Subject: Vector::SetCount()
Posted by mrjt on Wed, 12 Nov 2008 12:38:16 GMT
View Forum Message <> Reply to Message

Until very recently I hadn't realised that SetCount actually deallocates memory when the count is less than the allocation. This means you can't use a Vector for an iteration of an algorithm, then reuse the memory for the next iteration without deallocating/reallocating it.

This isn't a major issue in itself as I can always hack it in myself if necessary, but there is currently no mention of the deallocation in the documentation. This is particularly misleading because the allocation of extra space is specifically mentioned:
Quote:Changes count of elements in Vector to specified value. If required number of elements is greater than actual number, newly added elements are default constructed. If Vector has to increase capacity, the new capacity will exactly match required number of elements (unlike SetCountR).
Perhaps it could be changed? And would it be possible to add a function that allows setting of items without deallocation?

Subject: Re: Vector::SetCount()
Posted by mirek on Wed, 12 Nov 2008 21:09:39 GMT
View Forum Message <> Reply to Message

Are you sure? Do not see anything like that in the code:

```
template <class T>
void Vector<T>::Trim(int n)
{
 ASSERT(n >= 0 && n <= items);
 DestroyArray(vector + n, vector + items);
 items = n;
}

template <class T>
void Vector<T>::SetCount(int n) {
 Chk();
 ASSERT(n >= 0);
 if(n == items) return;
 if(n < items)
  Trim(n);
 else {
  if(n > alloc) ReAllocF(n);
  ConstructArray(vector + items, vector + n);
  items = n;
 }
}
```

Subject: Re: Vector::SetCount()
Posted by mrjt on Thu, 13 Nov 2008 12:42:55 GMT
View Forum Message <> Reply to Message

Apologies, I had assumed DestroyArray was doing deallocation without checking. Foolish mistake.