Modules and build tools

Building and bundling code

Shadi Lahham - Programmazione web - Frontend - Javascript

Modules

Modules

- Import from other files
  - Variables, functions, objects, etc
- Modern browsers only
  - Browser compatibility
- Don't work with local files
  - file:///url
- Need an HTTP server
  - http-server ,NginX, Apache, Wamp, XAMPP
  - Live Server - VSCode ext
  - The 8 Best Open Source Web Servers

Export & import

Must specify type="module" in <script> tag

```
<script src="scripts/helper.js" type="module"></script>
<script src="scripts/main.js" type="module"></script>

// helper.js -- exports a default function
export default function(msg) {
 return `I am a helper function. You passed "${msg}"`;
}

// main.js
import helper from './helper.js';
const msg = helper('external help');
console.log(msg);
```

Multiple exports

```js
// helper.js
export default (msg) => `Echo "${msg}"`;
const user = { name: 'james' };
const double = (x) => x * 2;
export { user, double };

// main.js
import echo, { user, double } from './helper.js'; // named imports must match
console.log(echo('external help'));
console.log(user.name);
console.log(double(5));
```

Renaming imports

```
// helper.js
export default (msg) => `Echo "${msg}"`;
const user = { name: 'james' };
const double = (x) => x * 2;
export { user, double };

// main.js
import anyName from './helper.js'; // imports the default export as anyName
console.log(anyName('external help'));

// alternative main.js - renaming named imports
import anyName, { user as someone, double as timesTwo} from './helper.js';
console.log(anyName('external help'));
console.log(someone.name);
console.log(timesTwo(5));
```

Renaming imports

```
// helper.js
export default (msg) => `Echo "${msg}"`;
const user = { name: 'james' };
const double = (x) => x * 2;
export { user, double };

// alternative main.js - renaming default
import { default as repeat, user as someone, double as timesTwo} from './helper.js';
console.log(repeat('external help'));

// alternative main.js - import all
import * as everything from './helper.js'; // renaming named imports
console.log(everything.default('external help'));
console.log(everything.user.name);
console.log(everything.double(5));
```

Best practice

```
// main.js
import helper from './helper.js'; // import just one entity, name it meaningfully
console.log(helper.name);

// helper.js
const api = {
 name: 'something'
 // ...
};
export default api; // export only one entity and export it as default
```

IIFE

What is an IIFE

Immediately invoked function expression

- The old way to create a scope for variables declared using var other than function scope
- Use in the past to create ‘modules’ of separated responsibilities
- Not required in modern Javascript
- You might encounter them in old code or isolated situations

IIFE example

```
// declare and immediately call a function
(function () {
  var message = 'Hello';
  console.log(message); // message exists in this scope
})();

console.log(message); // message is not defined
```

What is an IIFE

IIFE - MDN

Immediately invoked function expression

The many ways to write an IIFE

I Love My IIFE

Immediately-Invoked Function Expression (IIFE)

Minification

What is minification

- removing unnecessary or redundant data
- without affecting how the resource is processed by the browser
    - removing code comments
    - removing unnecessary spaces and formatting
    - removing unused code
    - using shorter variable and function names
- Possible to minify HTML, CSS and Javascript

Javascript minification

main.js

```
let greeter = (name, place) => `Mister ${name}
of${place}`;

// function with a side effect
let nameLogger = (name, place) => {
  let newName = `Mister ${name} of${place}`;
  console.log(newName);
  return newName;
};
```

main.min.js

```
let greeter=(e,r)=>`Mister ${e}
of${r}`,nameLogger=(e,r)=>{let o=`Mister ${e}
of${r}`;return console.log(o),o};
```

UglifyJS 3: Online JavaScript minifier

CSS minification

style.css

```
.geo-image {
  margin: 0 auto;
  width: 100%;
  position: relative;
}
.geo-image img {
  width: 100%;
  display: block;
}

/* image caption */
.geo-image figcaption {
  background-color: orange;
  position: absolute;
  bottom: 8px;
}
```

style.min.css

```
.geo-image{margin:0
auto;width:100%;position:relative}.geo-image
img{width:100%;display:block}.geo-image
figcaption{background-color:orange;position:absol
ute;left:-4px;bottom:8px}
```

[Minify CSS and JS]

Content Delivery Network

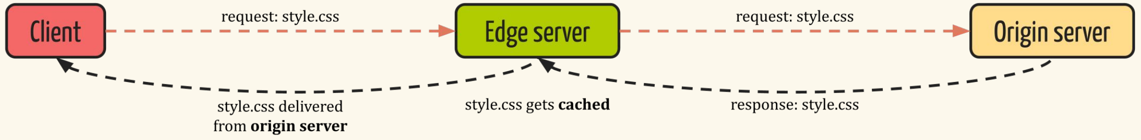
CDN - Content Delivery Network

- A large distributed system of servers deployed in multiple data centers across the Internet
- Serve content to end-users with high availability and high performance
- CDNs serve a large fraction of the Internet content today
    - web objects: text, graphics and scripts
    - downloadable objects: media files, software, documents
    - applications: e-commerce, portals
    - live streaming media
    - on-demand streaming media
    - social networks

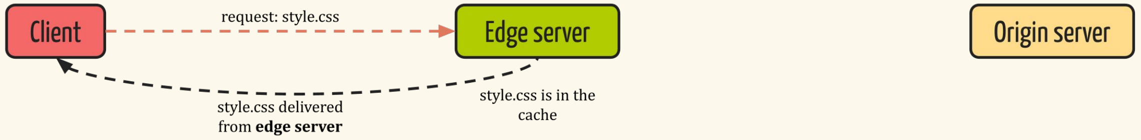
How CDNs work

Content Delivery Network

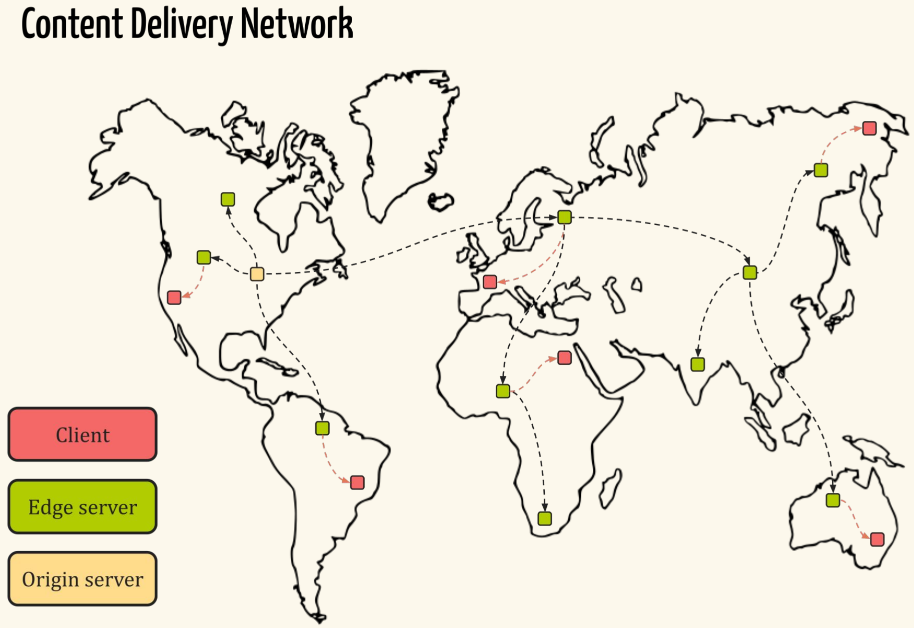
First request



Subsequent requests


Client
request: style.css
Edge server
Origin server
style.css delivered from **edge server**
style.css is in the cache

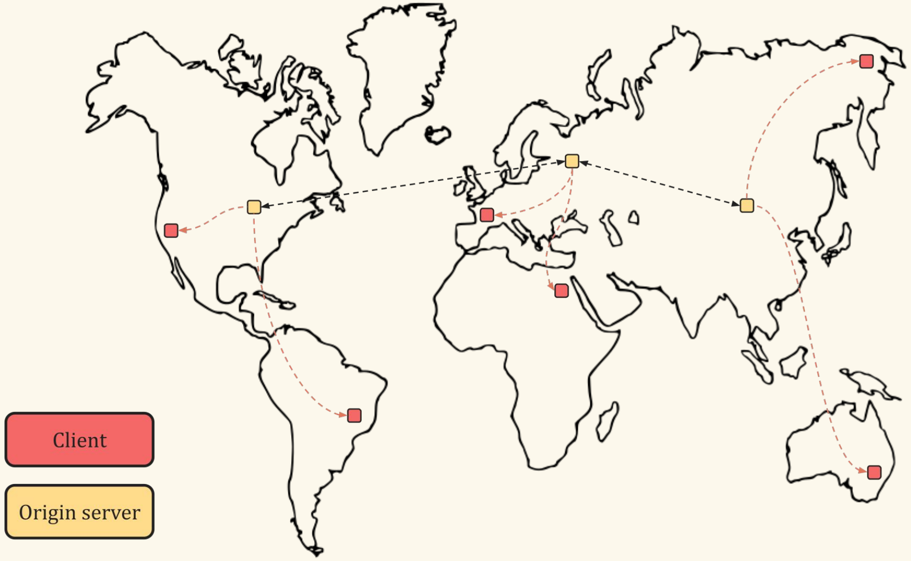
How CDNs work



© Shadi Lahham

How CDNs work

Network without a CDN



Using a CDN

```
<link href="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/css/bootstrap.min.css" rel="stylesheet">
<script src="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/js/bootstrap.bundle.min.js"></script>

<script crossorigin src="https://unpkg.com/react@17/umd/react.production.min.js"></script>
<script crossorigin src="https://unpkg.com/react-dom@17/umd/react-dom.production.min.js"></script>

<script src="https://cdnjs.cloudflare.com/ajax/libs/lodash.js/4.17.21/lodash.min.js"></script>
```

Some known CDNs
Cloudflare CDN
Google cloud CDN
Amazon CloudFront CDN
UNPKG

CDN package search
cdnjs

Build tools

Tools

Required to bundle code and also make modern code on older browsers

- NodeJS
  - package manager
  - installs packages
- Babel
  - compiler/transpiler
  - convert modern Javascript to versions compatible with older browsers
- Webpack
  - bundler
  - parses file imports to create bundles

Babel

Babel is translates modern Javascript to ES5 that older browsers understand

Technically Babel is a JS transpiler which can

- Transform syntax
- Polyfill missing features
- Perform source code transformations

Try Babel with your own code

Webpack

Webpack is a bundler that packs many JS module files into one bundle file

- Can produce one or a few bundles, e.g. bundle.js
- Supports many module systems
- Can be used for other types of files CSS, Images, JSON, SASS, etc
- Can minify and uglify Javascript code
- Managed by a configuration file called webpack.config.js and a CLI

Setup

Install NodeJS if it is not already installed

Create the following project folder and files:

```
project
├────src
|    ├──── index.html
|    ├────scripts
|    |    ├──── main.js
|    |    └──── helper.js
|    └────styles
|         └──── main.css
├────static
├──── package.json
└──── webpack.config.js
```

Modules to install

```
// run the following commands to install the required modules

npm i -D babel-loader @babel/core @babel/preset-env
npm i -D webpack webpack-dev-server webpack-cli
npm i -D html-webpack-plugin
npm i -D css-loader mini-css-extract-plugin
npm i core-js@3
```

index.html

```html
<!DOCTYPE html>
<html lang="en">

<head>
  <meta charset="utf-8">
  <meta http-equiv="X-UA-Compatible" content="IE=edge">
  <title>JS Webpack</title>
  <meta name="viewport" content="width=device-width, initial-scale=1">
</head>

<body>
  <!-- body content -->
  created with Webpack
</body>

</html>
```

Javascript files

```
// main.js
import '../styles/main.css';
import helper from './helper.js';

console.log(`I can run modern Javascript on older browsers`);
console.log(`message from helper: ${helper.msg}`);
const test = [1, 2, 3].includes(1);
console.log('test = ', test);


// helper.js
const api = {
  msg: 'I can use modules'
};
export default api;
```

package.json with webpack 5 dependencies

```json
{
  "scripts": {
    "watch": "webpack --watch",
    "start": "webpack-dev-server --open",
    "build": "webpack"
  },
  "devDependencies": {
    "@babel/core": "^7.17.7",
    "@babel/preset-env": "^7.16.11",
    "babel-loader": "^8.2.3",
    "css-loader": "^6.7.1",
    "html-webpack-plugin": "^5.5.0",
    "mini-css-extract-plugin": "^2.6.0",
    "webpack": "^5.70.0",
    "webpack-cli": "^4.9.2",
    "webpack-dev-server": "^4.7.4"
  },
  "dependencies": {
    "core-js": "^3.21.1"
  }
}
```

webpack.config.js

```
const path = require('path');
const HtmlWebpackPlugin = require('html-webpack-plugin');
const MiniCssExtractPlugin = require('mini-css-extract-plugin');

module.exports = {
  mode: 'production',
  entry: './src/scripts/main.js',
  output: { filename: '[name].bundle.js', path: path.resolve(__dirname, 'dist') },
  devServer: { static: { directory: path.join(__dirname, 'static'),publicPath: './static' } },
  plugins: [
    new HtmlWebpackPlugin({ template: './src/index.html' }),
    new MiniCssExtractPlugin()
  ],
```

webpack.config.js -- continued

```
  module: {
    rules: [
      { test: /\.css$/i,
        use: [MiniCssExtractPlugin.loader, 'css-loader'] },
      { test: /\.m?js$/,
        exclude: /(node_modules|bower_components)/,
        use: {
          loader: 'babel-loader',
          options: {
            presets: [[
                '@babel/preset-env',
                { targets: { edge: '80', firefox: '74', chrome: '80', safari: '13', ie: '11' },
                  useBuiltIns: 'usage',
                  corejs: '3.6.4' }
              ]]
          }
        }
      }
    ]
  }
};
```

Running

install dependencies (wait for installation to finish):

```
npm install
```

run webpack dev server:

```
npm start
```

build project bundle:

```
npm run build
```

testing:

test code on different browsers, especially on Edge in Internet Explorer mode

[What is Internet Explorer (IE) mode?]

[How to enable IE mode on Microsoft Edge]

Your turn

1.To CDN or not to CDN

Find sources link this article to understand the pros and cons of a CDN

- Write down as many pros and cons as you can think of
  - Explain why you think they are relevant
- Describe 2 scenarios where you think a CDN is required and 2 where it's not
  - Your examples should be realistic and should emphasize the pros or cons

Summarize your findings in a properly named markdown file

  - Markdown Guide
  - Online Markdown Editor - Dillinger

Bonus

2. IE friendly

- Implement some of exercises of the previous units as a webpack project
- The aims are
  - to rewrite the same exercises with modern JS syntax
  - to use webpack (and polyfills if necessary), to make the code compatible with IE11
- Document any important configuration or code changes in readme.md
- Test the projects in Edge using Internet Explorer mode

What is Internet Explorer (IE) mode?

How to enable IE mode on Microsoft Edge

References

Import

Export

Global variables and JavaScript modules

Global Variables in JavaScript

References

webpack

DevServer

Development

Output

References

webpack

HtmlWebpackPlugin

css-loader

sass-loader

References

Webpack 5 : Guide for beginners

Setting up the Webpack Dev Server

How to Webpack 5 - Setup Tutorial

Set up Webpack 5 for Basic Javascript Projects

Setting Up Webpack for JavaScript, TypeScript and using Webpack Server