

React Native + Tailwind (NativeWind)

Diego Muñoz

20 de noviembre de 2025

- Configurar **Tailwind** en **React Native + Expo** usando **JavaScript**.
- Usar **NativeWind** para clases tipo Tailwind.
- Ver conceptos básicos de **diseño móvil**: layout, scroll, safe area.

- Proyecto de la clase anterior:

```
yarn create expo hello-rn-js --template blank  
cd hello-rn-js
```

- Configuración de Yarn en `.yarnrc.yml`:

```
nodeLinker: node-modules
```

- App base en `App.js` o `App.jsx` en JS.

- Biblioteca que permite usar **clases tipo Tailwind** en React Native.
- Traduce **bg-blue-500**, **text-xl**, **flex-1**, etc. a estilos nativos.
- Funciona con Expo sin ejectar el proyecto.

Desde la carpeta del proyecto:

```
yarn add nativewind tailwindcss
```

```
npx tailwindcss init
```

- Esto crea `tailwind.config.js` (vacío al inicio).

```
module.exports = {  
  content: [  
    "./App.{js,jsx}",  
    "./src/**/*.{js,jsx}",  
  ],  
  theme: { extend: {} },  
  plugins: []  
};
```

- `content`: dónde buscar clases.
- `src/`: opcional, pero sirve para mantener nuestra estructura.

```
module.exports = function (api) {  
  api.cache(true);  
  return {  
    presets: ["babel-preset-expo"],  
    plugins: ["nativewind/babel"]  
  };  
};
```

- Plugin necesario para que NativeWind procese las clases.

Primer test con NativeWind

```
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>
  );
}
```

- Uso de `className` en vez de `style`.

- Pantalla pequeña y orientación variable.
- No hay **hover**, solo toque/gestos.
- Textos deben ser legibles en pantallas con DPI alto.
- Considerar **notch**, barra de estado y zonas no utilizables.
- Usuario en movimiento, no sentado frente a un monitor.

- Layout basado en **Flexbox**:
 - `flex-1`, `flex-row`, `items-center`, `justify-between`, etc.
- NativeWind mapea estas utilidades a estilos nativos.

```
<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>
```

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>
  );
}
```

- Agregamos **NativeWind + Tailwind** a un proyecto Expo en JS.
- Editamos `tailwind.config.js` y `babel.config.js`.
- Usamos `className` para estilos.
- Vimos:
 - Pantalla pequeña y DPI alto.
 - Flexbox como base de layout.
 - **ScrollView** para contenido largo.
 - **SafeAreaView** para zonas seguras.
 - Componentes táctiles (**Pressable**).

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