Module Mystery Tour

Demystifying webpack and what it does to our code.

"Webpack is the kernel of frontend framework" - Random guy on twitter

https://twitter.com/__sun__/status/999737587873992704?s=12

"...we don't really know how it works, we don't want to know..." - Same random guy on twitter

https://twitter.com/__sun__/status/999737587873992704?s=12

Agenda

1. What is webpack
2. Why even module?
3. History of modules
4. How does it do it?

```
/******/ (function(modules) { // webpackBootstra
/******/  // The module cache
/******/  var installedModules = {};
/******/
/******/  // The require function
/******/  function __webpack_require__(moduleId)
/******/
/******/    // Check if module is in cache
/******/    if(installedModules[moduleId]) {
/******/      return installedModules[moduleId].e
/******/    }
/******/    // Create a new module (and put it i
/******/    var module = installedModules[module
/******/      i: moduleId,
/******/      l: false,
/******/      exports: {}
/******/    };
/******/
/******/    // Execute the module function
/******/    modules[moduleId].call(module.exports
```

```
// Flag the module as loaded
/******/    module.l = true;
/******/
/******/    // Return the exports of the module
/******/    return module.exports;
/******/  }
/******/
/******/
/******/  // expose the modules object (__webpa
/******/  __webpack_require__.m = modules;
/******/
          .
/******/  // expose the module cache
/******/  __webpack_require__.c = installedModu
/******/ cache)
/******/  // define getter function for harmony
/******/  __webpack_require__.d = function(expo
/******/    if(!__webpack_require__.o(exports, r
/******/      Object.defineProperty(exports, nar
/******/        configurable: false,
/******/        enumerable: true,
/******/        get: getter
/******/      });
/******/    }
          )dule.exports, __webpack_requi
/******/  };
```

```
/******/  // define __esModule on exports
/******/  __webpack_require__.r = function(exports) {
/******/    Object.defineProperty(exports, '__esModule', { value: true });
/******/  };
/******/
/******/  // getDefaultExport function for compatibility with non-harmony mo
/******/  __webpack_require__.n = function(module) {
/******/    var getter = module && module.__esModule ?
/******/      function getDefault() { return module['default']; } :
/******/      function getModuleExports() { return module; };
/******/    __webpack_require__.d(getter, 'a', getter);
/******/    return getter;
/******/  };
/******/
/******/  // Object.prototype.hasOwnProperty.call
/******/  __webpack_require__.o = function(object, property) { return Object
/******/
/******/  // __webpack_public_path__
/******/  __webpack_require__.p = "";
/******/
/******/
/******/  // Load entry module and return exports
/******/  return __webpack_require__(__webpack_require__.s = "./src/index.js
/******/ })
```

What is webpack?

Static Module Bundler

Huh?

Write things like this without browser support!

```
import React from 'react'
import ReactDOM from 'react-dom'
import Header from './header'
```

Why?

1. Encapsulation 🕵️‍♂️
    - Only expose what you want (privates)
    - No globals
2. Dependency Graph 📉
    - KNOW who is using every module

History

Script Tags

- Make sure to get that order right!
- No dependencies 👎

```html
<script src="./myfile.js" />
<!-- DON'T MOVE THIS -->
<script src="./myfile2.js" />
<script src="./myfile3.js" />
```

Common JS (CJS)

1. Started around 2009?
2. Not built with browser in mind
   - Synchronous - bad for perf
3. Needs a server or additional build steps to work in browser (node, browserify)

```
// Synchronously load jquery and all of its dependencies
const $ = require('jquery')
const bar = require('./bar')

$('button').each(bar).fade()
```

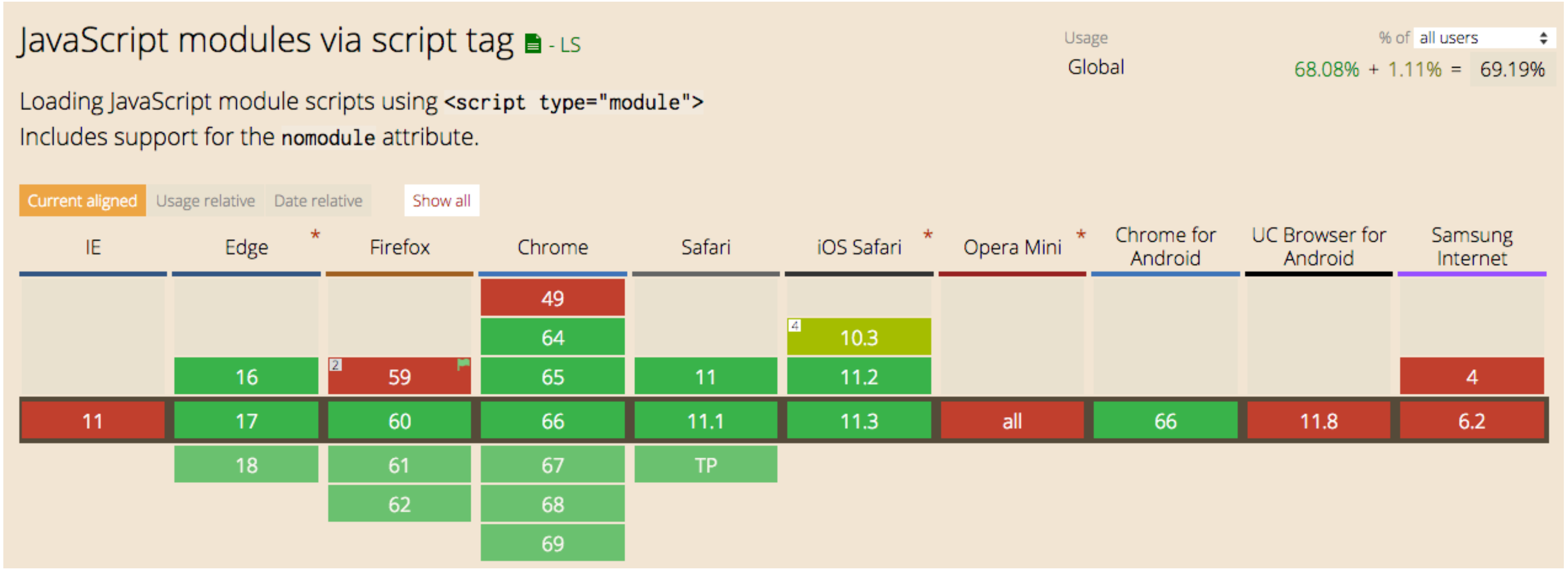
Require JS (AMD)

1. Based on AMD
2. Started around the same time as CommonJS
3. Works in browser by default, no extra tooling needed

```
// Dependency List
define(['require', 'jquery'], function(require, $) {
 // Returns a function that is your "module"
 return function() {
   const bar = require('./bar')
   $('button')
   .each(function(el) {
     bar(el)
   })
   .fade()
 }
})
```

ES Modules (ESM)

https://caniuse.com/#feat=es6-module



JavaScript modules via script tag - LS

Usage % of all users
Global 68.08% + 1.11% = 69.19%

Loading JavaScript module scripts using `<script type="module">` Includes support for the `nomodule` attribute.

IE	Edge *	Firefox	Chrome	Safari	iOS Safari *	Opera Mini *	Chrome for Android	UC Browser for Android	Samsung Internet
			🟥 49						
			🟩 64		4 🟨 10.3				
	🟩 16	2 🟥 59	🟩 65	🟩 11	🟩 11.2				🟥 4
🟥 11	🟩 17	🟩 60	🟩 66	🟩 11.1	🟩 11.3	🟥 all	🟩 66	🟥 11.8	🟥 6.2
	🟩 18	🟩 61	🟩 67	🟩 TP					
		🟩 62	🟩 68						
			🟩 69						

- ES6 (2015) 🎈
- In browsers now
- Great explanation 🤸

```js
// entry.js
import $ from 'jquery'
import bar from './bar'

$('button').each(bar).fade()
```

```html
<!-- index.html -->
<script type="module" src="./entry.js" />
```

```js
// entry.js
import $ from 'jquery'
import bar from './bar'

$('button').each(bar).fade()
```

```html
<!-- index.html -->
<script type="module" src="./entry.js" />
```

How?

Loaders

```
import all from './data/apples.js'

export const pick = type =>
  all.find(apple =>
    apple.type.toLowerCase() == type.toLowerCase()
  )

export const favorite = type => {
  const apple = pick(type)
  return apple
    ? { ...pick(type), favorite: true }
    : null
}
```

```
// Rest Spread Polyfill
var _extends =
  Object.assign ||
  function(target) {
    for (var i = 1; i < arguments.length; i++) {
      var source = arguments[i]
      for (var key in source) {
        if (Object.prototype.hasOwnProperty.call(source, key)) {
          target[key] = source[key]
        }
      }
    }
    return target
  }

import all from './data/apples.js'

export var pick = function pick(type) { /**/ }

export var favorite = function favorite(type) {
  var apple = pick(type)
  return apple ? _extends({}, pick(type), { favorite: true }) : null
}
```

Basic babel-loader implementation

```
var babel = require('babel-core')
var loaderUtils = require('loader-utils')

module.exports = function(source) {
  var options = loaderUtils.getOptions(this) || {}
  var result = babel.transform(source, options)
  var code = result.code
  var map = result.map // source map

  this.callback(null, code, map)
}
```

```
var babel = require('babel-core')
var loaderUtils = require('loader-utils')

module.exports = function(source) {
  var options = loaderUtils.getOptions(this) || {}
  var result = babel.transform(source, options)
  var code = result.code
  var map = result.map // source map

  this.callback(null, code, map)
}
```

```
var babel = require('babel-core')
var loaderUtils = require('loader-utils')

module.exports = function(source) {
  var options = loaderUtils.getOptions(this) || {}
  var result = babel.transform(source, options)
  var code = result.code
  var map = result.map // source map

  this.callback(null, code, map)
}
```

```
var babel = require('babel-core')
var loaderUtils = require('loader-utils')

module.exports = function(source) {
  var options = loaderUtils.getOptions(this) || {}
  var result = babel.transform(source, options)
  var code = result.code
  var map = result.map // source map

  this.callback(null, code, map)
}
```

How Webpack Works

1. Create a graph of all modules

```js
// index.js
import { startsWith, favorite } from './apple'

startsWith('j').map(apple => favorite(apple.type))
```

```js
// apple.js
import all from './data/apples.json'

export const pick = type => {}
export const startsWith = char => {}
export const favorite = type => {}
```

```
+ index.js
  → `startsWith` from './apple'
  → `favorite` from './apple'

+ apple.js
  → json from './data/apples'
```

2. Create a graph of all chunks

```
EntryPoint [main]
 + Chunk main
   - index.js
   - apple.js

 + ChunkGroup [async]
   - apple.js
   - someLazyLoadedFile.js        import('./someLazyLoadedFile.js')
```

3. Optimize / Concat / Assign Ids

1. Move shared async code into parent

```
EntryPoint [main]
 + Chunk main
   - index.js
   - apple.js
+ ChunkGroup [async] (parent: main)
   - apple.js
   - someLazyLoadedFile.js
```

```
EntryPoint [main]
 + Chunk main
   - index.js
   - apple.js
+ ChunkGroup [async] (parent: main)
   - someLazyLoadedFile.js
```

3. Optimize / Concat / Assign Ids

1. Move shared async code into parent
2. Scope hoisting, move source code into same function wrapper
3. Assign Ids to module

Generate Runtime Code

Build Asset Files

References

1. https://hacks.mozilla.org/2018/03/es-modules-a-cartoon-deep-dive/
2. https://github.com/TheLarkInn/artsy-webpack-tour
3. https://www.youtube.com/watch?v=UNMkLHzofQI
4. https://webpack.js.org/concepts/