数字商品生产协作工具

GitHub 原理与实战

主讲人：庄表伟

时间：2024.11

目 录

01 引言

02 GitHub的诞生历史

03 GitHub的核心功能

04 参与开源的建议

05 GitHub与职业发展

06 GitHub实战

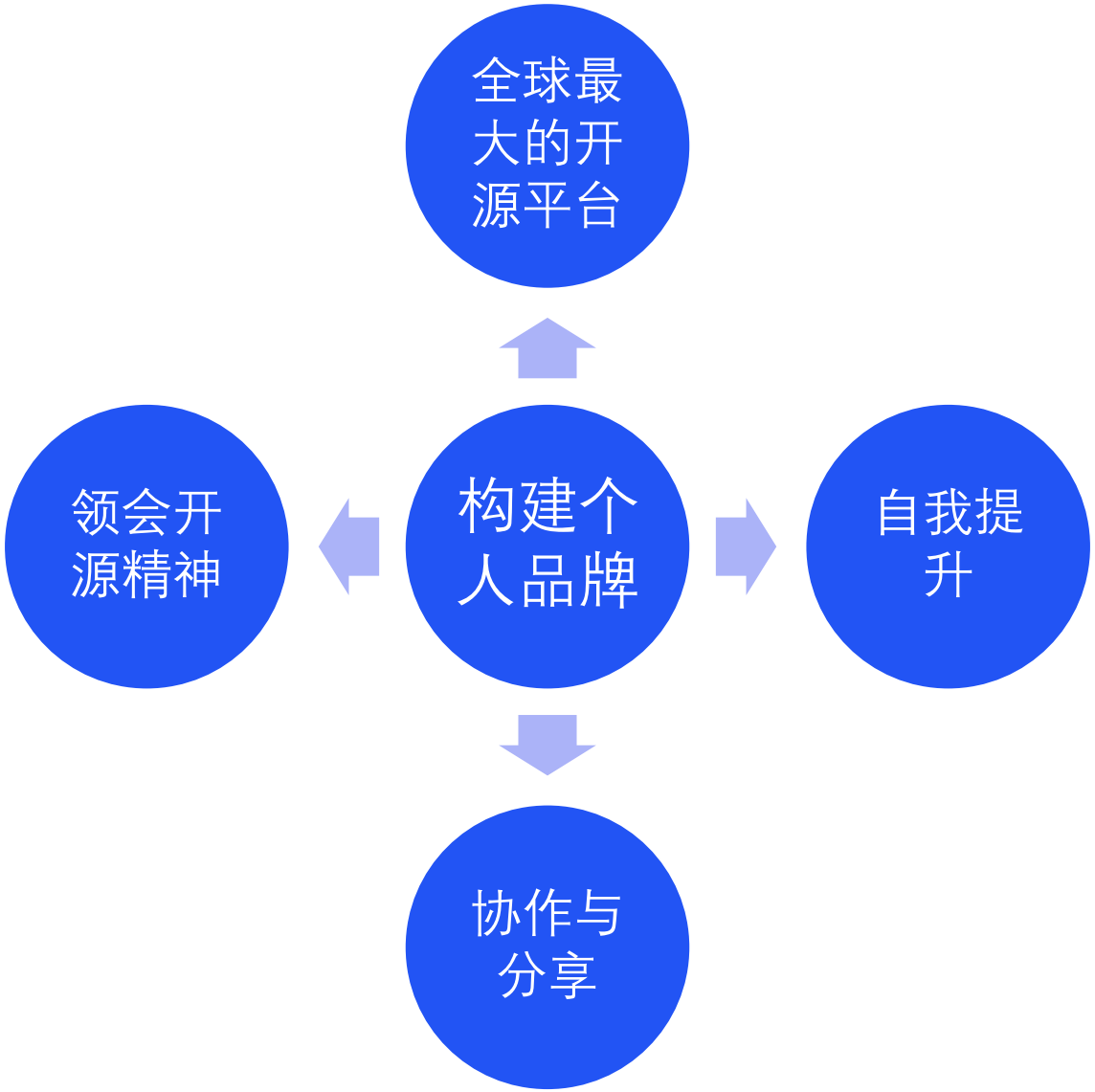
01

引言

GitHub在开源和技术社区中的重要性

- 一组数据
  - 根据2024 年 GitHub Octoverse，全球开发者为 GitHub 上的超过 5.18 亿个项目做出了超过 52 亿次贡献。其中针对开源项目的贡献，约10亿次。
  - 根据《2023中国开源年度报告》，GitHub 全域活跃仓库数量达到了 8,792 万，比上一年增长了 4.06%

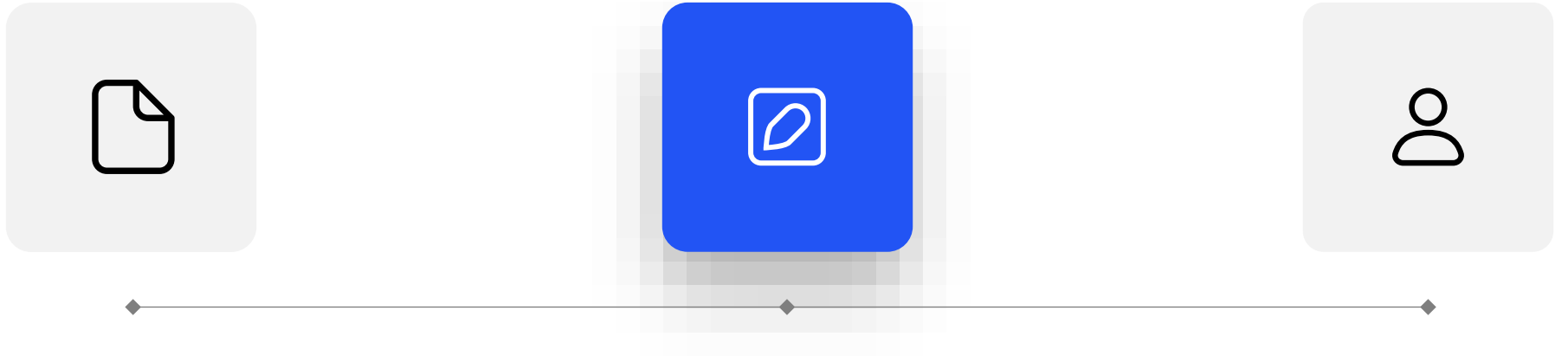
为什么要学习和使用GitHub?



02

GitHub的诞生历史

Git的背景：Linus Torvalds与Git的创造



版本控制的需求

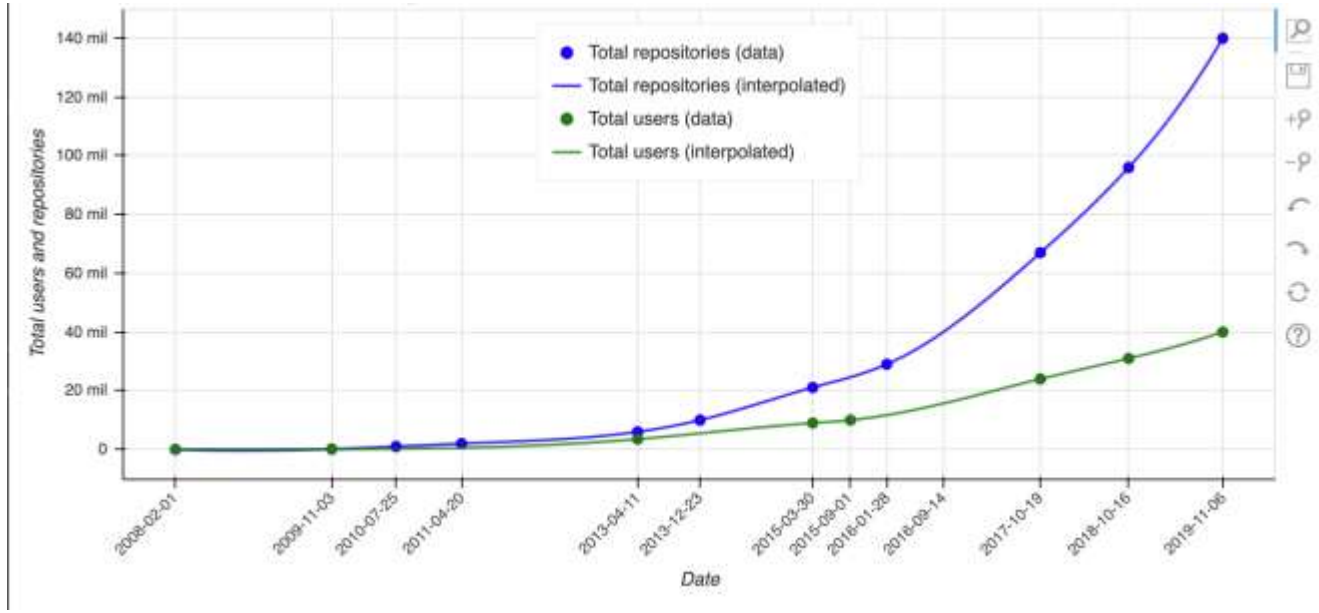
Linux内核开发的挑战

Git的诞生：2005年

GitHub的诞生：Tom Preston-Werner、Chris Wanstrath和PJ Hyett的创业故事

- **GitHub的创立：** 虽然Git作为一个开源项目已经成功，但它的使用主要局限于命令行工具，缺乏一个直观的界面和易于协作的功能。GitHub的创始人认为，Git可以更好地为开发者提供服务，尤其是在协作开发和开源项目的支持上。于是，**Tom Preston-Werner**、**Chris Wanstrath**和**PJ Hyett**在2008年共同创办了GitHub。
- **GitHub的初衷和使命：**GitHub的创始人从一开始就明确表示，他们的目标是为开发者提供一个可以自由共享、协作和发展项目的地方。他们相信开源和共享代码的精神能够推动技术进步，并帮助全球的开发者更高效地合作。GitHub不仅仅是一个代码托管平台，它更是一个连接开发者的社区，促进技术交流和创新。
- **GitHub的早期功能：**
  - 简化版本控制，用户友好的界面：开发者可以在浏览器中查看代码，提交代码，查看分支，进行代码审查等操作，而不需要依赖命令行工具。
  - 支持开源项目，开源项目的“社交化”：引入了类似社交媒体的功能，如“Star”（标星）、“Fork”（分叉）、“Follow”（关注）等。这些功能使得GitHub不仅是一个代码托管平台，也是一个社区平台
  - 社交功能，社交化的开发平台：通过Fork和Pull Request等功能与他人协作，这种社交化的开源模式，使得更多的开发者能够快速加入开源项目，贡献代码或反馈问题。

GitHub的发展历程



小公司

- 2008年成立

行业领军者

- 2012年，注册用户数突破100万

微软收购GitHub

- 2018年，微软以75亿美元收购

全球最大开源基地

- 常年稳坐全球第一的宝座

GitHub的功能发展

01

GitHub Pages：静态页面托管服务，2008年推出

02

GitHub Marketplace：应用扩展市场，2017年推出

03

GitHub Actions：CI/CD服务，2018年推出

04

GitHub Copilot：基于AI的开发辅助工具，2021年推出

其他代码托管平台的发展历程

- SourceForge创立于1999年，是最早的开源项目托管平台之一。到2006年，SourceForge已经成为全球最大的开源项目托管平台之一，托管了超过10万个项目。如今，SourceForge仍在运行，但其市场份额和影响力已大不如前。
- Google Code于2006年推出，是Google为开源社区提供的免费代码托管服务。它支持SVN、Mercurial（Hg）和Git三种版本控制系统，并提供简单的项目管理功能。在2015年宣布关闭Google Code，鼓励用户迁移到GitHub或Bitbucket。
- Gitee（原名“码云”）于2013年由开源中国（OSChina）推出，是中国本土化的代码托管平台。2020年，Gitee被选为中国开源代码托管平台的试点单位，旨在减少对国外平台的依赖。截至2024年，Gitee已经成为中国最大的代码托管平台之。
- GitLab由Dmitriy Zaporozhets和Valery Sizov于2011年创立，以完全开源、可自托管的特性在早期吸引了大量用户。2013年，GitLab Inc.成立，公司总部设在旧金山，2021年10月，GitLab在纳斯达克上市，成为一家公开上市公司，最新市值 106.5 亿美元。
- 其他代码托管平台：Launchpad（2004），Bitbucket（2008），Gitea（2016），GitCode（2020），GitLink（2022），AtomGit（2023）

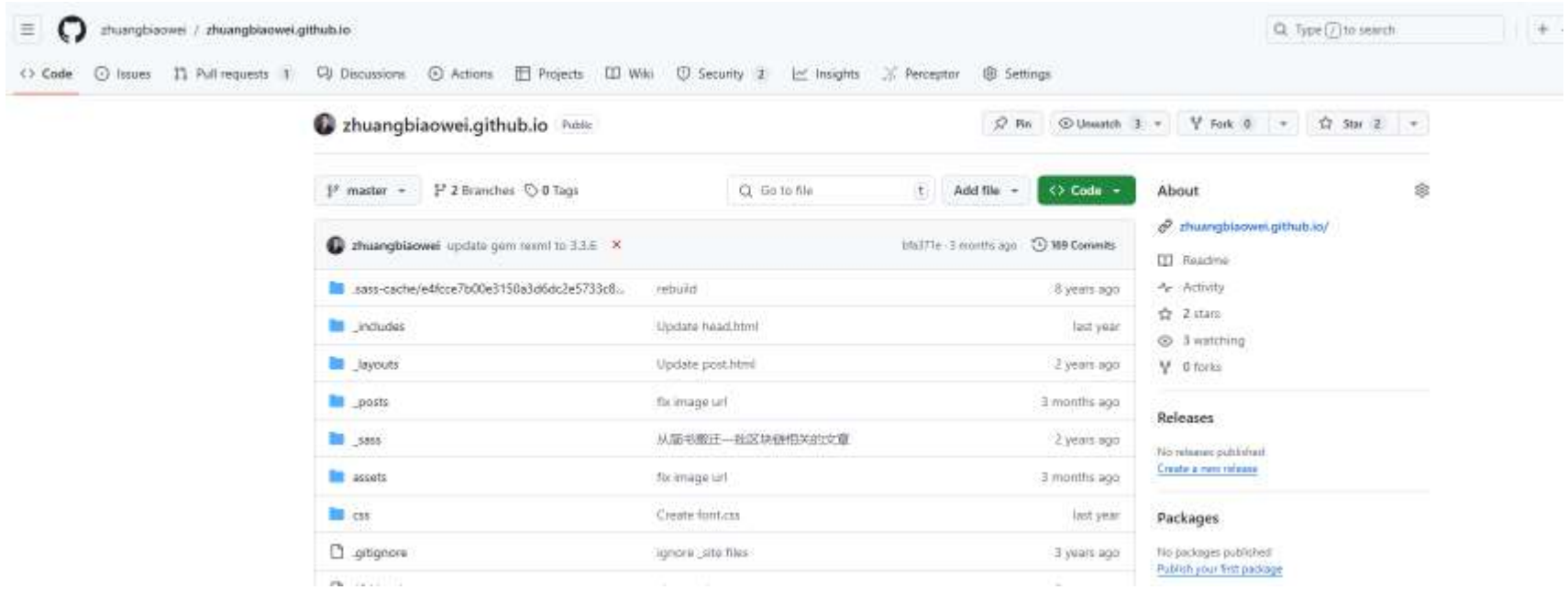
03

GitHub的核心功能

版本控制与Git的结合

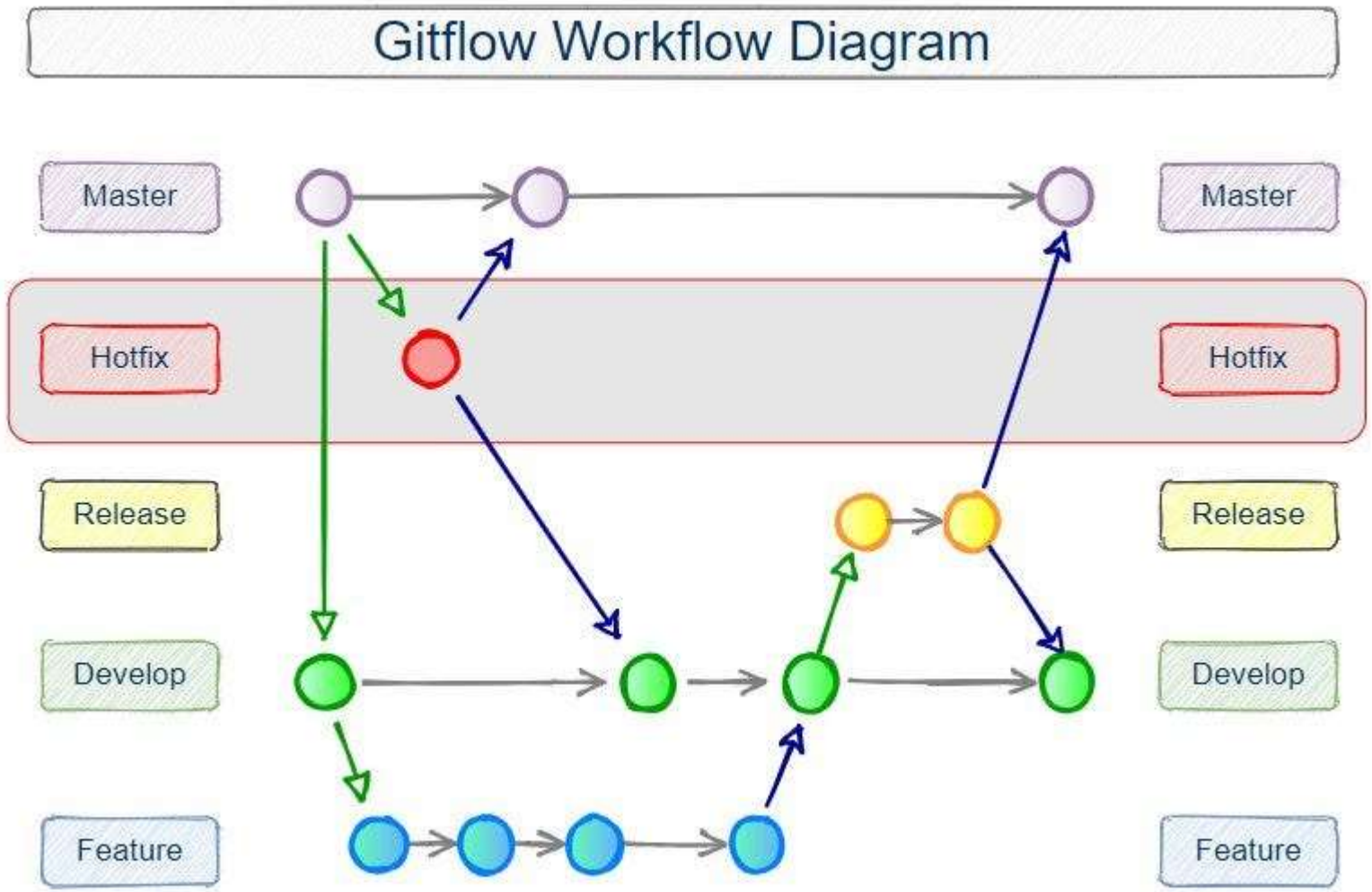
- 在GitHub上，版本控制与Git的结合使得开发者可以在平台上通过图形化界面或命令行轻松执行Git的操作，如提交（Commit）、推送（Push）、拉取（Pull）、合并（Merge）等，同时借助GitHub的社交和协作功能，实现跨团队、跨地域的协同工作。

https://github.com/zhuangbiaowei/zhuangbiaowei.github.io

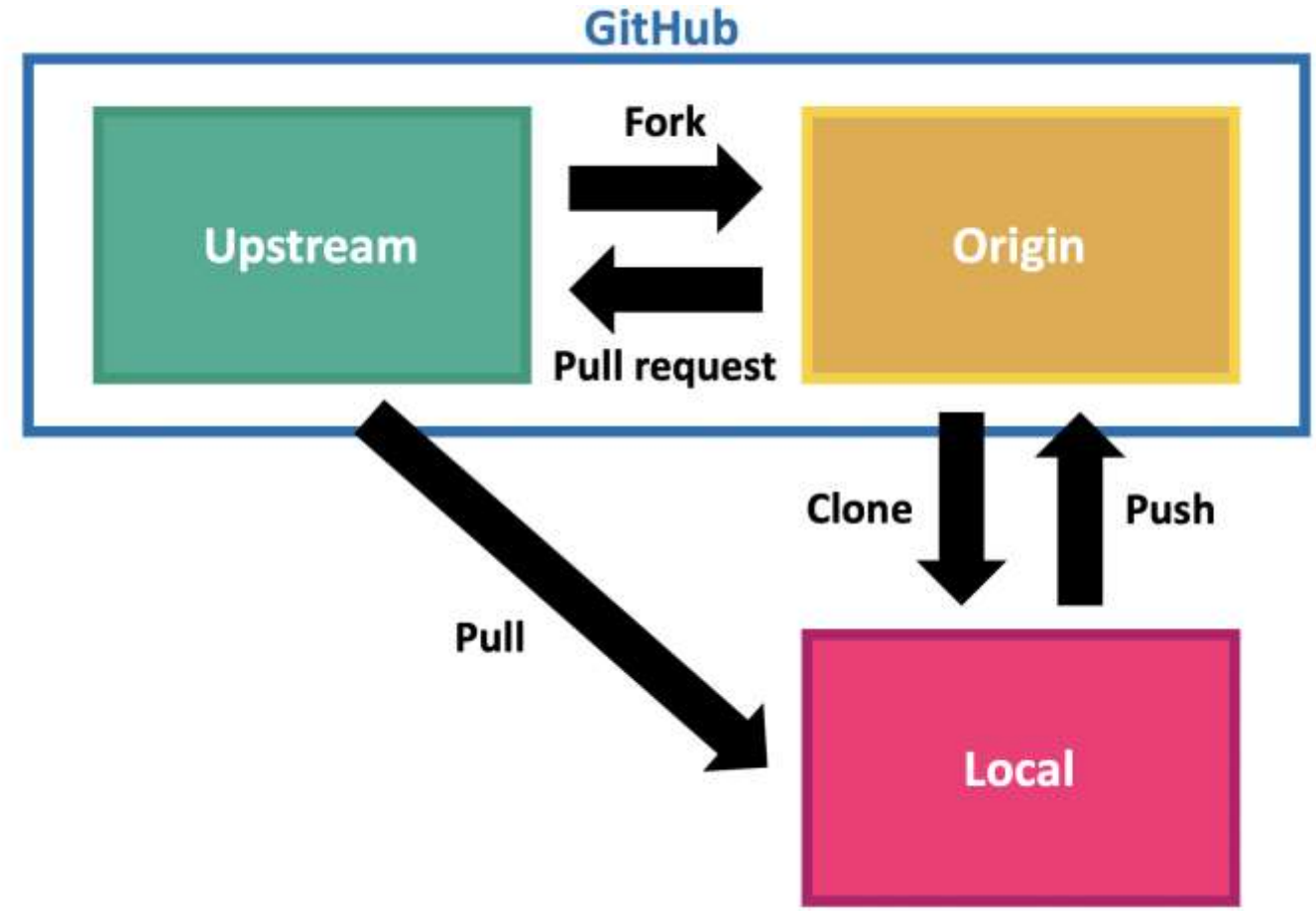


仓库（Repository）与分支（Branch）

- 仓库（Repository）
  - 公开仓库
  - 私有仓库
- 分支（Branch）
  - 管理和创建分支
  - 合并分支（Merge）



分叉（Fork）与拉取请求（Pull Request）



拉取请求（PR）是GitHub中的协作功能，它允许开发者将自己在某个分支上的更改提议合并到主分支或其他目标分支。PR不仅仅是代码的合并请求，它还包括代码审查、讨论和审计等环节，是多人协作开发的核心工具。

版本发布（Release）与标签（Tag）

版本发布（Release）： 在软件开发中，版本发布是项目的重要环节，它标志着软件的一次正式发布。GitHub提供了**Release**功能，允许开发者为项目创建版本，并在每个版本发布时附上相关的版本号、说明和文件。Release通常会关联到GitHub中的标签（Tag），以便开发者能够轻松获取不同版本的代码。

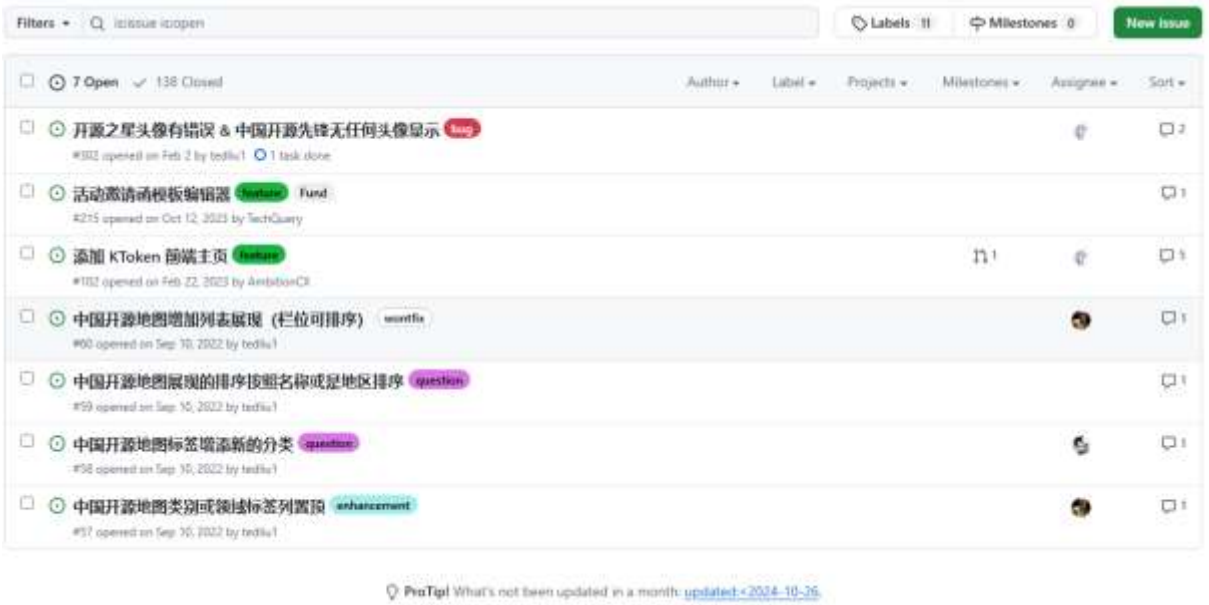
- **版本说明**：在创建Release时，开发者可以为该版本提供详细的发布说明，列出版本的功能、新增特性、修复的bug等信息。
- **二进制文件**：除了源码，GitHub Release还可以附加可下载的二进制文件，方便用户直接下载已编译好的程序。

标签（Tag）： 标签是Git的一种功能，它用于标记代码的特定版本。通常，标签用于标记发布版本或重要的里程碑。通过标签，开发者可以方便地查找和回滚到某个特定版本。

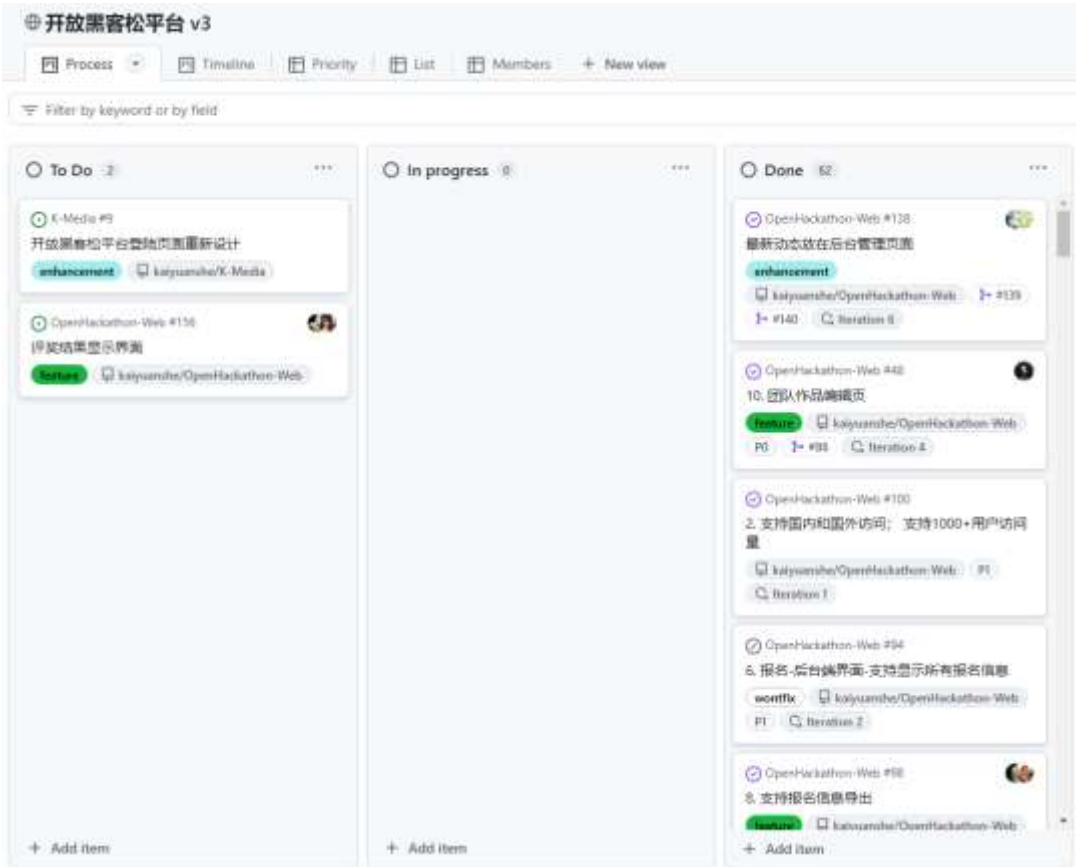
- **轻标签与附注标签**：Git支持两种类型的标签：轻标签（Lightweight Tag）和附注标签（Annotated Tag）。附注标签除了标记版本，还能附带信息（如版本号、作者、日期等）。
- **标签与Release的关系**：GitHub上的Release通常与标签结合使用。每次创建Release时，都会为其创建一个对应的标签，使得开发者可以通过标签快速定位到对应的版本。

Issues和Project Boards的管理

https://github.com/kaiyuanshe/kaiyuanshe.github.io/issues



https://github.com/orgs/kaiyuanshe/projects/9/views/10



04

参与开源的建议

如何找到适合的开源项目

01

根据兴趣和技能选择项目。

02

GitHub的搜索功能。

03

浏览热门项目和趋势。

04

查阅开源社区。

05

订阅新闻源、公众号。

06

查看项目的维护状况。

初学者如何开始：小的贡献和学习资源

01

从文档贡献开始

02

解决小的Bug和问题

03

查阅贡献指南

04

学习资源

官方指南：https://opensource.guide/zh-hans/how-to-contribute/

为开源项目做贡献的10个步骤：https://www.51cto.com/article/625473.html

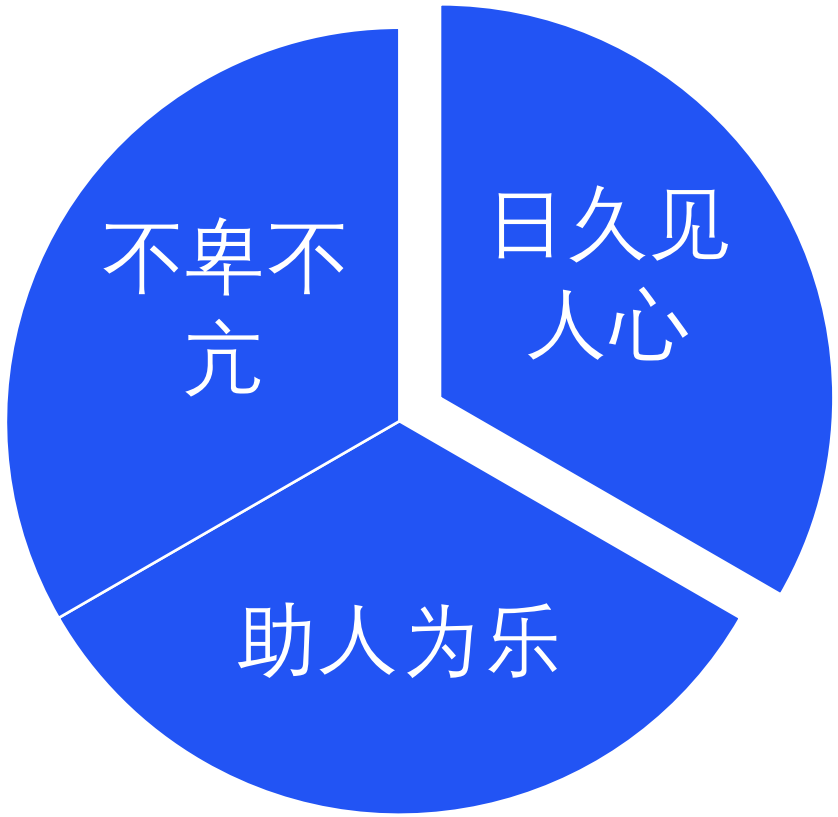
提交Pull Request的最佳实践

1. **克隆项目并创建新分支**:
   - 在Fork了项目并将其克隆到本地后，建议为你的工作创建一个新的分支，而不是直接在主分支上进行修改。
   - 这样可以确保你的代码更改与项目的主分支保持分离，便于管理和合并。
2. **频繁提交并保持提交信息清晰**:
   - 提交时要保持信息简洁明了。一个好的提交信息应描述清楚你对代码做了什么修改，为什么做这些修改。
   - 每次做一个功能或修复时，尽量提交一次，以便其他人可以清楚地看到你的修改。
3. **确保代码符合项目的编码规范**:
   - 在提交PR之前，检查代码是否符合项目的编码规范。许多开源项目都会提供一些代码规范（如PEP-8规范对于Python项目）。你可以使用代码格式化工具（如prettier、black等）来自动检查和格式化代码。
4. **测试你的代码**:
   - 提交PR前，确保你的代码经过充分的测试。对于修改的功能，最好编写单元测试（Unit Test），确保你的更改不会破坏其他功能。
   - 许多开源项目会有自动化测试（如CI/CD），提交时这些测试会自动运行。如果有失败的测试，PR会被标记为需要修复。
5. **编写清晰的PR描述**:
   - 创建Pull Request时，写清楚你做了哪些更改，解决了什么问题，或者添加了哪些功能。清晰的描述有助于审查者快速理解你的工作。
6. **保持PR简洁**:
   - 尽量避免在一个PR中做过多的修改。小而精的PR更容易审查和合并。如果要添加多个功能或修复多个Bug，最好分开提交不同的PR。

遇到问题时如何与项目维护者沟通

1. **通过Issue提问：**
   - 如果在使用开源项目时遇到问题，可以先检查是否已有类似的Issue。如果没有，可以自己创建一个新的Issue，简要描述问题，并附上相关的错误信息、代码或截图，方便维护者理解问题。
2. **与维护者的沟通：**
   - 如果你需要更多帮助或反馈，可以在Issue中与项目的维护者或其他贡献者讨论。保持礼貌，简明扼要地表达你的问题，并提供足够的信息。
   - 如果你向维护者提交了PR，且没有收到回应，可以通过PR评论或重新提交PR来提醒维护者。
3. **参与项目讨论：**
   - 一些项目会使用Discussions功能来进行更深入的讨论。在Discussions中，你可以提出问题、讨论功能需求或与社区其他成员分享经验。
4. **尊重维护者的时间：**
   - 维护开源项目的时间是有限的，因此如果你提交了Issue或PR，给维护者足够的时间进行反馈。尊重开源社区的贡献者，理解他们的优先级和工作量。

持续参与与贡献的技巧：从贡献者到管理者



05

GitHub与职业发展

GitHub作为职业技能的一部分

技术能力的展示。

GitHub上的技能证明。

GitHub作为持续学习的工具。

GitHub作为技术认知的展示平台。

如何通过GitHub提升职业形象与建立专业网络

通过GitHub与其他开发者建立联系。

优化GitHub个人资料。

使用GitHub作为简历的补充。

提升知名度与行业影响力。

参与招聘和技术交流活动。

创建和管理开源项目。

06

GitHub实战

创建一个GitHub账户

https://docs.github.com/zh/get-started/start-your-journey/creating-an-account-on-github

Hello World

https://docs.github.com/zh/get-started/start-your-journey/hello-world

克隆一个仓库

https://docs.github.com/zh/repositories/creating-and-managing-repositories/cloning-a-repository

Fork一个仓库并发起一个Pull Request

https://docs.github.com/zh/pull-requests/collaborating-with-pull-requests/proposing-changes-to-your-work-with-pull-requests/creating-a-pull-request-from-a-fork

https://github.com/zhuangbiaowei/learn-with-open-source

更多文档

https://docs.github.com/zh

YOUR LOGO

谢谢大家

主讲人：庄表伟

时间：2024.11