Webpack

Version 4.x

Agenda

1 OVERVIEW

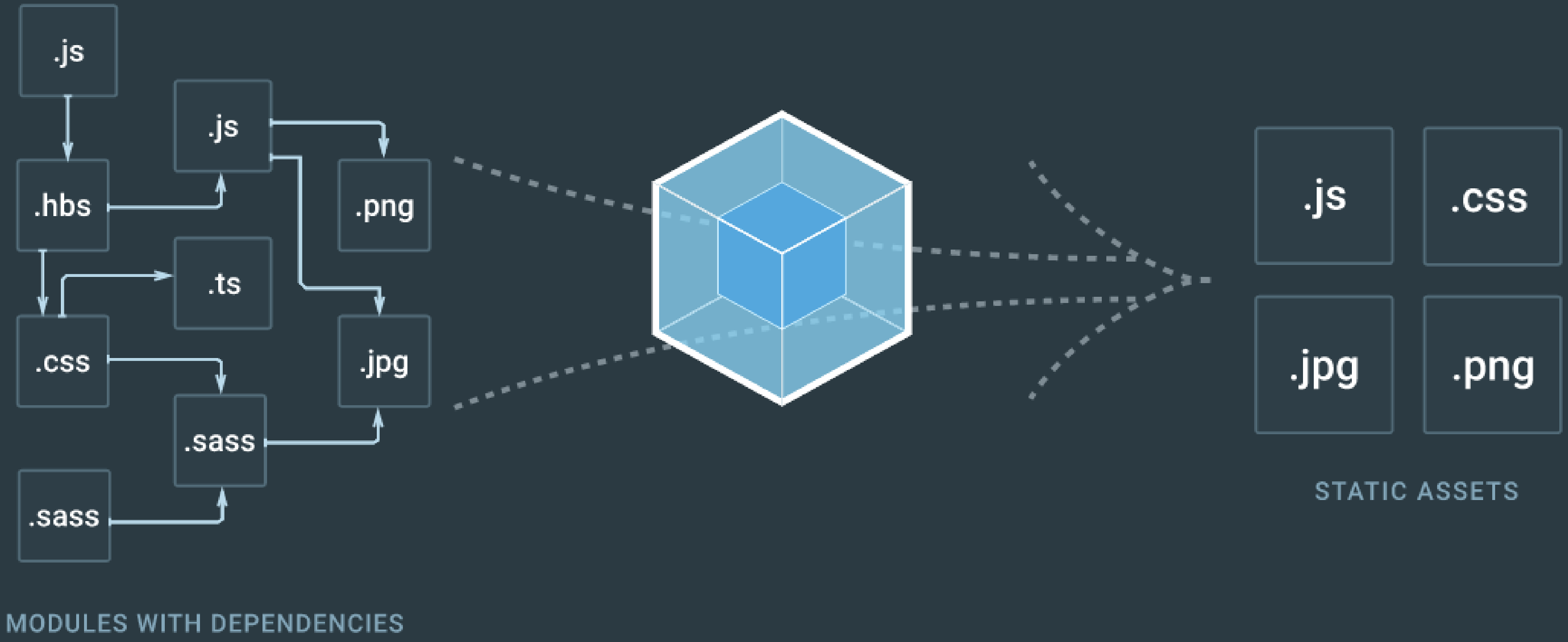
2 WEBPACK CONFIG

3 PLUGINS

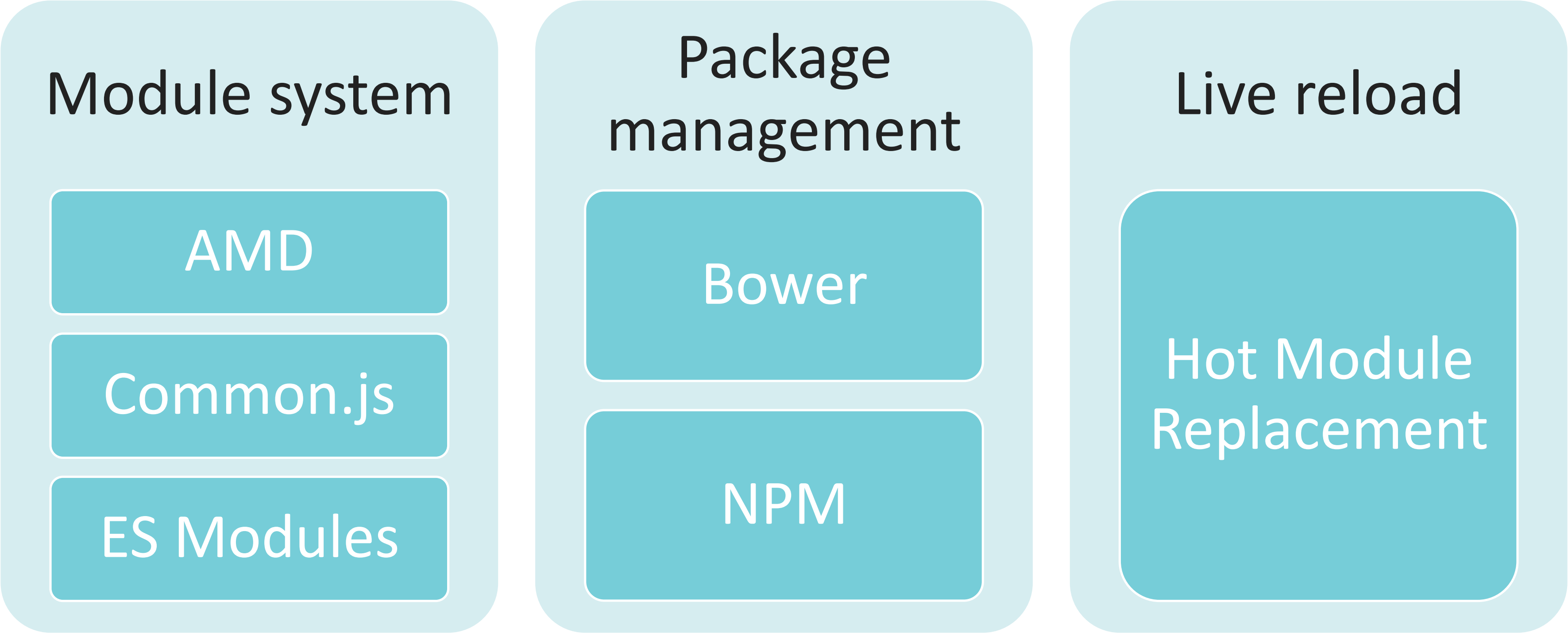
4 LOADERS

5 WEBPACK DEV SERVER

OVERVIEW


.js
.js
.hbs
.png
.ts
.css
.jpg
.sass
.sass
MODULES WITH DEPENDENCIES
.js
.css
.jpg
.png
STATIC ASSETS

Quick features overview



Module system

AMD

Common.js

ES Modules

Package management

Bower

NPM

Live reload

Hot Module Replacement

NOT LONG TIME AGO…

Without modules

- Have to know "correct" order (inter-dependencies)
- Have to make sure correct dependencies are added (on each page)
- Inefficient fetching (HTTP 1.1 concurrency limit)
- No package management, meaning you have to copy/paste in prebuilt vendor code
- ...

```html
<script src="js/jquery.js"></script>
<script src="js/jquery-ui.js"></script>
<script src="js/some-jquery-plugin.js"></script>
<script src="js/react.js"></script>
<script src="js/other.js"></script>
<script src="js/home.js"></script>
```

JavaScript module pattern

- The Revealing Module Pattern is one of the most popular ways of creating modules.
- Using the return statement we can return a object literal that ‘reveals’ only the methods or properties we want to be publicly available.

```
var modularpattern = (function () {
    // your module code goes here
    var sum = 0;

    return {
        add: function () {
            sum = sum + 1;
            return sum;
        },
        reset: function () {
            return sum = 0;
        }
    }
}());

alert(modularpattern.add());    // alerts: 1
alert(modularpattern.add());    // alerts: 2
alert(modularpattern.reset());  // alerts: 0
```

AMD modules and require.js

- Inefficient fetching (HTTP 1.1 concurrency limit)
- No package management, meaning you have to copy/paste in prebuilt vendor code
- One more lib in your code

main.js

```
define(function (require) {
    // Load any app-specific modules
    // with a relative require call,
    // like:
    var messages = require('./messages');

    print(messages.getHello());
});
```

messages.js

```
define(function () {
    return {
        getHello: function () {
            return 'Hello World';
        }
    };
});
```

The webpack way

- Webpack is a module bundler, not a task runner.
- Everything is a module
- Analyzes dependencies among your modules (not only JS but also CSS, HTML, etc.) and generates assets.
- Understands multiple standards for how to define dependencies and export values: AMD, CommonJS, ES modules and so on.

App.jsx

```
import * as React from 'react';

export class App extends React.Component {
    public render() {
        return <div>Hello</div>
    }
}
```

WEBPACK INTRO

Goals

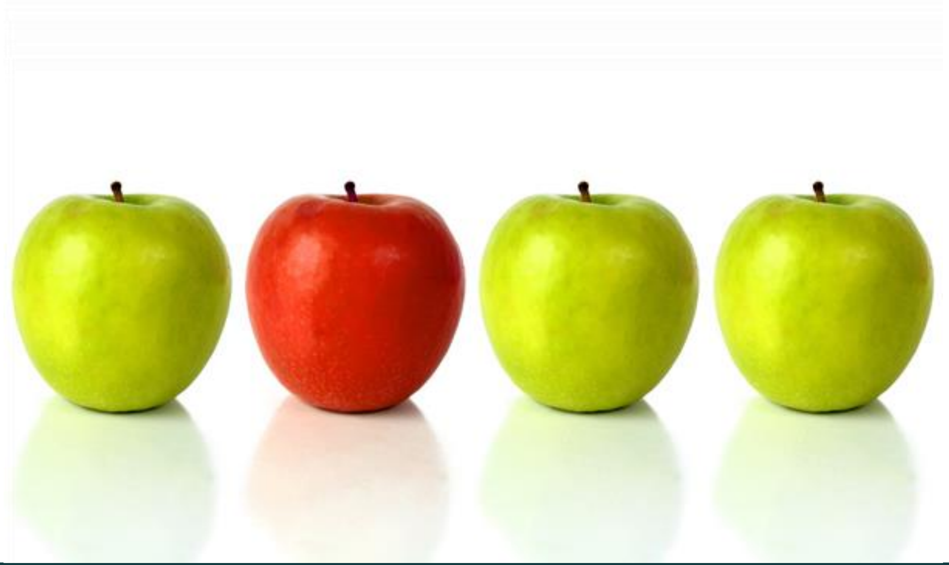
WEBPACK WAS DEVELOPED WITH THE FOLLOWING AIMS:

- Code splitting and on demand loading
- Low initial load time
- Every static asset as a module
- 3rd-party libraries as modules
- Suited for big projects

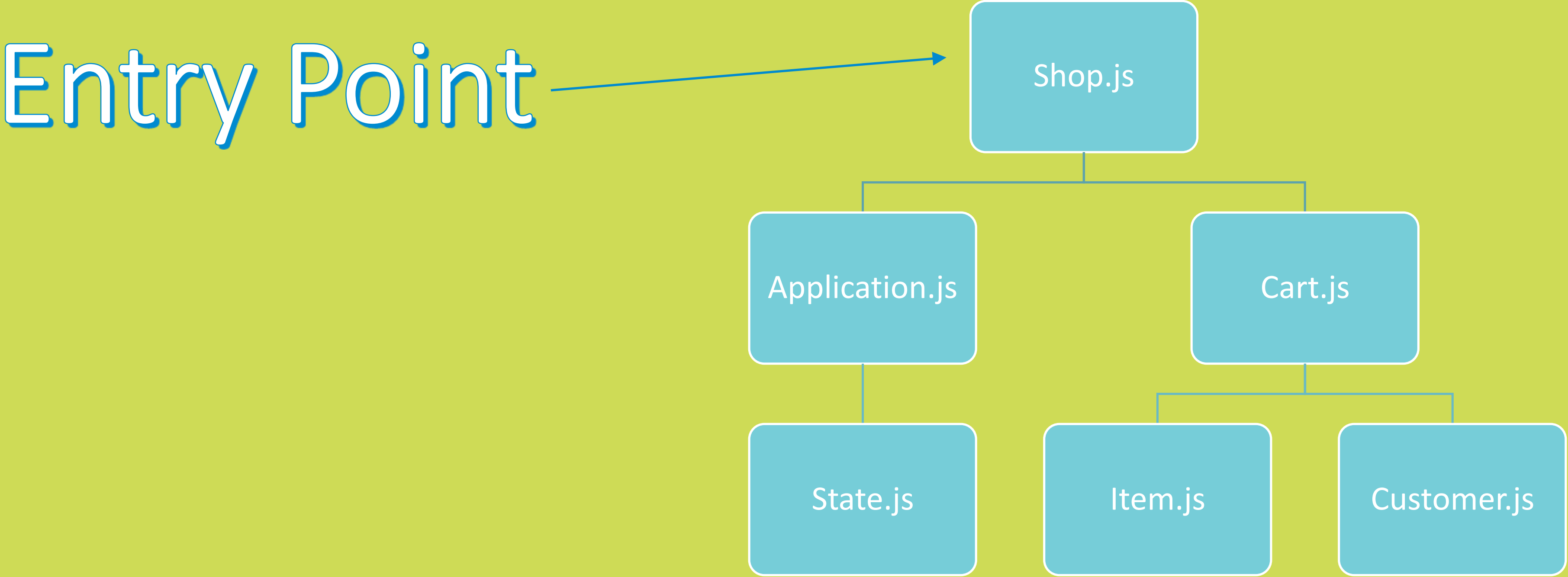
Distinctions

WEBPACK HAS VIVID DISTINCTIONS THAT DIFFERS IT FROM OTHERS

- Can use dependency management (npm)
- Every module declares their own dependencies, so the bundler can build the dependency graph
- No more globals (unless you specifically declare them)
- Everything always loads in the correct order
- Enables you to test each module in isolation



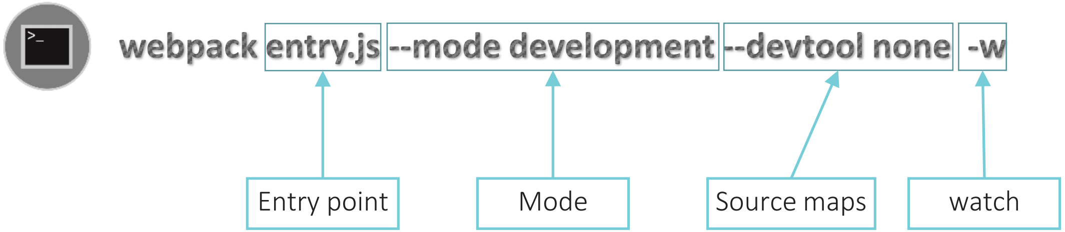
Webpack uses dependency resolution to build bundle



Webpack installation

```
npm i -g webpack webpack-cli
```

WEBPACK CLI



```
webpack entry.js --mode development --devtool none -w
```

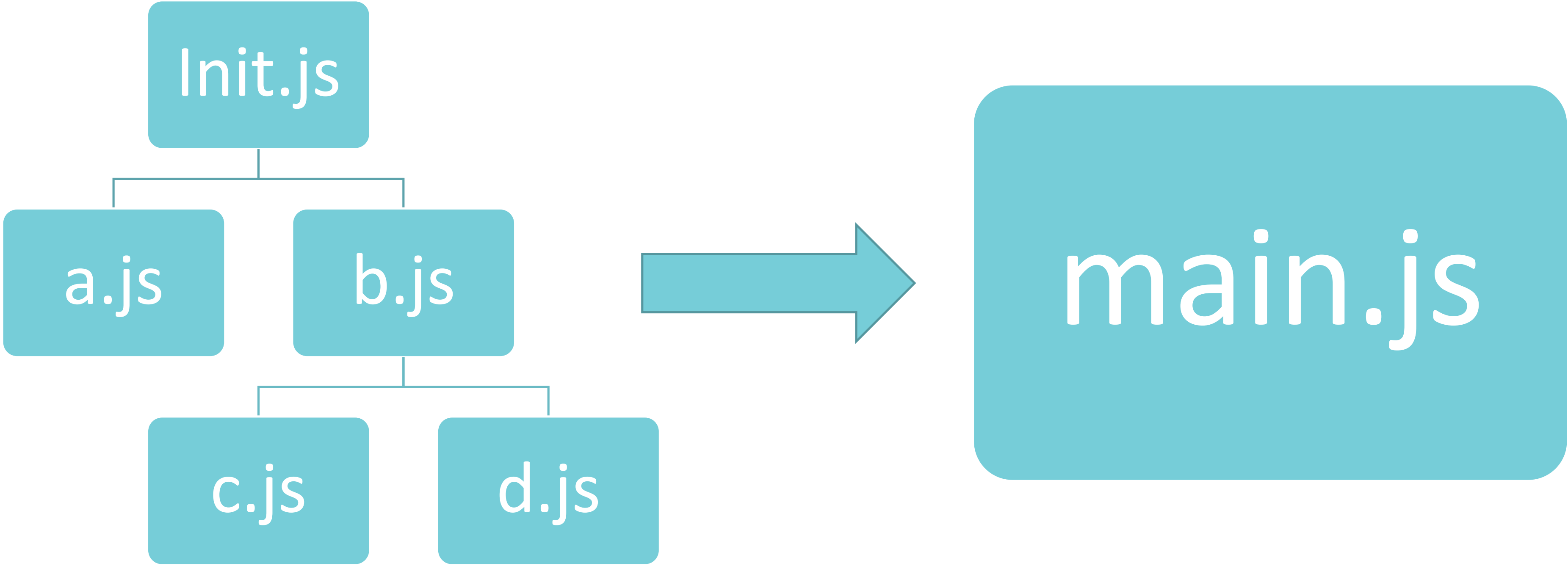
Webpack Comand Line Interface has several important options:

- Entry point – root JavaScript file (required)
- Mode (required)
  - development – speedup build
  - production – reduce bundle size
- Source map – can be on of https://webpack.js.org/configuration/devtool/. None means original source
- Watch – rebuild on file change (optional)

First try

1.first try

```
webpack init.js --mode development
```



Example with NPM

2-npm

- Webpack recognizes modules from “*node_modules*” and can add them to bundle

```
npm i
```

```
webpack example.js --mode development
```

example.js

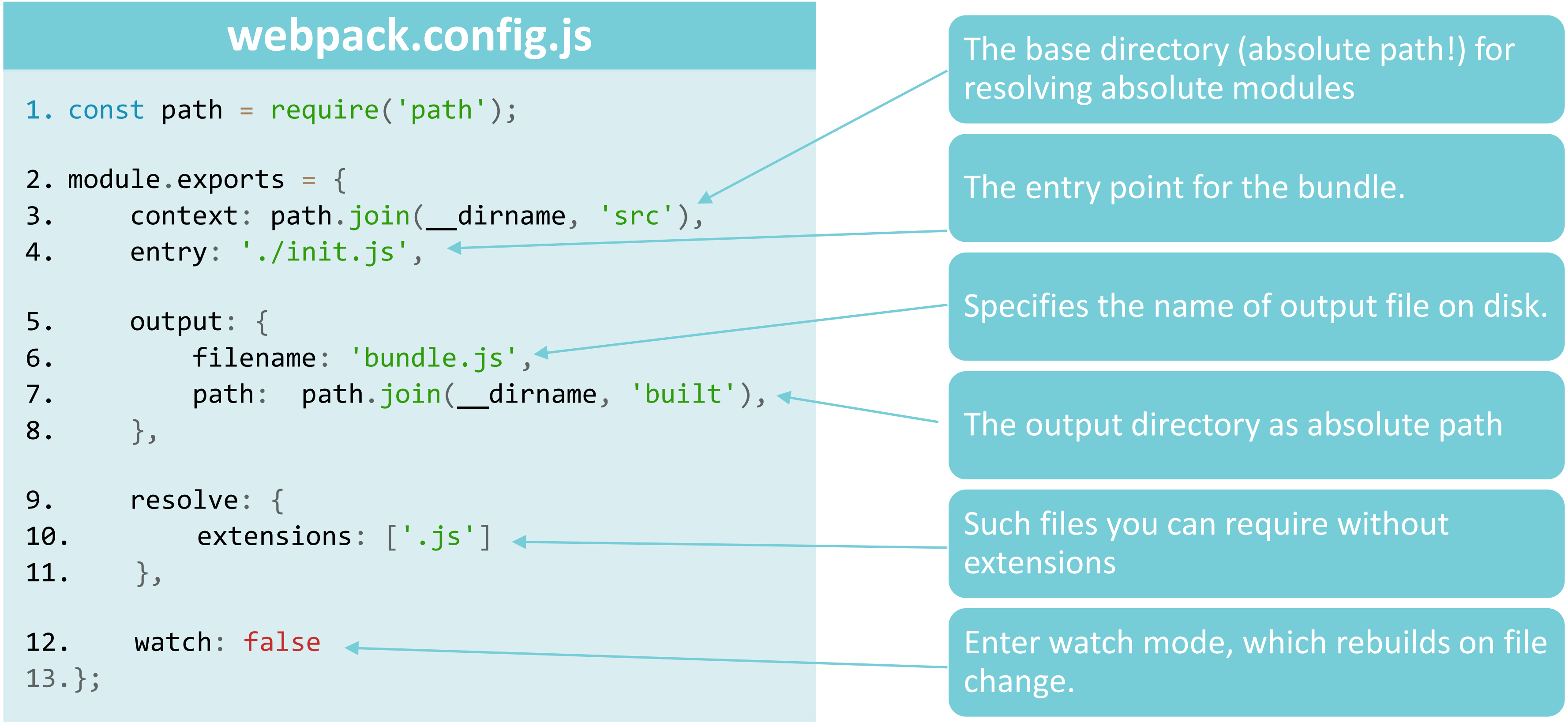
```js
import * as _ from 'lodash';

const result = _.uniq([2, 1, 2]);
console.log(result); // => [2, 1]
```

WEBPACK CONFIG FILE

Config file

3-config



webpack.config.js

```
const path = require('path');

module.exports = {
    context: path.join(__dirname, 'src'),
    entry: './init.js',

    output: {
        filename: 'bundle.js',
        path:  path.join(__dirname, 'built'),
    },

    resolve: {
         extensions: ['.js']
     },

     watch: false
};
```

The base directory (absolute path!) for resolving absolute modules

The entry point for the bundle.

Specifies the name of output file on disk.

The output directory as absolute path

Such files you can require without extensions

Enter watch mode, which rebuilds on file change.

0CJS

4-0CJS

- 0CJS basically means a Zero Config app. It's the idea that you need the minir or zero config to get a JavaScript project up and running.
- With version 4, webpack now has a platform for zero config, this means the would be no need for a webpack.config.js file.
- All you need to do is have a ./src/index.js file and whenever you run the wel command in the terminal, webpack knows to use that file as an entry point the application. This might come in handy for small projects.

Multiple entry points

5-multiple-entry-points

- To use multiple entry points you can pass an object to the entry option.
- Each value is treated as an entry point and the key represents the name of the entry point
- Dependency to an entry point is not allowed
- Pay attention to `filename`, it differs from the previous one as we have several entry points but they can’t have the same name

webpack.config.js

```
const path = require('path');

module.exports = {
    context: path.join(__dirname, 'src'),
    entry: {
        home: "./home",
        order: "./order",
        profile: "./profile",
        shop: "./shop"
    }
}
```

Common Chunks

6-split-chunks

Webpack searches for common modules between entry points

- minSize (default: 30000) Minimum size for a chunk.
- minChunks (default: 1) Minimum number of chunks that share a module before splitting

webpack.config.js

```
module.exports = {
…
  optimization: {
    splitChunks: {
      cacheGroups: {
        commons: {
          name: "commons",
          chunks: "all",
          minSize: 0,
          minChunks: 2
        }
      }
    }
  }
};
```

Plugins

Plugins

Plugins are used to customize the webpack build process.

Plugins aims:

- Automatically pull out common stuff to one or more separate files (code splitting)
- Minification
- Hot module replacement (live reload)
- And so on

webpack.config.js

```
module.exports = {
    entry: {
        app: './src/app.js',
        search: './src/search.js'
    },
    output: {
        filename: '[name].js',
        path: __dirname + '/built'
    },
    plugins: [new SomePlugin()]
};
```

Sensitive paths

7-case-sensitive-paths-webpack-plugin

This is a plugin for Webpack that forces import statements to match case with the target file on the disk.

webpack.config.js

```
const path = require('path');
const CaseSensitivePathsPlugin = require('case-sensitive-paths-webpack-plugin');

module.exports = {
    entry: './src/init.js',

    output: {
        path: path.join(__dirname, "built"),
        filename: 'index_bundle.js',
    },

    plugins: [
        new CaseSensitivePathsPlugin()
    ]
};
```

DefinePlugin

8-define-plugin

The DefinePlugin allows you to create global constants which can be configured at compile time.

webpack.config.js

```
const path = require('path');
const webpack = require('webpack');

module.exports = {
 …
    plugins: [
        new webpack.DefinePlugin({
            VERSION: JSON.stringify("5fa3b9"),
            BROWSER_SUPPORTS_HTML5: true,
         })
    ]
};
```

bundle.js

```
…
console.log("Running App version " + "5fa3b9");

if (true) {
    console.log("Browser is GOOD");
}
…
```

ProvidePlugin

9-provide-plugin

This is a webpack plugin that simplifies creation of HTML files to serve your webpack bundles

webpack.config.js

```javascript
const path = require("path");
const webpack = require("webpack");

module.exports = {
…
  plugins: [
    new webpack.ProvidePlugin({
      _: "lodash"
    })
  ]
};
```

example.js

```javascript
const result =_.intersection([2, 1], [2, 3]);
console.log(result);  // => [2]
```

HTML Webpack Plugin

10-html-webpack-plugin

This is a webpack plugin that simplifies creation of HTML files to serve your webpack bundles

webpack.config.js

```
const path = require('path');
const webpack = require('webpack');

module.exports = {
    entry: 'index.js',
    output: {
        path: 'dist',
        filename: 'index_bundle.js'
    },
    plugins: [new HtmlWebpackPlugin()]
};
```

Index.html

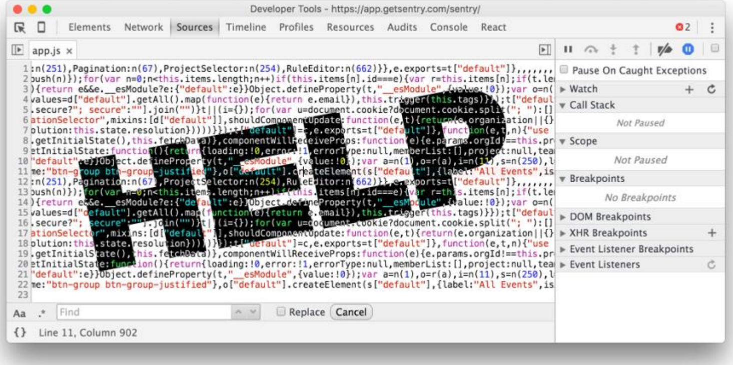
```
<!DOCTYPE html>
<html>
<head>
    <meta charset="UTF-8">
    <title>Webpack App</title>
</head>
<body>
    <script src="index_bundle.js"></script>
</body>
</html>
```

Source maps

11-source-maps

IN ORDER TO USE SOURCE MAP YOU SHOULD CHANGE “DEVTOOL” PROPERTY IN CONFIG TO THE VALUE WHICH AVAILABLE IN THIS LIST:

- eval (default in “development” mode)
- eval-source-map
- source-map
- more in docs



```
:n(251),Pagination:n(67),ProjectSelector:n(254),RuleEditor:n(662)}},e.exports=t["default"]},,,,,,,
ush(n)}});for(var n=0;n<this.items.length;n++)if(this.items[n].id===e){var r=this.items[n];if(t.le
){return e&&e.__esModule?e:{"default":e}}Object.defineProperty(t,"__esModule",{value:!0});var o=n(
values=d["default"].getAll().map(function(e){return e.email}),this.trigger(this.tags)}});t["defaul
.secure?"; secure":""].join("")}t||(i={});for(var u=document.cookie?document.cookie.split("; "):[]
ationSelector",mixins:[d["default"]],shouldComponentUpdate:function(e,t){return(e.organization||{}
olution:this.state.resolution}))))}});t["default"]=c,e.exports=t["default"]},function(e,t,n){"use
.getInitialState(),this.fetchData)},componentWillReceiveProps:function(e){e.params.orgId!==this.pr
etInitialState:function(){return{loading:!0,error:!1,errorType:null,memberList:[],project:null,tea
"default":e}}Object.defineProperty(t,"__esModule",{value:!0});var a=n(1),o=r(a),i=n(11),s=n(250),l
me:"btn-group btn-group-justified"},o["default"].createElement(s["default"],{label:"All Events",is
:n(251),Pagination:n(67),ProjectSelector:n(254),RuleEditor:n(662)}},e.exports=t["default"]},,,,,,,
ush(n)}});for(var n=0;n<this.items.length;n++)if(this.items[n].id===e){var r=this.items[n];if(t.le
){return e&&e.__esModule?e:{"default":e}}Object.defineProperty(t,"__esModule",{value:!0});var o=n(
values=d["default"].getAll().map(function(e){return e.email}),this.trigger(this.tags)}});t["defaul
.secure?"; secure":""].join("")}t||(i={});for(var u=document.cookie?document.cookie.split("; "):[]
ationSelector",mixins:[d["default"]],shouldComponentUpdate:function(e,t){return(e.organization||{}
olution:this.state.resolution}))))}});t["default"]=c,e.exports=t["default"]},function(e,t,n){"use
.getInitialState(),this.fetchData)},componentWillReceiveProps:function(e){e.params.orgId!==this.pr
etInitialState:function(){return{loading:!0,error:!1,errorType:null,memberList:[],project:null,tea
"default":e}}Object.defineProperty(t,"__esModule",{value:!0});var a=n(1),o=r(a),i=n(11),s=n(250),l
me:"btn-group btn-group-justified"},o["default"].createElement(s["default"],{label:"All Events",is
```

{} Line 11, Column 902

Loaders

Loaders

EACH LOADER IS AN OBJECT IN THE ARRAY WITH THREE MAIN PROPERTIES

- The test property tells the loader which file type to check for.
- The exclude property tells the loader which folders to stay away from or alternatively there is an include property which specifies which to stay in.
- The loader property is a string value which corresponds to the loader in your node_modules folder.

webpack.config.js

```
const path = require('path');
module.exports = {
    context: path.join(__dirname, 'src'),
    entry: {
        home: "./Home",
        order: "./Order"
    },
    output: {
        path: path.join(__dirname, "built"),
        filename: "[name].js"
    },
    resolve: {
        extensions: ['.ts', '.js']
    },
    module: {
        rules: [{
            test: /\.ts$/,
            loader: 'awesome-typescript-loader',
            options: {
                useCache: true
            }
        }]
    }
}
```

Typescript Loader

12-awesome-typescript-loader

- Parse TypeScript
- This loader was created mostly to speed-up compilation in my own projects.

webpack.config.js

```
module.exports = {
    // Currently we need to add '.ts' to the resolve.extensions array.
    resolve: {
        extensions: ['.ts', '.tsx', '.js', '.jsx']
    },
      // Add the loader for .ts files.
    module: {
        rules: [
          {
              test: /\.tsx?$/,
              loader: 'awesome-typescript-loader'
          }
        ]
    }
};
```

Babel Loader

13-babel

- Parse modern JavaScript

webpack.config.js

```
module.exports = {
    // Currently we need to add '.jsx' to the resolve.extensions array.
    resolve: {
        extensions: ['.js', '.jsx']
    },
    // Add the loader for .js files.
    module: {
        rules: [
          {
              test: /\.jsx?$/,
              loader: 'babel-loader'
          }
        ]
    }
};
```

CSS

14-css

In order to use CSS you have to use two loaders: style-loader and css-loader and one more plugin ExtractTextPlugin

webpack.config.js

```
const path = require('path');
const ExtractTextPlugin = require("extract-text-webpack-plugin");

module.exports = {
…
    module: {
        rules: [{
            test: /\.css$/,
            use: ExtractTextPlugin.extract({ fallback: "style-loader", use: "css-loader" })
        }]
    },
    plugins: [
        new ExtractTextPlugin("[name].css")
    ]
}
```

LESS

15-less

In order to use LESS you have to add less loader

webpack.config.js

```
const path = require('path');
const ExtractTextPlugin = require("extract-text-webpack-plugin");

module.exports = {
…
    module: {
        rules: [{
            test: /\.less$/,
            use: ExtractTextPlugin.extract({ fallback: 'style-loader', use: ['css-loader', 'less-loader'] })
        }]
    },
    plugins: [
        new ExtractTextPlugin({ filename: 'style.css' })
    ]
}
```

File loader

16-file-loader

By using this loader you can require any file

webpack.config.js

```
const path = require('path');

module.exports = {
…
    module: {
        rules: [{
            test: /\.(ttf|eot|svg|woff|png)$/,
            loader: "file-loader",
            options: {
                name: '[path][name].[ext]?[hash]'
            }
        }]
    }

}
```

Dev Server

Dev server

17-dev-server

The webpack-dev-server is a little node.js Express server, which uses the webpack-dev-middleware to serve a webpack bundle.

Install dev server globally

```
npm install –g webpack-dev-server
```

Run server where “webpack.config.js” is placed

```
webpack-dev-server
```

Hot module replacement

18-hot-module-replacement

Hot Module Replacement (HMR) exchanges, adds, or removes modules while an application is running without a page reload.

It can be applied for example to styles

webpack.config.js

```
module.exports = {
…
    plugins: [
        new webpack.HotModuleReplacementPlugin()
    ],

    devServer: {
        hot: true,
    }
};
```

Thanks for your attention