Subject: Re: ArrayCtrl Write To Freed Memory Detected
Posted by r1kon on Tue, 13 Dec 2011 21:29:29 GMT

That's kind of what I figured..I've got GuiLock __; at the top of the workhorse thread that receives all of the data and does MOST of the setting. However, there are still other threads that deal with it.

Here's something that's been bothering me though -- do I need to set GuiLock __; on those that are writing to the array control ONLY?? Or do I need to do it for those that are READING from the array control as well?

The other thing is I set up GuiLock __; on every write to the GUI, and it made my program verrrrryyyy slow to respond (but never crashed! lol). I think it may be due to the fact that I have some indefinite loops and the GuiLock's may be waiting for one another to complete, thus slowing everything down quite a bit.

So, I took them all out but that one (in the workhorse thread)..so it sounds like I need to put them all back, but in a "cleaner" manor.

My question now is..what is the best way to do that? Is it ONLY set to lock within the confines of the nest? For example:

```
for(int i = 0; i < 1000; i++)
{

  for(int z = 0; z < 1000; z++)
  {
    Sleep(1);
    if(something > something_else)
    {
      GuiLock __;
      myArr.Set(z,0,t_("Data to be written"));
    }
    something++;
  }
}
```

Does GuiLock in this case set and then released at the end of the if() statement? Or does it stay set through the entire for() loop? So, the same code snippit..which place would be the best to put GuiLock __;?

```
//HERE? GuiLock __;
for(int i = 0; i < 1000; i++)
{
  //HERE? GuiLock __;
  for(int z = 0; z < 1000; z++)
  {
```

```
  Sleep(1);
  if(something > something_else)
  {
    //Or leave it here? GuiLock __;
    myArr.Set(z,0,t_("Data to be written"));
  }
  something++;
 }
}
```

I hope I'm not being annoying in asking -- I just need to make my multi-thread application as streamlined as possible without it killing the GUI responce.

Thanks in advance..I really appreciate it!

-Kevin