#### Raccourcis

#### v⇥ var statement

var ${1:name}

va⇥ var assignment

var ${1:name} = ${2:value}

l⇥ let statement

let ${1:name}

la⇥ let assignment awaited

let ${1:name} = await ${2:value}

ly⇥ let yielded assignment

let ${1:name} = yield ${2:value}

c⇥ const statement

const ${1:name}

cd⇥ const from destructuring

const { ${1:name} } = ${2:value}

ca⇥ const assignment awaited

const ${1:name} = await ${2:value}

cd⇥ const from destructuring awaited

const { ${1:name} } = await ${2:value}

cf⇥ const arrow function assignment

const ${1:name} = (${2:arguments}) => {\n\treturn ${0}\n}

cy⇥ const yielded assignment

const ${1:name} = yield ${2:value}

Flow Control

i⇥ if statement

if (${1:condition}) {

${0}

}

te⇥ ternary statement

${1:cond} ? ${2:true} : ${3: false}

ta⇥ ternary statement

const ${0} = ${1:cond} ? ${2:true} : ${3: false}

el⇥ else statement

else {

${0}

}

ife⇥ else statement

if (${1:condition}) {

${0}

} else {

}

ei⇥ else if statement

else if (${1:condition}) {

${0}

}

fl⇥ for loop (ES6)

for (let ${1:i} = 0, ${2:len} = ${3:iterable}.length ${1:i} < ${2:len}; ${1:i}++) {

${0}

}

fi⇥ for in loop (ES6)

for (let ${1:key} in ${2:source}) {

if (${2:source}.hasOwnProperty(${1:key})) {

${0}

}

}

fo⇥ for of loop (ES6)

for (const ${1:key} of ${2:source}) {

${0}

}

wl⇥ while loop

while (${1:condition}) {

${0}

}

wid⇥ while iteration decrementing

let ${1:array}Index = ${1:array}.length

while (${1:array}Index--) {

${0}

}

tc⇥ try/catch

try {

${0}

} catch (${1:err}) {

}

tf⇥ try/finally

try {

${0}

} finally {

}

tcf⇥ try/catch/finally

try {

${0}

} catch (${1:err}) {

} finally {

}

Functions

fan⇥ anonymous function

function (${1:arguments}) {${0}}

fn⇥ named function

function ${1:name}(${2:arguments}) {

${0}

}

asf⇥ async function

async function (${1:arguments}) {

${0}

}

aa⇥ async arrow function with

async (${1:arguments}) => {

${0}

}

iife⇥ immediately-invoked function expression (IIFE)

;(function (${1:arguments}) {

${0}

})(${2})

aiife⇥ async immediately-invoked function expression

;(async (${1:arguments}) => {

${0}

})(${2})

fa⇥ function apply

${1:fn}.apply(${2:this}, ${3:arguments})

fc⇥ function call

${1:fn}.call(${2:this}, ${3:arguments})

fb⇥ function bind

${1:fn}.bind(${2:this}, ${3:arguments})

af⇥ arrow function (ES6)

(${1:arguments}) => ${2:statement}

fd⇥ arrow function with destructuring

({${1:arguments}}) => ${2:statement}

fdr⇥ arrow function with destructuring returning destructured

({${1:arguments}}) => ${1:arguments}

f⇥ arrow function with body (ES6)

(${1:arguments}) => {

${0}

}

fr⇥ arrow function with return (ES6)

(${1:arguments}) => {

return ${0}

}

gf⇥ generator function (ES6)

function\* (${1:arguments}) {

${0}

}

gfn⇥ named generator function (ES6)

function\* ${1:name}(${1:arguments}) {

${0}

}

Iterables

fe⇥ forEach loop

${1:iterable}.forEach((${2:item}) => {

${0}

})

map⇥ map function

${1:iterable}.map((${2:item}) => {

${0}

})

reduce⇥ reduce function

${1:iterable}.reduce((${2:previous}, ${3:current}) => {

${0}

}${4:, initial})

filter⇥ filter function

${1:iterable}.filter((${2:item}) => {

${0}

})

find⇥ ES6 find function

${1:iterable}.find((${2:item}) => {

${0}

})

every⇥ every function

${1:iterable}.every((${2:item}) => {

${0}

})

some⇥ some function

${1:iterable}.some((${2:item}) => {

${0}

})

Objects and classes

cs⇥ class (ES6)

class ${1:name} {

constructor(${2:arguments}) {

${0}

}

}

csx⇥ extend a class (ES6)

class ${1:name} extends ${2:base} {

constructor(${2:arguments}) {

super(${2:arguments})

${0}

}

}

m⇥ method (ES6 syntax)

${1:method} (${2:arguments}) {

${0}

}

get⇥ getter (ES6 syntax)

get ${1:property} () {

${0}

}

set⇥ setter (ES6 syntax)

set ${1:property} (${2:value}) {

${0}

}

gs⇥ getter and setter (ES6 syntax)

get ${1:property} () {

${0}

}

set ${1:property} (${2:value}) {

}

proto⇥ prototype method

${1:Class}.prototype.${2:methodName} = function (${3:arguments}) {

${0}

}

ok Object.keys

Object.keys(${1:obj})

ov Object.values

Object.values(${1:obj})

oe Object.entries

Object.entries(${1:obj})

oc Object.create

Object.create(${1:obj})

oa Object.assign

Object.assign(${1:dest}, ${2:source})

og Object.getOwnPropertyDescriptor

Object.getOwnPropertyDescriptor(${1:dest}, '${2:prop}')

od Object.defineProperty

Object.defineProperty(${1:dest}, '${2:prop}', {

${0}

})

Returning values

r⇥ return

return ${0}

rt⇥ return this

return this

rn⇥ return null

return null

ro⇥ return new object

return {

${0}

}

ra⇥ return new array

return [

${0}

]

rp⇥ return Promise (ES6)

return new Promise((resolve, reject) => {

${0}

})

tof⇥ typeof comparison

typeof ${1:source} === '${2:undefined}'

tf⇥ this

this.

iof⇥ instanceof comparison

${1:source} instanceof ${2:Object}

ia⇥ isArray

Array.isArray(${1:source})

Promises

pa⇥ Promise.all

Promise.all(${1:value})

p⇥ new Promise (ES6)

new Promise((resolve, reject) => {

${0}

})

pt⇥ Promise.then

${1:promise}.then((${2:value}) => {

${0}

})

pc⇥ Promise.catch

${1:promise}.catch(error => {

${0}

})

ES6 modules

e⇥ module export

export ${1:member}

ed⇥ module default export

export default ${1:member}

edf⇥ module default export function

export default function ${1:name} (${2:arguments}) {\n\t${0}\n}

ec⇥ module export const

export const ${1:member} = ${2:value}

ef⇥ module export const

export function ${1:member} (${2:arguments}) {\n\t${0}\n}

im⇥ module import

import ${1:\*} from '${2:module}'

ia⇥ module import as

import ${1:\*} as ${2:name} from '${3:module}'

id⇥ module import destructuring

import { $1 } from '${2:module}'

BDD testing (Mocha, Jasmine, etc.)

desc⇥ describe

describe('${1:description}', function () {

${0}

})

it⇥ asynchronous "it"

it('${1:description}', async () => {

${0}

})

itd⇥ "it" with callback

it('${1:description}', (done) => {

${0}

})

its⇥ "it" synchronous

it('${1:description}', () => {

${0}

})

bf⇥ before test suite

before(function () {

${0}

})

bfe⇥ before each test

beforeEach(function () {

${0}

})

aft⇥ after test suite

after(function () {

${0}

})

afe⇥ after each test

afterEach(function () {

${0}

})

Timers

st⇥ setTimeout

setTimeout(() => {

${0}

}, ${1:delay})

si⇥ setInterval

setTimeout(() => {

${0}

}, ${1:delay})

sim⇥ setImmediate

setImmediate(() => {

${0}

})

DOM

ae⇥ addEventListener

${1:document}.addEventListener('${2:event}', ${3:ev} => {

${0}

})

rel⇥ removeEventListener

${1:document}.removeEventListener('${2:event}', ${3:listener})

evc dom event cancel default and propagation

ev.preventDefault()

ev.stopPropagation()

return false

gi⇥ getElementById

${1:document}.getElementById('${2:id}')

gc⇥ getElementsByClassName

Array.from(${1:document}.getElementsByClassName('${2:class}'))

gt⇥ getElementsByTagName

Array.from(${1:document}.getElementsByTagName('${2:tag}'))

qs⇥ querySelector

${1:document}.querySelector('${2:selector}')

qsa⇥ querySelectorAll

Array.from(${1:document}.querySelectorAll('${2:selector}'))

cdf⇥ createDocumentFragment

${1:document}.createDocumentFragment(${2:elem});

cel⇥ createElement

${1:document}.createElement(${2:elem});

heac⇥ appendChild

${1:document}.appendChild(${2:elem});

herc⇥ removeChild

${1:document}.removeChild(${2:elem});

hecla⇥ classList.add

${1:document}.classList.add('${2:class}');

hect⇥ classList.toggle

${1:document}.classList.toggle('${2:class}');

heclr⇥ classList.remove

${1:document}.classList.remove('${2:class}');

hega⇥ getAttribute

${1:document}.getAttribute('${2:attr}');

hesa⇥ setAttribute

${1:document}.setAttribute('${2:attr}', ${3:value});

hera⇥ removeAttribute

${1:document}.removeAttribute('${2:attr}');

Node.js

cb⇥ Node.js style callback

function (err, ${1:value}) {

if (err) throw err

t${0}

}

rq⇥ require a module

require('${1:module}')

cr⇥ require and assign a module

const ${1:module} = require('${1:module}')

em⇥ export member

exports.${1:name} = ${2:value}

me⇥ module.exports

module.exports = ${1:name}

on⇥ attach an event handler

${1:emitter}.on('${2:event}', (${3:arguments}) => {

${0}

})

Miscellaneous

uss⇥ use strict

'use strict'

js⇥ JSON Stringify

JSON.stringify($0)

jp⇥ JSON Parse

JSON.parse($0)

a⇥ await

await ${0}

apa⇥ Promise.all

await Promise.all(${1:value})

apm⇥ Promise.all map

await Promise.all(${1:array}.map((async ${2:value}) => {\n\t${0}\n}))

ast⇥ Promise sleep

await new Promise((r) => setTimeout(r, ${0}))

Console

cl⇥ console.log

console.log(${0})

cv⇥ console.log

console.log('${0}:', ${0})

ce⇥ console.error

console.error(${0})

cw⇥ console.warn

console.warn(${0})

cod⇥ console.dir

console.dir(${0})

React snippets

Are only enabled in jsx or tsx files. If you write your jsx in js files, you need to copy the react.json files manually and add it to your custom snippets.

Why do we include them here?

If you're not writing react, including them should not really bother you because they are not short as the regular JS snippets. Also IMHO react is the leading solution for FE apps deserves to be included by default, because any JS dev will have to write some react eventually over the course of his/her careeer. By having them in a single package we can easily make sure --there aren't any conflicts in the trigger prefixes.

Supported languages (file extensions)

* JavaScript (.js)
* TypeScript (.ts)
* JavaScript React (.jsx)
* TypeScript React (.tsx)

These were originally taken from <https://github.com/TimonVS/vscode-react-standard> because the maintainer wasn't able to publish a new version for months even when there was a considerable flaw in the released version. Below is a list of all available snippets and the triggers of each one.

| Trigger | Content |
| --- | --- |
| j | jsx element |
| dp | destructuring of props |
| ds | destructuring of props |
| jc | jsx self-closed element |
| jm | jsx elements map |
| jmr | jsx elements map with return |
| rfc | functional component. Prefer for 99% of new react component |
| rfce | functional component with emotion css import |
| rcc | class component skeleton |
| rccp | class component skeleton with prop types after the class |
| rcjc | class component skeleton without import and default export lines |
| rcfc | class component skeleton that contains all the lifecycle methods |
| rfcp | stateless component with prop types skeleton |
| rpt | empty propTypes declaration |
| con | class default constructor with props |
| conc | class default constructor with props and context |
| est | empty state object |
| cwm | componentWillMount method |
| cdm | componentDidMount method |
| cwr | componentWillReceiveProps method |
| cgd | componentGetDerivedStateFromProps method |
| scu | shouldComponentUpdate method |
| cwup | componentWillUpdate method |
| cdup | componentDidUpdate method |
| cwun | componentWillUnmount method |
| ren | render method |
| sst | this.setState with object as parameter |
| ssf | this.setState with function as parameter |
| tp | this.props |
| ts | this.state |
| us | useState |
| ue | useEffect |
| uec | useEffect with a cleanup function |
| ur | useRef |
| cc | createContext |
| uc | useContext |
| ume | useMemo |
| ------- | ---------------------------------------------------------------- |
| uq | useQuery to be used with graphql-codegen |
| uqc | useQuery that loads up data for current component, to be used with graphql-codegen |
| um | useMutation to be used with graphql-codegen |
| uqg | useQuery with raw gql |
| umg | useMutation with raw gql |

There are also snippets to be triggered with a text selection(trigger via insert snippet command):

jsx element wrap selection

The following table lists all the snippets that can be used for prop types. Every snippet regarding prop types begins with pt so it's easy to group it all together and explore all the available options. On top of that each prop type snippets has one equivalent when we need to declare that this property is also required. For example pta creates the PropTypes.array and ptar creates the PropTypes.array.isRequired

| Trigger | Content |
| --- | --- |
| pta | PropTypes.array, |
| ptar | PropTypes.array.isRequired, |
| ptb | PropTypes.bool, |
| ptbr | PropTypes.bool.isRequired, |
| ptf | PropTypes.func, |
| ptfr | PropTypes.func.isRequired, |
| ptn | PropTypes.number, |
| ptnr | PropTypes.number.isRequired, |
| pto | PropTypes.object., |
| ptor | PropTypes.object.isRequired, |
| pts | PropTypes.string, |
| ptsr | PropTypes.string.isRequired, |
| ptnd | PropTypes.node, |
| ptndr | PropTypes.node.isRequired, |
| ptel | PropTypes.element, |
| ptelr | PropTypes.element.isRequired, |
| pti | PropTypes.instanceOf(ClassName), |
| ptir | PropTypes.instanceOf(ClassName).isRequired, |
| pte | PropTypes.oneOf(['News', 'Photos']), |
| pter | PropTypes.oneOf(['News', 'Photos']).isRequired, |
| ptet | PropTypes.oneOfType([PropTypes.string, PropTypes.number]), |
| ptetr | PropTypes.oneOfType([PropTypes.string, PropTypes.number]).isRequired, |
| ptao | PropTypes.arrayOf(PropTypes.number), |
| ptaor | PropTypes.arrayOf(PropTypes.number).isRequired, |
| ptoo | PropTypes.objectOf(PropTypes.number), |
| ptoor | PropTypes.objectOf(PropTypes.number).isRequired, |
| ptsh | PropTypes.shape({color: PropTypes.string, fontSize: PropTypes.number}), |
| ptshr | PropTypes.shape({color: PropTypes.string, fontSize: PropTypes.number}).isRequired, |