36 本章总结

更新时间：2019-08-19 14:18:48

> 老骥伏枥，志在千里； 烈士暮年，壮心不已。
>
> ——曹操

#本章总结

这一节的集结工作很快就完成了，这一章里我们一共做了两个事情，第一步是把之前开发的代码处理成一个可以在线上使用的最终压缩版本的文件，第二步是把这个最终文件又改成了一个符合 npm 规则的插件，并发布到了官方的 npm 源上。下来我们回忆一下这两个步骤中我们都做了什么，又会有什么问题。

文件方式的集成

这一节内容里，我们介绍里 npm 的用法，并使用 npm 安装了 postcss-cli、postcss-import、autoprefixer 和 cssnano 这几个插件，分别处理了代码的引入问题、兼容问题和压缩问题，最终把零散的源文件打包到一个压缩文件里。在这一节的最后，还是用 shell 脚本编写了一个自动打包的小工具，作为同学们的了解内容。这一节中，我们有下面几个问题：

1. 使用 npm 安装插件的时候，-D、-S、-g、–save、–save-dev这几个参数都有什么作用？

2. 如果不使用 npx，还有什么其他方式可以调用 postcss 命令？
3. 是否了解 browserslist 的配置方法？
4. 尝试用 webpack 重新完成这一步的内容。
5. 如果让你设计一个压缩 CSS 文件的工具，都可以做些什么？
6. 尝试在 shell 脚本里加入项目初始化的功能，可以通过执行脚本时使用不同的参数做不同的操作。

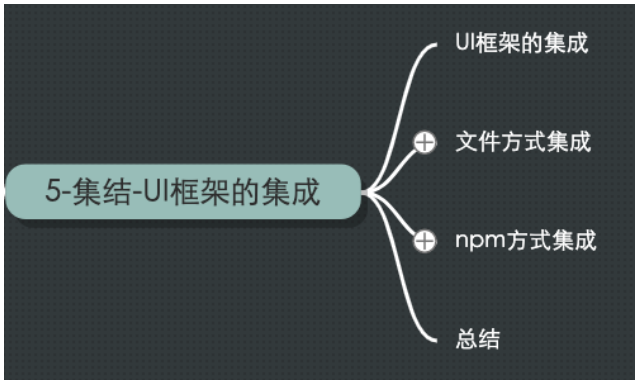
npm方式的集成

在以 npm 方式集成的这一步，我们按着 npm 的要求制作了一个 npm 包，包括初始化项目、修改对其他工具的依赖方式和添加说明文件，然后又在 npm 官方源上发布了我们自己的 npm 包。关于这一节的内容，可以有下面这几个问题：

1. 我们在项目中引入 npm 插件的时候，npm 都会去哪里找这个插件？顺序是怎样的？
2. 是不是了解 MarkDown 语法？平时用的多么？
3. 如何查看你现在用的 npm 源，怎么切换？
4. 在发布 npm 包的时候，为什么要把源切换到官方地址上？
5. 当我们发布了一个有问题的 npm 包，应该怎么做？

结语

上面的内容就是这一章的总结。到这里，这一章的“集结”工作就完成了，我们实现的这一套 UI 样式库也变成了正规军，可以在线上系统工作了。但我们打造的这套样式库到底好不好用，会不会有问题？到下一章，我们就会直接进入实战部分，用这个样式库实现几种常用的页面，把这些组件组合起来用，来检验下这个组件库的效果。这一章的内容比较少，结构如下：



我们这一章的内容就到这里了，下章开始进入实战部分。

}

35 使用 npm 形式集成
37 常用页面切图的设计与开发