1. 交互设计的要点

- 人们喜欢简单、值得信赖、适应性强的产品。
- 所有不必要的功能都是有成本的。
- 简单并不意味着最少化。简单朴素的设计仍然具有自身的特征和个性。
- 简单并不意味着欠缺或者低劣，也不意味着不注重装饰或者完全赤裸裸，而是说装饰应该紧密贴近设计本身，任何无关的要素都应该予以删除。
- 每个好的设计都是在考虑诸多限制之后给出的方案。最好是在设计之初就搞清楚存在哪些限制，然后才能保证自己的设计与用户的需求紧密贴合。
- 先理解用户，然后再考虑合适的设计。
- 软件使用环境是观察用户的最佳地点。
- 忽略专家用户的复杂需求，专家用户想要的功能往往会吓退主流用户。
- 专注于主流用户的目标，而不是专家用户或者是随意型用户。简单地满足主要的需求往往是主流用户的第一诉求。
- 主流用户和专家用户的关注点差异：

主流用户	专家用户
快速地完成任务，追求速度	完善地完成任务，追求细节
操控容易	操控精确
得到靠谱的结果	得到完美的结果
排斥一切差错，没有耐心	容忍差错，刨根问底
合适就行	精确匹配
实例和故事	原理

- 简单的用户体验是新手和初学者能够快速掌握的体验，或者是压力之下的主流用户的体验。
- 关注用户的感情需求和隐形需求。
- 简单就是感觉在掌控一切。
- 故事是描述用户体验的最好方式，使用故事的细节来表达设计的特性，把设计的所有约束都诉诸于文字，每一个细节都至关重要。故事要简短，简单的故事容易记住，复述和传播，也容易让人将故事的情景套用到别的环境中。故事要具象而不是抽象，描述用户的行为比介绍用户的性格更为让人印象深刻。故事要真实，最好来源于真实，合理的使用细节也使得故事更加真实。

- 为了使设计足够简单，需要使用极端的可用性目标而不是常规的可用性目标来考量。

2. 交互设计策略一：删除

2.1 删除

- 删除一切不必要的功能

2.2 避免错删

- 删除的前提是充分认识功能的价值，不能因为当前资源限制而删除那些最有意义却较难实现的功能，要纵览全局，保证交付那些最有价值和意义的功能。

2.3 关注核心

- 与新增功能相比，用户更关注基本功能的改进

2.4 砍掉残缺功能

- 删除掉实现得不够理想的功能。对于实现得不够理想的功能，应考虑为什么要留着它，而不是为什么应该删掉它，不要陷入“沉没成本误区”。

2.5 不求大而全，避免猜想，多做考察

- 不要因为对于用户需求的猜想而保留想要删除掉的功能。功能是否应该被删除应该着眼于软件的要解决的问题的核心，考察需求的重要程度，对于不必要，不常见的需求，可以选择忽略。避免做大而全的产品。

2.6 不能简单地根据用户需求而增加功能

- 搞清楚用 户到底遇到了什么问题，是不是需要由我们的软件来解决。要倾听用户意见，但是绝不能盲从。

2.7 关注方案而不是流程

- 如果一个小的变化导致了复杂的流程，就应该退一步去寻找更好的解决方案。

2.8 去掉负担

- 去掉那些可有可无的选项，内容和分散人们注意力的玩意，可以减轻用户负担，让用户专心做自己想做的事。去掉分散注意力的视觉元素，可以让用户感觉更快速，甚至更有安全感。

2.9 减少选择的数量

- 选择有限，用户反而更喜欢。

2.10 减少错误

- 从用户使用场景去充分考虑用户出错的可能方式从而提前规划避免用户出错，通过提示用户操作错误然后由用户自己修改的实现方式远不及提前发现用户错误而避免用户出错的实现方式，减少错误能让用户感觉到产品稳定易于掌控。

2.11 精简内容

- 减少文字，精简信息，减少干扰用户的视觉元素。

3. 交互设计策略二：组织

3.1 分块

- 把项组织到6到8个块中。分块越少，用户的选择越少，用户的负担越小。

3.2 围绕行为进行组织

- 着手组织之前，先要理解用户行为，将用户行为的步骤理清楚，并以此为根据通过组织分块来简化用户操作。

3.3 界限清晰

- 对要组织的项进行界限清晰的分类。借助用户调查来观察用户更倾向的分类。

3.4 字母表和格式

- 根据字母表和格式进行分类往往会把事情搞乱。

3.5 搜索和组织

- 在项的量可组织的情况下，对用户来说，组织的内容比搜索内容更简单。

3.6 时间线是组织的通用方式之一

3.7 网格

- 利用不可见的网格来对齐界面元素是吸引用户注意力的一种方式。

3.8 大小和位置

- 元素的大小体现元素的重要性。要想办法表现出不同的重要性，记住一条规则，二分之一重要性的元素的大小要做成四分之一。

3.9 色标

- 使用颜色来标记信息需要用户付出额外的学习理解精力，容易造成用户负担。在不必要的情况下添加颜色会导致困惑。在确保用户会愿意花时间学习并且会重复使用某种设计时，色标系统才是合适的，或者用户完全知道其含义的色标也没问题。

3.10 期望路径：人们并不总是走你为他们铺好的路

- 在描绘用户使用软件的路径时，不要被自己规划的清晰的路线图和布局所迷惑。不断使用软件的流程，观察用户使用软件的流程，然后找出期望路径。简单的组织，意味着用户在使用软件时会对什么流程更适应和实际使用，而不是设计者在规划中看到了什么清晰的逻辑。

4. 交互设计策略三：隐藏

- 隐藏比组织具有的优势：用户不会因为不常用的功能而分散注意力。为了删除而预先隐藏不可取，应该充分考虑是否需要删除，如需删除，务必从速，而非隐藏

4.1 不常用但不能少

- 主流用户很少使用，但是软件自身必不可少的功能，适合隐藏。这些功能与用户目标没有直接关系，这些功能的常见特点是事关细节、选项和偏好以及特定于某种分类的信息。

4.2 不要让用户自定义界面

4.3 不要根据用户习惯自动定制用户界面

4.4 渐进展示：隐藏部分功能但提供入口

- 展示核心功能，渐进展示扩展功能。

4.5 阶段展示：展示随着过程的深入而扩展。

- 为一个展示阶段设定一个场景便于用户理解。
- 通过讲故事的方式将各个阶段有逻辑易于理解地连贯地展开。
- 使用用户的语言对阶段细节进行描述，避免使用用户不易理解的术语和缩略语。
- 阶段划分的粒度要小，每个阶段要自成一体。

4.6 适时出现

- 成功的隐藏：尽可能彻底的隐藏所有需要隐藏的功能，但是，需要在合适的时机，合适的位置上及时显示相应的功能。

4.7 提示与线索

- 对于隐藏的功能的提示和线索，不能让用户感到抗拒和可怕，要让用户感到有信心使用这些隐藏功能。

4.8 界面空间关注点

- 在用户使用软件的某项功能时，用户的视线会聚焦于这片功能区中，对于此时需要提示的隐藏功能，提示应该出现在用户关注点附近，否则用户会难以发现提示

4.9 隐藏的规则总结

- 隐藏所有一次性设计和选项。
- 隐藏精确控制选项，但专家用户必须能够设置这些选项为始终可见。
- 不可强迫或者寄希望于主流用户使用自定义功能，不过可以给专家用户提供这个选项。
- 巧妙地隐藏。即首先彻底隐藏，然后适时出现。

5. 交互设计策略四：转移

5.1 平台间转移

- 在合适的平台上展示合适的信息和功能。

5.2 向用户转移

- 首先需要分析清楚在整个满足需求的流程闭环中，哪些流程该由用户来完成，哪些流程该由软件来完成。要调查了解用户到底擅长做什么，将用户擅长的事交给用户去做，将计算机擅长的事交给软件做。用户指挥，软件操作，就会给人简单的感觉。

5.3 聚合功能，创造开放式体验

- 将多个相似的功能组合到一起，然后让这个聚合的功能具备多个功能的能力。聚合功能使得功能理解变得简单，功能维护更加容易。

5.4 强化功能通用性，减少功能特性

- 开放性界面的秘诀在于，功能尽可能的接近底层而通用，减少适合中级用户的“便捷”特性。

5.5 向用户提供非结构化数据信息入口，由软件完成数据的结构化

5.6 信任

- 设计功能转移需要信任用户有能力识别功能的转移方向。信任就是要避免剥夺用户的决定权。构筑信任可以通过邀请用户参与测试软件原型，了解用户对于功能分布的理解程度，在此基础下，大胆地将决定权交给用户，让用户专注于选择和指挥，让软件专注于存储和计算。避免总是控制和指挥用户。