

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

```

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

```

|||||

childContextTypes |

getChildContext |||

React|||

context

|||||

|||||

context |||

```

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

```

|||||

contextTypes |||

context |||

contextTypes |||

this.context |||

|||||

contextTypes |||

context |||

```

void componentWillReceiveProps(
  object nextProps, object nextContext
)

```

```

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

 #1

   2017-04-09 10:10:41 UTC

   2017-04-09 10:10:41 UTC