黑客派

vue element-ui 优化打包 bundle js 大小

作者：ZhijianZhang

原文链接：https://hacpai.com/article/1539364780380

来源网站：黑客派

许可协议：署名-相同方式共享 4.0 国际 (CC BY-SA 4.0)

<h2 id="背景">背景</h2>
最近开发的一个项目使用了 <code>vue</code>+ <code>element-ui</code> 的技术栈，然，还有其他的一些工具库，比如 <code>axios</code>。
说一下我的开发步骤，基础结构是通过 <code>vue-cli@2.x</code> 创建的，手动的加入了 <ode>axios</code> <code>vuex</code> , <code>vue-router</code> 是脚手架自带的。
<h2 id="code-split">code split</h2>
<h3 id="1--路由懒加载">1. 路由懒加载</h3>
使用 <code>vue-router</code> 的时候，如果按照默认配置，所有的路由都会被打包到一个code>bundle.js</code> 文件中去（bundle 文件名一般是 <code>app.js</code>）。
进入 <code>router/index.js</code> 文件中，只需要将所有类似 <code>import Home from '@/components/home';</code> 替换为 <code>const Home = () => import('@/componens/home')</code>
其余部分不需要变。就能以最简单的形式做到根据路由来划分 webpack 打包的模块。这个时候行 <code>npm run build </code> 是就能看到多了很多小的 js 文件， 并且 <code>app.js</cod> 文件的体积也减小了。
附上代码示例：
<pre><code class="highlight-chroma">import Vue from 'vue'
import Router from 'vue-router'
const AdminIndex = () => import('@/components/admin-index')
const Home = () => import('@/components/home')

Vue.use(Router)

export default new Router({
routes: [
{
path: '/',
name: 'admin-index',
component: AdminIndex
},
{
path: '/home',
name: 'home',
component: Home
}
]
})

</code></pre>

<h3 id="2--组件懒加载">2. 组件懒加载</h3>
跟路由懒加载的形式一样，也是使用 <code>() => import('xxx')</code> 的形式。
果对于一个容器组件中，<code>import</code> 很多个组件进来，使用组件懒加载，能够继续减首次加载的文件大小。示例代码：
<pre><code class="highlight-chroma"><template>
  <div>
    <HomeHeader />

```
    &lt;SearchContainer  /&gt;
    &lt;HomeFooter /&gt;
  &lt;/div&gt;
&lt;/template&gt;

<script>
import HomeHeader from "./home-header";
import HomeFooter from "./home-footer";
import SearchContainer from "./containers/search-container";
import LoadingComponent from "@/components/common/loading";

export default {
name: "home",
components: {
HomeHeader,
HomeFooter,
SearchContainer,
LoadingComponent
}
};
</script>
</code></pre>

<p>优化之前：<br> <img src="https://static.hacpai.com/images/img-loading.svg" alt="2018
012235553709png" data-src="http://media.zhijianzhang.cn//file/2018/10/9c533c8ed9c04091
395c14ea1123941_20181012235553709.png"><br> <img src="https://static.hacpai.com/ima
es/img-loading.svg" alt="20181013002145418png" data-src="http://media.zhijianzhang.cn//fi
e/2018/10/eadb9c08e5ab40088f4288d4ff5caaea_20181013002145418.png"></p>
<p>组件懒加载优化之后：</p>
<pre><code class="highlight-chroma">&lt;template&gt;
  &lt;div&gt;
    &lt;HomeHeader /&gt;
    &lt;SearchContainer  /&gt;
    &lt;HomeFooter /&gt;
  &lt;/div&gt;
&lt;/template&gt;

<script>
const HomeHeader = () => import("./home-header");
const HomeFooter = () => import("./home-footer");
const SearchContainer = () => import("./containers/search-container");
export default {
name: "home",
components: {
```

HomeHeader,

HomeFooter,

SearchContainer

}

};

</script>

</code></pre>

<img src="https://static.hacpai.com/images/img-loading.svg" alt="20181013002333147ng" data-src="http://media.zhijianzhang.cn//file/2018/10/4e247f6c3ee94e55a12336990a9926f_20181013002333147.png">
从文件的个数中，不知道有没有看出什么？
<code>app.js</code> 是属于 项目的公共部分的代码。而声一个 <code>const HomeHeader = () => import("./home-header"); </code> 类似的组件，会创建一个 <code>n.js</code> 文件，达到了继续拆分比较大的 js 包的目的。
所以其实只要你愿意，可以一部分组件使用 <code>const HomeHeader = () => import("./ome-header");</code> 另一部分组件使用 <code>import HomeHeader from "./home-header" </code> 。
不过总的来说，除非一个组件过于庞大了，在我开发过程中，才会想着用组件拆分的形式。每一小组件都这样拆分，最终得到很多很多个小的 js 文件，反而是因为网络请求的原因，拖慢加载速度的
<h3 id="3--webpack-bundle-analyzer">3. webpack-bundle-analyzer</h3>
在做一个项目的一开始，其实我都没有去考虑过性能优化、code split 的事情，只有当逻辑越来多，开发的时候明显感觉到页面加载速度慢了，network 里看到 <code>bundle.js</code> 体积巨了，才会想着去做优化的考虑。
emmm 所以到底应该怎么拆？拆哪些部分？这个需要 <code>webpack-bundle-analyzer</coe> 来帮忙，code split 也要有理有据。
<h4 id="安装和配置">安装和配置</h4>
如果你跟我一样，使用的是 <code>vue-cli@2.9.x</code> 的话，<code>webpack-bundle-nalyzer</code> 插件是已经安装了的，webpack 也配置好了的。
反正你就全局搜索一下 <code>webpack-bundle-analyzer</code> 就好了，看看 <code>pakage.json</code> 中有没有依赖，<code>webpack</code> 配置中有没有，一般只在 <code>wbpack.prod.conf.js</code> 中，因为开发环境下也不会去看的。
如果实在没有，那就手动安装和配置好了。
<pre><code class="highlight-chroma">npm i webpack-bundle-analyzer -D
</code></pre>
在 build/webpack.prod.config.js 中添加配置：
<pre><code class="highlight-chroma">if (config.build.bundleAnalyzerReport) {
  const BundleAnalyzerPlugin = require('webpack-bundle-analyzer').BundleAnalyzerPlugin
  webpackConfig.plugins.push(new BundleAnalyzerPlugin())
}
</code></pre>
在 <code>package.json</code> 的 <code>script</code> 中添加：
<pre><code class="highlight-chroma">“analyz”：“NODE_ENV=production npm_config_report=true npm run build”
</code></pre>
如果你是 window 用户的话，应该是：
<pre><code class="highlight-chroma">“analyz”：“set NODE_ENV=production npm_config_report=true npm run build”
</code></pre>
执行：<code>npm run analyz</code>

等项目 build 完了之后，就会自动打开一个页面了，
<img src="https://static.hacpai.com/images/img-loading.svg" alt="20181013004902726
ng" data-src="http://media.zhijianzhang.cn//file/2018/10/ef1c7fdab7cf4a6f87cbbb96d97026
8_20181013004902726.png">
<h3 id="4--element-ui-库的优化">4. element-ui 库的优化</h3>
重点终于来了（不是标题党。。。。）。从上面的图中我们可以看到，<code>vendor.xxx.js</c
de> 实在是有点大，webpack build 完了之后，也细心的为我们标注出了 <code>big</code>:<br
<img src="https://static.hacpai.com/images/img-loading.svg" alt="20181013005117948png"
data-src="http://media.zhijianzhang.cn//file/2018/10/5db8706427a240a7abb42ac89d04561d
20181013005117948.png">
解决办法是对于 <code>element-ui</code> 这个 ui 库从 <code>vendor.xx.js</code> 文
中剥离出来，最简单的办法就是使用 公共的 cdn 了。这里再做一层更彻底的剥离，将 <code>vue</
ode>, <code>vuex</code>, <code>vue-router</code>,<code>axios</code> 等依赖文件，
部使用 cdn。
 <img src="https://static.hacpai.com/images/img-loading.svg" alt="201810
3005418947png" data-src="http://media.zhijianzhang.cn//file/2018/10/cdadfa3c1e8943b096
780702e65ee7e_20181013005418947.png">
看好这些依赖的版本，直接去 百度搜索相关的 cdn 文件。下面我直接贴我修改之后的代码：</p

<h4 id="index-html">index.html</h4>
<pre><code class="highlight-chroma">...
<link rel="stylesheet" href="https://unpkg.com/element-ui/lib/theme-chalk/index.css">

...
<body>

<!-- cdn 加速，减小 vendor.js 体积 -->
<script src="https://cdnjs.cloudflare.com/ajax/libs/vue/2.5.17-beta.0/vue.min.js"></script>
<script src="https://cdnjs.cloudflare.com/ajax/libs/vuex/3.0.1/vuex.min.js"></script>
<script src="https://cdnjs.cloudflare.com/ajax/libs/vue-router/3.0.1/vue-router.min.js"></scri
t>
<script src="https://cdnjs.cloudflare.com/ajax/libs/axios/0.18.0/axios.min.js"></script>
<script src="https://unpkg.com/element-ui/lib/index.js"></script>
<!-- built files will be auto injected -->
</body>
</code></pre>

<h4 id="package-json">package.json</h4>
上面引入过 cdn 文件的依赖， 全都可以去掉了。
<h4 id="main-js">main.js</h4>
删除或者注释跟 <code>element</code> 相关的代码：
<pre><code class="highlight-chroma">import ElementUI from 'element-ui';
...
import 'element-ui/lib/theme-chalk/index.css';
...
Vue.use(ElementUI);
...
</code></pre>
<h4 id="webpack-base-conf-js">webpack.base.conf.js</h4>
在 webpack 配置中添加外部扩展：

<pre><code class="highlight-chroma">module.exports = {
...
    // 外部扩展，通过 cdn 引入，不会被webpack打包
    externals: {
      'vue': 'Vue',
      'vue-router': 'VueRouter',
      'vuex': 'Vuex',
      'axios': 'axios'
    }
  }
</code></pre>
这个时候已经好了，清除一下项目 <code>node_modules</code> 中的删除的不需要的依赖，<code>uninstall</code> 也行，直接删除整个 <code>node_modules</code> 文件夹，重新 <code>npm install</code> 也行。
处理完 <code>node_modules</code> 之后，<code>npm start</code> 再次看一下我们化之后的结果：

开发状态下的 <code>app.js</code> 明显已经变小了，build 之后的文件也是。
这里需要注意的一点是，依赖库使用 cdn 文件来加载话，网络请求的速度与 cdn 的速度有关，果不放心别人的 cdn，将上述的 cdn 文件内容下载到本地放在 <code>static</code> 目录下当做态文件即可。
建议使用 CDN 引入依赖的用户在链接地址上锁定版本，以免将来升级时受到非兼容性更新的影。
vue 项目的 code split，差不多就是这样了。感谢阅读，Happy Coding！