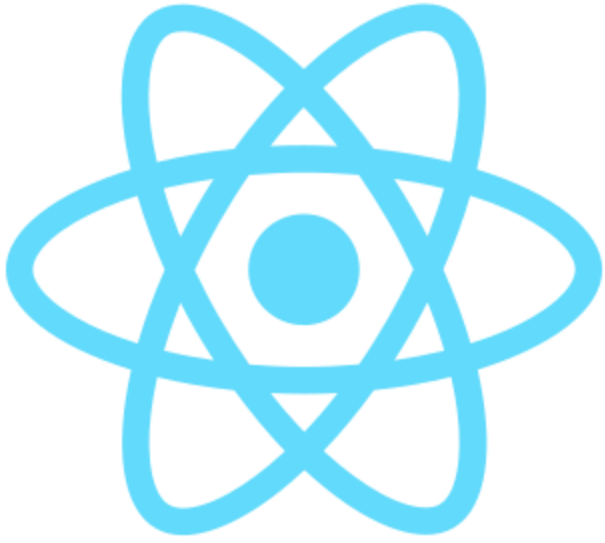


# Introduction to ReactJS


## Δημήτρης Σωτηρίου

Τεχνολογία Λογισμικού, Εθνικό Μετσόβιο Πολυτεχνείο


27 Φεβρουαρίου 2018

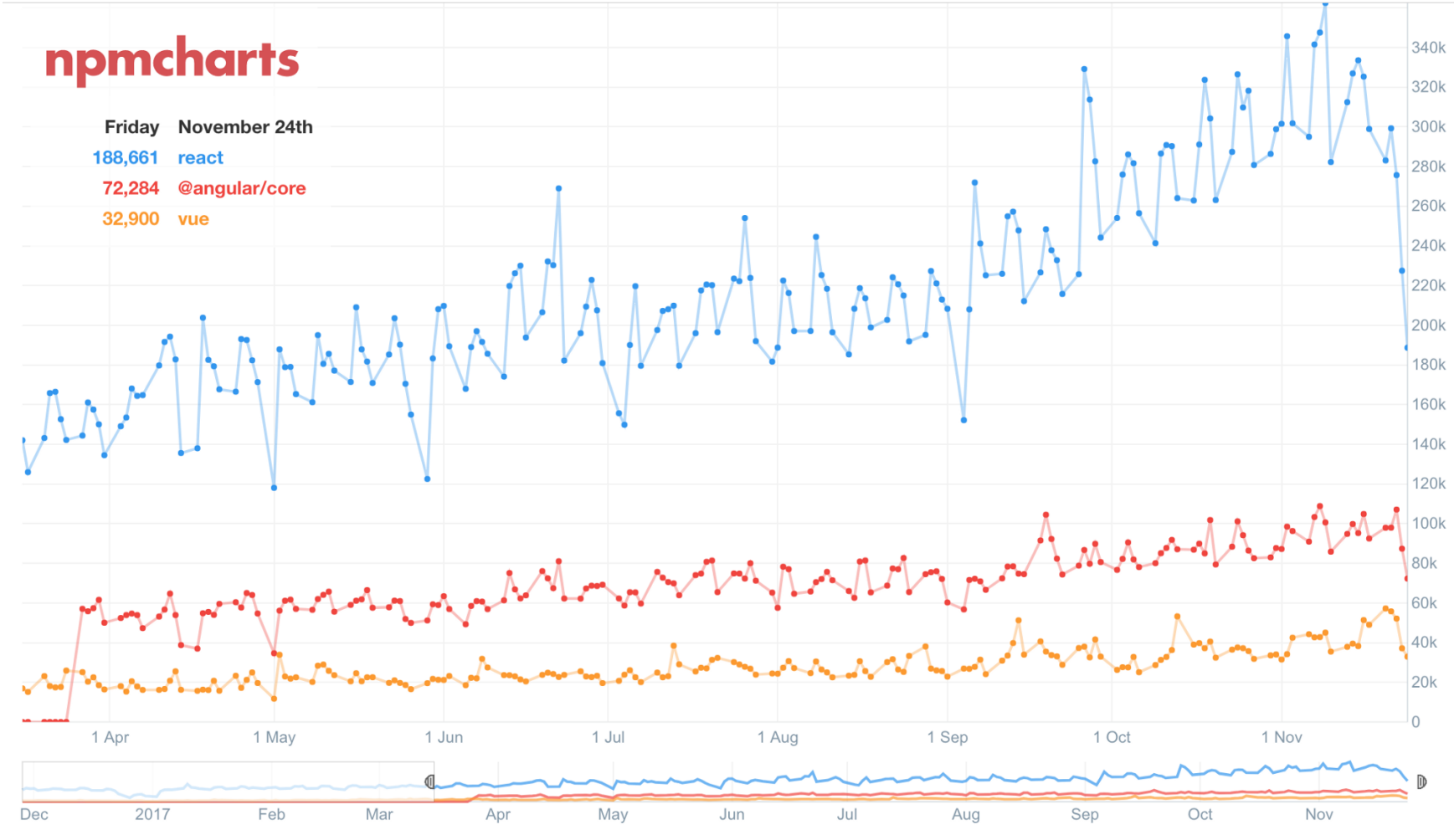
# History

Started 4 years ago

Created by Jordan Walke, inspired by XHP

Mainly maintained by Facebook, Instagram

Gained quickly in popularity


npmcharts
Friday November 24th
188,661 react
72,284 @angular/core
32,900 vue
340k
320k
300k
280k
260k
240k
220k
200k
180k
160k
140k
120k
100k
80k
60k
40k
20k
0
1 Apr
1 May
1 Jun
1 Jul
1 Aug
1 Sep
1 Oct
1 Nov
Dec
2017
Feb
Mar
Apr
May
Jun
Jul
Aug
Sep
Oct
Nov

# Features

Suited for single page applications

Speed, simplicity, scalability

View in MVC pattern

Separations of concerns with "components"

Resusability

JSX syntax

# Features (continued)

## One-way data flow

Immutable values passed to components

Single source of truth

Redux library

## Virtual DOM

In-memory data structure representing the DOM

Efficient DOM updates

Only changes are rendered

# JSX

Statically-typed, OOP language

Faster, safer, easier than plain Javascript

Produces React elements

Transpiles to Javascript function calls which evaluate to Javascript objects

Embedded values in JSX are escaped

# JSX examples

```
const hello = <h1>Hello, world!</h1>;
const div = <div tabIndex="0"></div>;

const greeting = x => `Hello, ${x}!`;
const hello2 = <h1>{greeting('world')}</h1>;

const nested =  (
  <div>
    <h1>Hello,</h1>
    <h2>World!</h2>
  </div>
);
```

```
"use strict";
var element = React.createElement(
  "h1",
  null,
  "Hello, world!"
);

var element = React.createElement("div", { tabIndex: "0" });
```

# React JSX examples

```jsx
const element = (
  <h1 className="greeting">
    Hello, world!
  </h1>
);

const element = React.createElement(
  'h1',
  {className: 'greeting'},
  'Hello, world!'
);

// Note: this structure is simplified
const element = {
  type: 'h1',
  props: {
    className: 'greeting',
    children: 'Hello, world'
  }
};
```

# React Components

## Functional & Class components

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

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

## Rendering

```
const element = <Welcome name="Sara" />;
```

# Class component example

```
import React from 'react';

class App extends React.Component {
  render() {
    const arr = [1, 2, 3];
    return (
      <div>
        {
          arr.length > 0 ?
            arr.map(function(int) {
              return (<div>Section {int}</div>)
            })
            : null
        }
      </div>
    );
  }
}

export default App;
```

# Component composition

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

function App() {
  return (
    <div>
      <Welcome name="Sara" />
      <Welcome name="Cahal" />
      <Welcome name="Edite" />
    </div>
  );
}

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

# Component refactoring (1)

```
function Comment(props) {
  return (
    <div className="Comment">
      <div className="UserInfo">
        <img className="Avatar"
          src={props.author.avatarUrl}
          alt={props.author.name}
        />
        <div className="UserInfo-name">
          {props.author.name}
        </div>
      </div>
      <div className="Comment-text">
        {props.text}
      </div>
      <div className="Comment-date">
        {formatDate(props.date)}
      </div>
    </div>
  );
}
```

# Component refactoring (2)

```
function Avatar(props) {
  return (
    <img className="Avatar"
      src={props.user.avatarUrl}
      alt={props.user.name}
    />
  );
}

function UserInfo(props) {
  return (
    <div className="UserInfo">
      <Avatar user={props.user} />
      <div className="UserInfo-name">
        {props.user.name}
      </div>
    </div>
  );
}
```

# Component refactoring (3)

```
function Comment(props) {
  return (
    <div className="Comment">
      <UserInfo user={props.author} />
      <div className="Comment-text">
        {props.text}
      </div>
      <div className="Comment-date">
        {formatDate(props.date)}
      </div>
    </div>
  );
}
```

# React in HTML

```
<div id="myReactApp"></div>

<script type="text/babel">
  class Greeter extends React.Component {
    render() {
      return <h1>{this.props.greeting}</h1>
    }
  }

  ReactDOM.render(<Greeter greeting="Hello World!" />,
  document.getElementById('myReactApp'));
</script>
```

# State & Lifecycle

Component acts as pure functions of their props

## Naive example

```
function Clock(props) {
  return (
    <div>
      <h1>Hello, world!</h1>
      <h2>It is {props.date.toLocaleTimeString()}.</h2>
    </div>
  );
}

function tick() {
  ReactDOM.render(
    <Clock date={new Date()} />,
    document.getElementById('root')
  );
}

setInterval(tick, 1000);
```

# State & Lifecycle (2)

```
class Clock extends React.Component {
  constructor(props) {
    super(props);
    this.state = {date: new Date()};
  }

  componentDidMount() {
    this.timerID = setInterval(
      () => this.tick(),
      1000
    );
  }

  componentWillUnmount() {
    clearInterval(this.timerID);
  }
```

```
  tick() {
    this.setState({
      date: new Date()
    });
  }

  render() {
    return (
      <div>
        <h1>Hello, world!</h1>
        <h2>It is {this.state.date.toLocaleTimeString()}.</h2>
      </div>
    );
  }
}

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

# Getting Started

```
npm init -y
npm install create-react-app
node_modules/.bin/create-react-app hello-world

cd hello-world
npm start
```

# Further material

## React Redux

UI as a function of state

State updates in response to actions

## React Router

Sync the UI with the URL

Dynamic route matching

# References


1. https://en.wikipedia.org/wiki/React_(JavaScript_library)
2. https://reactjs.org/docs


# Figures


1. https://en.wikipedia.org/wiki/React_(JavaScript_library)#/media/File:React-icon.svg
2. https://medium.jonasbandi.net/angular-vs-react-popularity-ea2659308cd5