**软件测试行业基本介绍**

为什么需要软件测试

1.一款软件从无到有会经历很多的开发阶段并由不同的人进行开发，所以最终产出的软件功能可能会存在问题，因此为了保证软件功能是可以用的，我们必须要测试

2.当前的软件已经不仅仅只看重功能，还会对软件是否容易上手，执行效率等有了更高的要求，这也是需要软件进行大量测试的

为什么选择软件测试

1.软件测试行业对于专业的软件测试人员需求慢慢变大

2.有的人喜欢创造世界所以他们做了开发，我们就是希望世界变得更加美好所以做了测试

为什么不让开发自己测试

1.当前行业有许多都是测试人员之前都是开发

2.专业度：软件测试和软件开发属于软件行业两个不同的技术方向，专人做专事对于质量更有保证

3.思维定势:开发更多的是去考虑如何实现软件的功能，而不会从用户的角度考虑如何奇葩的使用这些功能

4.测试力度：产品相当于开发的孩子，所以下手不会那么狠

**软件测试基本介绍**

软件测试就是对被测对象进行测试操作，从而验证实际结果与预期结果是否一致

软件测试的作用

通过测试工作可以发现并修复软件当中存在的缺陷，从而提高用户对产品的信心

测试可以记录软件运行过程中的一些数据，从而为决策提供数据支持

测试可以降低同类型产品开发遇到的问题

**测试原则(7条)：**所谓的测试原则指的就是我们在执行测试工作时必须要遵守的一些规则。

1．测试证明软件存在缺陷：无论执行什么样的测试操作都能证明当前软件是有缺陷的。

2．不能执行穷尽测试：有些功能是没有办法将所有的测试情况都逻列出来，所以任何的测试操作都有结束的时间。

3．缺陷存在群集现象：对于软件功能说，核心功能占 20%，非核心是 80%。在实际工作中我们会集中测试 20%的核心功能，所以这个部分发现缺陷的几率就会高于 80%。因此我们我们就会遇到缺陷都集中在 20% 功能模块里的现象。

4．某些测试需要依赖特殊的环境

5．测试应尽早介入：为了更多的发现和更好的解决软件中的缺陷，我们追求测试工作尽早的开展。

6．杀虫剂现象：同样的一个测试用例不能重的执行多次，因为软件会对它产生免疫。

7．不存在缺陷谬论：任何软件不可能是完美的。

**测试对象介绍**

对于当前的测试行业来说我们最经常测试的主体就是 软件( 主体功能 )，但是需要我们明白是一个软件也不仅仅只有功能需要测试。我们可以将软件分为三个部分组成：功能集合+使用说明书 + 配置数据。

对于一款软件来说从无到有需要不同的过程，我们可以将这个过程分为不同阶段，然后每个阶段都会相应有测试对象

1. 需求分析阶段：各种需求规格说明书。

2. 软件架构设计：API 接口文档( 接口测试 )

3. 编码实现阶段：源代码（ 白盒测试、单元测试 ）

4. 系统功能使用：软件功能主体（ 当前行业做的最多的一种测试 ）

**测试级别**

软件的开发都会依据相应的开发模型， 测试级别则指的就在这个模型当中我们人为定义的开发步骤。其中对于测试来说我们最常见的一种级别分类如下：

1．单元测试[ UT unit test ]：在软件测试中单元指的就是组成软件最小的底层代码结构，

 一般就是类、函数、组件（ 当下的软件测试行业，不会刻意要求测试人员对源代码进行测试 ）。

 2．集成测试( IT system ingertaion test ):将多个单元模块组合在一起，然后验证它们之间沟通的"桥梁"是否能正常工作( 接口测试 )

 3．系统测试( ST system test ):这是当前行业做的最多的一种测试。由测试人员充当用户然后对软件的功能主体进行测试。

 4．验收测试:

(1) a测试 ---- 内测

(2) b测试 -----公测

(3) UAT( user acceptance test )测试---- 由客户派出对于业务非常精通的人员来使用该软件，从而对功能进行测试。

(4) 验收测试的核心就是让用户为当前软件 "买单"

**系统测试分类**

1. 功能测试：验证当前的软件主体功能是否可用。

2. 兼容性测试：验证当前软件在不同的环境下是否还可以使用。

3. 安全测试：验证软件是否只是能授权用户提供功能使用。

4. 性能测试：相对于当前软件消耗的资源 它的产出能力。

**常见的系统测试方法**

**一、**按测试对象进行分类

 1．白盒测试：这种测试的主体就是软件的底层代码，不会在意外在的界面是否 OK ，只要求底层功能实现，同时逻辑正确。

 2．黑盒测试：这种测试就是指测试软件外在主体功能是否可用。

 3．灰盒测试：介于二者之间（ 接口测试 ）

4．上述三种方法当中的 "盒" 指的就是被测对象。

二、按测试对象是否执行分类

1．静态测试：指的就是测试不执行。

2．动态测试：将软件运行在真实的使用环境中进行测试。

 三、按测试手段进行分类

1．手工测试：由测试人员手动的对被测对象进行验证，优点就是可以灵活的改变测试操作及环境。

2．自动化测试：所谓自动化主要有二种形，一种是自已写测试脚本，另外一种就是通过第三方的工具对被测对象进行测试。优点就是可以高效率的去执行一些人工无法实现的操作。

**软件质量**

描述当前软件是否好用，在当前的软件行业里我们所采用的一套标准是基于 ISO 组织制定的。需要我们记忆的就是软件质量的六大特性：

1. 功能性：软件需要满足用户显式或者稳式的功能。

2. 易用性：软件易于学习 和上手使用。

3. 可靠性：指的就是软件必须实现需求当中指明的具体功能。

4. 效率性：类似于软件的性能。

5. 可维护性：要求软件具有将某个功能修复之后继续使用的能力。

6. 可移植性：当前软件可以从一个平台移植到另一个平台上去使用的能力。

【 功能靠用，效率可"以" 】

**软件测试流程**

需求分析

(1) 当前阶段的核心目的就是梳理清楚我们需要测试的点是什么。

(2) 需求的来源：需求规格说明书、API 文档、竟品分析、个人经验

设计测试用例

(1) 用例就是用户为了测试软件的某个功能而执行的操作过程。

(2) 设计用例是有方法的（ 等价类、边界值、判定表...... ）

评审用例：对当前的用例进行添加或者删除。

配置环境

(1) 环境：指的就是当前被测对象运行所需要的执行环境，做为测试人员需要具备配环境的能力。【 一般情况下都会使用一键安装的集成环境 】

(2) 环境分类：操作系统 + 服务器软件 + 数据库 + 软件底层代码的执行环境。

执行测试用例

(1) 一般在执行用例之前我们会做一个冒烟测试。 这种测试的核心就是快速的对当前软件的核心功能或者主体执行流程进行验证。如果冒烟测试阶段有问题，则可以将此版本回退给开发。

(2) 如果冒烟测试通过那么才会开展示全面的测试。

回归测试与缺陷报告

(1) 回归测试指的就是当我们将某个缺陷提交给开发之后，由它们进行修复，修复完成之后需要测试认员再次对其进行测试【回归测试】

(2) 缺陷跟踪：指的就是当测试人员发现某个缺陷之后需要一直对其进行状态的跟踪。

输出测试报告

将当前的测试过程中产生的数据进行可视化的输出。方便其它人去查看。

测试结束

当将整个测试过程中产生的一些文档进行整理归档，方便后续版本使用

**软件架构：**

所谓的软件架构我们可以理解为是用来指导我们软件开发的一种思想。 目前来说最常见的二种架构模式就是 B/S C/S

B---browser 浏览器

C---clent 客户端

S---server 服务端

**二种架构的比较**

1．标准：相对于 CS 架构来说 BS 架构的二端都是在使用现成的成熟产品。所以 BS 会显示的标准一些。

2．效率：相对于 BS 架构来说 CS 中的客户端可以分担一些数据的处理，因此执行效率会高一些。

3．安全：B S 架构当中的数据传输都是以 HTTP 协议进行的输出，而 HTTP 协议又是明文输出。可以被抓包，所以相对于 CS 架构来说 BS 就显得不那么安全【相对的】

4．升级： 架构只需要在服务器端将数据进地更新，BS前台只需要刷新页面就可以完成升级，而 CS 架构当中必须要将二端都进行更新。

5．开发成本：相对于 BS 架构来说 CS 当中的客户端需要自已开发，所以相对于来说成本会

**浏览器基本介绍**

一、浏览器是什么

浏览器本质就是一款软件，安装在操作系统之上。一般给用户提供浏览网页的服务。目前来说我们会人为的将所有浏览器总结出一个所谓的五大生产厂商。【 对于浏览器来说最核心技术就是内核 】

二、五大浏览器生产厂商

1．IE( 微软 )-----trident

2．Chrome ( 谷歌 )----blink

3．Firefox( 火狐 )-----gecko

4．Safari( 苹果 )-----webkit

5．Opera( 欧朋 )----presto【 现在已经放弃自已东西完全向 chrome 】

**常见的图片类型**

1. Jpg(jpeg) :这是一种可以高度保留图片色彩信息的格式。

2. Png：该类型的图片可以实现透明。

3. Gif：图片所占体积小，可以实现动图

4. Psd：它是一种分层的图片。

5．开发成本：相对于 BS 架构来说 CS 当中的客户端需要自已开发，所以相对于来说成本会

高一些。