REACT CHEATSHEET (V0.14)

powered by Azat Mardan, http://webapplog.com

INSTALLATION

React

```
<script src="https://fb.me/react-0.14.0.js"></script>
```

```
$ npm install react --save
```

```
$ bower install react --save
```

React DOM

```
<script src="https://fb.me/react-dom-0.14.0.js"></script>
```

```
$ npm install react-dom
```

```
$ bower install react-dom --save
```

RENDERING

Rendering (ES5)

```
ReactDOM.render(
  React.createElement(Link, {name: 'HackHall.com'}),
  document.getElementById('menu')
)
```

Rendering (ES5+JSX)

```
ReactDOM.render(
  <Link name='HackHall.com'/>,
  document.getElementById('menu')
)
```

Server-side Rendering

```
var ReactDOMServer = require('react-dom/server')
ReactDOMServer.renderToString(Link, {name: 'HackHall.com'})
ReactDOMServer.renderToStaticMarkup(Link, {name: 'HackHall.com'})
```

COMPONENTS

ES5

```
var Link = React.createClass({
  displayName: 'Link',
  render: function() {
    return React.createElement('a', {className: 'btn', title:
this.props.name}, 'Click ->', this.props.name);
  }
});
```

ES5 + JSX

```
var Link = React.createClass({
  render: function() {
    return <a className='btn' title={this.props.name}>Click ->
this.props.name</a>
  }
});
```

ES6 + JSX

```
class Link extends React.Component {
  render() {
    return <a className='btn' title={this.props.name}>Click ->
this.props.name</a>
  }
}
```

LIFECYCLE EVENTS

```
componentWillMount function()
```

```
componentDidMount function()
```

```
componentWillReceiveProps function(nextProps)
```

```
shouldComponentUpdate function(nextProps, nextState)-> bool
```

```
componentWillUpdate function(nextProps, nextState)
```

```
componentDidUpdate function(prevProps, prevState)
```

```
componentWillUnmount function()
```

ADVANCED COMPONENTS

Options (ES5)

`propTypes object` : Type validation in development mode

`getDefaultProps function()` : object of default props

`getInitialState function()` : object of the initial state

ES5:

```
var Link = React.createClass ({
  propTypes: { name: React.PropTypes.string },
  getDefaultProps: function() {
    return { initialCount: 0 }
  },
  getInitialState: function() {
    return {count: this.props.initialCount};
  },
  tick: function() {
    this.setState({count: this.state.count + 1});
  },
  render: function() {
    return React.createElement(
      'a',
      {className: 'btn', href: '#', title: this.props.name, onClick:
this.tick.bind(this)},
      'Click ->',
      (this.props.name? this.props.name : 'webapplog.com'),
      ' (Clicked: '+this.state.count+')'
    );
  }
})
```

ES5 + JSX:

```
var Link = React.createClass ({
  propTypes: { name: React.PropTypes.string },
  getDefaultProps: function() {
    return { initialCount: 0 }
  },
  getInitialState: function() {
    return {count: this.props.initialCount};
  },
  tick: function() {
    this.setState({count: this.state.count + 1});
  },
  render: function() {
    return (
      <a onClick={this.tick.bind(this)} href="#" className="btn"
title={this.props.name}>
        Click -> {(this.props.name? this.props.name : 'webapplog.com')}
        (Clicked: {this.state.count})
      </a>
    );
  }
})
```

ES6 + JSX:

```
export class Link extends React.Component {
  constructor(props) {
    super(props);
    this.state = {count: props.initialCount};
  }
  tick() {
    this.setState({count: this.state.count + 1});
  }
  render() {
    return (
      <a onClick={this.tick.bind(this)} href="#" className="btn"
title={this.props.name}>
        Click -> {(this.props.name? this.props.name : 'webapplog.com')}
        (Clicked: {this.state.count})
      </a>
    );
  }
}
Link.propTypes = { initialCount: React.PropTypes.number };
Link.defaultProps = { initialCount: 0 };
```

SEQUENCE OF LIFECYCLE EVENTS

Mounting	Updating			Unmounting
	Component Properties	Component State	Using forceUpdate()	
getDefaultProps()				
getInitialState()				
componentWillMount()				
	componentWillReceiveProps()			
	shouldComponentUpdate()			
	componentWillUpdate()			
	render()			
	componentDidUpdate()			
componentDidMount()				
				componentWillUnmount()

SPECIAL PROPS

`key` : Unique identifier for an element to turn arrays/lists into hashes for better performance, e.g., `key={id}`

`ref` : Reference to an element via `this.refs.NAME`, e.g., `ref="email"` will create `this.refs.email` DOM node or `ReactDOM.findDOMNode(this.refs.email)`

`style` : Accept an object of styles, instead of a string (immutable since v0.14), e.g., `style={{color: red}}`

`className` : the HTML `class` attribute, e.g., `className="btn"`

`htmlFor` : the HTML `for` attribute, e.g., `htmlFor="email"`

`dangerouslySetInnerHTML` : raw HTML by providing an object with the key `__html`

`children` : content of the element via `this.props.children`, e.g., `this.props.children[0]`

`data-NAME` : custom attribute, e.g., `data-tooltip-text="..."`

propTypes

Types available under `React.PropTypes` :

`any` `bool` `func` `number` `string`
`array` `element` `node` `object`

To make required, append `.isRequired`

More methods:

`instanceOf(constructor)`
`oneOf(['News', 'Photos'])`
`oneOfType([propType, propType])`

CUSTOM VALIDATION

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

COMPONENT PROPERTIES AND METHODS

Properties:

`this.refs` : Lists components with a `ref` prop

`this.props` : Any props passed to an element (immutable)

`this.state` : State set by setState and getInitialState (muttable) — avoid setting state manually with `this.state=...`

`this.isMounted` : Flag whether the element has a corresponding DOM node or not

Methods:

`setState(changes)` : Change state (partially) to `this.state` and trigger re-render

`replaceState(newState)` : Replace `this.state` and trigger re-render

`forceUpdate()` : Trigger DOM re-render immediately

REACT ADDONS

As npm modules:

`react-addons-css-transition-group`
`react-addons-perf`
`react-addons-test-utils`
`react-addons-pure-render-mixin`
`react-addons-linked-state-mixin`
`react-addons-clone-with-props`
`react-addons-create-fragment`
`react-addons-css-transition-group`
`react-addons-linked-state-mixin`
`react-addons-pure-render-mixin`
`react-addons-shallow-compare`
`react-addons-transition-group`
`react-addons-update`