

Re: alloca / _alloca / dynamic stack memory

Source: <http://www.tech-archive.net/Archive/VC/microsoft.public.vc.language/2007-03/msg00243.html>

- *From:* "Doug Harrison [MVP]" <dsh@xxxxxxx>
 - *Date:* Thu, 08 Mar 2007 22:42:02 -0600
-

On Thu, 8 Mar 2007 20:06:13 -0800, Michael Crawley
<MichaelCrawley@xxxxxxxxxxxxxxxxxxxxxxxxxxxx> wrote:

"Doug Harrison [MVP]" wrote:

On Thu, 8 Mar 2007 12:57:33 -0800, Michael Crawley
<MichaelCrawley@xxxxxxxxxxxxxxxxxxxxxxxxxxxx> wrote:

```
class _StackAlloc{ };
const _StackAlloc StackAlloc; // Used only to resolve
overloaded
method/operator

void* Object::operator new(size_t size, const _StackAlloc&
stackAlloc)
{
// Allocation on stack
// ...

// Return Pointer to allocated memory
__asm push pointer onto stack;
__asm jmp (return address); // Jump out of the current stack
frame
};

// Calls destructor, but does not free memory until the
// current function returns
void Object::Dispose(const StackAlloc& stackAlloc)
{
// Invoke Distructor
this->~Object(); // Virtual Base destructor

// Perform other operations specific to this
// memory allocation method
// ...
};
```

Re: alloca / _alloca / dynamic stack memory

```
class String : public Object
{
public:
// ...
private:
~String(){...}; // Means this type can only be on allocated on
dynamic
memory
};

int main()
{
String str("bad"); // Compiler error, cannot access private
distructor
String* str1 = new(Heap) String("better"); // Ok
String* str2 = new String("better"); // Ok
String* str3 = new(StackAlloc) String("better"); // OK, faster
than
str1/str2
```

What does String have for data?

Huh...? Perhaps a char* pointer and a length field. Perhaps String is not a good example, but I just wanted show how we would like to use the class... A variable-sized array would be more appropriate...

Assume you have your base class, and it does everything you want. Maybe I missed it, but I think it would help to show how String exploits what Object gives it. Also, are there any restrictions on String when used as a class member or the basis of an array type?

—
Doug Harrison
Visual C++ MVP