AppState

`AppState` can tell you if the app is in the foreground or background, and notify you when the state changes.

AppState is frequently used to determine the intent and proper behavior when handling push notifications.

App States

- `active` - The app is running in the foreground
- `background` - The app is running in the background. The user is either:
  - in another app
  - on the home screen
  - [Android] on another `Activity` (even if it was launched by your app)
- [iOS] `inactive` - This is a state that occurs when transitioning between foreground & background, and during periods of inactivity such as entering the multitasking view, opening the Notification Center or in the event of an incoming call.

For more information, see Apple's documentation

Basic Usage

To see the current state, you can check `AppState.currentState`, which will be kept up-to-date. However, `currentState` will be null at launch while `AppState` retrieves it over the bridge.

AppState Example

Expo

```
import React, {useRef, useState, useEffect} from 'react';
import {AppState, StyleSheet, Text, View} from 'react-
native';

const AppStateExample = () => {
  const appState = useRef(AppState.currentState);
  const [appStateVisible, setAppStateVisible] =
useState(appState.current);

  useEffect(() => {
    const subscription =
AppState.addEventListener('change', nextAppState => {
      if (
        appState.current.match(/inactive|background/) &&
        nextAppState === 'active'
      ) {
        console.log('App has come to the foreground!');
      }

      appState.current = nextAppState;
      setAppStateVisible(appState.current);
      console.log('AppState', appState.current);
    });

    return () => {
      subscription.remove();
    };
```

Preview | My Device | iOS | Android | Web

This example will only ever appear to say "Current state is: active" because the app is only visible to the user when in the `active` state, and the null state will happen only momentarily. If you want to experiment with the code we recommend to use your own device instead of embedded preview.

Reference

Events

change

This event is received when the app state has changed. The listener is called with one of the current app state values.

`memoryWarning`

This event is used in the need of throwing memory warning or releasing it.

`focus` Android

Received when the app gains focus (the user is interacting with the app).

`blur` Android

Received when the user is not actively interacting with the app. Useful in situations when the user pulls down the notification drawer. `AppState` won't change but the `blur` event will get fired.

Methods

`addEventListener()`

```
static addEventListener(
  type: AppStateEvent,
  listener: (state: AppStateStatus) => void,
): NativeEventSubscription;
```

Sets up a function that will be called whenever the specified event type on AppState occurs. Valid values for `eventType` are listed above. Returns the `EventSubscription`.

Properties

`currentState`

```
static currentState: AppStateStatus;
```

Is this page useful? 

Edit this page

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