

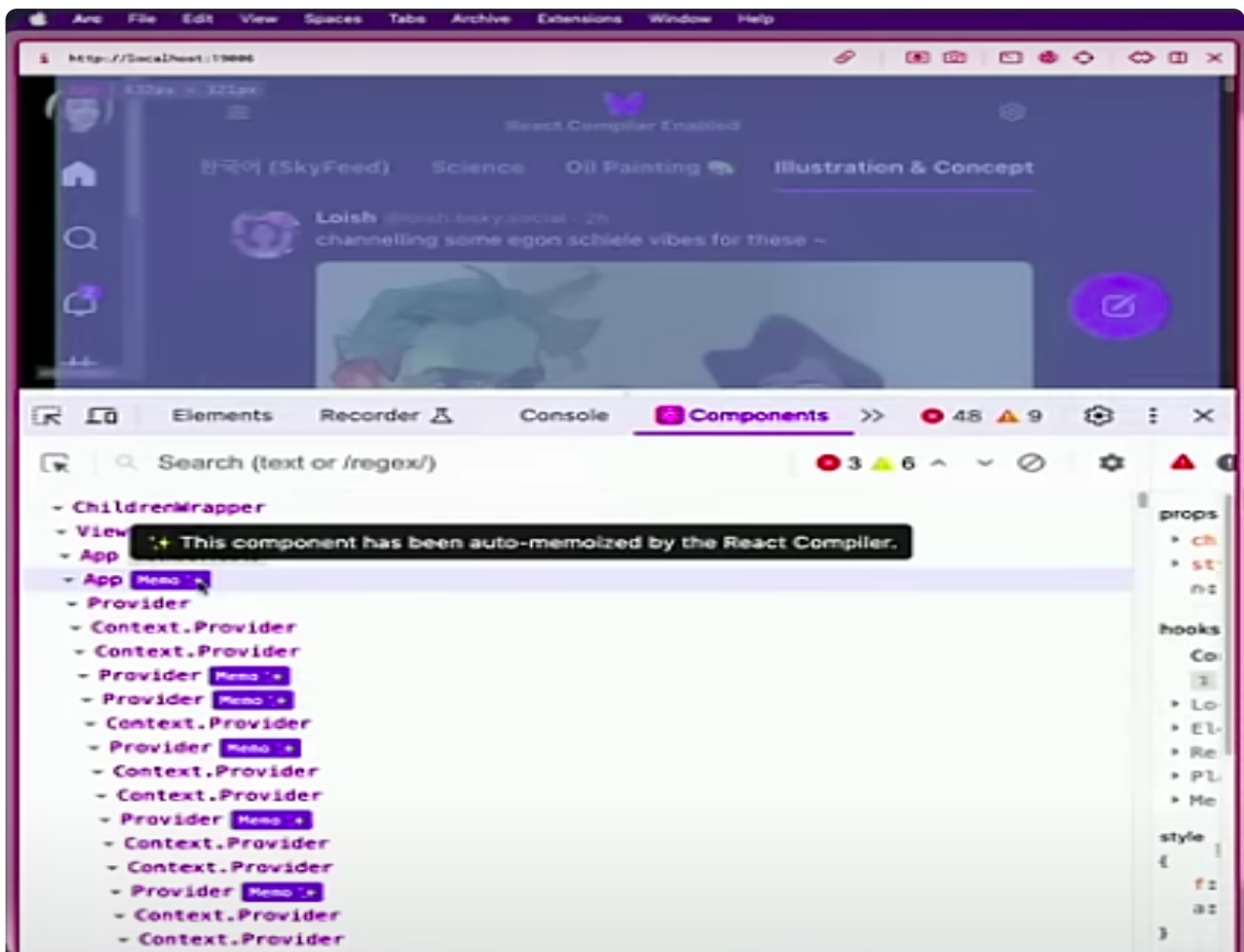


5. Forget About Memo

Demo

```
babel.config.js
module.exports = function (api) {
  const config = {
    presets: [
    ],
    plugins: [
      'babel-plugin-react-compiler',
      {
        runtimeModule: 'react-compiler-runtime',
        environment: {
          validateNoRefAccessInRender: true,
        },
      },
    ],
  };
  return config;
};
```

The screenshot shows a code editor with a dark theme. The file is named 'babel.config.js'. The code defines a function that returns a Babel configuration object. The configuration includes an empty array of presets and an array of plugins. The first plugin is 'babel-plugin-react-compiler'. The second plugin is an object with 'runtimeModule' set to 'react-compiler-runtime' and an 'environment' object containing 'validateNoRefAccessInRender' set to 'true'. The status bar at the bottom indicates the file is in JavaScript, using UTF-8 encoding, and is 10 lines long.

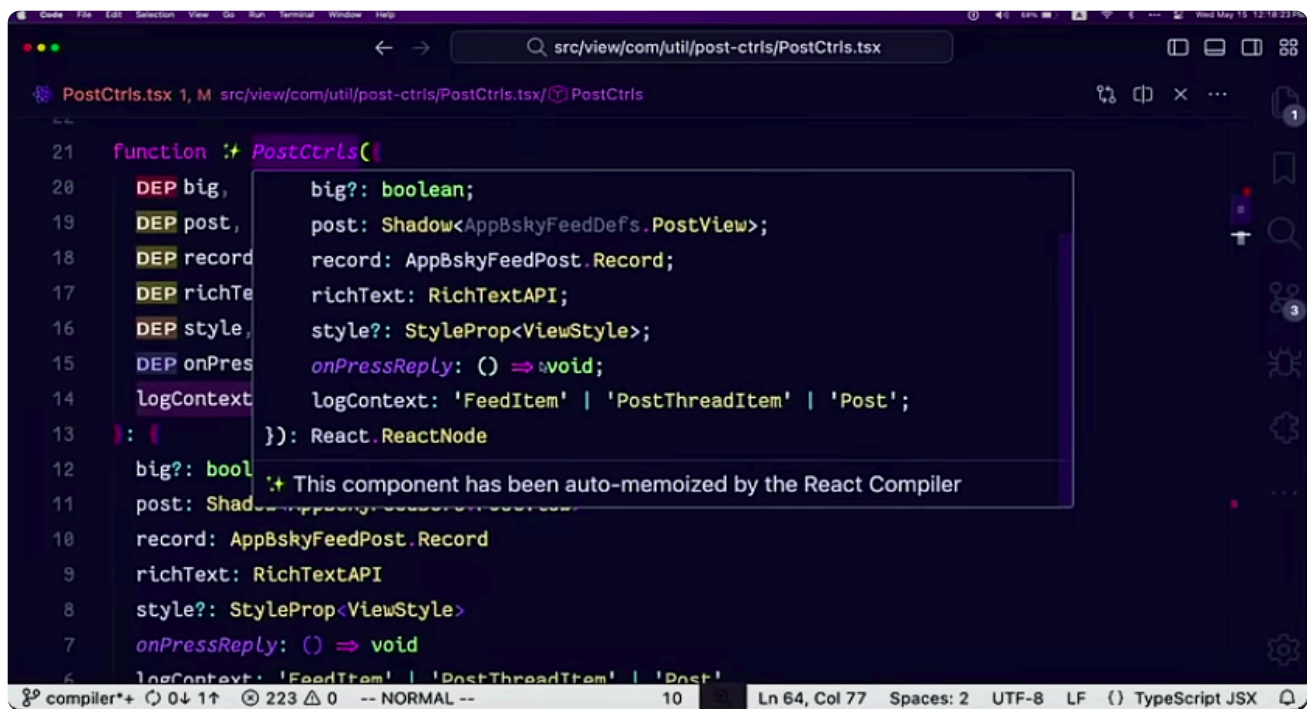


```
Code File Edit Selection View Go Run Terminal Window Help
>js pre|
PostCtrls.tsx 1, M src/ React Compiler: Toggle JS preview recently used
Icons: Toggle NestJS Preset (Workspace Level) other commands
Icons: Toggle Official JS Preset (User Level)
Icons: Toggle Official JSON Preset (User Level)
1
43 function Pos
1 big,
2 post,
3 record,
4 richText,
5 style,
6 onPressReply,
7 logContext,
8 ): {
9 big?: boolean
10 post: Shadow<AppBskyFeedDefs.PostView>
11 record: AppBskyFeedPost.Record
12 richText: RichTextAPI
13 style?: StyleProp<ViewStyle>
14 onPressReply: () => void
10 Ln 43, Col 1 Spaces: 2 UTF-8 LF {} TypeScript JSX
```

```
Code File Edit Selection View Go Run Terminal Window Help
src/view/com/util/post-ctrls/PostCtrls.tsx
PostCtrls.tsx 1, M src/view/com/util/post-ctrls/Pos React Compiler: /Users/lauren/code/social-app/src/view, X
1
43 function PostCtrls((
1 big,
2 post,
3 record,
4 richText,
5 style,
6 onPressReply,
7 logContext,
8 ): {
9 big?: boolean
10 post: Shadow<AppBskyFeedDefs.PostView>
11 record: AppBskyFeedPost.Record
12 richText: RichTextAPI
13 style?: StyleProp<ViewStyle>
14 onPressReply: () => void
10 Ln 43, Col 1 Spaces: 2 UTF-8 LF {} TypeScript JSX

function PostCtrls(t0) {
  const $ = useMemoCache(114)
  const { big, post, record, richText, style, onPressReply, logContext } = t0

  const theme = useTheme()
  const { _ } = useLingui()
  const { openComposer } = useComposerControls()
  const { closeModal } = useModalControls()
  const [queueLike, queueUnlike] = usePostLikeMutationQueue(post, logContext)
  const [queueRepost, queueUnrepost] = usePostRepostMutationQueue(post,
```



```
21 function PostCtrls(  
20   DEP big,      big?: boolean;  
19   DEP post,     post: Shadow<AppBskyFeedDefs.PostView>;  
18   DEP record,   record: AppBskyFeedPost.Record;  
17   DEP richText, richText: RichTextAPI;  
16   DEP style,    style?: StyleProp<ViewStyle>;  
15   DEP onPress,  onPressReply: () => void;  
14   logContext    logContext: 'FeedItem' | 'PostThreadItem' | 'Post';  
13 ): {           }): React.ReactNode  
12   big?: boolean; + This component has been auto-memoized by the React Compiler  
11   post: Shadow<AppBskyFeedDefs.PostView>;  
10   record: AppBskyFeedPost.Record  
9   richText: RichTextAPI  
8   style?: StyleProp<ViewStyle>  
7   onPressReply: () => void  
6   logContext: 'FeedItem' | 'PostThreadItem' | 'Post';
```

Ln 64, Col 77 Spaces: 2 UTF-8 LF TypeScript JSX



fb.me/react-compiler



Works on every platform

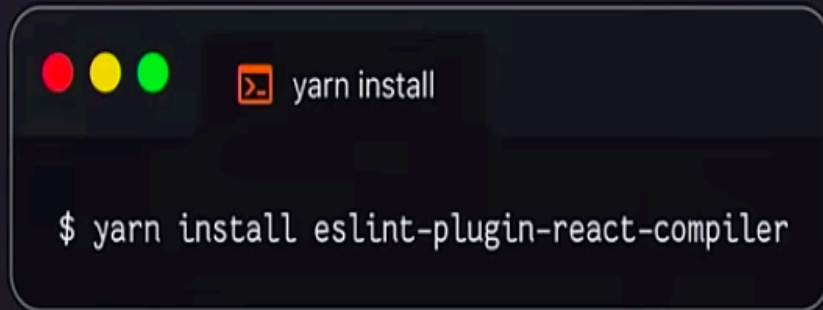


Automatic fine-grained memo



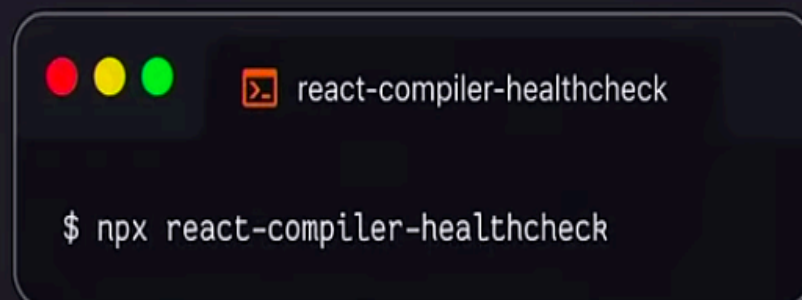
Drop in, no rewrites

Find and fix Rules of React issues

A terminal window with a dark background and rounded corners. It features three colored window control buttons (red, yellow, green) in the top-left corner. The prompt is a red icon with a white terminal symbol. The command 'yarn install' is entered and executed. Below the command, the output '\$ yarn install eslint-plugin-react-compiler' is shown.

```
yarn install  
  
$ yarn install eslint-plugin-react-compiler
```

Estimate the effort

A terminal window with a dark background and rounded corners. It features three colored window control buttons (red, yellow, green) in the top-left corner. The prompt is a red icon with a white terminal symbol. The command 'react-compiler-healthcheck' is entered and executed. Below the command, the output '\$ npx react-compiler-healthcheck' is shown.

```
react-compiler-healthcheck  
  
$ npx react-compiler-healthcheck
```

React Compiler memoizes Components and Hooks



The screenshot shows a code editor with three tabs: 'TableContainer.js', 'Compiled [DEV]', and 'expensivelyProcessAReallyLargeArrayOfObjects....'. The code in the 'Compiled [DEV]' tab is as follows:

```
// naive memoization example
let lastHash = null
let lastResult = null

function expensivelyProcessAReallyLargeArrayOfObjects(items) {
  const itemHash = hash(items)
  if (lastHash !== null) {
    return lastResult
  }
}
```