Webpack性能优化与原理分析

> 性能分析篇

1. 深度treeshaking webpack-deep-scope-plugin webpack-parallel-uglify-plugin purifycss-webpack
2. 开启多核压缩 happypack 多线程编译webpack 不支持的情况下使用thread-loader ,
   JavaScript的多核压缩可以开启terser-webpack-plugin (多核压缩uglifyjs-webpack-plugin 官方维护
   非官方维护webpack-parallel-uglify-plugin)

```
const TerserJSPlugin = require('terser-webpack-plugin');
module.exports = {
  optimization: {
    minimizer: [new TerserJSPlugin({
        cache: true, // 是否缓存
        parallel: true, // 是否并行打包
        sourceMap: true
    })],
  }
}
```

   CSS的多核压缩 optimize-css-assets-webpack-plugin
3. speed-measure-webpack-plugin 打包速度分析
4. progress-bar-webpack-plugin 打包进度展示
5. 对整个工程开启缓存 hard-source-webpack-plugin
6. webpack-dashboard 增强了 webpack 的输出，包含依赖的大小、进度和其他细节。
   webpack-bundle-analyzer 打包结果分析
   webpack --profile --json > stats.json
   14-1 http://alexkuz.github.io/webpack-chart/
   14-2 http://webpack.github.io/analyse/
7. inline-manifest-webpack-plugin 把runtime放到html里
   html-inline-css-webpack-plugin 把一些核心的CSS放到页面内部
   html-webpack-inline-source-plugin 内部资源引入
8. cache-loader loader的缓存 => 'babel-loader?cacheDirectory=true'
   exclude: /node_modules/, // 排除不处理的目录
   include: path.resolve(__dirname, 'src') // 精确指定要处理的目录
9. 开启全局的编译缓存 hard-source-webpack-plugin
10. 压缩图片 image-webpack-loader

11. HtmlWebpackPlugins压缩推荐选项

```
new HtmlWebpackPlugin({
  inlineSource: ".css$",
  template: path.join(__dirname, `src/${pageName}/index.html`),
  filename: `${pageName}.html`,
  chunks: ["vendors", pageName],
  inject: true,
  minify: {
    html5: true,
    collapseWhitespace: true,
    preserveLineBreaks: false,
    minifyCSS: true,
    minifyJS: true,
    removeComments: false,
  },
});
```

12. prepack-webpack-plugin 代码求值
13. @babel/plugin-syntax-dynamic-import 动态引入
14. Webpack5 不间断进程（continuous processes）和缓存

    对于大型复杂项目应用，在开发阶段，开发者一般习惯使用 Webpack --watch 选项或者 webpack-dev-server 启动一个不间断的进程（continuous processes）以达到最佳的构建速度和效率。
    Webpack --watch 选项和 webpack-dev-server 都会监听文件系统，进而在必要时，触发持续编译构建动作。
    原理其实就是轮询判断文件的最后编辑时间是否变化，某个文件发生了变化，并不会立刻告诉监听者，而是先缓存起来，等待aggregateTimeout（Webpack 的 --watch 选项内置的类似 batching 的能力）
    https://github.com/paulmillr/chokidar

> **工程优化与原理篇**

- 构建配置设计成一个库，比如：hjs-webpack、Neutrino、webpack-blocks
- 抽成一个工具进行管理，比如：create-react-app, kyt, nwb
- 更多的快速构建工具：lerna 、brunch、 rome 、snowpack （过往Browserify、Rollup.js、Gulp、Parcel、Microbundle）
- 更友好的提示错误
  friendly-errors-webpack-plugin
  webpack-build-notifier
  set-iterm2-badge && node-bash-title 标题和窗口内容修改

```
function() {
  this.hooks.done.tap('done', (stats) => {
    if (stats.compilation.errors && stats.compilation.errors.length
&& process.argv.indexOf('--watch') == -1)
    {
      console.log('build error');
      process.exit(1);
    }
  })
}
```

- splitchunks公用库的代码拆分 去除打包
  分离页面公用包 html-webpack-externals-plugin
- 使用动态 polyfill

  它会根据你的浏览器 UA 头，判断你是否支持某些特性，从而返回给你一个合适的 polyfill。

- 集成到CI 监控文件的大小 https://github.com/siddharthkp/bundlesize
- set-iterm2-badge && node-bash-title 标题和窗口内容修改