优秀的前端工程师

- 领域知识
- 能力、潜力
  - 编程
  - 架构
  - 工程
- 职业规划
  - 阿里要求、职级体系
  - 发钱给你的事不太可能会对你自身有好处，如果有->福报
  - 自己找机会
  - 开源
    - 1. 帮写文档
    - 2. fix bug
    - 3. 单步追踪
    - 4. 提交作者review
- **成就**
  - 做的所有努力，都需要通过成就来体现

职业发展：成长 成就 晋升

- 成就
  - 业务型成就（方法论、思考过程）
    - 业务目标
      - 理解公司业务的核心目标
      - 目标转化为目标
    - 技术方案
      - 业务指标到技术指标的转化
      - 形成纸面方案、完成小规模试验
    - 实施方案
      - 确定实施目标、参与人
      - 管理实施进度
    - 结果评估
      - 数据采集、数据报表
      - 向上级汇报
  - 工程型成就
    - 目标：质量、效率
    - 方案与实施
      - 规章制度
      - 库
      - 工具
      - 系统
    - 结果
      - 线上监控
  - 技术难题
    - 目标-公认的技术难点
    - 方案与实施-依赖扎实的编程能力、架构能力形成解决方案
    - 结果-问题解决

数据驱动的思考方式

- 目标
  - 分析业务目标
    - 促进用户活跃
    - 提升营收
  - 定数据指标
    - 评估活跃度：日活除以月活
- 现状(参照组)
  - 采集数据
    - 业务数据
    - 技术数据
      - performance
      - error
  - 建立数据展示系统
- 方案(最容易直接跳到这一步)
  - 设计技术方案
  - 预估数据（对上沟通）
- 实施
  - 小规模实验
  - 推广全公司落地
  - 形成制度
- 结果
  - 统计最终效果
  - 汇报

前端技能模型（解决问题）

- 能力
  - 编程
  - 架构
  - 工程
- 前端知识
- 领域知识

工具链（套件）

- 作用
- 分类
  - 脚手架 init、add
  - 本地调试 run、build
  - 单元测试 test
  - 发布 publish
- 工具链体系的设计
  - 版本问题
  - 数据统计

持续集成 vs 最终集成

- 客户端软件持续集成
  - Daily build
  - BVT, Build Verification Test
- 前端持续集成
  - Check-in build
  - Lint + Rule Check
    - Rule，单张图片<50k，总包<200k（4 屏全图）

技术架构

- 客户端架构：解决软件需求规模带来的复杂性（前端页面天然解耦）
- 服务端架构：解决大量用户访问带来的复杂性（用户设备）
- 前端架构：解决大量页面需求带来的重复劳动问题 - 复用
  - 库：有复用价值的代码
    - URL
      - IETF：https://tools.ietf.org/html/rfc3986
    - AJAX
      - 裸体不可用，需要封装，稳健
    - ENV
      - 判断环境
  - 组件：UI 上多次出现的元素。组件的定义和基础设施，就是**组件化方案**
    - 轮播
    - Tab
  - 模块：经常被使用的业务区块
    - 登录

SpriteJS 单元测试做的比较好

提升代码质量

- 多写
- 看开源项目

业务代码 -> 业务价值，能不能找到
眼里有没有活