ActionSheetIOS

Displays native to iOS Action Sheet component.

Example

ActionSheetIOS   ∧ **Expo**

```
import React, {useState} from 'react';
import {ActionSheetIOS, Button, StyleSheet, Text, View}
from 'react-native';

const App = () => {
  const [result, setResult] = useState('🔮');

  const onPress = () =>
    ActionSheetIOS.showActionSheetWithOptions(
      {
        options: ['Cancel', 'Generate number', 'Reset'],
        destructiveButtonIndex: 2,
        cancelButtonIndex: 0,
        userInterfaceStyle: 'dark',
      },
      buttonIndex => {
        if (buttonIndex === 0) {
          // cancel action
        } else if (buttonIndex === 1) {
          setResult(String(Math.floor(Math.random() * 100)
+ 1));
        } else if (buttonIndex === 2) {
          setResult('🔮');
        }
      },
    );
```

Download Expo Go and scan the QR code to get started.

Connected devices   0

Log in or set a Device ID to open this Snack from Expo Go on your device or simulator.

Preview ⬤◯ | My Device | iOS

Reference

Methods

`showActionSheetWithOptions()`

```
static showActionSheetWithOptions: (
  options: ActionSheetIOSOptions,
  callback: (buttonIndex: number) => void,
);
```

Display an iOS action sheet. The `options` object must contain one or more of:

- `options` (array of strings) - a list of button titles (required)
- `cancelButtonIndex` (int) - index of cancel button in `options`
- `cancelButtonTintColor` (string) - the color used for the change the text color of the cancel button
- `destructiveButtonIndex` (int or array of ints) - indices of destructive buttons in `options`
- `title` (string) - a title to show above the action sheet
- `message` (string) - a message to show below the title
- `anchor` (number) - the node to which the action sheet should be anchored (used for iPad)
- `tintColor` (string) - the color used for non-destructive button titles
- `disabledButtonIndices` (array of numbers) - a list of button indices which should be disabled
- `userInterfaceStyle` (string) - the interface style used for the action sheet, can be set to `light` or `dark`, otherwise the default system style will be used

The 'callback' function takes one parameter, the zero-based index of the selected item.

Minimal example:

```
ActionSheetIOS.showActionSheetWithOptions(
  {
    options: ['Cancel', 'Remove'],
    destructiveButtonIndex: 1,
    cancelButtonIndex: 0,
  },
  buttonIndex => {
    if (buttonIndex === 1) {
      /* destructive action */
```

```
    }
  },
);
```

dismissActionSheet()

```
static dismissActionSheet();
```

Dismisses the most upper iOS action sheet presented, if no action sheet is present a warning is displayed.

showShareActionSheetWithOptions()

```
static showShareActionSheetWithOptions: (
  options: ShareActionSheetIOSOptions,
  failureCallback: (error: Error) => void,
  successCallback: (success: boolean, method: string) => void,
);
```

Display the iOS share sheet. The `options` object should contain one or both of `message` and `url` and can additionally have a `subject` or `excludedActivityTypes`:

- `url` (string) - a URL to share
- `message` (string) - a message to share
- `subject` (string) - a subject for the message
- `excludedActivityTypes` (array) - the activities to exclude from the ActionSheet

> **Note:** If `url` points to a local file, or is a base64-encoded uri, the file it points to will be loaded and shared directly. In this way, you can share images, videos, PDF files, etc. If `url` points to a remote file or address it must conform to URL format as described in RFC 2396. For example, a web URL without a proper protocol (HTTP/HTTPS) will not be shared.

The 'failureCallback' function takes one parameter, an error object. The only property defined on this object is an optional `stack` property of type `string`.

The 'successCallback' function takes two parameters:

- a boolean value signifying success or failure
- a string that, in the case of success, indicates the method of sharing

Is this page useful? 👍 👎

✏️ Edit this page

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