flowup


& TS JS enthusiast


Speaker & ngBeer meetup organizer


Stay at home astronaut

Vojtech Mašek

Head of engineering


vmasek


VojtechMasek


@vmasek

flow^up

Optimizing Angular for size

flow^up

How to reduce app size

✓

- ➔ Compression
- ➔ Assets optimisation
- ➔ Lazy load modules
- ➔ Intelligent build tooling (angular CLI, webpack)
  - ◆ Minification (uglify)
  - ◆ Tree shake (webpack, rollup)
  - ◆ Ahead Of Time compilation (ngc)

✗

- ➔ Importing “non tree shakable” libs
  - ◆ Libs with side-effects
- ➔ Bad imports
  - ◆ Mistake in import can cost hundreds of kilobytes (rxjs/internal)
- ➔ One big shared module
- ➔ Messing with tracking scripts
- ➔ Non-optimized or multiple web fonts

flow^up

Compression

Gzip

Compressed file format

Supported by all browsers
(since IE6+, Firefox 2+, Chrome 1+ etc)

Brotli

Lossless compression algorithm

Supported by all current browsers
(Chrome 50+, Firefox 44+, Safari 11+, Edge 15)

Brotli

flow^up

- Optimizes text-based assets
- Top 1000 URLs, brotli performance is: [1]
  - **14%** smaller than gzip for **JavaScript**
  - **21%** smaller than gzip for **HTML**
  - **17%** smaller than gzip for **CSS**
- Wide server support
- Can compress faster than gzip and still produce smaller files

[1] - https://certsimple.com/blog/nginx-brotli

Tree shaking

flow^up

- Term used for **dead-code elimination**
- Relies on the static structure of ES2015 module syntax, i.e. `import` and `export`
- Tree shakable libraries shouldn't have side effects within their modules
  - Functions and properties that are “pure” are safe to prune/omit if unused

Ahead-of-Time (AoT) compilation

flowup

- Significant part of Angular not bundled as it is not needed in client
  - bootstrap is faster as only ½ the time is spent in scripting phase
- HTML-like templates are challenge for tools and compilers
  - Cannot be analyzed
  - Not sure what is referenced within the templates
- AoT compiler transpiles the Angular HTML-like templates
  - It uses TS/JS with ES2015 module imports
  - Makes efficient tree-shake during bundling
- Static-code analysis resulting in bigger size optimisation
- Allows us to perform type checking in templates

https://angular.io/guide/aot-compiler

Code splitting

flowup

- Separate chunks of code loaded on demand
- Application splitted into multiple files
  - main, feature chunks + other
- Behavior stays the same
- Common module

  holds the “shared” code

```
vmasek@flowup ~/PROJECTS/website master ●+ ng build --prod
Date: 2019-01-26T21:22:19.786Z
Hash: 653b4fda2d35d9b94136
Time: 17035ms
chunk {0} common.612690d14e24c01d7aa0.js (common) 29.6 kB  [rendered]
chunk {1} 1.1861e6b30129ed937f5a.js () 96.8 kB  [rendered]
chunk {2} 2.1f528d3bf38327b0440b.js () 158 kB  [rendered]
chunk {3} 3.92b025b8013a8b2fdc3c.js () 17.2 kB  [rendered]
chunk {4} runtime.9b26ce2a73b3bb05b24a.js (runtime) 2.48 kB [entry] [rendered]
chunk {5} 5.770c55e12503f6e16d4a.js () 150 kB  [rendered]
chunk {6} main.b484173cbde20add6382.js (main) 422 kB [initial] [rendered]
chunk {7} polyfills.b7e91c7dd6518cd8e517.js (polyfills) 44.9 kB [initial] [rendered]
chunk {8} styles.21d52a1267630ebf62c0.css (styles) 6.73 kB [initial] [rendered]
chunk {9} 9.ca97d2f0373d07be31f9.js () 20.4 kB  [rendered]
chunk {10} 10.e2d37934ef00caad50ca.js () 14.7 kB  [rendered]
chunk {11} 11.6e1430f04fd53848388f.js () 13.3 kB  [rendered]
chunk {12} 12.ed1d7d3ad4d44113796b.js () 31.4 kB  [rendered]
chunk {13} 13.b1ef392f05d420318ebf.js () 40.9 kB  [rendered]
chunk {14} 14.a2e13350f304b827242b.js () 1.02 kB  [rendered]
chunk {15} 15.63e4b4dfd6a5a4b0d07c.js () 37.4 kB  [rendered]
chunk {16} 16.30e321e44be5b61a38b0.js () 17.9 kB  [rendered]
chunk {17} 17.2b02d6d48520c7f648d6.js () 35.6 kB  [rendered]
```

Lazy loading feature modules

flowup

- Application loads modules when they are needed
- Angular has integrated solution for lazy loading modules and code splitting
  - use `loadChildren` instead of directly referencing `children` in routing

```
const siteRoutes: Routes = [
  {
    path: WebsiteSection.HomePage,
    loadChildren: () => import('./home-page/home-page.module').then( onfulfilled: m => m.HomePageModule),
  },
  {
    path: WebsiteSection.Services,
    loadChildren: './services/services.module#ServicesModule',
  },
];
```

https://angular.io/guide/lazy-loading-ngmodules

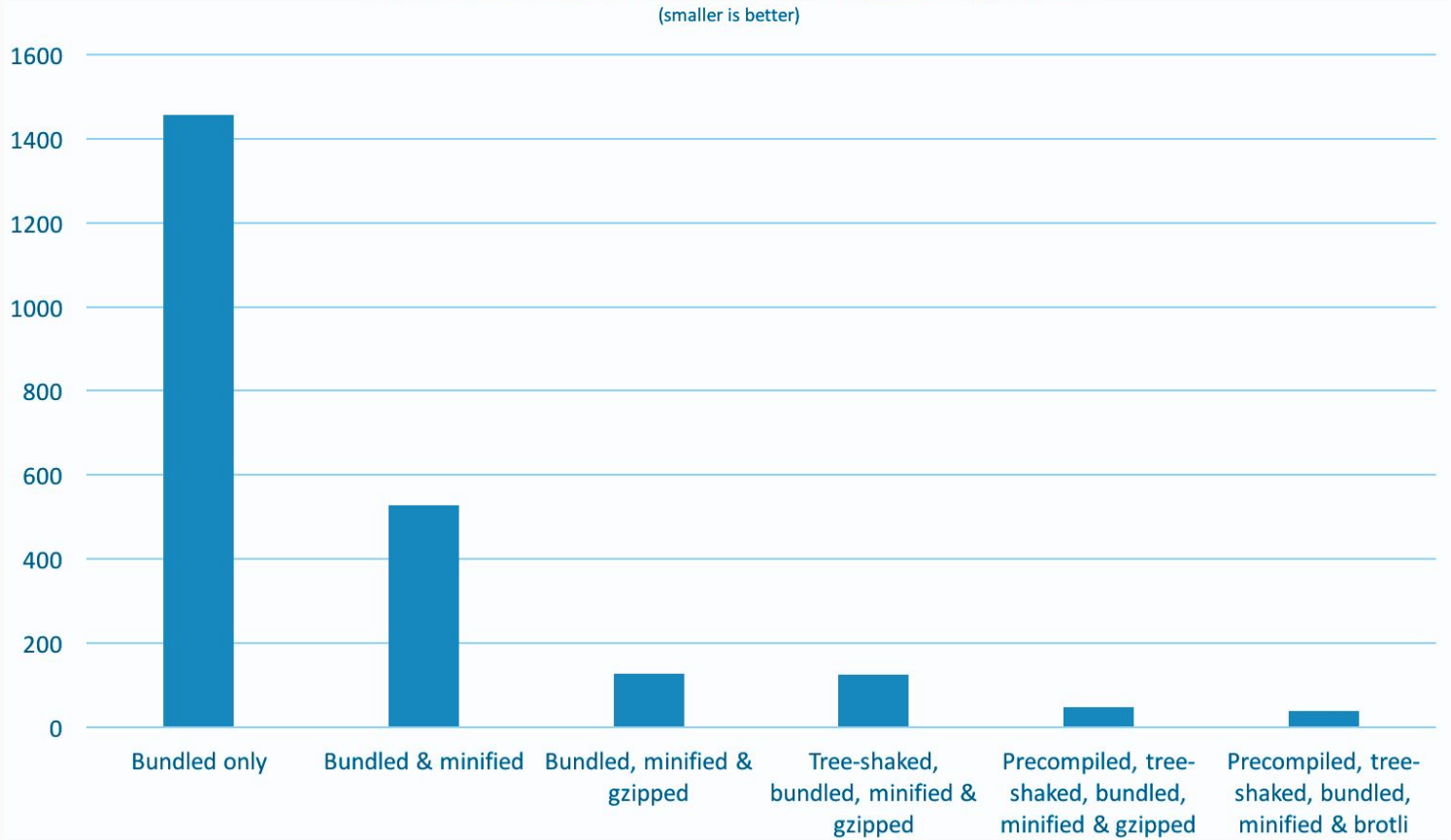
Preloading lazy modules

flow^up

- ➔ Intelligent can speedup application even more
- ➔ It affects **size** but also performance in matter of loading and bootstrap time
- ➔ There multiple strategies
  - ◆ Naive (build-in `PreloadAllModules` or `NoPreload`)
  - ◆ Link based (ngx-quicklink)
  - ◆ AI based (GuessJS)

https://angular.io/guide/router#preloading-background-loading-of-feature-areas

Bundle size of “hello world” in Angular (2) application

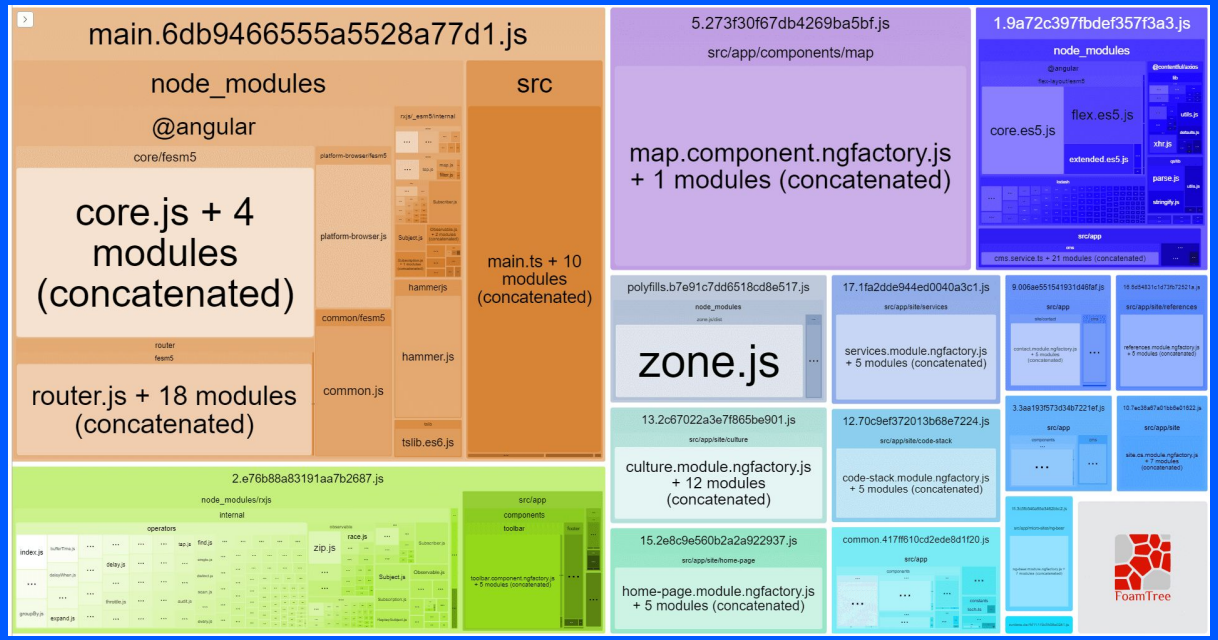


Source: https://blog.mgechev.com/2016/06/26/tree-shaking-angular2-production-build-rollup-javascript

Bundle size analysis

flowup

- Build your app `ng build --prod --stats-json`
  - This will add an extra *stats.json* with analysis of the parts wrapped inside the bundle
- Run `npx webpack-bundle-analyzer dist/stats.json`



rxjs@5.5.12

Reactive Extensions for modern JavaScript | 1 dependency | npm | github

BUNDLE SIZE

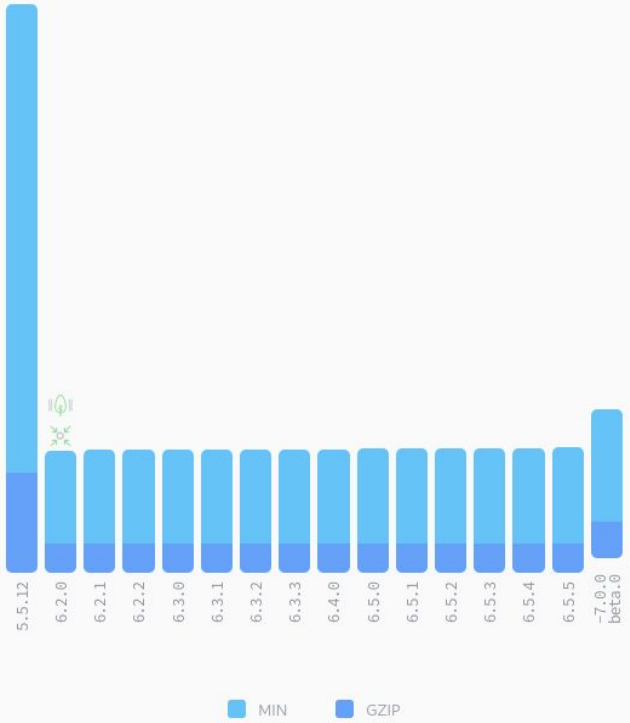
208.2 kB MINIFIED

36.7 kB MINIFIED + GZIPPED

DOWNLOAD TIME

1.22 s 2G EDGE

0.73 s EMERGING 3G



rxjs@5.5.12

Reactive Extensions for modern JavaScript | 1 dependency | npm | github

BUNDLE SIZE

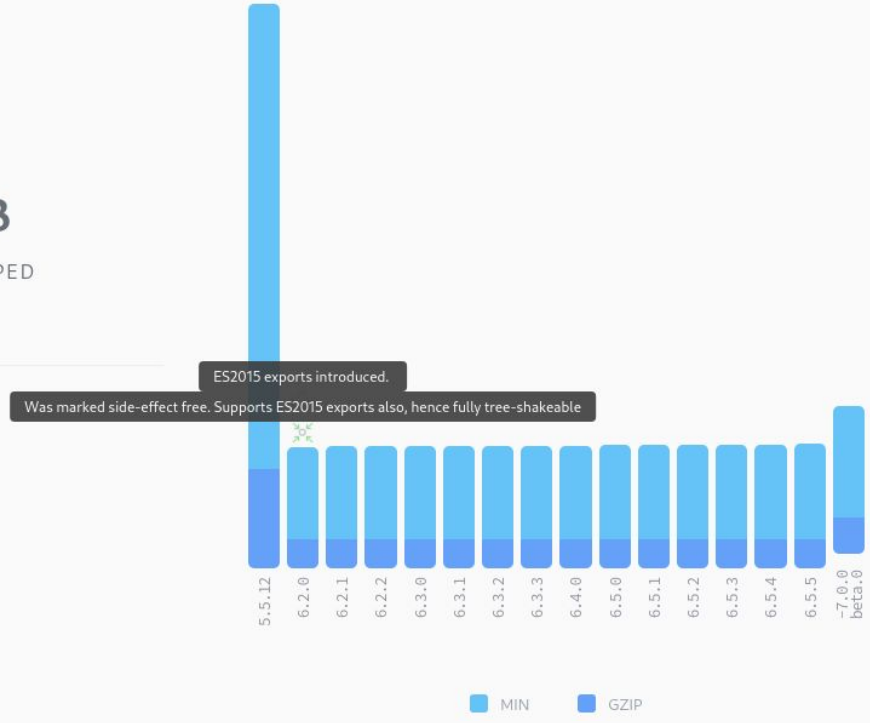
208.2 kB MINIFIED

36.7 kB MINIFIED + GZIPPED

DOWNLOAD TIME

1.22 s 2G EDGE

0.73 s EMERGING 3G



Bundle Phobia export analysis

flowup

Exports Analysis NEW

GZIP sizes of individual exports

Filter methods

a

ArgumentOutOfRangeError	239.0 B
AsyncSubject	3.3 kB
AsyncSubject	443.0 B
animationFrameScheduler	2.1 kB
animationFrameScheduler	443.0 B
asapScheduler	2.1 kB
asapScheduler	443.0 B
asyncScheduler	1.9 kB
asyncScheduler	443.0 B

b

BehaviorSubject	3.3 kB
BehaviorSubject	443.0 B
bindCallback	3.9 kB
bindCallback	443.0 B
bindNodeCallback	4.0 kB
bindNodeCallback	443.0 B

c

ConnectableObservable	3.6 kB
ConnectableObservable	443.0 B
combineLatest	3.7 kB
combineLatest	443.0 B
concat	4.3 kB
concat	443.0 B
config	244.0 B

d

defer	3.5 kB
defer	443.0 B

e

EMPTY	2.7 kB
EMPTY	443.0 B
EmptyError	231.0 B
empty	2.7 kB
empty	443.0 B

f

forkJoin	3.9 kB
forkJoin	443.0 B
from	3.4 kB
from	443.0 B
fromEvent	3.1 kB
fromEvent	443.0 B
fromEventPattern	3.0 kB
fromEventPattern	443.0 B

g

GroupedObservable	3.4 kB
GroupedObservable	443.0 B
generate	3.0 kB
generate	443.0 B

i

identity	178.0 B
iif	3.6 kB
iif	443.0 B
interval	3.5 kB
interval	443.0 B
isObservable	2.7 kB
isObservable	443.0 B

m

merge	4.3 kB
merge	443.0 B

n

NEVER	2.7 kB
NEVER	443.0 B
Notification	3.2 kB
Notification	443.0 B
NotificationKind	3.2 kB
NotificationKind	443.0 B
never	2.7 kB
never	443.0 B
noop	172.0 B

o

ObjectUnsubscribedError	256.0 B
Observable	2.6 kB
Observable	443.0 B
observable	210.0 B
of	2.8 kB
of	443.0 B
onErrorResumeNext	3.7 kB

Import size/cost plugin

flowup

- VS Code/WebStorm: *Import Cost*
- Nice overview of what are we importing

```
import 'fast-deep-equal';  1.29 kB (gzip: 603 B)
import 'hammerjs';  21.16 kB (gzip: 7.37 kB)
import 'rxjs';  50.5 kB (gzip: 12.1 kB)
import 'core-js';  90.51 kB (gzip: 30 kB)
import 'ngx-image-cropper';  516.56 kB (gzip: 128.66 kB)
import 'ngx-toastr';  522.77 kB (gzip: 128.74 kB)
import 'firebase';  827.21 kB (gzip: 223.94 kB)
import 'ngx-quill';  820.3 kB (gzip: 191.19 kB)
```

Budgets

flowup

- Configuration for you application size
- You can create budgets in angular.json config file [1]

`projects.{{ project-name }}.architect.build.configurations.{{ environment }}.budgets`

```
"budgets": [
  {
    "type": "initial",
    "maximumWarning": "200kb",
    "maximumError": "500kb"
  },
  {
    "type": "bundle",
    "name": "vendor",
    "baseline": "100kb",
    "warning": "50kb",
    "error": "100kb"
  }
]
```

```
vmasek@flowup ~/PROJECTS/website master ●+ ng build --prod
Date: 2019-01-26T21:22:19.786Z
Hash: 653b4fda2d35d9b94136
Time: 17035ms
chunk {0} common.612690d14e24c01d7aa0.js (common) 29.6 kB [rendered]
chunk {1} 1.1861e6b30129ed937f5a.js () 96.8 kB [rendered]
chunk {2} 2.1f528d3bf38327b0440b.js () 158 kB [rendered]
chunk {3} 3.92b025b8013a8b2fdc3c.js () 17.2 kB [rendered]
chunk {4} runtime.9b26ce2a73b3bb05b24a.js (runtime) 2.48 kB [entry] [rendered]
chunk {5} 5.770c55e12503f6e16d4a.js () 150 kB [rendered]
chunk {6} main.b484173cbde20add6382.js (main) 422 kB [initial] [rendered]
chunk {7} polyfills.b7e91c7dd6518cd8e517.js (polyfills) 44.9 kB [initial] [rendered]
chunk {8} styles.21d52a1267630ebf62c0.css (styles) 6.73 kB [initial] [rendered]
chunk {9} 9.ca97d2f0373d07be31f9.js () 20.4 kB [rendered]
chunk {10} 10.e2d37934ef00caad50ca.js () 14.7 kB [rendered]
chunk {11} 11.6e1430f04fd53848388f.js () 13.3 kB [rendered]
chunk {12} 12.ed1d7d3ad4d44113796b.js () 31.4 kB [rendered]
chunk {13} 13.b1ef392f05d420318ebf.js () 40.9 kB [rendered]
chunk {14} 14.a2e13350f304b827242b.js () 1.02 kB [rendered]
chunk {15} 15.63e4b4dfd6a5a4b0d07c.js () 37.4 kB [rendered]
chunk {16} 16.30e321e44be5b61a38b0.js () 17.9 kB [rendered]
chunk {17} 17.2b02d6d48520c7f648d6.js () 35.6 kB [rendered]

WARNING in budgets, maximum exceeded for initial. Budget 200 kB was exceeded by 276 kB.

WARNING in budgets, minimum exceeded for vendor. Budget 150 kB was not reached by 150 kB.

ERROR in budgets, minimum exceeded for vendor. Budget 200 kB was not reached by 200 kB.
```

[1] - https://angular.io/guide/build#configure-size-budgets

What is a polyfill?

flowup

A piece of code that provides the technology that you, the developer, expect the browser to provide natively.

Conditional polyfills

flowup

- No need to manually import and manage individual ES2015 polyfills required by Angular
- Controlled via **browserslist** config
- Works in conjunction with conditional JIT polyfills and will load ES2015 specific JIT polyfills as needed
- ES2015 polyfills are only loaded by browsers that require them
- Saves **~56KB** on native ES2015 browsers

```
vmasek@flowup ~/PROJECTS/build-example master ● ng build --prod
Date: 2019-01-26T21:03:34.897Z
Hash: 59a5cfc20f78bfc55909
Time: 9280ms
chunk {0} runtime.b57bf819d5bdce77f1c7.js (runtime) 1.41 kB [entry] [rendered]
chunk {1} es2015-polyfills.41976a8133a2445ac0d9.js (es2015-polyfills) 56.4 kB [initial] [rendered]
chunk {2} main.3e8a83393e45636251b0.js (main) 239 kB [initial] [rendered]
chunk {3} polyfills.f1a86b50434b1a515c5a.js (polyfills) 41 kB [initial] [rendered]
chunk {4} styles.3ff695c00d717f2d2a11.css (styles) 0 bytes [initial] [rendered]
```

Differential loading

flowup

- Modern browsers use *module* type in the script HTML tag and ignore a *nomodule* attribute
- Browserslist config used as single source of truth for what your app should support
- Currently only supported when using es2015 as a compilation target

```
<body>
  <app-root></app-root>
  <script src="main-es5.js" nomodule></script>
  <script src="runtime-es5.js" nomodule></script>
  <script src="polyfills-es5.js" nomodule></script>

  <script src="main-es2015.js" type="module"></script>
  <script src="runtime-es2015.js" type="module"></script>
  <script src="polyfills-es2015.js" type="module"></script>
</body>
```

https://dev.to/lacolaco/differential-loading-a-new-feature-of-angular-cli-v8-4jl
https://angular.io/guide/deployment#differential-loading

IVY renderer

flow^up

- Angular team released **tree-shakable core** and **renderer** in version 9
  - Instructions broken down into smaller, more atomic functions
  - "Hello, World" application as tiny as ~**3.3kB**
- Migration to  **9** is seamless
  - migration scripts will take care of most of the things

Optimize the static assets

flow^up

- Optimize SVG icons with **SVGOMG**
  - npm `svgo`
  -  `jakearchibald.github.io/svgomg`
- Reduce size of images
  - npm `image-webpack-loader`
  -  `tinypng.com`

Use WebP images if possible

flowup

- ➔ Lossless WebP images are **26% smaller** than PNG.
- ➔ Lossy WebP images are **25-34% smaller** than JPEG at equivalent quality

WebP image format

Image format that supports lossy and lossless compression, as well as animation and alpha transparency.

Usage

Global 79.3% + 0.19% = 79.49%

IE	Edge *	Firefox	Chrome	Safari	iOS Safari *	Opera Mini *	Chrome for Android	UC Browser for Android	Samsung Internet
		72			12.4				
	18	73	79		13.1				10.1
11	80	74	80	13	13.3	all	80	12.12	11.1
		75	81	13.1	13.4				
		76	82	TP					
			83						

```
<picture>
  <source type="image/webp" srcset="image.webp">
  <source type="image/jpeg" srcset="image.jpg">
  <img src="image.jpg" alt="Image desc">
</picture>
```

web.dev/serve-images-webp

Resize on demand

Paolo Nicolello - Rice gardens

Original 7952 x 5304 px and **9.1 MB**

Preview 2400 x 1600 px and **3.5 MB**

Resized to thumbnail of 480 x 320 px and **43.9 kB**

0.482% of the original

1.25% of the preview

Consider using some image delivery API

```
<img src="{{ url }}?w=480&h=320" />
```

- Automatically run Lighthouse on CI
  - npm `@lhci/cli`
  - GitHub `GoogleChrome/lighthouse-ci`
  - GitHub Auto run on every PR with GitHub Actions

flow^up

Thank you

Vojtech Mašek

Head of engineering


vmasek


VojtechMasek


@vmasek