目录

[一、管理方法 3](#_Toc62943305)

[目标 3](#_Toc62943306)

[需求 3](#_Toc62943307)

[流程 5](#_Toc62943308)

[时间 6](#_Toc62943309)

[沟通 6](#_Toc62943310)

[风险 6](#_Toc62943311)

[成长 6](#_Toc62943312)

[会议 6](#_Toc62943313)

[二、人生和职业管理 7](#_Toc62943314)

[专业技能 7](#_Toc62943315)

[程序员成长树 7](#_Toc62943316)

[专业能力快速提升的渠道 8](#_Toc62943317)

[时间的重要性 10](#_Toc62943318)

[梦想金字塔 11](#_Toc62943319)

[临界知识 11](#_Toc62943320)

[画出自己的技能树 15](#_Toc62943321)

[珍惜你的业余时间，那是你弯道超车的资本 15](#_Toc62943322)

[推荐的书籍 24](#_Toc62943323)

[《番茄工作法》 24](#_Toc62943324)

[推荐的书籍 24](#_Toc62943325)

[三、管理工具 25](#_Toc62943326)

[Remine 25](#_Toc62943327)

[安装 25](#_Toc62943328)

[非管理员如何新建项目 25](#_Toc62943329)

[项目排序 25](#_Toc62943330)

[四、代码规范 26](#_Toc62943331)

[命名风格 26](#_Toc62943332)

[五、管理技能 28](#_Toc62943333)

[管理就是做好一系列决策 28](#_Toc62943334)

[管理，就是面对事实解决问题 29](#_Toc62943335)

[六、工作感悟 30](#_Toc62943336)

[软件开发团队中，技术不是万能的，但没有技术是万万不能的 30](#_Toc62943337)

[详细制定自己软件开发专业知识学习计划，并注意及时修正和调整 31](#_Toc62943338)

[不要仅局限于对某项技术的表面使用上，哪怕你只是偶尔用一、二次 31](#_Toc62943339)

[在一种语言上编程，但别为其束缚了思想 32](#_Toc62943340)

[养成总结与反思的习惯 32](#_Toc62943341)

[理论与实践并重，内外双修 32](#_Toc62943342)

[心态有多开放，视野就有多开阔 33](#_Toc62943343)

[尽量参加开源项目的开发 33](#_Toc62943344)

[不要将自己的知识面仅仅局限于技术方面 34](#_Toc62943345)

# 一、管理方法

目标

当项目经理开始一个项目的时候，不论客户说什么，需求文档是怎么写的，甚至不管合同是怎么签的，你心里必须把一件事情想得非常清楚，那就是： 客户（老板）到底为什么要花钱做这件事？

项目经理必须学会从“业务”的角度去思考问题，而不仅是技术实现。

作为项目经理，当你选择技术方案、制定项目规划、确定验收标准时，客户的“业务目标”远比“技术实现”重要得多。首先要搞明白的是为什么做，而不是怎么做。

需求



综上所述，需要要找对人。

项目经理需要“强势”一些，但不是“强硬”，用各种可能的办法“逼迫”老板和你一起把问题考虑清楚。你可以要求他签字，可以软磨硬泡，可以晓之以理动之以情……

多数情况下，需求并不来自技术部门，而是来自业务部门。作为项目经理，一定要能够把自己提升到业务的高度，有能力和业务层面的人员直接进行沟通对话。

项目合同中一定不能存在所谓的“敞口协议”，也就是无法量化进行验收的条款。

流程

同样一件事情，对于“技术牛人”来讲，5天就可以完成，对于一般技术人员，通过标准的流程与方法，需要10天完成。对企业来讲，这两种做法哪一种更有价值呢？

流程的价值在于结果的“可复制性”，或者说是可控性。 靠人完成的工作往往有非常大的风险，因为人是最容易出问题的。同时，对于企业管理层来讲，如果只有一个项目在跑，一切细节尽在掌握，完全不需要流程。但如果有大量项目同时进行，不可能对每一个细节、每一个人的能力都有准确的把握。这时候一套成熟的机制与流程才是最为安全可靠的。 对于项目经理而言（特别是大企业中的项目经理），往往要适应在流程中工作。是的，你可以做得更好，但公司不是这样做事情的。你要学会的是在一系列的流程与管理框架中推动项目，提高效率。

好运不可能总是降临，做事情关键是要有组织有计划，这样才能达到预期效果。

作为项目经理，切忌陷入大量细节当中。即使你同时是专家，要处理技术问题，但项目经理这个角色要求你跳出这些问题，从业务、管理、团队、项目计划、资源协调等层面对项目进行控制。 你要保证：每一个细节都有人在做，但一定不能都是你在做。

时间

时间就是效益，我们自身的高运转，就是区别于别人的竞争力。

沟通

经常与团队同事交流，了解工作情况的一手信息。

工作是团队性的，需要通过及时有效的沟通，整合团队所有成员的经验和智慧。

更重要的是，你要让项目团队中的人都看到，你是在为大家努力争取资源的，否则你会一点点地丧失威信。

风险

一个简单的例子： 理想情况下，需要5个人做10天……你要想到什么？ 5个人够吗？需要什么样的技能？有经验的需要10天，没经验的呢？ 你有5个有经验的人，但他们有时间吗？这10天他们还有其他什么工作安排？

成长

要擅长从困难与挫折中习得经验，保持不断学习与总结的状态。

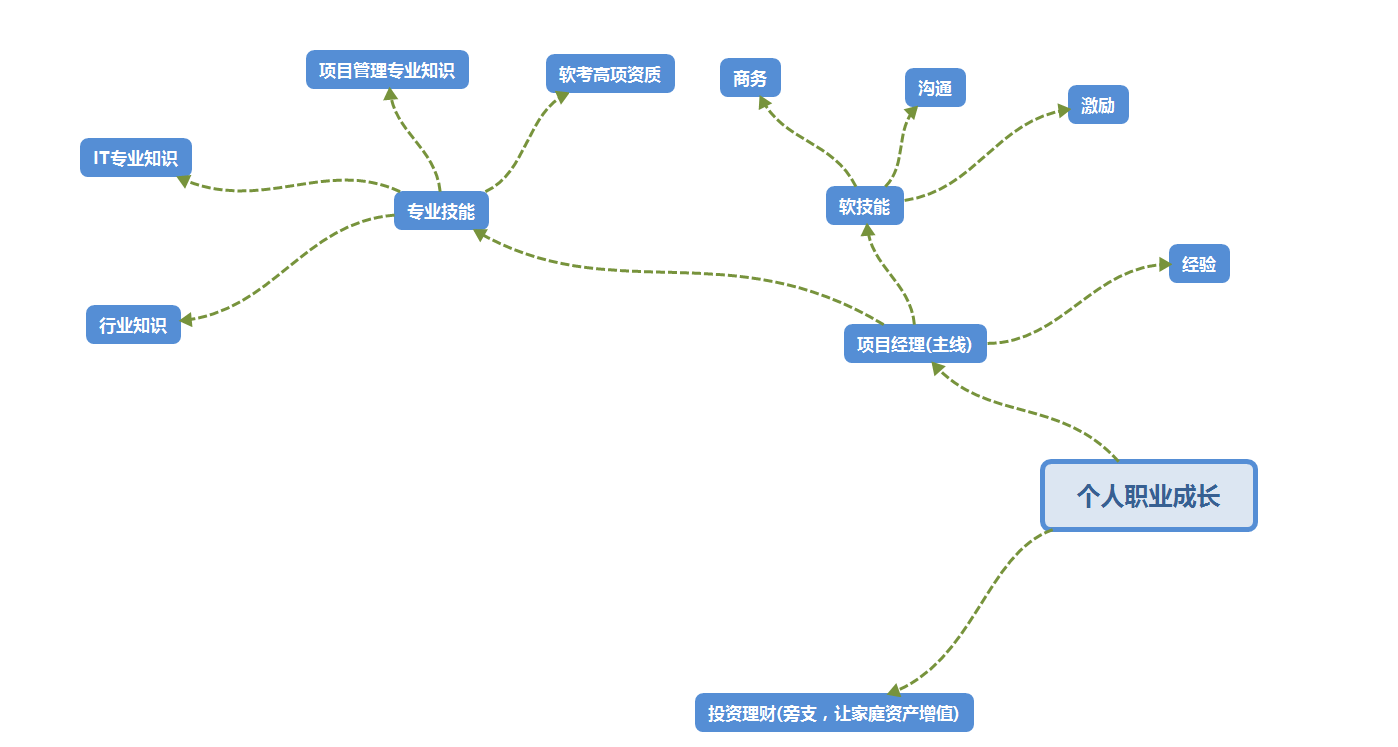
成功源自克服内心的成长障碍。

会议

每天早上 10 点 30 会开晨会，汇报昨天做了什么工作，遇到了哪些问题，怎么解决的，目前还遗留什么问题，今天的工作任务是什么，每周部门都会开例会，还会有一场技术分享，让大家多接触点，多学点，扩展一下自己的视野。

# 二、人生和职业管理

专业技能



G:\07-职业成长（含职称）\个人职业成长.xmind

程序员成长树

初级程序员：做一些静态的界面；

程序员：做一些增删改查的小模块；

中级程序员：做逻辑较复杂的模块；

高级程序员：做核心模块；

项目经理：系统的整体架构；

部门经理：多项目的管理；

总裁：多部门以及企业的发展规划。

理想的情况是一专，多能。必须有一样精通的，但是要知道，所谓的高手决不是只会一项，到了他们的级别，什么都一样，看的是学习能力和推理能力。所以，先静下心来积累经验。

### 专业能力快速提升的渠道

想要让自己的专业能力获得快速提升，我们的建议是：

1．构建专业知识地图（知识树）

每一个领域、每个岗位都有自己的专业知识地图，有的可能已经存在，可以根据自己的情况采用，有的可能需要自行总结。

知识地图是动态的，需要不断对其中的知识进行更新，而非固定不变。

2．取得相关资质

几乎在每个领域，都有相关的资质，比如注册会计师、心理咨询师、国家理财规划师、特一级厨师、英语六级、潜水证等，通过学习、考试获得这些资质，本身既是能力的证明，同时也会在学习的过程中获得专业能力提升。

然而也并不是取得相关资质就够了，那也只是长征中的一个节点。比如，很多人取得了心理咨询的证书，然而并没有面对面心理咨询的实际经验，这种情况下也是难以被信服的。

3．不断积累案例

不管是哪个专业领域，都需要多和客户接触，多做项目，尤其是不同的项目，从中总结共性和个性，不断增加自己在这个领域内的经验值。有些人做项目的时候并不用心，每次都套用一样的模板，试图用一种方法解决所有问题，这样是没办法积累起丰富经验的。认真地对待每一个项目，认真地总结每个项目的得失，才会获得快速有效的成长。

4．达到一定年限

比如在行就规定，法律专业，想要成为在行行家，至少需要有五年的法律相关工作经验。这种年限，其实也是专业能力的证明。当然年限只是一种参考，如果是混日子的年限法，其实含金量也不高。

5．尝试撰写专业书籍

不见得要出版，但是可以依据自己对专业的理解，尝试撰写专业书籍，这是系统整理自己知识脉络的有效方法，也是以终为始的学习方法。可以把写书设为目标，为了写成一本专业书籍，需要进行大量的阅读，对于工作本身的观察和分析角度，包括研究深度，都会不一样，而且，也许真的有出版机会呢？

秋叶的第一本书《超越对手：软件项目经理的18种实用技能》其实就是这么总结出来的。

6．通过社交网络建设个人品牌

不管是通过赤兔进行分享，还是开设微信公众号发布文章，一个人的专业能力往往需要多和他人互动才能够获得更多反馈，个人品牌的建设过程和写书是一样的道理，既能够在过程中得到锻炼，也能在拥有个人品牌之后带来更多的机会。

未来学家丹尼尔·平克在《全新思维》一书里讲过，未来最重要的六种技能是：

设计感：每个人都必须培养一种艺术感，好的设计可以改变世界，不管从事什么样的工作，都可以像艺术家一样工作。

故事感：会讲故事，用故事的形式表达自己、与人沟通，故事性会越来越成为商品和服务在拥挤的市场中脱颖而出的一件法宝。如果你对故事感的理解还不够，可以去了解一下的故事，他从故事里看到商机，自称“首席故事官”。

交响能力：可以理解为整合事物的能力、跨界能力，善于找出看上去毫不相干的领域之间的联系，善于组合别人所不在意的要素，实现创新。

共情能力：可以理解为同理心，能够设身处地体验他人处境，从而达到感受和理解他人情感的能力。

娱乐感：简单说来就是会玩，人需要具备适当的幽默感（可千万别误解为会说黄段子！），不仅会玩，更要玩出水平。

探寻意义：能够寻找生存的意义，认真对待精神追求，寻找真正的幸福。

上面的这段话，用大白话来讲，就是：要有品位、懂艺术，要会讲故事，能跨界，有同理心，会玩出花样来，而且要有追求，会探寻事物的意义。

时间的重要性

你的时间值不值钱，取决于你用你的时间去做什么，你用你的时间去抢券，当然是不值钱的，但如果你用你的时间去学习，去提升自己，这时间就值钱了。

过来人的经验告诉你们，你与同龄人之间，之所以差距越来越大，就在于对时间的利用上。

过来人的经验再告诉你们，买东西不要冲着便宜、优惠而买，要冲着自己是否需要而买，不需要的东西，再便宜买了也无用，只会徒增累赘，自己需要的东西，应该果断立刻就买。如果需要的东西暂时没钱买，那就应该努力挣钱才对，怎么努力挣钱？用你最宝贵的时间，去提升自己，当自己有能力了，挣钱是自然而然的事。

穷人思维总想着花时间省钱，但是当你接触过一些成功人士之后，他们的富人思维永远是花钱省时间。当然，我们很多人现阶段还没到富人随心所欲花钱省时间的地步，但是起码在自己力所能及的范围内，花点钱省时间的行为我是很鼓励的，比如科学上网，比如去健身房，比如买书、买课程学习，实在经济暂时不允许你花钱，至少不要无端的浪费自己的时间吧。

梦想金字塔

写出10年、20年后自己希望成为的样子。

临界知识

知识与知识是不一样的，有些知识比其他知识的威力更大。少数的知识能够给我们带来关键的影响，这就是临界知识。

认知优势的阶段：

1）知识数量构建认知优势；

2）知识获取速度构建认知优势；

3）知识深度构建认知优势.

现阶段已经是知识深度构建认知优势的时代了。

**所谓临界知识，便是我们经过深度思考后发现的具有普遍指导意义的规律或定律。**

如果我们的学习是在了解问题本质，了解解决方案的底层规律，能够让我们认清楚问题背后的实质，那我们就是在提升认知效率。

哪些知识值得学？一是执行能力，比如时间管理、商务礼仪、沟通谈判等；二是专业能力，是指能够打完整战役、解决系统问题的能力；三是结构能力，就是掌握临界知识，认知事物更加底层的结构与规律。

读书一定要花时间、耐心和思考力，将获得的新知识和已有的知识进行网络状的联系。

学习临界知识需要具备的两个心态：

1、绿灯思维

想要通过接触新观点和知识快速成长，就要面对与自己传统认知不一样的地方。而在这种情况下，我们很容易激起习惯性防卫。要改变这一点，就要培养一个底层心态——绿灯思维，积极地考虑新观点里有价值的地方。

2、以慢为快

越是快速提升的能力，反而越需要下慢工夫。真正的快，是以慢为快。换句话说就是：结硬寨，打呆仗。

如何在生活中提升我们的学习能力呢？常用方法有三个：反思、以教为学、刻意练习。

1)反思：提升知识掌握的层次

反思不是总结，至少两者的侧重点不同。总结是对结果的好坏进行分析；而反思是对产生结果的原因进行分析，或者说是对假设进行校正。

作者提到了他自己曾经用过的反思方法，每天早上写精进日志：昨天我做得不好的是什么？我当时是怎么考虑的？如果我重新来做会有哪些改进？坚持每天按照这个标准，来对照我的行为！

作者列举了将心理学的”损失规避效应“应用到业务中的事例，在一项业务中让客户加10000元包打印费，客户感觉自己是在损失，而不愿意这样做。同时联想到心理学的”损失规避效应“，人们更面对损失的痛苦感大大超过面对获得时的快感，于是改为在原有合同价的基础上加10000元，如果客户自己打印，价格就优惠10000元，达到了很好的效果。同时，我们也可以想到生活中很多这样的安全都可以用到这个知识，比如加200元上门安装，改为自取优惠200元，这样是不是会有更好的效果呢？

还有一种反思方方法，对标管理：提前设定一个期望的标准，然后每天反思，与之比较寻找差距。

2) 以教为学

以教为学，也就是把教别人的过程作为自己学习的过程。就好像是一个人是老师，同时又是学生。

3) 刻意练习

刻意练习首先要求我们要走出舒适区，时刻保持在学习区。

通过3F，即Focus专注、Feedback反馈、Fit纠正，从而形成心理表征。

前面谈了提升学习能力的心态和方法，下面是具体执行中的三个技巧：记录、定期回顾、付费购买。

记录的过程其实是联想、启发、归纳、演绎的过程，是调用自己所有的知识去重新理解过去一天发生的事情。而这种通过记录、调用知识来解构和重构问题的过程，才是记录最有价值的部分，也是记录帮助学习快速提升的关键环节。

前面我们讲了记录的重要性以及方法技巧，但是更重要的是定期回顾这些记录。

花钱买学习时间。比如为自己买更好的学习工具，实际上在为自己买时间；选择更好的出行方式，来为自己节省出学习时间；花钱买一些可节省自己时间的服务，也是在为自己的学习买时间。

作者举了万维钢《精英日课》里面的例子，一篇文章《权威的合法性从哪里来？》，文章给定了一个场景：孩子不听管教怎么办？学生不听老师的话怎么办？下级违抗上级怎么办？也就是原本应该拥有权威的场合，却失去了权威，应该怎么办？

万维钢给出了三个方法：1、你管理的每一个下属，他的声音都必须被你听到；2、你制定的规则必须稳定，有预测性，不能朝令夕改；3、执法必须公平，一视同仁。

李叫兽有一篇很火的文章，《为什么你有10年经验，但却成不了专家？》，里面提到了核心关键在于是否进行了刻意练习。

那么如何刻意练习来撑握临界知识，主要有两点：

1、抓住问题的本质进行练习

只有抓住问题的本质进行练习，才能在相同的时间里比别人成长得更快。

2、大量地持续练习

小马宋曾经在知乎上有一个问题，我知道一个快速提升方案写作能力的方法。设置价格10元，无数人购买。但答案却出人意料：阅读超一万份经典文案。

我们总以为成功人士背后都有一条捷径，殊不知，外表光鲜的背后都是大量持续练习的结果。

学习知识的终极目的无非三个：解释问题、解决问题、预测问题。

我们能不能获得预见性认知呢？肯定是可以的，如果你能够了解一件事情的基本发展规律，你就能借此做出一些对未来的判断。你掌握的结构越多，对未来的了解就越多。

用临界知识构建自己的“能力圈”：能力圈是由你真正擅长并懂得的知识组成的，而且在这些领域里，你要比90%的人都做得好。只要能找到一项能力，能够超过周围90%的人，这就可以作为你的能力圈。

个人思考：对于软件工程师，能力圈并不是编程语言，而是对系统的整体思考，架构、性能等宏观性的知识……

画出自己的技能树

为了达到金字塔中的目标还缺少什么能力和资源？

除了梦想以外，平时生活中有什么兴趣、爱好或想要学习的东西？

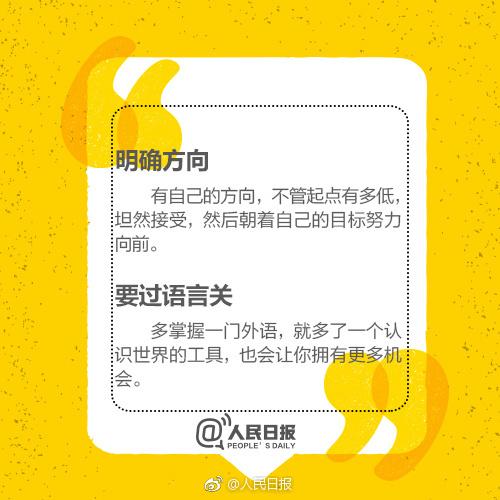
1. 股票和理财

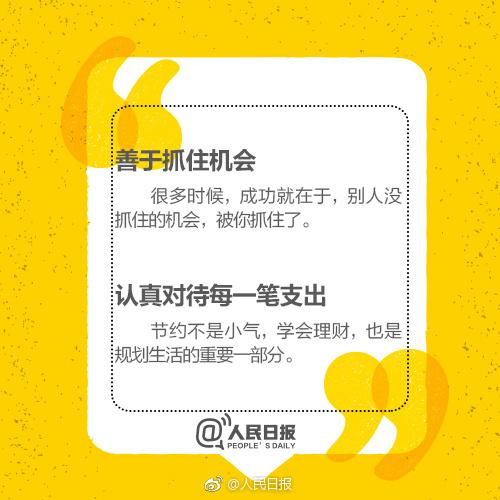
将以上技能列举出来，画成一颗技能树。

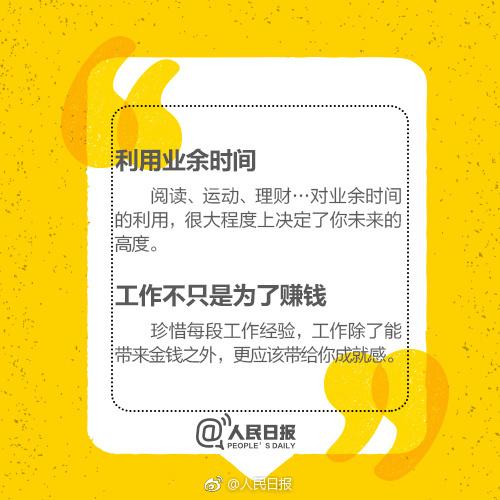
一步步延伸和分解具体可执行的步骤，并制定详尽的计划，定出年、月、周计划。

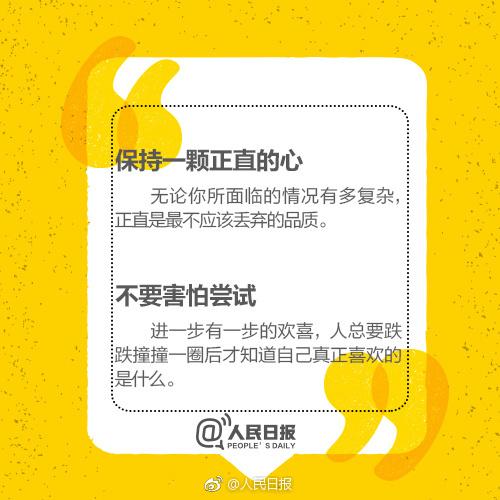
## 珍惜你的业余时间，那是你弯道超车的资本

二十几岁的你该明白，没有什么捷径能让你脱颖而出。一个人有怎样的人生，很大程度是自己决定的。自己不努力改变，没人能替你成长。年轻时追求安逸，白白耗费时光，是对自己最大的残忍。世界很小，请带着梦想一起奔跑；世界很大，请带着坚持努力成长。

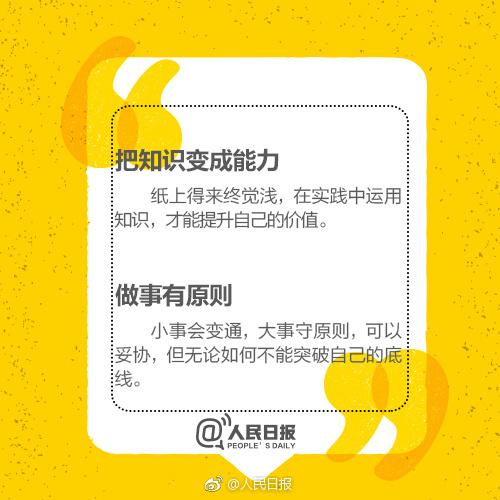


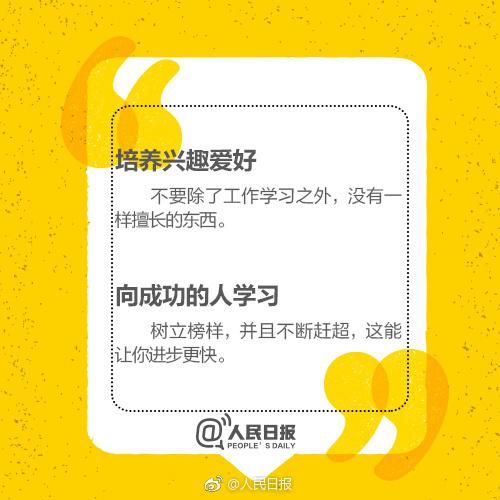


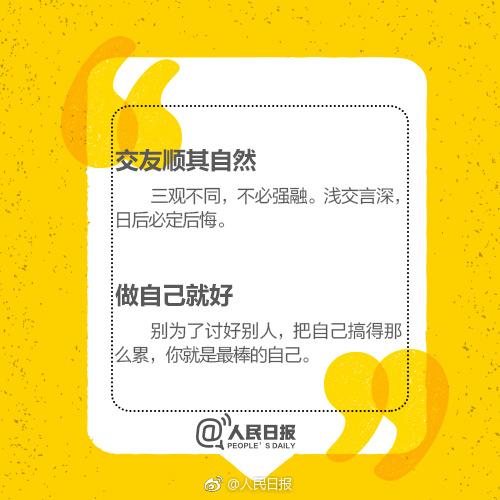


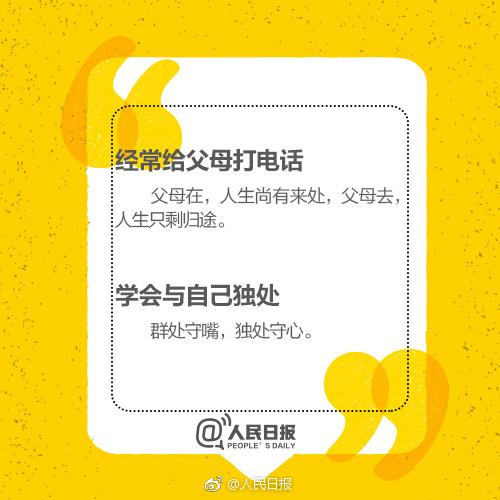












推荐的书籍

《番茄工作法》

推荐的书籍

《番茄工作法》

《高效能人士的七个习惯》

《getting things done》

《奇特的一生》

《晨间日记的奇迹》

《记事本圆梦计划》

《拖延心理学》

《把时间当朋友》

《好好学习——个人知识管理精进指南》

# 三、管理工具

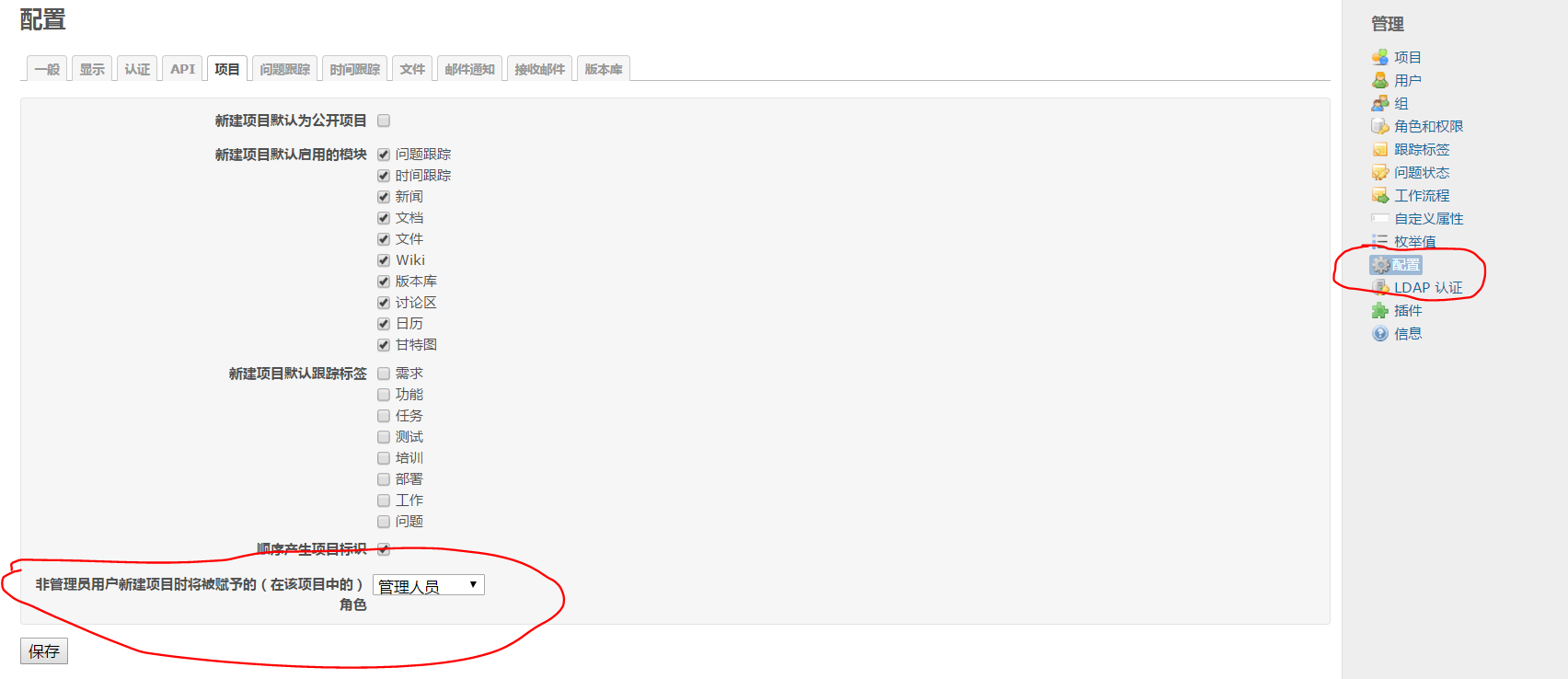
Remine

安装

bitnami-redmine-3.4.4-1-windows-installer.exe，一键化安装，apache和mysql检测到有会让填写其他的端口号,mysql如果继续选择3306，会停掉原来的服务，重新安装，但库不会删。

非管理员如何新建项目

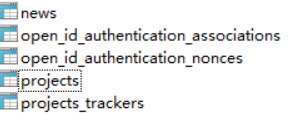
默认管理员才能新建项目，要将非管理员作为管理人员加入到某个项目中。

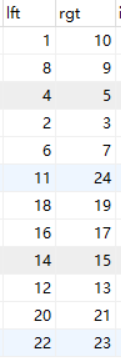


项目排序

项目录入后排序很乱，操作如下：

找到projects表，





这两个字段原有值范围不变，只是重新按顺序排一下。

# 四、代码规范

命名风格

1. 【强制】代码中的命名均不能以下划线或美元符号开始，也不能以下划线或美元符号结束。 反例：\_name/\_\_name/$Object/name\_/name$/Object$

2. 【强制】代码中的命名严禁使用拼音与英文混合的方式，更不允许直接使用中文的方式。 说明：正确的英文拼写和语法可以让阅读者易于理解，避免歧义。注意，即使纯拼音命名方式也要避免采用。

正例：alibaba / taobao / youku / hangzhou 等国际通用的名称，可视同英文。 反例：DaZhePromotion [打折] / getPingfenByName() [评分] / int 某变量 = 3

3. 【强制】类名使用UpperCamelCase风格，必须遵从驼峰形式，但以下情形例外：DO / BO / DTO / VO / AO 正例：MarcoPolo / UserDO / XmlService / TcpUdpDeal / TaPromotion 反例：macroPolo / UserDo / XMLService / TCPUDPDeal / TAPromotion

4. 【强制】方法名、参数名、成员变量、局部变量都统一使用lowerCamelCase风格，必须遵从驼峰形式。 正例： localValue / getHttpMessage() / inputUserId

5. 【强制】常量命名全部大写，单词间用下划线隔开，力求语义表达完整清楚，不要嫌名字长。 正例：MAX\_STOCK\_COUNT 反例：MAX\_COUNT

6. 【强制】抽象类命名使用Abstract或Base开头；异常类命名使用Exception结尾；测试类命名以它要测试的类的名称开始，以Test结尾。

7. 【强制】中括号是数组类型的一部分，数组定义如下：String[] args; 反例：使用String args[]的方式来定义。

8. 【强制】POJO类中布尔类型的变量，都不要加is，否则部分框架解析会引起序列化错误。 反例：定义为基本数据类型Boolean isDeleted；的属性，它的方法也是isDeleted()，RPC 框架在反向解析的时候，“以为”对应的属性名称是deleted，导致属性获取不到，进而抛出异常。

9. 【强制】包名统一使用小写，点分隔符之间有且仅有一个自然语义的英语单词。包名统一使用单数形式，但是类名如果有复数含义，类名可以使用复数形式。 正例： 应用工具类包名为com.alibaba.open.util、类名为MessageUtils（此规则参考spring的框架结构）

10. 【强制】杜绝完全不规范的缩写，避免望文不知义。 反例：AbstractClass“缩写”命名成AbsClass；condition“缩写”命名成 condi，此类随意缩写严重降低了代码的可阅读性。

# 五、管理技能

管理就是做好一系列决策

知识点一：没有不需要管理的员工

刘芹先生说，小公司最好没有管理，并不是一点都不管。而是先把事情想明白，把规则定清楚，然后找到有驱动力的员工，让他们去施展发挥。

大部分管理者都是小白，没有丰富的管理经验，经常幻想早点当上甩手掌柜。如果不认真对待管理，就会产生很多不必要的问题。

知识点二：告诉下属什么是最重要的

所谓管理，就是要做好一系列决策。决策的核心，就是让下属明白，日常工作中，什么是最重要的。

不要考验员工的悟性，用悟性来评价员工能力，是非常糟糕的一件事。

知识点三：区分组织决策和管理决策

只讲到这还不够，我们还要更进一步地理解决策。因为，在实际管理中，决策又分成两种主要的类型：组织决策和管理决策。

所谓组织决策，主要针对一个团队、一家公司、一个组织。比如： 2019 年收入目标；2019 年营销预算；未来 3 年的核心业务方向；

所谓管理决策，主要针对员工的具体工作。

只有区分了组织决策和管理决策，管理者才能真正有效地开展工作。而经验少的管理者，经常用组织决策代替管理决策，这是错误的。

在实际管理中，我们要多做管理决策，少讲组织决策。

知识点四：没有完美的决策

只要没有太大问题，就早点定下来，一边干一边优化。作为管理者，我们一定要能定事儿，定错了也不要紧，扛下来就好。只要没有太大问题，就早点定下来，一边干一边优化。作为管理者，我们一定要能定事儿，定错了也不要紧，扛下来就好。

知识点五：定期和下属复盘

大部分同学，都能理解了决策的重要性。尤其是决策的核心，就是告诉下属，什么是最重要的。只不过，仅仅告诉下属，还不够。经验丰富的管理者，在实际工作中，都养成了一项好习惯：定期和下属复盘，你目前负责的工作中，最重要的 3 件事是什么？

管理，就是面对事实解决问题

经验不丰富的管理者，在做决策的时候，经常因为考虑得太周全，导致迟迟做不出决策，影响了工作进度。

刚做管理者时，普遍都有一个误区：怕出错。这种心理越强烈，越容易让我们忽略：管理工作的本质，就是持续争取资源、优化动作，直到达成目标。

知识点一：不要证明自己的对错

他特别想和我证明，他的方案是对的，他的想法是对的。却忽略了更重要的一点：无论他做得对还是错，我和他的目标都是一致的。何况，对错并不关键，能让整个项目顺利推进，让这次活动顺利举办，才是最重要的。作为他的领导，我向他了解项目进展，并不是为了判断他做得对不对，而是帮他判断一下项目的进展是否合理，是否需要给他提供一些支持。

知识点二：不要评价上司的对错

你的职责，都是帮助老板达成他的目标。他的方案怎么样并不重要，他的目标是什么最重要。

知识点三：定量比定性更重要

管理是要对绩效负责的，比对错更重要的是行动，能验证对错的也是行动。

你发现一个方案有些问题，然后你说：这个方案不合理。这就叫定性。为什么定性思维要不得？很简单，定性会让行动暂停，对事情的推进没有帮助。

那什么叫定量思维？就是你发现一个方案不合理，然后你说，这个方案在某某方面不合理，我们可以把 A 改成 B，或者再增加一点 C，这样就会更好。

这就叫定量思维。有这种思维方式的人，会非常注重行动，结果也会更有效。

知识点四：分析原因时多谈差距

有经验的管理者，在分析原因时，更重视直接说明差距。比如，第一季度的真实工作量不够，那就干脆弄清楚，和正常的一个季度相比，到底差多少天？比如，第一季度招聘的新员工比较多，到底是多少？培训做得不到位，到底做了几次培训，应该做几次培训？下个季度怎么办，才能保证把这几次培训补上去？等等。当我们更重视差距的时候，我们也会获得更清晰的行动计划，尤其会更清楚，接下来要投入哪些资源。我们的管理工作，也自然更有效了。

# 六、产品

# 关于青春饭

在国内，没有哪一行是不吃青春饭的。

首先青春就代表有无限可能，在年轻的时候做点事情，无论结果如何，自己总不会后悔。

等青春饭吃完了，接下来拼的就是年轻时候的积累有多沉，人脉有多广，思考有多深。

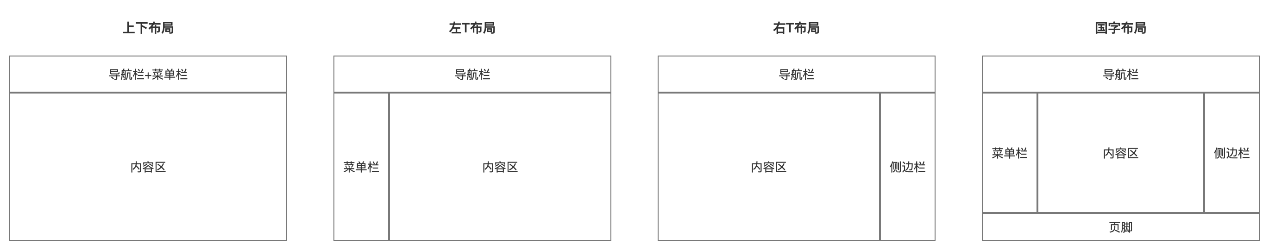
中年饭，吃的是半辈子的阅历煮成的饭，只不过有些人成色不好，难以下咽；有些人配菜齐全，有滋有味。

# Web产品的原型布局需要遵循什么规范

原型布局是指按照合适的位置排放页面各种元素，PM需要根据产品的特性采用合适的布局设计；

同一个产品不同页面的布局规则应该尽量保持统一，即大部分页面都是采用一样的布局，特殊页面可以单独处理。





一、上下布局

上下布局包含导航栏、内容区。

导航栏位于画布位置(0,0)，宽度建议采用1200px，高度建议80px。

内容区位于画布位置(0,100)，宽度1200px，高度自适应。



二、左T布局

左T布局包含导航栏、菜单栏、内容区。

导航栏位于画布位置(0,0)，宽度建议采用1200px，高度建议80px。

菜单栏位于画布位置(0,100)，大小取决于每个菜单项的大小160\*40px。

内容区位于画布位置(180,100)，宽度1020px，高度自适应。



三、右T布局

左T布局包含导航栏、内容区、侧边栏。

导航栏位于画布位置(0,0)，宽度建议采用1200px，高度建议80px。

内容区位于画布位置(0,100)，宽度1020px，高度自适应。

侧边栏位于画布位置(970,100)，大小取决于每个菜单项的大小160\*40px。



四、国字布局

国字布局包含导航栏、菜单栏、内容区、侧边栏、页脚。

导航栏位于画布位置(0,0)，宽度建议采用1200px，高度建议80px。

菜单栏位于画布位置(0,100)，大小取决于每个菜单项的大小160\*40px。

内容区位于画布位置(180,100)，宽度1020px，高度自适应。

侧边栏位于画布位置(1010,100)，大小取决于每个菜单项的大小160\*40px。

页脚位于页面位置(0,640)，高度位置根据实际情况调整。



Web后台产品，建议遵循左T布局来画原型，可以很好的展示产品的菜单导航，方便运营快速找到自己所需的功能进行使用。

Web用户产品，建议遵循国字布局来画原型，可以很好的展示产品的各种内容，用户可以方便的查看网站的全部内容和结构。

01 列表项规范

列表项包含字段名和对应的字段值。我们需要了解他们的常用规范。

1. 列表项数量

建议列表项的数量不要太多，这样一屏很难展示所有的列表项或者看不全每个列表项的值。

# Web后台产品的列表页规范

PM需要学会思考少即是多，而不是粗暴的展示所有数据给到用户。

Web后台产品的列表页通常由列表+分页+ 搜索组成，接下来分别介绍他们的逻辑规范和应用场景。

01 列表项规范

建议列表项的数量不要太多，这样一屏很难展示所有的列表项或者看不全每个列表项的值。PM需要学会思考少即是多，而不是粗暴的展示所有数据给到用户。

列表项宽度建议使用%表示，从而保证一屏幕可以看到所有列表项。如果列表项宽度使用px表示，则无法100%保证不会超过一屏幕，即需要左右滚动。原因是列表项1宽度+列表项2宽度+…往往大于屏幕宽度，而且即使减少宽度也无法同时满足不同用户的屏幕分辨率。

如果列表项对应的字段值会出现很多的字符串，超出宽度的部分则会自动换行。如果不想换行，则可以约束最多显示前N字(后面的用省略号…表示，悬浮的时候用toast显示完整的信息)

02 对齐规范

1. 列表项对齐

列表项默认靠左对齐，根据需要也可以居中对齐。如果列表项很少并且字段值的字符数也不多，可以居中对齐。如果部分列表项的字段值的字符数有多有少，此时居中对齐会比较丑。

2. 标题项对齐

列表项的字段名和字段值，需要保证对齐方式一致。不要出现字段名靠左对齐而字段值居中对齐这样的奇葩效果。

03 分页规范

1. 每页条数

每页默认显示N条数据，超过则分页显示，默认展示第一页。N通常为10、15、20。列表底部最好显示每页显示N条，方便用户理解。

2. 显示总条数

显示当前列表的总条数，当搜索列表的时候则显示对应结果的总条数。

3. 显示总页码

显示当前列表的总页码，总页码=总条数/每月条数的结果取整+1，如无余数则不+1。

4. 跳转到第N页

通常有4种实现方式①点击对应页码去跳转②点击上一页下一页去跳转③点击首页尾页去跳转④输入页码直接跳转。

04 搜索规范

搜索后进入详情页，然后返回列表依然保留原先筛选条件的搜索结果列表。比如我搜索指定余额区间的用户，然后我查看他们的详情，然后返回到该页面的时候依然是这些搜索结果。而不需要重新搜索。

# 可视化硬核指南

涵盖了**设计原则、图表分类、图表的选用、样式设计、交互设计、仪表板设计**等方面。有人对其进行了翻译，希望能够分享给更多对数据可视化有兴趣的同学！

**全文章节目录：**

**原则**

**类型**

**选择图表**

**样式**

**行为**

**仪表板**

**数据可视化**

数据可视化就是用图形描绘信息。

**原则**

数据可视化是一种以图形描绘密集和复杂信息的表现形式。数据可视化的视觉效果旨在使数据容易对比，并用它来讲故事，以此来帮助用户做出决策。

数据可视化可以表达不同类型和规模的数据，包括从几个数据点到有大量变量的数据集。

**类型**

数据可视化可以以不同的形式表达。图表是表达数据的常用方式，因为它们能够展示和对比多种不同的数据。

图表类型的选择主要取决于两点：要表现的数据和表现该数据的用意。该指南描述各种类型的图表及其用例。

**图表类型**

**1. 随时间变化**

随时间变化的图表显示一段时间的数据，例如多个类别之间的趋势或比较。

**常见用例包括：**

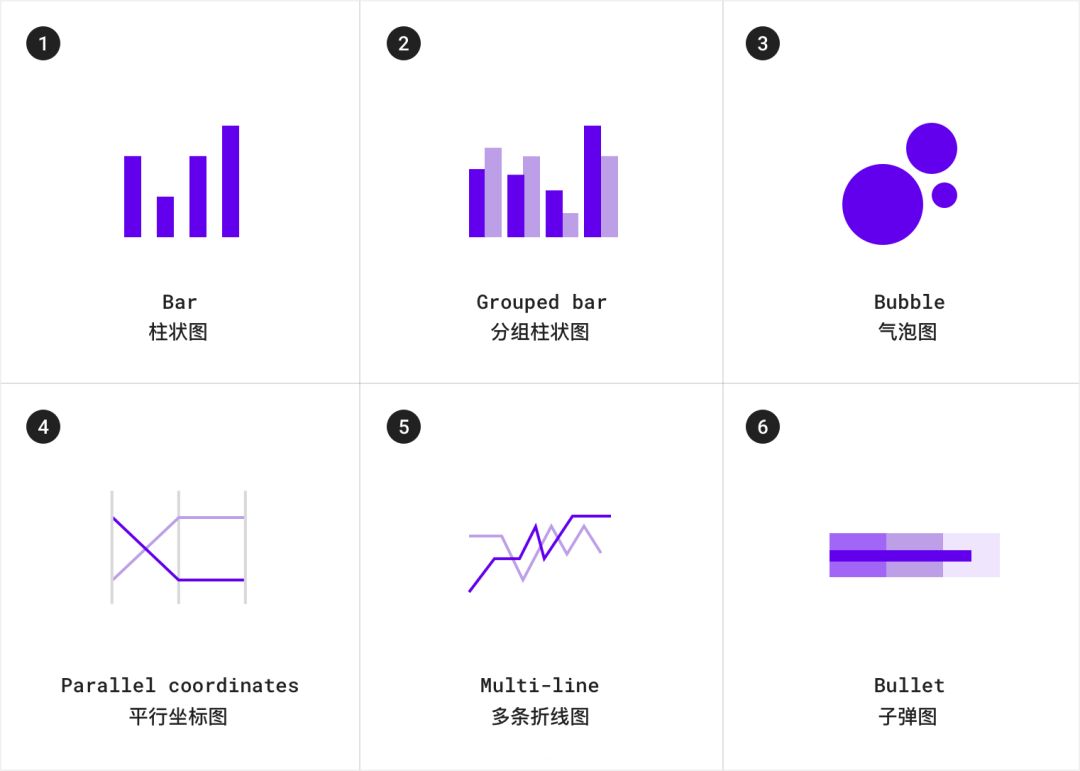
股价表现、卫生统计、年表

**2. 类别比较**

类别比较图表是多个不同类别数据之间的比较。

**常见用例包括：**

不同国家的收入、热门场地时间、团队分配



**3. 排名**

排名图表显示项目在有序列表中的位置。

**常见用例包括：**

选举结果、性能统计

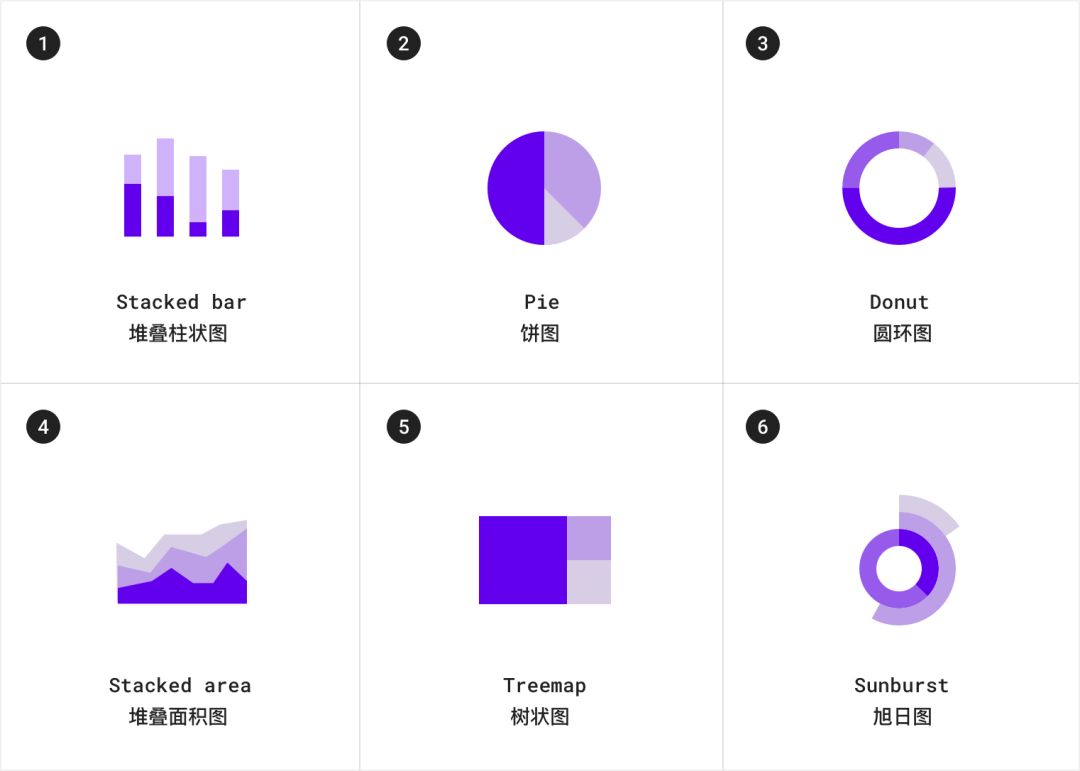


**4. 占比**

占比类图表显示了局部与整体的关系。

**常见用例包括：**

产品类别的综合收入、预算



**5. 关联**

关联类图表显示两个或以上变量之间的关系。

**常见用例包括：**

收入和预期寿命

**6. 分布**

分布类图表显示每个值在数据集中出现的频率。

**常见用例包括：**

人口分布、收入分布

**7. 流程**

流程类图表显示了多个状态之间的数据移动。

**常见用例包括：**

资金转移、投票计数和选举结果



**8. 关系**

关系图表显示多个项目之间的关系。

**常见用例包括：**

社交网络、词图

**选择图表**

面对多种类型的图表，以下指南提供了关于如何选择合适的图表见解。

**显示随时间的变化**

可以使用时间序列图表来表示随时间的变化，就是按时间顺序表示数据点的图表。表示随时间变化的图表包括：折线图，柱状图（条形图）和面积图。

\*基线值是y轴上的起始值。

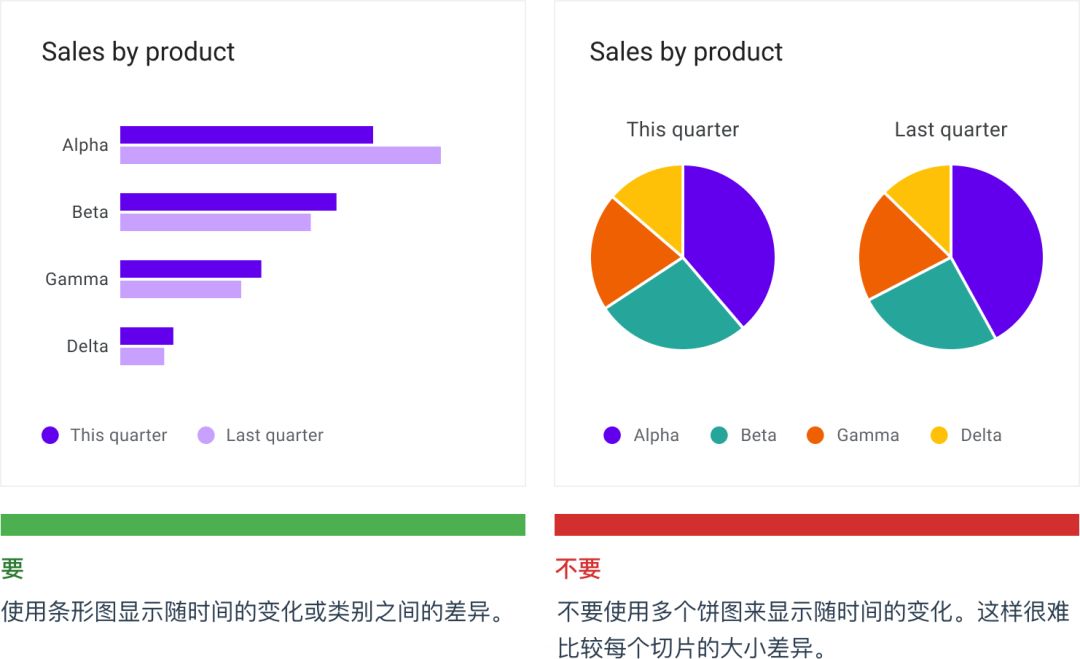
**柱状图（条形图）和饼图**

柱状图（条形图）和饼图都可用于显示比例，表示部分与总体的对比。

**·**柱状图（条形图）使用共同的基线，通过条形长度表示数量

**·**饼图使用圆的圆弧或角度表示整体的一部分

柱状图（条形图），折线图和堆叠面积图在显示随时间的变化方面比饼图更有效地。由于这三个图表都是使用相同的基线，因此可以更轻松地根据条形长度比较值的差异。



**面积图**

面积图有多种类型，包括堆叠面积图和层叠面积图：

**·**堆叠面积图显示多个时间序列（在同一时间段内）堆叠在一起

**·**层叠面积图显示多个时间序列（在同一时间段内）重叠在一起

层叠面积图建议不要使用超过两个时间序列，因为这样做会使数据模糊不清。取而代之，应当使用堆叠面积图来比较一个时间间隔内的多个值（横轴表示时间）。

**样式**

数据可视化使用自定义样式和形状，使数据更容易理解，以适合用户需求。

**图表可以从以下方面进行优化：**

**·**图形元素

**·**文字排版

**·**图标

**·**轴和标签

**·**图例和注释

**不同类型数据的样式设计**

**可视化编码**是将数据转换为可视形式的过程。独特的图形属性可应用于定量数据（如温度，价格或速度）和定性数据（如类别，风味或表达式）。

**这些图形属性包括：**

**·**形状

**·**颜色

**·**大小

**·**面积

**·**体积

**·**长度

**·**角度

**·**位置

**·**方向

**·**密度

**不同属性的表现**

多个视觉处理方法可以综合应用于数据点的多个方面。例如，在条形图中，条形颜色可以表示类别，而条形长度可以表示值（如人口数量）。



形状可用于表示定性数据。在此图表中，每个类别由特定形状（圆形，正方形和三角形）表示，这样可以在一张图表中轻松实现特定范围的比较，同时也可以进行类别之间比较。

**1. 形状**

图表可以运用形状，以多种方式展示数据。形状的设计可以是有趣的、曲线的，或者精确和高保真的等等。

**形状精确程度**

图表可以展示不同精度程度的数据。用于细致研究的数据应该用适合交互的形状（在触摸大小和功能可见性方面）展示。而旨在表达一般概念或趋势的数据可以使用细节较少的形状。

**2. 颜色**

颜色可用于以四种主要方式区分图表数据：

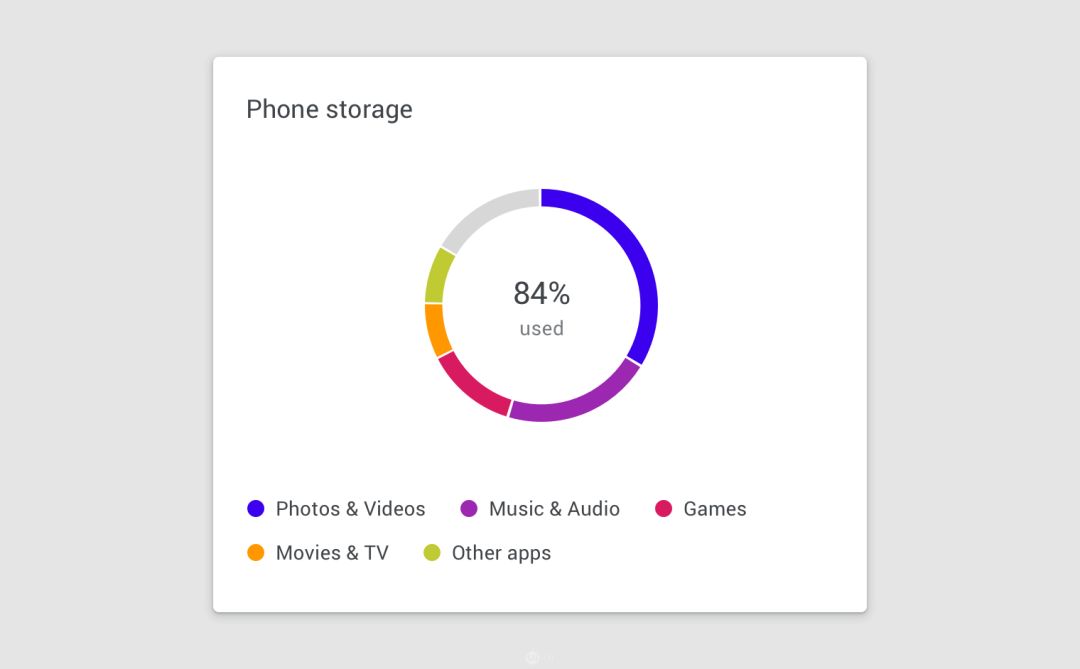
**·**区分类别

**·**表示数量

**·**突出特定数据

**·**表示含义

**颜色区分类别**



例：圆环图中，颜色用于表示类别。

**颜色表示数量**

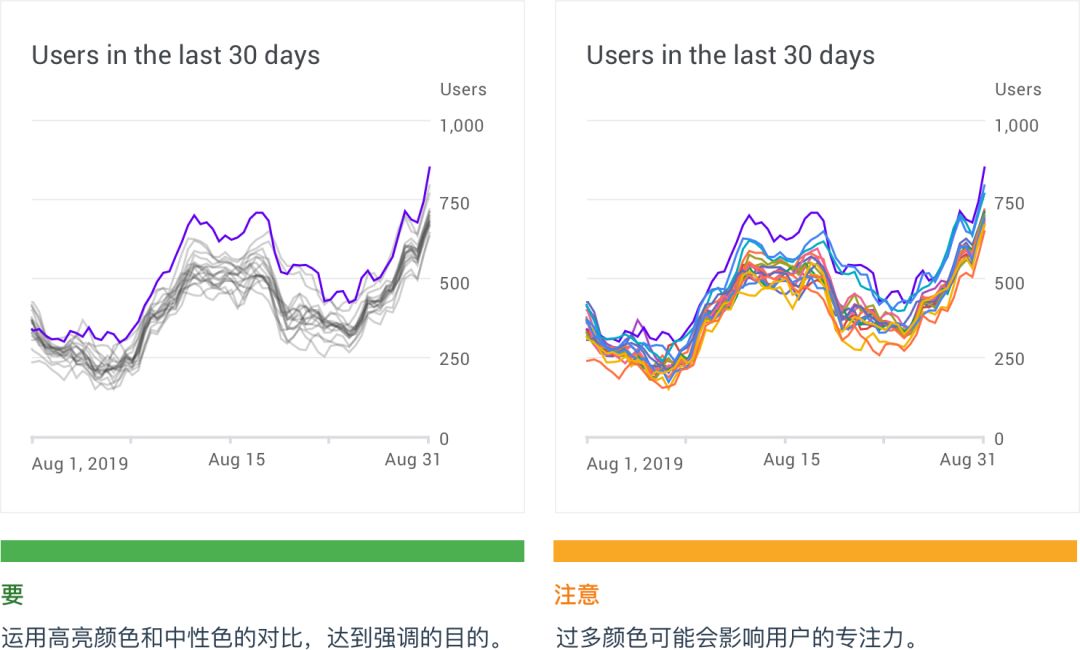
例：地图中，颜色用于表示数据值。

**颜色突出数据**

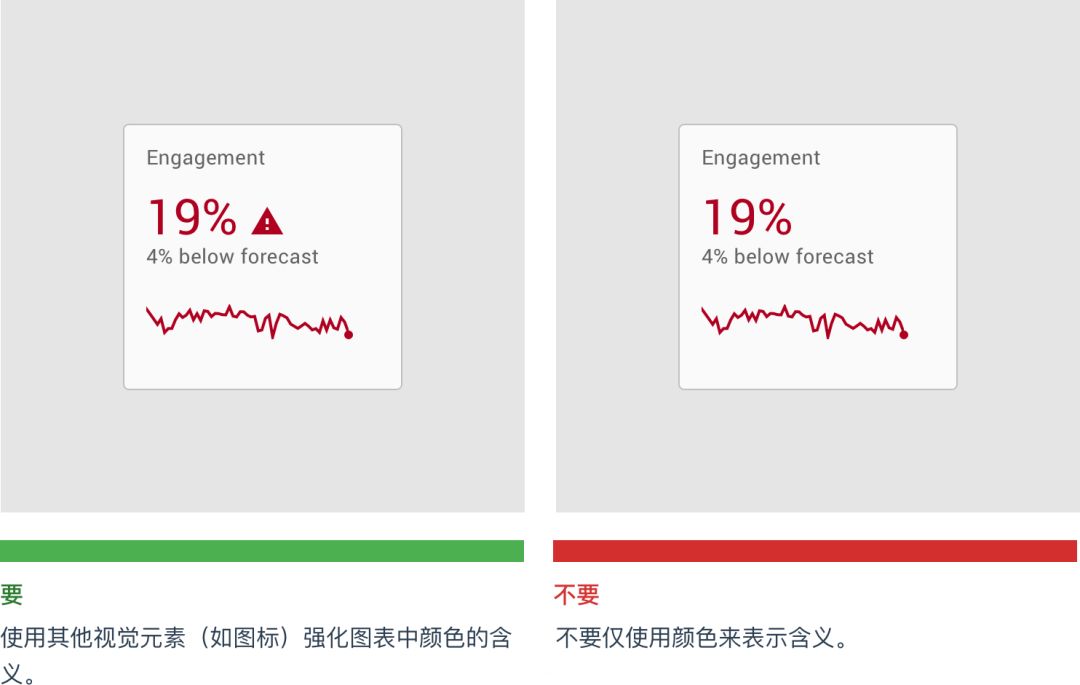
例：散点图中，颜色用于突出特定数据。

**重点区域**

在不滥用的情况下，颜色可以突出焦点区域。不建议大量使用高亮颜色，因为它们会分散用户注意力，影响用户的专注力。



**颜色表示含义**



**无障碍**

为了适应看不到颜色差异的用户，您可以使用其他方法来强调数据，例如高对比度着色，形状或纹理。

将文本标签应用于数据还有助于说明其含义，同时消除对图例的需求。

**3. 线**

图表中的线可以表示数据的特性，例如层次结构，突出和比较。线条可以有多种不同的样式，例如点划线或不同的不透明度。

**线可以应用于特定元素，包括：**

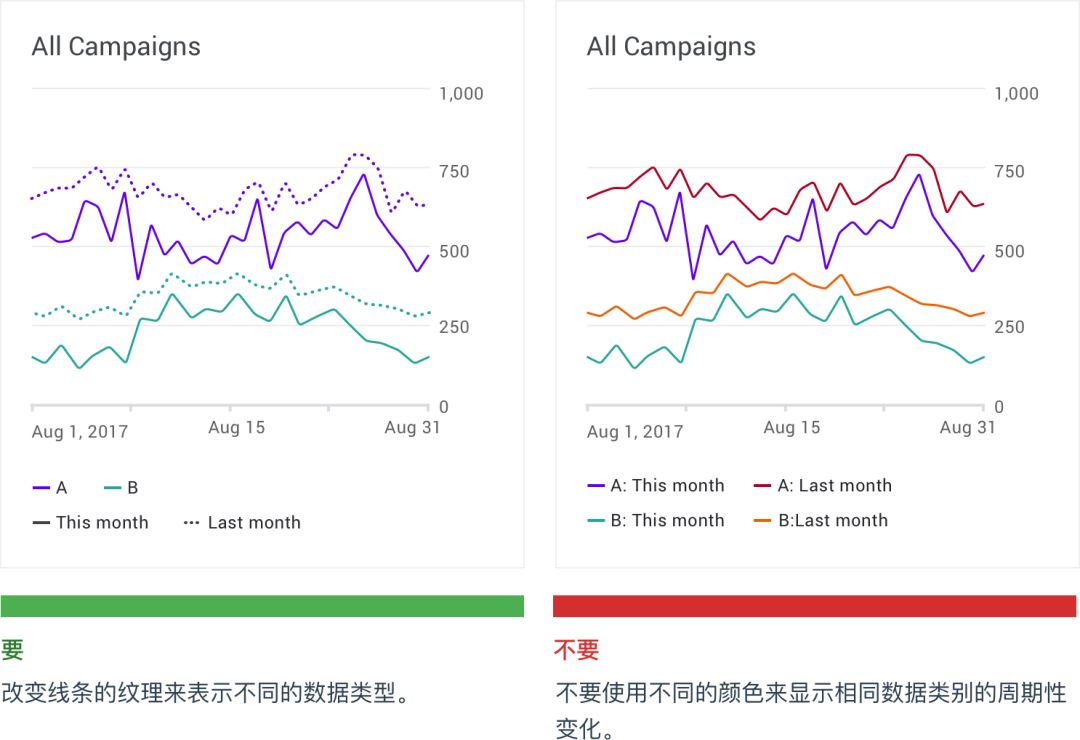
**·**注释

**·**预测元素

**·**比较工具

**·**可靠区间

**·**异常



**4. 文字排版**

文本可用于不同的图表元素，包括：

**·**图表标题

**·**数据标签

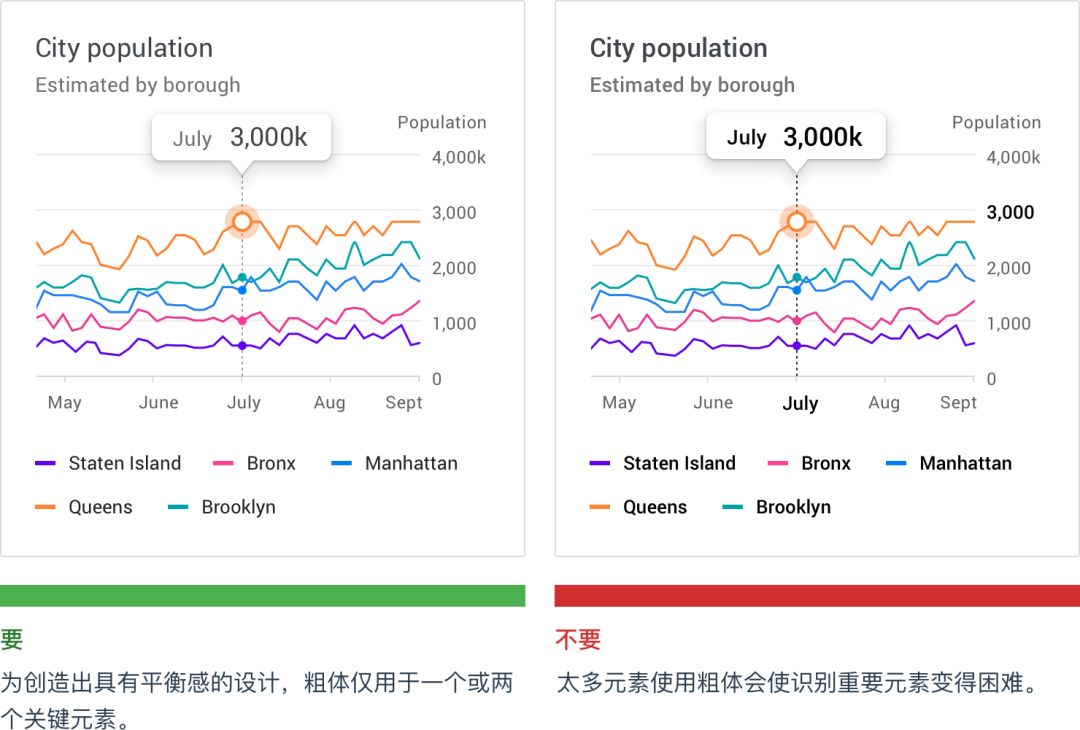
**·**轴标签

**·**图例

图表标题通常是具有最高层次结构的文本，轴标签和图例具有最低级别的层次结构。

**字重**

标题和字重的变化可以表达内容在层次结构中的重要程度。但是应该保持克制，使用有限的字体样式。



**5. 图标**

图标可以表示图表中不同类型的数据，并提高图表的整体可用性。

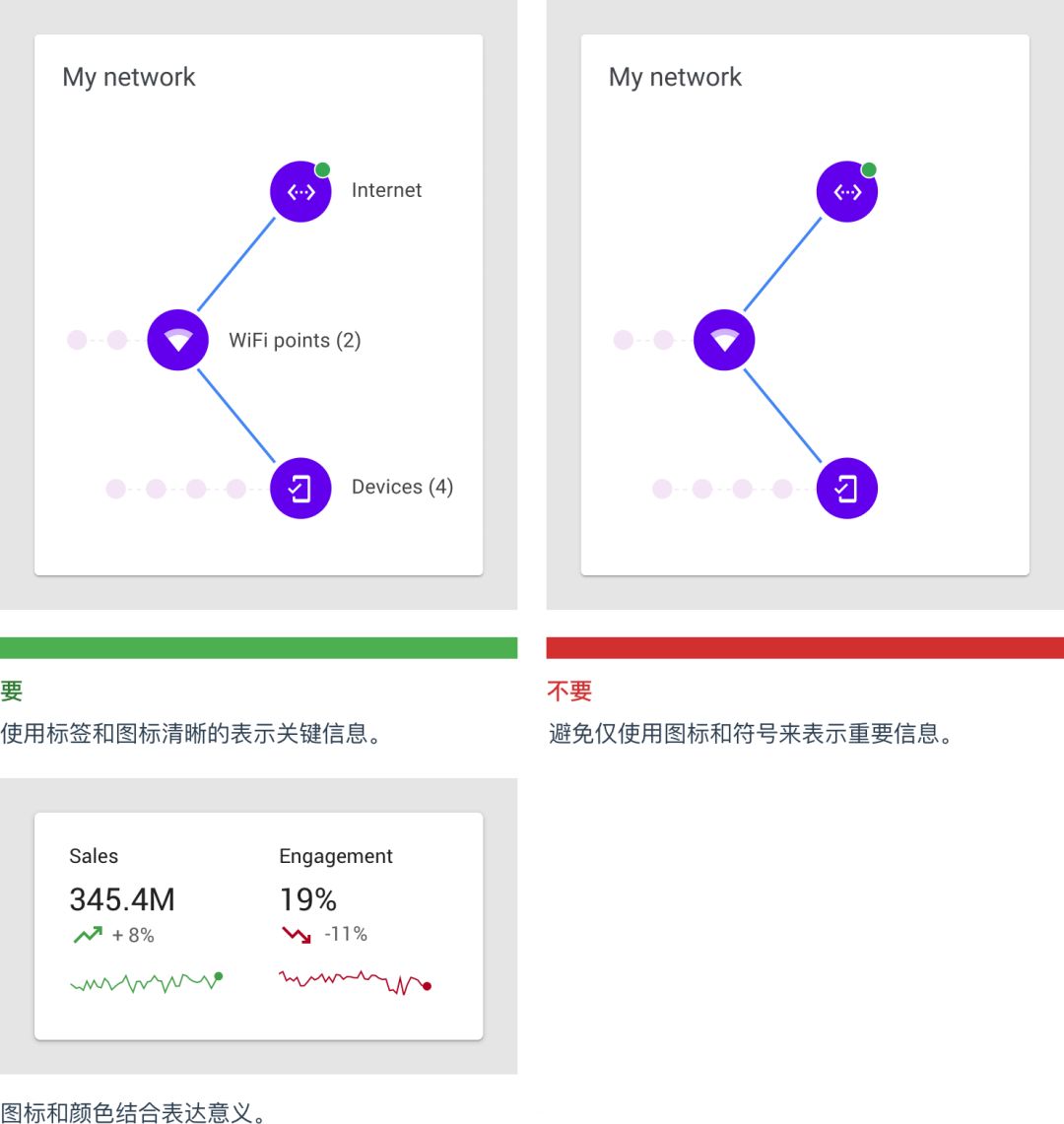
**图标可用于：**

**· 分类数据：**用于区分组或类别

**· UI控件和操作：**例如筛选，缩放，保存和下载

**· 状态：**例如错误，空状态，完成状态和危险

在图表中使用图标时，建议使用通用可识别符号，尤其是在表示操作或状态时，例如：保存，下载，完成，错误和危险。



**6. 坐标轴**

一个或多个坐标轴显示数据的比例和范围。例如，折线图沿水平和垂直坐标轴显示一系列值。



**柱状图（条形图）基线**

柱状图（条形图）应从为零的基线（y轴上的起始值）开始。从不为零的基线开始可能导致数据被错误地理解。

**坐标轴标签**

标签的设计应体现图表中最重要的数据。应根据需要使用标签，并在UI中保持一致性。他们的出现不应该妨碍查看图表。

**文字方向**

为便于阅读，文本标签应水平放置在图表上。

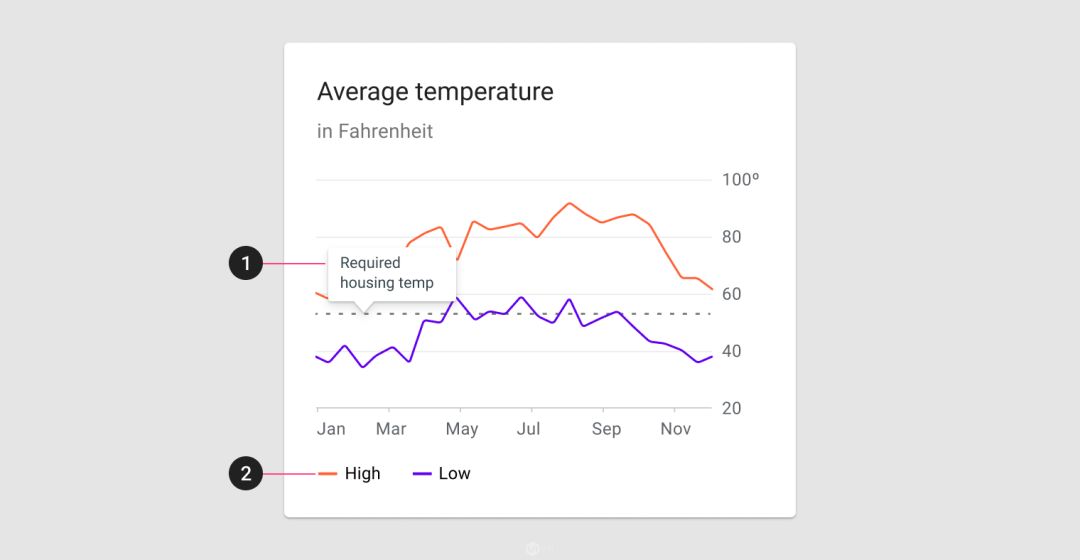
**文字标签不应该：**

**·**旋转

**·**垂直堆叠

**7. 图例和注释**

图例和注释描述了图表的信息。注释应突出显示数据点，数据异常值和任何值得注意的内容。



1. 注释

2. 图例

在PC端，建议在图表下方放置图例。在移动端，将图例放在图表上方，以便在交互过程中保持可见。

**标签和图例**

在简单图表中，可以使用直接标签。在密集的图表（或更大的图表组的一部分）中，可以用图例。

**8. 小显示屏**

可穿戴设备（或其他小屏幕）上显示的图表应该是移动端或PC端图表的简化版本。

**行为**

图表具有交互模式，使用户可以控制图表数据。这些模式可以使用户专注于图表的特定值或范围。

以下推荐的交互模式，样式和效果（如触觉反馈）可以提高用户对图表数据的理解：

**渐进式披露**

提供了按需求逐步展示详细信息的明确途径。

**直接操作**

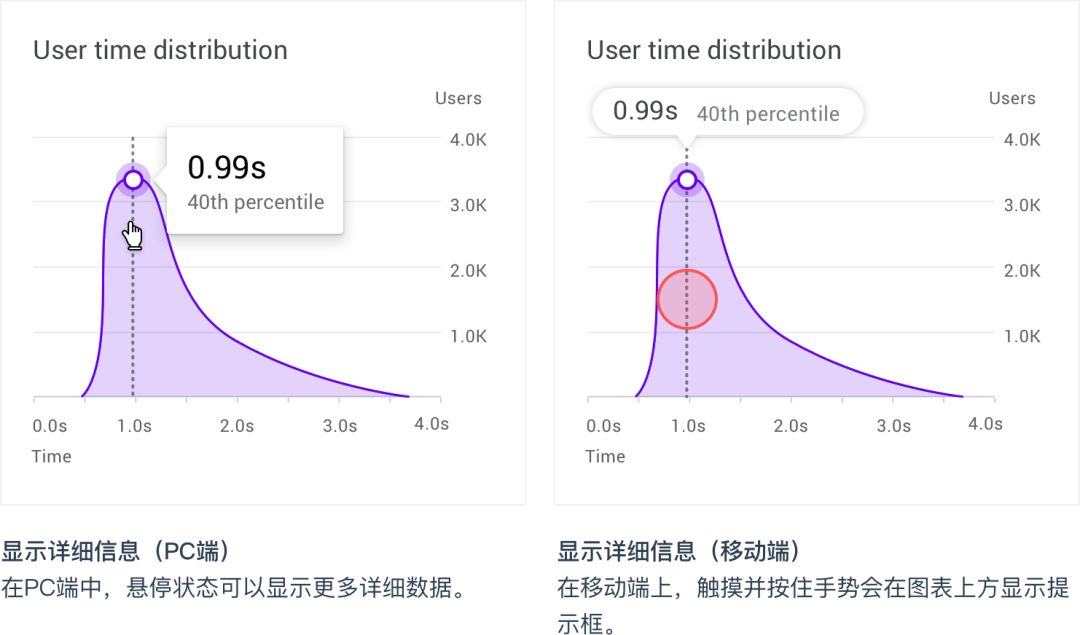
允许用户直接对UI元素进行操作，最大限度地减少屏幕上所需的操作数量，包括：缩放和平移，分页和数据控件。

**改变视角**

使一种设计可以适用于不同的用户和数据类型，例如数据控件和动效。

**1. 渐进式披露**

使用渐进式披露显示图表详细信息，允许用户根据需要查看特定数据点。



**2. 缩放和平移**

缩放和平移是常用的图表交互，会影响用户对图表数据深入的研究和探索。

**缩放**

缩放改变界面显示的远近。设备类型决定了如何执行缩放。

**·**在PC端，通过单击、拖动或滚动进行缩放

**·**在移动端，通过捏合进行缩放

当缩放不是主要操作时，可以通过单击和拖动（在PC端）或双击（在移动端）来实现。

**平移**

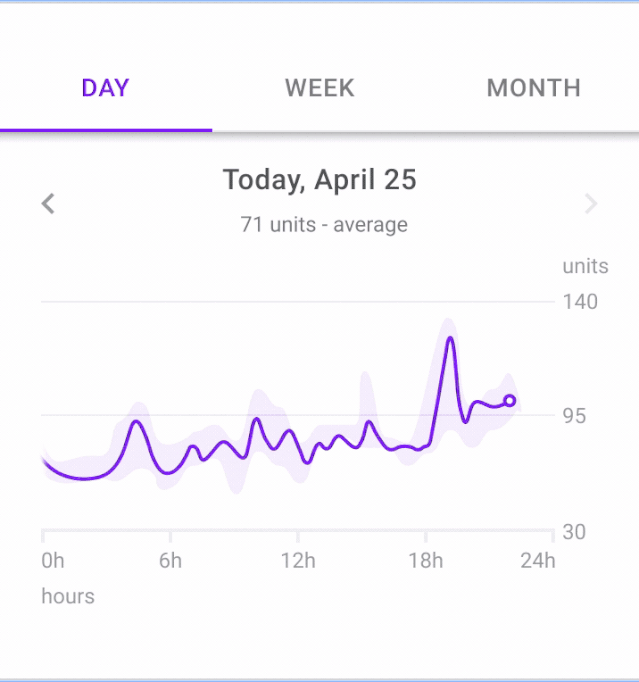
平移让用户能够看到屏幕之外的界面。它应该合理的展示数据的价值。例如，如果图表的一个维度比另一个维度更重要，则平移的方向可以仅限于该维度。

**·**平移通常与缩放功能同时使用。

**·**在移动端，平移通常通过手势实现，例如单指滑动。

**3. 分页**

在移动端，分页是一种常见模式，让用户通过向右或向左滑动来查看上一个或下一个图表。

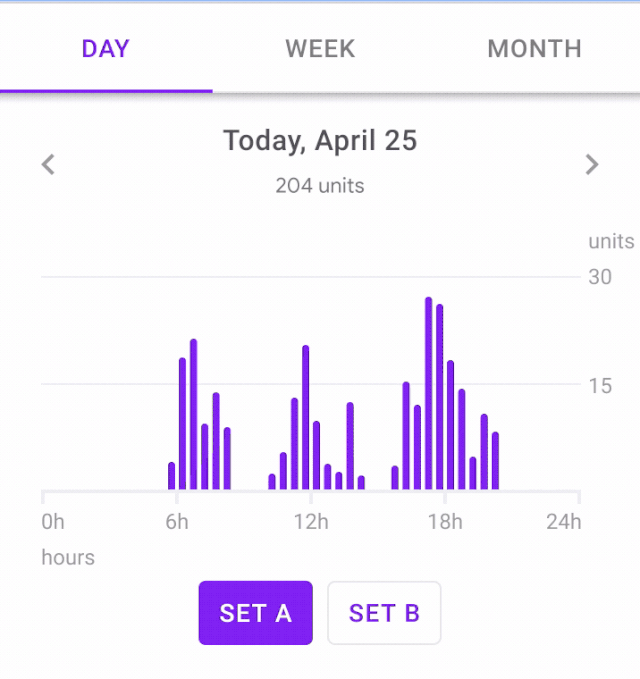


在移动端，用户可以向右滑动以查看前一天。

**4. 数据控制**

可以使用切换控件，选项卡和下拉菜单筛选或改变数据。

用户调节控件时，这些控件还可以显示指标。

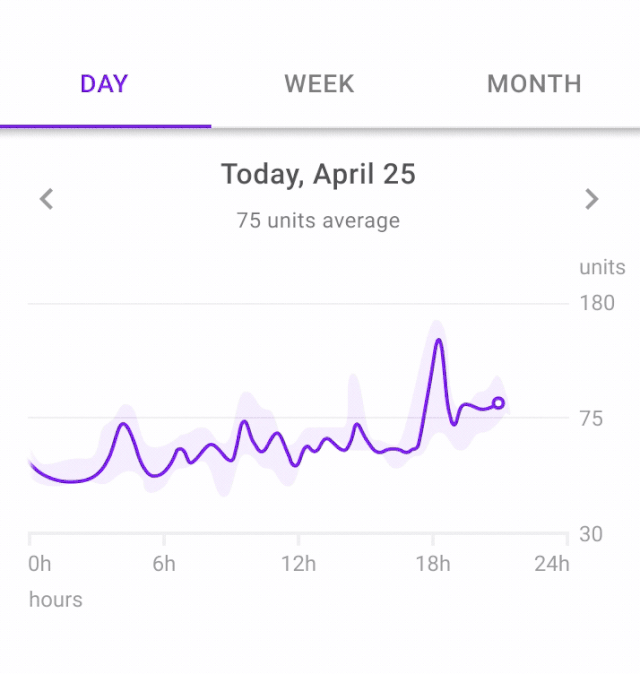


切换控件，选项卡和下拉菜单可以更改或筛选数据。

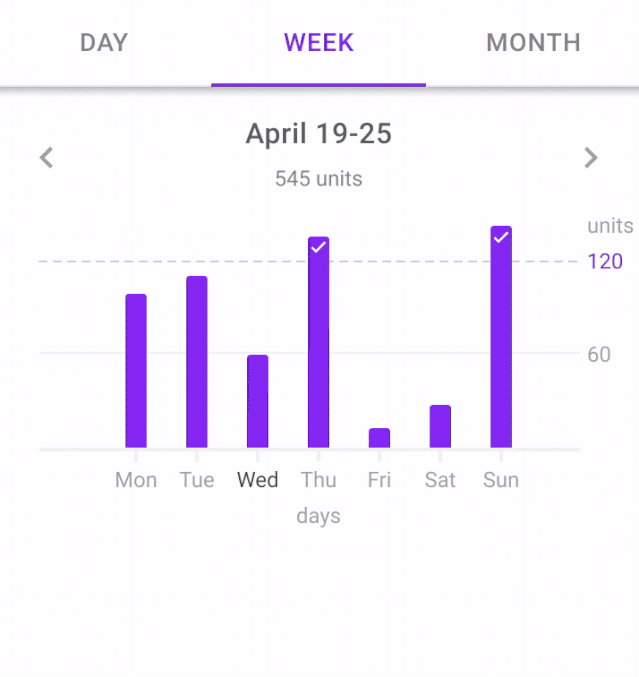
**5. 动效**

动效可以强化数据之间的联系，提升交互体验。应该有目的地使用运动（不是装饰性地），表达不同状态和空间之间的联系。

运动应该是合理，平稳，反应灵敏，不会妨碍用户的使用。



在此示例中，图表数据从按天显示动态切换到按周显示。转换期间不会显示所选日期范围之外的数据，从而降低了复杂性。

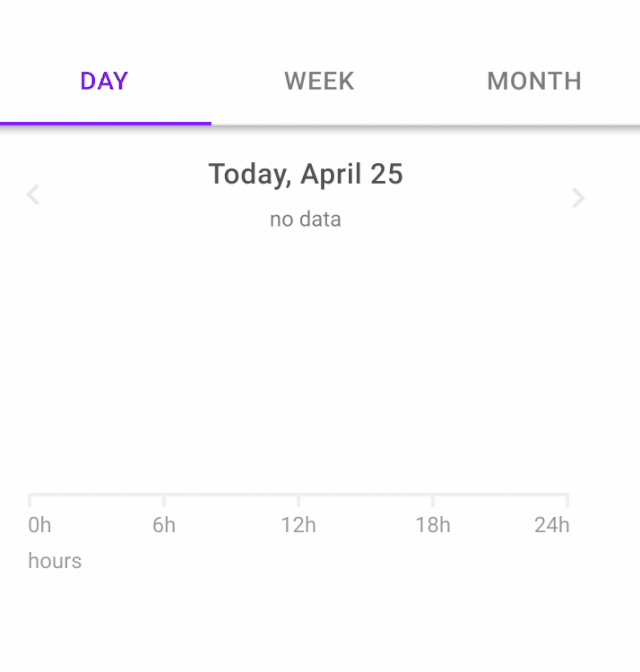


动画能够体现两个不同图表的相关性。

**6. 空状态**

图表数据为空的情况下，可以提供相关数据的预期。

在合适的情况下，可以展示角色动画创造愉悦和鼓励。



有特色的动画提升了空状态的效果。

**仪表板**

在称为仪表板的UI界面中，数据可视化通过一系列图表实现。多个独立的图表有时可以比一个复杂的图表更好地表达故事。

**仪表板设计**

仪表板的目的应在其布局，样式和交互模式中体现。无论是用来制作演示文稿还是深入研究数据，它的设计应该适合它的使用方式。

**仪表板应该：**

**·**突出最重要信息（使用布局）

**·**根据信息层级确定信息的焦点（使用颜色，位置，大小和视觉权重）

应根据对数据的需求确定信息的优先级并进行安排。在此示例中设计仪表板，考虑了以下用户问题：

1. 需要注意的问题  
2. 发生问题的时间

3.发生问题的位置

4.受问题影响的其他变量

**1. 分析类仪表板**

分析仪类表板让用户能够研究多组数据并发现趋势。通常，这些仪表板包含能够深入洞察数据的复杂图表。

**用例包括：**

**·**随时间变化的突出趋势

**·**回答“为什么”和“假设”的问题

**·**预测

**·**创建有深度的报告

**分析类仪表板示例：**

**·**跟踪广告活动的收效

**·**跟踪产品在其整个生命周期中的销售额和收入

**·**随时间变化的城市人口趋势

**·**跟踪随时间变化气候数据

分析类仪表板显示气候数据

**2. 操作类仪表板**

操作类仪表板旨在回答一组预设的问题。它们通常用于完成与监控相关的任务。

在大多数情况下，这些类型的仪表板具有一系列关于当前信息的简单图表。

**用例包括：**

**·**跟踪目标的当前进度

**·**实时跟踪系统性能

**操作类仪表板示例：**

**·**跟踪呼叫中心的活动，例如呼叫音量，等待时间，呼叫长度或呼叫类型

**·**监控在云端应用程序的运行状况

**·**显示股市情况

**·**监控赛车上的遥测数据

操作类仪表板显示设备存储指标

**3. 演示类仪表板**

演示类仪表板是为感兴趣的主题提供的展示视图。

这些仪表板通常包括一些小图表或数据卡片，用动态标题描述每个图表的趋势和见解。

**用例包括：**

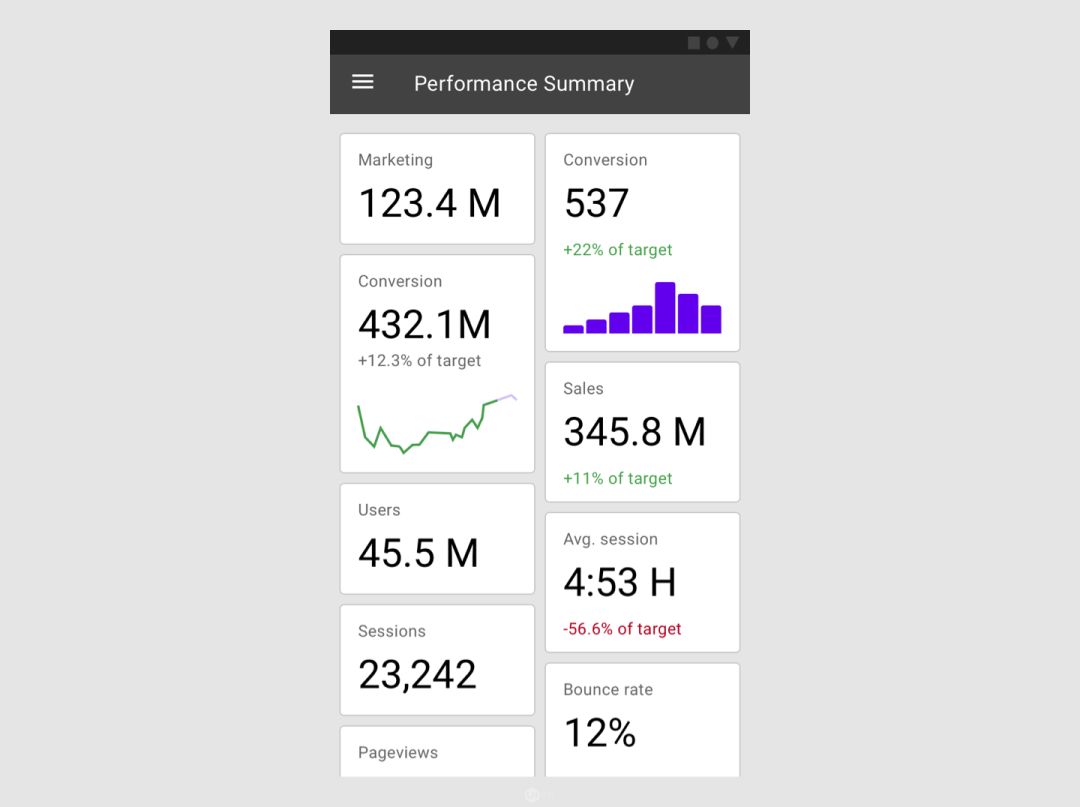
**·**提供关键绩效指标的总览

**·**创建高级执行情况的概要

**演示类仪表板示例：**

· 提供投资账户绩效的总览

· 提供产品销售和市场份额数据的概要



# 七、工作感悟

软件开发团队中，技术不是万能的，但没有技术是万万不能的

在技术型团队中，技术与人品同等重要，当然长相也比较重要哈，尤其在MM比较多的团队中。在软件项目团队中，技术水平是受人重视和尊重的重要砝码。无论你是做管理、系统分析、设计、编码，还是产品管理、测试、文档、实施、维护，多少你都要有技术基础。算我孤陋寡闻，我还真没有亲眼看到过一个外行带领一个软件开发团队成功地完成过软件开发项目，哪怕就一个，也没有看到。倒是曾经看到过一个“高学历的牛人”(非技术型)带一堆人做完过一个项目，项目交付的第二天，项目组成员扔下一句“再也受不了啦！”四分五裂、各奔东西。那个项目的“成功度”大家可想而知了。

详细制定自己软件开发专业知识学习计划，并注意及时修正和调整

请牢记：“如果一个软件开发人员在1、2年内都没有更新过自己的知识，那么，其实他已经不再属于这个行业了。”不要告诉自己没有时间。来自时间管理领域的著名的“三八原则”告诫我们：另外的那8小时如何使用将决定你的人生成败！本人自毕业以来，平均每天实际学习时间超过2小时。

不要仅局限于对某项技术的表面使用上，哪怕你只是偶尔用一、二次

“对任何事物不究就里”是任何行业的工程师所不应该具备的素质。开发Windows应用程序，看看Windows程序的设计、加载、执行原理，分析一下PE文件格式，试试用SDK开发从头开发一个Windows应用程序；用VC＋＋、 Delphi、Java、.Net开发应用程序，花时间去研究一下MFC、VCL、J2EE、.Net它们框架设计或者源码；除了会用J2EE、 JBoss、Spring、Hibernate等等优秀的开源产品或者框架，抽空看看大师们是如何抽象、分析、设计和实现那些类似问题的通用解决方案的。试着这样做做，你以后的工作将会少遇到一些让你不明就里、一头雾水的问题，因为，很多东西你“知其然且知其所以然”！

在一种语言上编程，但别为其束缚了思想

“代码大全”中说：“深入一门语言编程，不要浮于表面”。深入一门语言开发还远远不足，任何编程语言的存在都有其自身的理由，所以也没有哪门语言是“包治百病”的“灵丹妙药”。编程语言对开发人员解决具体问题的思路和方式的影响与束缚的例子俯拾皆是。我的经验是：用面对对象工具开发某些关键模块时，为什么不可以借鉴C、C51、汇编的模块化封装方式？用传统的桌面开发工具(目前主要有VC++、Delphi)进行系统体统结构设计时，为什么不可以参考来自Java社区的IoC、AOP设计思想，甚至借鉴像Spring、Hibernate、JBoss等等优秀的开源框架？在进行类似于实时通信、数据采集等功能的设计、实现时，为什么不可以引用来自实时系统、嵌入式系统的优秀的体系框架与模式？为什么一切都必须以个人、团队在当然开发语言上的传统或者经验来解决问题？？？“他山之石、可以攻玉”。

养成总结与反思的习惯

众所周知，对软件开发人员而言，有、无经验的一个显著区别是：无经验者完成任何任务时都从头开始，而有经验者往往通过重组自己的可复用模块、类库来解决问题（其实这个结论不应该被局限在软件开发领域、可以延伸到很多方面）。这并不是说，所有可复用的东西都必须自己实现，别人成熟的通过测试的成果也可以收集、整理、集成到自己的知识库中。但是，最好还是自己实现，这样没有知识产权、版权等问题，关键是自己实现后能真正掌握这个知识点，拥有这个技能。

理论与实践并重，内外双修

工程师的内涵是：以工程师的眼光观察、分析事物和世界。一个合格的软件工程师，是真正理解了软件产品的本质及软件产品研发的思想精髓的人（个人观点、欢迎探讨）。掌握软件开发语言、应用语言工具解决工作中的具体问题、完成目标任务是软件工程师的主要工作，但从软件工程师这个角度来看，这只是外在的东西，并非重要的、本质的工作。学习、掌握软件产品开发理论知识、软件开发方法论，并在实践中理解、应用软件产品的分析、设计、实现思想来解决具体的软件产品研发问题，才是真正的软件工程师的工作。站在成熟理论与可靠方法论的高度思考、分析、解决问题，并在具体实践中验证和修正这些思想与方式，最终形成自己的理论体系和实用方法论。

心态有多开放，视野就有多开阔

不要抱着自己的技术和成果，等到它们都已经过时变成垃圾了，才拿出来丢人现眼。请及时发布自己的研究成果：开发的产品、有创意的设计或代码，公布出来让大家交流或者使用，你的成果才有进化和升华的机会。想想自己2000年间开发的那些Windows系统工具，5、6年之后的今天，还是那个样子，今天流行的好多Windows系统工具都比自己的晚，但进化得很好，且有那么多用户在使用。并且，不要保守自己的技术和思想，尽可能地与人交流与分享，或者传授给开发团队的成员。“与人交换苹果之后，每个人还是只有一个苹果；但交换思想之后，每个人都拥有两种思想”，道理大家都懂，但有多少人真正能做到呢？

尽量参加开源项目的开发

网络早已不再只是“虚拟世界”，网上有很多的开源项目、合作开发项目、外包项目，这都是涉猎工作以外的知识的绝好机会，并且能够结识更广的人缘。不要因为工作是做ERP，就不去学习和了解嵌入式、实时、通信、网络等方面的技术，反过来也是一样。如果当别人拿着合同找你合作，你却这也不会，那也不熟时，你将后悔莫及。

不要将自己的知识面仅仅局限于技术方面

诺贝尔经济学奖得主西蒙教授的研究结果表明： “对于一个有一定基础的人来说，他只要真正肯下功夫，在6个月内就可以掌握任何一门学问。”教育心理学界为感谢西蒙教授的研究成果，故命名为西蒙学习法。可见，掌握一门陌生的学问远远没有想象的那么高难、深奥。多方吸取、广泛涉猎。极力夯实自己的影响圈、尽量扩大自己的关注圈。财务、经济、税务、管理等等知识，有空花时间看看，韬光养晦、未雨绸缪。