Appearance

```
import {Appearance} from 'react-native';
```

The `Appearance` module exposes information about the user's appearance preferences, such as their preferred color scheme (light or dark).

Developer notes

Android iOS **Web**

> The `Appearance` API is inspired by the Media Queries draft from the W3C. The color scheme preference is modeled after the `prefers-color-scheme` CSS media feature.

Example

You can use the `Appearance` module to determine if the user prefers a dark color scheme:

```
const colorScheme = Appearance.getColorScheme();
if (colorScheme === 'dark') {
  // Use dark color scheme
}
```

Although the color scheme is available immediately, this may change (e.g. scheduled color scheme change at sunrise or sunset). Any rendering logic or styles that depend on the user preferred color scheme should try to call this function on every render, rather than caching the value. For example, you may use the `useColorScheme` React hook as it provides and subscribes to color scheme updates, or you may use inline styles rather than setting a value in a `StyleSheet`.

Reference

Methods

`getColorScheme()`

```
static getColorScheme(): 'light' | 'dark' | null;
```

Indicates the current user preferred color scheme. The value may be updated later, either through direct user action (e.g. theme selection in device settings or application-level selected user interface style via `setColorScheme`) or on a schedule (e.g. light and dark themes that follow the day/night cycle).

Supported color schemes:

- `light`: The user prefers a light color theme.
- `dark`: The user prefers a dark color theme.
- null: The user has not indicated a preferred color theme.

See also: `useColorScheme` hook.

> Note: `getColorScheme()` will always return `light` when debugging with Chrome.

`setColorScheme()`

```
static setColorScheme('light' | 'dark' | null): void;
```

Force the application to always adopt a light or dark interface style. The default value is `null` which causes the application to inherit the system's interface style. If you assign a different value, the new style applies to the application and all native elements within the application (Alerts, Pickers etc).

Supported color schemes:

- `light` : Apply light user interface style.
- `dark` : Apply dark user interface style.
- null: Follow the system's interface style.

> Note: The change will not affect the system's selected interface style or any style set in other applications.

`addChangeListener()`

```
static addChangeListener(
  listener: (preferences: {colorScheme: 'light' | 'dark' | null}) => void,
): NativeEventSubscription;
```

Add an event handler that is fired when appearance preferences change.

*Last updated on **Jun 21, 2023***