

React 入门

React 入门

React 入门

- [React 入门](#)
- [React 入门](#)
- [React 入门](#)
- [Context 入门](#)
- [CSS](#)
- [React 入门](#)
- [GraphQL 入门](#)
- [JSX 入门](#)
- [React 入门](#)
- [React-Router](#)
- [ReactDOM](#)
- [Redux 入门](#)
- [React 入门](#)
- [React 入门](#)
- [TypeScript 入门](#)
- [Webpack](#)
- [WebpackDevServer](#)

React

React 

- Elements API
- Components API
- Prototypes

React Elements

Elements API 

 React 

```
{
  type: /* ... */,
  key: /* ... */,
  ref: /* ... */,
  props: {
    // ...
    children: /* ... */,
    // ...
  }
}
```



React is “a” component



```
export default class App extends React.Component {  
  
  constructor(props) {  
    super(props);  
    this.state = {};  
  }  
  
  render() {  
  }  
}
```

React is HTML `<div>` `<h1>` `<p>`
`<MyTitle>`



```
const App = () => (  
  <Mui.RaisedButton label="Default" />  
)
```



- `this.props`
- `this.state`

`props` `state` `state`
`setState` `state`

setState

||||| setState ||||| state |||| render |||||||||||||

||||||||||||||||| this.state |||||||

```
this.setState({mykey: 'my new value'});
```

||||||||||||||||| this.state |

```
this.setState(function (previousState, currentProps) {  
  return {myInteger: previousState.myInteger + 1};  
});
```

|||||||||||||||||

this.setState ||||||| this.state

|||||||||||||||||

```
class Select extends React.Component {  
  constructor(props, context) {  
    super(props, context)  
    this.state = {selection: 0};  
  }  
  
  render() {  
    return (  
      <ul>  
        <li onClick={() => this.onSelect(1)}>1</li>  
        <li onClick={() => this.onSelect(2)}>2</li>  
        <li onClick={() => this.onSelect(3)}>3</li>  
      </ul>  
    )  
  }  
  
  onSelect(value) {  
    this.setState({selection: value});  
    console.log(this.state.selection);  
  }  
}
```

useState

```
this.setState(  
  {...},  
  () => console.log('set state!')  
)
```

false ☐ ☐ ☐

UI

defaultProps

props

```
class Account extends Component {
  static defaultProps = {
    email: '',
    gender: '',
  };

  constructor(props) {
    super(props);
  }

  render() {
    const { name, email, gender } = this.props;
    return (
      <div className="mod-account">
        <p>Name: {name}</p>
        <p>Email: {email}</p>
        <p>Gender: {gender}</p>
      </div>
    )
  }
}
```

constructor

ref

ref

DOM□□□□

```
render: function() {
  return <TextInput ref={(c) => this._input = c} />;
},
componentDidMount: function() {
  this._input.focus();
},
```

ref

DOM [illegible]

```
<input type="text"
      ref={
        (e) => e ? e.selectionStart = this.props.task.length : null
      }
      autoFocus={true}
      defaultValue={this.props.task}
      onBlur={this.finishEdit}
      onKeyDown={this.checkEnter} />;
```

ref

ref="input" ☐

```
this.refs.input
```

--	--	--

```
<Parent><Child /></Parent>
```

--	--	--	--	--	--

Parent ☐ ☐ ☐ ☐ ☐

Child

--	--	--	--	--

Parent ☐ ☐ ☐ ☐

```
this.props.children
```

key

--	--	--	--	--	--	--	--

```
render: function() {  
  var results = this.props.results;
```

```
return (  
  <ol>  
    {results.map(function(result) {  
      return <li key={result.id}>{result.text}</li>;  
    })}  
  </ol>  
);  
}
```

□□□□□□□□

li □ key □□□□□□□□□□□□

UI

--	--	--	--	--	--	--	--

UI[]

UI

--	--	--	--	--	--

UI

--	--	--	--	--	--	--

```
function NameBox(name) {
  return { fontWeight: 'bold', labelContent: name };
}
```

[illegible]

```
function FancyUserBox(user) {
  return {
    borderStyle: '1px solid blue',
    childContent: [
      'Name: ',
      NameBox(user.firstName + ' ' + user.lastName)
    ]
  };
}
```

composition $\square$ ☐ React

--	--	--	--	--	--	--	--

component□□

`element`

element

--	--	--	--	--	--	--	--

[illegible]

immutable

[illegible]

type ☐ props ☐

□□ type □□□□□□□□

element

--	--	--	--	--

DOM

--	--	--

type

--	--	--	--	--	--	--	--

props 

```
{
  type: 'button',
  props: {
    className: 'button button-blue',
    children: {
      type: 'b',
      props: {
        children: 'OK!'
      }
    }
  }
}
```



```
<button class='button button-blue'>
  <b>
    OK!
  </b>
</button>
```

children

type React

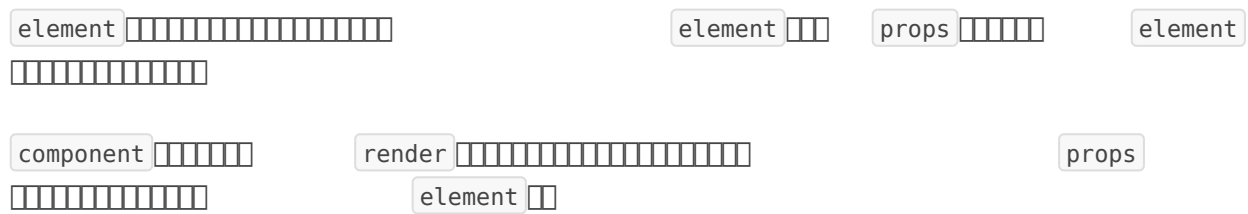
```
{
  type: Button,
  props: {
    color: 'blue',
    children: 'OK!'
  }
}
```

children

```
const DeleteAccount = () => ({
  type: 'div',
  props: {
    children: [{
      type: 'p',
      props: {
```

□□ JSX□□□□

element



- [React - Basic Theoretical Concepts](#), by sebmarkbage

Context

Context

```
var Button = React.createClass({
  render: function() {
    return (
      <button style={{background: this.props.color}}>
        {this.props.children}
      </button>
    );
  }
});

var Message = React.createClass({
  render: function() {
    return (
      <div>
        {this.props.text} <Button color={this.props.color}>Delete</Button>
      </div>
    );
  }
});

var MessageList = React.createClass({
  render: function() {
    var color = "purple";
    var children = this.props.messages.map(function(message) {
      return <Message text={message.text} color={color} />;
    });
    return <div>{children}</div>;
  }
});
```

color

context

```
graph LR; childContextTypes --> getChildContext; getChildContext --> React; React --> context; context --> context2[context];
```

Diagram illustrating the state of contextTypes and context after the first call to render. contextTypes is an array of 5 elements, and context is an object with 5 properties.

```

boolean shouldComponentUpdate(
  object nextProps, object nextState, object nextContext
)

void componentWillUpdate(
  object nextProps, object nextState, object nextContext
)

void componentDidUpdate(
  object prevProps, object prevState, object prevContext
)

```

□□□□□□□□□□

context □□□□

contextTypes □□□□□□□□□□

```

function Button(props, context) {
  return (
    <button style={{background: context.color}}>
      {props.children}
    </button>
  );
}

Button.contextTypes = {color: React.PropTypes.string};

```

□□ state □ props □□□□

getChildContext □□□□□□□□□□□□□□□□

context □□□

```

var MediaQuery = React.createClass({
  getInitialState: function(){
    return {type:'desktop'};
  },
  childContextTypes: {
    type: React.PropTypes.string
  },
  getChildContext: function() {
    return {type: this.state.type};
  },
  componentDidMount: function(){
    var checkMediaQuery = function(){
      var type = window.matchMedia("(min-width: 1025px)").matches ? 'desktop' : 'mobile';
      if (type !== this.state.type){

```

```
        this.setState({type:type});
    }
};

window.addEventListener('resize', checkMediaQuery);
checkMediaQuery();
},
render: function(){
    return this.props.children;
}
});
```

context 



 context  Modal 


context

CSS

```
const style = {color: red}  
<div style={style}></div>
```



```
var Message = React.createClass({
  getInitialState: function() {
    return { message: this.props.message };
  },

  _messageChange: function(e) {
    this.setState({ message: e.target.value });
  },

  render: function() {
    return (
      <div>
        <span>Message: {this.state.message}</span>
        <br />
        Message: <input type="text"
          value={this.state.message}
          onChange={this._messageChange} />
      </div>
    );
  },
});
```



onChange

e.target.value



GraphQL

GraphQL API
 GraphQL

[illegible]

REST



```
{
  "id": "c49bc1e1-26c1-49c5-b9c5-c89e24be0ac4",
  "url": "https://pics.example.com/dog",
  "caption": "My Dog",
  "author": {
    "id": "5a0cba48-c014-4cf0-b6fb-0bf7514b8165",
    "name": "Alice"
  }
}
```

GraphQL□□□□□□□□□□

```
node(id: "c49bc1e1-26c1-49c5-b9c5-c89e24be0ac4") {  
  id,  
  url,  
  caption,  
  author {  
    id,  
    name  
  }  
}
```

JSX



JSX HTML

```
<div dangerouslySetInnerHTML={{__html: 'cc &copy; 2015'}} />
```



JSX

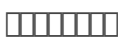
```
/* ... */
```



```
const App = <div> { /*  */ } </div>;
```



```
<Person
  /*
  */
  name={name}
/>
```



```
// 1. 直接传值
const component = <Component name={name} value={value} />;

// 2. 传对象
const component = <Component />;
component.props.name = name;
component.props.value = value;

// 3. 解构赋值
const data = { name: 'foo', value: 'bar' };
const component = <Component {...data} />;
```

React 16.8 之前

HTML 属性名

JavaScript

属性名

- `class` 属性名 `className`
- `for` 属性名 `htmlFor`

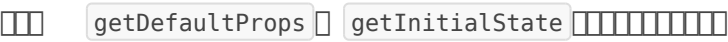
属性名

`data-` 属性名

```
<div data-attr="xxx">content</div>
```



- getDefaultProps
- getInitialState
- componentWillMount
- render
- componentDidMount



```
// 初始化 this.props
getDefaultProps: function(){
  return { /* something here */ };
}

// 初始化 this.state
getInitialState: function(){
  return { /* something here */ };
}
```



-
- componentWillUnmount



- getInitialState
- componentWillMount
- render
- componentDidMount

[illegible]

- `Props`
- `componentWillReceiveProps`
- `shouldComponentUpdate`
- `componentWillUpdate`
- `render`
- `componentDidUpdate`

--	--	--	--	--	--	--

state

1. `state`
2. `shouldComponentUpdate`
3. `componentWillUpdate`
4. `render`
5. `componentDidUpdate`

componentDidUpdate

```
componentDidUpdate
```

[illegible]

render

--	--	--	--	--	--	--

```
componentDidUpdate: function(prevProps, prevState){
  //
}
```

[illegible]

```
componentDidUpdate: function() {
  this.scrollToElement();
},
```

```
render: function() {  
  return [...]
```

```
this.scrollElement
```

[illegible]

```
this.scrollElement
```

DOMTTTTT

DOM

[illegible]

```
requestAnimationFrame
```

```
setTimeout
```

--	--	--	--	--	--	--	--

DOM

```
scrollElement: function() {  
    var this = this;
```

```

    setTimeout(function () {
      window.requestAnimationFrame(function() {
        var node = _this.getDOMNode();
        if (node !== undefined) {
          node.scrollTop = node.scrollHeight;
        }
      })
    }, 0)
  },
  componentDidMount: function() {
    this.scrollToElement();
  },

```

shouldComponentUpdate

```

    state: {
      shouldComponentUpdate: true,
      render: true
    },
    props: {
      true: true,
      false: false
    }
  }

```

```
shouldComponentUpdate: function(nextProps, nextState){
  // return a boolean value
  return true;
}
```

Color

```
getInitialState: function () {
  return {
    colors: new Immutable.List(this.props.colors)
  };
},

_addColor: function (newColor) {
  this.setState({
    colors = this.state.colors.push(newColor)
  });
},

render: function () {
```

```
return (
  <div>
    <ColorList colors={this.state.colors} />
    <ColorForm addColor={this._addColor} />
  </div>
);
}
```

```

graph LR
    subgraph Row1
        direction LR
        C1[Color] --- CL1[ColorList] --- TC1[this.state.colors]
    end
    subgraph Row2
        direction LR
        TC2[this.state.colors] --- SCU[shouldComponentUpdate]
    end

```

```
shouldComponentUpdate: function (nextProps, nextState) {
  return nextProps.colors !== this.props.colors;
}
```

Diagram illustrating the state of a component after a successful update:

- `shouldComponentUpdate` returns `true`.
- `nextState` is `state`.
- `nextProps` is `props`.
- `this.state` is `state`.
- `this.props` is `props`.
- `shouldComponentUpdate` is called with `state` and `props`.
- `this.setState` is called.

componentWillUpdate

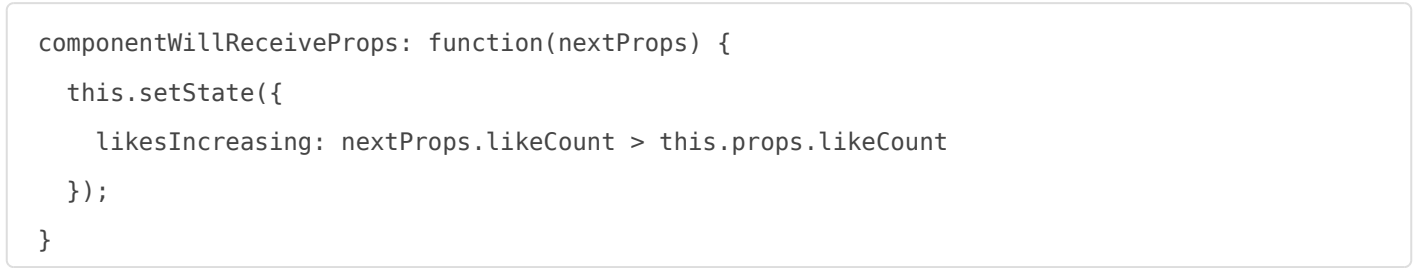
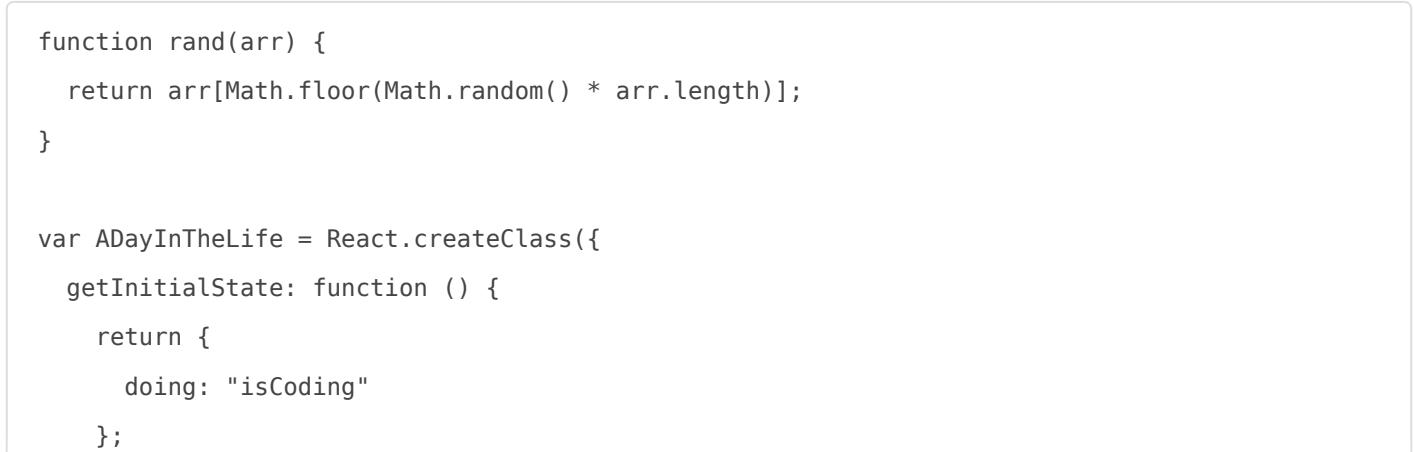
`shouldComponentUpdate` `true` `componentWillUpdate`

```
componentWillUpdate: function(nextProps, nextState){
  // perform any preparations for an upcoming update
}
```

`this.setState`

componentWillReceiveProps

```
graph LR; A[componentWillReceiveProps] --> B[render()];
```



```

},

handleButtonClick: function () {
  var activities = ["isCoding", "isEating", "isSleeping"];
  var randAct = rand(activities);
  this.setState({
    doing: randAct
  });
},

render: function () {
  return (
    <div>
      <button onClick={this.handleButtonClick}>Next Activity</button>
      <Jake activity={this.state.doing} />
    </div>
  );
}

});

React.render(<ADayInTheLife />, document.body);

```

□□□□□□□□

ADayInTheLife □□□□□

Jake □

□□□□□

Jake □□□□

```

var Jake = React.createClass({

  getDefaultProps: function () {
    return {
      activity: "nothingYet"
    };
  },

  getInitialState: function () {
    return {
      thinking: "nothing yet"
    };
  },

```

```

calcThinking: function (activity) {
  var thoughts = {
    isCoding: "yay, code",
    isEating: "yum, code",
    isSleeping: "where's the code?"
  };

  this.setState({
    thinking: thoughts[activity]
  })
},

componentWillReceiveProps: function (nextProps) {
  this.calcThinking(nextProps.activity);
},

render: function () {
  return <div>Jake: <b>{this.props.activity}</b> and thinking
"{this.state.thinking}".</div>;
}
});

```

Diagram illustrating the state of the component after the first render:

- State object: `{ calcThinking: true, Jake: true, state: 'idle' }`
- Props object: `{ componentWillReceiveProps: true, Jake: true, state: 'idle' }`
- The state object is passed to the component via the `state` prop.

componentWillMount

`componentWillMount`

`props` `state`

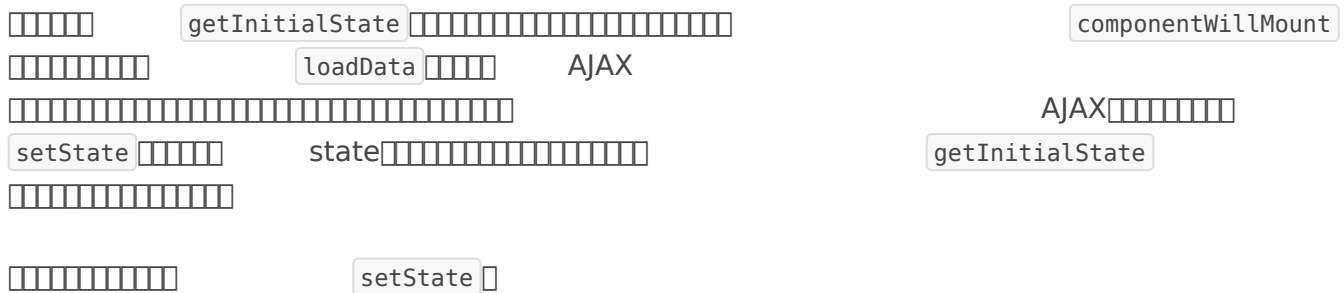
```
getInitialState: function() {
  return {
    items: this.props.initialItems || [],
    sort: this.props.config.sort || { column: "", order: "" },
    columns: this.props.config.columns
  };
},
componentWillMount: function() {
  this.loadData(this.props.dataSource);
}
```

```

},
loadData: function(dataSource) {
  if (!dataSource) return;

  $.get(dataSource).done(function(data) {
    console.log("Received data");
    this.setState({items: data});
  }.bind(this)).fail(function(error, a, b) {
    console.log("Error loading JSON");
  });
},

```



componentWillUnmount



React-Router



```
$ npm install react-router
```



```
import { Router, Route } from 'react-router';

React.render((
  <Router>
    <Route path="/" component={App}>
      <Route path="about" component={About}/>
      <Route path="users" component={Users}>
        <Route path="/user/:userId" component={User}/>
      </Route>
      <Route path="*" component={NoMatch}/>
    </Route>
  </Router>
), document.body);
```



```
// first we import some components
import { Router, Route, Link } from 'react-router';
// the histories are imported separately for smaller builds
import { history } from 'react-router/lib/HashHistory';
var App = React.createClass({
  render () {
    return (
      <div>
        <h1>App</h1>
        /* change the <a>s to <Links>s */
        <ul>
          <li><Link to="/about">About</Link></li>
          <li><Link to="/inbox">Inbox</Link></li>
        </ul>
      </div>
    );
  }
});
```

```
    {/*
      next we replace `` with `this.props.children`
      the router will figure out the children for us
    */}
    {this.props.children}
  </div>
)
}
});

// Finally we render a `Router` component with some `Route`s, it'll do all
// the fancy routing stuff for us.
React.render((
  <Router history={history}>
    <Route path="/" component={App}>
      <Route path="about" component={About}/>
      <Route path="inbox" component={Inbox}/>
    </Route>
  </Router>
), document.body);
```

ReactDOM

ReactDOM □□□□□□

- findDOMNode
- unmountComponentAtNode
- render

findDOMNode()

findDOMNode □□□□□□□□ DOM □□

```
componentDidMount() {  
  // this □□□□□□□□  
  const dom = ReactDOM.findDOMNode(this);  
}
```

□□□□□□□□□□□□□□□□

componentDidMount □ componentDidUpdate □□□□□

render()

render □□□□□□□□□□ DOM □□□

```
ReactDOMComponent render(  
  ReactElement element,  
  DOMElement container,  
  [function callback]  
)
```

□□□□□□□□□□

DOM □□□□□□□□□□□□□□□□□□□□

render □□□

null

□□□□□□□□□□□□□□□□

Redux



Reducer

[illegible]

```
function exampleReducer(state, action) {
  return state.changedBasedOn(action)
}
```



React.createElement

React.createElement [type] [props] [children] React [type] [props] [children] React

```
React.createElement(Layout, {
  title: 'React Static Site',
  children: 'Hello World'
});
```

ReactDOMServer.renderToString

react-dom/server [options] renderToString [element] React [element] HTML

[options] ReactDOM.render() [element] DOM [element] React

[options] DOM [element]

ReactDOMServer.renderToStaticMarkup

ReactDOMServer.renderToStaticMarkup [element] renderToString [element] React

[options] data-react-id [element]

React

React

- (property) state
-

A B

- Shallow rendering
- Regular test utilities and jsdom when needed
- mocha + jsdom + expect + expect-jsx + babel-plugin-rewire

DOM

mocha-jsdom describe jsdom() Node.js
DOM API window document history

jsdom useEach Mocha beforeEach/afterEach before/after
false

react-addons-test-utils React renderIntoDocument
React Document DOM
window window.document
window.document.createElement React React
DOM setState

react-dom findDOMNode React DOM
DOM
ref DOM
findDOMNode render null false findDOMNode null

findDOMNode() DOM

```
import React from 'react';  
import {findDOMNode} from 'react-dom';  
import {renderIntoDocument} from 'react-addons-test-utils';
```

```

import jsdom from 'mocha-jsdom';
import expect from 'expect';

class Label extends React.Component {
  render() {
    return <span>Hello {this.props.name}</span>;
  }
}

class Button extends React.Component {
  render() {
    return <div><Label name={this.props.name} /></div>;
  }
}

describe('Real dom test', () => {
  jsdom({useEach: true});

  it('works', () => {
    let component = renderIntoDocument(<Button name="John" />);
    let DOMNode = findDOMNode(component);
    expect(
      DOMNode.querySelector('span')
        .textContent
    ).toEqual('Hello John');
  });
});

```



```

import React from 'react';
import {
  renderIntoDocument,
  findRenderedDOMComponentWithTag
} from 'react-addons-test-utils';
import jsdom from 'mocha-jsdom';
import expect from 'expect';

class Label extends React.Component {
  render() {

```

```

    return <span>Hello {this.props.name}</span>;
  }
}

class Button extends React.Component {
  render() {
    return <div><Label name={this.props.name} /></div>;
  }
}

describe('Real Test Utilities', () => {
  jsdom({useEach: true});

  it('works', () => {
    let component = renderIntoDocument(<Button name="John" />);
    expect(
      findRenderedDOMComponentWithTag(component, 'span')
        .textContent
    ).toEqual('Hello John');
  });
});

```



```

import React from 'react';
import expect from 'expect';
import {createRenderer} from 'react-addons-test-utils';

class Label extends React.Component {
  render() {
    return <span>Hello {this.props.name}</span>;
  }
}

class Button extends React.Component {
  render() {
    return <div><Label name={this.props.name} /></div>;
  }
}

```

```
describe('Shallow rendering', () => {
  it('works', () => {
    let renderer = createRenderer();
    renderer.render(<Button name="John" />);
    let actualElement = renderer.getRenderOutput();
    let expectedElement = <div><Label name="John" /></div>;
    expect(actualElement).toEqual(expectedElement);
  });
});
```



```
import React from 'react';
import expect from 'expect';
import {createRenderer} from 'react-addons-test-utils';

class Label extends React.Component {
  render() {
    return <span>Hello {this.props.name}</span>;
  }
}

class Button extends React.Component {
  render() {
    return <div onClick={this.props.click}><Label name={this.props.name} /></div>;
  }
}

describe('Shallow rendering on* handlers', () => {
  it('works', () => {
    let renderer = createRenderer();
    let hasClicked = false;
    let click = () => hasClicked = true;
    renderer.render(<Button name="John" click={click} />);
    renderer.getRenderOutput().props.onClick();
    expect(hasClicked).toBe(true);
  });
});
```

```
});  
});
```



- Vincent Voyer, [Our journey through React component unit testing](#)
- Marcin Grzywaczewski, [Approaches to testing React components - an overview](#)

TypeScript



```
$ npm install --global typescript
```

TypeScript

.ts tsc

TS

JavaScript

```
# example.js  
$ tsc example.ts
```



TypeScript 9

- Boolean
- Number
- String
- Array
- Tuple
- Enum
- Any
- Void
- Function



```
//  
const lie : boolean = false;  
const pi : number = 3.14159;  
const tree_of_knowledge : string = "Yggdrasil";  
  
//  
const divine_lovers : string[] = ["Zeus", "Aphrodite"];  
const digits : Array<number> = [143219876, 112347890];
```

Tuple

[(Item Type), . . . , (Item Type)]

```
let date_triplet : [number, number, number];
date_triplet = [31, 6, 2016];

let athena : [string, number];
athena = ['Athena', 9386];
var name : string = athena[0];
const age : number = athena[1];
```

Enums

Enum

```
enum Color { Red, Green, Blue };
const red : Color = Color.Red;
console.log(Color[0]); // 'Red'

// TypeScript
enum RomanceLanguages { Spanish = 1, French, Italian, Romanian, Portuguese };
console.log(RomanceLanguages[4]); // 'Romanian'
console.log(RomanceLanguages[0]); // undefined
```

Any

```
let mystery : any = 4; // number
mystery = "four"; // string -- no error

const not_only_strings : any[] = [];
not_only_strings.push("This Works!")
not_only_strings.push(42); // This does too.
```

Void

undefined null

```
let the_void : void = undefined;
the_void = null;

the_void = "nothing"; // Error
```

Function

```
function capitalizeName (name : string) : string {
    return name.toUpperCase();
}

let multiply : (first : number, second : number) => number;
multiply = function (first, second) {
    return first * second;
}

let multiplyFirst : ( first : number) => ((second : number) => number);
multiplyFirst = function (first) {
    return function (num) {
        return first * num;
    }
}
```

Class

TypeScript#####

```
class ListComponent {
    constructor () {
        this.things = [];
    }
}
```

TypeScript##### `this` #####

```
class ListComponent {
    things : Array<ListItem>; // Or -- things : ListItem[];
    constructor () {
        this.things = [];
    }
}
```

`public` `private`

```
class ListComponent {
    public things : Array<ListItem>; // Or -- things : ListItem[];
```

```

    constructor () {
        this.things = [];
    }
}

class Person {
    constructor (
        public first_name : string,
        public last_name : last_name
    ) { }
}

```

// 编译

```

class Person {
    public first_name : string;
    public last_name : string;
    constructor (
        first_name : string,
        last_name : last_name
    ) { }
}

```

private 私有

```

class ListComponent {
    private _things : Array<ListItem>;
    constructor () {
        this._things = [];
    }
    get things () : Array<ListItem> { return this._things; }
}

```

TypeScript 编译 ES6

```

class CompletedListItem extends ListItem {
    completed : boolean;
    constructor (name : string) {
        super(name);
        this.completed = true;
    }
}

```

```
}  
}
```

Interface Class

```
interface User {  
    name : string;  
    email : string;  
}  
  
class RegisteredUser implements User {  
    constructor (  
        public name : string,  
        public email : string  
    ) { }  
}
```

```
interface User {  
    name : string;  
    email : string;  
    avatar? : Object;  
}  
  
// 属性  
interface User {  
    // PROPERTIES  
    name : string;  
    email : string;  
    avatar? : Object;  
  
    // API  
    print () : void;  
}
```

- Peleke Sengstacke, [From JavaScript to TypeScript, Pt. IIA: Using Classes, Interfaces, & Mixins](#)

Webpack

Webpack bundler

Webpack

- chunk
-



Webpack

```
$ webpack ./src/App.js ./build/bundle.js
```

 App.js bundle.js

 App.js

```
let $ = require('jquery');
$("p").css({ color: 'red' });
```

 bundle.js App.js jquery



--watch -w

```
$ webpack -w js/profile.js dist/bundle.js
$ webpack --watch
```

--devtool Source map

```
$ webpack -w --devtool source-map js/profile.js dist/bundle.js
```

--config

```
$ webpack --config ./path/to/webpack.dev.js
```



webpack.config.js

Webpack

webpack.config.js

CommonJS

```
module.exports = {  
  entry: '...',  
  output: '...',  
  // ...  
};
```

Webpack

entry

entry

```
entry: 'app.js',  
  
//   
entry: [  
  'react-hot-loader/patch',  
  'webpack-dev-server/client?http://localhost:8080',  
  'webpack/hot/only-dev-server',  
  './src/client.jsx'  
],
```

entry

output

```
module.exports = {  
  entry: {  
    desktop: './src/desktop.js',  
    mobile: './src/mobile.js'  
  },  
  output: {
```

```
    path: './dist',
    filename: '[name].bundle.js',
    // 打包 bundle.js 文件
    chunkFilename: '[id].common.js'
  }
};
```

output 配置

output 配置

```
output: {
  // 输出文件的路径
  path: './dist',
  // 输出文件名
  filename: 'bundle.js',
  // 输出文件的路径
  // 输出文件的名称
  publicPath: '/',
  // 输出文件的名称
  libraryTarget: "var",
  // 输出文件的名称
  library: "Foo"
},
```

Webpack 配置

47a0cc1b840cb310842cb85fb5b6116c.js

```
module.exports = {
  entry: './src/main.js',
  output: {
    path: './dist',
    filename: '[hash].js'
  }
};
```

externals 配置

externals 配置

```
// 通过externals来使用 jquery 模块
externals: {
  "jquery": "jQuery"
},
```

module 配置

module 配置

Babel 配置

module.rules

```
module: {
  loaders: [
    { test: /\.css$/, loader: 'style!css' }
  ]
},
```

resolve 配置

resolve 配置

1 resolve.modules

resolve.modules 配置

```
resolve: {
  modules: ['node_modules', 'src']
}
```

resolve.modules 配置

src/dir1/foo.js

```
// src/dir1/foo.js
import Bar from '../dir2/bar';
```

src/dir2/bar.js resolve.modules 配置

```
import Bar from 'dir2/bar';
```

2 resolve.extensions

resolve.extensions

Webpack

```
resolve: {
  extensions: ['.js', '.jsx']
},
```

3 resolve.mainFields

package.json

main

Node

package.json

- browser
- module ES6 CommonJS main
- main index.js

d3 package.json

```
{
  ...
  main: 'build/d3.Node.js',
  browser: 'build/d3.js',
  module: 'index',
  ...
}
```

resolve.mainFields

Webpack

Webpack

target

webworker

web

resolve.mainFields

mainFields: ["browser", "module", "main"]

Webpack

package.json

browser

module

module

main

target

node

resolve.mainFields

mainFields: ["module", "main"]

plugins

plugins

require()

require

```
require(['animals'], function(Animal) {
    var bear = new Animal('bear');
});
```

require.ensure()

[illegible][illegible][illegible]

```
require.ensure
```

```
require('./a');

if (condition) {
  require.ensure([], function(require){
    require('./b');
    console.log('done!');
  });
}
```

--	--	--	--	--	--

a.js

b.js

bundle.js

[illegible]

bundle.js

bundel.js

condition

--	--	--	--	--

0.bundle.js

```
require.ensure
```

[illegible]

0.bundle.js

[illegible]

require

require

0.bundle.js

```
require.ensure
```

```
require.ensure
```

```
require.ensure([], function(require){
  require('./a');
}, 'foo');

require.ensure(['foo'], function(require){
  require('./b');
});
```

```
require.ensure(['./b.js'], function(require) {
  require('./c.js');
});
```

Loader

loader□

```
module.exports = {
  module: {
    loaders: [
      { test: /\.css$/, loader: 'style!css' }
    ]
  }
};
```

■■■■■■■■

```
{ test: /\.js$/,
  exclude: /(node_modules|bower_components)/,
  loader: 'babel-loader',
  query: {
    presets: ['es2015']
  }
},
```

■■■■■■■■

.css ■■■■■■

```
require('./css/style.css');
```

■■■■ preloader

```
preLoaders: [
  { test: /\.js?$/, exclude: /node_modules/, loader: 'eslint-loader', include: __dirname +
    '/' }
],
```

Plugin

■■■■■■■■

■■■

```
plugins: [
  new webpack.optimize.UglifyJsPlugin({
    compress: {
      warnings: false
```

```
module.exports = {
  . . . . .,
  module: {
    loaders: [
      // for less files
      { test: /\.less$/,
        exclude: /(node_modules|bower_components)/,
        loader: ExtractTextPlugin.extract("style-loader", "css-loader", "less-loader")
      }
    ]
  },
  plugins: [
    new ExtractTextPlugin("style.css")
  ]
}
```

Node API

Webpack  Node  Webpack



Webpack

```
var webpack = require('webpack');
var WebpackDevServer = require('webpack-dev-server');
var config = require('./webpack.config');

var server = new WebpackDevServer(webpack(config), {
  contentBase: './dev',
  publicPath: config.output.publicPath,
  hot: true
});

server.listen(3000, 'localhost', function (err, result) {
```

```

    if (err) {
      console.log(err);
    }
    console.log('Listening at localhost:3000');
  });
};

```

```

// require the webpack Node.js library
var webpack = require('webpack');

webpack({
  // The first argument is your webpack config
  entry: './src/entry.js',
  output: {
    path: './dist',
    filename: 'bundle.js'
  }
}, function(err, stats) {
  // The second argument is a callback function that returns
  // more information about your bundle when it is complete
});

```

□□□□□

```

var webpack = require('webpack');

// Create an instance of the compiler
var compiler = webpack({ /* webpack config */ });

// Run the compiler manually
compiler.run(function(err, stats) { });

```

watch □□□□□□□□□□□□□□

```

compiler.watch(/* watchDelay */ 200, function(err, stats) { });

```

□□□□

- [Module Bundling with Webpack: Getting Started Guide](#) by Chimeremeze Ukah

WebpackDevServer

WebpackDevServer 是什么

Webpack 是什么

```
var server = new WebpackDevServer(webpack(config), {
  contentBase: './dev',
  publicPath: config.output.publicPath,
  hot: true
});

server.listen(3000, 'localhost', function (err, result) {
  if (err) {
    console.log(err);
  }
  console.log('Listening at localhost:3000');
});
```

- contentBase 是 HTTP 服务的根目录
- publicPath 是打包后文件的输出路径
publicPath: "/assets/" 文件: "bundle.js" 打包后: bundle.js 输出到 /assets/bundle.js



webpack.config.js 配置

devServer 配置

Dev server 是什么

- hot 热更新
- contentBase 静态资源根目录
- publicPath 打包后文件的输出路径
- historyApiFallback 404 页面重定向

webpack.config.js

```
devServer: {
  contentBase: './dist',
  // 使用 hot-loader
  hot: true,
```

```
// 开启BrowserHistory
historyApiFallback: true
},
```

Dev server 启动命令

```
$ webpack-dev-server --inline --hot --content-base ./path/to/index.html
```