Advanced React

Nikhil Singhal

Dog Generator

weblab.to/dog-gen

(no need to follow along though)

`children` prop

Use case: you want to re-use some wrapper logic

```
<Router>
  <Feed path="/" userId={userId} />
  <Profile path="/profile/:userId" />
  <Chatbook path="/chat/" userId={userId} />
  <NotFound default />
</Router>
```

dog-generator branch: children

HOCs

Components that take components as props

dog-generator branch: hocs

Other use cases

- Debugging
- Data fetching
  - Wrapper component adds useEffect to fetch data and passes it as a prop
- Anything where you might use “inheritance” in another language

Custom Hooks

Literally just functions

dog-generator branch: timer

Other Built-In Hooks

useContext:	Less data passing
useCallback / useMemo:	Less unnecessary recomputation
useRef:	Keep mutable objects around / DOM manip
useReducer:	Simplify / Modularize logic
useDebugValue:	See values in debugging tools

Sad Reacts

Things to NOT do in React.js

```
onst onKeyDown = event => {
 if (event.keyCode === 13) {
   // Pressed enter
   // TODO: how do we get the text in the <input>?
 }      Shannen Wu, 2 years ago • a beaut
;

eturn (
 <div className="u-flex u-flexColumn u-flex-alignCenter">
   <label className="App-label" htmlFor="dog-breed">
     enter dog breed:
   </label>
   <input onKeyDown={onKeyDown} id="dog-breed" autoComplete="off" />
   <Card contents={<GoodBoiMeter breed={breed} />} />
```

Can we use document.getElementById()?

.... i mean technically yes it would work

Should we use document.getElementById()?

No!

- IDs are unique, so we can’t have two of this component
- This might mess up React shannenigans

```
const onKeyDown = event => {
  if (event.keyCode === 13) {
    // Pressed enter
    const breed = event.target.value;

    // update the state using the breed      You, seconds ago • Uncommi
  }
};

return (
  <div className="u-flex u-flexColumn u-flex-alignCenter">
    <label className="App-label" htmlFor="dog-breed">
      enter dog breed:
    </label>
    <input onKeyDown={onKeyDown} id="dog-breed" autoComplete="off" />
    <Card contents={<GoodBoiMeter breed={breed} />} />
    <Card contents={<DogViewer breed={breed} iteration={iteration} />}
    Count: {timer}
  </div>
);
```

Tells you what happened

```
const onKeyDown = event => {
  if (event.keyCode === 13) {
    // Pressed enter
    const breed = event.target.value;

    // update the state using the breed      You, seconds ago • Uncommi
  }
};

return (
  <div className="u-flex u-flexColumn u-flex-alignCenter">
    <label className="App-label" htmlFor="dog-breed">
      enter dog breed:
    </label>
    <input onKeyDown={onKeyDown} id="dog-breed" autoComplete="off" />
    <Card contents={<GoodBoiMeter breed={breed} />} />
    <Card contents={<DogViewer breed={breed} iteration={iteration} />}
    Count: {timer}
  </div>
);
```

Tells you what happened

Where did the event happen? <input>!

```
const onKeyDown = event => {
  if (event.keyCode === 13) {
    // Pressed enter
    const breed = event.target.value;

    // update the state using the breed      You, seconds ago • Uncommi
  }
};

return (
  <div className="u-flex u-flexColumn u-flex-alignCenter">
    <label className="App-label" htmlFor="dog-breed">
      enter dog breed:
    </label>
    <input onKeyDown={onKeyDown} id="dog-breed" autoComplete="off" />
    <Card contents={<GoodBoiMeter breed={breed} />} />
    <Card contents={<DogViewer breed={breed} iteration={iteration} />}
    Count: {timer}
  </div>
);
```

What if event.target isn’t the element we want?

What if event.target isn’t the element we want?

Controlled Components!

- Synchronize input values with React state
  - Value is always available to read
  - Update the state to modify the value

dog-generator branch: controlled

What if event.target isn't the element we want?

Refs!

- Keep a "ref" (reference) to the DOM node that corresponds to a particular component
  - Save the ref in a variable somewhere
  - Access the value in the input with `ref.current`

dog-generator branch: refs

React Developer Tools

weblab.to/chrome-dev-tools

-or-

weblab.to/firefox-dev-tools

demo

visualizing the component tree

debugging with devtools

where’s the bug???

performance!

rankings, render times, flame graphs, oh my!

some more sad reacts

plz do not

Copy props into state

plz do not

Use indices as keys

```
return (
  <div className="TodoList-container">
    <h1>{this.props.title}</h1>
    <ul>
      {this.state.todos.map((item, i) => (
        <ListItem
          key={`listItem-${i}`}
          content={item.todo}
          deleteTodo={() => this.deleteTodo(item.key)}
        />
      ))}
    </ul>
```

plz do not

Forget to prefix your class names with the component name

plz do not

Use .bind()

plz do not

Put more than one component in a file

Reading Error Messages

What's the error?

```
Uncaught ReferenceError: DogViewer is not defined                                App.js:43
    at App.render (App.js:43)
    at finishClassComponent (react-dom.development.js:18470)
    at updateClassComponent (react-dom.development.js:18423)
    at beginWork$1 (react-dom.development.js:20186)
    at HTMLUnknownElement.callCallback (react-dom.development.js:336)
    at Object.invokeGuardedCallbackDev (react-dom.development.js:385)
    at invokeGuardedCallback (react-dom.development.js:440)
    at beginWork$$1 (react-dom.development.js:25780)
    at performUnitOfWork (react-dom.development.js:24695)
    at workLoopSync (react-dom.development.js:24671)
▶The above error occurred in the <App> component:                               react-dom.development.js:21843
    in App

Consider adding an error boundary to your tree to customize error handling behavior.
Visit https://fb.me/react-error-boundaries to learn more about error boundaries.
Uncaught ReferenceError: DogViewer is not defined                                react-dom.development.js:25206
    at App.render (App.js:43)
    at finishClassComponent (react-dom.development.js:18470)
    at updateClassComponent (react-dom.development.js:18423)
    at beginWork$1 (react-dom.development.js:20186)
    at HTMLUnknownElement.callCallback (react-dom.development.js:336)
    at Object.invokeGuardedCallbackDev (react-dom.development.js:385)
    at invokeGuardedCallback (react-dom.development.js:440)
    at beginWork$$1 (react-dom.development.js:25780)
    at performUnitOfWork (react-dom.development.js:24695)
    at workLoopSync (react-dom.development.js:24671)
```

What's the error?

```
Uncaught ReferenceError: DogViewer is not defined                                App.js:43
    at App.render (App.js:43)
    at finishClassComponent (react-dom.development.js:18470)
    at updateClassComponent (react-dom.development.js:18423)
    at beginWork$1 (react-dom.development.js:20186)
    at HTMLUnknownElement.callCallback (react-dom.development.js:336)
    at Object.invokeGuardedCallbackDev (react-dom.development.js:385)
    at invokeGuardedCallback (react-dom.development.js:440)
    at beginWork$$1 (react-dom.development.js:25780)
    at performUnitOfWork (react-dom.development.js:24695)
    at workLoopSync (react-dom.development.js:24671)
▸The above error occurred in the <App> component:                  react-dom.development.js:21843
    in App

Consider adding an error boundary to your tree to customize error handling behavior.
Visit https://fb.me/react-error-boundaries to learn more about error boundaries.
Uncaught ReferenceError: DogViewer is not defined                  react-dom.development.js:25206
    at App.render (App.js:43)
    at finishClassComponent (react-dom.development.js:18470)
    at updateClassComponent (react-dom.development.js:18423)
    at beginWork$1 (react-dom.development.js:20186)
    at HTMLUnknownElement.callCallback (react-dom.development.js:336)
    at Object.invokeGuardedCallbackDev (react-dom.development.js:385)
    at invokeGuardedCallback (react-dom.development.js:440)
    at beginWork$$1 (react-dom.development.js:25780)
    at performUnitOfWork (react-dom.development.js:24695)
    at workLoopSync (react-dom.development.js:24671)
```

forgot to import at the top

```
Uncaught ReferenceError: DogViewer is not defined
    at App.render (App.js:43)
```

What's the error?

```
ERROR in ./client/src/components/App.js
Module build failed (from ./node_modules/babel-loader/lib/index.js):
SyntaxError: /Users/shannenwu/mit/webdev-staff/dog-generator/client/src/componen
ts/App.js: Adjacent JSX elements must be wrapped in an enclosing tag. Did you wa
nt a JSX fragment <>...</>? (42:8)

  40 |           enter dog breed:{" "}
  41 |         </label>
> 42 |         <input onKeyDown={this.onKeyDown} id="dog-breed" />
     |         ^
  43 |         <GoodBoiMeter breed={breed} />
  44 |         <DogViewer breed={breed} iteration={iteration} />
  45 |       // </div>
    at Object.raise (/Users/shannenwu/mit/webdev-staff/dog-generator/node_module
s/@babel/parser/lib/index.js:7012:17)
    at Object.jsxParseElementAt (/Users/shannenwu/mit/webdev-staff/dog-generator
/node_modules/@babel/parser/lib/index.js:4104:18)
    at Object.jsxParseElement (/Users/shannenwu/mit/webdev-staff/dog-generator/n
ode_modules/@babel/parser/lib/index.js:4114:17)
    at Object.parseExprAtom (/Users/shannenwu/mit/webdev-staff/dog-generator/nod
e_modules/@babel/parser/lib/index.js:4121:19)
    at Object.parseExprSubscripts (/Users/shannenwu/mit/webdev-staff/dog-generat
or/node_modules/@babel/parser/lib/index.js:9237:23)
    at Object.parseMaybeUnary (/Users/shannenwu/mit/webdev-staff/dog-generator/n
ode_modules/@babel/parser/lib/index.js:9217:21)
    at Object.parseExprOps (/Users/shannenwu/mit/webdev-staff/dog-generator/node
_modules/@babel/parser/lib/index.js:9083:23)
    at Object.parseMaybeConditional (/Users/shannenwu/mit/webdev-staff/dog-gener
ator/node_modules/@babel/parser/lib/index.js:9056:23)
    at Object.parseMaybeAssign (/Users/shannenwu/mit/webdev-staff/dog-generator/
node_modules/@babel/parser/lib/index.js:9015:21)
    at Object.parseParenAndDistinguishExpression (/Users/shannenwu/mit/webdev-st
aff/dog-generator/node_modules/@babel/parser/lib/index.js:9799:28)
 @ ./client/src/index.js 3:0-38 5:36-39
 @ multi @babel/polyfill ./client/src/index.js
i ｢wdm｣: Failed to compile.
```

forgot to return a single component in render

```
ERROR in ./client/src/components/App.js
Module build failed (from ./node_modules/babel-loader/lib/index.js):
SyntaxError: /Users/shannenwu/mit/webdev-staff/dog-generator/client/src/compon
ts/App.js: Adjacent JSX elements must be wrapped in an enclosing tag. Did you wa
nt a JSX fragment <>...</>? (42:8)

  40 |           enter dog breed:{" "}
  41 |         </label>
> 42 |         <input onKeyDown={this.onKeyDown} id="dog-breed" />
     |         ^
  43 |         <GoodBoiMeter breed={breed} />
  44 |         <DogViewer breed={breed} iteration={iteration} />
  45 |       // </div>
    at Object.raise (/Users/shannenwu/mit/webdev-staff/dog-generator/node_module
s/@babel/parser/lib/index.js:7012:17)
    at Object.jsxParseElementAt (/Users/shannenwu/mit/webdev-staff/dog-generator
/node_modules/@babel/parser/lib/index.js:4104:18)
    at Object.jsxParseElement (/Users/shannenwu/mit/webdev-staff/dog-generator/n
ode_modules/@babel/parser/lib/index.js:4114:17)
    at Object.parseExprAtom (/Users/shannenwu/mit/webdev-staff/dog-generator/nod
e_modules/@babel/parser/lib/index.js:4121:19)
    at Object.parseExprSubscripts (/Users/shannenwu/mit/webdev-staff/dog-generat
or/node_modules/@babel/parser/lib/index.js:9237:23)
    at Object.parseMaybeUnary (/Users/shannenwu/mit/webdev-staff/dog-generator/n
ode_modules/@babel/parser/lib/index.js:9217:21)
    at Object.parseExprOps (/Users/shannenwu/mit/webdev-staff/dog-generator/node
_modules/@babel/parser/lib/index.js:9083:23)
    at Object.parseMaybeConditional (/Users/shannenwu/mit/webdev-staff/dog-gener
ator/node_modules/@babel/parser/lib/index.js:9056:23)
    at Object.parseMaybeAssign (/Users/shannenwu/mit/webdev-staff/dog-generator/
node_modules/@babel/parser/lib/index.js:9015:21)
    at Object.parseParenAndDistinguishExpression (/Users/shannenwu/mit/webdev-st
aff/dog-generator/node_modules/@babel/parser/lib/index.js:9799:28)
 @ ./client/src/index.js 3:0-38 5:36-39
 @ multi @babel/polyfill ./client/src/index.js
i [wdm]: Failed to compile.
```

react attributes are sometimes named differently

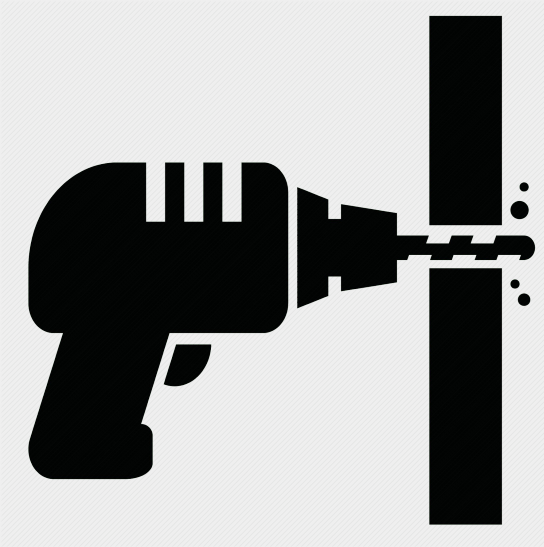
```
Warning: Invalid DOM property `for`. Did you mean `htmlFor`?
    in label (created by App)
    in div (created by App)
    in App
```

Context

motivation

avoid prop drilling!

example: `userId` in catbook



limitations

mucks up your beautiful component tree

can only use 1 context per component (...until you learn hooks!)

still can only pass down

read the docs before you use it

example: `userId`

- create `UserContext.js`
- import it in `App` and `CommentsBlock`
- add `Provider` in `App` with `value={this.state.userId}`
- set `contextType` on `CommentsBlock`
- consume as `this.context`
- now we can delete the `userId` prop!

Function Components

general form:

```
function Welcome(props) {
  return <h1>Hello, {props.name}</h1>;
}
```

equivalent!

```
class Welcome extends React.Component {
  render() {
    return <h1>Hello, {this.props.name}</h1>;
  }
}
```

```
function Welcome(props) {
  return <h1>Hello, {props.name}</h1>;
}
```

let’s convert dog generator!

GoodBoiMeter

ok, but what about state?

- Everything gets condensed into the `useState()` hook!

ok, but what's a hook?

tl;dr: special functions you can use in the function components!

Advantages:

- More direct/concise way to access the same parts of React
- No worrying about how classes work (or usage of `this`)
- Cleaner separation of side effects
- More modular — can write your own!

Read this before you use hooks yourself

App.js

rules of hooks

- Only call hooks at the top level
  - Never inside if statements, loops, etc.
- Only call hooks from React functions
  - Function components or custom hooks

ok, but what about lifecycle things?

- Everything can be done with the `useEffect()` hook!
  - `componentDidMount()` — `useEffect()` with empty dependencies
  - `componentDidUpdate()` — `useEffect()` with the right dependencies

- Requires a very different way of thinking about the React lifecycle

DogViewer

Other hooks

`useContext`

`useCallback` / `useMemo`

`useRef`

`useReducer`

`useDebugValue`

.... or write your own!