02

OPEN ORIENTED

凹 凸 实 验 室

React 学习笔记（二）

Simba

.符号访问组件集合


```
const MyComponents = {
  DatePicker: function DatePicker(props) {
    return <div>Imagine a {props.color} datepicker here.</div>;
  }
}

function BlueDatePicker() {
  return <MyComponents.DatePicker color="blue" />;
}
```

扩展运算符（ES6）


```
function App1() {
  return <Greeting firstName="Ben" lastName="Hector" />;
}

function App2() {
  const props = {firstName: 'Ben', lastName: 'Hector'};
  return <Greeting {...props} />;
}
```

参数类型检查


```
class Hi2 extends React.Component {
  render() {
    return (
      <h2>Hi, {this.props.name}</h2>
    );
  }
}
Hi2.propTypes = {
  name: React.PropTypes.number,
  customProp: function(props, propName, componentName) {
  //自定义
  }
}
```

React without JSX


```
class Hello extends React.Component {
  render() {
    return React.createElement('div', null, `Hello ${this.props.toWhat}`);
  }
}

ReactDOM.render(
  React.createElement(Hello, {toWhat: 'World'}, null),
  document.getElementById('root')
);
```

更多

DOM的引用

```
<input type=“text"  ref={(input) => { this.textInput = input; }} />
```

动画和过渡效果

```
import ReactCSSTransitionGroup from 'react-addons-css-transition-group'
```

SSR

```
ReactDOMServer.renderToString(element)
```

https://github.com/webpack/react-webpack-server-side-example

Flux

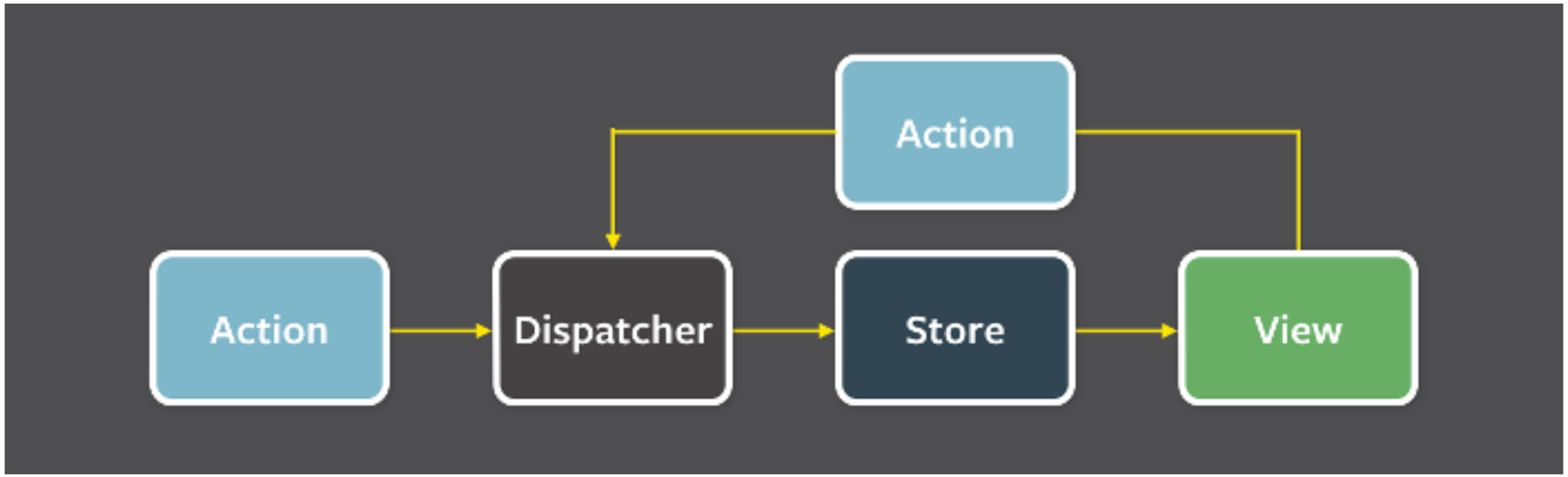
Views：UI层

Actions：行为

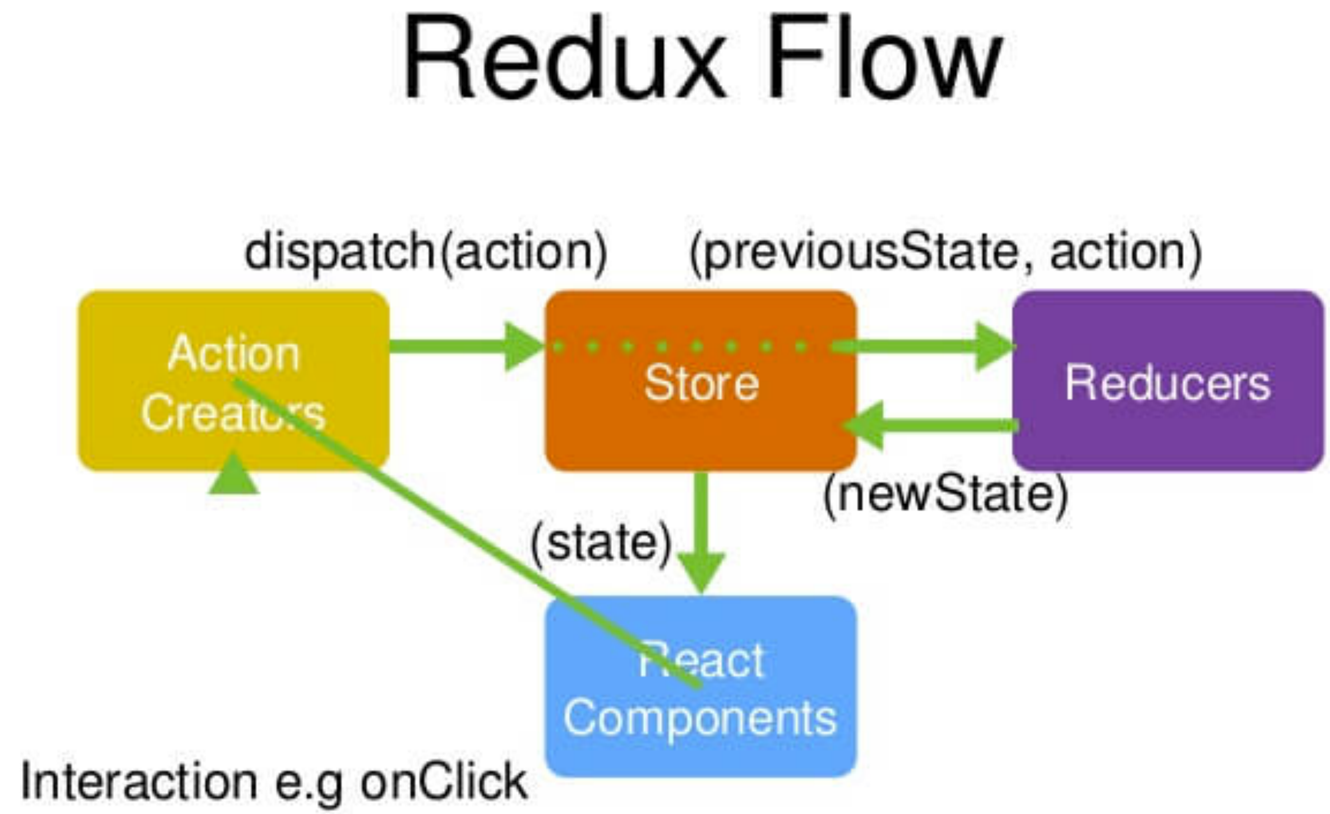
Dispatcher: 接受actions，并处理回调

Stores: 管理应用的状态，并通知View层进行更新

Flux



Redux


Redux Flow
dispatch(action)
(previousState, action)
Action Creators
Store
Reducers
(newState)
(state)
React Components
Interaction e.g onClick
React + Redux
@nikgraf

React-Redux

connect方法 → 将states、actions、components联系起来，生成容器组件

<Provider> 组件 → 所有子组件就默认都可以拿到state

THANKS

FOR YOUR WATCHING

02

OPEN ORIENTED

凹 凸 实 验 室