

Platform

Example

Platform API Example



```
import React from 'react';
import {Platform, StyleSheet, Text, ScrollView} from 'react-native';

const App = () => {
  return (
    <ScrollView contentContainerStyle={styles.container}>
      <Text>OS</Text>
      <Text style={styles.value}>{Platform.OS}</Text>
      <Text>OS Version</Text>
      <Text style={styles.value}>{Platform.Version}</Text>
      <Text>isTV</Text>
      <Text style={styles.value}>{Platform.isTV.toString()}</Text>
      {Platform.OS === 'ios' && (
        <>
          <Text>isPad</Text>
          <Text style={styles.value}>{Platform.isPad.toString()}</Text>
        </>
      )}
      <Text>Constants</Text>
      <Text style={styles.value}>
        {JSON.stringify(Platform.constants, null, 2)}
      </Text>
    </ScrollView>
  );
};
```

Preview



My Device

iOS

Android

Reference












Properties

constants

```
static constants: PlatformConstants;
```

Returns an object which contains all available common and specific constants related to the platform.

Properties:

| NAME | TYPE | OPTIONAL | DESCRIPTION |
|---|---------|----------|--|
| isTesting | boolean | No | |
| reactNativeVersion | object | No | Information about React Native version. Keys are <code>major</code> , <code>minor</code> , <code>patch</code> with optional <code>prerelease</code> and values are number s. |
| Version  Android | number | No | OS version constant specific to Android. |
| Release  Android | string | No | |
| Serial  Android | string | No | Hardware serial number of an Android device. |
| Fingerprint  Android | string | No | A string that uniquely identifies the build. |
| Model  Android | string | No | The end-user-visible name for the Android device. |
| Brand  Android | string | No | The consumer-visible brand with which the product/hardware will be associated. |
| Manufacturer  Android | string | No | The manufacturer of the Android device. |
| ServerHost  Android | string | Yes | |
| uiMode  Android | string | No | Possible values are: <code>'car'</code> , <code>'desk'</code> , <code>'normal'</code> , <code>'tv'</code> , <code>'watch'</code> and <code>'unknown'</code> . Read more about Android ModeType . |
| forceTouchAvailable  iOS | boolean | No | Indicate the availability of 3D Touch on a device. |
| interfaceIdiom  iOS | string | No | The interface type for the device. Read more about UIUserInterfaceIdiom . |

| NAME | TYPE | OPTIONAL | DESCRIPTION |
|------------------|--------|----------|--------------------------------------|
| osVersion ◀ iOS | string | No | OS version constant specific to iOS. |
| systemName ◀ iOS | string | No | OS name constant specific to iOS. |

isPad ◀ iOS

```
static isPad: boolean;
```

Returns a boolean which defines if device is an iPad.

| TYPE |
|---------|
| boolean |

isTV

```
static isTV: boolean;
```

Returns a boolean which defines if device is a TV.

| TYPE |
|---------|
| boolean |

isTesting

```
static isTesting: boolean;
```

Returns a boolean which defines if application is running in Developer Mode with testing flag set.

TYPE

boolean

OS

```
static OS: 'android' | 'ios';
```

Returns string value representing the current OS.

TYPE

enum('android', 'ios')

Version

```
static Version: 'number' | 'string';
```

Returns the version of the OS.

TYPEnumber  Androidstring  iOS

Methods

select()

```
static select(config: Record<string, T>): T;
```

Returns the most fitting value for the platform you are currently running on.

Parameters:

| NAME | TYPE | REQUIRED | DESCRIPTION |
|--------|--------|----------|-------------------------------|
| config | object | Yes | See config description below. |

Select method returns the most fitting value for the platform you are currently running on. That is, if you're running on a phone, `android` and `ios` keys will take preference. If those are not specified, `native` key will be used and then the `default` key.

The `config` parameter is an object with the following keys:

- `android` (any)
- `ios` (any)
- `native` (any)
- `default` (any)

Example usage:

```
import {Platform, StyleSheet} from 'react-native';

const styles = StyleSheet.create({
  container: {
    flex: 1,
    ...Platform.select({
      android: {
        backgroundColor: 'green',
      },
      ios: {
        backgroundColor: 'red',
      },
      default: {
        // other platforms, web for example
        backgroundColor: 'blue',
      },
    }),
  },
});
```

This will result in a container having `flex: 1` on all platforms, a green background color on Android, a red background color on iOS, and a blue background color on other platforms.

Since the value of the corresponding platform key can be of type `any`, `select` method can also be used to return platform-specific components, like below:

```
const Component = Platform.select({  
  ios: () => require('ComponentIOS'),  
  android: () => require('ComponentAndroid'),  
})();
```

```
<Component />;
```

```
const Component = Platform.select({  
  native: () => require('ComponentForNative'),  
  default: () => require('ComponentForWeb'),  
})();
```

```
<Component />;
```

Is this page useful?  

 Edit this page

Last updated on **Jun 21, 2023**