Version: 3.x

Troubleshooting

Initialization issues

Reanimated has four core components that compose its code:

- C++,
- Java,
- JavaScript,
- Reanimated Babel plugin.

All of them are supposed to work correctly only within the same minor version. Therefore, having any of those pieces in your code - directly or indirectly - separate from `react-native-reanimated`, especially having any code transpiled with a different version of the aforementioned plugin will result in undefined behavior and errors.

Failed to create a worklet

Problem: This usually happens when Reanimated is not properly installed, e.g. forgetting to include the Reanimated Babel plugin in `babel.config.js`.

Solution: See installation docs at https://docs.swmansion.com/react-native-reanimated/docs/fundamentals/installation#babel-plugin for more information.

Native part of Reanimated doesn't seem to be initialized

Problem: This issue happens when Reanimated fails to initialize its native side from JavaScript.

Solution:

1. If you recently installed or upgraded Reanimated, make sure to rebuild your app.
2. Check if your platform is supported by Reanimated. Currently we support:
   - Android
   - iOS

    - macOS
    - Web
3. If you are using Reanimated in a brownfield app, make sure to initialize the native library manually.

Unknown version of Reanimated Babel plugin

Problem: This happens when JavaScript side of Reanimated fails to get Reanimated Babel plugin version.

Solution:

1. Part of your code might be transpiled with an outdated version of Reanimated Babel plugin. See **Mismatch between JavaScript code version and Reanimated Babel plugin version**.
2. You use release bundle with debug build of the app. This is not supported. See **Using dev bundle in release app build is not supported** for more information.

Mismatch between JavaScript code version and Reanimated Babel plugin version

Problem: This can happen when you use code that was transpiled with an outdated version of Reanimated Babel plugin.

Solution: Try resetting your Metro bundler cache with `yarn start --reset-cache`, `npm start -- --reset-cache` or `expo start -c` and run the app again.

If this didn't help, you probably have a dependency that contains already transformed worklets bundled with an outdated version of the Reanimated Babel plugin. You can find the offending code that was given alongside the error to find that dependency.

Using dev bundle in a release app build is not supported

Problem: This happens when you use a release build of your app with a development JavaScript bundle.

Solution: See this post for more information.

Couldn't determine the version of the native part of Reanimated

Problem: This happens when Reanimated fails to determine the version of its native part.

Solution: Check if you have rebuilt your app after upgrading `react-native-reanimated`. If you use Expo Go, you must use the exact version which is bundled into Expo SDK.

Mismatch between JavaScript part and native part of Reanimated

Problem: This happens when Reanimated has different versions of its JavaScript and native parts.

Solution: Check if you have rebuilt your app after upgrading `react-native-reanimated`. If you use Expo Go, you must use the exact version which is bundled into Expo SDK.

Multiple versions of Reanimated were detected

Problem: This error usually happens when in your project exists more than one instance of Reanimated. It can occur when some of your dependency has installed Reanimated inside their own `node_modules` instead of using it as a peer dependency. In this case two different versions of Reanimated JavaScript module might try to initialize its native part. You can check which libraries are using Reanimated e.g. with `yarn why react-native-reanimated` or `npm ls react-native-reanimated`.

Solution: Modify your `package.json` file accordingly:

- if you use `yarn`, you should add `resolution` property:

```
"resolutions": {
  "react-native-reanimated": <Reanimated version>
}
```

- if you use `npm`, you should add `overrides` property:

```
"overrides": {
  "react-native-reanimated": <Reanimated version>
}
```

After that make sure to run your package manager again, either `yarn` or `npm install`.

Outdated version of React Native for New Architecture

Problem: Reanimated supports New Architecture (Fabric) only on the latest minor release of React Native.

Solution: Please upgrade to at least React Native 0.72 or downgrade to an older version of Reanimated 3.

Threading issues

Tried to synchronously call a non-worklet function on the UI thread

Problem: This can happen when you try to call a function that is not marked as a worklet from a worklet. E.g.:

```
function callee() {
  console.log('hello');
}
function caller() {
  'worklet';
  callee(); // <- this will throw in `runOnUI`
}
runOnUI(caller)();
```

In this example, `callee` cannot be called from a worklet ran on UI thread because there is no corresponding UI function for `callee`.

Solution:

1. If you want to synchronously execute this method, mark it as a worklet using `worklet` directive:

```
 function callee() {
+  'worklet';
```

```
    console.log("hello");
  }
```

2. If you want to execute this function on the JS thread, wrap it using `runOnJS`:

```
  function caller() {
    'worklet';
-   callee();
+   runOnJS(callee)();
  }
```

Check **this page** to learn more about Reanimated and its worklets.

Edit this page