario muchas veces en movimiento.

- Layout basado en **Flexbox**:
 - flex-1, flex-row, items-center, justify-between, etc.

```
<View className="flex-1 flex-row items-center justify-between px-4">  
  <Text className="text-white">Left</Text>  
  <Text className="text-white">Right</Text>  
</View>
```

- NativeWind mapea estas utilidades a estilos nativos.

Scroll básico

Cuando el contenido no cabe en una pantalla:

```
import { ScrollView, Text, View } from "react-native";

export default function App() {
  return (
    <ScrollView className="flex-1 bg-slate-900">
      <View className="p-6">
        <Text className="text-2xl font-bold text-white mb-4">About</Text>
        <Text className="text-base text-slate-200 mb-2">Some text</Text>
      </View>
    </ScrollView>
  );
}
```

Safe area

- Zonas ocupadas por notch, barra de estado, barra de gestos.
- Evitar que el contenido quede “debajo” de esas áreas.

```
import { SafeAreaView, Text } from "react-native";

export default function App() {
  return (
    <SafeAreaView className="flex-1 bg-slate-900">
      <Text className="text-xl text-white m-4">
        Content inside safe area
      </Text>
    </SafeAreaView>
  );
}
```

Toque en vez de click

- Componentes interactivos típicos:
 - Pressable
 - TouchableOpacity

```
import { Text, Pressable, View } from "react-native";
export default function App() {
  return (
    <View className="flex-1 items-center justify-center bg-slate-900">
      <Pressable className="px-6 py-3 rounded-full bg-emerald-500">
        <Text className="text-white text-base font-semibold">Tap</Text>
      </Pressable>
    </View>
  );
}
```

- Usamos `npx rn-new --nativewind --yarn` para generar un proyecto Expo con:
 - NativeWind
 - Tailwind v3
 - Babel + Metro configurados
- Trabajamos con código estilo **JS** aunque los archivos sean **.tsx**.
- Vimos:
 - Pantalla pequeña y DPI alto.
 - Flexbox.
 - **ScrollView**.
 - **SafeAreaView**.
 - Componentes táctiles (**Pressable**).

- Armar una pantalla **“About Me”** móvil con NativeWind:
 - Foto, nombre, descripción breve.
 - Botón de acción.
- Combinar:
 - **ScrollView**
 - layout con Flexbox
 - botones táctiles (**Pressable**)