React.js (v0.14) cheatsheet

Deprecated: this guide targets an old version of React (v0.14). See the updated React cheatsheet for new versions.

Components

```
var Component = React.createClass({ render:
function () { return <div>Hello {this.props.name}
</div>; } });
```

```
ReactDOM.render(<Component name="John" />,
document.body);
```

Use the React.js jsfiddle to start hacking. (or the unofficial jsbin)

Nesting

```
var UserAvatar = React.
UserProfile = React.cre
```

```
var Info = React.create
<div> <UserAvatar src={
<UserProfile username={
</div>; } });
```

Nest components to sepa
components.

States & Properties

States and props

```
<MyComponent fullscreen={true} />
```

```
// props this.props.fullscreen //=> true // state
this.setState({ username: 'rstacruz' });
this.replaceState({ ... }); this.state.username
//=> 'rstacruz'
```

```
render: function () { return <div className=
{this.props.fullscreen ? 'full' : ''}> Welcome,
{this.state.username} </div>; }
```

Setting defaults

```
React.createClass({ get
return { comments: [] }
function () { return {
```

Pre-populates this.state
this.props.name.

Use props (this.props) to access parameters passed from the parent. Use states (this.state) to manage dynamic data.

Components

Component API

```
ReactDOM.findDOMNode(c) // 0.14+
React.findDOMNode(c) // 0.13 c.getDOMNode() // 0.12 below
```

```
c.forceUpdate() c.isMounted() c.state c.props
c.setState({ ... }) c.replaceState({ ... })
c.setProps({ ... }) // for deprecation
c.replaceProps({ ... }) // for deprecation c.refs
```

These are methods available for `Component` instances. See Component API.

Component specs

```
render()
```

```
getInitialState()
getDefaultProps()
```

```
mixins: [ ... ]
propTypes: { ... }
statics: { ... }
displayName: "..."
```

Methods and properties y
specs.

Lifecycle

Mounting

`componentWillMount()`	Before rendering (no DOM yet)
`componentDidMount()`	After rendering

Before initial rendering occurs. Add your DOM stuff on didMount (events, timers, etc). See reference.

Updating

```
componentWillRecei
```

```
shouldComponentUpd
```

Unmounting

`componentWillUnmount()`	Invoked before DOM removal

Clear your DOM stuff here (probably done on didMount). See reference.

Examples

Example: loading data

```
React.createClass({ componentDidMount: function () { $.get(this.props.url, function (data) { this.setState(data); }.bind(this)); }, render: function () { return <CommentList data={this.state.data} /> } });
```

See initial AJAX data.

DOM nodes

References

```
<input ref="myInput">
```

```
this.refs.myInput
ReactDOM.findDOMNode(this.refs.myInput).focus()
ReactDOM.findDOMNode(this.refs.myInput).value
```

Two-way binding

```
Email: <input type="text" valueLink={this.linkState('email')} />
```

componentWillUpdat

componentDidUpdate

Called when parents chan
These are not called for in

DOM Events

Add attributes like onChan

```
<input type="text" valu
onChange={this.handleCh
```

```
handleChange: function(
value: event.target.val
```

Allows access to DOM no

```
React.createClass({ mixins:
[React.addons.LinkedStateMixin] });
```

```
this.state.email
```

Use LinkedStateMixin for easier two-way binding.

Property validation

Basic types

```
React.createClass({ propTypes: { email:
React.PropTypes.string, seats:
React.PropTypes.number, settings:
React.PropTypes.object, callback:
React.PropTypes.func, isClosed:
React.PropTypes.bool, any: React.PropTypes.any, }
});
```

Primitive types: `.string`, `.number`, `.func`, and `.bool`. See `propTypes`.

Required types

```
propTypes: { requiredFu
React.PropTypes.func.is
React.PropTypes.any.isR
```

Add `.isRequired`.

React elements

```
propTypes: { element: R
react element node: Rea
string, element // ...c
```

Use `.element`, `.node`.

Enumerables

```
propTypes: { enum:
React.PropTypes.oneOf(['M','F']), // enum union:
React.PropTypes.oneOfType([ // any
React.PropTypes.string, React.PropTypes.number ]),
```

Use `.oneOf`, `.oneOfType`.

Arrays and objects

```
propTypes: { array: Rea
React.PropTypes.arrayOf
object: React.PropTypes
React.PropTypes.objectO
message: React.PropType
object2: React.PropType
React.PropTypes.string,
React.PropTypes.number
```

Use `.array[Of]`, `.object[`

Custom validation

```
propTypes: { customProp: function(props, propName,
componentName) { if
(!/matchme/.test(props[propName])) { return new
Error('Validation failed!'); } } }
```

Supply your own function.

Other features

Class set

```
var cx = require('classnames'); render: function() { var classes = cx({ 'message': true, 'message-important': this.props.isImportant, 'message-read': this.props.isRead }); return <div className={classes}>Great Scott!</div>; }
```

Manipulate DOM classes with classnames, previously known as `React.addons.classSet`. See Class set.

Propagating propertie

```
<VideoPlayer src="video

var VideoPlayer = React
function() { /* propaga
sub component */ return
controls='false' />; }
```

See Transferring props.

Mixins

```
var SetIntervalMixin =
function() { .. } }

var TickTock = React.cr
[SetIntervalMixin] }
```

See addons for some built

Top level API

```
React.createClass({ ... }) React.isValidElement(c)
ReactDOM.findDOMNode(c) // 0.14+
ReactDOM.render(<Component />, domnode, [callback])
// 0.14+ ReactDOM.unmountComponentAtNode(domnode)
// 0.14+ ReactDOMServer.renderToString(<Component
/>) // 0.14+
ReactDOMServer.renderToStaticMarkup(<Component />)
// 0.14+
```

JSX patterns

Style shorthand

```
var style = { backgroundImage: 'url(x.jpg)',
height: 10 }; return <div style={style}></div>;
```

See inline styles.

InnerHTML

```
function markdownify()
<div dangerouslySetInne
markdownify()}} />
```

See dangerously set innerl

Lists

```
var TodoList = React.createClass({ render:
function() { function item(itemText) { return <li>
{itemText}</li>; }; return <ul>
{this.props.items.map(item)}</ul>; } });
```

See also

- Animations

Over 380 curated cheatsheets, by developers for developers.

Devhints home

Other React cheatsheets

- React.js cheatsheet
- Redux cheatsheet
- Enzyme cheatsheet
- Enzyme v2 cheatsheet
- Awesome Redux cheatsheet
- Flux architecture cheatsheet

Top cheatsheets

- Elixir cheatsheet
- ES2015+ cheatsheet
- React.js cheatsheet
- Vimdiff cheatsheet
- Vim cheatsheet
- Vim scripting cheatsheet