如何看待 left-pad 事件?

---

johnhax.net/2016/all-about-left-pad

（观赏本 slide 请使用 Chrome 45+ 等支持 ES6、Fetch API 等新特性的浏览器）

by Hax

Introduction of my company and myself

百姓网

www.baixing.com

分类信息

58赶集合并后自动排名上升一位

昨天新三板上市敲钟

Send your resume to

简历请投

heshijun@baixing.com

github: @hax

zhihu: 贺师俊

weibo: @johnhax

争议性

JavaScript: The World's Best Programming Language

程序员的圣战之
TAB vs SPACE

回避

嘲笑

left-pad

回顾事件

- How one developer just broke thousands of projects
- 开发者对 npm 公司不满，unpublish 了自己的所有模块

技术批评

- Have We Forgotten How To Program?
- 我们是不是早已忘记该如何好好地编程?

知乎讨论

- 如何看待 Azer Koçulu 删除了自己的所有 npm 库?
- NPM中将函数当作一个包发布的方式是否合理?
- 如何评价《The current state of JS programming》?

行为艺术

- left-pad as Service
- Gives you five
- is 13?

狂欢

- Ruby
- C#

- Rust
- Haskell

回避

嘲笑

我就是认真

如何看待 left-pad 事件?

- 谁是这次事件的罪魁祸首？azer？kik 公司？npm 公司？
- azer 是英雄还是熊孩子？
- npm 公司是不是邪恶的公司？
- 开源是不是注定不靠谱？
- 包是不是应该有命名空间？
- 我们都忘记怎么编程了吗？
- 小模块是好是坏？
- leftpad 本身应该怎么实现？
- 第三方依赖是好是坏？
- 是不是应该使用 shrinkwrap/lock（锁定依赖）？
- 是不是应该使用 bundle/pack（打包依赖）？
- 是不是应该自建仓库？
- js/nodejs/npm/前端…… 生态是不是有问题？

- 谁是这次事件的罪魁祸首？azer？kik 公司？npm 公司？
- azer 是英雄还是熊孩子？
- npm 公司是不是邪恶的公司？

谁是这次事件的罪魁祸首?

- azer
- kik 公司
- npm 公司

- Azer NPM 撤包事件全信件

翻译是否准确?

立场先行?

缺乏npm官方回应?

- A discussion about the breaking of the Internet
- I’ ve Just Liberated My Modules
- kik, left-pad, and npm

开源不是单向的贡献，而是互惠的

No Warranty ≠ 无责任

应该遵循 npm 的规则

（当然我们也可以质疑 npm 的规则/行为/假设）

关于Azer

伊拉克 土耳其 战争

I Owe My Career To An Iraqi Immigrant

一些臆测……

冲动是魔鬼

- 谁是这次事件的罪魁祸首？azer？kik 公司？npm 公司？
- azer 是英雄还是熊孩子？
- npm 公司是不是邪恶的公司？

开源是不是注定不靠谱?

left-pad#4

npm 官方应对

快速发现问题

各方快速响应

自发的临时解决方案

npm 官方方案

事后追踪和改进

社区各种讨论

开源是不是注定不靠谱?

开源社区很靠谱!

包是不是应该有命名空间?

npm#798

包命名空间有利有弊

不是本次事件的核心原因

我们都忘记怎么编程了吗?

小模块是好是坏?

AMA: One-line node modules

重新思考

我们是不是早已忘记该如何好好地编程?

Deprecation notice of line-numbers

AMA: What are your thoughts on the left-pad npm case?

关键不在于行数

而在于抽象

第三方依赖是好是坏?

你总需要依赖他人

依赖 => 信任

需要找到平衡

依赖带来的问题

- 依赖太多太深，安装慢
- 依赖链太深，安全隐患
- 依赖太复杂，升级困难

npm 依赖体系与其他的不同

- npm 从很早就支持局部依赖，并很方便的安装管理包
- JavaScript 历史上缺少标准库，所以基础小包多
- 动态语言支持 duck type，从而可以多版本并存
- js module 没有 global namespace，从而可以多版本并存

JS/NPM 的历史条件决定了

它和小模块方法论是相适应的

所以我们不能简单套用其他语言

的经验来评判 JS/NPM（反之亦然）

JS/NPM 给出了一种新的可能性

当然也会面对新的挑战

社区正在涌现相关解决方案

- ied, npminstall
- david, alinode
- greenkeeper

- 是不是应该使用 shrinkwrap/lock（锁定依赖）？
- 是不是应该使用 bundle/pack（打包依赖）？
- 是不是应该自建仓库?

It depends

但我建议只在真的必要的时候使用，也就是

- 通常不需要 shrinkwrap/lock
- 通常不需要 bundle/pack
- 通常不需要自建仓库

left-pad 本身应该怎么实现?

月影的实现

左耳朵耗子

论left-pad的实现

性能优化应该针对真实 use case

microbenchmark 意义不大

优先使用标准库和 polyfill

js/nodejs/npm/前端……

编程方式/体系/生态/社区……

是不是有问题?

有?

没有?