

Re: Simple question on memory behavior

Source:

<http://www.tech-archive.net/Archive/DotNet/microsoft.public.dotnet.framework.clr/2004-05/0110.html>

From: Brian Tyler (*brianmtyler_at_yahoo.com*)

Date: 05/10/04

Date: Mon, 10 May 2004 15:12:18 -0500

1. (a) To be specific, the object pointed to by tmpPage can be collected after line 5, but that object points to another object (PageName). That has a second (at least) reference to it via IStickAround.CurrentPageName and thus the string would not be collected.

As a side note: Strings are a special case because they are immutable and the CLR takes advantage of this fact. It keeps a "global string repository", so code such as:

```
Foo a = new Foo();  
Foo b = new Foo();  
a.MyName = "abc";  
b.MyName = "abc";
```

actually results in just 3 objects. Even though "abc" seems like two string objects, a.MyName ends up pointing to the same memory as b.MyName.

2. Not sure I fully understand the question. If PageItem has properties, then they must be individually assigned. CopyTo() is typically an array copy method provided by objects, but that isn't the same as properties...

"Tyson Brown" <tyson.brown.REMOVE@sbcglobal.net> wrote in message news:D4A6578A-EA9F-4C59-B30C-7C08AFAC0132@microsoft.com...

> *Mattias,*

>

> *Thanks for responding.*

>

> >>> a) *If so, does this mean that tmpPage object lasts the duration of the page too, regardless of Line5?*

>

> >> *No.*

>

> *Could you explain this? I had always thought that assigned on ref type to another variable or property, would keep the original object around because the GC says, "sorry there is still information in use, I can't clean you up, try Gen 1 (or God forbid, Gen 2)". Is this wrong? By marking the object as null, even though the value still exists in memory through another pointer,*

the original object is indeed collected?

>

> *I apologize for the ignorance, I am just really trying to get my brain
around this and understand GC.*

>

> *Thanks,*

>

> *Tyson*