**写出高效的Bug测试报告的9点建议**

**1.清晰地描述Bug：**描述Bug时要用简短的陈述句并能准确指出问题所在。描述中可能需要提供一些步骤来重现这个 Bug，同时这个简短Bug描述必须能够准确地表达出问题的本质所在。例如，假如针对一个来自服务器的错误，Bug描述要对当完成什么操作时，这个服务器错误就会发生做详尽的说明。

**2.不要放过你判断：**虽然你满怀信心地确信你发现的Bug的真实性，但你没写到Bug报告中，这好像代表你正放过这个发现的Bug。很可能会发生一起论战，这将反映出你作为测试人员的优越感。你主要的目标应该是让你的Bug报告令人信服，以支持你发现的Bug，唯一的目的是让Bug 最终关毕。在Bug报告中试着使用外交的表达方式，而不要使用官方的表述来赞成这个Bug，这样你的报告反而会令人不愉快。最好的方法是使用建议的方式。愉快的方式总能被采用。

**3．重现的步骤：**如何利用对条件设置的解释以重现并获得Bug的精确点，这必须要在Bug报告中讲述清楚。例如，对于一个绘图软件，测试人员在找Bug之前，需要和开发人员就他已经做了什么进行交流。细节必须详细说明，像按什么顺序，点击了哪个按钮。对于按照提示输入命令而运行的程序，在测试Bug之前，应该详细地说明输入命令的详细信息。

**4.使用简洁的语言：**人们不喜欢读包含复杂的专业术语和绕口的大段的段落。一个好的Bug报告要包含短的但是表达清晰的语子。它应该只包含与Bug有关的论述。不必要把Bug报告做的过于复杂和写太多事实而篇幅过于长。避免解说过多对重现Bug没有任何帮助的细节。大家都普遍知道的事，就不必写在Bug报告中了。

**5.引用相关的例子：**大部分情况下，要重现一个特殊的Bug，必须输入一些特殊的数据。但是不要做模糊的表述，像提供一个联系表中无效的人名并保存，应该说在名字域中输入像[035bbb@$%](mailto:035bbb@$%25)这样无效的输入并点击保存。为了使Bug能快速得到处理，测试人员必须努力提供所有相关的、关键的信息来帮助开发人员。

**6.提供参考信息：**以防一个特殊的Bug与说明文档或[**其他**](javascript:;)的关于工程的文档相冲突，Bug报告必须得供充分的关于这种特殊情况的参考信息或与文档中相冲突条款的数目。

**7.为Bug分配优先级和严重等级**——没有为Bug设置严重级别和优先级别的Bug报告是不完整的。

　　Bug严重级别：指的是这个Bug破坏系统的危险程度。Bug严重级别说明了这个Bug的破坏程度。严重级别是与Bug紧密联系、永恒不变的一个特性。

　　Bug的严重级别分为四类，下面进行分别描述：

* 严重级别——严重：这是最危险的级别。发现了严重级别的Bug后测试就不允许继续进行了，除特殊点外。弹出一些错误信息或系统瘫痪导致全部或部分的应用被迫关毕这些都属于严重级别的Bug。可以通过判断某个[**工作**](javascript:;)区的不合理性来判断系统的危险级别。如一些菜单项缺失或者未对需要特设的安全许可才能访问的功能设置权限，这类bug都属于致命性的Bug。
* 严重级别——高：高的严重级别指的是导致产品不能按照预期的要求那样运行或者导致一些功能不能正常运行而不能满足客户需求的错误。这种类别的Bug可以通过某种工作区来解决。这种类型的Bug可能就是错误，像[**数据库**](javascript:;)中因为计算或不正确的文件格式导致记录更新失败。像这样的例子还有很多。
* 严重级别——中等：这种类型的缺陷对应用程序的性能没有影响。但是由于没有实现协议上的一些标准或客户的要求，这些缺陷也是不可接受的。因为简单的工作区可以达到性能方面要求的目标，所以中等缺陷相对容易解决。像一些可见的链接未连接到相应的文本网页上，这类缺陷就属于中等缺陷。
* 严重级别——低：低优先级和很小的缺陷属于这类缺陷，这种缺陷不会影响到产品的功能。严重级别为低的这种缺陷一般不会发生在应用的常用模块中，对业务有极小的影响。这种缺陷一般是用户界面、装点方面的美观问题。

Bug的优先级：指的是Bug 要求被解决的紧急程度。它描述了Bug的重要性。Bug的优先级别可能会根据测试的日程安排而改变。一共有三个优先级别，如下：

* 高优先级：如果这类的缺陷不立即修正，将会影响客户终端常用功能的使用。因此这类缺陷要给以最高的优先级以对其立即处理。
* 中优先级：如果这类缺陷对用户常用的功能有较大的影响，那么这类的缺陷就要设置为中优先级。这类缺陷被分配高的优先级，所以要在当前软件版本发布之前解决这些相关的问题。如果由于时间方面的限制而无法解决这类问题，那么针对这类问题的补丁或服务包必须要发布。
* 低优先级：对客户端软件的性能没有大的影响的缺陷一般被认定为低优先级的缺陷。在当前版本发布之前努力去修正他们，如果由于时间的限制，无法修正，可以等到在下个版本中修正。

**8.通过抓屏截图来解释**——正如谚语说的“一图胜千言”。当我们发现一个错误，最好对这个错误进行抓屏截图。如果错误是可见的，抓屏将帮助开发者准确地理解这个问题。这个阶段开发者应该首先集中集力清晰理解这个问题，而不用试着去解决这个问题。

　　这样的抓屏应该做为证剧附在Bug报告中。这样测试人员就可以很好地更清晰地和开发人员交流解释这个Bug。

**9.时刻准备着向开发人员展示你发现的Bug：**Bug报告中最有趣的部分是，软件测试人员需要时刻准备着向开发人员展示他发现的Bug，同时需要说服开发者，报告中的Bug都是真实存在的且需要解决的，因为它们将影响应用的性能。在这个过程中，软件测试工程师一定要为下面的几种情况做好准备：

　　（1）开发者通常会说某个特定Bug不能重现。报告这个Bug的最好方法是向开发者展示它。可以请开发者看看在你计算机上运行的情景，加载应用，为这个问题提供真实的证明。这样他们可以真实地看到并了解你是如何操作应用的，如何和应用交互的，软件对提供的输入有怎样的反应。应该避免在最短的时间内一腔热情地报告大量的不能重现的Bug。

　　（2）有时软件测试人员会在特定的环境中发现偶尔才会发生的缺陷。当压力达到最大极限，处于测试中的应用处于崩溃状态时才会出现这种缺陷。当软件测试人员向开发者展示同样情景以证明这种缺陷时，这时应用程序又运行正常了，这让测试人员很觉尴尬。因些一个好的测试人员要有耐心，同时要保留测试数据和抓屏等信息，为证明自己的观点建立一个可防御的机制。

　　（3）如果测试人员提供给开发人员一个包含各种操作和输入数据等的表，当开发人员按表中的说明在他的系统上运行时，并没有发现任何错误。这说明测试人员没有提供足够的信息。开发者所用的系统和测试人员的系统有可能在配置上会有所不同，这个会导致一些错误并不能在开发者的计算机系统上出现。测试人员也可能没有完全理解程序的需求，测试人员和开发人员看的是同样的界面，但却有着不同的看法。对于同一个测试结果，测试人员认为它是错误的但开发人员却认为它是正确的，这种情况也可能会发生。所以为了避免这种情况，最好从你认为的需求是什么，你真正看到了什么,发生了什么来支持你的观点。