# 一．NPM是干什么的？

[**npm 是干什么的？（非教程）**](https://zhuanlan.zhihu.com/p/24357770)

[方应杭](https://www.zhihu.com/people/zhihusucks)

[**方应杭**](https://www.zhihu.com/people/zhihusucks)

公众号 XDML，微信 hungervalley ，暗号：知乎

网上的 npm 教程主要都在讲怎么安装、配置和使用 npm，却不告诉新人「为什么要使用 npm」。今天我就来讲讲这个话题。

本文目标读者是「不太了解 npm 的新人」，大神您别看了，不然又说我啰嗦了 😂。

**社区**

程序员自古以来就有社区文化：

社区的意思是：拥有共同职业或兴趣的人们，自发组织在一起，通过分享信息和资源进行合作。虚拟社区的参与者经常会在线讨论相关话题，或访问某些网站。

前端程序员也有社区，世界上最大的前端社区应该就是 GitHub 了。前端通过 GitHub 来

1. 分享源代码（线上代码仓库）
2. 讨论问题（Issue 列表）
3. 收集学习资源和常去的网站（比如我收集的[优质中文前端博客](https://link.zhihu.com/?target=https%3A//github.com/FrankFang/best-chinese-front-end-blogs)）

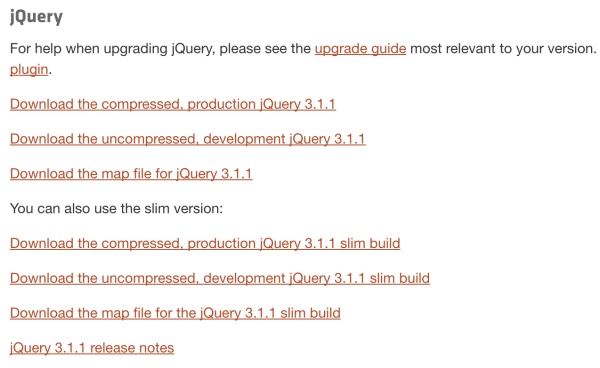
加入社区最大的好处之一是，你可以使用别人贡献的代码，你也可以贡献代码给别人用。

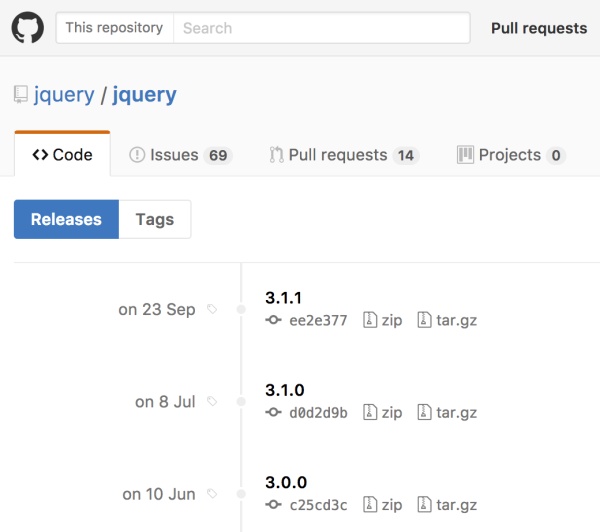
**共享代码**

前端是怎么共享代码的呢？

**在 GitHub 还没有兴起的年代，前端是通过网址来共享代码**

比如你想使用 jQuery，那么你点击 jQuery 网站上提供的链接就可以下载 jQuery，放到自己的网站上使用

**GItHub 兴起之后，社区中也有人使用 GitHub 的下载功能：**



**麻烦**

当一个网站依赖的代码越来越多，程序员发现这是一件很麻烦的事情：

1. 去 jQuery 官网下载 jQuery
2. 去 BootStrap 官网下载 BootStrap
3. 去 Underscore 官网下载 Underscore
4. ……

有些程序员就受不鸟了，一个[拥有三大美德](http://zhihu.com/question/48406009/answer/134182064)的程序员 [Isaac Z. Schlueter](https://link.zhihu.com/?target=https%3A//github.com/isaacs) （以下简称 Isaaz）给出一个解决方案：用一个工具把这些代码集中到一起来管理吧！

这个工具就是他用 JavaScript （运行在 Node.js 上）写的 npm，全称是 Node Package Manager

**具体步骤**

NPM 的思路大概是这样的：

1. 买个服务器作为代码仓库（registry），在里面放所有需要被共享的代码

2. 发邮件通知 jQuery、Bootstrap、Underscore 作者使用 npm publish 把代码提交到 registry 上，分别取名 jquery、bootstrap 和 underscore（注意大小写）

3. 社区里的其他人如果想使用这些代码，就把 jquery、bootstrap 和 underscore 写到 package.json 里，然后运行 npm install ，npm 就会帮他们下载代码

4. 下载完的代码出现在 node\_modules 目录里，可以随意使用了。

这些可以被使用的代码被叫做「包」（package），这就是 NPM 名字的由来：Node Package(包) Manager(管理器)。

**发展**

Isaaz 通知 jQuery 作者 John Resig，他会答应吗？这事儿不一定啊，对不对。

只有社区里的人都觉得 「npm 是个宝」的时候，John Resig 才会考虑使用 npm。

那么 npm 是怎么火的呢？

npm 的发展是跟 Node.js 的发展相辅相成的。

Node.js 是由一个在德国工作的美国程序员 Ryan Dahl 写的。他写了 Node.js，但是 Node.js 缺少一个包管理器，于是他和 npm 的作者一拍即合、抱团取暖，最终 Node.js 内置了 npm。

后来的事情大家都知道，Node.js 火了。

随着 Node.js 的火爆，大家开始用 npm 来共享 JS 代码了，于是 jQuery 作者也将 jQuery 发布到 npm 了。

所以现在，你可以使用 npm install jquery 来下载 jQuery 代码。

现在用 npm 来分享代码已经成了前端的标配。

**后续**

Node.js 目前由 Ryan Dahl 当时所在的公司 joyent 继续开发。Ryan Dahl 现在已经去研究 AI 和机器学习了，并且[他把 Node.js 的维护权交给了 Isaaz](https://link.zhihu.com/?target=https%3A//groups.google.com/forum/%23%21topic/nodejs/hfajgpvGTLY)。（我是不是也应该去研究 AI 和机器学习啊教练）

而 Isaaz 维护了一段时间后，辞职了，成立了一个公司专门维护 npm 的 registry，公司名叫做 [npm 股份有限公司](https://link.zhihu.com/?target=https%3A//www.npmjs.com/about%23about-npm-inc)……谁说开源不能赚钱的~

**社区的力量**

回顾前端的发展是你会发现，都是社区里的某个人，发布了一份代码，最终影响前端几年的走向。比如 jQuery，比如 Node.js，比如 npm。（其实其他语言也是这样的）

所以，社区的力量是巨大的。

# 二．什么是前端自动化

作者：鱼群  
链接：https://zhuanlan.zhihu.com/p/28483358  
来源：知乎  
著作权归作者所有。商业转载请联系作者获得授权，非商业转载请注明出处。

## 一、什么是前端自动化，前端自动化能干什么？

你是敲代码的没错，你是一名开发人员，你的工作是创造而不是重复，那就节省出这些时间来进入前端自动化的时代：

1.自动编译（将less，sass等自动编译）

2.自动合并（将页面引入的多个js文件，或者css文件，合并为同一个且压缩）

3.自动刷新（IDE保存，浏览器不用刷新，自动看到效果）

4.自动部署（自动将项目打包部署到指定目录）

5.自动同步（能够方便实现多个浏览器窗口，同步点击，输入，调试）

既然前端自动化这么好，那就从开始吧。

下面以环境搭建=>工具搭建=>项目架构搭建来把这些鬼简单介绍一下：

## 二、环境搭建 Node.js与npm

*1.Node.js环境搭建：*

Node.js是一个基于Chrome V8引擎的javascript的运行环境，其特点是单线程、事件驱动，非阻塞I/O模型，非常轻便高效，其包管理工具npm，是全球最大的开源库生态系统。

*2.Node.js是做什么的呢：*

本来浏览器在显示我们看到的网站的时候，会做很多事情，页面渲染，js渲染等等，然后node把其中js渲染的引擎拿出来，并且使用了谷歌的V8，然后在其外面又封装了一层api，让其拥有了文件读写，网络等操作，提供了一个服务端的运行环境，但却是运行的javascipt。所以说node.js给前端开发行业带来了一场工业革命。

*3.****NPM是什么****：*

NPM是随同NodeJS一起安装的包管理工具，能解决NodeJS代码部署上的很多问题，常见的使用场景有以下几种：

* 允许用户从NPM服务器下载别人编写的第三方包到本地使用。
* 允许用户从NPM服务器下载并安装别人编写的命令行程序到本地使用。
* 允许用户将自己编写的包或命令行程序上传到NPM服务器供别人使用。

*4.Node.js与npm的安装：*

打开[node.js官网](http://link.zhihu.com/?target=https%3A//nodejs.org/)去下载吧~安装好的node会自带npm。

打开命令提示符，执行下面两句，看一下版本号就可以了：

node -v // 查看当前node版本

npm -v // 查看当前npm版本

*5.npm怎么用呢*

使用npm安装插件：命令提示符执行

npm install <name> [-g] [--save-dev]

5.1 **<name>**：node插件名称。例：**npm install gulp-less --save-dev**

5.2、**-g**：全局安装。将会安装在C:\Users\Administrator\AppData\Roaming\npm，并且写入系统环境变量； 非全局安装：将会安装在当前定位目录； 全局安装可以通过命令行在任何地方调用它，本地安装将安装在定位目录的node\_modules文件夹下，通过require()调用；

5.3、**--save**：将保存配置信息至package.json（package.json是node.js的项目配置文件，在初始化文件 npm install 时会根据你配置文件中的条目进行安装）；

5.4、**-dev**：保存至package.json的devDependencies节点，不指定-dev将保存至dependencies节点；一般保存在dependencies的像这些express/ejs/body-parser等等。

5.5、使用npm卸载插件：

npm uninstall <name> [-g] [--save-dev]

5.6、使用npm更新插件：

npm update <name> [-g] [--save-dev]

npm update [--save-dev] // 更新全部插件

5.7、查看npm帮助：**npm help**

5.8、当前目录已安装插件：**npm lis**

*6.选装 cnpm*

npm很慢，那来cnpm吧，果然万能的淘宝，哈哈

npm install cnpm -g --registry=https://registry.npm.taobao.org

安装好后看下版本号就可以直接用啦，用法和npm一致，就是npm换成cnpm就好了~

## 三、工具搭建 Gulp、Grunt、webpack、browserify

node和npm明白是干什么的了，搭建环境吗，那可以直接开敲代码了吗？no，no，no，你还需要搭建工具：

*1.Gulp / Grunt* ***简化前端流程***

Gulp / Grunt 是工具链、构建工具，它们能够优化前端工作流程。比如自动刷新页面、combo、压缩css、js、编译less等等。使用Gulp/Grunt，然后配置你需要的插件，就可以替代手工实现自动化工作。

*2.browserify / webpack* ***JS模块化方案***

browserify / webpack : 是文件打包工具，可以把项目的各种js文、css文件等打包合并成一个或多个文件，主要用于模块化方案，预编译模块的方案。因为它是预编译的，所以不需要在浏览器中加载解释器。你可在本地直接写JS，不管是 AMD / CMD / ES6 风格的模块化，它都能认识，并且编译成浏览器认识的JS。

目前我只接触过gulp+webpack的项目，所以就这两个工具下载安装一下吧：

*3.gulp与webpack的安装*

安装gulp

npm install gulp -g // 全局安装gulp

gulp -v // 出现版本号即为正确安装

安装webpack

npm install webpack -g // 全局安装webpack

webpack -v // 出现版本号即为正确安装

全局安装过后，仍需在项目根目录进行本地安装，全局安装gulp是为了执行gulp任务，本地安装gulp则是为了调用gulp插件的功能。

## 四、项目架构搭建

### 1.package.json

还记得上面说过的package.json嘛？package.json是基于node.js项目必不可少的配置文件，它是存放在项目根目录的普通json文件。

可以通过 npm init 来新建一个package.json文件

由于本文不是一篇实用性项目构建的流程文章，就不具体展开package.json的写入及配置流程了，我的第一篇blog里面有基于vue-cli的构建，可以参考。这篇文章就是想通过大体的构建过程来弄清楚各部分的功能。

### 2.直接使用脚手架

脚手架是什么：stackoverflow上的一个回答是这样的

“脚手架”是一种元编程的方法，用于构建基于数据库的应用。许多MVC框架都有运用这种思想。程序员编写一份specification（规格说明书），来描述怎样去使用数据库；而由（脚手架的）编译器来根据这份specification生成相应的代码，进行增、删、改、查数据库的操作。我们把这种模式称为"脚手架"，在脚手架上面去更高效的建造出强大的应用！

我的理解呢，脚手架就是给你的项目先打个架子，写个框子，然后你按着写入的方式去一点点填充满你的程序。

安装vue脚手架

npm install vue-cli

根据webpack创建项目

vue init webpack 项目名字<项目名不能用中文>

react的脚手架则是 [create-react-app](http://link.zhihu.com/?target=https%3A//github.com/facebookincubator/create-react-app/) 等，有很多模版可以自己选择，这样就避免了自己构建那么多复杂的流程。

**五、总结一下**

搭建环境：node.js 等

搭建工具： Gulp、Grunt、webpack、browserify 等

框架：React、vue、Angular等

库：jQuery、Prototype等

插件：太多了……

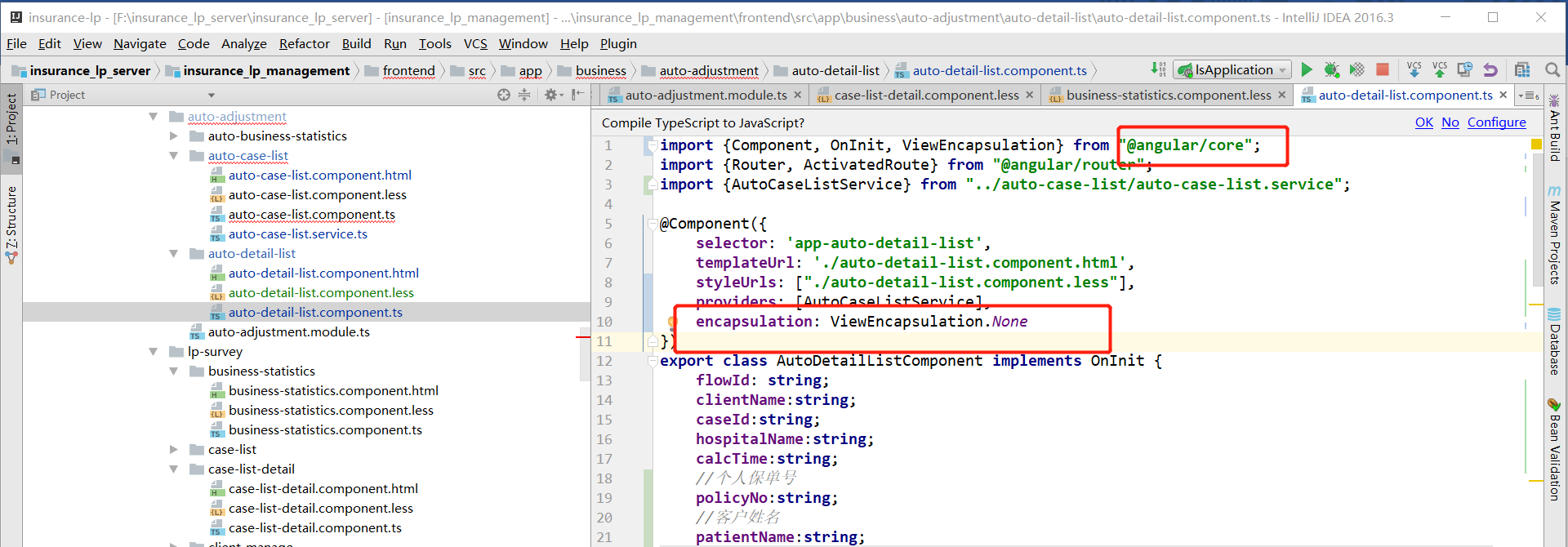
写到这也差不多了，要不然越写越偏，马上就开始建项目了这是……不知道看完这篇文章你明白node，npm这些都是做什么的了吗，其实…… 会用就行了（捂脸）。前端自动化算是前段工程化的开端，然而还有模块化、组件化、规范化等等一系列，路漫漫而修远，脚踏实地慢慢走吧~可能过一阵就会自己练习一个react的项目，到时候会记录下来，然后做一个从头到尾的项目流程。

# 三．Angular2

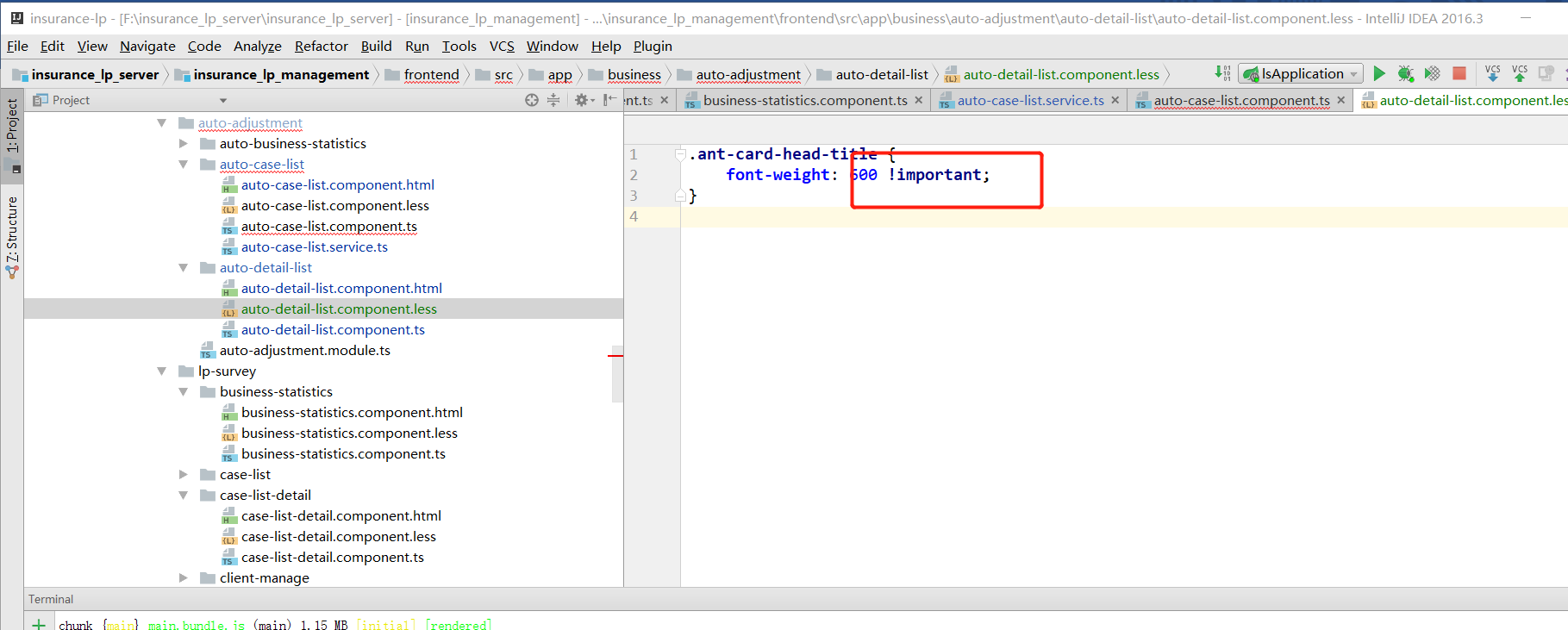
## 一．基本概念

## 二．Ng-zero

### 1.修改自定义样式需要引入属性encapsulation: ViewEncapsulation.*None*



然后less中



看节目面效果：



加粗了：

