

React JS

Notes for Professionals

Chapter 3: Using ReactJS with TypeScript

Section 3.1: ReactJS component written in TypeScript

Actually you can use ReactJS components in TypeScript as in Facebook's example. Just replace to tsx:

```
//HelloMessage.tsx
var HelloMessage = React.createClass({
  render: function() {
    return <div>Hello {this.props.name}</div>;
  }
});
ReactDOM.render(<HelloMessage name="John" />, mountNode);
```

But in order to make full use of TypeScript's main feature (static type checking) should be to tsx:

1) convert React.createClass example to ES6 Class:

```
//HelloMessage.tsx
class HelloMessage extends React.Component {
  render() {
    return <div>Hello {this.props.name}</div>;
  }
}
ReactDOM.render(<HelloMessage name="John" />, mountNode);
```

2) next add Props and State interfaces:

```
interface HelloMessageProps {
  name: string;
}
interface HelloMessageState {
  //empty in our case
}
class HelloMessage extends React.Component<HelloMessageProps, HelloMessageState> {
  constructor(props: HelloMessageProps) {
    super(props);
  }
  render() {
    return <div>Hello {this.props.name}</div>;
  }
}
ReactDOM.render(<HelloMessage name="John" />, mountNode);
```

Section 3.2: Installation and Setup

To use typescript with react in a node project, you must first have a project directory initialized with npm:

install the directory with `npm init`

Installing via `npm` or `yarn`

Chapter 10: React Routing

Section 10.1: Example Routes.js file, followed by use of Router Link in component

Place a file like the following in your top level directory. It defines which components to render for which paths.

```
import React from 'react';
import { Route, IndexRoute } from 'react-router';
import New from './containers/new-guest';
import Show from './containers/show';

import Index from './containers/home';
import App from './components/app';

export default (
  <Route path="/" component={App}>
    <IndexRoute component={Index} />
    <Route path="/posts/new" component={New} />
    <Route path="/posts/:id" component={Show} />
  </Route>
);
```

Now in your top level `index.js` that is your entry point to the app, you need only render this Router component like so:

```
import React from 'react';
import ReactDOM from 'react-dom';
import { Router, browserHistory } from 'react-router';
// import the routes component we created in routes.js
import routes from './routes';
```

```
// entry point
ReactDOM.render(
  <Router history={browserHistory} routes={routes} />
  document.getElementById('main')
);
```

Now it is simply a matter of using `Link` instead of `<a>` tags throughout your application. Using `Link` will communicate with React Router to change the React Router route to the specified link, which will in turn render the correct component as defined in `routes.js`.

```
import React from 'react';
import { Link } from 'react-router';

export default function PostButton(props) {
  return (
    <Link to={`/posts/${props.postId}`} />
    <div className="post-button">
      {props.title}
      <span> {props.tags}</span>
    </div>
  );
}
```

React JS Notes for Professionals

Chapter 14: React AJAX call

Section 14.1: HTTP GET request

Sometimes a component needs to render some data from a remote endpoint (e.g. a REST API). A standard practice is to make such calls in `componentDidMount` method.

Here is an example, using `superagent` as AJAX helper:

```
import React from 'react';
import request from 'superagent';

class App extends React.Component {
  constructor() {
    super();
    this.state = {};
  }
  componentDidMount() {
    request
      .get('/posts')
      .query({ range: '1..5' })
      .query({ order: 'desc' })
      .set('accept', 'application/json')
      .end((err, resp) => {
        if (!err) {
          this.setState({ metadata: resp.text });
        }
      });
  }
  render() {
    return (
      <div>{this.state.metadata} { `waiting for response...`} </div>
    );
  }
}
```

```
ReactDOM.render((App />, document.getElementById('root'));
```

A request can be initiated by invoking the appropriate method on the request object, then calling `.end()` to send the request. Setting header fields is simple, invoke `.set()` with a field name and value.

The `query()` method accepts objects, which when used with the GET method will form a query-string. The following will produce the path `/search?query=Manys&range=1..5&order=desc`.

```
POST requests
request.post('/posts')
  .set('Content-Type', 'application/json')
  .send({ name: 'js', 'age': '20' });
end(callback);
```

See [Superagent docs](#) for more details.

React JS Notes for Professionals

100+ pages
of professional hints and tricks

Contents

About	1
Chapter 1: Getting started with React	2
Section 1.1: What is ReactJS?	2
Section 1.2: Installation or Setup	3
Section 1.3: Hello World with Stateless Functions	4
Section 1.4: Absolute Basics of Creating Reusable Components	5
Section 1.5: Create React App	6
Section 1.6: Hello World	7
Section 1.7: Hello World Component	8
Chapter 2: Components	11
Section 2.1: Creating Components	11
Section 2.2: Basic Component	13
Section 2.3: Nesting Components	14
Section 2.4: Props	16
Section 2.5: Component states - Dynamic user-interface	17
Section 2.6: Variations of Stateless Functional Components	19
Section 2.7: setState pitfalls	20
Chapter 3: Using ReactJS with TypeScript	22
Section 3.1: ReactJS component written in TypeScript	22
Section 3.2: Installation and Setup	22
Section 3.3: Stateless React Components in TypeScript	23
Section 3.4: Stateless and property-less Components	24
Chapter 4: State in React	25
Section 4.1: Basic State	25
Section 4.2: Common Antipattern	25
Section 4.3: setState()	26
Section 4.4: State, Events And Managed Controls	28
Chapter 5: Props in React	30
Section 5.1: Introduction	30
Section 5.2: Default props	30
Section 5.3: PropTypes	31
Section 5.4: Passing down props using spread operator	32
Section 5.5: Props.children and component composition	33
Section 5.6: Detecting the type of Children components	34
Chapter 6: React Component Lifecycle	35
Section 6.1: Component Creation	35
Section 6.2: Component Removal	37
Section 6.3: Component Update	38
Section 6.4: Lifecycle method call in different states	39
Section 6.5: React Component Container	40
Chapter 7: Forms and User Input	42
Section 7.1: Controlled Components	42
Section 7.2: Uncontrolled Components	42
Chapter 8: React Boilerplate [React + Babel + Webpack]	44
Section 8.1: react-starter project	44
Section 8.2: Setting up the project	45

Chapter 9: Using ReactJS with jQuery	48
Section 9.1: ReactJS with jQuery	48
Chapter 10: React Routing	50
Section 10.1: Example Routes.js file, followed by use of Router Link in component	50
Section 10.2: React Routing Async	51
Chapter 11: Communicate Between Components	52
Section 11.1: Communication between Stateless Functional Components	52
Chapter 12: How to setup a basic webpack, react and babel environment	54
Section 12.1: How to build a pipeline for a customized "Hello world" with images	54
Chapter 13: React.createClass vs extends React.Component	58
Section 13.1: Create React Component	58
Section 13.2: "this" Context	58
Section 13.3: Declare Default Props and PropTypes	60
Section 13.4: Mixins	62
Section 13.5: Set Initial State	62
Section 13.6: ES6/React "this" keyword with ajax to get data from server	63
Chapter 14: React AJAX call	65
Section 14.1: HTTP GET request	65
Section 14.2: HTTP GET request and looping through data	66
Section 14.3: Ajax in React without a third party library - a.k.a with VanillaJS	66
Chapter 15: Communication Between Components	68
Section 15.1: Child to Parent Components	68
Section 15.2: Not-related Components	68
Section 15.3: Parent to Child Components	69
Chapter 16: Stateless Functional Components	71
Section 16.1: Stateless Functional Component	71
Chapter 17: Performance	74
Section 17.1: Performance measurement with ReactJS	74
Section 17.2: React's diff algorithm	74
Section 17.3: The Basics - HTML DOM vs Virtual DOM	75
Section 17.4: Tips & Tricks	76
Chapter 18: Introduction to Server-Side Rendering	77
Section 18.1: Rendering components	77
Chapter 19: Setting Up React Environment	78
Section 19.1: Simple React Component	78
Section 19.2: Install all dependencies	78
Section 19.3: Configure webpack	78
Section 19.4: Configure babel	79
Section 19.5: HTML file to use react component	79
Section 19.6: Transpile and bundle your component	79
Chapter 20: Using React with Flow	80
Section 20.1: Using Flow to check prop types of stateless functional components	80
Section 20.2: Using Flow to check prop types	80
Chapter 21: JSX	81
Section 21.1: Props in JSX	81
Section 21.2: Children in JSX	82
Chapter 22: React Forms	85
Section 22.1: Controlled Components	85

Chapter 23: User interface solutions	87
Section 23.1: Basic Pane	87
Section 23.2: Panel	87
Section 23.3: Tab	88
Section 23.4: PanelGroup	88
Section 23.5: Example view with `PanelGroup`s	89
Chapter 24: Using ReactJS in Flux way	91
Section 24.1: Data Flow	91
Chapter 25: React, Webpack & TypeScript installation	92
Section 25.1: webpack.config.js	92
Section 25.2: tsconfig.json	92
Section 25.3: My First Component	93
Chapter 26: How and why to use keys in React	94
Section 26.1: Basic Example	94
Chapter 27: Keys in react	95
Section 27.1: Using the id of an element	95
Section 27.2: Using the array index	95
Chapter 28: Higher Order Components	97
Section 28.1: Higher Order Component that checks for authentication	97
Section 28.2: Simple Higher Order Component	98
Chapter 29: React with Redux	99
Section 29.1: Using Connect	99
Appendix A: Installation	100
Section A.1: Simple setup	100
Section A.2: Using webpack-dev-server	101
Appendix B: React Tools	103
Section B.1: Links	103
Credits	104
You may also like	106

About

Please feel free to share this PDF with anyone for free,
latest version of this book can be downloaded from:

<https://goalkicker.com/ReactJSBook>

This *React JS Notes for Professionals* book is compiled from [Stack Overflow Documentation](#), the content is written by the beautiful people at Stack Overflow. Text content is released under Creative Commons BY-SA, see credits at the end of this book whom contributed to the various chapters. Images may be copyright of their respective owners unless otherwise specified

This is an unofficial free book created for educational purposes and is not affiliated with official React JS group(s) or company(s) nor Stack Overflow. All trademarks and registered trademarks are the property of their respective company owners

The information presented in this book is not guaranteed to be correct nor accurate, use at your own risk

Please send feedback and corrections to web@petercv.com

[Click here to download full PDF material](#)