



# NETFLIX

Context **All\*** The Things

**how to talk about React, 101:**

**`"\${something about react}" – probably Dan  
Abramov or Ryan Florence`**

**“What Stephen said about **context**  
is *totally* valid” – probably Dan  
Abramov or Ryan Florence**

Let's talk about “**context**” in the  
**context** of “React **Context**.”

**But before we do that, let's set  
some context.**

**Welcome to Hulu.**

**Hulu** is a great place to work.

**I've been at Hulu for almost 2 years.**



**Hulu** likes context, not control.

**That makes zero sense. Sorta.  
Let's try it again. With CODE.**

**Welcome to**  
**context.getWhereStephenWorks()**

**context.getWhereStephenWorks()**  
**is a great place to work.**

**Boom. Those slides work  
everywhere for every talk ever.**



# NETFLIX

React Context

Here's a PNG...

# Context

[Edit on GitHub](#)

One of React's biggest strengths is that it's easy to track the flow of data through your React components. When you look at a component, you can easily see exactly which props are being passed in which makes your apps easy to reason about.

Occasionally, you want to pass data through the component tree without having to pass the props down manually at every level. React's "context" feature lets you do this.

## Note:

Context is an advanced and experimental feature. The API is likely to change in future releases.

Most applications will never need to use context. Especially if you are just getting started with React, you likely do not want to use context. Using context will make your code harder to understand because it makes the data flow less clear. It is similar to using global variables to pass state through your application.

**If you have to use context, use it sparingly.**

Regardless of whether you're building an application or a library, try to isolate your use of context to a small area and avoid using the context API directly when possible so that it's easier to upgrade when the API changes.

...of a website.

**“we don’t need no stinking warnings.” – probably Dan Abramov or Ryan Florence**



**“context is a React Way™ to help you say: It’s dangerous to go alone, take this” – probably Dan Abramov or Ryan Florence**

**Wizard : Sword : Link : Monster Adventure**

**Wizard : Pillow : Link : Pillow Fight**

**“You might already be using **context**.”**  
**– probably Dan Abramov**  
**or Ryan Florence**

# Contrived React Router Example:

```
import React from 'react';

class SecretInbox extends Component {
  componentWillMount(nextProps) {
    if (!nextProps.authorized) {
      // something changed with our authorization state, we need to abort!
      // let's programmatically redirect them to a login route:

      this.props.handleUnauthorizedAccess();
      console.warn('whew, almost leaked secret emails');
    }
  }

  render() {
    return {this.props.authorized &&
      <div className='secret-inbox react-div-soup'>
        For Your Authorized Eyes Only
      </div>
    };
  }
}
```

# Contrived React Router Example:

```
import React from 'react';

class SecretInbox extends Component {
  componentWillReceiveProps(nextProps) {
    if (!nextProps.authorized) {
      // something changed with our authorization state, we need to abort!
      // let's programmatically redirect them to a login route:

      this.props.handleUnauthorizedAccess();
      console.warn('whew, almost leaked secret emails');
    }
  }

  render() {
    return {this.props.authorized &&
      <div className='secret-inbox react-div-soup'>
        For Your Authorized Eyes Only
      </div>
    };
  }
}
```

**<Because {...props}>**

**<I {...props}>**

**<Just {...props}>**

**<Want {...props}>**

**<To {...props}>**

**<Stop {...props}>**

**<Passing {...props}>**

# Contrived React Router Example:

```
import React from 'react';

class SecretInbox extends Component {
  componentWillMount(nextProps) {
    if (!nextProps.authorized) {
      // something changed with our authorization state, we need to abort!
      // let's programmatically redirect them to a login route:

      this.context.router.push({ pathname: '/login' });
      console.warn('whew, almost leaked secret emails');
    }
  }

  render() {
    return <div className='secret-inbox react-div-soup'>
      For Your Authorized Eyes Only
    </div>;
  }
}
```



**<Provider>**

**<App>**

**<Router>**

**<Route>**

**<Container>**

**<Etc>**

**<SecretInbox />**

# parent/grandparent

- **childContextTypes**
  - defines what context is intended to be **supplied**
- **getChildContext()**
  - returns an object
  - keys are merged on to context (last merge wins)

# child

- **contextTypes**
  - defines what context is intended to be **consumed**
  - looks just like propTypes
- **this.context || (props, context)**
  - is an object
  - expects keys defined to be present and of same type




# NETFLIX

let's create-react-app

**(links / things to look at)**

**@netflixuie, @sprjrx**



Fin. Thanks!

**NETFLIX**