Profile Different Interactions

In this lesson, we'll discuss different interactions which are recorded during the profiled session.

WE'LL COVER THE FOLLOWING

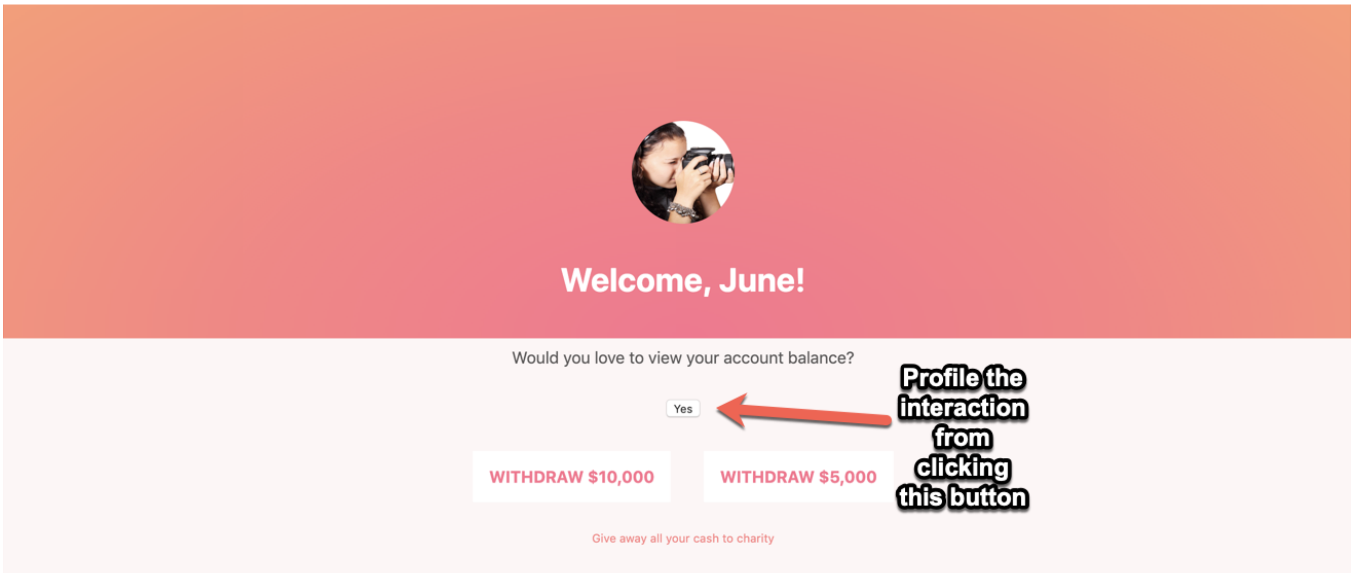
- Using `React.memo`
- `displayName` Property

It's easy to assume that because we had fewer re-renders in the "withdraw amount" interaction, we now have an optimized app.

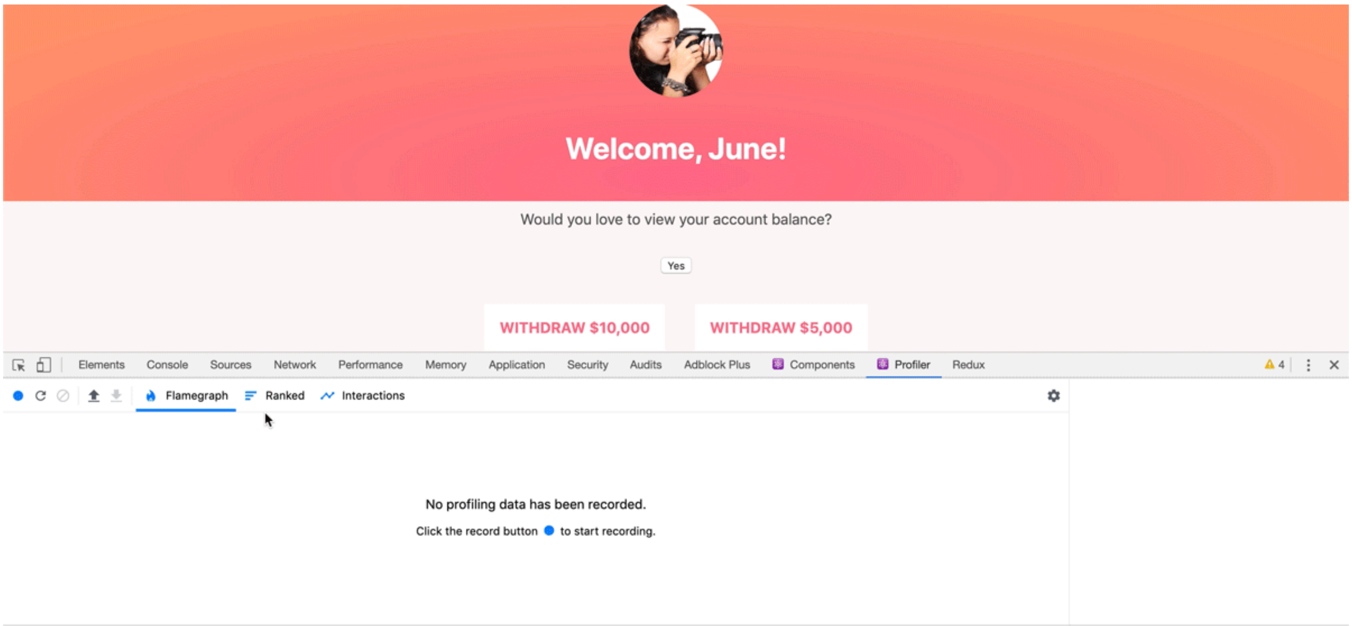
That's not correct.

`App` is now a `PureComponent`, but what happens when `App` gets rendered due to a `state` change?

Well, let's profile a different interaction. This time load up the application and profile the interaction for viewing the account balance.


Welcome, June!
Would you love to view your account balance?
Yes
WITHDRAW $10,000
WITHDRAW $5,000
Give away all your cash to charity
Profile the interaction from clicking this button

If you go ahead and profile the interaction, we get a completely different result than the "withdraw amount" interaction.

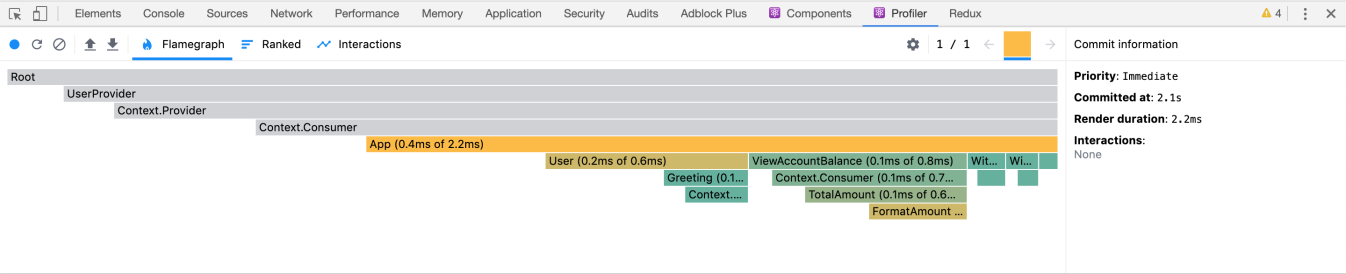


No profiling data has been recorded.

Click the record button ● to start recording.

Screencast showing the interaction being profiled.

What’s changed?

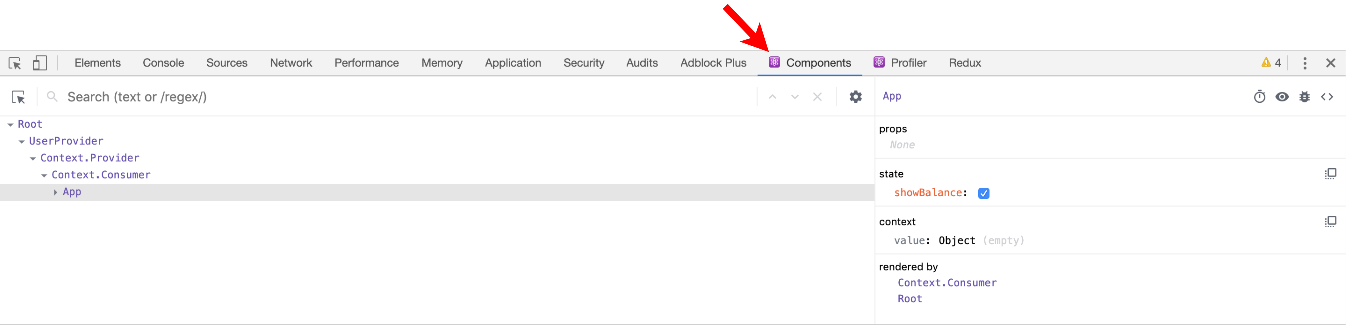


Commit information

Priority: `Immediate`
Committed at: `2.1s`
Render duration: `2.2ms`
Interactions:
None

From the flame graph above, every child component of `App` has been re-rendered. They had nothing to do with this visual update, so those are wasted renders.

If you want to check the hierarchy of components, you can click the components tab:



```
▾ Root
  ▾ UserProvider
    ▾ Context.Provider
      ▾ Context.Consumer
        ▸ App
```

App

props
None

state
showBalance: ☑

context
value: Object (empty)

rendered by
Context.Consumer
Root

Using `React.memo`

Since these child components are functional components, let’s use

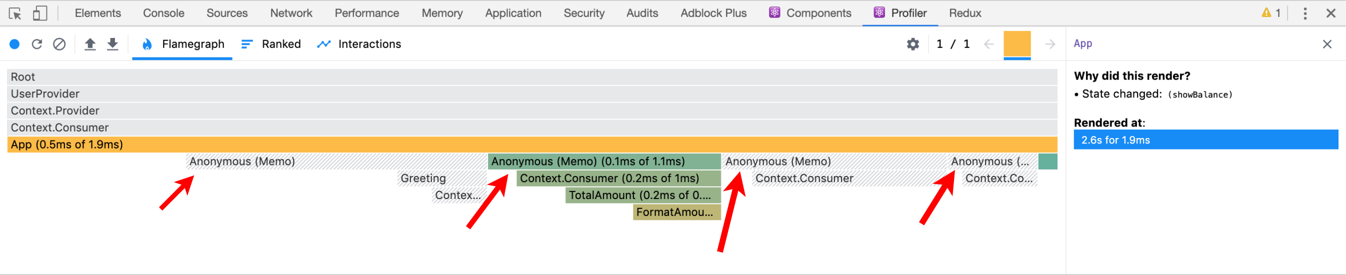
Since these child components are functional components, let's use `React.memo` to memoize the render results so they don't change unless there's a change in `props`.

```
// User.js
import { memo } from 'react'
const User = memo(({ profilePic }) => {
  ...
})
// ViewAccountBalance.js
import { memo } from 'react'
const ViewAccountBalance = memo(({ showBalance, displayBalance }) => {
      ...
})
// WithdrawButton.js
import { memo } from 'react'
const WithdrawButton = memo(({ amount }) => {
    ...
  )
})
```

Now, only `ViewAccountBalance` and other child components are re-rendered. That's okay.

When you view your flame chart, you may see something slightly different.

The names of the component may not be shown. You get the generic name `Anonymous(Memo)` and it becomes difficult to track which component is what.



Why did this render?
• State changed: (showBalance)

Rendered at:
2.6s for 1.9ms

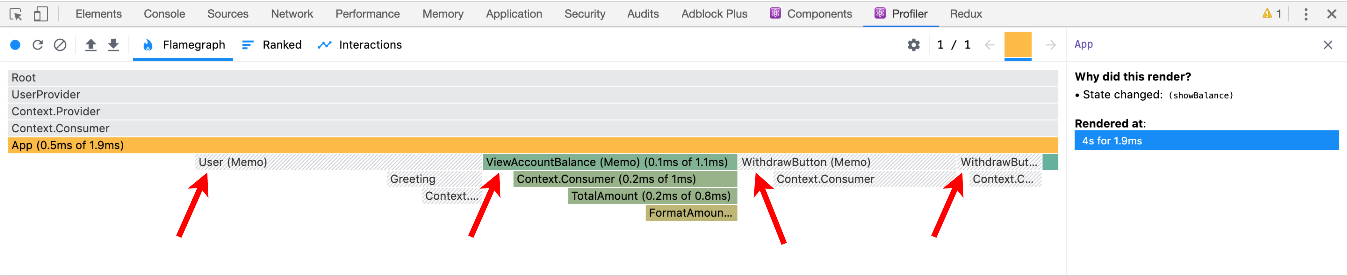
`displayName` Property

To change this, set the `displayName` property for the memoized components.

Below is an example.

```
// ViewAccountBalance.js
const ViewAccountBalance = memo(({ showBalance, displayBalance }) => {
  ...
})
// set the displayName here
ViewAccountBalance.displayName = 'ViewAccountBalance'
```

Let’s go ahead and do this for all the memoized functional components.



Why did this render?

- State changed: (showBalance)

Rendered at:

4s for 1.9ms

In the next lesson, we’ll discuss the `Provider` value and how it is useful.