**课程大纲：**

1. 神马是敏捷？
2. 神马是SCRUM？

SCRUM是当前最火的一种敏捷开发方法，有用户故事、冲刺、燃尽图等很多很酷的玩法，有牛B的产品负责人、SCRUM Master，有超强的自组织团队。

* 1. SCRUM是神马东西？
  2. 极限编程（XP）与SCRUM的关系。
  3. 牛B的产品负责人、SCRUM Master及“自组织”团队。

1. SCRUM的敏捷最佳实践

SCRUM的敏捷实践有很多，每个实践都有独到的魅力，并且实践之间有逻辑关系。学员需要通过具体的实例来实战这些最佳实践，并且需要带着问题进入下一章。

* 1. 需求方面最佳实践：客户故事、客户全程参与
  2. 设计方面的最佳实践：简单设计
  3. 测试方面最佳实践：测试驱动开发、自动化测试
  4. 编码方面最佳实践：重构、结对编程、代码共有、强调编码标准
  5. 项目管理方面最佳实践：冲刺（小版本发布）、燃尽图、持续集成、站立会议、每周工作40小时
  6. 其他最佳实践：隐喻

1. 敏捷在中国的“水土不服”
   1. 软件项目的“两大限死、两不确定” 。
   2. 中国IT人的特点。
   3. 敏捷需要“强矩阵”而不是“弱矩阵”。
2. 打造“自组织”团队（自我管理团队）

项目的问题说到底是人的问题， 提升项目组成员的“软技能”，让大家都在同一条船上，是打造高效率团队的关键。

* 1. 解决项目组成员的地位不平等问题。
     + 技术人员的先天优越性 VS 测试、实施人员的低人一等
     + 外科手术式团队
     + “无领导”的扁平架构
  2. 谁来做决策？
     + 项目经理“话事制”——独裁主义？
     + 达成一致——人人有份？
     + 专家决策——我是老大我说了算？
  3. 鼓励“犯错”！
     + “犯错”是不是应该受到惩罚？
     + 你喜欢无错误的“优秀”员工吗？
     + 挑肥拣瘦，不敢担当？——可能就是因为没有鼓励“犯错”！
     + 失败、错误、总结是成功他妈！无错误的人很可能是无水平的人。
  4. 让团队“自组织”！

1. 打造良好敏捷团队文化的超级利器：赞美与同理心
   1. 读《为什么技术人员喜欢攻击人》有感
   2. 自我设限与防卫心理
   3. 赞美的威力、赞美的技巧
   4. 小猪、绵羊、奶牛的故事
   5. 什么是同理心？同理心的四个等级
2. 实践用户故事

开发人员抱怨：“需求变化太快，软件系统经常要修改，都连续加班几个星期了，还要改，需求分析怎么做的？”需求分析人员无可奈何地说：“明明是根据客户的阐述，做了记录，然后开发出了软件，客户却说很多地方不符合他们的意思，又要求修改，我怎么办？”

——敏捷需求分析如何破解这些问题呢？

* 1. “传统”需求分析方法有什么问题？
  2. 敏捷需求分析的核心原理
  3. 产品负责人与开发团队的关系
  4. 项目的战略分析
  5. 应用“用户故事”进行涉众分析
  6. 持续拆分“用户故事”

1. 实用敏捷估算、项目计划与跟踪

项目管理不是项目经理一个人的事情，应该让所有人参与进来，每个人自觉管理自己的工作，主动报告和解决问题，共同为项目的成功而努力！

* 1. 你有估算潜质吗？
  2. 预算与估算基础知识
     + 给定预算做估算
     + 估算的两种情况
     + 什么叫项目规模？
  3. 如何估算？
     + 估什么？
     + 谁来估？
     + 怎样估？
  4. 玩转估算（估算实战）
     + 基于用户故事的估算
     + 估算扑克
     + 估算经验的积累
  5. 计划执行与跟踪
     + 故事看板（任务看板）
     + 每日例会及每日例会进阶
     + 项目总结会

1. 测试先行——实践测试驱动

直到最后几天，测试工程师们才能见到软件的“庐山真面目”，但是不见不知道一见吓一跳，软件的问题巨多，甚至很多功能没有实现，然则距离“项目死期”（交付日）已经没有几天了！测试驱动是保证项目组一直都在做正确事情的最佳保障，可减少大量的不必要返工，测试驱动其实没有这么复杂和高难度，可以马上实践，本节将为你支招！

* 1. 测试症候群
  2. 测试人员的主要职责及应掌握的技能
  3. 测试要干什么？
     + 测试设计（案例分析）
     + 测试用例（案例分析）
     + 测试工作的难度和挑战性不亚于软件设计！
  4. 项目组中谁来负责测试？
     + 开发与测试的黄金比例？
     + “潮水式”测试人员安排
  5. 如何测试驱动？
     + 测试从需求开始！
     + “提前”测试 VS“正规”测试
     + 发布管理与缺陷的最终裁定权

1. 提升代码质量

文档可以没有，代码不能没有！为了节省时间不写文档，直奔代码，但可惜得到的往往是烂代码，烂代码编译出来的就是烂软件！代码质量成为我们心头之痛，不抓代码质量，后患无穷，马上猛抓代码质量，工期又无法保障！如何才能打破这个怪圈，逐步改善代码质量呢？

* 1. 编码基础知识
     + 面向过程、面向对象、面向……
     + 编程基本功
     + 零缺陷意识
  2. 编码的问题及“ 漂亮”的代码
     + 编码综合征
     + 代码的低级问题与高级问题
     + 什么是“漂亮”的代码？
  3. 编码规范
     + 空降的规范没有好结果——仅制定能执行的规范
     + 规范的执行
  4. 代码评审
     + 案例分析
     + 实践建议

1. 敏捷不是银弹