软件用户界面开发

* 开发前提条件：交互设计
  + 关系用户界面的外观与用户行为
  + 不受软件约束，而是根据用户的需求与特点，结合人机工程和心理学的设计原理进行设计
  + 建立用户模型
* 开发目标成果：界面软件
  + 关系到代码的生成即代码本身的意义，有代码开发人员进行软件开发
  + 很多时候，对用户来说最好的交互方式，往往是程序员最难实现的方式
    - 思考：这一点深有感触（先吐槽一下）。不过仔细想一下，造成这种情况也有特定的原因：
      * 软件逻辑往往复杂抽象难以理解，而用户界面需要简单具体，这个由复杂到简单的过程导致了软件开发中“额外的”复杂性，这只能由开发人员来承担
      * 甚至很多情况下用户本身的需求也并不确定
* 软件开发周期
  + 定义
    - 可行性分析
      * 调查用户的界面要求
      * 人机界面涉及的软硬件环境
      * 市场调研
    - 需求分析
      * 用户特性分析
      * 应用环境分析
      * 任务分析
  + 构造
    - 概要设计
      * 基于需求分析形成UI概要设计
      * 技术方案概要设计
    - 详细设计
      * 界面详细设计
        + 屏幕显示和布局设计
        + 控制模式设计
        + 细化并确定用户模型
        + 用户匹配度完善
      * 引导、提示、报错界面设计
      * 技术方案详细设计
    - 界面实现
    - 综合测试与评估
  + 维护
    - 运行与维护
      * 改正性维护
      * 完善性维护
      * 适应性维护
        + 常常有出人意料的结果
      * 预防性维护
* UI评估模型
  + QoE用户体验质量评估
  + 用户判断行为与设计师诱导行为一致->好的设计
* 软件UI设计开发原则
  + 以用户为中心原则
    - 用户对系统的期望和态度
    - 人机匹配性
      * 了解用户意图
      * 软件适应用户
    - 用户的技能和弱点
    - 用户的知识背景
  + 一致性原则
    - 概念模式一致性
    - 外观布局一致性
    - 交互方式一致性
    - 信息显示格式一致性
  + 提供信息反馈
    - 信息反馈形式
      * 文本、图形、动画、声音
    - 布局简洁明了
  + 利用显示效果实现内容与形式统一
    - 合理配色
    - 静态动态结合
  + 使用图形和比喻
  + 提供友善的出错处理
    - 检测错误
    - 提供错误信息
    - 提供应对策略
    - 提供数据保护和恢复功能
    - 不干扰熟练用户工作
  + 提供快捷方式
  + 允许Undo
  + 快速响应，低系统开销
  + 设计良好的帮助、引导和提示界面

图形用户界面GUI

* 发展回顾
  + MIT旋风一号（1950）
  + GUI概念由施乐公司的Alan Kay提出
  + Alto计算机（1973）
  + Star8010（1981）
  + Macintosh（1984）
  + Windows系统（1985，1990）
* GUI主流形式
  + WIMP界面
    - Windows，Icons，Menus，Pointers
  + 数据可视化界面
    - 数据图形化：线图、饼图、动态图、高维图
    - 明确直观地表现数据的深层语义
  + 沉浸式交互
    - VR、AR
* 实例
  + Echarts
* GUI特点
  + 信息量大
    - 视觉通道本身特点
    - 图形包含的语义丰富
  + 已于用户理解记忆学习
  + 图形交互便捷多样功能强大
* 常用GUI开发工具
  + QT
  + wxWidgets
  + GTK+
  + WPF
  + Web端GUI工具和3D引擎
  + 游戏引擎UI系统

QoE

* 用户体验指标
* 收集用户行为数据，衡量用户主观体验
* 问卷调查、用户评价反馈、数据指导QoE

QoS

* 技术服务质量指标

影响QoE指标的因素

* UI设计
* 系统QoS（性能指标）
* 地区、直播间互动（空间维度）
* 时事状态、日期、时间端（时间维度）
* 同时应用的其他系统

QoE与QoS间的关系

* 二者有不同的量化统计指标和方法
* 二者相关联
* QoE有时可以通过QoS指标来评价
* QoS设计也需要考虑用户体验

很多时候，对用户来说最好的交互方式，往往是程序员最难实现的方式

* 思考：这一点深有感触（先吐槽一下）。不过仔细想一下，造成这种情况也有特定的原因：
  + 软件逻辑往往复杂抽象难以理解，而用户界面需要简单具体，这个由复杂到简单的过程导致了软件开发中“额外的”复杂性，这只能由开发人员来承担
  + 甚至很多情况下用户本身的需求也并不确定

出入一点通项目用户体验质量评估

* 首先本项目是以让住户出入小区尽可能方便为目的，为管理员提供方便的数据检索方法的一套软件系统
* 对用户端来说，为了追求简洁，用户接收到的信息是非常少的，因此用户的体验质量评估比较简单。首次使用成功通过后用户的信任程度会提升到一个比较高的水平，之后再成功通过，用户的信任程度只会有极少量的提高。
* 对于管理员端来说，用户需要接收大量的信息，但是在前端界面的帮助之下，信息的筛选会变得比较容易，在数据可视化方面，软件的功能非常稳定，因此影响用户信任程度的就主要是信息的真实性，而在系统正常运行的前提下，数据是真实的，因此，用户信任程度会从一开始的比较低到第一次使用后提高之后稳定下来。