



# React: displayName

Because trying to debug a nest of Unknown components is kind of meh..

# Functional Components

```
const LogoBackdrop = ({ url, color, children }) => {  
  const className = 'bc-program-card__logo-backdrop'  
  const style = { backgroundColor: color }  
  
  const sharedProps = { className, style }  
  
  if (url) {  
    return (  
      <a href={url} {...sharedProps} aria-hidden tabIndex="-1">  
        {children}  
      </a>  
    )  
  } else {  
    return <div {...sharedProps}>{children}</div>  
  }  
}
```



# React 16.6



React

Docs

Tutorial

Community

Blog

## React v16.6.0: lazy, memo and createContext

October 23, 2018 by [Sebastian Markbåge](#)

Today we're releasing React 16.6 with a few new convenient features. A form of PureComponent/shouldComponentUpdate for function components, a way to do code splitting using Suspense and an easier way to consume Context from class components.

<https://reactjs.org/blog/2018/10/23/react-v-16-6.html>



# React.memo

Like `React.PureComponent`, but functional!

## `React.memo`

```
const MyComponent = React.memo(function MyComponent(props) {  
  /* render using props */  
});
```

`React.memo` is a higher order component. It's similar to `React.PureComponent` but for function components instead of classes.

<https://reactjs.org/docs/react-api.html#reactmemo>

# Memoized Functional Components

```
const LogoBackdrop = React.memo( type: ({ url, color, children }) => {
  const className = 'bc-program-card__logo-backdrop'
  const style = { backgroundColor: color }

  const sharedProps = { className, style }

  if (url) {
    return (
      <a href={url} {...sharedProps} aria-hidden tabIndex="-1">
        {children}
      </a>
    )
  } else {
    return <div {...sharedProps}>{children}</div>
  }
})
```

# React Developer Tools



chrome web store



## React Developer Tools

Offered by: Facebook



1,086

[Developer Tools](#)



1,437,000 users

# Memo -> Memo -> Unknown -> WTF?

```
▼ <Unknown csrfToken="VZAo4FU9cKUJxFjzMohsFVa"
={false} isLoadingNewList={false} showLoad
▼ <div>
  ▼ <ul className="row bc-panels bc-panels
    ▼ <Unknown isLoggedInIn={true} loginUrl
      ► <Memo key="20f81590-a309-46d6-ac3
        VZAo4FU9cKUJxFjzMohsFVa7jz5YGB1e0
      ▼ <Memo key="bugcrowdswag" isLoggedIn
        VZAo4FU9cKUJxFjzMohsFVa7jz5YGB1e0
      ▼ <ProgramCard>
        ▼ <li className="col-xs-12 col
          ▼ <div className="bc-panel b
            ▼ <div className="bc-pr
              ▼ <Memo url="/bugcrow
                ► <Unknown url="/b
              </Memo>
              ▼ <h4 className="bc-p
                <a href="/bugcro
              </h4>
              ▼ <Memo isTrial={fal
                <Unknown isTrial
              </Memo>
              <p className="bc-p
            </div>
          </div>
        </Memo>
```

```
<div className="bc-panel">  
  <Memo name="Bugcrowd Sw..."  
    http://bugcrowd.test:30  
    <div className="bc-pr..."  
      <Memo url="/bugcro..."  
        <Unknown url="/b...  
      </Memo>  
      <h4 className="bc-..."  
        <a href="/bugcro...  
      </h4>  
      <Memo isTrial={fal...  
        <Unknown isTrial...  
      </Memo>  
    </div>  
  </div>
```



# displayName?

## **displayName**

The `displayName` string is used in debugging messages. Usually, you don't need to set it explicitly because it's inferred from the name of the function or class that defines the component. You might want to set it explicitly if you want to display a different name for debugging purposes or when you create a higher-order component, see [Wrap the Display Name for Easy Debugging](https://reactjs.org/docs/react-component.html#displayname) for details.

<https://reactjs.org/docs/react-component.html#displayname>





# Maybe not..?

ES6 'Fat Arrow' **anonymous** function

```
const LogoBackdrop = React.memo( type: ({ url, color, children }) => {  
  const className = 'bc-program-card__logo-backdrop'  
  const style = { backgroundColor: color }  
}
```

**Named** function

```
const LogoBackdrop = React.memo( function LogoBackdrop({ url, color, children }) {  
  const className = 'bc-program-card__logo-backdrop'  
  const style = { backgroundColor: color }  
}
```



# Wrapping Display Name

## Convention: Wrap the Display Name for Easy Debugging

The container components created by HOCs show up in the React Developer Tools like any other component. To ease debugging, choose a display name that communicates that it's the result of a HOC.

The most common technique is to wrap the display name of the wrapped component. So if your higher-order component is named `withSubscription`, and the wrapped component's display name is `CommentList`, use the display name `WithSubscription(CommentList)`:

<https://reactjs.org/docs/higher-order-components.html#convention-wrap-the-display-name-for-easy-debugging>

# memoWithDisplayName

```
/** Explicitly set the displayName on the component that is passed in. ...*/
export const withDisplayName = (displayName, component) => {
  component.displayName = displayName
  return component
}

/** Explicitly set the displayName on the component that is passed in, then wrap
export const memoWithDisplayName = (
  displayName,
  component,
  areEqual = undefined
) => {
  const memoComponent = memo(withDisplavName(displavName, component), areEqual)
  memoComponent.displayName = `Memo(${component.displayName})`
  return memoComponent
}
```

app/javascript/src/shared/react-helpers.js

# memoWithDisplayName

```
const LogoBackdrop = memoWithDisplayName(  
  displayName: 'LogoBackdrop',  
  component: ({ url, color, children }) => {  
    const className = 'bc-program-card__logo-backdrop'  
    const style = { backgroundColor: color }  
  
    const sharedProps = { className, style }  
  
    if (url) {  
      return (  
        <a href={url} {...sharedProps} aria-hidden tabIndex="-1">  
          {children}  
        </a>  
      )  
    } else {  
      return <div {...sharedProps}>{children}</div>  
    }  
  }  
)
```

# Components.. withDisplayName(s)!

```
▼ <ProgramList csrfToken="/zCbzpWBg1rXeRqV0Ium50LHC/qdc9BY"
  isLoadingMore={false} isLoadingNewList={false}
  ▼ <div>
    ▼ <ul className="row bc-panels bc-panel"
      ▼ <ProgramEntries isLoggedIn={true}
        ► <Memo(Program) key="a29f7b09-d62"
          /zCbzpWBg1rXeRqV0Ium50LHC/qdc9BY
        ▼ <Memo(Program) key="bugcrowdswag"
          /zCbzpWBg1rXeRqV0Ium50LHC/qdc9BY
      ▼ <ProgramCard>
        ▼ <li className="col-xs-12 col-sm-6"
          ▼ <div className="bc-panel"
            ▼ <Memo(ProgramHeader) na
              logo="http://bugcrowd."
              ▼ <div className="bc-p
                ▼ <Memo(LogoBackDrop
                  ► <LogoBackDrop u
                </Memo(LogoBackDrop
              ▼ <h4 className="bc-
                <a href="/bugcr
                </h4>
                ▼ <Memo(Badge) isTr
                  <Badge isTrial=
                </Memo(Badge)>
                <p className="bc-p
              </div>
            </Memo(ProgramHeader)>
```



```
▼ <div className="bc-panel"
  ▼ <Memo(ProgramHeader) na
    logo="http://bugcrowd."
    ▼ <div className="bc-p
      ▼ <Memo(LogoBackDrop
        ► <LogoBackDrop u
      </Memo(LogoBackDrop
    ▼ <h4 className="bc-
      <a href="/bugcr
    </h4>
    ▼ <Memo(Badge) isTr
      <Badge isTrial=
    </Memo(Badge)>
```



But how do I know when I need it?

But how do I know when I need it?



But how do I know when I need it?







# ESLint: react/display-name

## Prevent missing displayName in a React component definition (react/display-name)

DisplayName allows you to name your component. This name is used by React in debugging messages.

### Rule Details

The following patterns are considered warnings:

```
var Hello = createReactClass({
  render: function() {
    return <div>Hello {this.props.name}</div>;
  }
});
```

<https://github.com/yannickcr/eslint-plugin-react/blob/master/docs/rules/display-name.md>



Thank you!

