
Chérie, j'ai rétréci
mes bundles JS
We ❤️ Speed 2019

Antoine Kahn-Dubois

Lead Developper chez Theodo

github.com/antkahn

Au menu

- Pourquoi diminuer la taille de son JavaScript ?
- Comment mettre en place le Code Splitting
- Comment choisir ses librairies ?
- Pour aller plus loin: le Bundle Splitting

Au menu

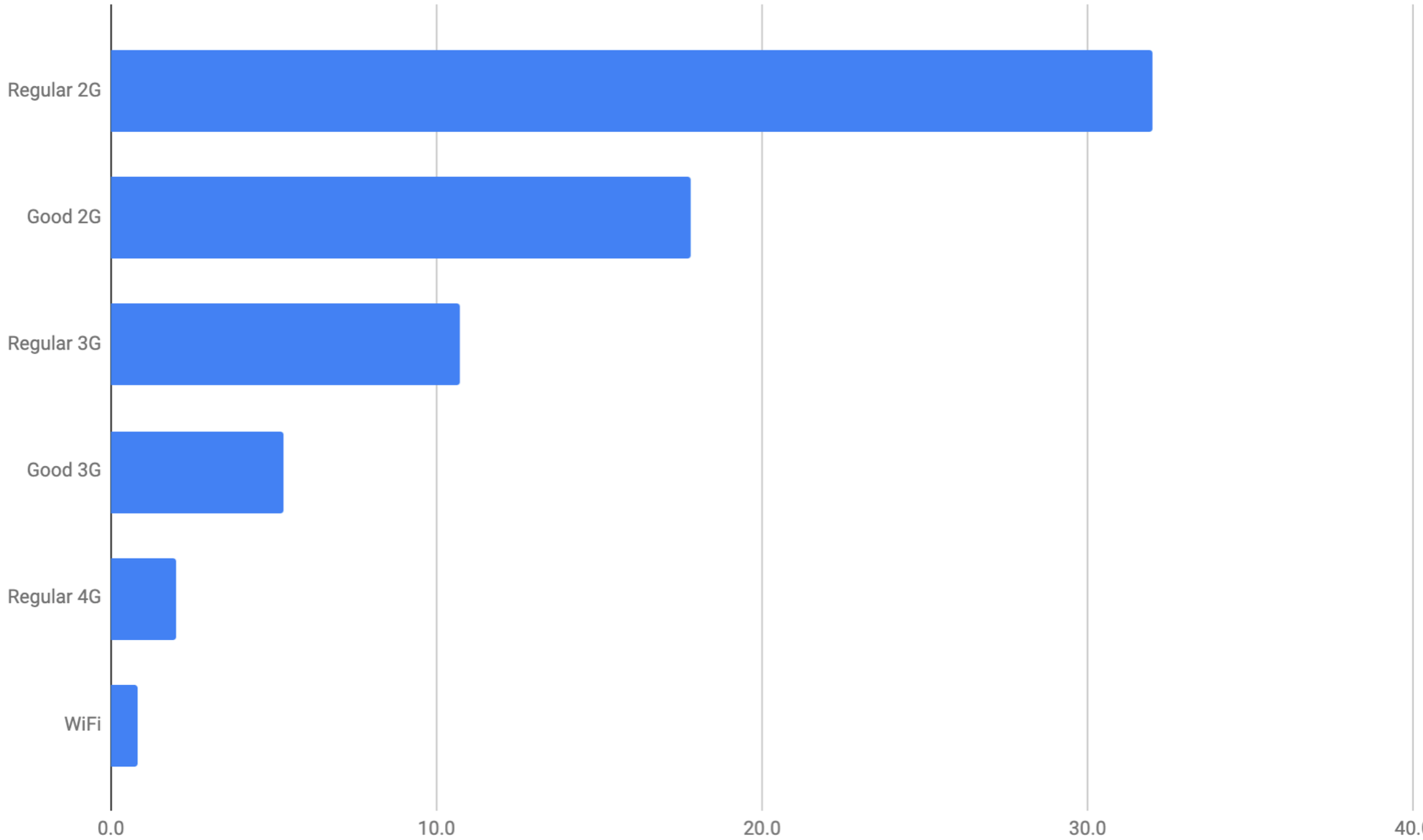
- Pourquoi diminuer la taille de son JavaScript ?
- Comment mettre en place le Code Splitting
- Comment choisir ses librairies ?
- Pour aller plus loin: le Bundle Splitting

Pourquoi réduire la taille de son JavaScript ?

Réduire le temps de:

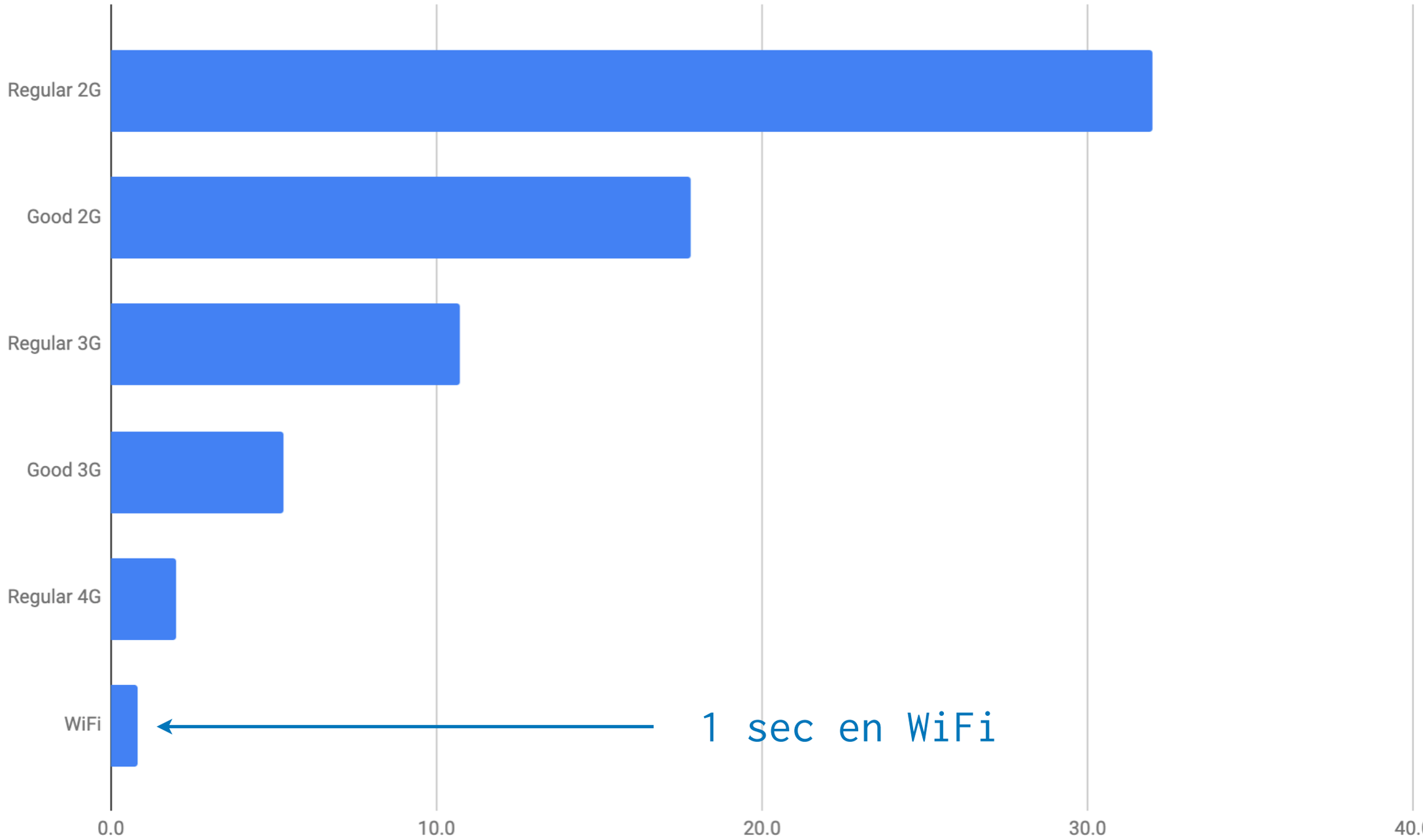
- Téléchargement

Temps pour télécharger 1MB de JavaScript



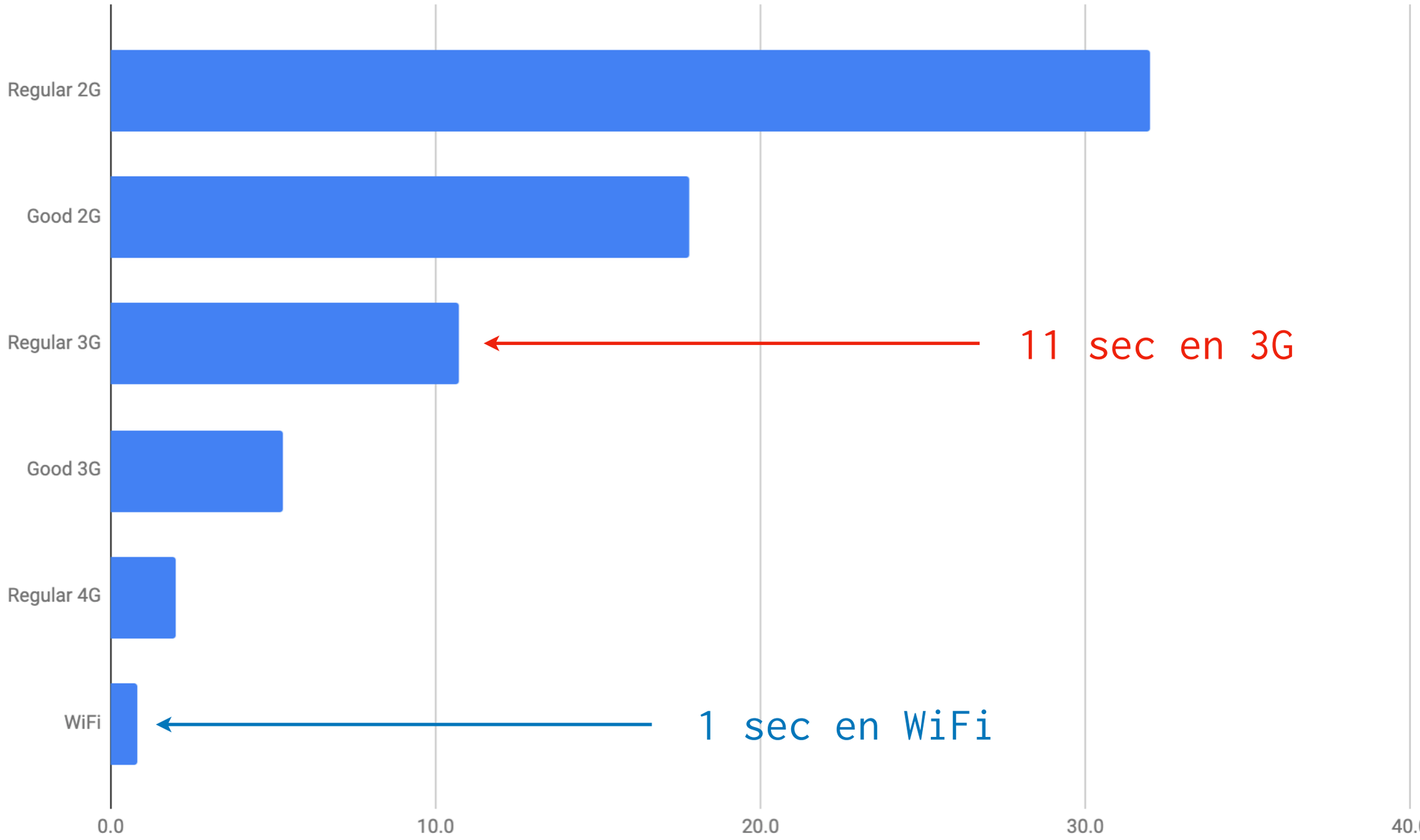
Source: ma calculette

Temps pour télécharger 1MB de JavaScript



Source: ma calculette

Temps pour télécharger 1MB de JavaScript



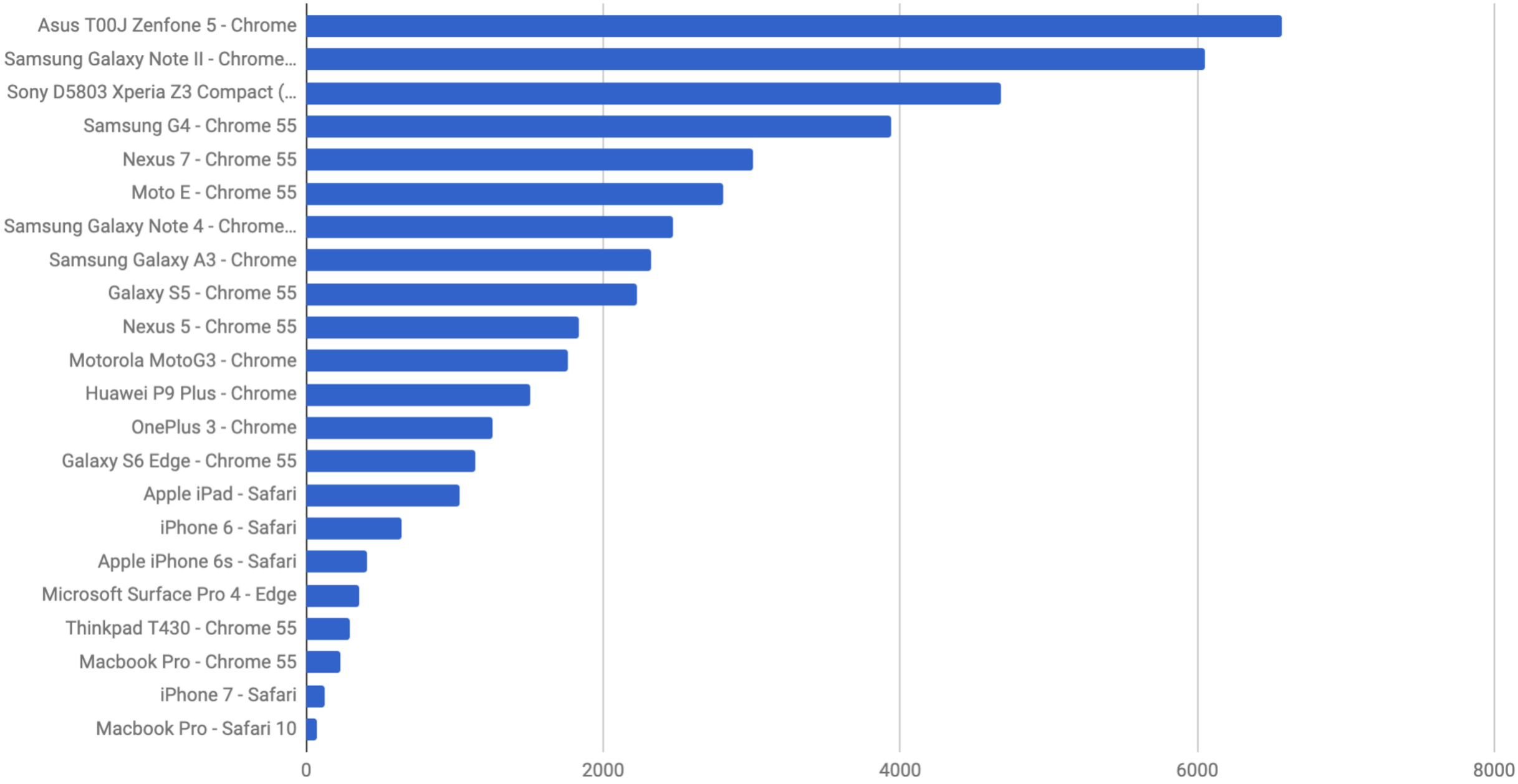
Source: ma calculette

Pourquoi réduire la taille de son JavaScript ?

Réduire le temps de:

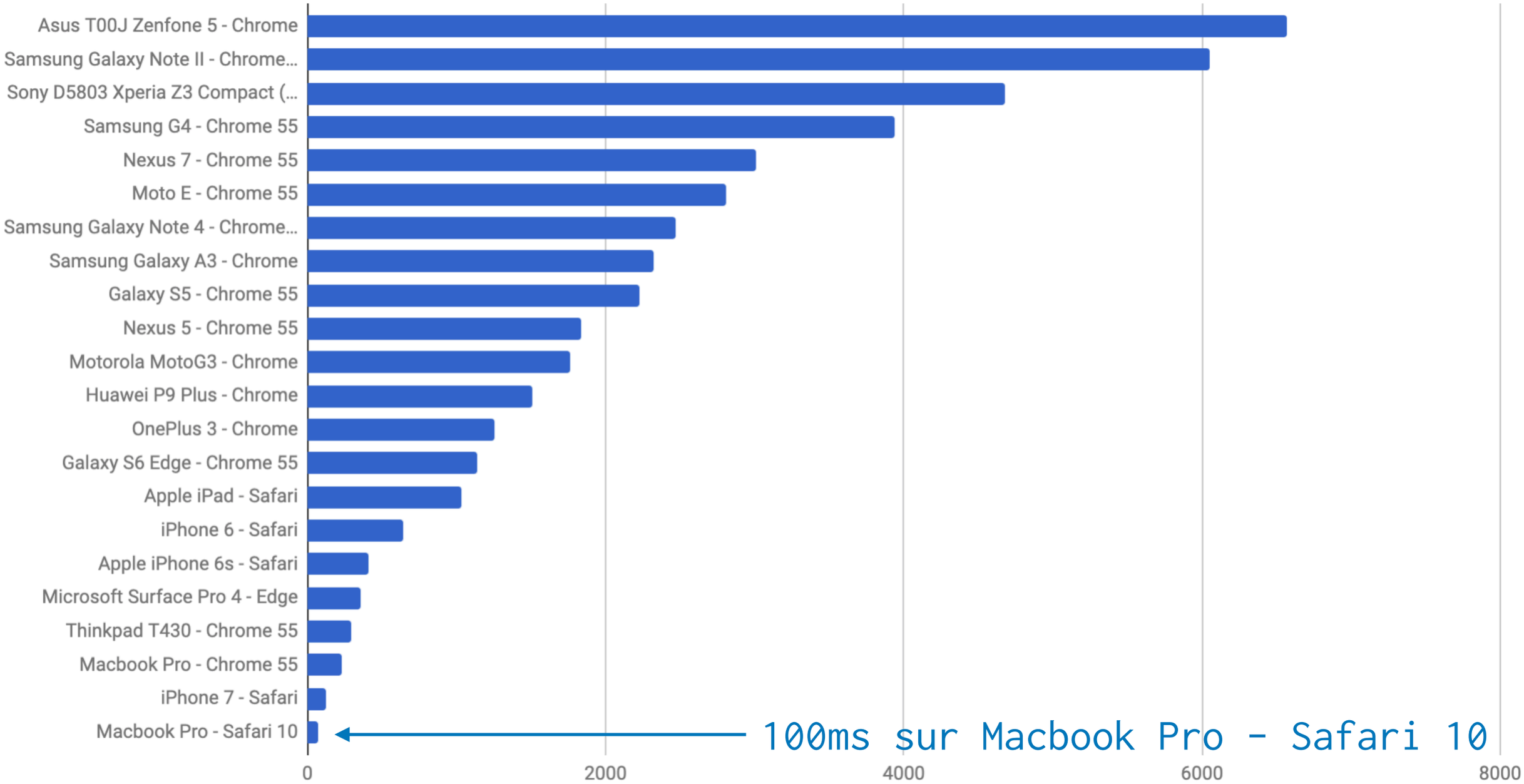
- Téléchargement
- Parsing

Temps pour parser 1MB de JavaScript



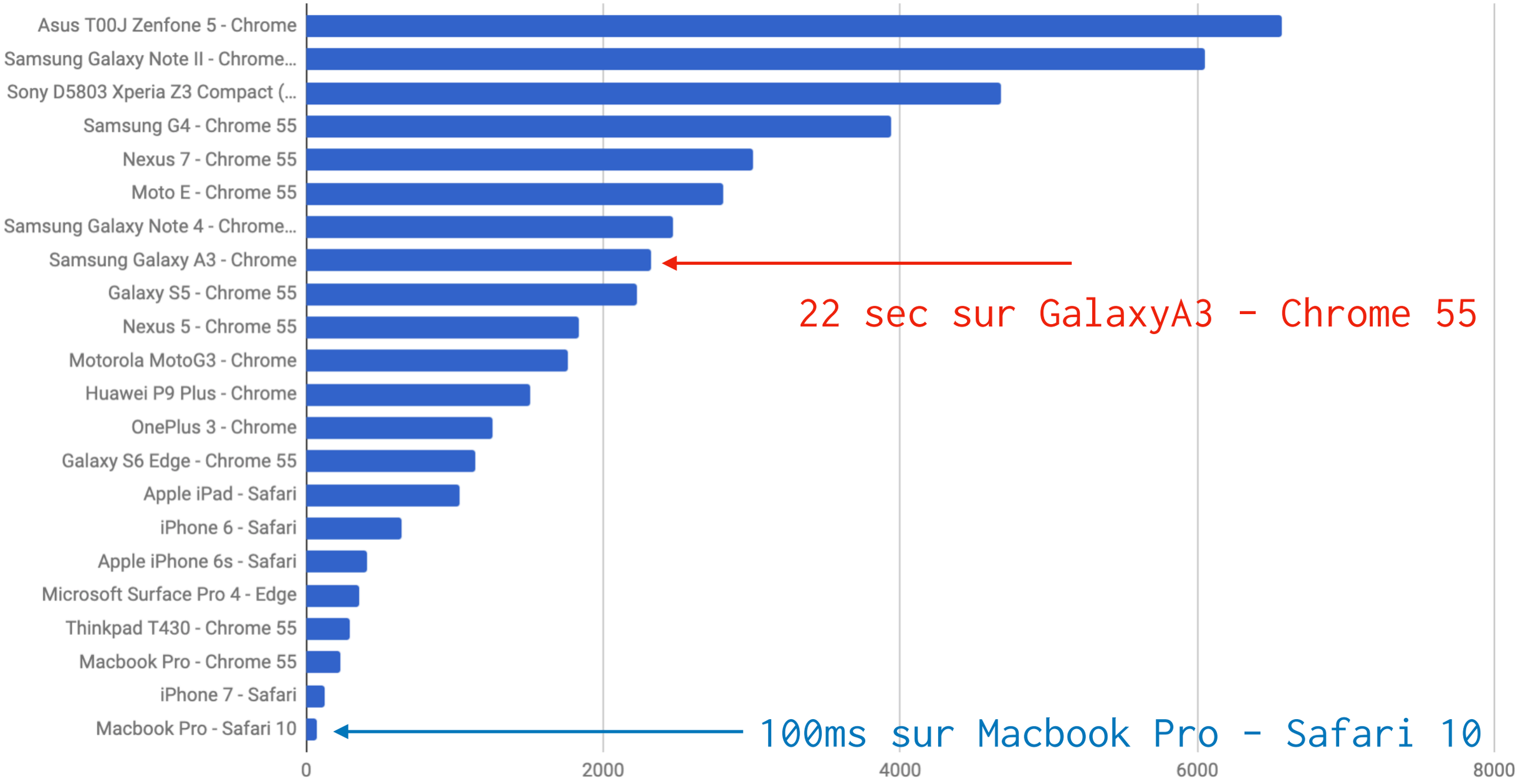
Source: @addyosmani, cost of JavaScript 2018

Temps pour parser 1MB de JavaScript



Source: @addyosmani, cost of JavaScript 2018

Temps pour parser 1MB de JavaScript



Source: @addyosmani, cost of JavaScript 2018

Pourquoi réduire la taille de son JavaScript ?

Réduire le temps de:

- Téléchargement
- Parsing
- Compilation & Exécution

Pourquoi réduire la taille de son JavaScript ?

Sauver la planète !

On va consommer moins d'énergie pour:

- Déplacer les fichiers du serveur au client
- Parser les fichiers chez le client

Pourquoi réduire la taille de son JavaScript ?

Taille conseillée: < 200KB Gzippé

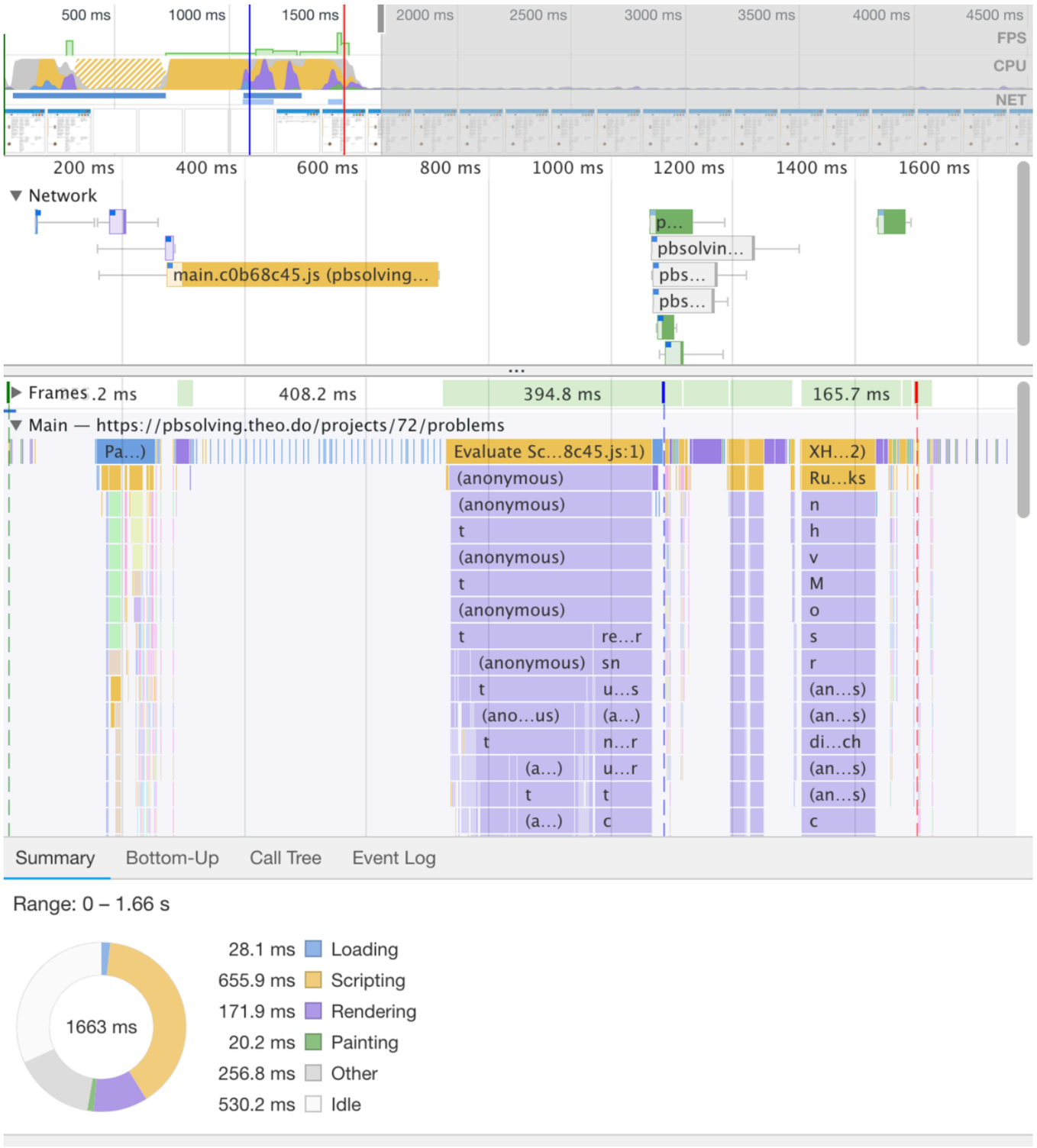
Au menu

- Pourquoi diminuer la taille de son JavaScript ?
- Comment mettre en place le Code Splitting
- Comment choisir ses librairies ?
- Pour aller plus loin: le Bundle Splitting

Application de test

- Single Page Application
- En React/Redux
- Une page de login
- Une page avec un tableau de data et des modales

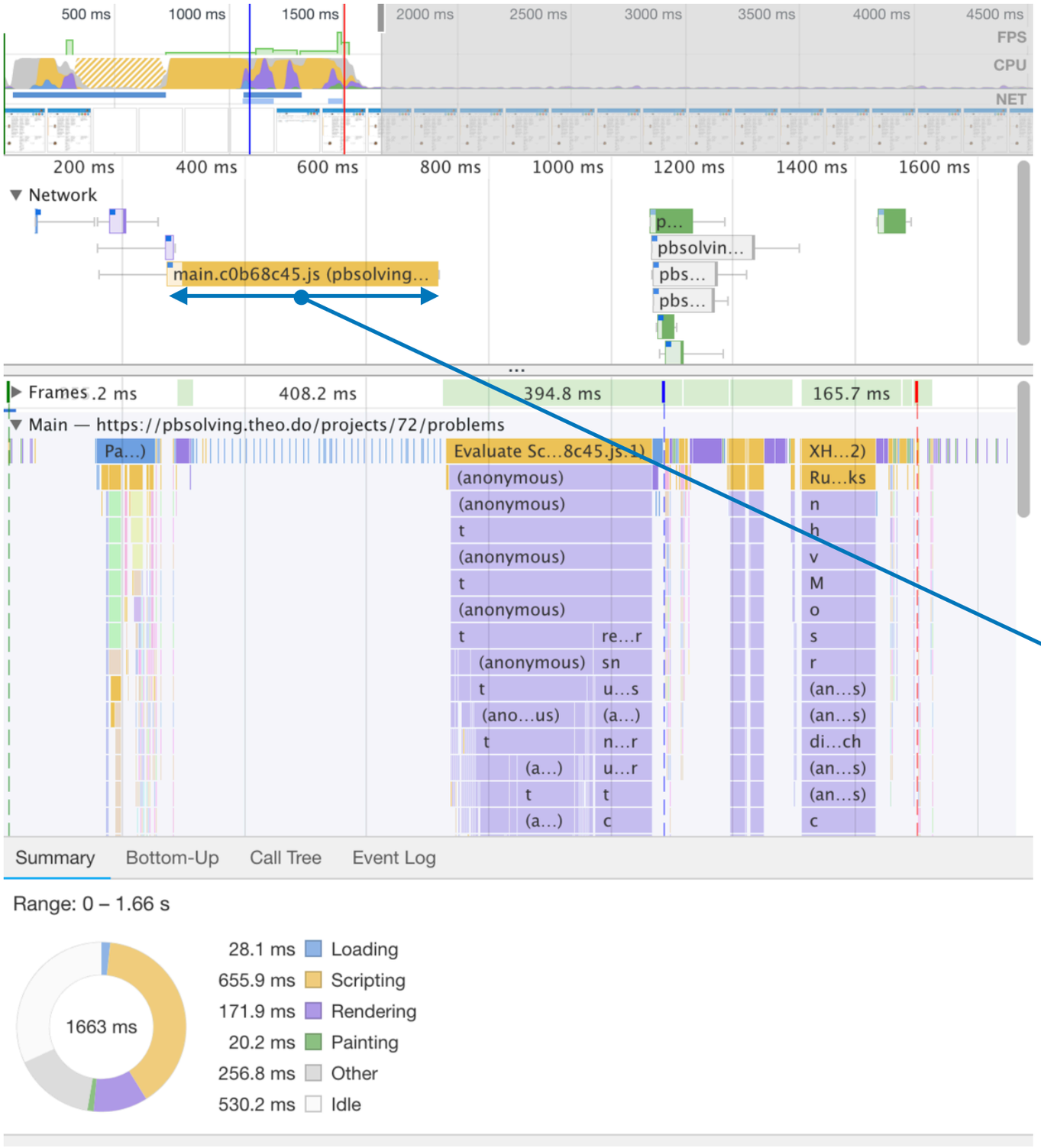
État initial (page principale)



Temps de chargement: 1.3s

Sur Chrome, Macbook Pro

État initial (page principale)



500ms Téléchargement

État initial (page principale)

350ms Parsing, compilation & execution

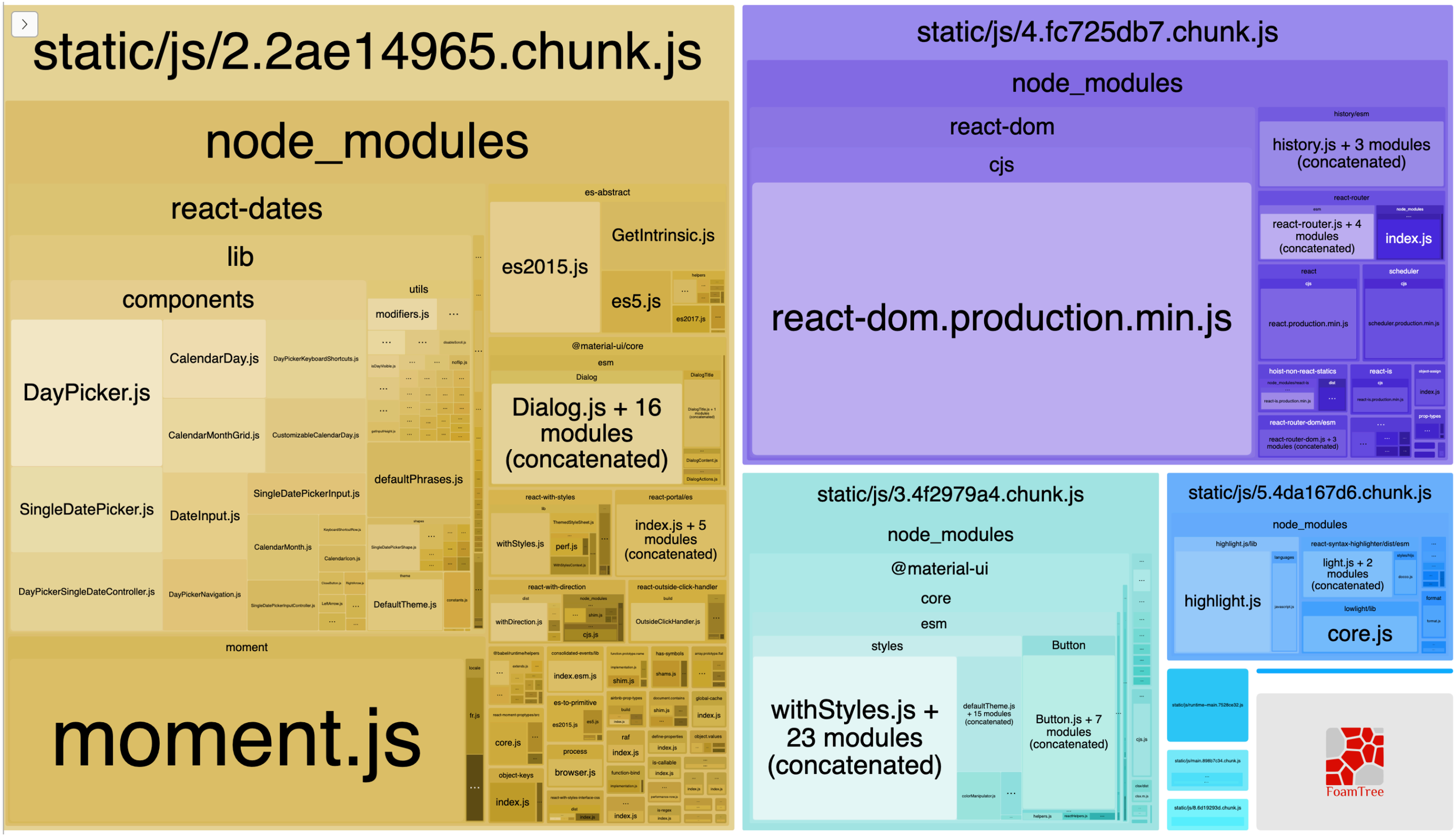
500ms Téléchargement

Qu’y a-t-il dans mon Bundle ?

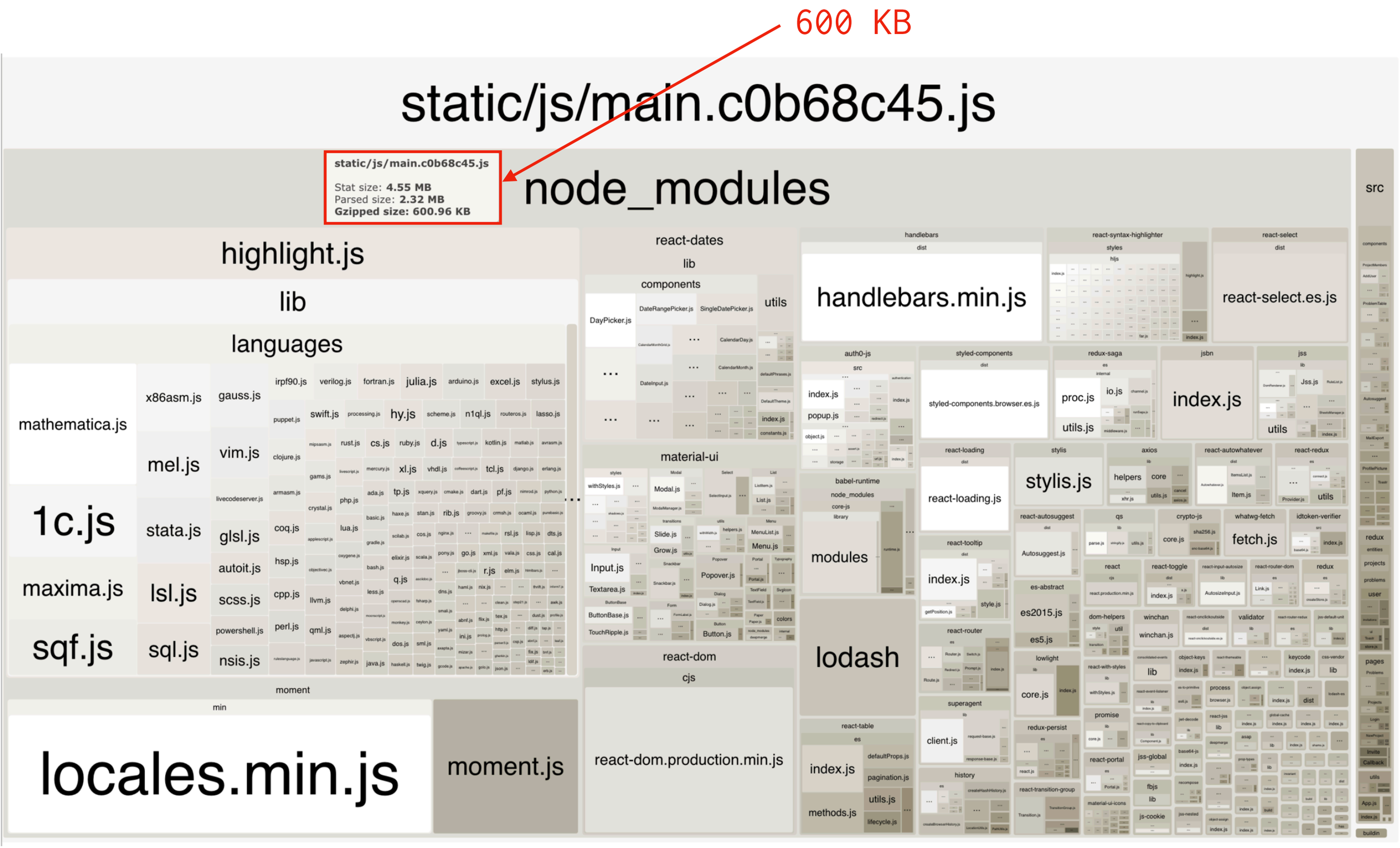
Qu’y a-t-il dans mon bundle ?

L’outil idéal: webpack-bundle-analyzer
github.com/webpack-contrib/webpack-bundle-analyzer

C’est quoi webpack-bundle-analyzer ?



Ce qu'il y a dans mon bundle

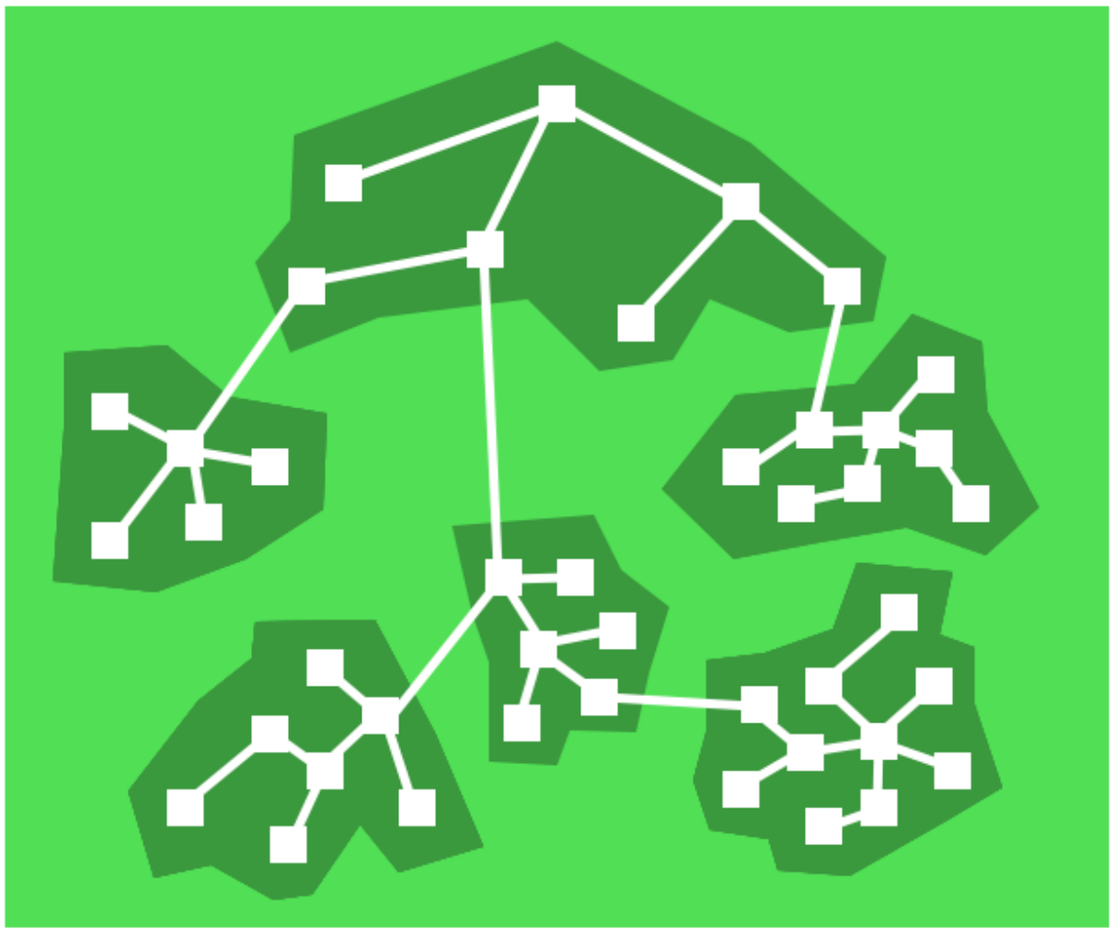


Code Splitting

Le but: apporter uniquement à l'utilisateur ce dont il a besoin, quand il en a besoin.

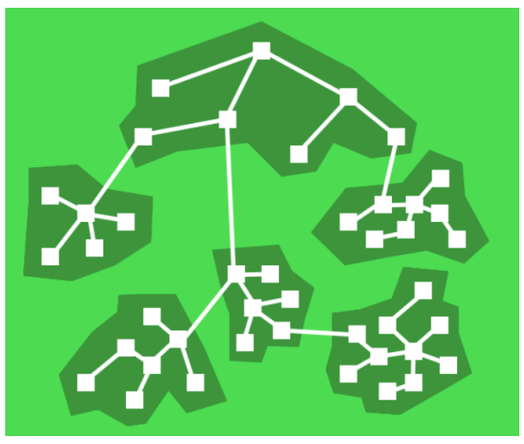
On va découper notre JavaScript en plusieurs chunks, et lorsque l'utilisateur voudra faire une action, on lui enverra seulement le JavaScript nécessaire.

Code Splitting



Source: github.com/jamiebuilds/react-loadable

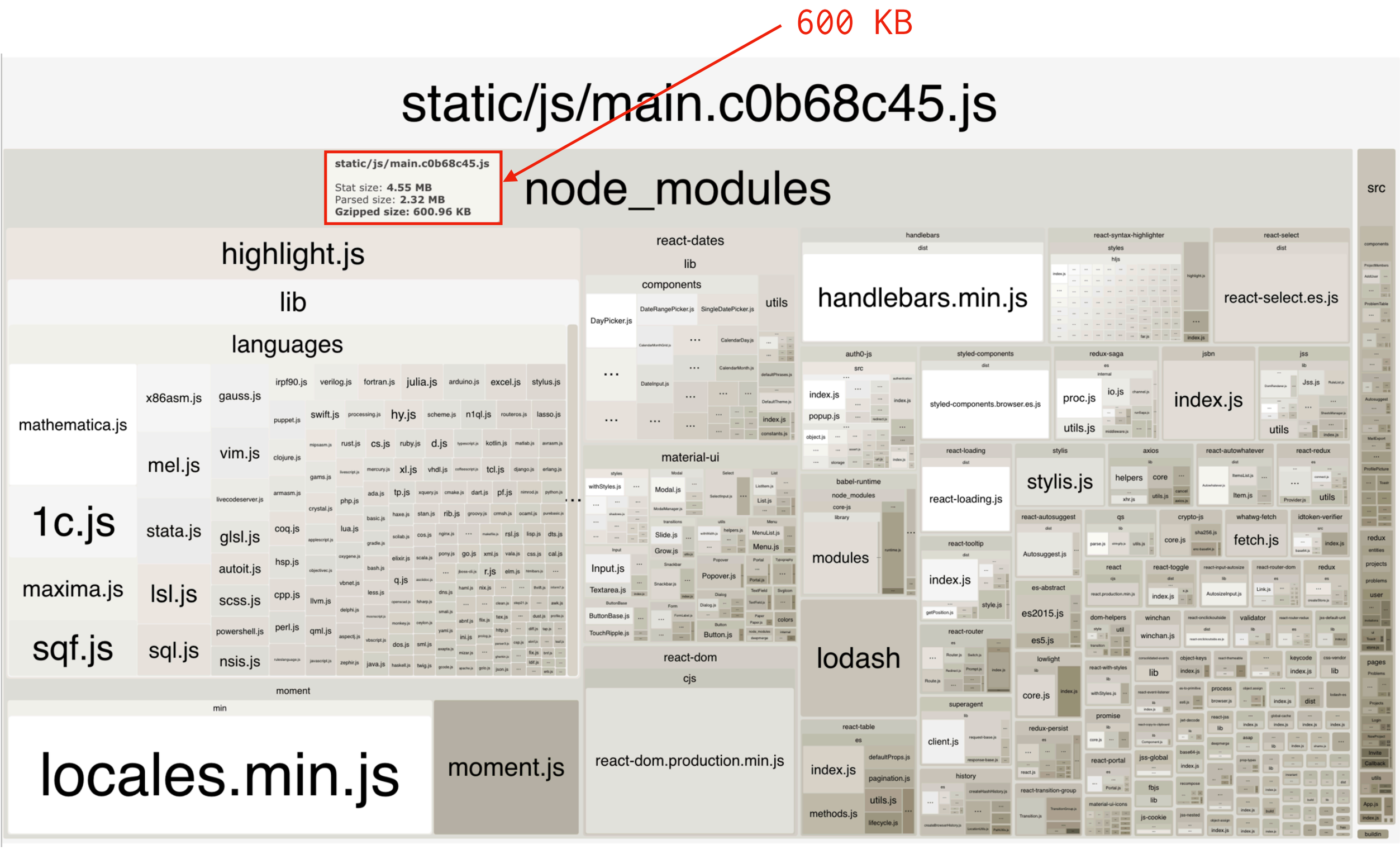
Code Splitting

import	**import()**
Statique	Dynamique
Synchrone	Asynchrone
Concatène les fichiers	Crée des chunks
	

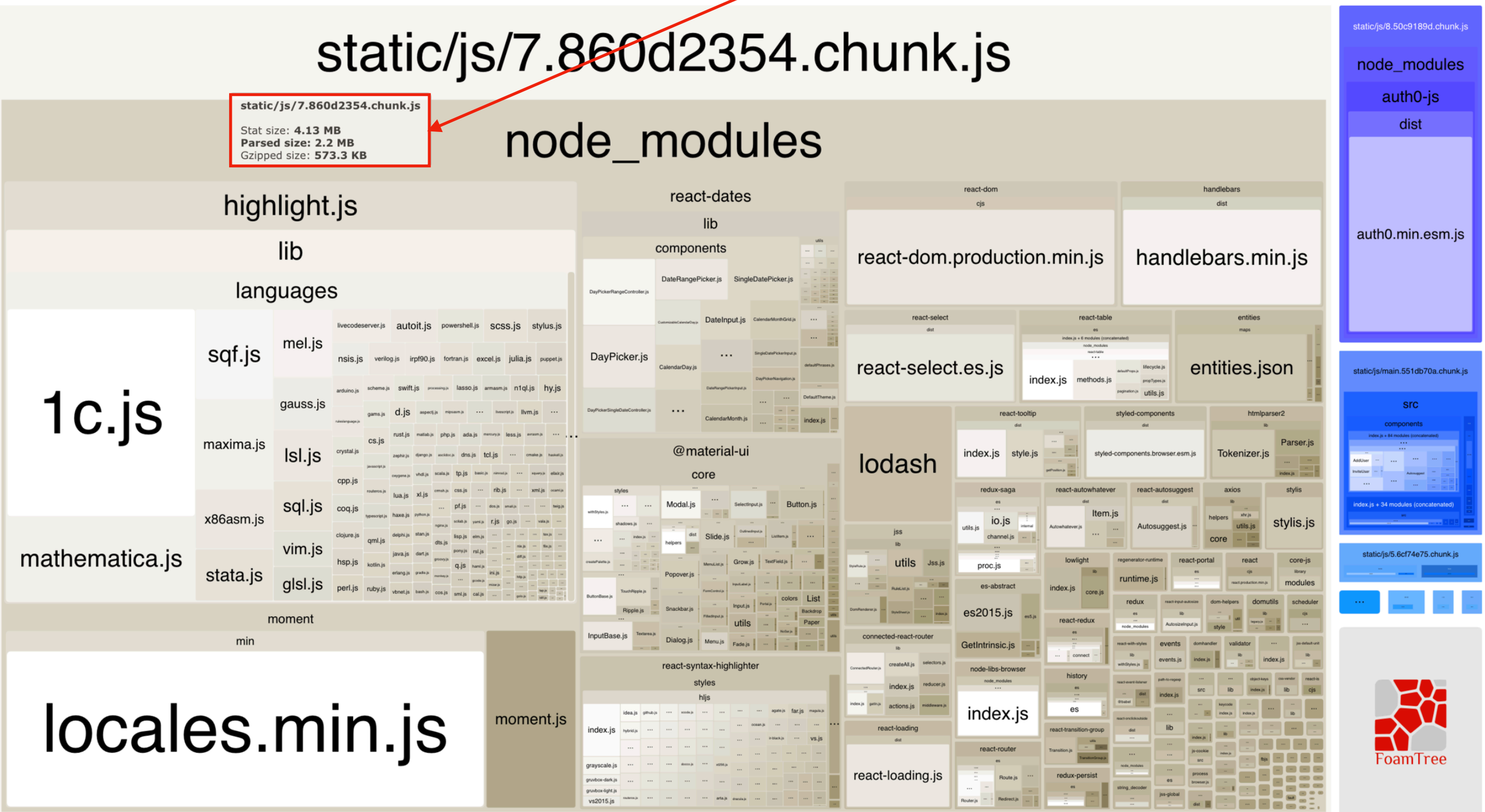
Code Splitting par page

```
const Callback = lazy(() => import('./pages/Callback'));
const Invite = lazy(() => import('./pages/Invite'));
const Login = lazy(() => import('./pages/Login'));
const NewProject = lazy(() => import('./pages/NewProject'));
const Problems = lazy(() => import('./pages/Problems'));
const Projects = lazy(() => import('./pages/Projects'));

const history = createBrowserHistory();
const { persistor, store } = configureStore(history);
@@ -29,7 +28,7 @@ const App = () => (
  <Provider store={store}>
    <PersistGate loading={null} persistor={persistor}>
      <ConnectedRouter history={history}>
-       <Fragment>
+       <Suspense fallback={<div>Loading...</div>}>
          <Toastr />
          <Header />
          <Switch>
@@ -49,7 +48,7 @@ const App = () => (
            <Route exact path="/login" component={Login} />
            <Route component={Login} />
          </Switch>
-       </Fragment>
+       </Suspense>
```

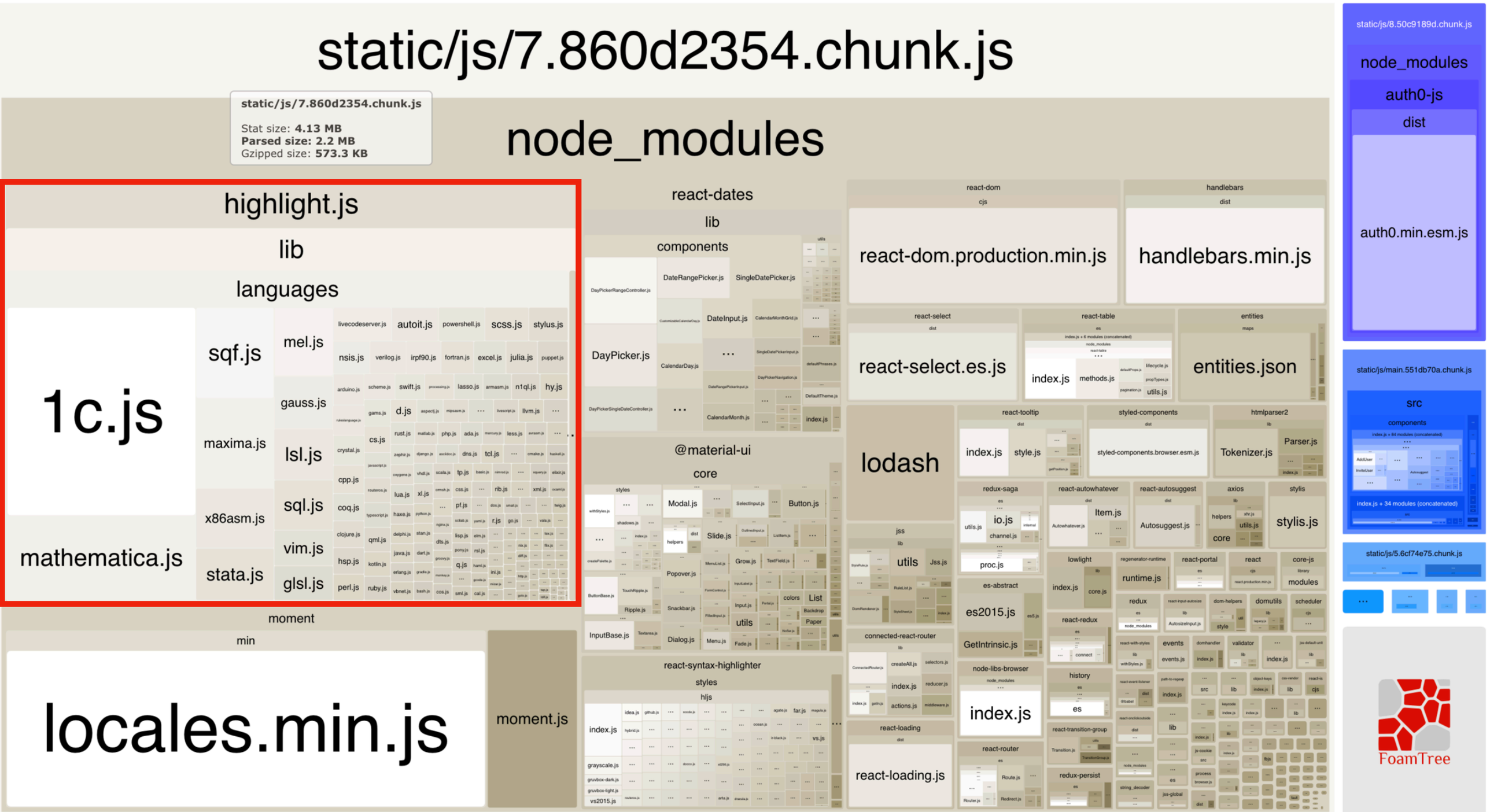
Avant Code Splitting par Page



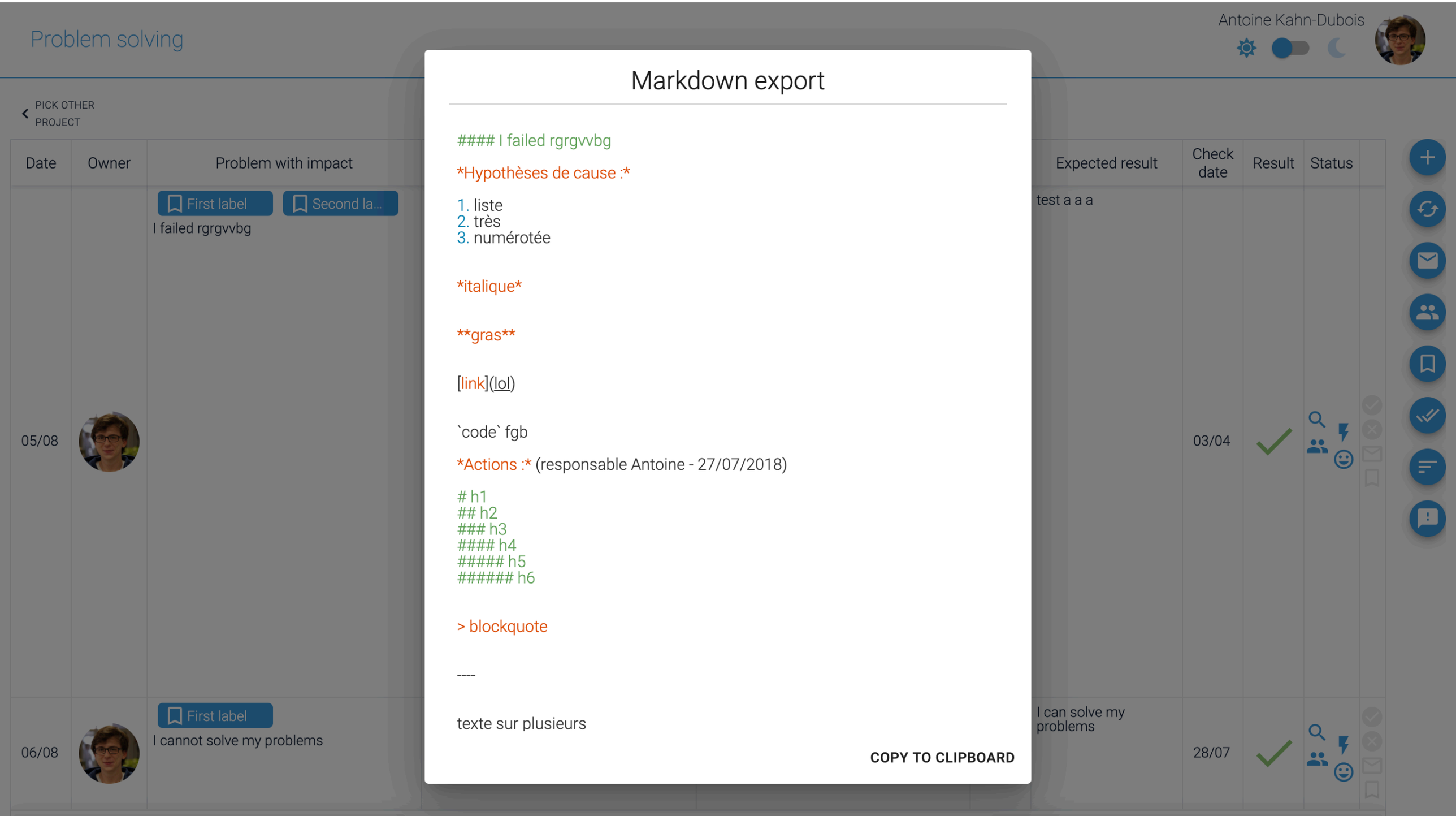
Après Code Splitting par Page



highlight.js



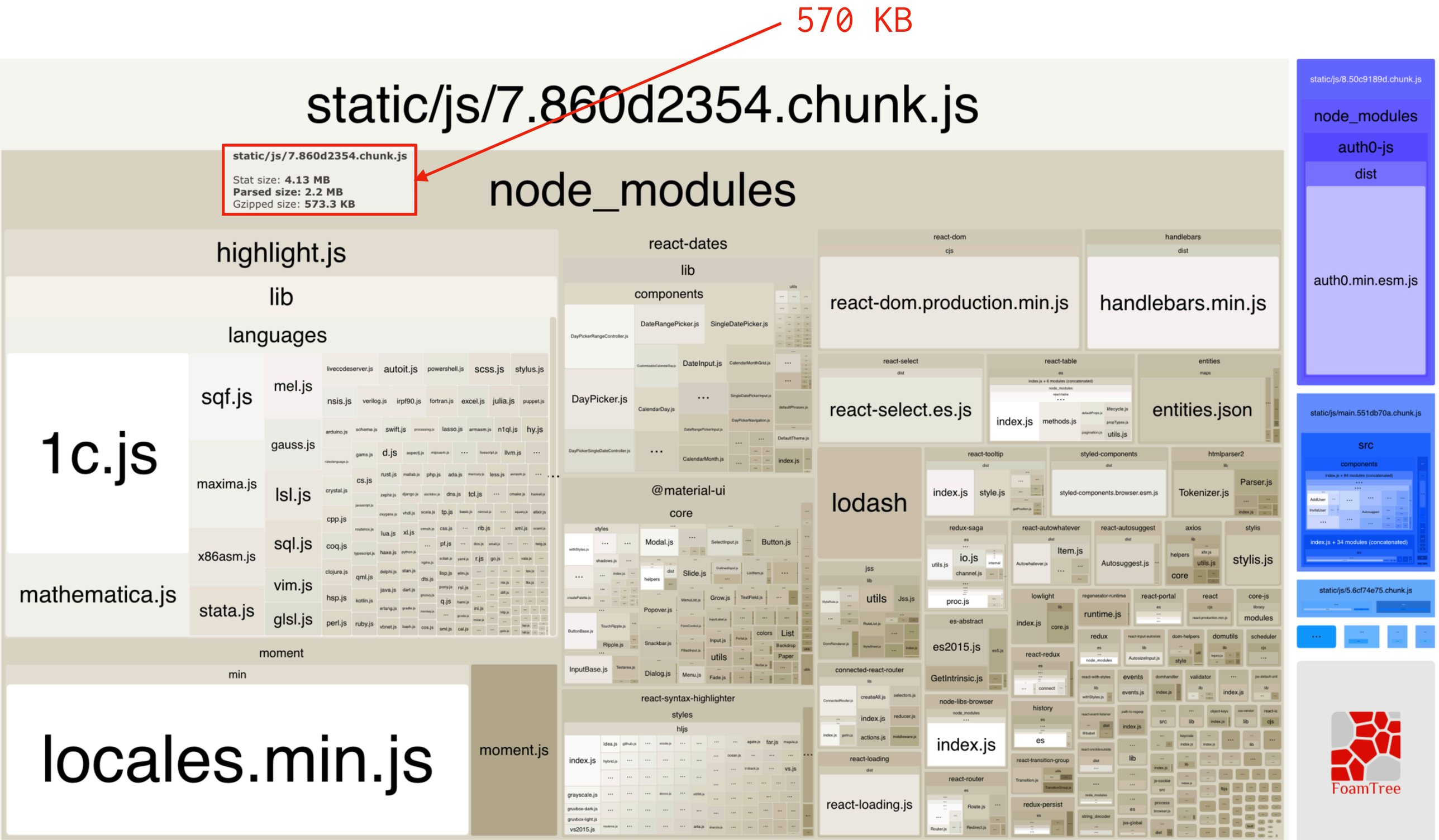
highlight.js n’est utilisé que dans une modale



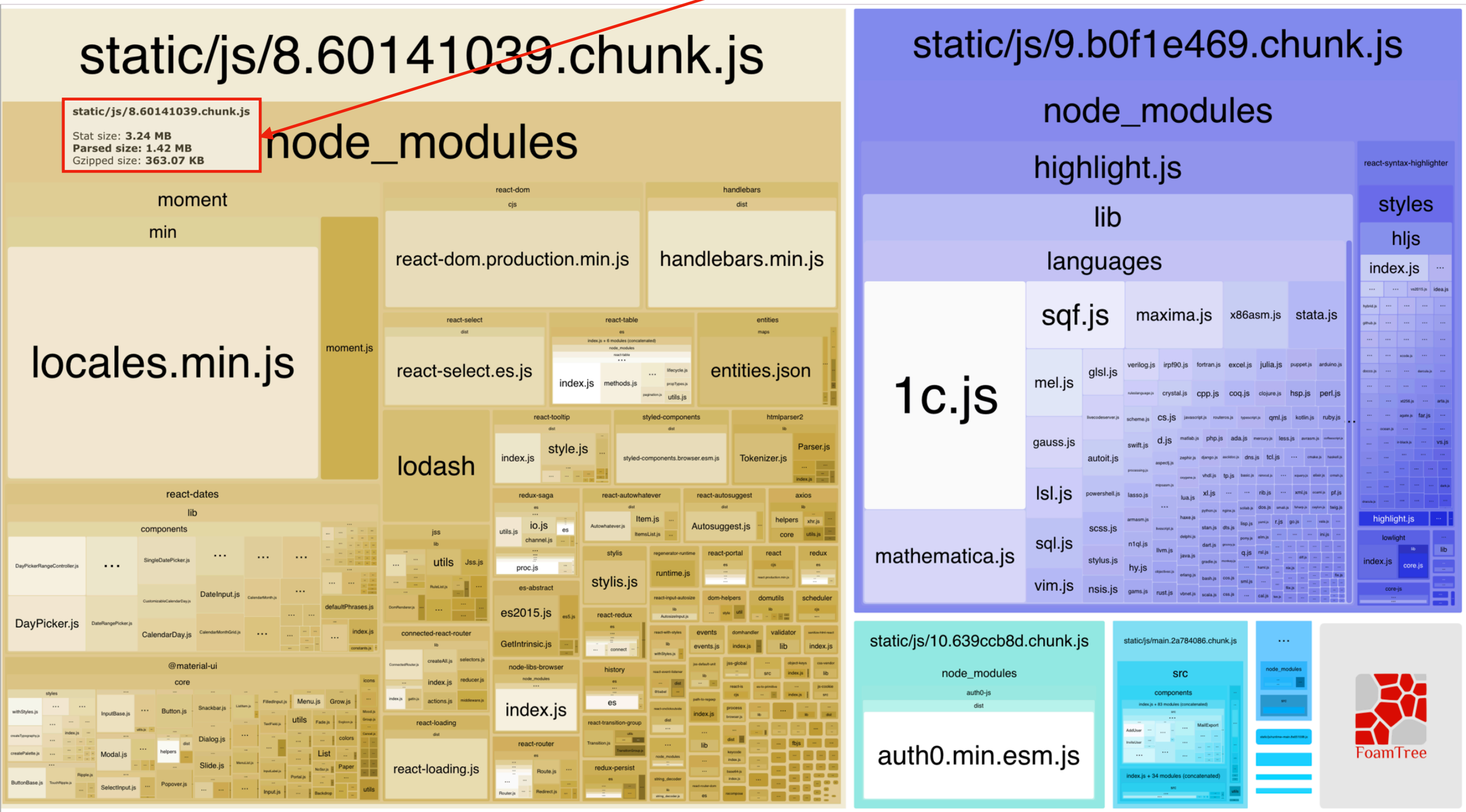
On Code Split la modale

```
const DailyMailExportModal = lazy(() => import('../DailyMailExportModal'));

type PropsType = {
  problem: ProblemType,
@@ -93,11 +95,15 @@ class ProblemTableCellAction extends Component<PropsType, St
          onClick={this.handleOpen}
        />
        <ReactTooltip place="left" type="info" effect="solid" id="paste" />
        <DailyMailExportModal
          problem={problem}
          onClose={this.handleClose}
          open={this.state.isModalOpened}
        />
        {this.state.isModalOpened && (
          <Suspense fallback={null}>
            <DailyMailExportModal
              problem={problem}
              onClose={this.handleClose}
              open={this.state.isModalOpened}
            />
          </Suspense>
        )}
```

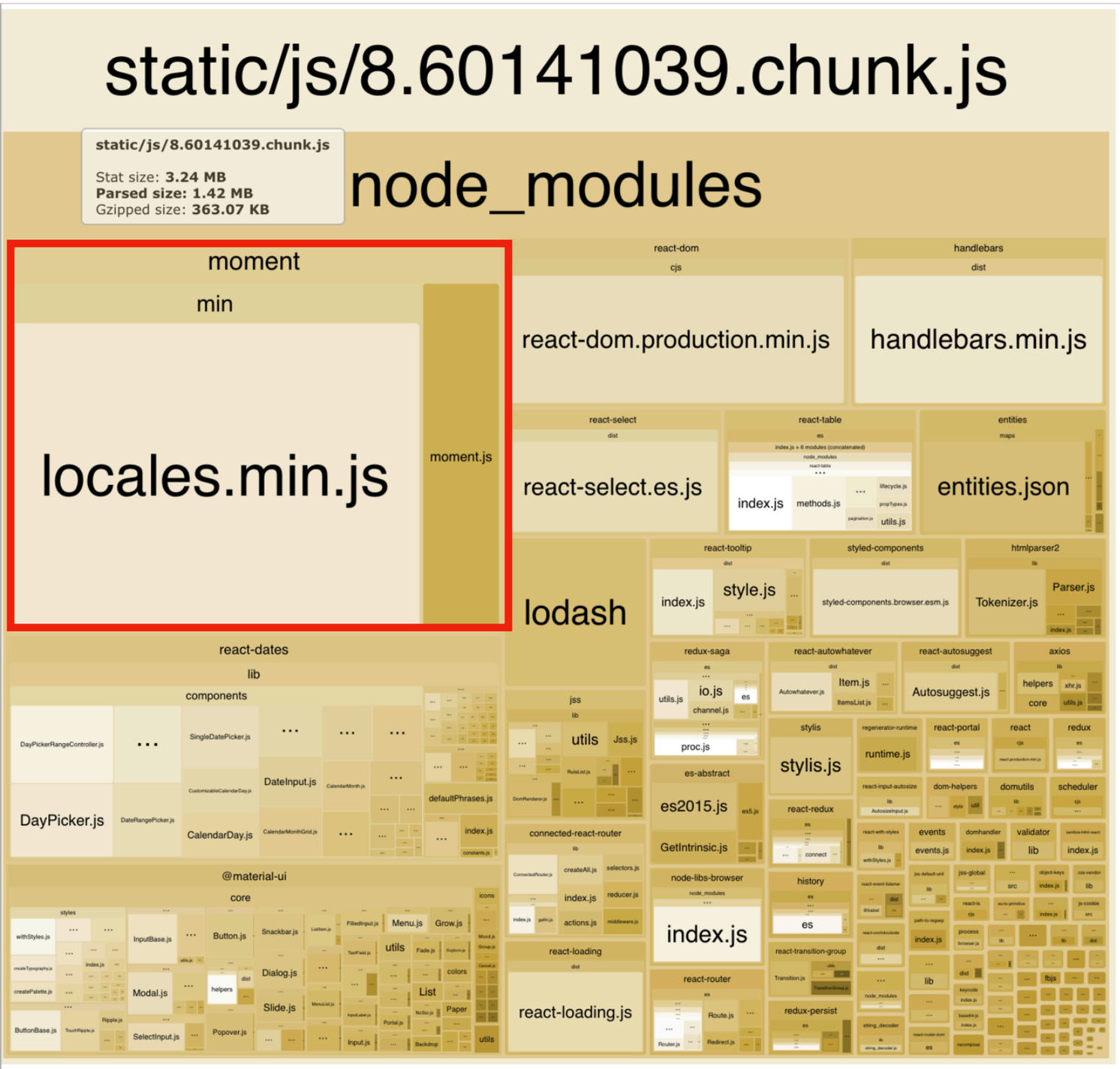
Avant le code split de la modale



Après le code split de la modale



moment / highlight

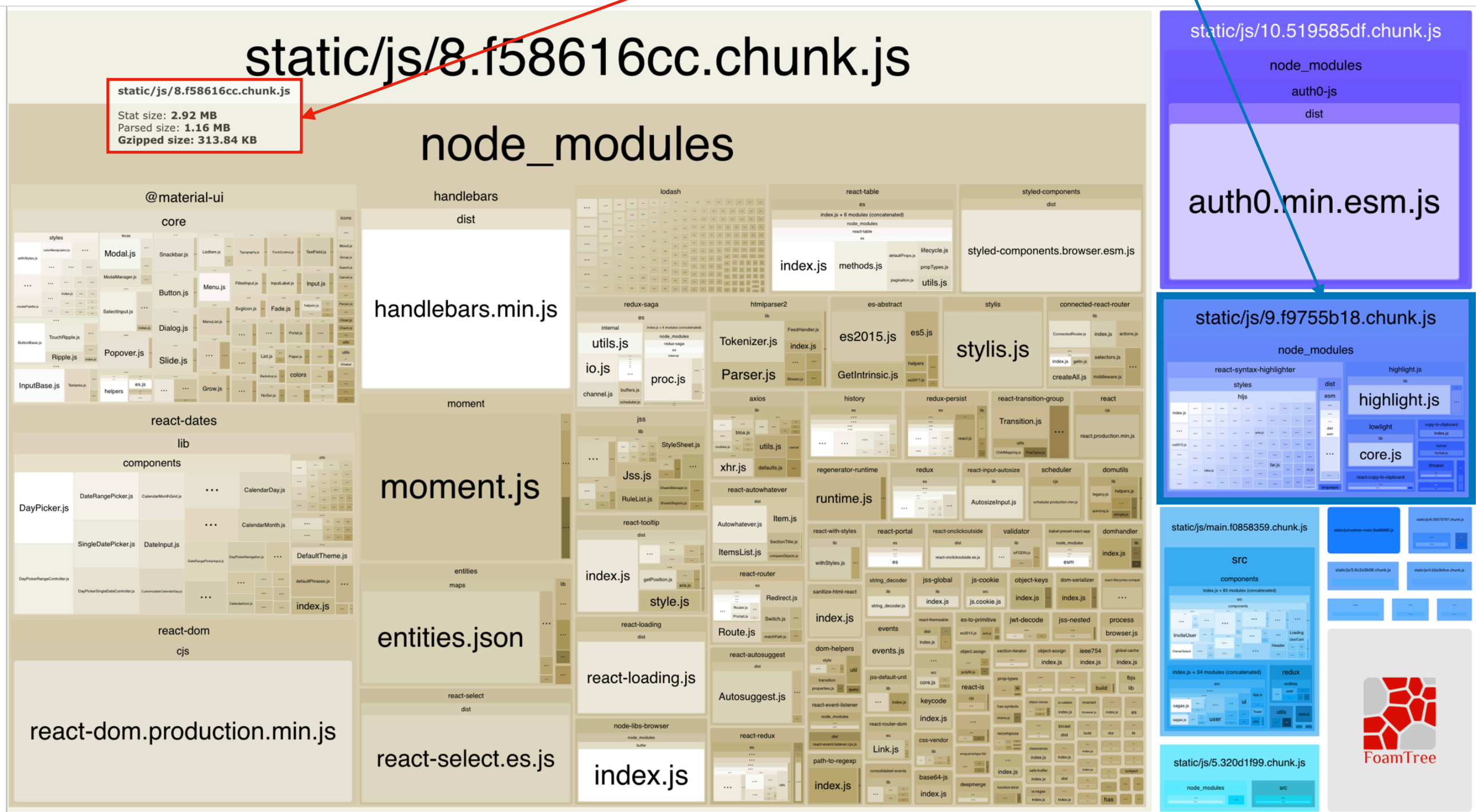


moment / highlight

Les grosses librairies contiennent beaucoup de fichiers spécifiques.

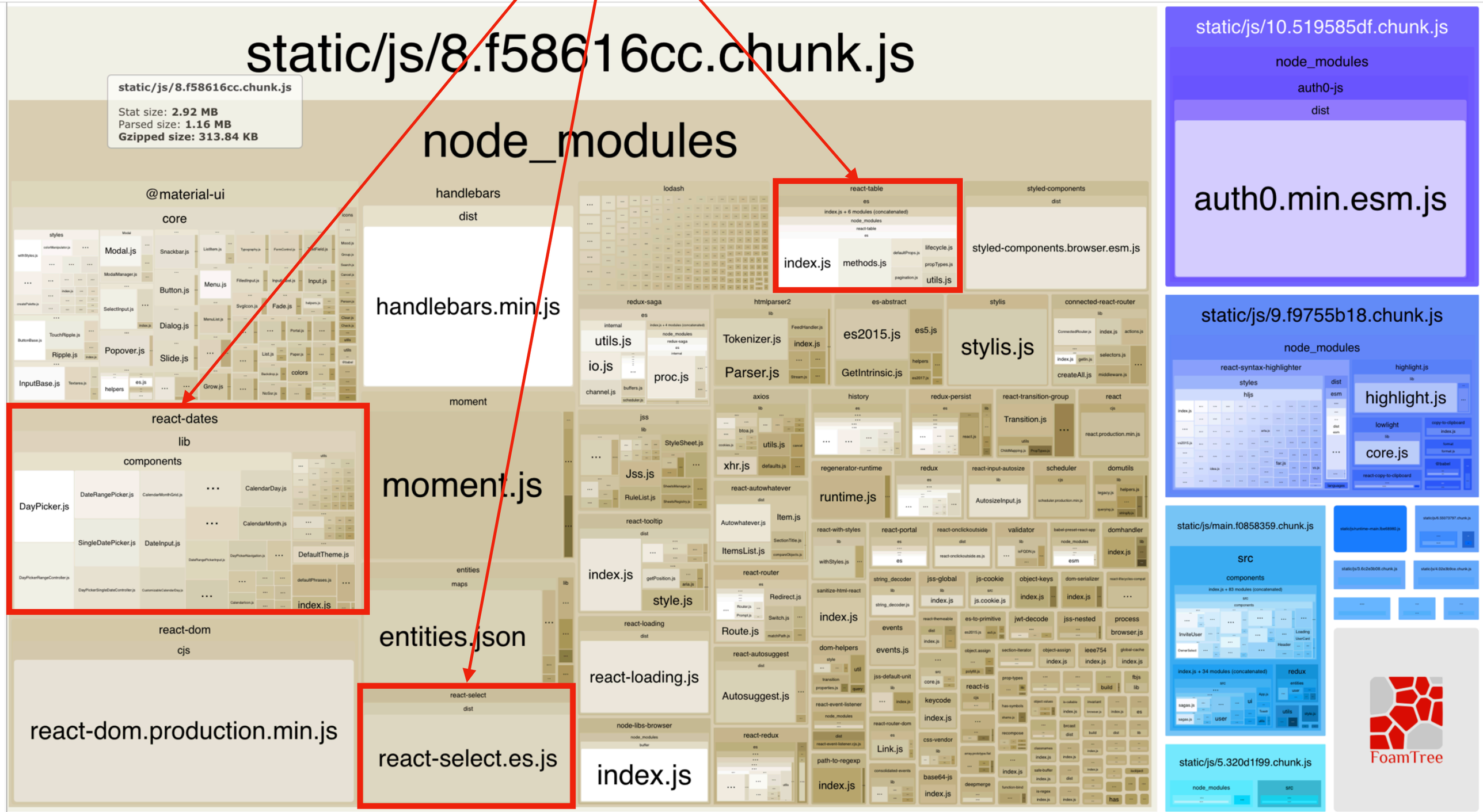
On importe seulement de ces librairies les bouts qui nous intéressent. Pour moment, seulement le français et l’anglais; pour highlight, seulement le markdown.

Après l’allégement de moment/highlight



Bizarre…

Librairies utilisées sur une seule page



Lorsque j’exporte mes composants en global…

Tree: 9edae94503 ▾ **problem-solving** / client / src / components / **index.js**

 **antkahn** allow user to change his mail export

2 contributors  

19 lines (17 sloc) | 869 Bytes

```
1   // @flow
2
3   export { default as Autosuggest } from './Autosuggest';
4   export { default as Header } from './Header';
5   export { default as Loading } from './Loading';
6   export { default as MailExport } from './MailExport';
7   export { default as PageContent } from './PageContent';
8   export { default as PrivateRoute } from './PrivateRoute';
9   export { default as ProblemTable } from './ProblemTable';
10  export { default as ProblemTableCell } from './ProblemTableCell';
11  export {
12    default as ProblemTableCellDatePicker,
13  } from './ProblemTableCellDatePicker';
14  export { default as ProblemTableCellOwner } from './ProblemTableCellOwner';
15  export { default as ProblemTableCellResult } from './ProblemTableCellResult';
16  export { default as ProfilePicture } from './ProfilePicture';
17  export { default as ProjectMembers } from './ProjectMembers';
18  export { default as UserCard } from './UserCard';
```

J’importe mes composants en global !

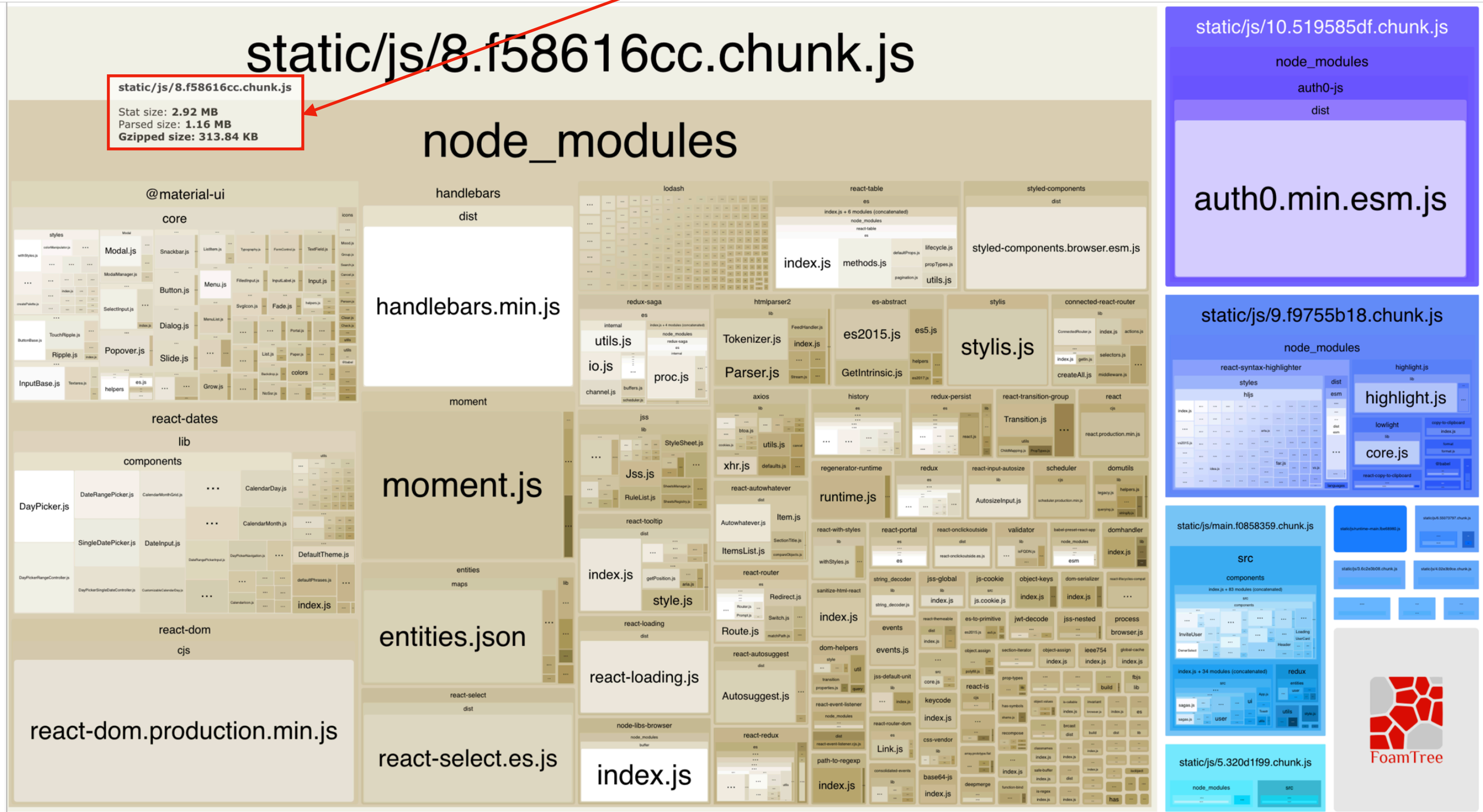
```
// @flow

import React, { Component } from 'react';

import type { Match } from 'react-router-dom';
import type { Action } from '../../types/redux';
import { Loading } from '../../components';

export type PropsType = {
  inviteUser: (token: string) => Action,
  match: Match,
  push: (pathname: string) => Action,
};

export default class Invite extends Component<PropsType> {
  props: PropsType;
```

J’importe seulement les composants dont j’ai besoin

2 client/src/pages/Invite/Invite.component.js

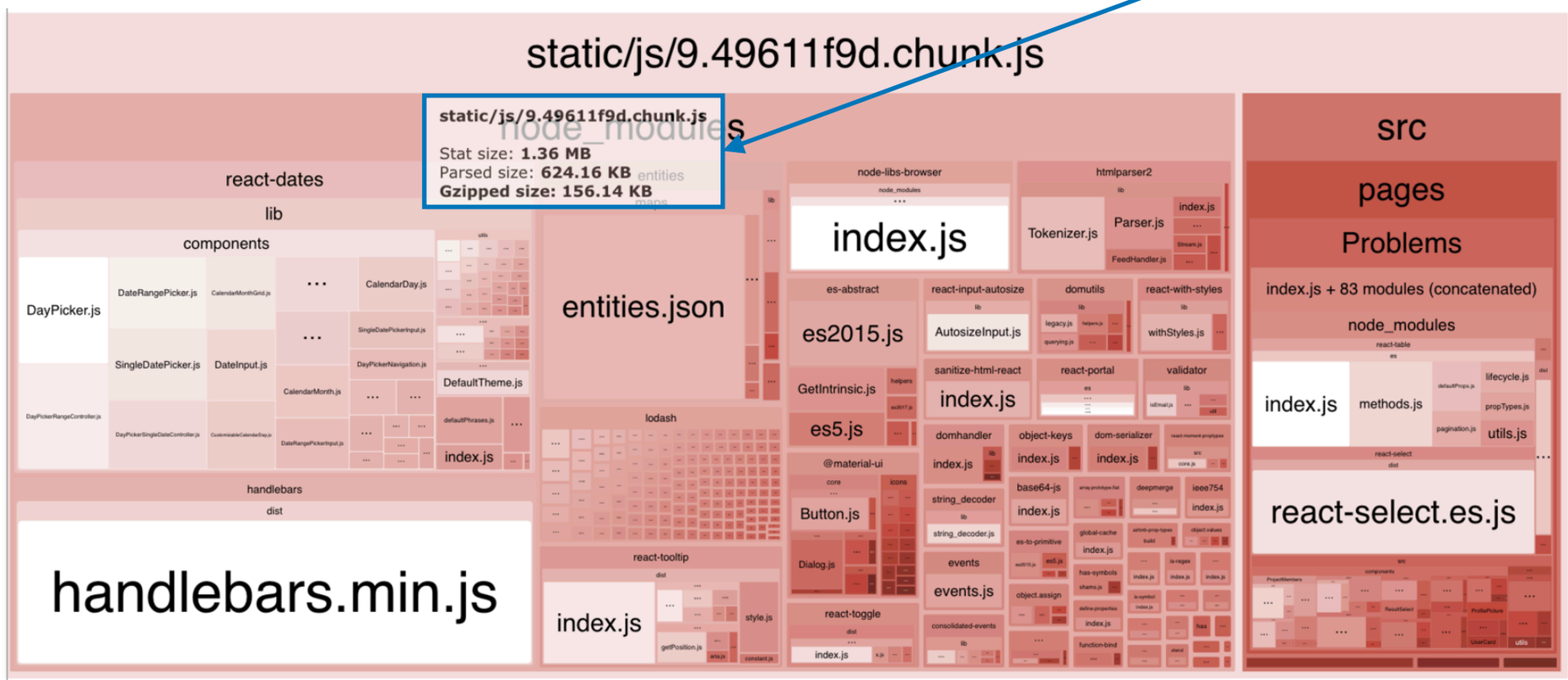
```
@@ -4,7 +4,7 @@ import React, { Component } from 'react';
 
 
   import type { Match } from 'react-router-dom';
   import type { Action } from '../../types/redux';
- import { Loading } from '../../components';
+ import Loading from '../../components/Loading';
 
   export type PropsType = {
     inviteUser: (token: string) => Action,
```

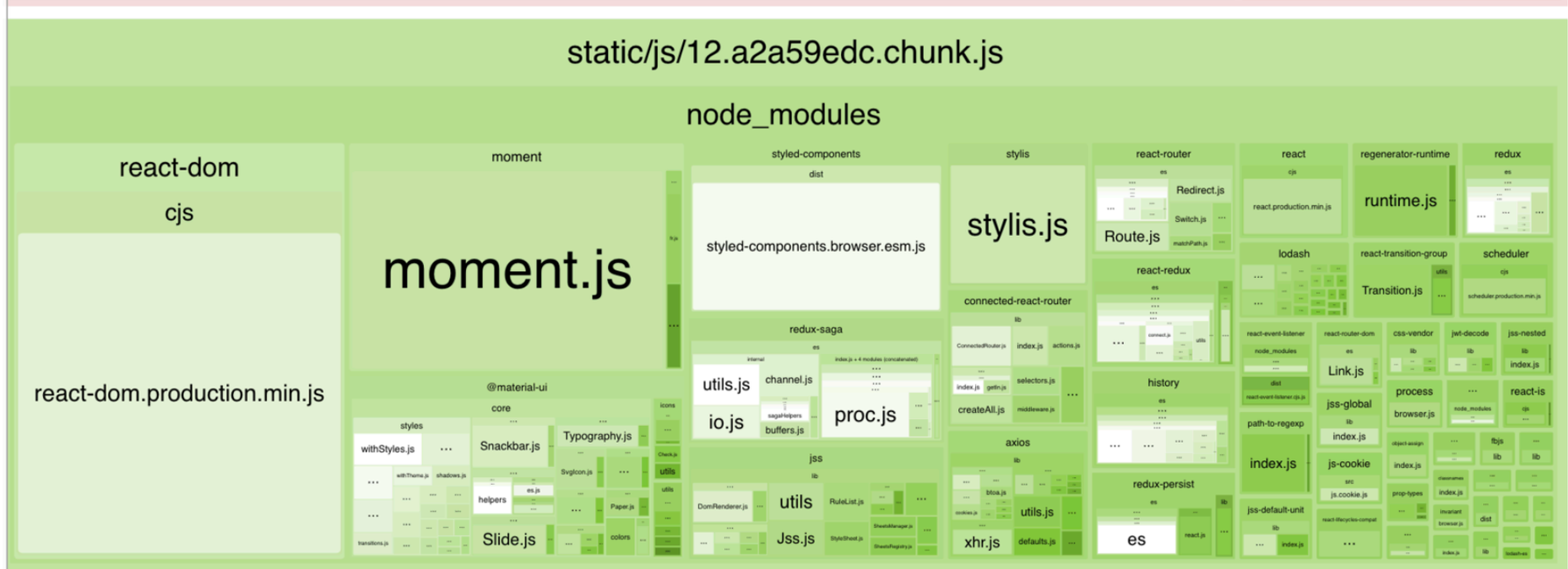
Avant la suppression des imports globaux

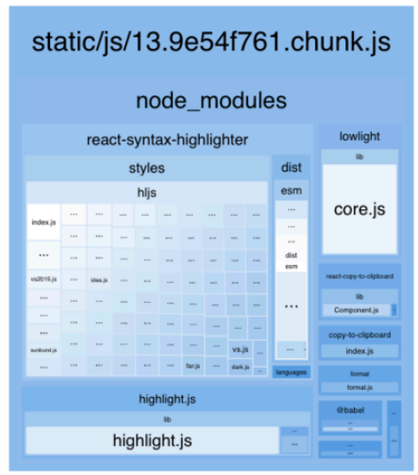


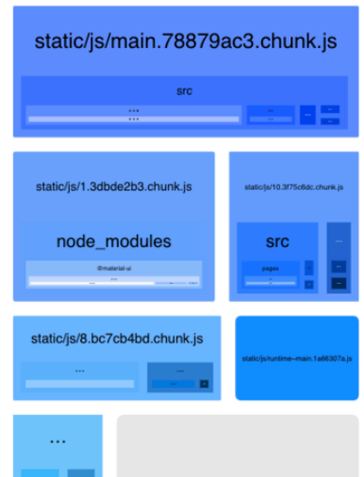
Après la suppression des imports globaux

150 KB (-160 KB)









Sur quels composants utiliser le code splitting ?

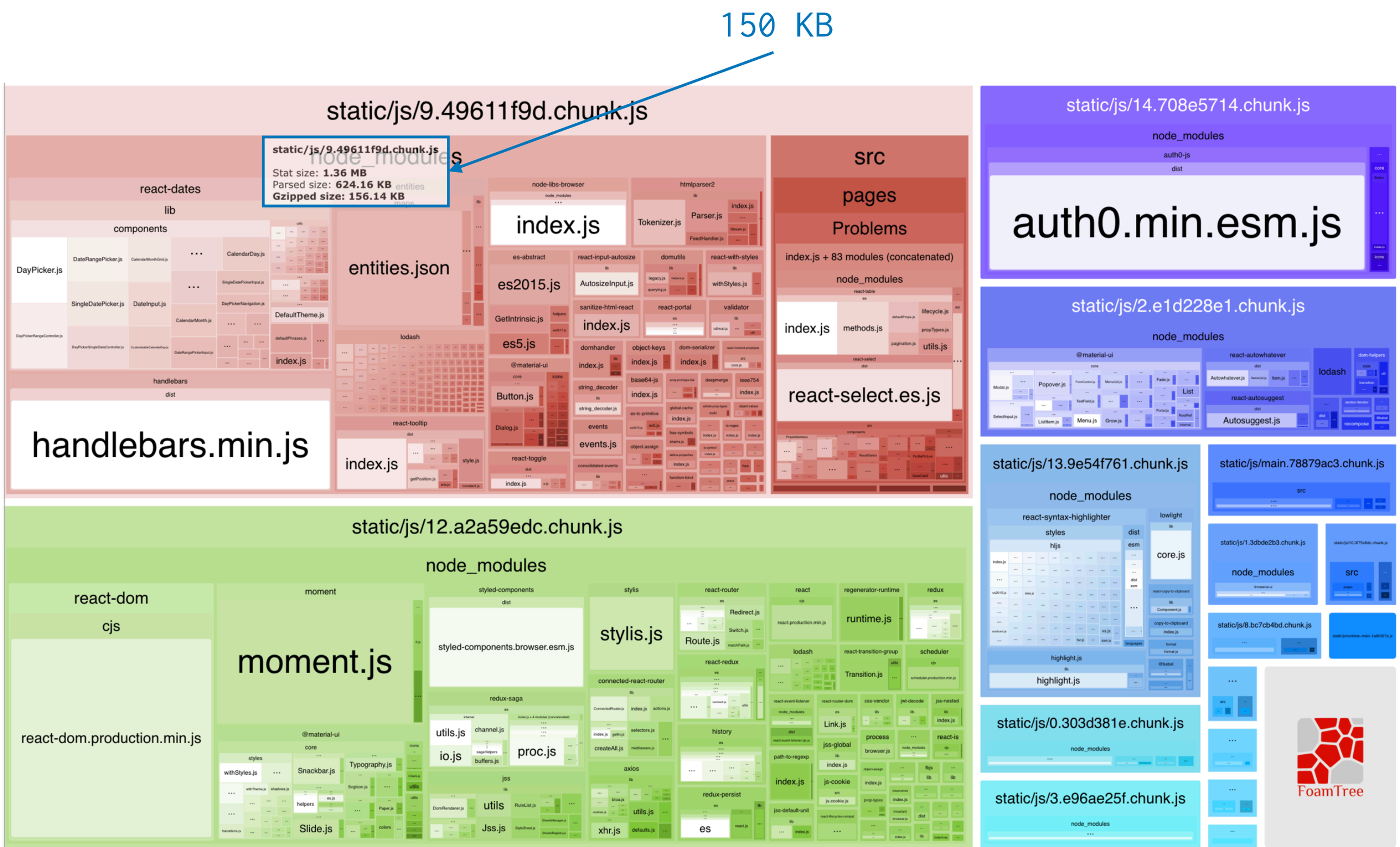
Sur quels composants utiliser le code splitting ?

Dès que l’élément n’est pas visible à l’arrivée sur la page:

- Les autres pages
- Les modales
- Les date pickers
- Les composants non affichés dans le viewport

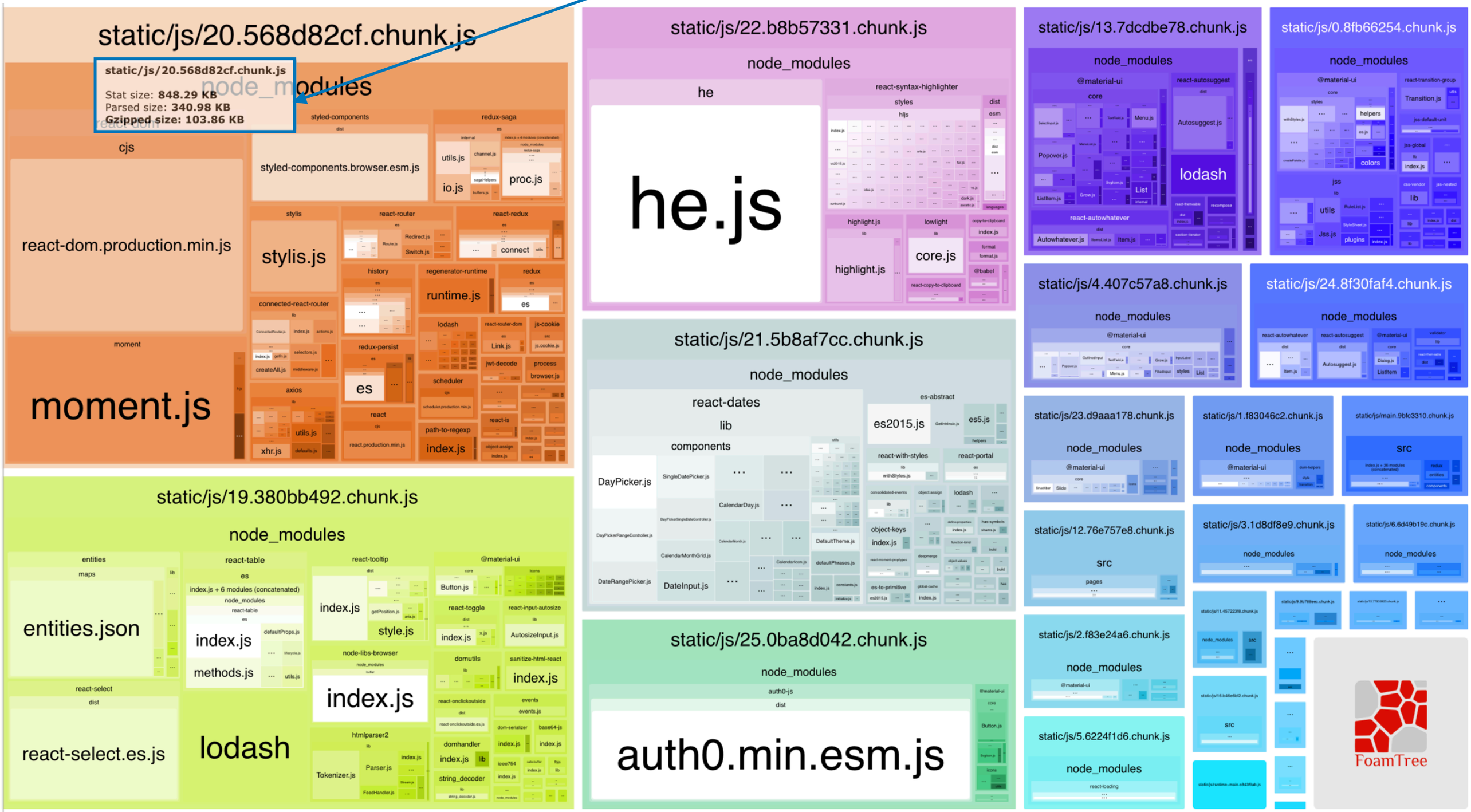

On split le reste…

Avant Code Splitting



Après Code Splitting

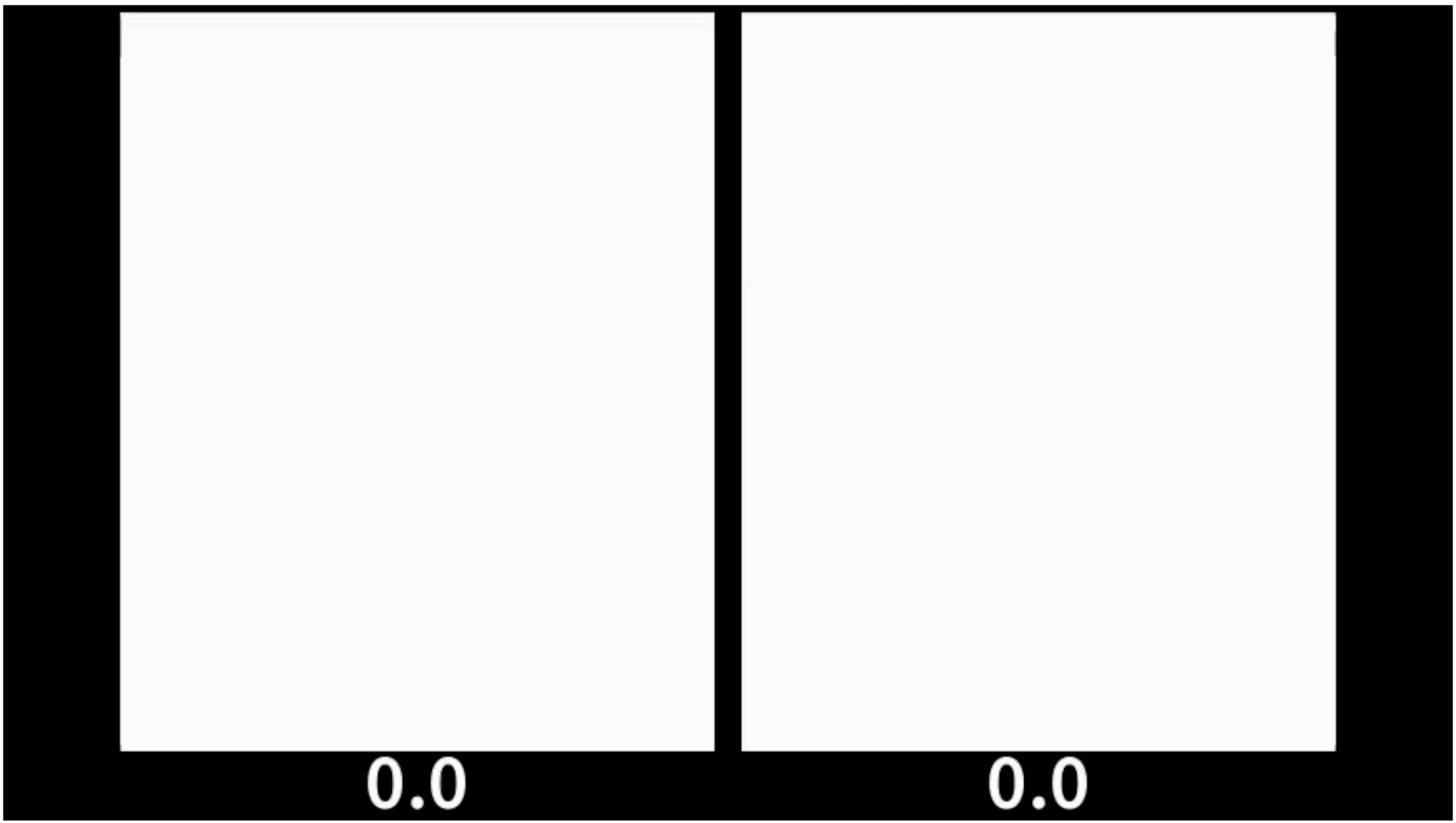
100 KB (-50KB)




Avant: 600KB
Après: 100KB + 75KB

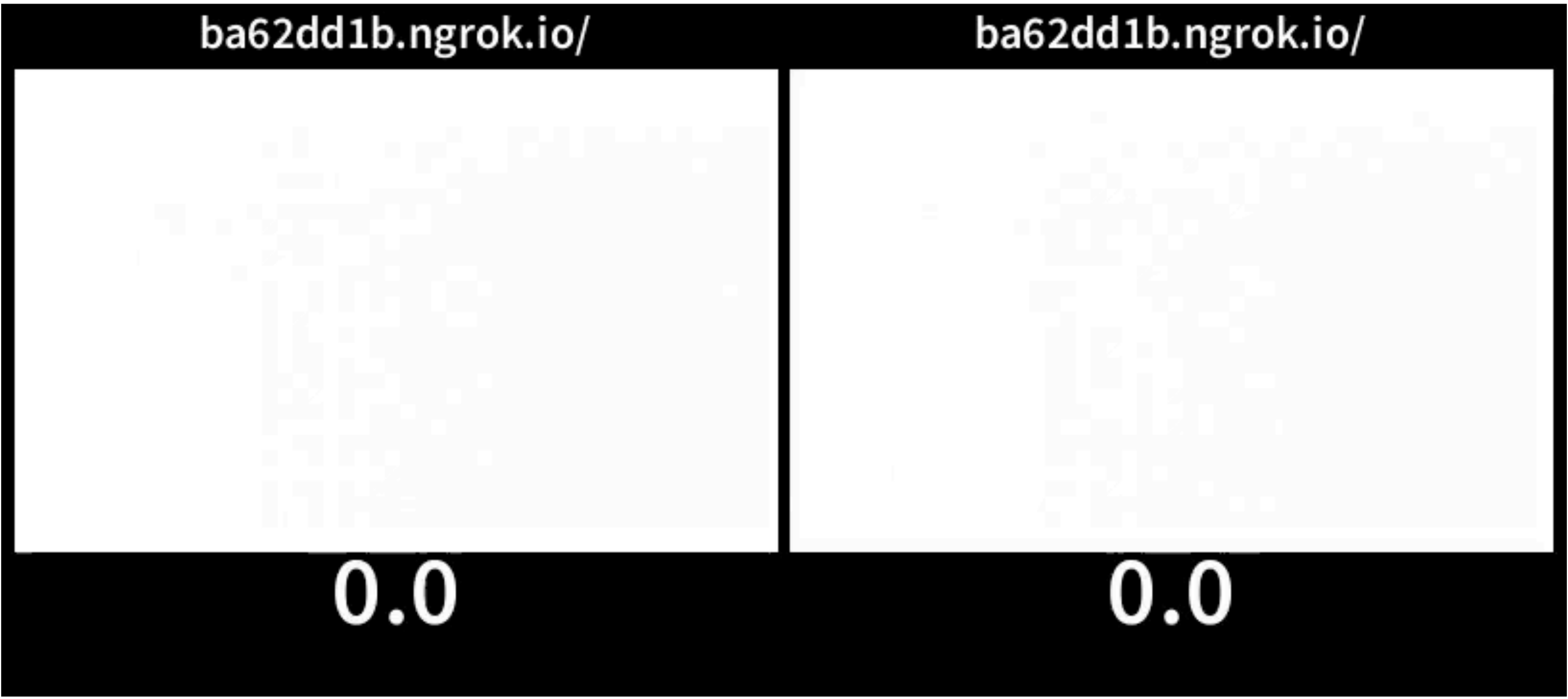
Comparaison (page principale)

Ipad, iOS12, 3G



Comparaison (page login)

Desktop, Chrome, 3G



Au menu

- Pourquoi diminuer la taille de son JavaScript ?
- Comment mettre en place le Code Splitting
- Comment choisir ses librairies ?
- Pour aller plus loin: le Bundle Splitting

Comment choisir la meilleure librairie de graphes ?

bundlephobia.com

BUNDLEPHOBIA

BADGES OPENCOLLECTIVE SCAN PACKAGE.JSON [B]

echarts@4.3.0

A powerful charting and visualization library for browser | 1 dependency

BUNDLE SIZE

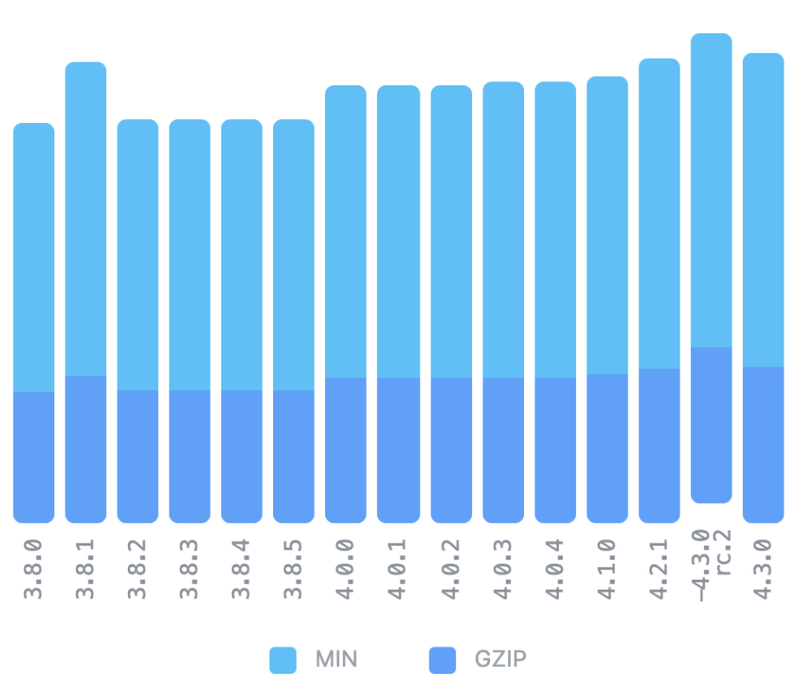
784.7 kB
MINIFIED

261.3 kB
MINIFIED + GZIPPED

DOWNLOAD TIME

8.71 s
2G EDGE [1]

5.23 s
EMERGING 3G [1]



Composition

Permet de voir si la librairie utilise des dépendances que vous avez déjà sur votre projet

Meilleur choix ?

Similar Packages

General purpose Charting libraries

d3

Data-Driven Documents

70 % SMALLER

79.36 kB MIN + GZIP

chart.js

Simple HTML5 charts using the canvas element.

57 % SMALLER

112.06 kB MIN + GZIP

frappe-charts

https://frappe.github.io/charts

93 % SMALLER

17.10 kB MIN + GZIP

highcharts

JavaScript charting framework

70 % SMALLER

79.57 kB MIN + GZIP

SUGGEST ANOTHER

Mon choix

recharts@1.7.1

tree-shakeable | some side-effects | 11 dependencies

BUNDLE SIZE

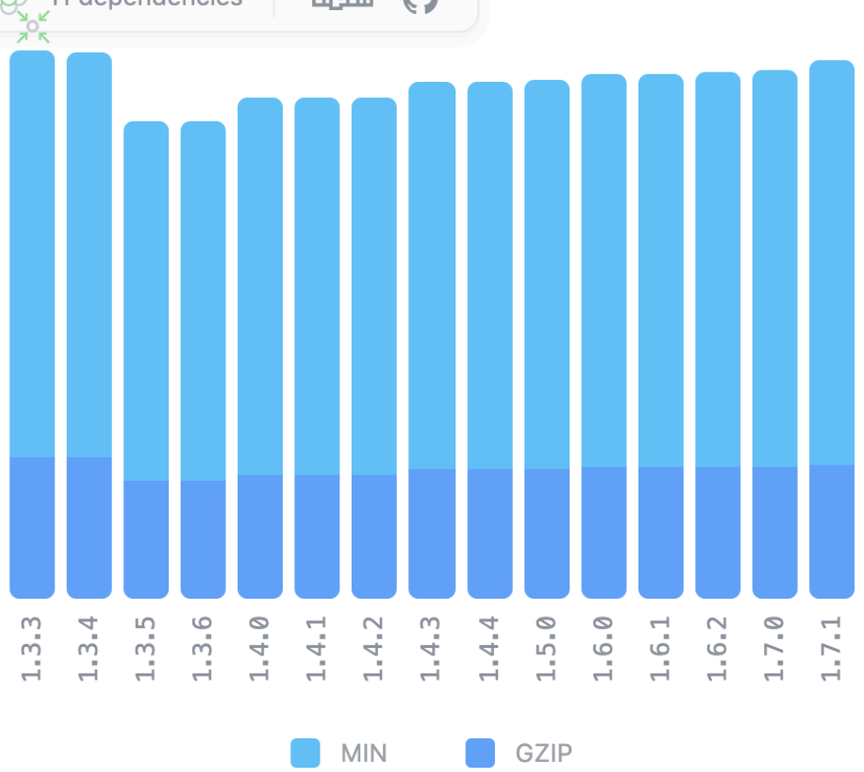
488.5 kB MINIFIED

120.5 kB MINIFIED + GZIPPED

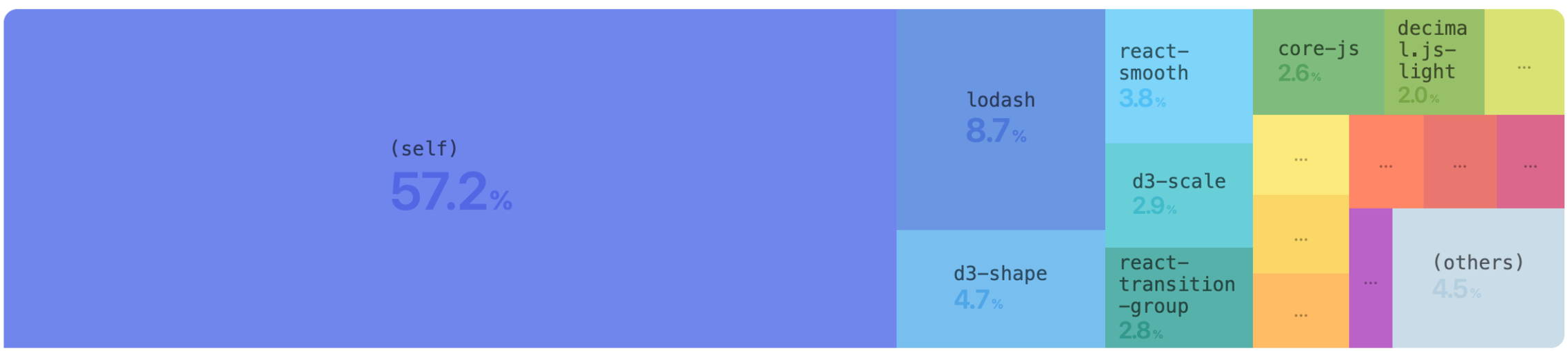
DOWNLOAD TIME

4.02 s 2G EDGE

2.41 s EMERGING 3G



Composition



Mon choix

Tree shakeable

recharts@1.7.1

tree-shakeable | some side-effects | 11 dependencies | npm

BUNDLE SIZE

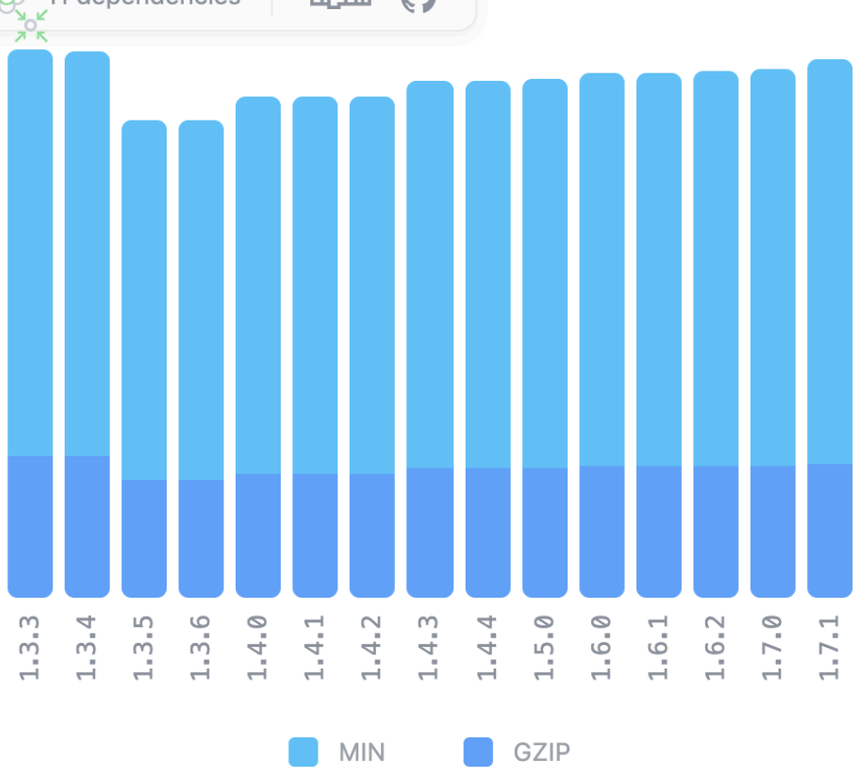
488.5 kB MINIFIED

120.5 kB MINIFIED + GZIPPED

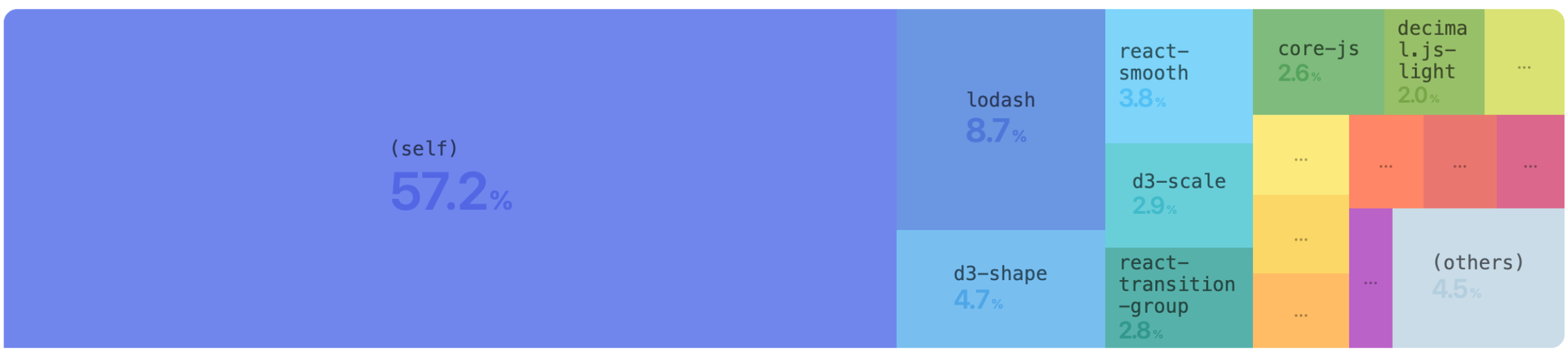
DOWNLOAD TIME

4.02 s 2G EDGE

2.41 s EMERGING 3G



Composition



Mon choix

Tree shakeable

recharts@1.7.1

tree-shakeable | some side-effects | 11 dependencies | npm

BUNDLE SIZE

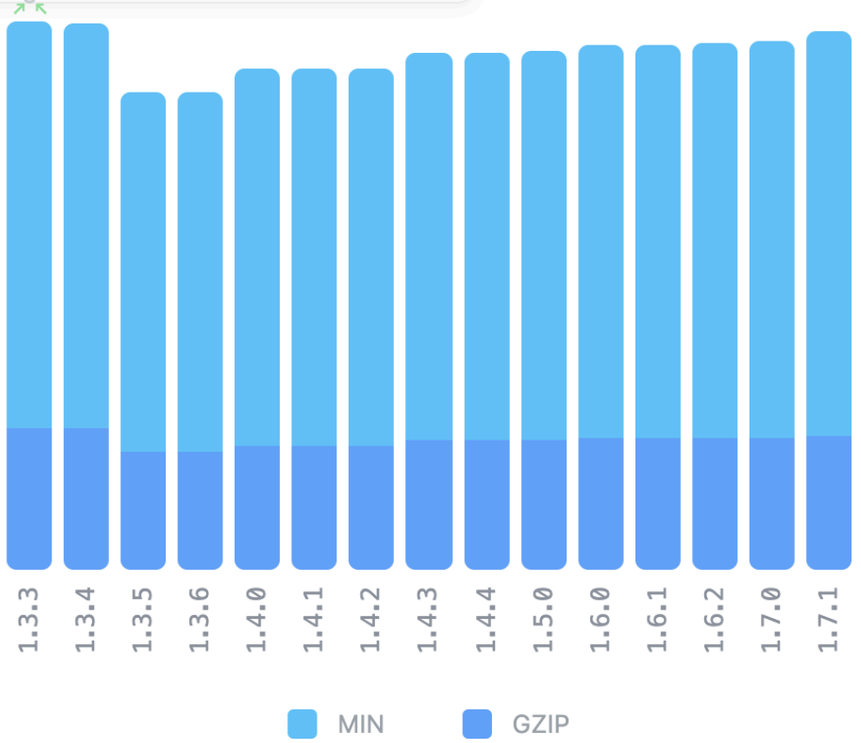
488.5 kB MINIFIED

120.5 kB MINIFIED + GZIPPED

DOWNLOAD TIME

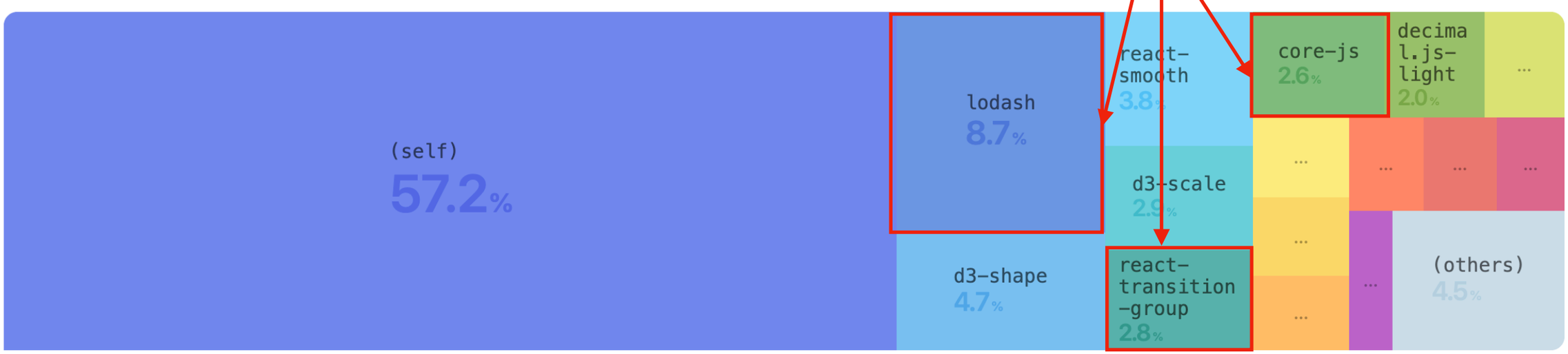
4.02 s 2G EDGE

2.41 s EMERGING 3G



Composition

Librairies déjà utilisées

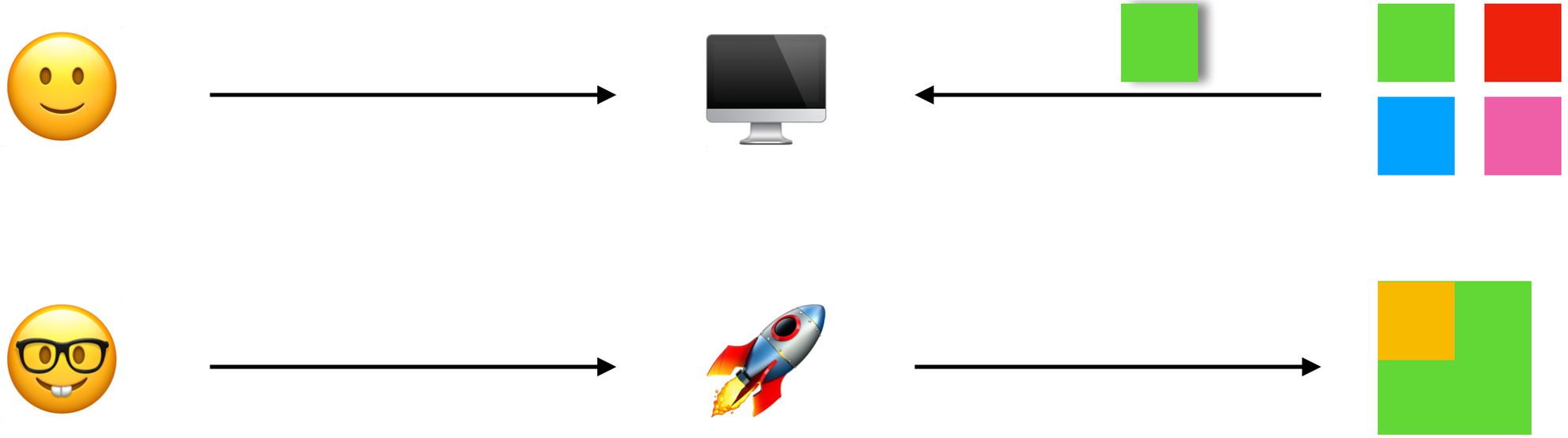


Au menu

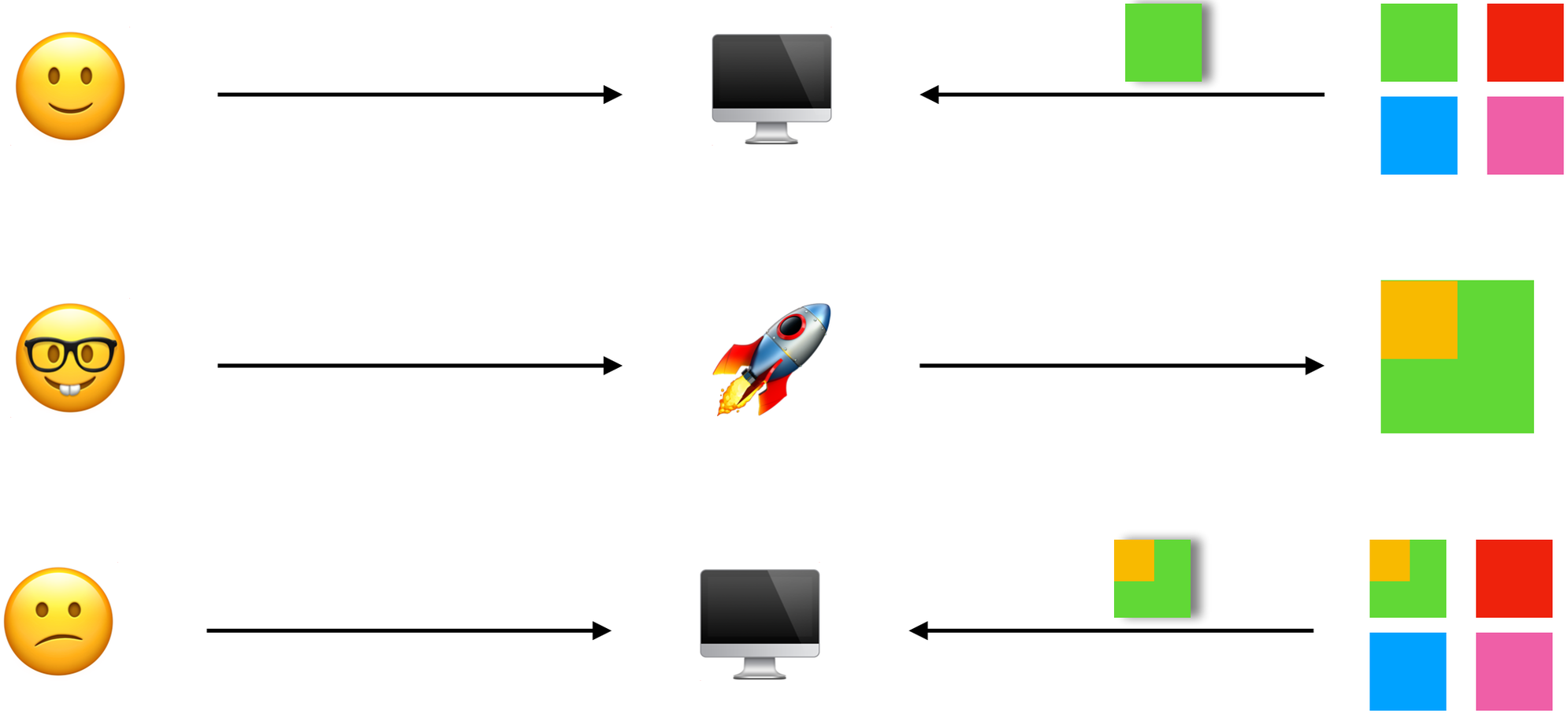
- Pourquoi diminuer la taille de son JavaScript ?
- Comment mettre en place le Code Splitting
- Comment choisir ses librairies ?
- Pour aller plus loin: le Bundle Splitting

Bundle Splitting

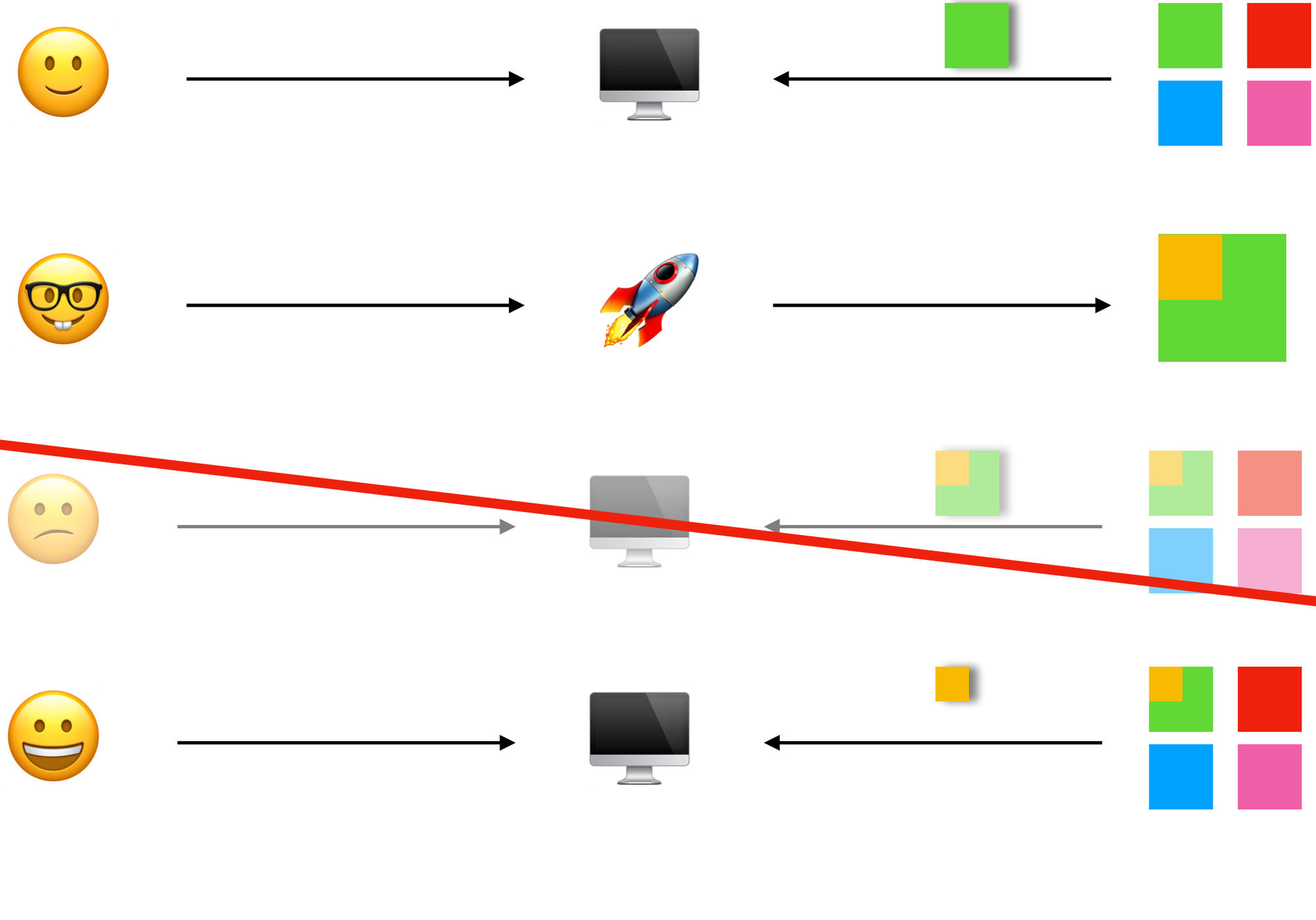
Bundle Splitting



Bundle Splitting



Bundle Splitting



Bundle Splitting

- Le nom d’un chunk dépend de son contenu

Bundle Splitting

- Le nom d'un chunk dépend de son contenu
- Un chunk déjà téléchargé va rester dans le cache navigateur de l'utilisateur

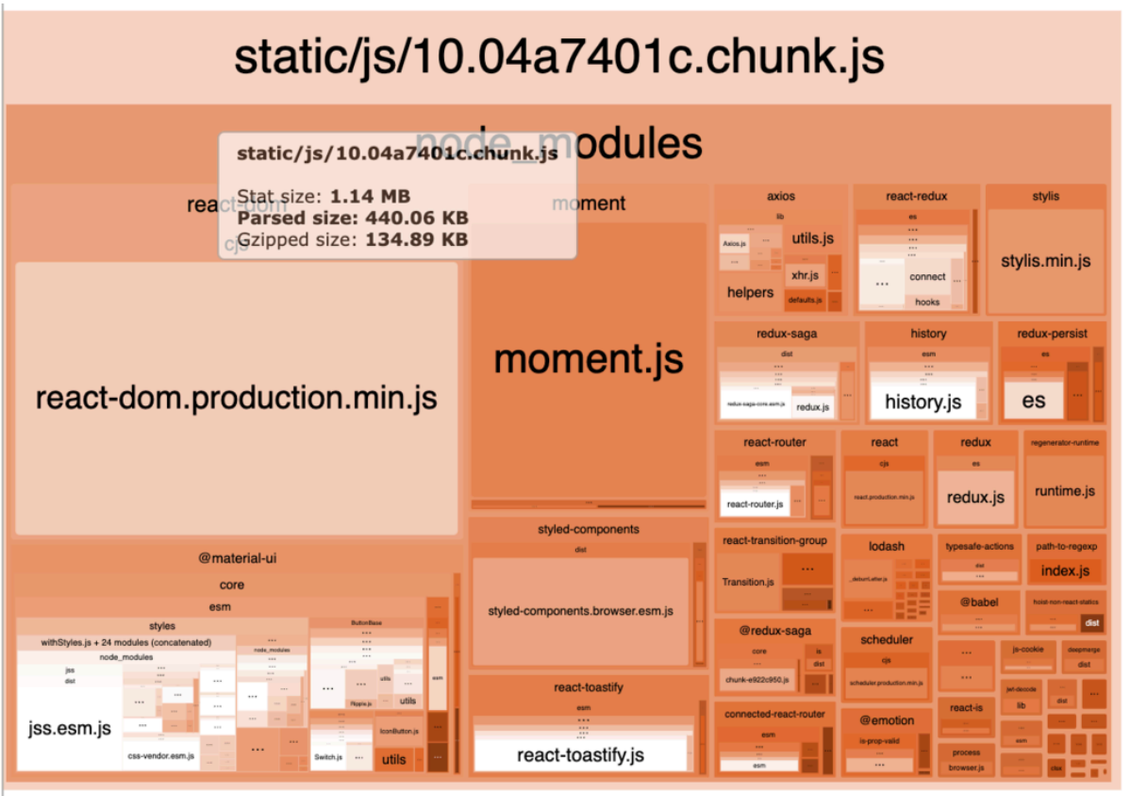
Bundle Splitting

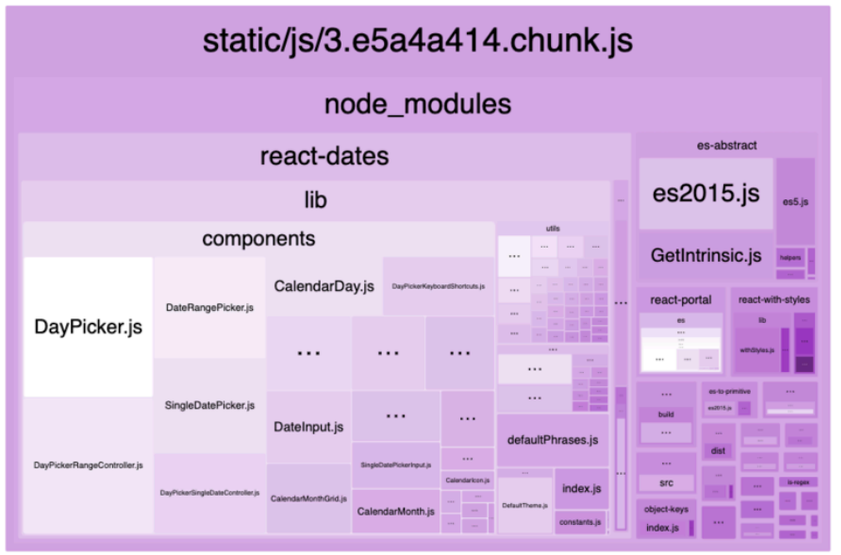
- Le nom d'un chunk dépend de son contenu
- Un chunk déjà téléchargé va rester dans le cache navigateur de l'utilisateur
- On va créer un chunk par librairie externe

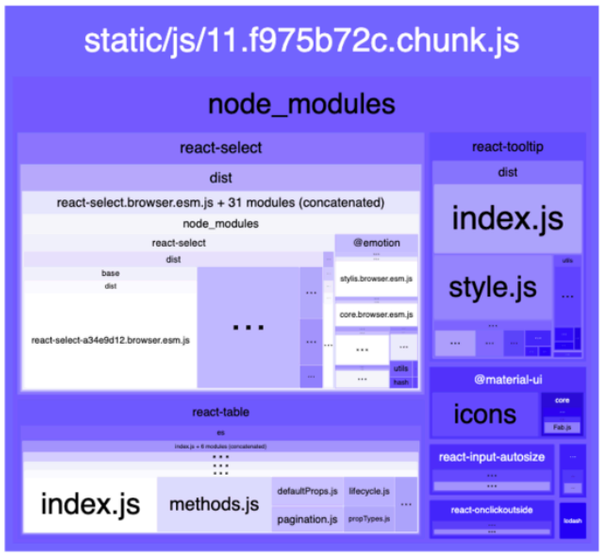
Bundle Splitting

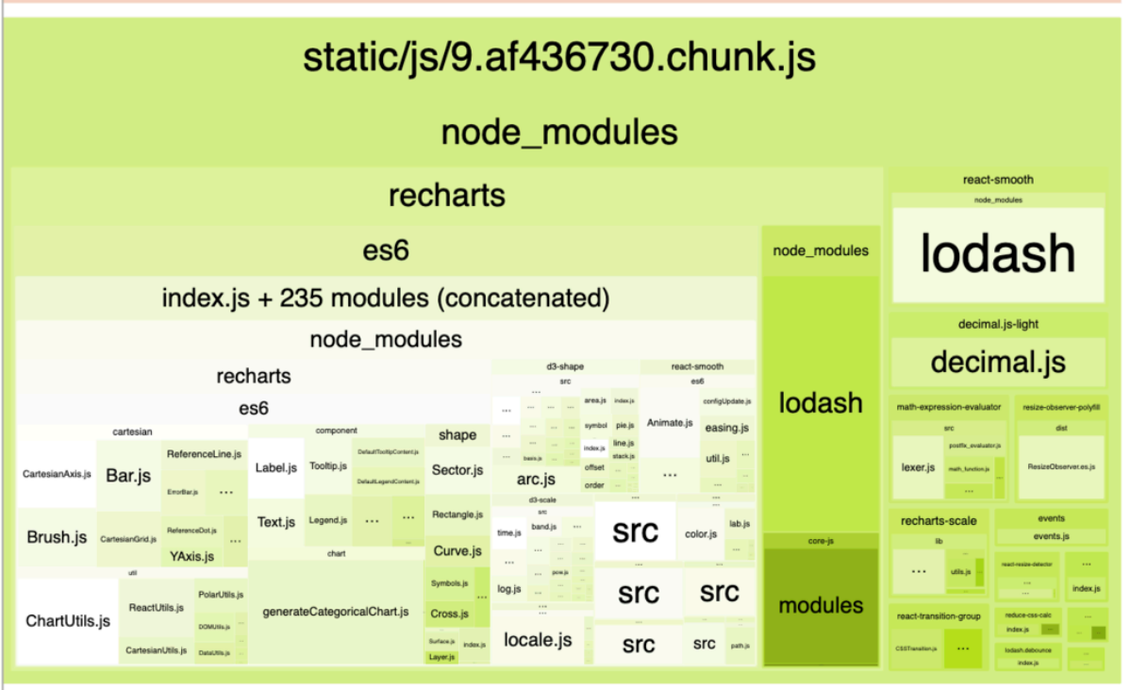
- Le nom d'un chunk dépend de son contenu
- Un chunk déjà téléchargé va rester dans le cache navigateur de l'utilisateur
- On va créer un chunk par librairie externe
- Cela nécessite HTTP/2, car on va faire une cinquantaine de requêtes par page

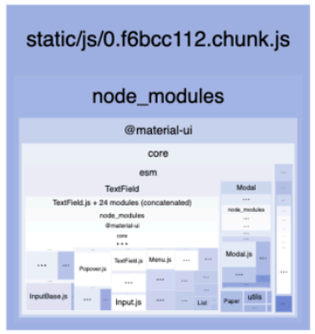
Avant Bundle Splitting

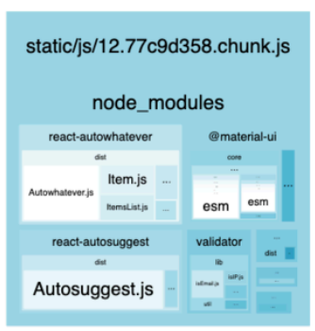


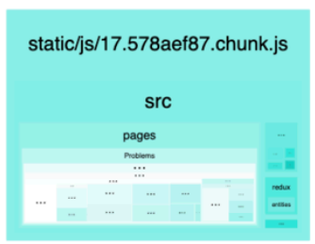


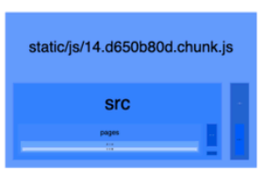


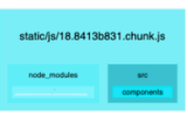




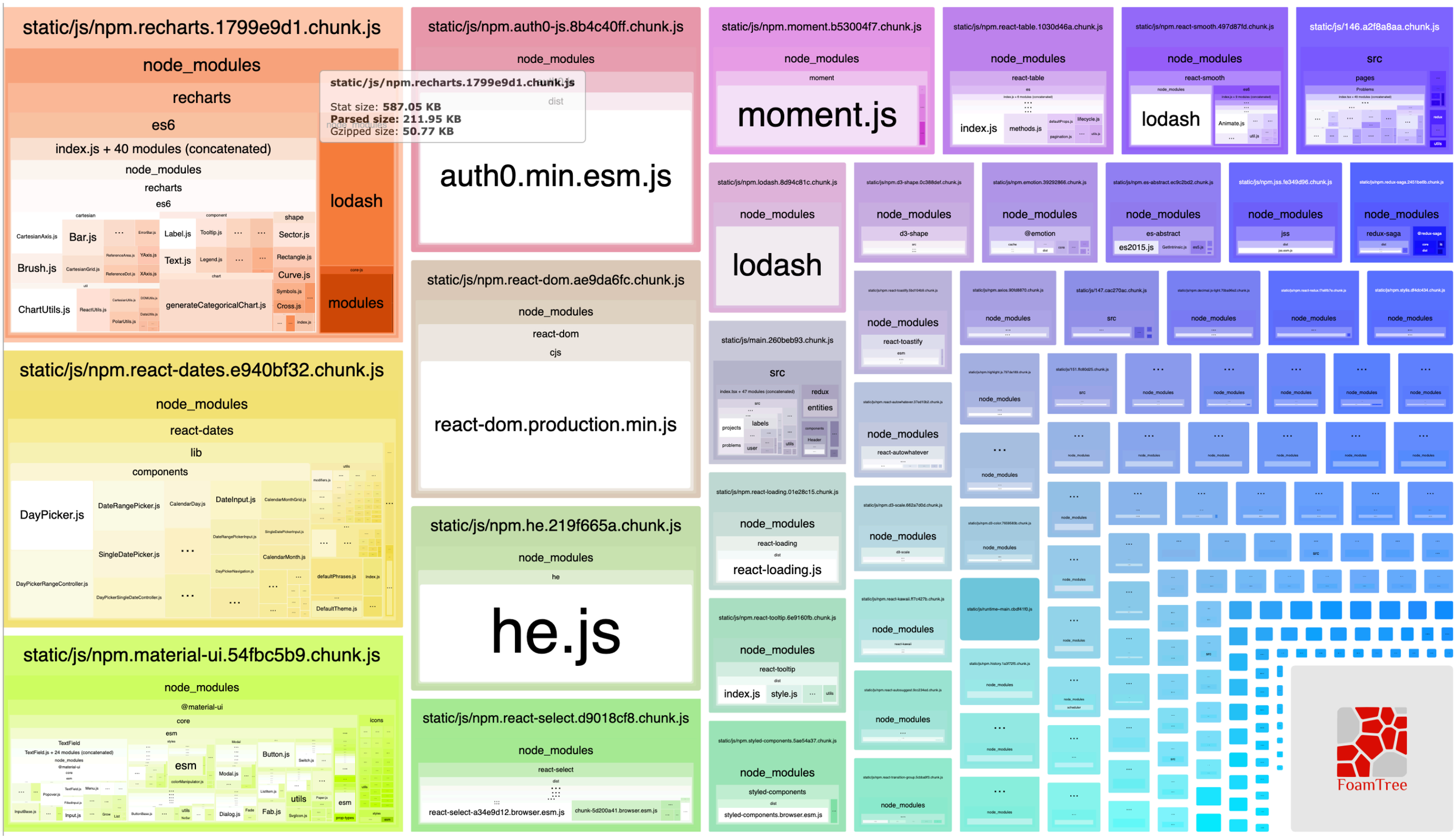




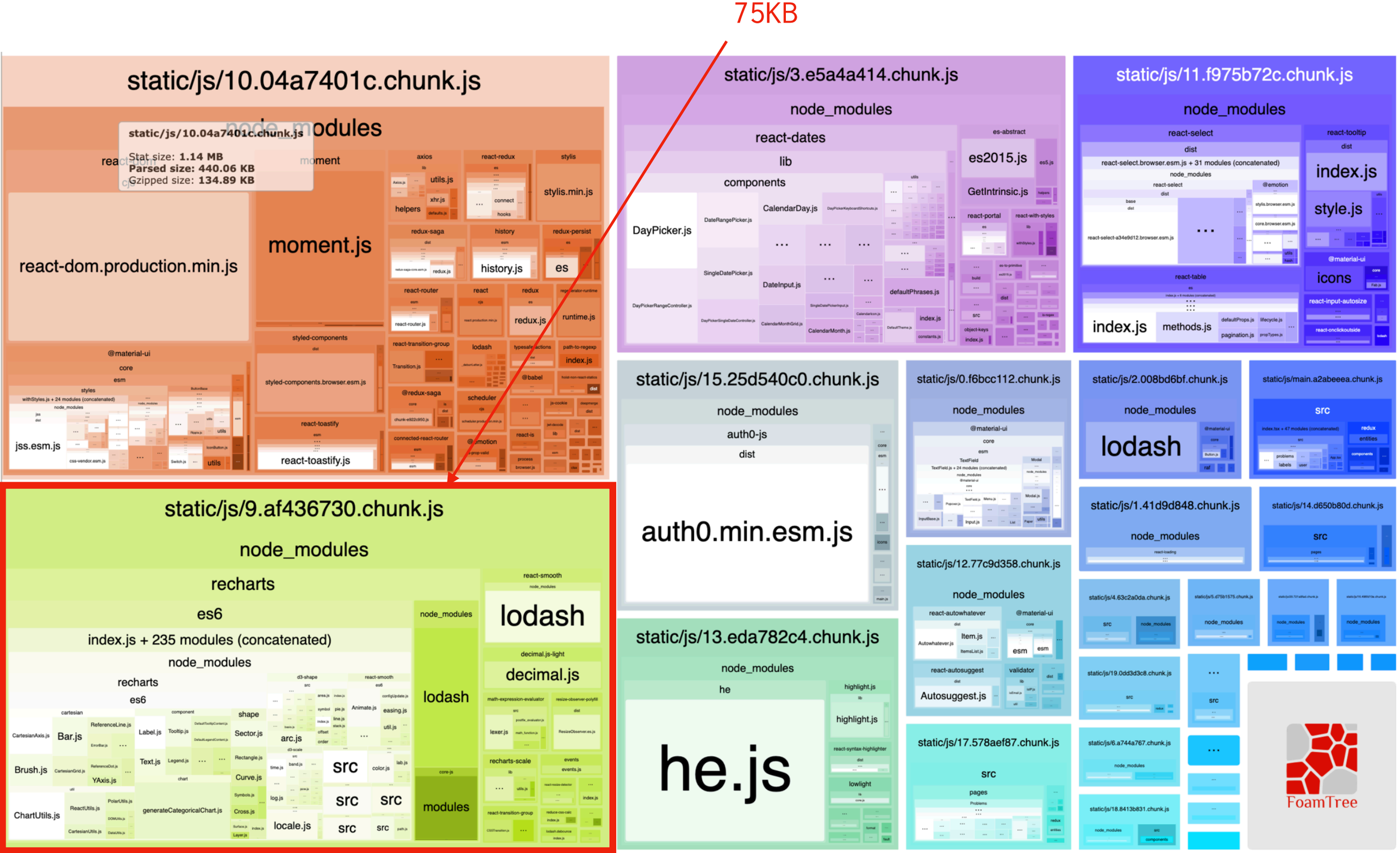




Après Bundle Splitting

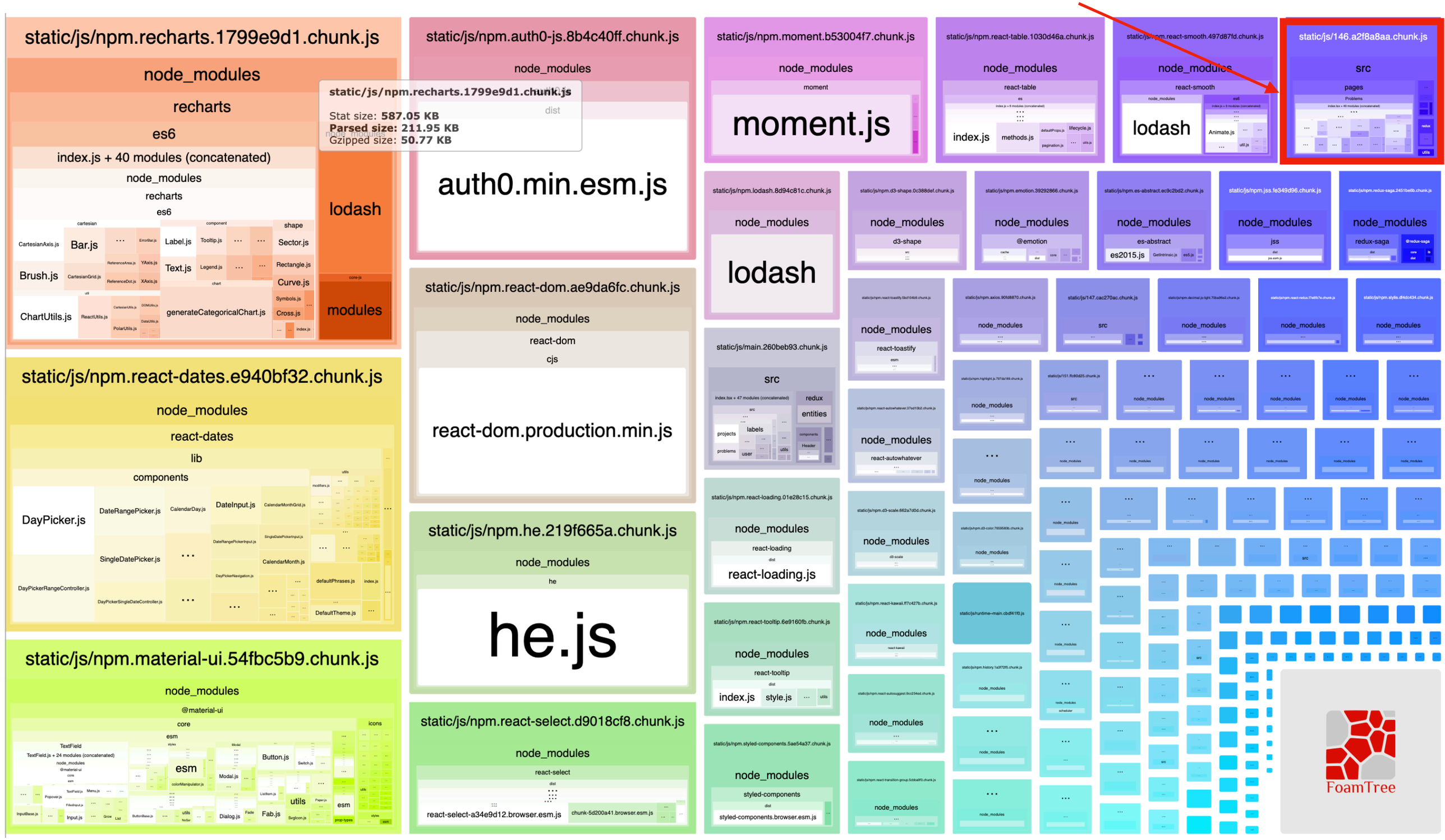


Modification de code, avant Bundle Splitting



Modification de code, après Bundle Splitting

5KB (-70KB)



On fait le bilan

On fait le bilan

Grâce au Code Splitting: on apporte uniquement à l'utilisateur ce dont il a besoin, quand il en a besoin.

On fait le bilan

Grâce au Code Splitting: on apporte uniquement à l'utilisateur ce dont il a besoin, quand il en a besoin.

Grâce au Bundle Splitting: on apporte uniquement à l'utilisateur ce qui a changé depuis sa dernière visite.

Pour résumer

Pour résumer

- Garder son JavaScript à moins de 200KB pour rendre une page

Pour résumer

- Garder son JavaScript à moins de 200KB pour rendre une page
- Utiliser le Code Splitting pour chaque composant non nécessaire au rendu initial

Pour résumer

- Garder son JavaScript à moins de 200KB pour rendre une page
- Utiliser le Code Splitting pour chaque composant non nécessaire au rendu initial
- Utiliser les imports allégés des grosses librairies

Pour résumer

- Garder son JavaScript à moins de 200KB pour rendre une page
- Utiliser le Code Splitting pour chaque composant non nécessaire au rendu initial
- Utiliser les imports allégés des grosses librairies
- Ne pas exporter globalement ses composants ou fonctions

Pour résumer

- Garder son JavaScript à moins de 200KB pour rendre une page
- Utiliser le Code Splitting pour chaque composant non nécessaire au rendu initial
- Utiliser les imports allégés des grosses librairies
- Ne pas exporter globalement ses composants ou fonctions
- Utiliser le Bundle Splitting si on est en HTTP/2

Pour s’entrainer

J’ai créé un exercice qui reprend tous les gestes que je viens de présenter

Pour s’entrainer

github.com/theodo/code-splitting-dojo

Merci !

Des questions ?