如何寫出無法維護的代碼

2011 年 6 月 3 日陳皓發表評論閱讀評論 68,975 人閱讀

酷殼裡有很多我覺得很不錯的文章，但是訪問量最大的卻是那篇《6 個變態的 Hello World》，和它能在本站右邊欄“ 全站熱門 ”中出現的還有“ 如何加密源代碼 ”，以及編程真難啊等這樣的文章。可見本站的讀者們的偏好，我也相信你們都是“身懷絕技”的程序員。所以，今天給大家推薦這篇文章，相信一定能觸動大家的興奮點。

這篇文章的原文在這裡（http://mindprod.com/jgloss/unmain.html），我看完後我想說——

1. 什麼叫“創造力”，創造力就是——就算是要乾一件爛事都能幹得那麼漂亮那麼有創意的能力。
2. 什麼叫“抓狂”，抓狂就是——以一種沉著老練的不屈不撓的一本正經的精神一點一點把你推向崩潰的邊緣。

我把文章節選了一些，也並沒有完全翻譯，簡譯一下，也加入了一些自己的調侃。對於有下面這些編程習慣的朋友，請大家對號入座。另外，維護程序的朋友們，你們死定了！！

If builders built buildings the way programmers write programs, then the first woodpecker that came along would destroy civilization. （如果建築師蓋房子就像程序員寫程序一樣，那麼，第一隻到來的啄木鳥就能毀掉我們的文明）

~ Gerald Weinberg (born: 1933-10-27 age: 77) Weinberg's Second Law

程序命名

- 容易輸入的名字。比如：Fred，asdf
- 單字母的變量名。比如：a,b,c，x,y,z（陳皓注：如果不夠用，可以考慮 a1,a2,a3,a4,….）
- 有創意地拼寫錯誤。比如：SetPintleOpening， SetPintalClosing。這樣可以讓人很難搜索代碼。
- 抽象。比如：ProcessData，DoIt，GetData…抽像到就跟什麼都沒說一樣。
- 縮寫。比如：WTF，RTFSC …… （陳皓注：使用拼音縮寫也同樣給力，比如： BT，TMD，TJJTDS）
- 隨機大寫字母。比如：gEtnuMbER..
- 重用命名。在內嵌的語句塊中使用相同的變量名有奇效。
- 使用重音字母。比如：int ínt（注：第二個 ínt 不是 int）
- 使用下劃線。比如：_，__，___。
- 使用不同的語言。比如混用英語，德語，或是中文拼音。
- 使用字符命名。比如：slash，asterix，comma…
- 使用無關的單詞。比如：god，superman，iloveu….
- 混淆 1 和 l。字母 l 和數字 1 有時候是看不出來的。

偽裝欺詐

- 把註釋和代碼交織在一起。

```
for (j=0; j<array_len; j+ =8)
{
        total += array[j+0 ];
        total += array[j+1 ];
        total += array[j+2 ]; /* Main body
of
        total += array[j+3]; * loop is
unrolled
```

```
        total += array[j+4]; * for greater
speed.
        total += array[j+5]; */
        total += array[j+6 ];
        total += array[j+7 ];
}
```

- 隱藏宏定義。如：#define a=ba=0-b，當人們看到 a=b 時，誰也想不到那是一個宏。
- 換行。如下所示，下面的示例使用搜索 xy_z 變得困難。

```
#define local_var
xy\
_z // local_var OK
```

- 代碼和顯示不一致。比如，你的界面顯示叫 postal code，但是代碼裡確叫 zipcode.
- 隱藏全局變量。把使用全局變量以函數參數的方式傳遞給函數，這樣可以讓人覺得那 個變量不是全局變量。
- 使用同意詞。如：

```
#define xxx global_var // in file std.h
#define xy_z xxx // in file ..\other\substd.h
#define local_var xy_z // in file
..\codestd\inst.h
```

- 使用相似的變量名。如：單詞相似，swimmer 和 swimner，字母相似：i1I1|或 o008。parselnt 和 parseInt， DOCalc 和 DOCalc。還有這一組：xy_Z，xy__z，_xy_z，_xyz，XY_Z，xY_z，Xy_z。
- 重載函數。使用相同的函數名，但是其功能和具體實現完全沒有關係。
- 操作符重載。重載操作符可以讓你的代碼變得詭異，感謝 CCTV，感謝 C++。這個東西是可以把混亂代碼提高到一種藝術的形式。比如：重載一個類的!操作符，但實際功能並不是取反，讓其返回一個整數。於是，如果你使用！!操作符，那麼，有意思的事就發生了——先是調用類的重載!操作符，然後把其返回的整數給!成了布爾變量，如果是!!!呢？呵呵。
- #define。看過本站那些混亂代碼的文章，你都會知道宏定義和預編譯對於寫出不可讀的代碼的重大意義。不過，一個具有想像力的東西是——在頭文件中使用預編譯來查看這個頭文件被 include 了幾次，而被 include 不同的次數時，其中的函數定義完全不一樣。

```
#ifndef DONE
#ifdef TWICE
// put stuff here to declare 3rd time
around
void g( char * str);
#define DONE
#else // TWICE
#ifdef ONCE
// put stuff here to declare 2nd time
around<
void g( void * str);
#define TWICE
#else // ONCE
```

```
// put stuff here to declare 1st time
around
void g(std::string str);
#define ONCE
#endif // ONCE
#endif // TWICE
#endif // DONE
```

文檔和註釋

- 在註釋中撒謊。你不用真的去撒謊，只需在改代碼的時候不要更新註釋就可以了。
- 註釋明顯的東西。比如：`/* add 1 to i */`。（參看本站的“ 五種應該避免的註釋 ”）
- 只註釋是什麼，而不是為什麼。
- 不要註釋秘密。如果你開發一個航班系統，請你一定要保證每有一個新的航班被加入，就得要修改 25 個以上的位置的程序。千萬別把這個事寫在文檔中。
- 注重細節。當你設計一個很複雜的算法的時候，你一定要把所有的詳細細設計都寫下來，沒有 100 頁不能罷休，段落要有 5 級以上，段落編號要有 500 個以上，例如：`1.2.4.6 .3.13 — Display all impacts for activity where selected mitigations can apply (short pseudocode omitted).`這樣，當你寫代碼的時候，你就可以讓你的代碼和文