JavaScript (ES6) code snippets - Visual Studio Marketplace

VS Code JavaScript (ES6) snippets

Visual Studio Marketplace v1.8.0 | installs 3994276 | rating average: 4.78/5 (23 ratings)

This extension contains code snippets for JavaScript in ES6 syntax for Vs Code editor (supports both JavaScript and TypeScript).

Note

All the snippets include the final semicolon `;` There is a fork of those snippets here made by @jmsv where semicolons are not included. So feel free to use them according to your needs.

codestream
Request and perform code reviews from inside your IDE. Review any code, even if it's a work-in-progress that hasn't been committed yet, and use jump-to-definition, your favorite keybindings, and other IDE tools.
Try it free

Installation

In order to install an extension you need to launch the Command Palette (Ctrl + Shift + P or Cmd + Shift + P) and type Extensions. There you have either the option to show the already installed snippets or install new ones. Search for *JavaScript (ES6) code snippets* and install it.

Supported languages (file extensions)

- JavaScript (.js)
- TypeScript (.ts)
- JavaScript React (.jsx)
- TypeScript React (.tsx)
- Html (.html)
- Vue (.vue)

Snippets

Below is a list of all available snippets and the triggers of each one. The ⇥ means the `TAB` key.

Import and export

Trigger	Content
`imp→`	imports entire module `import fs from 'fs';`
`imn→`	imports entire module without module name `import 'animate.css'`

Trigger	Content
`imd→`	imports only a portion of the module using destructing `import {rename} from 'fs';`
`ime→`	imports everything as alias from the module `import * as localAlias from 'fs';`
`ima→`	imports only a portion of the module as alias `import { rename as localRename } from 'fs';`
`rqr→`	require package `require('');`
`req→`	require package to const `const packageName = require('packageName');`
`mde→`	default module.exports `module.exports = {};`
`env→`	exports name variable `export const nameVariable = localVariable;`
`enf→`	exports name function `export const log = (parameter) => { console.log(parameter);};`
`edf→`	exports default function `export default function fileName (parameter){ console.log(parameter);};`
`ecl→`	exports default class `export default class Calculator { };`
`ece→`	exports default class by extending a base one `export default class Calculator extends BaseClass { };`

Class helpers

Trigger	Content
`con→`	adds default constructor in the class `constructor() {}`
`met→`	creates a method inside a class `add() {}`
`pge→`	creates a getter property `get propertyName() {return value;}`
`pse→`	creates a setter property `set propertyName(value) {}`

Various methods

Trigger	Content
`fre→`	forEach loop in ES6 syntax `array.forEach(currentItem => {})`
`fof→`	for ... of loop `for(const item of object) {}`
`fin→`	for ... in loop `for(const item in object) {}`
`anfn→`	creates an anonymous function `(params) => {}`
`nfn→`	creates a named function `const add = (params) => {}`
`dob→`	destructing object syntax `const {rename} = fs`
`dar→`	destructing array syntax `const [first, second] = [1,2]`
`sti→`	set interval helper method `setInterval(() => {});`
`sto→`	set timeout helper method `setTimeout(() => {});`
`prom→`	creates a new Promise `return new Promise((resolve, reject) => {});`

Trigger	Content
`thenc→`	adds then and catch declaration to a promise `.then((res) => {}).catch((err) => {});`

Console methods

Trigger	Content
`cas→`	console alert method `console.assert(expression, object)`
`ccl→`	console clear `console.clear()`
`cco→`	console count `console.count(label)`
`cdb→`	console debug `console.debug(object)`
`cdi→`	console dir `console.dir`
`cer→`	console error `console.error(object)`
`cgr→`	console group `console.group(label)`
`cge→`	console groupEnd `console.groupEnd()`
`clg→`	console log `console.log(object)`
`clo→`	console log object with name `console.log('object :>> ', object);`
`ctr→`	console trace `console.trace(object)`
`cwa→`	console warn `console.warn`
`cin→`	console info `console.info`
`clt→`	console table `console.table`
`cti→`	console time `console.time`
`cte→`	console timeEnd `console.timeEnd`