



**How writing just one import the
wrong way slows down your website**

François Martin

NDC { Oslo }

karakun

François Martin

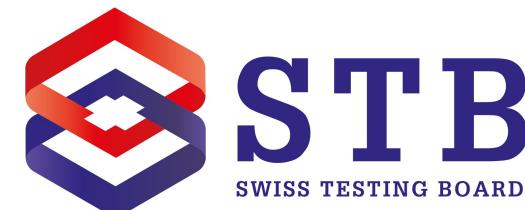
Senior Full Stack Software Engineer

✉️ francois.martin@karakun.com

📞 [martinfrancois](https://www.martinfrancois.ch)

linkedin [/in/francoismartin](https://www.linkedin.com/in/francoismartin)

X [@fmartin_](https://twitter.com/fmartin_)







Let's start with the basics...

Terminology: Bundler, Bundle (simplified)

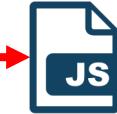
package.json:

```
{  
  "name": "hello-react",  
  "dependencies": {  
    "react": "17.0.1",  
    "react-dom": "17.0.1",  
    "lodash": "4.17.20"  
  },  
  "devDependencies": {  
    "webpack": "5.4.0"  
  }  
}
```

index.html



bundle.js



Bundler

```
<!doctype html>  
<html lang="en">  
  <body>  
    <script src="bundle.js"></script>  
  </body>  
</html>
```

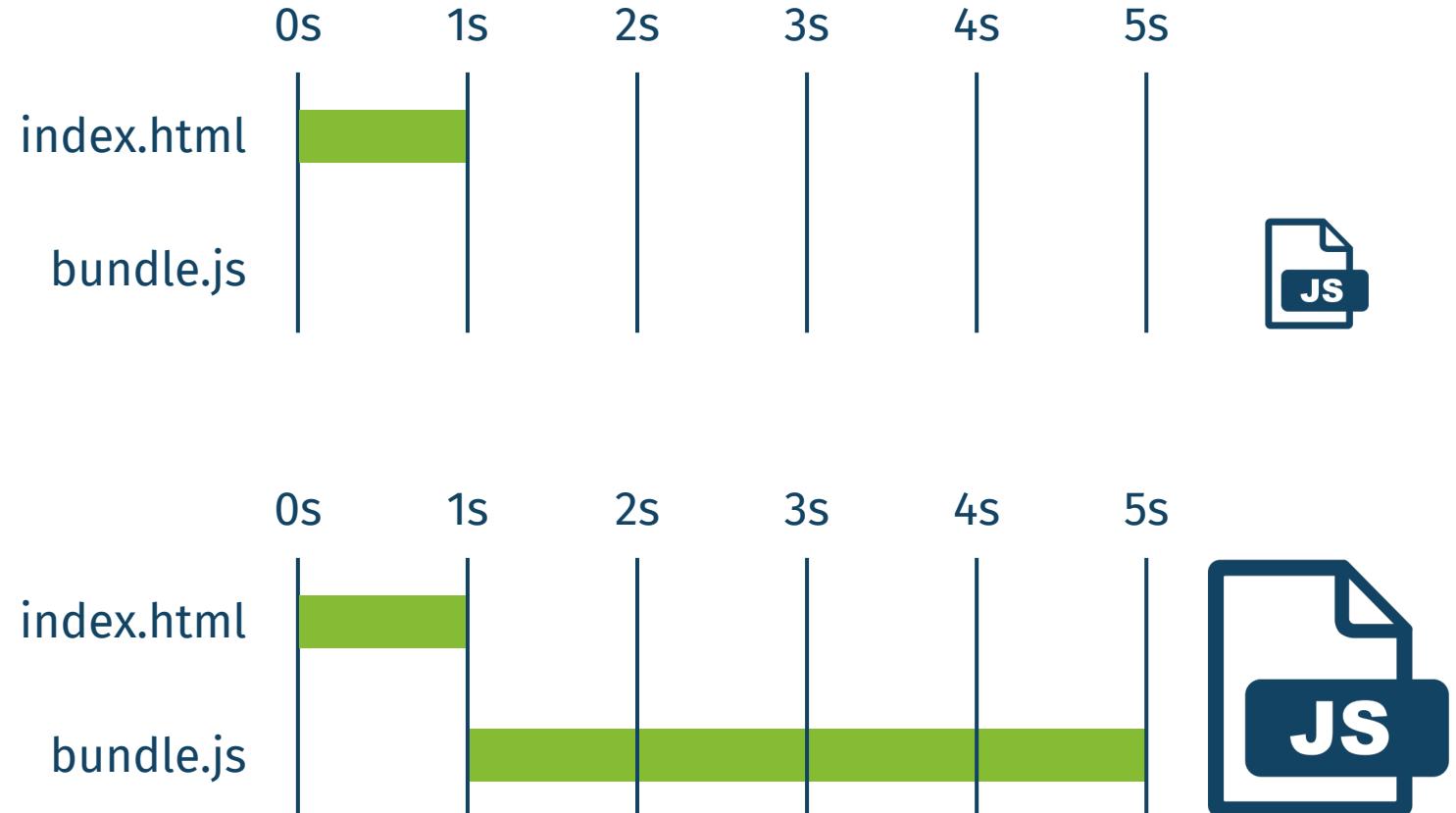
Source Code + Dependencies

Bundle Size



```
"dependencies": {  
  "react": "17.0.1",  
  "react-dom": "17.0.1",  
}
```

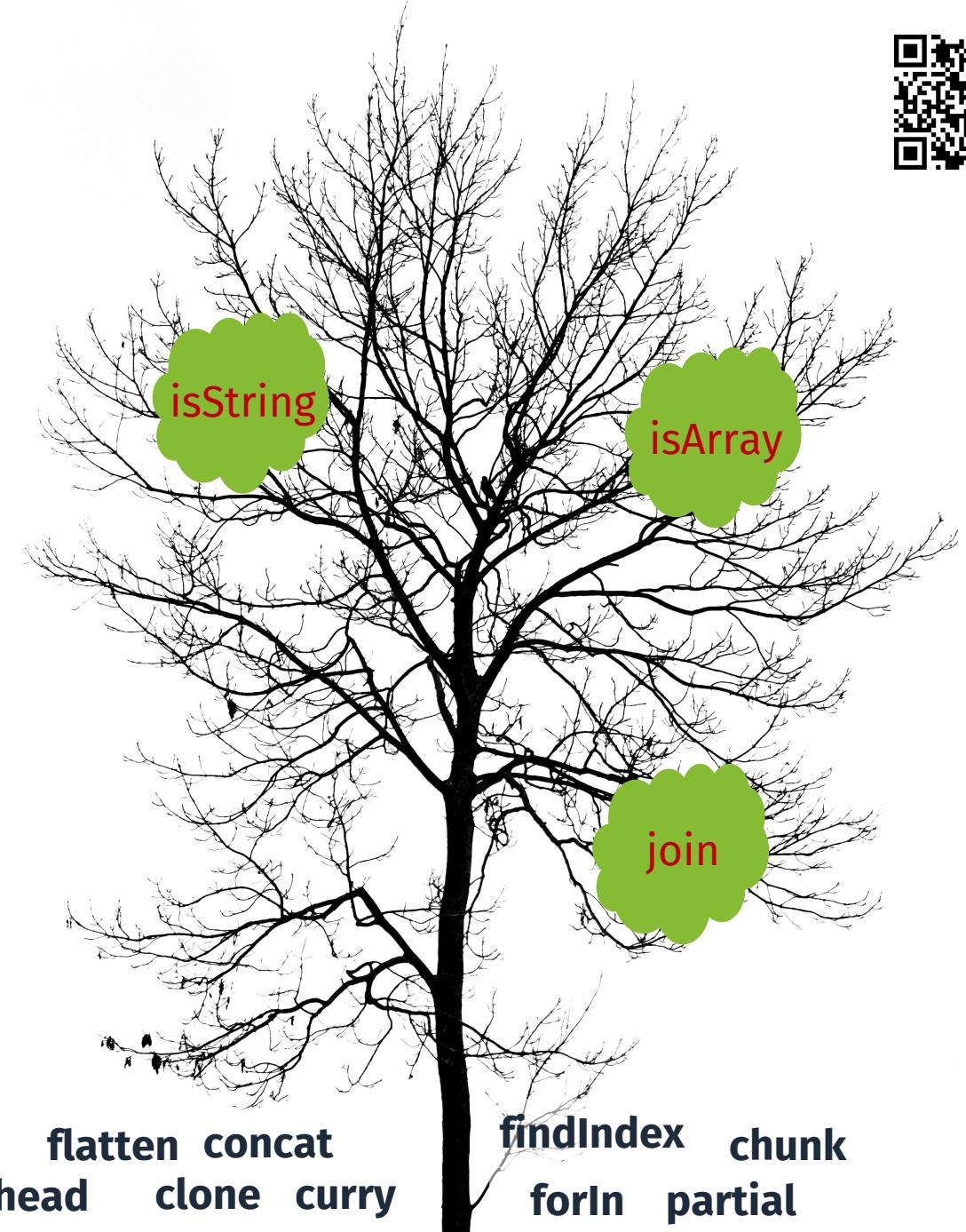
```
"dependencies": {  
  "react": "17.0.1",  
  "react-dom": "17.0.1",  
  "lodash": "4.17.20",  
  "@material-ui/core": "4.11.0",  
  "@material-ui/icons": "4.9.1",  
  "@material-ui/styles": "4.10.0",  
  "chartjs": "2.9.4",  
  "moment": "2.29.1",  
  "reactstrap": "8.7.1",  
  "the-whole-internet": "1.0.0"  
}
```



Example

```
import isString from "lodash/isString";
import isArray from "lodash/isArray";
import join from "lodash/join";

function shakeIt(input) {
  if (isString(input)) {
    return input;
  } else if (isArray(input)) {
    return join(input, "-");
  } else {
    return input.toString();
  }
}
```



Definition



- **Dead-code elimination**
- Relies on static structure of ES2015 module syntax (**import, export**)
- Name and concept popularized by bundler **rollup**.



Rich Harris

[Follow](#)

Dec 22, 2015 · 3 min read



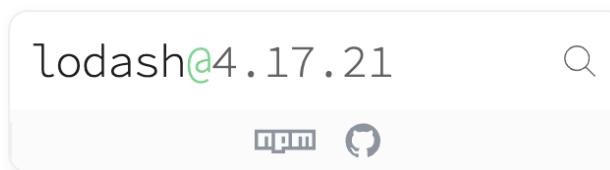
I've been working (albeit sporadically of late, admittedly) on a tool called [Rollup](#), which bundles together JavaScript modules. One of its features is *tree-shaking*, by which I mean that it only includes the bits of code your bundle actually needs to run.

- Supported by [rollup](#) since 2015
- Supported by [webpack](#) since version 2 (2016)

Adding lodash, good idea?



BUNDLEPHOBIA



Over half the size of vue!

BUNDLEPHOBIA



BUNDLE SIZE

69.8 kB	24.4 kB
MINIFIED	MINIFIED + GZIPPED

Yes, if done correctly,
will be tree-shaken!

DOWNLOAD TIME

488 ms	28 ms
SLOW 3G	EMERGING 4G

BUNDLE SIZE

112.7 kB	40.7 kB
MINIFIED	MINIFIED + GZIPPED

DOWNLOAD TIME

0.81 s	47 ms
SLOW 3G	EMERGING 4G



How to use Tree Shaking?

Most of the advice on tree shaking online: **Use ES2015 style imports / exports**

Exports:

- ✗ `function foo() {...};
module.exports = foo;`
- ✓ `export function foo() {...};`

Imports:

- ✗ `const foo = require('./foo');`
- ✗ `import foo from './foo';`
- ✓ `import { foo } from './foo';`

NEVER do this!

✗ `import lodash from 'lodash';`

**Will include EVERYTHING from lodash
in your bundle!**

Impact on Bundle Size



How much does `isString` from `lodash` increase the bundle size?

```
import { isString } from 'lodash';
```

Bundle size before: 196.924 KB

Bundle size after: 220.844 KB (**+23.92 KB**)

Wait a second...

What??? It just added the entire `lodash` to the bundle?

BUNDLEPHOBIA
BUNDLE SIZE
24.4 kB
MINIFIED + GZIPPED

Dunning-Kruger Effect





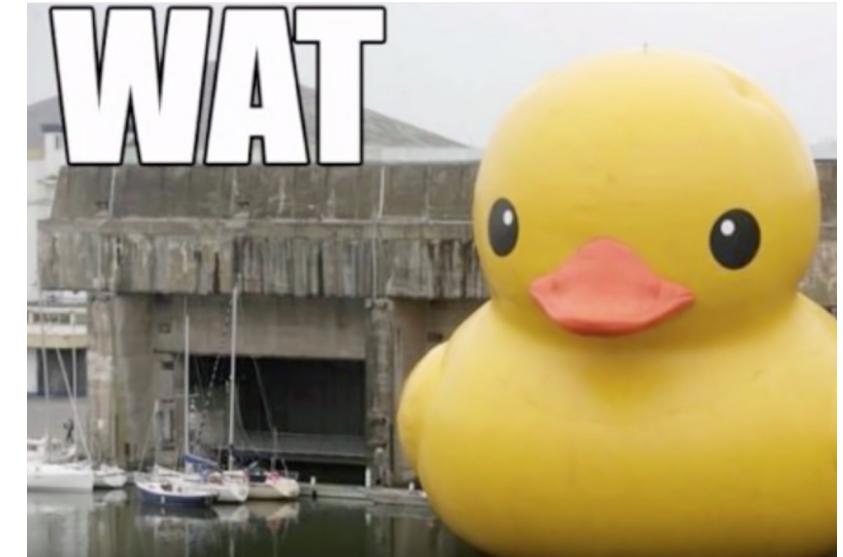
What the internet doesn't tell you

... about Tree Shaking

- Looking at the implementation of lodash's `isString`:

```
function isString(value) {  
  return typeof value == 'string' || ...;  
}  
module.exports = isString;
```

- Turns out, when using ES2015 imports together with CommonJS exports, it results in **importing the whole module instead**
 - No warnings by the IDE!
 - No warnings by the linter!
 - No warnings by the bundler!





How to use Tree Shaking here?

✖ Copy the implementation of `isString` from `lodash` into your codebase.

- `lodash` is structured in a way that uses one js file per method (*cherry-picking syntax*):

```
import isString from 'lodash/isString';      (+0.21 KB)
```

- `lodash` offers package with ES2015 style imports called `lodash-es`:

```
import { isString } from 'lodash-es';        (+0.22 KB)
```



But there must be an easier way!

There is (at least kind of...)

Import Cost plugin for [IntelliJ](#) and [VSCode](#) estimates bundle size impact for imports:

```
1 import { isString } from 'lodash';| 72.94 kB (gzip: 25.32 kB)
2
3   export function foo(input) {
4     return isString(input);
5   }
6
```

But there must be an easier way!



There is (at least kind of...)

Bundlephobia will tell you which packages are tree-shakeable:

BUNDLEPHOBIA

lodash-es@4.17.21

tree-shakeable side-effect free npm

BUNDLEPHOBIA

lodash@4.17.21

npm

BUNDLE SIZE

98 kB

MINIFIED

30.6 kB

MINIFIED +
GZIPPED

BUNDLE SIZE

69.8 kB

MINIFIED

24.4 kB

MINIFIED + GZIPPED



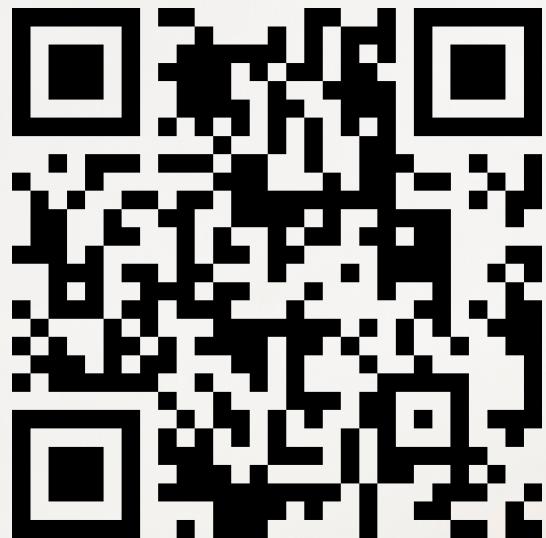
Pull Request Time!

- „sure, you want to introduce lodash just because of `isString`? Thats a big fish!“
- **Difficult to predict by eye how (and if) bundlers will tree-shake**
- Using Import cost plugin and Bundlephobia can help make decisions quicker
- As with performance optimization in general:
 - **The only way to know for sure is by measuring!**
- Don't be afraid to use dependencies
 - But make sure they are tree-shaked!
 - Consider submitting a PR to the dependency if open source

Tree Shaking!



Slides:



<https://fm.ht/not25>

