



链滴

开源经验谈

作者: [88250](#)

原文链接: <https://ld246.com/article/1501603146956>

来源网站: [链滴](#)

许可协议: [署名-相同方式共享 4.0 国际 \(CC BY-SA 4.0\)](#)

<h2 id="Bootstrap">Bootstrap</h2>

<p>古话说得好：“万事开头难”。</p>

<p>开头在本质上是解决“Bootstrapping”，人们一直使用这个比喻来描述解决启动问题的方法，一个人试图用自己的鞋带将自己拉起来。比如按下电脑开机按钮后初始化系统被广泛称作自举引导（称 booting）。</p>

<p>想做好一个开源项目，就先得规划一个好的启动方式。</p>

<h2 id="GitHub">GitHub</h2>

<p>10 多年前做开源项目托管的话基本只有一个选择，那就是 SourceForge。

期间也有过一些其他大厂的项目托管服务，比如 Google Code，Sun/Oracle 的 Project Kenai，现都停服了。目前 GitHub 已然是全球最大的开源项目托管服务商了，促成这现状的最大原因有这几个方面吧：</p>

git 版本控制系统的流行，并做了一些更简单的工作流，比如 Pull Request

更好、更现代化的用户体验，比如 issue comments 的实时推送

API 开放平台，构建第三方应用的生态系统

持续不断的改进，GitHub 隔三差五会有一些细节改进，并且桌面客户端工具也在不断演进

<p>目前 GitHub 是开源项目托管的不二之选。国内也有类 GitHub 的项目托管服务，比如码云和 Coding，希望它们越做越好 :thumbs up:</p>

<h2 id="Repo">Repo</h2>

<p>建库是第一步，首先需要确定的是开个人项目还是组织项目，这两者有些许不同：</p>

个人项目：项目如果受欢迎一般来说人们都会浏览下作者主页并关注，有利于作者本人涨粉

组织项目：如果一开始就规划了相关系列项目，建议选择组织类型

<p>第二步就是需要一个好的名字：</p>

和需求/技术相关或者有些特定含义

库名命名优先是全小写，如果需要分隔单词则使用 <code>-</code>，最后遵循开发语言、框的约定

如果项目是一个系列的，需要考虑好前缀，或者按前面说的走组织项目

<h2 id="Init-Commit">Init Commit</h2>

<p>初始提交尽量是核心功能可用的，这样能给别人一个好印象。并且必须写好 README，内容至少覆盖如下几个方面：</p>

项目介绍：该项目是什么，主要用于解决什么问题，这个部分主要阐述项目的动机

功能特性：有哪些主要功能，独特的特性有哪些

安装方式：如何安装，最好也介绍一下开发环境搭建

技术依赖：该项目主要依赖哪些项目，这样能让别人大体上知道复杂度和上手难度

开源协议：一定要让用户知道，非常重要

<p>除了 README，GitHub 还建议加入编码规范、贡献指南说明，这个可以后期慢慢加入。</p>

<p>文本的介绍是一方面，另一方面是多媒体格式，如果能有使用视频是最好的，但至少项目的 logo 是要有的，一开始稍微丑一点没关系 :blush:，可以鼓励其他人参与贡献。</p>

<h2 id="Document">Document</h2>

<p>项目托管服务商一般都提供了 wiki 服务，主要用于项目相关文档。直接使用 wiki 的最大好处是 以让其他人参与进来维护文档，比如有错别字时其他人可以很方便地帮忙修改。</p>

但直接使用 wiki 服务也有个致命的缺点，就是当项目要迁移托管商时就很麻烦，不同服务商支的文档语法不尽相同，排版也比较麻烦。推荐的做法是**项目的文档自己搭建一个站点来管理**，虽然运维麻烦一些，但这样不存在迁移问题，并且自由度也更大一些。

建议通过论坛系统来维护文档，这样方便用户进行提问，也促进了项目社区的发展。

Issues

Issue 最好先建立模板，让用户提问的时候有章可循。不过即使有模板，issue 描述不清晰还是常见的，用户的系统环境千奇百怪，遇到奇葩的系统不工作其实也正常。

关于 issue 你未来可能会面临如下情况：

-

- [练习使用 GitHub 的](https://ld246.com/forward?goto=https%3A%2F%2Fgithub.com%2Fb3log%2Fwde%2Fissues%2F279)
- [由于其他软件不会用导致的](https://ld246.com/forward?goto=https%3A%2F%2Fgithub.com%2Fb3log%2Fwde%2Fissues%2F295)
- [宣泄个人情感的](https://ld246.com/forward?goto=https%3A%2F%2Fgithub.com%2Fb3log%2Fso%2Fissues%2F12112)

良好的 issue 交互情况：

-

- [提 issue 的人描述清晰](https://ld246.com/forward?goto=https%3A%2F%2Fgithub.com%2Fb3log%2Fwde%2Fissues%2F267)
- [提 issue 的人跟踪 bug 有始有终](https://ld246.com/forward?goto=https%3A%2F%2Fgithub.com%2Fb3log%2Fwde%2Fissues%2F270)
- [提 issue 的人提完顺带 fix](https://ld246.com/forward?goto=https%3A%2F%2Fgithub.com%2Fb3log%2Fwde%2Fissues%2F300)

总之通过 issue 来追踪管理变更是很好的做法，发布版本时还可以根据 issues 生成 [changelogs](https://ld246.com/forward?goto=https%3A%2F%2Fhacpai.com%2FCHANGE_LOGS.html)。

不过对于 issue 也有我觉得比较难处理的情况：

-

- [提的 issue 构想太大，作者 hold 不](https://ld246.com/forward?goto=https%3A%2F%2Fgithub.com%2Fb3log%2Fwde%2Fissues%2F292)
- [和作者本意相矛盾](https://ld246.com/forward?goto=https%3A%2F%2Fgithub.com%2Fb3log%2Fso%2Fissues%2F12124)

总之，做开源很重要一点就是和参与者、用户保持交流，等项目用的人多了以后责任也会变大，以不能太随意，尽量做到变更可追溯。

回归本质

总之，开始一个开源项目时最重要的一点就是要问自己：*为什么要做这个开源项目？*

这个问题的答案将解释该项目的动机并且为项目设定了目标。

开源项目成千上万，并且同质化严重，但总有一些开源项目能够脱颖而出，因为这些项目抓住了质：

-

- 开发者自己是用户：开发者自己不用是发现不了问题的，特别是后期需求
- 做真正有用的事情：不一定要受众很广，但要有刚需，能够解决问题
- 长期投入精力情感：你投入的情感使它与众不同，保持细水长流

以上都是我的经验之谈，如果你想更全面了解开源，请访问 GitHub 出品的 [Open Source Guide](https://ld46.com/forward?goto=https%3A%2F%2Fopensource.guide)。

<h2 id="本文作者">本文作者</h2>

<p>https://github.com/88250 , 欢迎关注。 </p>