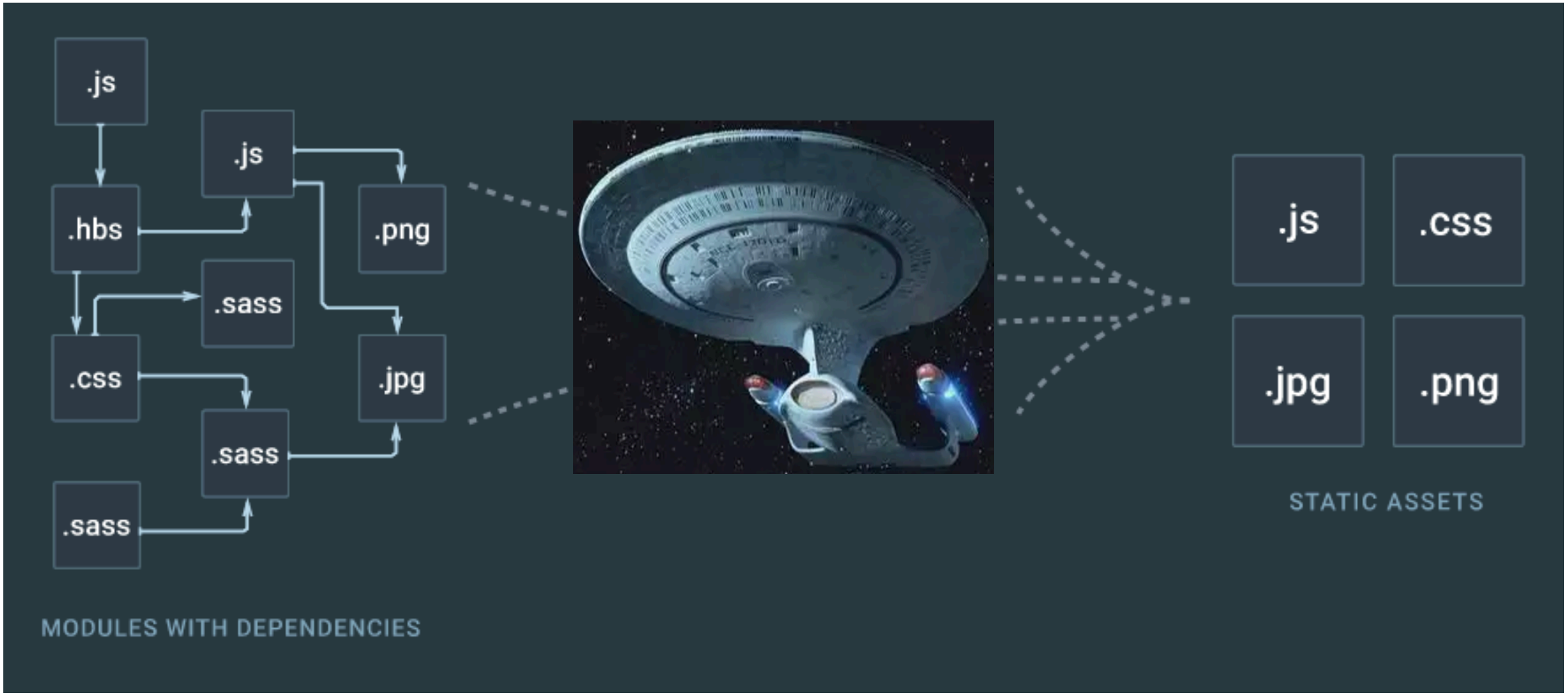
Webpack

The Next Generation

Learnings while configuring Webpack for our team

What is Webpack?



Why Use Webpack?

- Reduce HTTP request overhead
- Long term caching
- Code splitting
- File Transforms
- All static files are picked up by webpack (js, png, html, css, hbs, etc…)
- Webpack Dev Server
- Provides unparalleled level of automation to your build process

```
plugins: [
    new webpack.optimize.CommonsChunkPlugin({
        names: ['vendor', 'manifest'],
        minChunks: Infinity
    }),
    new HtmlWebpackPlugin({
        template: path.join(__dirname, "/public/index.html"),
        chunks: ["vendor", "main", "manifest"] // bundles injected to template
    }),
    new ExtractTextPlugin('[name].[contenthash:5].css')
],
devServer: {
    contentBase: path.join(__dirname, 'build'),
    port: 3000,
    publicPath: '/',
    setup: function (app) {
        app.get('/api/data', function(req, res) {
            res.json({some: 'data'});
        });
    },
    stats: {
        chunks: false // gets rid of noisy chunks output
    }
},
devtool: 'cheap-eval-source-map',
resolve: {
  extensions: ['.js', '.css'],
  alias: {
    mainCSS$: path.resolve(__dirname, 'static-content/css/deep/down/in/folders/main.css')
  }
}
```


That’s a lot of Config

Four Key Concepts

Entry

- The root of your application
- Where should webpack start looking for dependencies?

Output

- Where should bundled code be emitted?
- What is the bundle's name?

Loaders

- Perform actions (transformations) on **files** of a particular type

Plugins

- Perform actions on **chunks** of your bundled output.
- CommonsChunkPlugin
  ExtractTextPlugin

Learning 1: Resolve

- Give commonly used module paths **an alias**

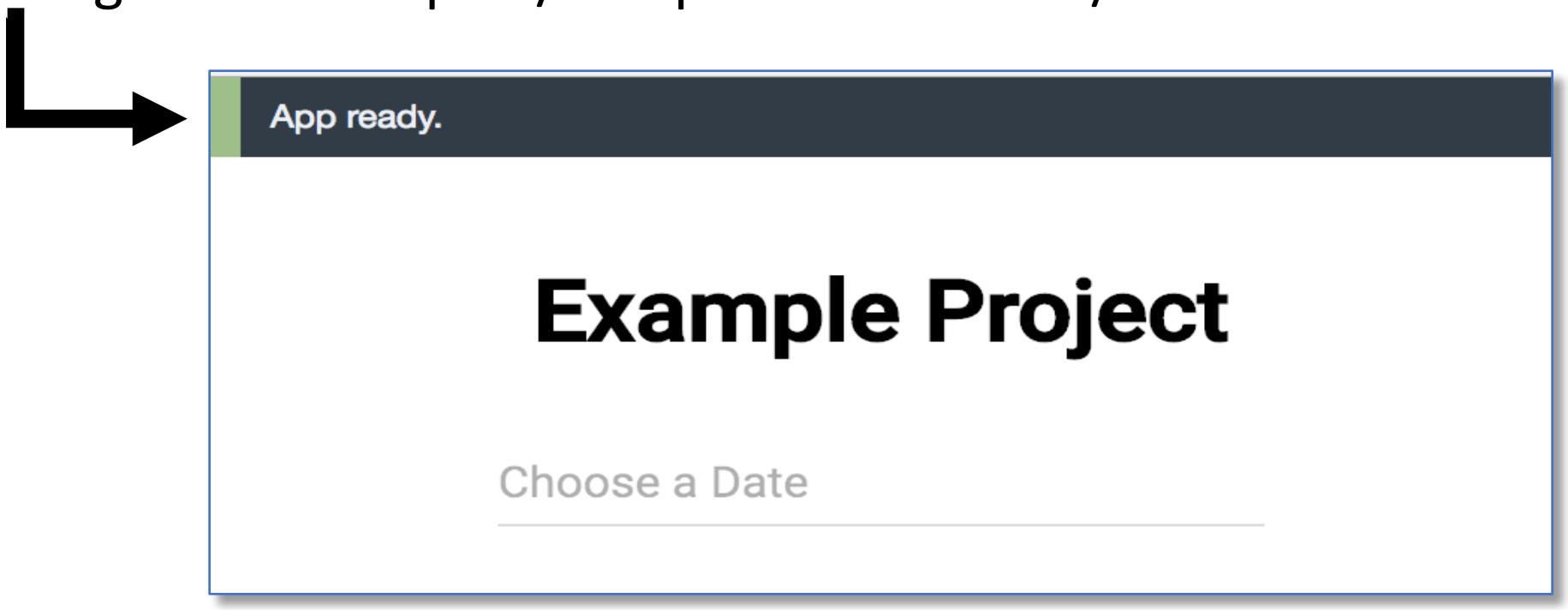
- Reduces the occurrence of ../../../../folder1/folder2/filename.js **import pasta** in your app

```
resolve: {
  extensions: ['.js', '.css'],
  alias: {
    mainCSS$: path.resolve(__dirname, 'static-content/css/deep/down/in/folders/main.css')
  }
}
```

Learning 2: webpack-dev-server

- Proxy requests to different domains through webpack-dev-server
  - Avoid CORS warnings on requests

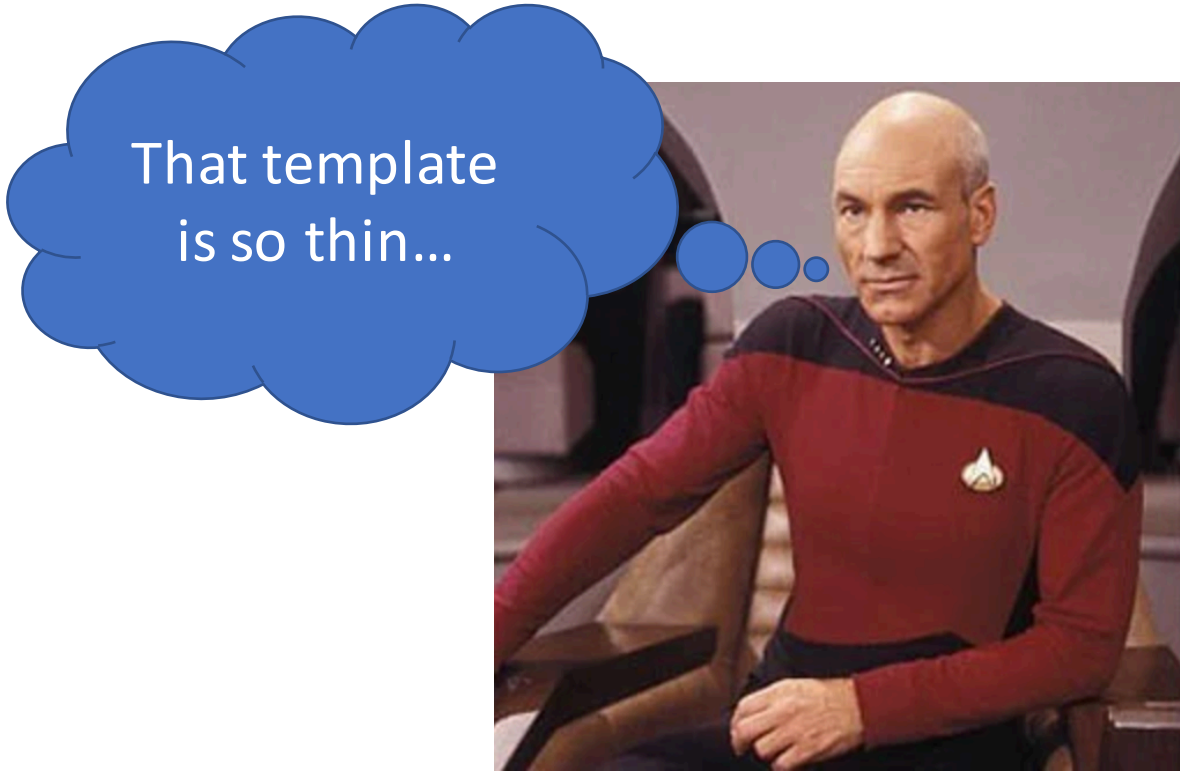
- Navigate to base:port/webpack-dev-server/ for build status in browser!



Learning 3: HtmlWebpackPlugin

- HtmlWebpackPlugin injects bundles in to your initial html template!
  - Scripts injected at the end of body, CSS injected in head

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8"/>
        <title>Webpack Example</title>
    </head>
    <body>
        <div id="root"></div>
    </body>
</html>
```



Learning 4: manifest chunk

- Webpack was adding manifest map of chunks to bundles to vendor bundle… vendor bundle was changing on every main bundle update

- Move manifest to CommonsChunkPlugin, splits it out!

```
                                    Asset       Size  Chunks                    Chunk Names
  vendor.5e134b46b174d87a301a.bundle.js    2.78 MB       0  [emitted]  [big]  vendor
    main.e9a81ba52eb8168f3df1.bundle.js    80.9 kB       1  [emitted]         main
manifest.70625f8e7535eee0f463.bundle.js    5.93 kB       2  [emitted]         manifest
                           index.html  418 bytes          [emitted]
Child html-webpack-plugin for "index.html":
         Asset     Size  Chunks  Chunk Names
    index.html  1.42 MB       0
webpack: Compiled successfully.
```

”Talk is cheap…
show me the code!”

~Puxuan He

https://github.com/bambielli/webpack-example

Webpack 2

- Released out of beta last February
- No more transpiling ES6 imports to CommonJS
- **Tree Shaking** – dead code removal
- Some breaking changes to the API…

https://webpack.js.org/guides/migrating/


WEBPACK