Subject: Re: C++ FQA
Posted by mirek on Thu, 08 Nov 2007 04:34:24 GMT
View Forum Message <> Reply to Message

Mindtraveller wrote on Wed, 07 November 2007 18:13cbpporterThey are by far not uncontrolable, and only happen in some apps.OK, as a developer who likes D programming paradigm, and who wants writing efficient applications, I will consider changing language to D if
1) There`s a simple way to guarantee no hangups at all.
2) My .exe will be <10%-15% slower than corresponding C++ code in any case.

3) Add to mix memory requirements. In that "Alice" benchmark, D used about 2-3 times more memory than U++ (if I remember well).

BTW, it is some time when I last tried it. I would really be very glad if somebody reproduced my results; here is U++ code:

```
#include <Core/Core.h>

using namespace Upp;

#define BENCHMARK // for benchmark purposes, output is omitted
#ifdef  BENCHMARK
#define BENCHBEG for(int i = 0; i < 1000; i++) {
#define BENCHEND }
#else
#define BENCHBEG
#define BENCHEND
#endif


void main(int argc, const char *argv[])
{
 VectorMap<String, int> map;
 BENCHBEG
 for(int i = 1; i < argc; i++) {
  String f = LoadFile(argv[i]);
  int line = 1;
  const char *q = f;
  for(;;) {
   int c = *q;
   if(IsAlpha(c)) {
    const char *b = q++;
    while(IsAlNum(*q)) q++;
    map.GetAdd(String(b, q), 0)++;
   }
   else {
    if(!c) break;
    if(c == '\n')
     ++line;
    q++;
   }
  }
 }
 BENCHEND
 Vector<int> order = GetSortOrder(map.GetKeys());
#ifndef BENCHMARK
 for(int i = 0; i < order.GetCount(); i++)
  Cout() << map.GetKey(order[i]) << ": " << map[order[i]] << '\n';
#endif
 printf("%d\n", map.GetCount());
}
```

(I had to loop over it more times, otherwise the execution for Alice.txt was too fast to be measurable).

Quote:
P.S. luzr, I just thought that D and U++ comparison is not quite honest. It would be better to compare internal language features of D and C++. According to the tests described in D site (I looked at them as you recommended), D is no slower than C++ (at least in some cases?). So porting the U++ classes and algorithms to D, adopting them for D specifics could make U+D (U++ for D) as fast as original U++. It`s just a theory, of course.

I think this is not quite possible, as language features are way too different.

OTOH, in this particular benchmark, D is using internal language features, while C++/U++ is using library. Still, we are faster and the code is shorter. That IMO says a lot about language flexibility.

Mirek