# 远离敏捷状态的 Scrum 框架

> 尽信书不如无书 ——《孟子·尽心下》

期望我的每一篇文字都会招来一些拥有观点的声音，为了这个目的我尽可能使我的文章充满观点，有些时候是犀利的观点。

敏捷入门的话很多人一般都会选择 Scrum 框架进行学习和导入。但笔者在国内导入敏捷的时候，发现 Scrum 框架有很多缺陷，而且这些缺陷甚至让团队远离“敏捷”的状态（ Being Agile ）。商业上的成功让这一框架家喻户晓，但认证也让这一框架拥有无数的原教旨主义者。有些时候当我想表达一些观点的时候，好像很多人的潜台词就是Scrum是不可撼动的，那好……今天咱们就掰一掰。

## 那什么是敏捷呢？



很多人会说敏捷就是一种心态（ Mindset ），老太太 Linda Rising 说有两种心态：

- 固定心态认为
  - 能力：恒定的，如身高
  - 目标：有面子
  - 挑战：避免
  - 失败：影响身份
  - 努力：没有天赋的人
  - 对挑战的响应：无助的

- 敏捷心态认为
  - 能力：能成长，如肌肉
  - 目标：去学习
  - 挑战：拥抱
  - 失败：提供信息
  - 努力：成为高手的路径
  - 对挑战的响应：执着、韧劲

她又说到：

- **敏捷心态**相信所有的一切都是在过程之中；
- 敏捷软件开发过程不是固定的。当我们越了解软件开发过程就会越变化和成长。因为这个过程永远没有完美的一天，所以变化和成长会不停的延续下去；

如果说用一句话来形容什么是敏捷：

> Ever tried. Ever failed. No matter. Try Again. Fail again. Fail better. ——Samuel Beckett, Irish poet(1906-1989)

观点：

> - 敏捷是一种不停尝试、不停调整、不停优化的状态。（Being Agile）

参考：

> 敏捷软件开发宣言：描述了敏捷软件开发的价值观
> The Power of an Agile Mindset - Linda Rising：敏捷心态的幻灯片
> 消失的仪式感：Doing 与 Being的文章
> Mindset：描述固定心态和增长性心态书籍的豆瓣链接

## 但你会说：这太难了



是，这太难了。读到这里你肯定有一种被命运抛弃的感觉。你会说看上面的东西好像啥都没说啊，那我怎么做呢？是的，一般人都必须从“守”开始。什么是“守”？守就是告诉你什么你做什么。


那告诉什么呢？那就是：

- 需求如何一步一步变成软件；
- 有几个会议，如何开会；
- 有几个角色，到底干些啥；

但如果你止步于“守”的阶段，不好意思，你不要说你和敏捷有什么关系。尽管你可能2周产出一个能上线的软件，尽管需求可以实现增量型的交付，尽管设定了几个角色。

敏捷其实更强调“破”和“离”的状态，而不是“守”的状态。尽管“守”的过程是重要的。“守”你可以理解就是Doing，“破”或“离”就是Being。（Being就是上一段所说的内

容）

用两张图来表达这两种区别（绿色的是你做“敏捷”的事情）：

| | 周一 | 周二 | 周三 | 周四 | 周五 |
|---|---|---|---|---|---|
| 9:00-9:30 | 晨会 | 晨会 | 晨会 | 晨会 | 晨会 |
| 9:30-10:00 | 计划会议 | | | | |
| 10:00-10:30 | | | | | |
| 10:30-11:00 | | | | | |
| 11:00-11:30 | | | | | |
| 11:30-12:00 | | | | | |
| 13:00-13:30 | | | | | |
| 13:30-14:00 | | | 需求梳理会议 | | |
| 14:00-14:30 | | | | | |
| 14:30-15:00 | | | | | |
| 15:00-15:30 | | | | | |
| 15:30-16:00 | | | | | |
| 16:00-16:30 | | | | | |
| 16:30-17:00 | | | | | |
| 17:00-17:30 | | | | | 审核会议 |
| 17:30-18:00 | | | | | 回顾会议 |



| | 周一 | 周二 | 周三 | 周四 | 周五 |
|---|---|---|---|---|---|
| 9:00-9:30 | 晨会 | 晨会 | 晨会 | 晨会 | 晨会 |
| 9:30-10:00 | 计划会议 | | | | |
| 10:00-10:30 | | | | | |
| 10:30-11:00 | | | | | |
| 11:00-11:30 | | | | | |
| 11:30-12:00 | | | | | |
| 13:00-13:30 | | | | | |
| 13:30-14:00 | | | 需求梳理会议 | | |
| 14:00-14:30 | | | | | |
| 14:30-15:00 | | | | | |
| 15:00-15:30 | | | | | |
| 15:30-16:00 | | | | | |
| 16:00-16:30 | | | | | |
| 16:30-17:00 | | | | | |
| 17:00-17:30 | | | | | 审核会议 |
| 17:30-18:00 | | | | | 回顾会议 |

这两种时间分配比例达到 **9比80**，这也就是说Doing重要，因为如果不是Doing我们不知道如何走向Being。但真正的敏捷是融合在日常工作之中，而不是所谓的几个会议、角色、工件之中。

如何让大家用Doing来关注到Being的状态，是所谓“**框架**”应该关注的东西。也就是说，框架说让你躺下，闭着眼睛。自然就会达到休息的目的。一般人可能对如何休息不知道该如何去做，但一个框架性的东西可以让人关注具体从而实现一个较为模糊的目标。

观点：

> - 一个撇开实践的敏捷理论就是扯淡，必须让人们先做起来（Doing），然后通过练习慢慢提高到（Being）状态。

参考：

> 让看板成为团队的镜子：看板演进：如何通过不停的做事情而达到某种状态

# Scrum 框架与我对框架的期望

前几天我的这个文章的创意发出的时候，很多人在群里讨论。看到大家的讨论很有意思，我在这里非常明确的说出我的观点：



1. Scrum 就是 3355的组合
    - 3个角色（Product Owner产品负责人、Development Team开发团队、Scrum Master 斯格拉姆马斯塔）
    - 3个工件（Product Backlog - 产品待办列表、Sprint Backlog - Sprint待办列表、产品增量）
    - 5个事件（Sprint、Sprint计划会议、Daily Scrum每日站会、Sprint评审会议、Sprint回顾会议）（注意哈，如果叫Stand-up、Daily Stand-up、Daily Stand、Stand-up Meeting这不是Scrum中的词汇哦）
    - 5个价值观（开放、尊重、勇气、专注、承诺）

2. 同时强调了“完成”的定义（DoD）
    - 也就是什么叫“完成”的标准需要慢慢增加和扩大；

3. 在3355的背后是
    - 透明、检视、适应（这也是3355的目标）

Scrum就这么多东西。如果任何一个Scrum的培训，没讲到以上的内容。我建议你去向培训机构要求退钱。;)

在开拍Scrum之前，先说说我喜欢它的地方吧：

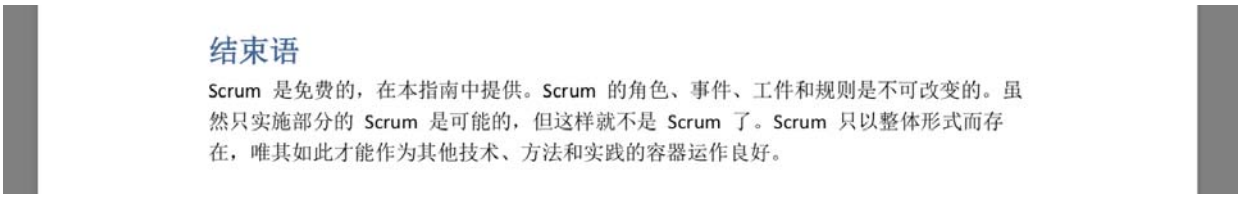
1. 对错（爱憎）分明
   在导入敏捷过程的时候没有什么比告诉别人对或错更为有效；
2. 结构简单
   简单可以在很多方面获得好处，比如教学方面、让别人更快理解方面，当然理论本

身的硬伤也会因为简单而更难显现；

3. 花钱考证
   你没看错，确实这是优点。良好的商业运作使得有持续的资金注入，让受众更广泛、也让讲师有意愿提高授课水平。

在开始描述我对一个框架的期望之前，我要强调一下Scrum的另一个可能大部分人会忽略的关键点：

在《Scrum指南》中第20页的**“结束语”**部分：


结束语

Scrum 是免费的，在本指南中提供。Scrum 的角色、事件、工件和规则是不可改变的。虽然只实施部分的 Scrum 是可能的，但这样就不是 Scrum 了。Scrum 只以整体形式而存在，唯其如此才能作为其他技术、方法和实践的容器运作良好。

看到了没有，不能改变。整体应用叫Scrum，部分应用叫ScrumBut。那太好了，我喜欢这个爱憎分明的家伙。我后面的文字，部分瞄准的也是它这句话。

我对一个“敏捷”框架的期望：

> 1. 简单（一个高中毕业的人就能够明白）且关注关键点；
> 2. 100%能落地（能够实施）且能够灵活的应对现实中的不同情况（覆盖广）；
> 3. 关注实践（Doing）并通过实践自然达到敏捷的状态（Being）；



观点：

> - Scrum是一个爱憎（对错）分明，多一分不多少一分则少的过程实施框架；（它本身是硬的，不能调整）
> - 人们通过具体的行为或做法来理解事物，一个好的**“框架”**会让大家从具体的事情着手并逐步走向更深层次的理解与实践（Doing -> Being）；

参考：

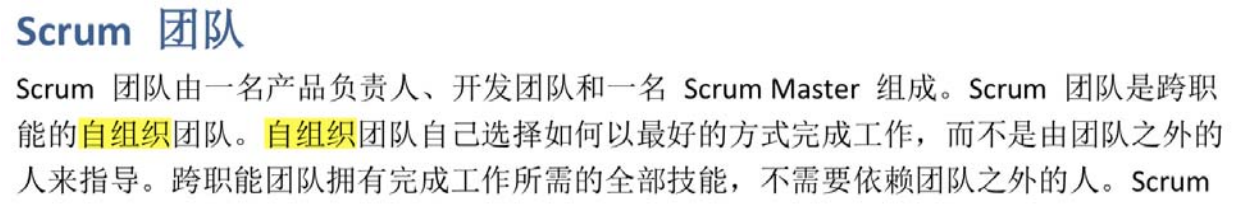
> Scrum指南：权威定义Scrum的文字；
> Scrum简章：一般我的培训课上都发这个资料；

# 关于纯真信仰的“自组织”

- 我曾经问过我所带领的团队成员：“你觉得自由是一件好事吗？”当时有些人点头有些人摇头。我非常明确的对他们说：**自由不一定是个好东西，对于那些无法独立思**

**考的人，自由对他们就是负担。**刚才那句话我一般会说两遍，当说完的时候我也看到很多赞同的点头。
- 中国从来就是精英社会，而不是民粹体系。你可能真诚的说：来看看你想干点什么活？他估计会回答你：领导，您说干哪块咱们就干哪块吧。这种状态根深蒂固，并且估计80%的团队成员是这样的。自组织个毛啊（这句话真心看着不像专业教练说的）。
- 真正重要的是
    - 让团队中少量的核心分子真正的积极起来，拥抱协作的敏捷文化；
    - 给剩下大部分跟着跑的团队成员成长空间，让他们知道如何提出请求，如何反馈意见，了解成为核心成员的成长路线。

Scrum指南第七页



Scrum 团队

Scrum 团队由一名产品负责人、开发团队和一名 Scrum Master 组成。Scrum 团队是跨职能的自组织团队。自组织团队自己选择如何以最好的方式完成工作，而不是由团队之外的人来指导。跨职能团队拥有完成工作所需的全部技能，不需要依赖团队之外的人。Scrum

Scrum指南第八页

开发团队具有下列特点：

- 他们是自组织的。没有人（即使是 Scrum Master）有权告诉开发团队应该如何把产品待办列表变成潜在可发布的功能增量；



观点：

> - 在中国自组织就是一个神话，现实之道是团结精英推行任何改进；

参考：

> Scrum与Kanban，激进与保守：路宁对于Scrum和看板的评价

# Scrum 框架的反敏捷之处

（1）关注角色而不是要达到的效果

- 伟大而又神奇的Scrum Master（斯格拉姆·马斯塔）
    - 好消息是：经过2天或3天的培训、考试之后，你会获得Scrum联盟发出的Scrum Master证书。恭喜你你被联盟认证了！
    - 坏消息是：在我经历过的国内团队之中，我认为靠谱的斯格拉姆·马斯塔人数在20人以下。

    - 也恰恰是因为不靠谱人员数量庞大，对团队造成了很坏的影响。这也使得我在国内导入的敏捷过程完全**不引入这一概念**。
    - 斯格拉姆·马斯塔的终极目标是消灭自己（形成真正的自组织团队之后），但在《Scrum指南》8-9页之中没有任何文字描述。而这一角色又堂而皇之的成为框架永恒的一部分。

- 缥缈的Product Owner（产品负责人）
    - 拜李国彪（Bill Li）所赐;P，基本所有Product Owner都被翻译成产品负责人了。但这是负责人吗？这不是**产品业主**嘛？（关于这点他也经常开玩笑）
    - 在中国的团队之中到底有多少能做得了主的“产品负责人”，增加了无数新的ScrumBut词汇。比如：PPO(Proxy Product Owner)，POA(Product Owner Assistant)；
    - 不好意思，你这玩的不是Scrum是ScrumBut。但纠结角色的时候是不是可以看看他/她管的事是否大家都可以分担了呢？
    - 产品负责人是负责管理产品待办列表的唯一负责人。产品待办列表的管理包括：
        - 清晰地表述产品待办列表项；
        - 对产品待办列表项进行排序，使其最好地实现目标和使命；
        - 优化开发团队所执行工作的价值；
        - 确保产品待办列表对所有人是可见、透明和清晰的，同时显示 Scrum 团队下一步要做的工作；以及
        - 确保开发团队对产品待办列表项有足够深的了解。
    - 在导入敏捷过程的时候，恰恰有很多个角色或个人能够满足这些要求。那太过于强调这个角色是否又是一个合理的坚持呢？



（2）关注会议而非会议之前的准备

- Scrum有对会议的精细安排，比如建议多长时间，比如结构分成几个部分。但恰恰是这种划分使得使用者的注意力关注到会议本身。会议的时间越来越长，效率越来越低。但只要有特定的会议并且关注Scrum所要求的方向就达标啊。

（3）冗余的概念

- Product Backlog和Sprint Backlog
    - Scrum框架告诉我们说Product Backlog是PO的资产，Sprint Backlog是团队的资产。但本身作为需求的表现形式来说，就应该是平衡业务与技术。一方面如果推迟进行技术角度的拆分其实就是放大项目后期的风险；另一方面团队需要尽早考虑后面的交付目标。所以对于需求不应该区分这两个概念，应该用一个概念所代替。这里建议就用User Story（用户故事）代替Product Backlog和Sprint Backlog。

（4）框架的缺失

- 需求DEEP中的D就是扯淡
    - 这里的D就是Detailed Appropriately（细节合适的）：越临近开发越精细，离开发越远越粗略。多精细？多粗略？框架中这部分内容的缺失是遗憾的。初学者根本无法掌握精细和粗略的火候；

    - 这点在我实施敏捷导入的是否非常关键，也就是如何能够实现“渐进式需求分析”；
    - 建议使用：SBDEEP（详细的解释请参见《Product Backlog的细节合适就是扯谈》）

（5）不精确的词汇

- Backlog：这个词是啥意思？就是要干的事，但咱们了解Scrum 中 Product Backlog概念的人知道。Backlog放在下面的不一定要做啊，随着时间的推移，新的需求加入，优先级低的Backlog可能永远都不会做。
    - 我个人也倾向不使用Requirement这个词，Item、Story可能会更为合适。
    - 这里给Kanban方法点个赞，在那个世界这种东西叫Idea。就是个想法，想法做不做不一定。

- Scrum Master：啥叫Master？大师？两天培训考试之后就大师了？所以我更倾向叫这些人斯格拉姆·马斯塔。这种不是基于能力的认证实在让我痛心，记得我之前朋友圈发过一条信息：认证认证了什么？认证的边界是什么？
    - Sprint：冲刺？这词汇我每次培训都花了不少言语进行解释。这里不是说的猛，不是拼命，而是短。但越了解Scrum其实感觉，框架就是让团队成员抱起团来往前冲。很多实施敏捷的团队说敏捷就是加班，可能也有几分源于此吧（YY一下）。



（6）框架补充的堆砌（非Scrum框架本身）

- 正因为Scrum只是一个框架，所以很多培训讲师为了使Scrum更为落地填充了很多内容。但一些内容确实有些过时了，有点时候感觉是知识的堆砌，甚至某些部分过于关注感受的冲击，比如：
    - 过时：估算扑克
        - 1、2、3、5【8、20、40、60、100】括号中的数字卡片都可以扔掉了，估算能拆成这么大的东西，风险无法控制。
        - 打扑克、伸手指头太慢了。悲催的是有些Scrum培训还在教这种方式，比较靠谱的解释是他们扑克印多了。建议使用《批量用户故事快速估算方法》

    - 堆砌：在课程中一堆需求分析的方法，一堆创新思维的工具，一堆估算的方法，一堆模型。真正用到的又有几个？

总结一下

> 1. 关注角色而不是要达到的效果：反敏捷评级【四颗星】
> 2. 关注会议而非会议之前的准备：反敏捷评级【五颗星】
> 3. 冗余的概念：反敏捷评级【三颗星】
> 4. 框架的缺失：反敏捷评级【一颗星】
> 5. 不精确的词汇：反敏捷评级【一颗星】
> 6. 框架补充的堆砌（非Scrum框架本身）：反敏捷评级【一颗星】

回到我对一个敏捷框架的期望，并综合上面所表达的观点进行Scrum满足程度评估：

> （1）简单（一个高中毕业的人就能够明白）且关注关键点；
>
> 足够简单，但有些关键点它错过了。比如Detailed Appropriately的Product Backlog。
>
> （2）100%能落地（能够实施）且能够灵活的应对现实中的不同情况（覆盖广）；
>
> - 虽然能在小型团队100%落地，但其僵化的框架本身无法在其他场景进行应用。所以在任何培训之前，建议明确把其限制进行非常明确的标注；
> - 其本身的僵化无法应对现实中的许多情况，比如不靠谱的Scrum Master，模糊的PO；
>
> （3）关注实践（Doing）并通过实践自然达到敏捷的状态（Being）；远远没有达到我的期望

观点：

> - 对于一个框架你可以任性，但这也显示出了对细节的不负责任；
> - Scrum商业化的成功，使得学员不是在学习，而是在消费。这种消费与讲师形成的共生进化关系，产生了更多的包装。包装产生业界大牛，大牛导致更多人的膜拜。这恰恰说明为啥CST这么火，CEC无人问津的现状（国内8个CST，1个CEC，1个CTC）。



# 我所坚持的导入框架

好，干货时间到哈。基于 Scrum 的框架吧，说说我的调整：

- 2个角色
  - 改进委员会
    - 3-9人组成的领导力团队；
    - 职责：
      - 定义迭代目标；
      - 识别债务任务；
      - 改进团队：缩短LeadTime（前置时间）、降低版本缺陷数、提高团队节奏指数、起床指数、纪律指数；
  - 团队
    - 20人以下的交付单元，包括改进委员会所有成员；
    - 职责：
      - 完成迭代目标，完成债务任务；

        - 实现工分增量需求，制造出可上线的产品增量；

- 2个工件
    - 工分增量
        - 特性需求。由改进委员会确认的形式（可以由团队外部产生）进行产出，符合SBDEEP原则；
            - S：Shit（大便，未经沟通的充满风险的需求）
            - B：Brick（板砖，经过沟通完成估算的需求；特点是粒度相似风险可控）
            - D：Diamond（钻石，明确需求边界的需求；比如切图、验收条件等完善）
            - E：Emergent（浮现的）
            - E：Estimated（有估算的，估算为1、2、3、5这几种工分）
            - P：Prioritized（排序的，需要与债务任务一起进行排序，形成统一序列）

    - 债务任务
        - 这种任务不进行估算，但计算完成时间。包括：
            - 缺陷（Bug）功能债务
            - 技术债务（Tech Debt）
            - 团队债务（影响LeadTime前置时间、版本缺陷、节奏感、起床指数、纪律指数的任何任务）

    - 5个事件
    - 迭代
        - 同Scrum的迭代概念，增加一个要求：迭代内只能实现“钻石”级别的需求；
        - 需求在改进委员会所有成员的认可前提下，迭代之中可以增加新的“钻石”级别需求；

    - 迭代计划会议
        - 进入迭代的仪式
        - 要求：
            - 明确本迭代开发范围以及开发细节在会议之前必须完成；
            - 会议中团队中的所有人举手确定明确本迭代的所有细节以及迭代目标；
            - 改进委员会代表选读本迭代目标，并宣布迭代正式开始；

    - 每日站会
        - 每天早上进入工作状态的仪式
        - 要求：
            - 每个人说出自己昨天工作的关键点；
            - 改进委员会成员说出今天自己的工作安排；
            - 如果发现任何一个人与最新交付状态信息不同步的时候，改进委员会成员要明确指出此违规项；

        - 站会不是等着把昨天事情拖到今天再说的会议；
        - 举例：【开发】我昨天移交测试了一个需求；【测试】哪个需求？（这里发生了信息不一致的现象）

- 演示会议
    - 改进委员会成员引导的，确认工分的仪式；
    - 要求：
        - 邀请业务相关方参与会议；
        - 工分获得的标准需要改进委员会成员提前形成规范；
        - 用用户能够感知的方式展示工作成果；
        - 此工作成果能够进行发布；

- 改进委员会回顾会议
    - 特征：改进委员会小组思考债务提高测量指标的会议
    - 要求：
        - 收集开发团队的测量指标（所有人）
        - 针对测量指标产生技术债务和团队债务任务并明确落实人；

- 5个测量
- LeadTime前置时间：从需求提出到上线的时间；
- 版本缺陷数：每个上线版本所发现的缺陷；
- 节奏指数：主观指数
    - 5分：团队非常有节奏感，有条不紊
    - 1分：杂乱无序

- 起床指数：主观指数，我早起床之后，想到上班
    - 5分：就安奈不住心中激动的心情。
    - 1分：就让我死在床上吧

- 纪律指数：主观指数，团队的任务执行
    - 5分：如钢铁般的纪律，特种兵一般
    - 1分：毫无纪律可言，乌合之众，土匪一般

迭代理解地图：

- 迭代计划会议：每个人都没问题；
- 非框架会议：需求梳理会议（理解校准）
- 演示会议：证明你做完了；


观点：

- 把价值观放入框架是假把式，重要的是如何通过框架实现潜在的价值观；
- 适应和调整是自然的，当你看到东西并拥有选择的时候；



## 放弃质疑的知识崇拜与跟随惯性

前几年我尝试用3个多小时的时间把我现在的工作内容和我母亲讲清楚，言语犀利的母亲淡淡的说：**哦，你就是个骗子**。当时哑口无言的我竟然无法辩驳。可能也就是那个时间点之后，在工作之中我不停的在心中问自己：当前的工作与骗子有多少关系。

这可能也是我不愿意全身心投入讲师队伍中的一个理由，我害怕我不停的讲啊讲啊，把自己的脑袋洗了无数遍……骗自己的骗子是多么的恐怖啊……

还记得很久之前在ThoughtWorks工作的时候，从英国和印度来的架构师与我们一起工作。我们对他们非常认可，很多细节的决定都埋没了自己的观点。回想起来当时我们的观点是如此的正确。对于敏捷的理论我们也是对着很多外国牛人抱着崇敬的目光仰视着他们。

但这一切就是这个样子吗？

我去年已经放弃了CSM、CSPO、CSP、管理3.0讲师、Happy Melly的证书。轻装迎接更为严酷的挑战……

观点总结：

- 敏捷是一种不停尝试、不停调整、不停优化的状态。（ Being Agile ）
- 一个撇开实践的敏捷理论就是扯淡，必须让人们先做起来（ Doing ），然后通过练习慢慢提高到（ Being ）状态。
- Scrum是一个爱憎（对错）分明，多一分不多少一分则少的过程实施框架；（它本身是硬的，不能调整）
- 人们通过具体的行为或做法来理解事物，一个好的**“框架”**会让大家从具体的事情着手并逐步走向更深层次的理解与实践（ Doing -> Being ）；
- 在中国自组织就是一个神话，现实之道是团结精英推行任何改进；
- 对于一个框架你可以任性，但这也显示出了对细节的不负责任；
- Scrum商业化的成功，使得学员不是在学习，而是在消费。这种消费与讲师形成的共生进化关系，产生了更多的包装。包装产生业界大牛，大牛导致更多人的膜拜。这恰恰说明为啥CST这么火，CEC无人问津的现状（国内8个CST，1个CEC，1个CTC）。
- 把价值观放入框架是假把式，重要的是如何通过框架实现潜在的价值观；
- 适应和调整是自然的，当你看到东西并拥有选择的时候；

感谢王威（ Andy Wang ），要不是他，这篇文章不会出现在你们面前。

也谢谢读到这里的你，我说的这些你本来就都知道，我只是勾起了你的回忆而已……