



Comparison

webpack is not the only module bundler out there. If you are choosing between using webpack or any of the bundlers below, here is a feature-by-feature comparison on how webpack fares against the current competition.

Feature	Additional chunks are loaded on demand
webpack/webpack	yes
jrburke/requirejs	yes
substack/node-browserify	no
jspm/jspm-cli	System.import
rollup/rollup	no
brunch/brunch	no
Feature	AMD <code>define</code>
webpack/webpack	yes
jrburke/requirejs	yes
substack/node-browserify	deamdify
jspm/jspm-cli	yes
rollup/rollup	rollup-plugin-amd
brunch/brunch	yes
Feature	AMD <code>require</code>
webpack/webpack	yes
jrburke/requirejs	yes
substack/node-browserify	no
jspm/jspm-cli	yes
rollup/rollup	no
brunch/brunch	yes
Feature	AMD <code>require</code> loads on demand
webpack/webpack	yes
jrburke/requirejs	with manual configuration
substack/node-browserify	no
jspm/jspm-cli	yes
rollup/rollup	no
brunch/brunch	no
Feature	CommonJS <code>exports</code>
webpack/webpack	yes
jrburke/requirejs	only wrapping in <code>define</code>
substack/node-browserify	yes
jspm/jspm-cli	yes
rollup/rollup	commonjs-plugin

brunch/brunch	yes
Feature	CommonJS <code>require</code>
webpack/webpack	yes
jrburke/requirejs	only wrapping in <code>define</code>
substack/node-browserify	yes
jspm/jspm-cli	yes
rollup/rollup	commonjs-plugin
brunch/brunch	yes
Feature	CommonJS <code>require.resolve</code>
webpack/webpack	yes
jrburke/requirejs	no
substack/node-browserify	no
jspm/jspm-cli	no
rollup/rollup	no
brunch/brunch	
Feature	Concat in require <code>require("./fi" + "le")</code>
webpack/webpack	yes
jrburke/requirejs	no ♦
substack/node-browserify	no
jspm/jspm-cli	no
rollup/rollup	no
brunch/brunch	
Feature	Debugging support
webpack/webpack	SourceUrl, SourceMaps
jrburke/requirejs	not required
substack/node-browserify	SourceMaps
jspm/jspm-cli	SourceUrl, SourceMaps
rollup/rollup	SourceUrl, SourceMaps
brunch/brunch	SourceMaps
Feature	Dependencies
webpack/webpack	19MB / 127 packages
jrburke/requirejs	11MB / 118 packages
substack/node-browserify	1.2MB / 1 package
jspm/jspm-cli	26MB / 131 packages
rollup/rollup	?MB / 3 packages
brunch/brunch	
Feature	ES2015 <code>import</code> / <code>export</code>
webpack/webpack	yes (webpack 2)
jrburke/requirejs	no
substack/node-browserify	no
jspm/jspm-cli	yes

rollup/rollup	yes
brunch/brunch	yes, via es6 module transpiler
Feature	Expressions in require (guided) <code>require("./templates/" + template)</code>
webpack/webpack	yes (all files matching included)
jrburke/requirejs	no ♦
substack/node-browserify	no
jspm/jspm-cli	no
rollup/rollup	no
brunch/brunch	no
Feature	Expressions in require (free) <code>require(moduleName)</code>
webpack/webpack	with manual configuration
jrburke/requirejs	no ♦
substack/node-browserify	no
jspm/jspm-cli	no
rollup/rollup	no
brunch/brunch	
Feature	Generate a single bundle
webpack/webpack	yes
jrburke/requirejs	yes ♦
substack/node-browserify	yes
jspm/jspm-cli	yes
rollup/rollup	yes
brunch/brunch	yes
Feature	Indirect require <code>var r = require; r("./file")</code>
webpack/webpack	yes
jrburke/requirejs	no ♦
substack/node-browserify	no
jspm/jspm-cli	no
rollup/rollup	no
brunch/brunch	
Feature	Load each file separate
webpack/webpack	no
jrburke/requirejs	yes
substack/node-browserify	no
jspm/jspm-cli	yes
rollup/rollup	no
brunch/brunch	no
Feature	Mangle path names
webpack/webpack	yes
jrburke/requirejs	no
substack/node-browserify	partial

substack/node-browserify	partial
jspm/jspm-cli	yes
rollup/rollup	not required (path names are not included in the bundle)
brunch/brunch	no
Feature	Minimizing
webpack/webpack	uglify
jrburke/requirejs	uglify, closure compiler
substack/node-browserify	uglifyify
jspm/jspm-cli	yes
rollup/rollup	uglify-plugin
brunch/brunch	UglifyJS-brunch
Feature	Multi pages build with common bundle
webpack/webpack	with manual configuration
jrburke/requirejs	yes
substack/node-browserify	with manual configuration
jspm/jspm-cli	with bundle arithmetic
rollup/rollup	no
brunch/brunch	no
Feature	Multiple bundles
webpack/webpack	yes
jrburke/requirejs	with manual configuration
substack/node-browserify	with manual configuration
jspm/jspm-cli	yes
rollup/rollup	no
brunch/brunch	yes
Feature	Node.js built-in libs <code>require("path")</code>
webpack/webpack	yes
jrburke/requirejs	no
substack/node-browserify	yes
jspm/jspm-cli	yes
rollup/rollup	node-resolve-plugin
brunch/brunch	
Feature	Other Node.js stuff
webpack/webpack	process, __dir/filename, global
jrburke/requirejs	-
substack/node-browserify	process, __dir/filename, global
jspm/jspm-cli	process, __dir/filename, global for cjs
rollup/rollup	global (commonjs-plugin)
brunch/brunch	
Feature	Plugins
webpack/webpack	yes

jrburke/requirejs	yes
substack/node-browserify	yes
jspm/jspm-cli	yes
rollup/rollup	yes
brunch/brunch	yes
Feature	Preprocessing
webpack/webpack	loaders, transforms
jrburke/requirejs	loaders
substack/node-browserify	transforms
jspm/jspm-cli	plugin translate
rollup/rollup	plugin transforms
brunch/brunch	compilers, optimizers
Feature	Replacement for browser
webpack/webpack	<code>web_modules</code> , <code>.web.js</code> , package.json field, alias config option
jrburke/requirejs	alias option
substack/node-browserify	package.json field, alias option
jspm/jspm-cli	package.json, alias option
rollup/rollup	no
brunch/brunch	
Feature	Requirable files
webpack/webpack	file system
jrburke/requirejs	web
substack/node-browserify	file system
jspm/jspm-cli	through plugins
rollup/rollup	file system or through plugins
brunch/brunch	file system
Feature	Runtime overhead
webpack/webpack	243B + 20B per module + 4B per dependency
jrburke/requirejs	14.7kB + 0B per module + (3B + X) per dependency
substack/node-browserify	415B + 25B per module + (6B + 2X) per dependency
jspm/jspm-cli	5.5kB for self-executing bundles, 38kB for full loader and polyfill, 0 plain modules, 293B CJS, 139B ES2015 System.register before gzip
rollup/rollup	none for ES2015 modules (other formats may have)
brunch/brunch	
Feature	Watch mode
webpack/webpack	yes
jrburke/requirejs	not required
substack/node-browserify	watchify
jspm/jspm-cli	not needed in dev
rollup/rollup	rollup-watch
brunch/brunch	yes

◆ in production mode (opposite in development mode)

X is the length of the path string

Bundling vs. Loading

It's important to note some key differences between *loading* and *bundling* modules. A tool like [SystemJS](#), which can be found under the hood of [JSPM](#), is used to load and transpile modules at runtime in the browser. This differs significantly from webpack, where modules are transpiled (through "loaders") and bundled before hitting the browser.

Each method has its advantages and disadvantages. Loading and transpiling modules at runtime can add a lot of overhead for larger sites and applications comprised of many modules. For this reason, SystemJS makes more sense for smaller projects where fewer modules are required. However, this may change a bit as [HTTP/2](#) will improve the speed at which files can be transferred from server to client. Note that HTTP/2 doesn't change anything about *transpiling* modules, which will always take longer when done client-side.

Further Reading

- [JSPM vs. webpack](#)
- [webpack vs. Browserify vs. SystemJS](#)

Contributors



bebraw



chrisVillanueva



pksjce



simon04



tashian

