Basics

https://webpack.js.org/guides/migrating/
themes/{your theme}/webpack.config.js is the file we are editing to migrate webpack.
Search that link above with the text mentioned here if you want more information about these changes

This page was also useful for the common migration problems
http://javascriptplayground.com/blog/2016/10/moving-to-webpack-2/

Webpack v3 was released literally 8 days ago on release channel, so we will be updating to v3 instead as there are no large syntactic changes unlike v1 → v2.
https://github.com/webpack/webpack/releases/tag/v3.0.0

Node needs to be updated for Webpack v3. It's a good idea to do this anyway as node doesn't really change syntactically between versions, mostly under the hood. We are currently node 4.2.6. Upgrading to node 8.x.x as it a LTS version. Node 4.x.x will stop being maintained as of April '18, so it is a good idea to upgrade to the LTS version now so that it is supported for the foreseeable future https://nodejs.org/en/blog/release/v8.0.0/

- Not many changes between v4 and v5
    - https://github.com/nodejs/node/wiki/Breaking-changes-between-v4-and-v5
- Went through changes between v5 and v6 too, all backend stuff that you guys probably haven't had to deal with. Things like allowing more path variations etc
    - https://github.com/nodejs/node/wiki/Breaking-changes-between-v5-and-v6

Updating NPM

You must add the node v8 package source, then upgrade through apt-get

- curl -sL https://deb.nodesource.com/setup_8.x | sudo -E bash -
- sudo apt-get upgrade -y nodejs

node -v should then be v8

NPM checks

You may need to give npm access to network to check versions on update, I had to anyway. Either prepend the following with sudo, or do the chown request it asks.
For this process, I basically went along the requirements for different loaders and updated them to latest versions.

- See out of date packages
    - npm outdated
- Install webpack latest
    - npm install --save-dev webpack@3
- Install webpack-dev-server latest
    - npm install --save-dev webpack-dev-server@2
- Other extra packages we had to update
    - extract-text-webpack-plugin@2
    - esline-loader@1.8.0
    - autoprefixer-loader@3
    - url-loader@0.5.9
    - sass-loader@6
    - babel-eslint@7
    - babel-loader@7.1
    - babel-core@6
    - less-loader@4
    - node-sass@4
- Checking unused dependencies NOT DONE
    - npm install depcheck
    - https://github.com/depcheck/depcheck
    - It would be a better idea to do this with you around so that I know what to remove where it comes to less, svg and other loaders that are all connected to webpack.

Code Migration

Breaks

- resolve.modulesDirectories → resolve.modules
- ExtractTextPlugin.extract() and new ExtractTextPlugin()
    - Now takes an options object as first parameter. Updated styleLoader func and plugins list
- 'Options' inside the module.exports for loaders must not be defined in exports anymore. Instead they are now loaded through an options property inside the webpack.LoaderOptionsPlugin. Minimise moved from UglifyJs to LoaderOptionsPlugin. Context passed along to loaders
    - ```
      new webpack.LoaderOptionsPlugin({
          minimize: true,
          options: {
              eslint: {configFile: '.eslintrc'},
               context: __dirname
          }
      })
      ```
      ```

- Preloaders no longer exist. It was not using it, but there is one eslint loader that was commented out using the preloaders, so I have made it work if that one is uncommented so you know how it works.
    - Moved to be same list as loaders inside module and has enforce: “pre” property
- Module-bind in server.js doesn’t exist anymore. The functionality is in plugins anyway, commenting out.
- No longer okay to omit -loader. Expose-loader and babel-loader changed.
- Babel-polyfill installed and babel removed, as per babel documentation. Babel-polyfill added to entry and removed from source/index.js
- Devtool should not be false if not using devtools on prod

New defaults

- UglifyJsPlugin no longer switches loaders into minimize mode.
    - Add webpack.LoaderOptionsPlugin to production Plugins as uglifyJs only used in production
- The compress.warnings option of the UglifyJsPlugin now defaults to false instead of true, hiding uglifyjs warnings by default.
    - Set compress.warnings of UglifyJs to true in production plugins
- Devtool source-map, so UglifyJsPlugin has sourceMap: true as now defaults to false

Possible changes

- Loaders can now be defined in lists instead of using ‘!’ to separate loaders. This can be done by changing module.loaders to module.rules and doing the following changes:
    - ```
      {
        test: /\.css$/,
        loader: ‘style-loader!css-loader!' + autoPrefixLoader()
      },
      ```
      becomes
    - ```
      {
        test: /\.css$/,
        use: [‘style-loader’, ‘css-loader’, ‘autoprefixer-loader?whatever’]
      },
      ```
- https://medium.com/webpack/webpack-3-official-release-15fd2dd8f07b webpack 3 changes and additions. I’m sure there’s a bunch of these as well as webpack 2 things we could use.