ModernWeb'20

技術在我們手上 世界就在我們手上

Hello!

我是趙勗博

Shubo

歐姆雷特 資深前端工程師

FB: Shubo 寫個 Code

Find me at
shubochao@gmail.com

Organized iThome

ModernWeb '20

你的 JS 該減肥了！
5 個提升網頁載入速度的技巧

Shubo

Organized iThome

ModernWeb.20

你的 JS 該減肥了！5個提升網頁載入速度的技巧

優化網頁載入速度

為何要優化 / 測速工具 / 如何決定優先順序

我的 JS 很肥會怎樣嗎？

瀏覽器運作原理 / Code Splitting

拆分出 Vendor Bundle

運用 webpack-bundle-analyzer 作分析

ModernWeb.20

你的 JS 該減肥了！5個提升網頁載入速度的技巧

Dynamic Loading

根據路徑 / 對肥大套件作 Dynamic Loading

Tree Shaking

移除沒用到的程式碼 / ESM import export

其他有用小技巧

使用 CDN 版本 / 使用 preset-env 減少 polyfill 體積

ModernWeb.tw '20

你的 JS 該減肥了！5個提升網頁載入速度的技巧

優化網頁載入速度

為何要優化 / 測速工具 / 如何決定優先順序

我的 JS 很肥會怎樣嗎？

瀏覽器運作原理 / Code Splitting

拆分出 Vendor Bundle

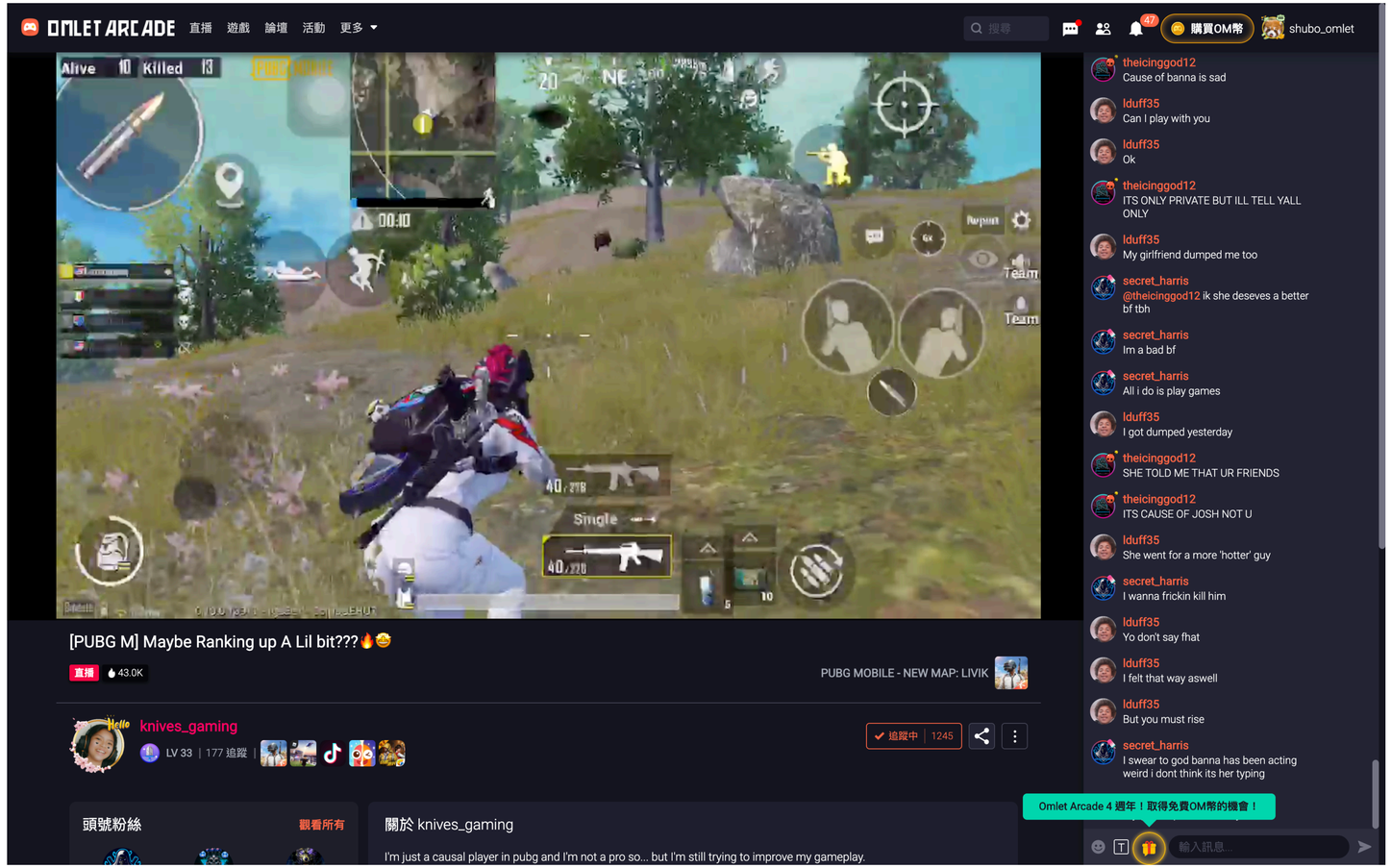
運用 webpack-bundle-analyzer 作分析

為何要優化網頁載入速度

- 載入速度＝使用者體驗
- 根據 Amazon 內部統計，慢100ms = 營收減少 1%
- SEO

Introduction

- 歐姆雷特
- Omlet Arcade
- 手機遊戲直播社群
- React.js + Webpack
- JS bundle 體積 -43%
- 載入速度 -30%



優化載入速度的第一步：善用測速工具

- Google PageSpeed Insight

如何決定優化的優先順序？

- 修正以後改善最多的項目
- 業務範圍中最容易改動的部分
- 先做出成果，得到主管信任，爭取更多優化空間
- 例：優化圖片尺寸，涉及 app 上傳 / 後端處理
- 例：優化 JavaScript bundle 體積，前端可自己修改 build 設定

ModernWeb.20

你的 JS 該減肥了！5個提升網頁載入速度的技巧

優化網頁載入速度

為何要優化 / 測速工具 / 如何決定優先順序

我的 JS 很肥會怎樣嗎？

瀏覽器運作原理 / Code Splitting

拆分出 Vendor Bundle

運用 webpack-bundle-analyzer 作分析

我的 JS 很肥，會怎樣嗎？

- 結論：JS Bundle 越大，網頁載入越慢
- 瀏覽器：解析 HTML -> 建立 DOM tree -> 渲染畫面
- JS “blocks” DOM tree 的建立過程
- 下載 + 執行 JS 越慢，block 越久，越慢渲染畫面

Code Splitting

- 將單一 JS 檔案拆分成許多小塊 (chunk)
  - 平行載入
  - 有需要時才載入
  - 各自可被快取

ModernWeb.20

你的 JS 該減肥了！5個提升網頁載入速度的技巧

優化網頁載入速度

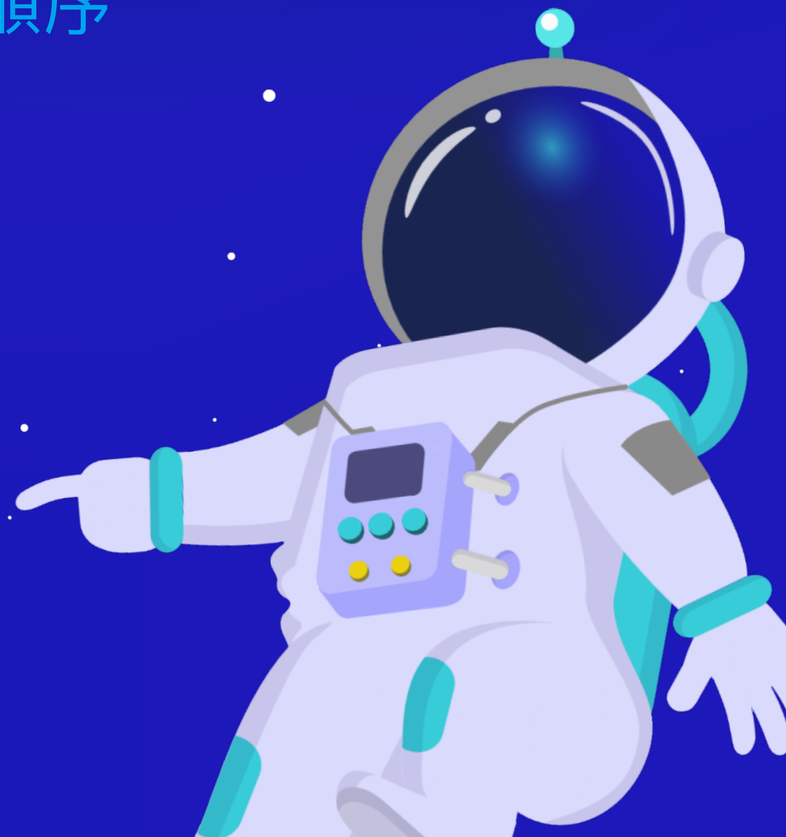
為何要優化 / 測速工具 / 如何決定優先順序

我的 JS 很肥會怎樣嗎？

瀏覽器運作原理 / Code Splitting

拆分出 Vendor Bundle

運用 webpack-bundle-analyzer 作分析



拆分出 Vendor bundle

- Webpack runtime and manifest: 負責模組之間互動，體積可忽略不計
- Application Bundle: UI / 商業邏輯，經常變動
- Vendor Bundle: 第三方套件 / node_modules，不太變動

Why Vendor Bundle?

- 通常變動不頻繁
- 可被快取
- 此配置在有 cache 情況下載入更快

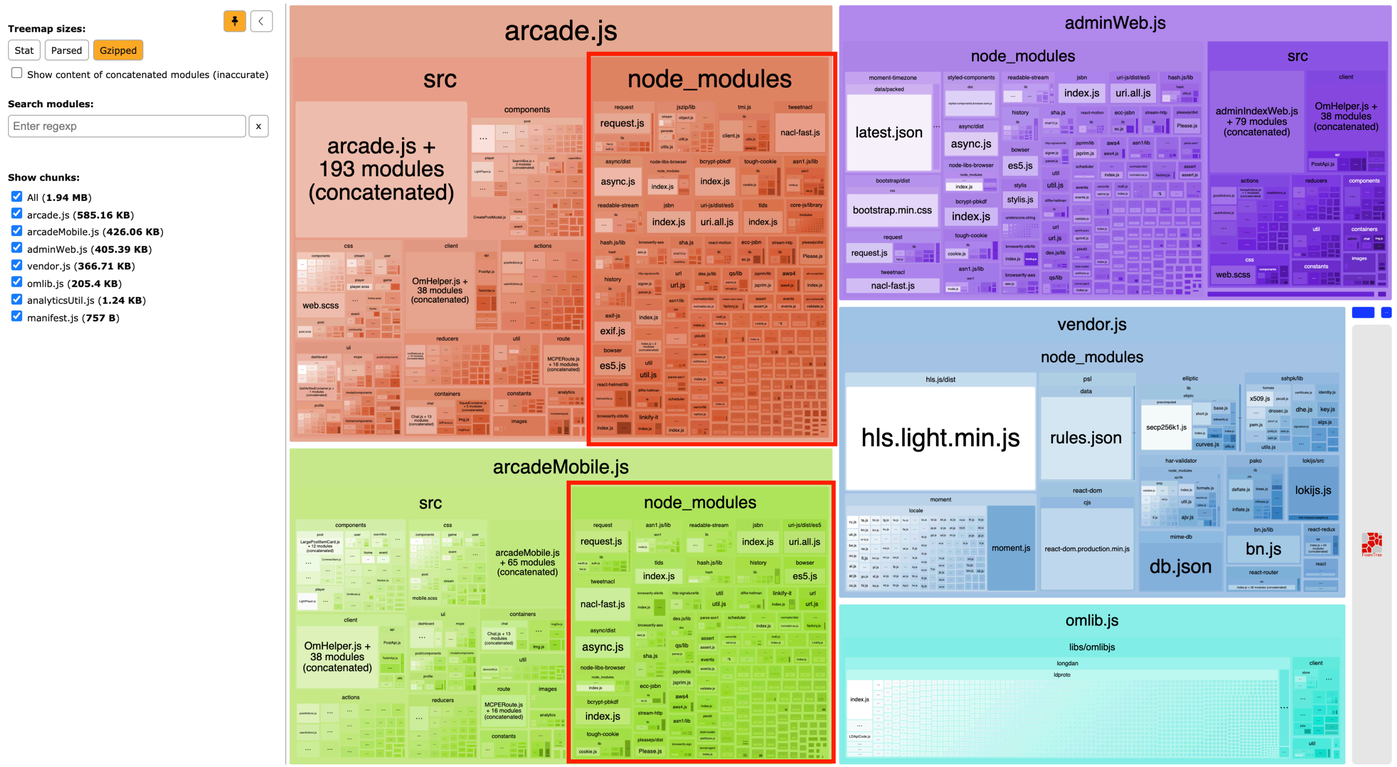
未優化前 Bundle 規劃

	桌機版	手機版	說明
arcade.js	**585KB**	**426KB**	**商業邏輯**
omlib.js	**205KB**	**205KB**	**內部 lib / API 規格**
vendor.js	**366KB**	**366KB**	**第三方套件**
Sum	1156KB	997KB	

ModernWeb '20

運用工具分析配置是否合理

- 運用 webpack-bundle-analyzer 作視覺化分析
- application bundle
  - 裡面有 node_modules
  - 併入 vendor bundle 更好



ModernWeb'20

取出 Vendor bundle

- 將 node_modules 統一打包成 vendor.js

```
// webpack.config.js
optimization: {
  splitChunks: {
    cacheGroups: {
      vendor: {
        test: /[\\/]node_modules[\\/]/,
        name: 'vendor',
      },
      omlib: {
        test: /[\\/]libs[\\/]/,
        name: 'omlib',
        chunks: 'all',
      },
    },
  },
},
```

ModernWeb '20

抽出 Vendor Bundle：結果

	桌機版	手機版	說明
arcade.js	~~585KB~~ 310KB	~~426KB~~ 218KB	商業邏輯
omlib.js	205KB	205KB	內部 lib / API 規格
vendor.js	~~366KB~~ 632KB	~~366KB~~ 632KB	第三方套件
Sum	1156KB	~~997KB~~ 1055KB*	

效果：減少 application bundle 的大小，加快再訪者載入速度

ModernWeb '20

ModernWeb.20

你的 JS 該減肥了！5個提升網頁載入速度的技巧

Dynamic Import

根據路徑 / 對肥大套件作 Dynamic Loading

Tree Shaking

移除沒用到的程式碼 / ESM import export

其他有用小技巧

使用 CDN 版本 / 使用 preset-env 減少 polyfill 體積

Dynamic Import

- 對新用戶而言，網站整體大小不變
- 如何減少整體下載量？動態下載需要的模組
- Dynamic Import: 需要用到某段程式碼的時候才透過網路下載 JS bundle

ESM import()

- Webpack 支援 ESM import() 語法實現 dynamic import

```
function getComponent() {
  return import(/* webpackChunkName: "lodash" */ 'lodash')
    .then(({ default: _ }) => {
      const element = document.createElement('div');
      element.innerHTML = _.join(['Hello', 'webpack'], ' ');
      return element;
    })
    .catch(error => 'An error occurred while loading the component');
}

getComponent()
  .then(component => {
    document.body.appendChild(component);
  })
```

```
...
                  Asset      Size          Chunks             Chunk Names
        index.bundle.js  7.88 KiB           index  [emitted]  index
vendors~lodash.bundle.js   547 KiB  vendors~lodash  [emitted]  vendors~lodash
Entrypoint index = index.bundle.js
...
```

ModernWeb '20

根據路徑作 Dynamic Import

- 為何要根據路徑？
  - GA數據顯示使用者大多停留在熱門頁面，換頁次數少
- react-router 支援 dynamic import

```
<Route path="/" component={AppRoot}>
  <IndexRoute
    getComponent={() => {
      return import('containers/HomeContainer').then(({ default: HomeContainer }) => HomeContainer);
    }}
  />
  <Route path="/games"
    getComponent={() => {
      return import('containers/GamesContainer').then(({ default: GamesContainer }) => GamesContainer);
    }}
  />
  {/* Other routes */}
</Route>
```

ModernWeb '20

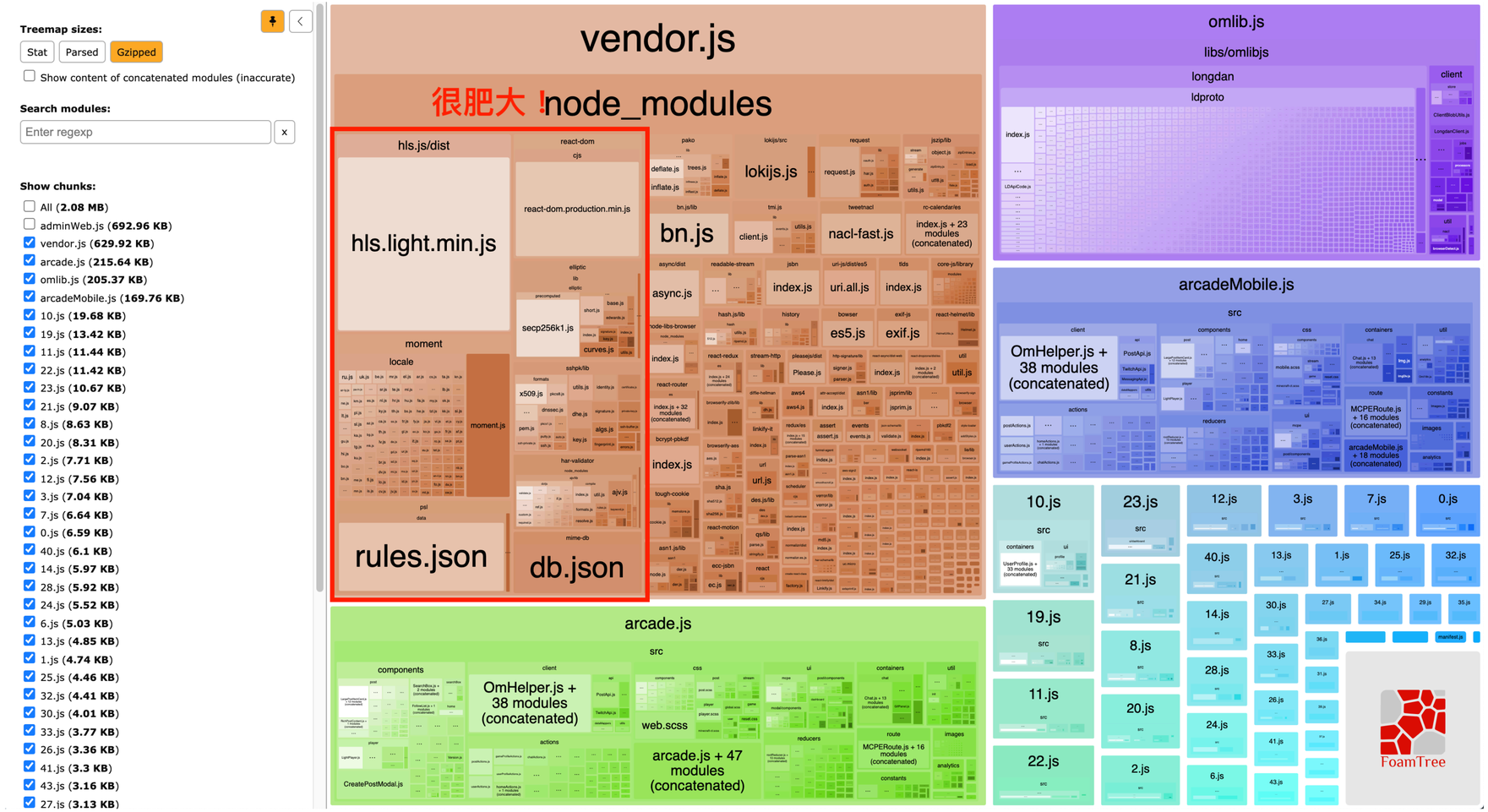
根據路徑作 Dynamic Import：結果

	桌機版	手機版	說明
arcade.js	~~310KB~~ 235KB	~~218KB~~ 189KB	商業邏輯
omlib.js	205KB	205KB	第三方套件
vendor.js	632KB	632KB	內部 lib / API 規格
Sum	~~1156KB~~ 1072KB*	~~1055KB~~ 1026KB*	* 體積最大的首頁~20KB

效果：減少 application bundle 下載量

ModernWeb '20

對肥大套件作 Dynamic Import



對肥大套件作 Dynamic Import

- 哪些套件適合 dynamic import?
  - 很肥大
  - 使用頻率低
  - 範例: JSZip (27KB)

```
class DropZone extends Component {
  componentDidMount() {
    this.importJSZip = import('jszip/dist/jszip.min.js').then(({ default: JSZip }) => JSZip);
  }

  onDrop() {
    this.importJSZip.then(JSZip => {
      const newZip = new JSZip();
      // 開始壓縮
    });
  }
}
```

```
// webpack.config.js
splitChunks: {
  cacheGroups: {
    vendor: {
      test: /[\\/]node_modules[\\/](?!jszip)/, // Exclude modules that need to be dynamically loaded
      name: 'vendor',
    },
  },
}.
```

肥大的第三方套件們

- hls.js: 77KB
- moment.js: 64KB
- JSZip: 27KB
- request: 70KB (本來專案是跨平台 web / node.js)

對肥大套件作 Dynamic Import：結果

	桌機版	手機版	說明
arcade.js	**235KB**	**189KB**	**商業邏輯**
omlib.js	**205KB**	**205KB**	**內部 lib / API 規格**
vendor.js	~~**632KB**~~ **267KB**	~~**632KB**~~ **267KB**	**第三方套件**
Sum	~~1072KB~~ 784KB*	~~1026KB~~ 707KB*	*以使用頻率最高的首頁為例

效果：大幅減少整體下載量

ModernWeb '20

ModernWeb.20

你的 JS 該減肥了！5個提升網頁載入速度的技巧

Dynamic Import

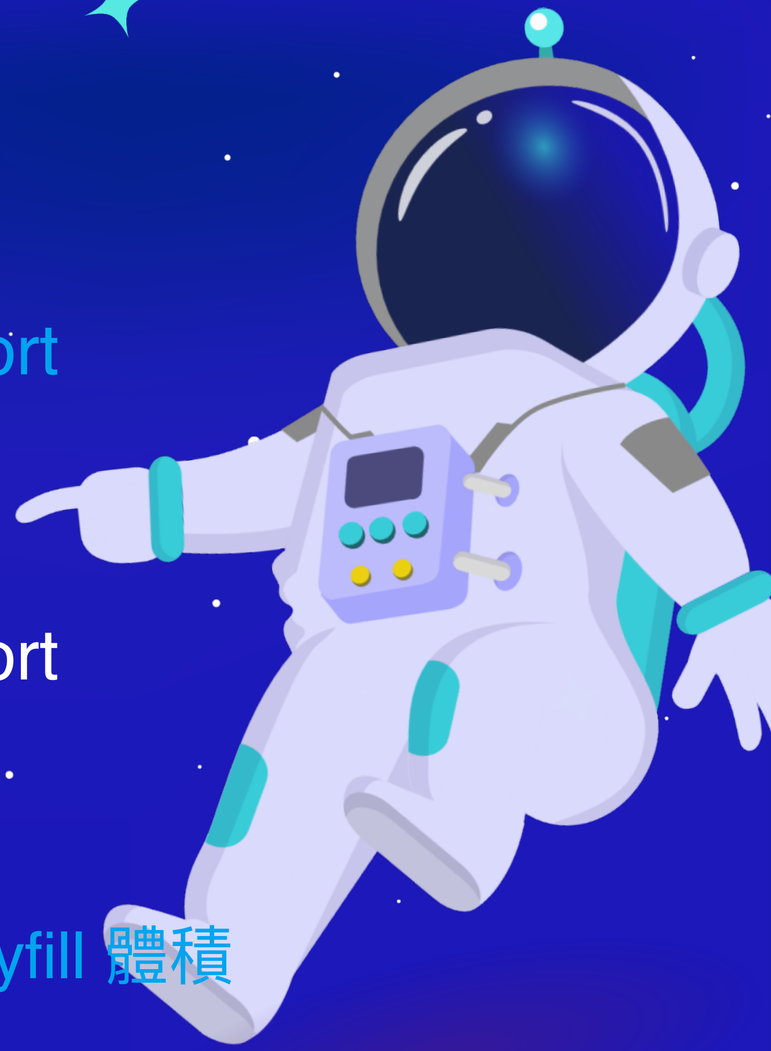
根據路徑 / 對肥大套件作 Dynamic Import

Tree Shaking

移除沒用到的程式碼 / ESM import export

其他有用小技巧

使用 CDN 版本 / 使用 preset-env 減少 polyfill 體積



Tree-Shaking

- 將沒用到的程式碼從 JS bundle 中移除
- 必須使用 ESM import/export (靜態結構)

如何設定 Tree-shaking

- 將專案中 CommonJS 的 require/module.exports 語法改寫成 ES Module 的 import/export 語法
- 在 package.json 中標示具有 side effects 的模組 (例如 CSS)

```
// Package.json
"sideEffects": [
  "*.css",
  "*.scss"
],
```

Tree Shaking：結果

	桌機版	手機版	說明
arcade.js	235KB	189KB	商業邏輯
omlib.js	~~205KB~~ 87KB	~~205KB~~ 87KB	內部 lib / API 規格
vendor.js	267KB	267KB	第三方套件
Sum	~~784KB~~ 666KB	~~707KB~~ 589KB	

效果：大幅減少整體下載量

ModernWeb '20

ModernWeb.20

你的 JS 該減肥了！5個提升網頁載入速度的技巧

Dynamic Import

根據路徑 / 對肥大套件作 Dynamic Import

Tree Shaking

移除沒用到的程式碼 / ESM import export

其他有用小技巧

使用 CDN 版本 / 使用 preset-env 減少 polyfill 體積

第三方套件使用 CDN 版本

- 經常被使用的第三方套件如 React / jQuery
- 容易被快取
- react-dom: 36KB

ModernWeb '20

使用 preset-env 減少 Polyfill 體積

- Polyfill: 讓舊的瀏覽器也能支援新的 API
- preset-env: preset-2015 的加強版
  - preset: babel-loader 的語法 plugin 集合
  - 根據支援瀏覽器清單，自動引入 polyfill
  - 結果：~~31KB~~ 18KB

最終結果

	桌機版	手機版	說明
arcade.js	~~585KB~~ 235KB	~~426KB~~ 189KB	商業邏輯
omlib.js	~~205KB~~ 87KB	~~205KB~~ 87KB	內部 lib / API 規格
vendor.js	~~366KB~~ 267KB	~~366KB~~ 267KB	第三方套件
Sum	~~1156KB~~ 666KB (-43%)	~~997KB~~ 589KB (-41%)	

- 根據 GA 數據，平均載入所需時間減少 30%

ModernWeb '20

結論

- 拆分出 vendor bundle
- 根據路徑做 dynamic import
- 對肥大第三方套件作 dynamic import
- 使用 tree shaking 移除沒用到的程式碼
- 熱門第三方套件使用 CDN 版本
- 使用 preset-env 減少 Polyfill 體積

ModernWeb '20

ModernWeb '20

Thanks!

我是趙勗博

Shubo

歐姆雷特 資深前端工程師

FB: Shubo 寫個 Code

Find me at
shubochao@gmail.com