snippet **rrcc**

# "React Redux Class Component"

import React, { Component} from 'react';

import PropTypes from 'prop-types';

import { connect } from 'react-redux';

import styles from './${2:$1}.css';

class ${1:`!v expand('%:t:r')`} extends Component {

static propTypes = {

children: PropTypes.node,

className: PropTypes.string,

dispatch: PropTypes.func.isRequired,

};

constructor(props) {

super(props);

}

render() {

return (

<div className={styles.base}>

$3

</div>

);

}

}

function mapStateToProps(state) {

return {};

}

export default connect(mapStateToProps)($1);

snippet **rcc**

# "React Class Component"

import React, { Component} from 'react';

import PropTypes from 'prop-types';

import styles from './${2:$1}.css';

class ${1:`!v expand('%:t:r')`} extends Component {

static propTypes = {

children: PropTypes.node,

className: PropTypes.string,

};

constructor(props) {

super(props);

}

render() {

return (

<div className={styles.base}>

$3

</div>

);

}

}

export default $1;

snippet **rfc**

# "React Functional Component"

import React from 'react';

import PropTypes from 'prop-types';

import styles from './${2:$1}.css';

function ${1:`!v expand('%:t:r')`}(${3:{...props}}) {

return (

<div className={styles.base}>

$4

</div>

);

}

$1.defaultProps = {};

$1.propTypes = {};

export default $1;

snippet **rsc**

# "React Styled Component"

import styled from 'styled-components';

const ${1:`!v expand('%:t:r')`} = styled.${2:div}\`

$3

\`;

export default $1;

snippet **rsci**

# "React Styled Component Interpolation"

import styled, { css } from 'styled-components';

const ${1:`!v expand('%:t:r')`} = styled.${2:div}\`${props => css\`

${3:${props.$4 && \`

$5

## \`}}

\`}\`;

export default $1;

snippet **pp**

## ${props => props.${1}};

snippet **cn**

## className="$1"

snippet **dp**

## "Default Props"

${1:`!v expand('%:t:r')`.}defaultProps = {

$2

};

snippet **set**

# "Set State"

this.setState({

${1}: ${2}

});

snippet **props**

## this.props.${1}

snippet **state**

## this.state.${1}

snippet **ref**

## ref={${1:ref} => { this.${2:name} = $1; }}

# Component Lifecycle

snippet **cwm**

## "Component Will Mount"

componentWillMount() {

$1

}

snippet **cdm**

## "Component Did Mount"

componentDidMount() {

$1

}

snippet **cwrp**

## "Component Will Receive Props"

componentWillReceiveProps(nextProps) {

$1

}

snippet **scup**

## "Should Component Update"

shouldComponentUpdate(nextProps, nextState) {

$1

}

snippet **cwup**

## "Component Will Update"

componentWillUpdate(nextProps, nextState) {

$1

}

snippet **cdup**

## "Component Did Update"

componentDidUpdate(prevProps, prevState) {

$1

}

snippet **cwu**

## "Component Will Unmount"

componentWillUnmount() {

$1

}

snippet **ren**

## "Render"

render() {

return ${1:(

${2:<div>${3}</div>}

);}

}

# PropTypes

snippet **pt**

## "PropTypes Definition"

${1:`!v expand('%:t:r')`.}propTypes = {

${2:className}: ${3:PropTypes.string},

};

snippet **pt.a**

## "PropTypes Array"

PropTypes.array${1:,}

snippet **pt.b**

## "PropTypes Boolean"

PropTypes.bool${1:,}

snippet **pt.f**

## "PropTypes Function"

PropTypes.func${1:,}

snippet **pt.n**

## "PropTypes Number"

PropTypes.number${1:,}

snippet **pt.o**

"PropTypes Object"

PropTypes.object${1:,}

snippet **pt.s**

## "PropType String"

PropTypes.string${1:,}

snippet **pt.e**

## "PropTypes Element"

PropTypes.element${1:,}

snippet **pt.io**

## "PropTypes instanceOf"

PropTypes.instanceOf(${2:PropTypes.string})${1:,}

snippet **pt.one**

## "PropTypes oneOf"

PropTypes.oneOf(['$2'$3])${1:,}

snippet **pt.onet**

## "PropTypes oneOfType"

PropTypes.oneOfType([

$2

])${1:,}

snippet **pt.ao**

## "PropTypes arrayOf"

PropTypes.arrayOf(${2:PropTypes.string})${1:,}

snippet **pt.oo**

## "PropTypes objectOf"

PropTypes.objectOf(${2:PropTypes.string})${1:,}

snippet **pt.sh**

## "PropTyes Shape"

PropTypes.shape({

$2

})${1:,}