

c开发策略-之-错误处理 PDF转换可能丢失图片或格式，建议  
阅读原文

[https://www.100test.com/kao\\_ti2020/246/2021\\_2022\\_c\\_E5\\_BC\\_80\\_E5\\_8F\\_91\\_E7\\_AD\\_96\\_c97\\_246243.htm](https://www.100test.com/kao_ti2020/246/2021_2022_c_E5_BC_80_E5_8F_91_E7_AD_96_c97_246243.htm) 在使用任何语言进行应用程序开发时，我们都应该提前规划好如何处理错误。Java和c中普遍使用异常来进行错误处理，但是c语言，因为没有提供一个很优雅的正常机制，所以明确如何进行错误处理显得很重要。C语言中的错误处理有多种方式，总结如下：大家可以讨论这些处理方式的优劣，这样等以后在程序开发中，我们可以从整体上为程序设计更好的错误处理方法。

1. 返回值方式：用函数的返回值标志函数是否执行成功。比如成功返回1，失败返回0。这种方式的好处是简单方便，而且不影响效率，保持了c语言的高效率。但是仍然有问题，一个问题是代码可读性的问题，如果每个函数都有这样的返回值的话，为了保持程序的正确运行，我们必须对每个函数进行正确性验证，就是在调用函数的时候检查他的返回值，这样程序代码很大一部分就可能花费在错误处理上。第二个问题就是函数的返回值冲突的问题。假设strlen函数也可能会出错，使用这种错误处理策略他的返回值应该标志它是否执行成功，但是函数计算的字符串的长度值如何自然地传递出来？最后一个问题可能是最重要的：它不强制你处理错误，而且在不进行处理的情况下，程序仍然能够运行，但结果是不可预知的。
2. 全局errno方式：就是在出现错误的时候，将错误代码记录到一个全局变量errno中。比如waitpid()函数在被信号中断的情况下，将errno设置为EINTR（一宏定义常量）。这种方式解决了返回值方式遇到的返回值冲突问题，而且

效率方面也是非常令人愉悦的。但是它要求用户在调用函数后检查errno的值，这种保证是脆弱的，程序仍然有可能在不处理那些errno的情况下“安然”地运行，导致未定义的结果。另一个问题出在多线程方面，errno不是线程安全的，多个线程操作同一个errno会造成混乱。

3. 错误封装：就是将每个有错误返回值的函数分别用一个函数包起来，比如waitpid()函数可以封装成Waitpid()（首字母大写），在这个函数中处理相应的错误。这种错误处理方法可以很好的解决很多问题，应该说效果很好，但是有几个方面需要商榷，一是，并不是每个函数的错误都以一种方式进行处理，另一方面，听说c语言的函数调用开销相对很高，在函数外面再包上一层会影响性能。

4. 异常：关于异常的说明和实现可以参考它的优点是能模拟实现c中异常的一些优点。但是这个异常机制很脆弱，使用时要注意很多问题，而且它的性能开销肯定也会不小。

5. Goto语句：当发生错误时，利用goto语句跳到相应的错误处理函数中。因为一直以来对goto语句的偏见，和goto语句本身对程序结构性的影响，所以本人一直以来没有用过这种方式，也不知道这种方式会有什么优劣。总的来说，每个方式都不是尽善尽美的，不知道大家遇到这些问题是怎么处理的？

100Test 下载频道开通，各类考试题目直接下载。详细请访问 [www.100test.com](http://www.100test.com)