

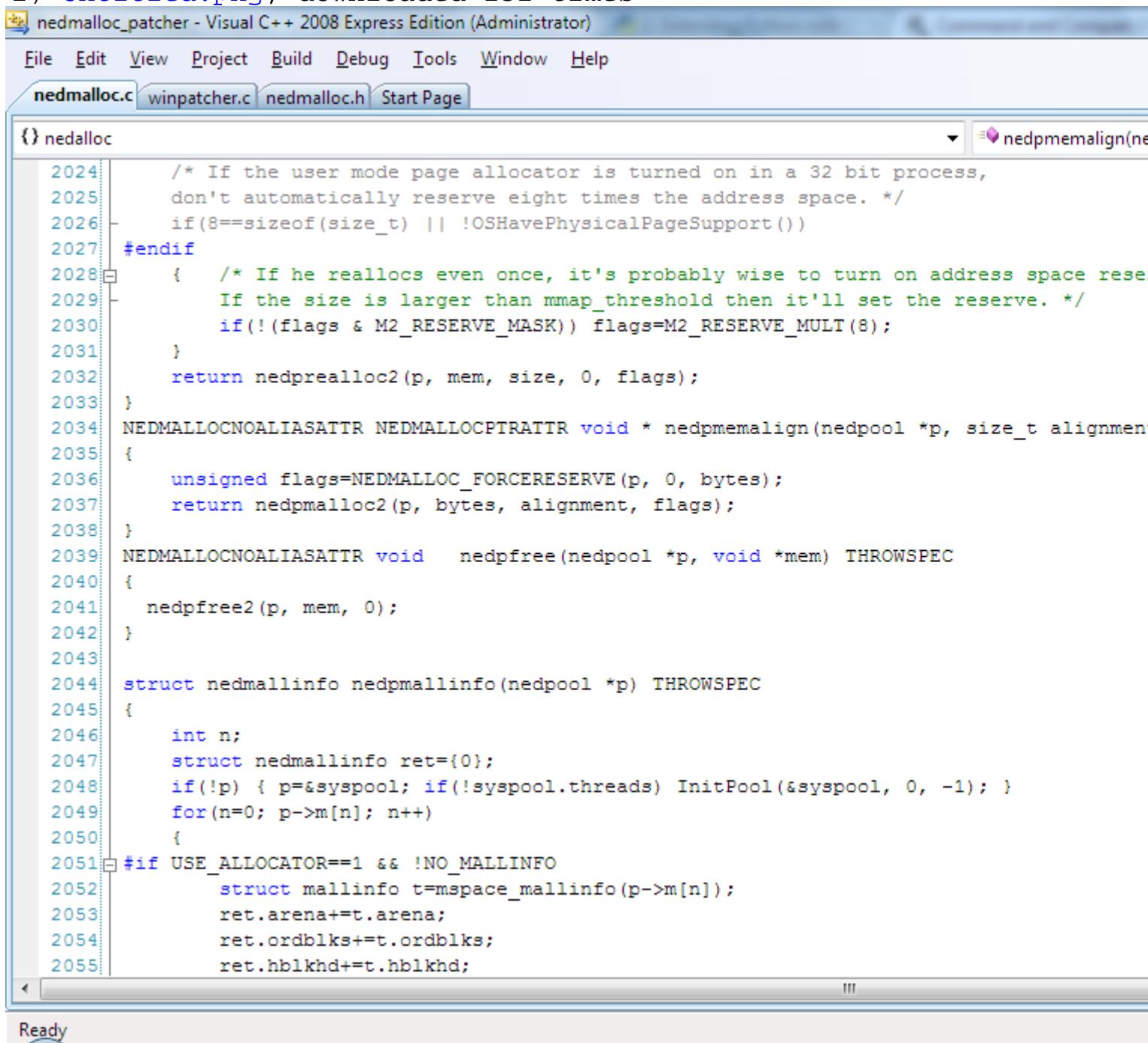
Subject: Re: A notice to anyone planning to write code for scripts 4.0
Posted by iRANian on Wed, 31 Aug 2011 15:52:48 GMT

[View Forum Message](#) <> [Reply to Message](#)

Neh, I mean like this:

File Attachments

1) [Untitled.png](#), downloaded 231 times



The screenshot shows a Windows desktop environment with the Visual Studio 2008 Express Edition interface. The title bar reads "nedmalloc_patcher - Visual C++ 2008 Express Edition (Administrator)". The menu bar includes File, Edit, View, Project, Build, Debug, Tools, Window, and Help. Below the menu is a tab bar with "nedmalloc.c" (selected), "winpatcher.c", "nedmalloc.h", and "Start Page". The main code editor area displays the "nedmalloc.c" file content. The code is written in C and includes several conditional compilation directives (#ifdef, #ifndef, #endif). It defines functions for memory allocation and deallocation, such as "nedpmemalign" and "nedpfree". The code also includes comments explaining the logic for handling different memory allocation scenarios. The code editor has line numbers on the left and syntax highlighting for C code.

```
2024:     /* If the user mode page allocator is turned on in a 32 bit process,
2025:      don't automatically reserve eight times the address space. */
2026:      if(8==sizeof(size_t) || !OSHavePhysicalPageSupport())
2027:      #endif
2028:      { /* If he reallocs even once, it's probably wise to turn on address space reserve.
2029:         If the size is larger than mmap_threshold then it'll set the reserve. */
2030:         if(!(flags & M2_RESERVED_MASK)) flags=M2_RESERVED_MULT(8);
2031:      }
2032:      return nedprealloc2(p, mem, size, 0, flags);
2033: }
2034: NEDMALLOCNOALIASATTR NEDMALLOCPTRATTR void * nedpmemalign(nedpool *p, size_t alignment, size_t bytes)
2035: {
2036:     unsigned flags=NEDMALLOC_FORCERESERVE(p, 0, bytes);
2037:     return nedpmalloc2(p, bytes, alignment, flags);
2038: }
2039: NEDMALLOCNOALIASATTR void nedpfree(nedpool *p, void *mem) THROWSPEC
2040: {
2041:     nedpfree2(p, mem, 0);
2042: }
2043:
2044: struct nedmallinfo nedpmallinfo(nedpool *p) THROWSPEC
2045: {
2046:     int n;
2047:     struct nedmallinfo ret={0};
2048:     if(!p) { p=&syspool; if(!syspool.threads) InitPool(&syspool, 0, -1); }
2049:     for(n=0; p->m[n]; n++)
2050:     {
2051: #if USE_ALLOCATOR==1 && !NO_MALLINFO
2052:         struct mallinfo t=mspace_mallinfo(p->m[n]);
2053:         ret.arena+=t.arena;
2054:         ret.ordblks+=t.ordblks;
2055:         ret.hblkhd+=t.hblkhd;
2056:     }
2057: }
```

Ready