

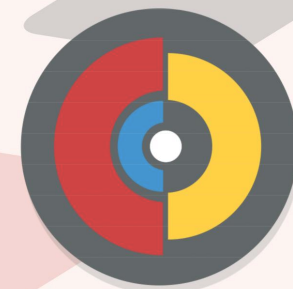
安同校园行 河海大学站 | 欢迎

```
./mach build; git push; moz-phab
```

如何维护 Firefox 的架构适配

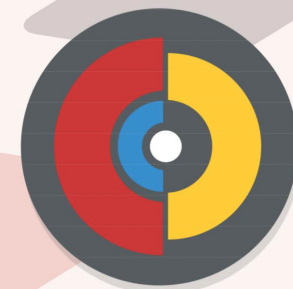
王雪瑞 (xen0n)

AOSC 贡献者, 《咱龙了吗?》主创



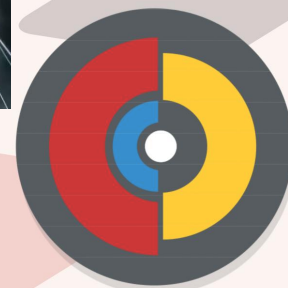
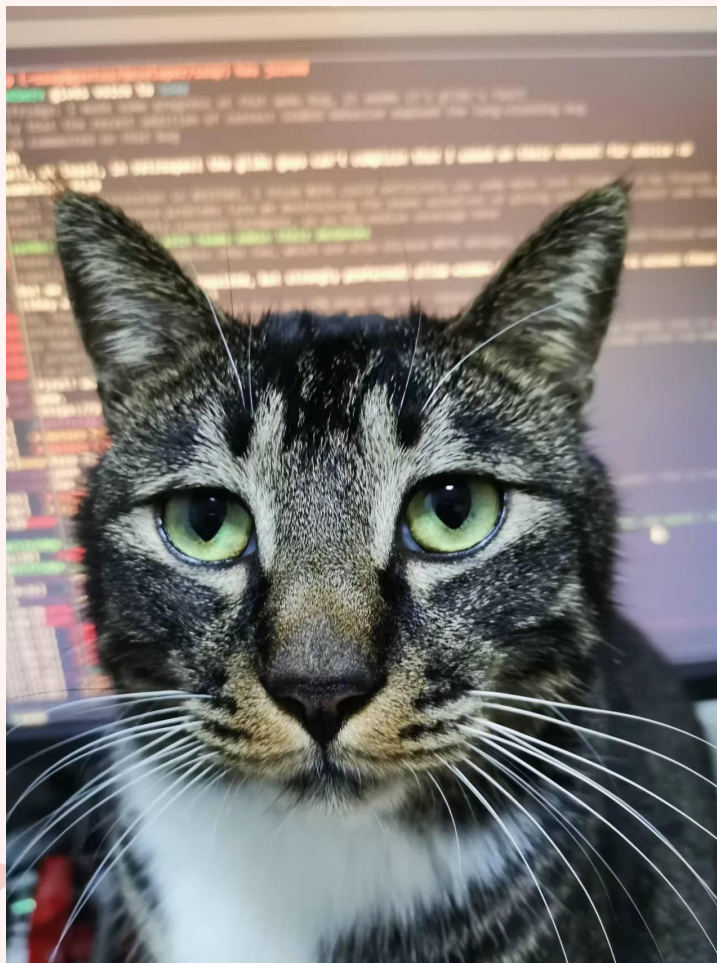
叠甲环节

- 观点部分的内容仅代表个人立场
- 所有需要提及具体公司、商标的场合，均出于善意与语言组织必要



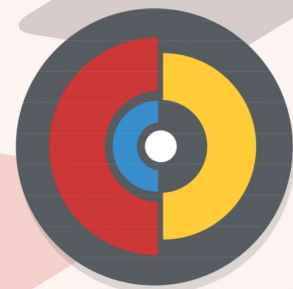
我是谁？

- GitHub [@xen0n](#)
- FLOSS 贡献者，爱折腾
 - AOSC 贡献者
 - Gentoo 开发者
 - 龙架构 (LoongArch) 基础软件
- 鸽派内容创作者
 - 咱龙了吗? areweloongyet.com



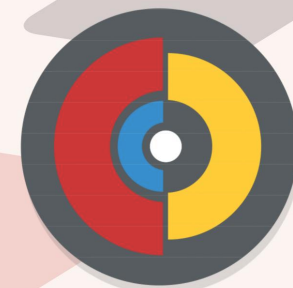
内容提要

- 如何维护 Firefox 的架构适配
- 如何维护 Firefox 的架构适配
- 如何维护 Firefox 的架构适配



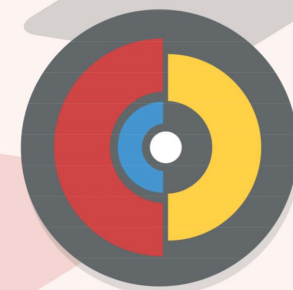
内容提要

- 如何维护 Firefox 的**架构适配**
- 如何**维护** Firefox 的架构适配
- **如何维护 Firefox** 的架构适配
- 什么是适配
- 什么是软件包维护
- 实战解析：为龙架构 Firefox 做些优化！



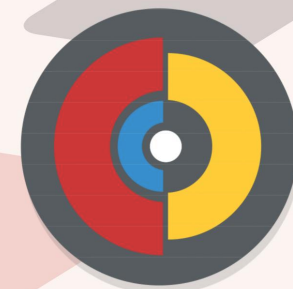
第一部分

什么是适配？



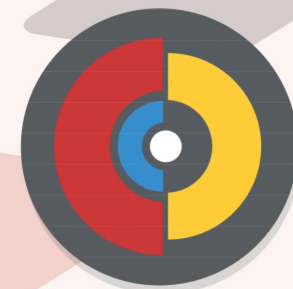
浏览器不是跨平台的吗，怎么还要适配

- 跨平台不是「消灭」平台差异，而是尽量不向上层「暴露」平台差异
- 跨平台技术的「底层」一定平台细节满满
- 平台适配的目的：
让软件在目标平台上「能用」、「好用」
 - 在目标平台能够成功构建
 - 功能特性与其他平台实现对齐（feature parity）
 - 尽量发挥目标平台的优势



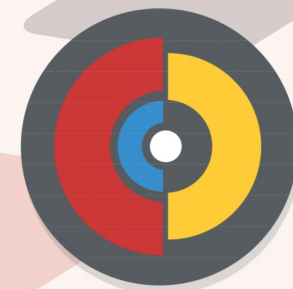
什么是「平台」？

- 可以理解为我们对运行环境的抽象概括
- 一般取「处理器架构、操作系统、用户态环境」的组合就足够了
 - 叫作「目标元组」(target tuple)
 - 有时也用到「供应商」字段
- 举例
 - `aarch64-linux-android`
 - `hppa2.0-unknown-linux-gnu`
 - `loongarch64-gentoo-linux-musl`



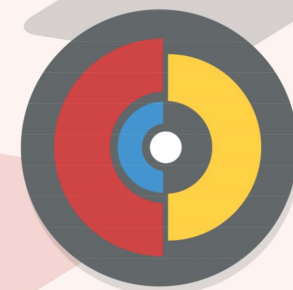
现代浏览器都有哪些平台相关特性

- 文件系统、网络、进程间通信等
 - 仰仗操作系统原语，如 POSIX 或 Win32 API
- JavaScript 运行时
 - JIT 到宿主架构机器码
- 音视频、GPU
 - 平台提供的 API 各不相同，如 OpenGL / Metal / D3D
- 桌面环境集成
 - GUI 套件不同，如 GTK(2/3/4) / Qt / Win32 / ...



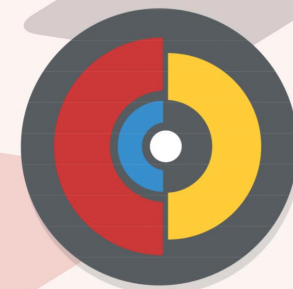
第二部分

什么是维护？



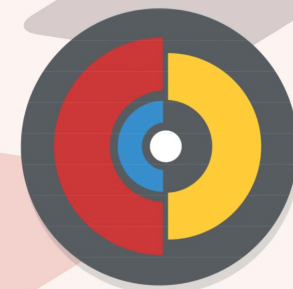
「维护」的两个视角

- 不宜将「制作」软件与「打包」软件混为一谈
 - 正如「种菜」与「炒菜」大相径庭
 - 有时涉及矛盾需求的取舍，例如：是否捆绑（bundle）或静态链接依赖组件？
- 但这两种工作是「软件维护」的一体两面
 - 只做上游不打包：可能对下游打包者造成困扰
 - 只打包不推上游：下游补丁集愈发沉重，维护不可持续
 - 应保持开放心态，持续学习，尽量做到协调



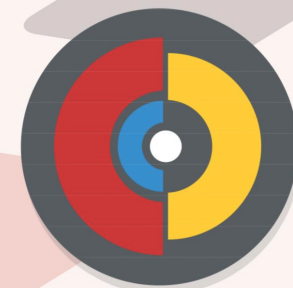
上游视角的「维护」

- 好比「种菜」
 - 关注点在于品种、农技、产品作为菜的各项指标等等
- 实现需求：新功能、需求变更；性能、安全等方面
- 软件工程：重构；质量保障；CI/CD etc.
- 支持承诺：哪些工况下发现的问题才算作问题；已发布的版本维护多久 etc.



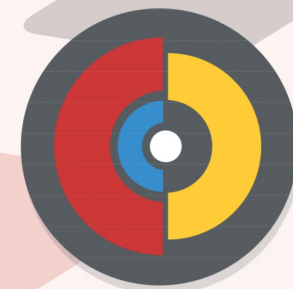
打包人视角的「维护」

- 好比「炒菜」
 - 关注点在于本店面向什么食客群体，进而以何种烹饪技法将手头的原材料做出预期的味型
- 配置：启用应该启用的，禁用应该禁用的
- 缝补：补充我需要考虑而上游没考虑到的
 - 上游捆绑 (bundle) 依赖库，而我必须引用系统里的
 - 上游不支持某个平台/支持有小问题，而我必须完美支持
 - 上游的 A 功能只能搭配 X 库，而我只有跟 X 95% 像的 Y
- 跟进版本更新，维护下游补丁集：「菜谱」(recipe)



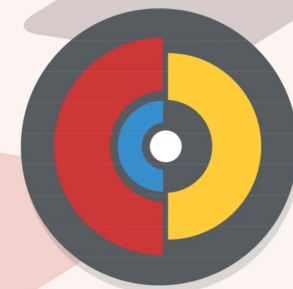
打包人需要具备的素质

- 沟通能力！
 - 肯定要改东西，啥都不用改就能用的话，打包人都失业了
 - 叫别人改东西约等于叫人承认自己错了，迟早会遇到矛盾
- 快速了解陌生的上游项目
 - 技术栈 & 架构、构建系统、关键组件的业务逻辑
 - 在这个项目办事的正规流程
- 熟悉目标平台与发行版的各种技术规范
 - 并且知道它们制定出来是为了追求或者防止什么，这样才能避免教条主义，进而保障用户体验



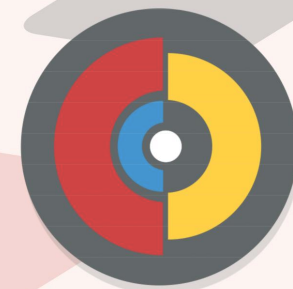
为何建议打包人把补丁推上游

- 问题一般都有共性，架构/平台适配问题尤其如此
 - 只有在这个架构/平台才会遇到
 - 不发声则问题难以得到关注、解决
- 降低下游持续维护工作量
- 惠及所有相关发行版及用户



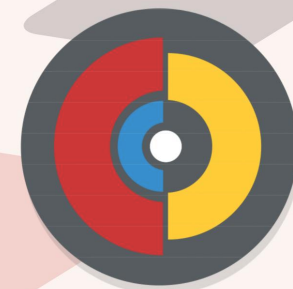
实战解析

为龙架构 Firefox 做些优化!



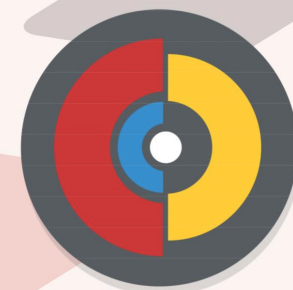
一些知识铺垫：Firefox

- Firefox 的主线代码仓库叫 mozilla-central
- 方便的 N 合 1 仓库叫 mozilla-unified
- 以上仓库可以用 hg 也可以用 git 访问
- Firefox 的构建系统是自家的，基于 Python 的 mozbuild
- Firefox 依赖很多第三方组件
- 补丁提交流程在[这里](#)



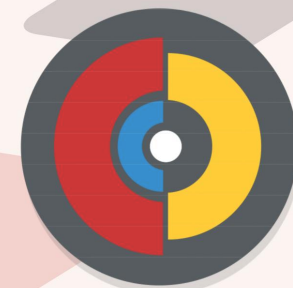
一些知识铺垫：龙架构 (LoongArch)

- 是一种精简指令集，能看出有其他精简指令集的影响，但和它们都不一样
- 由基础指令集与可选的几个扩展组成
 - LSX = Loongson SIMD eXtension, 128-bit SIMD
 - LASX = Loongson Advanced SIMD eXtension, 256-bit



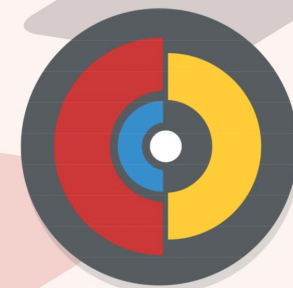
案例一：停止因噎废食

- 注意到先前的补丁集在 Firefox 135 打不上了
 - `git blame` 定位问题原因
- 发现有人把 libpng 的 LSX 优化[禁用了](#)
 - 理解动机：此人关注的是修复编译，暂且无暇顾及性能
- 寻找共同立场：既要正常构建，也要性能
 - 不禁用 LSX 优化的话怎么失败的？表象为缺少函数实现
 - Firefox 的第三方依赖管理机制：复制，但是有选择性
 - 根因：上一个升级 libpng 版本的人没有把龙架构优化的源码带进来
- 解决方案：修复 libpng 代码同步配置，补充遗漏的文件



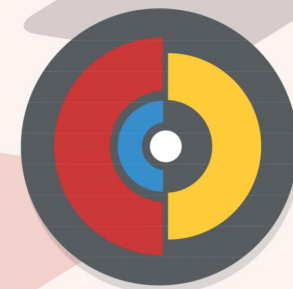
案例二：启用 WebRTC 的正确姿势

- AOSC OS 早期 (2023.12) 即为龙架构 Firefox 启用了 WebRTC
 - 但是后续发现维护不便，观察补丁，改动的 moz.build 文件是自动生成的！
- Firefox 对 libwebrtc 的集成方式
 - 也是第三方组件，来自 Chromium，原本采用 GN 构建
 - 检索资料，发现它的 moz.build 文件是自动生成的
 - 根因：该生成器不知道有龙架构
- 解决方案：为 mozbuild GN 处理器增加龙架构支持，更新 libwebrtc moz.build



后记：如何与龙友们建立联系

- 论坛
 - 「龙芯爱好者社区」：<https://loongbbs.cn>
 - 龙架构非官方开源社区：<https://bbs.loongarch.org>
- GitHub [@loongson-community](#)
 - 咱龙了吗? [areweloongyet](#)
 - 跨项目协作讨论区 [discussions](#)
 - 具体项目的龙架构相关事项，一般优先在相应项目讨论，并在此处建立跟踪工单，这样圈子里的人都能看到



感谢您能听完!

问答环节

