

Re: Circular references not possible?

Source:

<http://www.tech-archive.net/Archive/DotNet/microsoft.public.dotnet.languages.vb/2004-03/2750.html>

From: Jay B. Harlow [MVP – Outlook] (Jay_Harlow_MVP_at_msn.com)

Date: 03/12/04

Date: Fri, 12 Mar 2004 17:17:26 -0600

Raghu

> *Idea of separating interfaces and having the rest of the app depend
> only on the interfaces is the reason I've put all my interfaces in a
> separate project and localized the object creation to the factory.
> From what you stated earlier you put all your CLASSES in a separate project
and localized the object creation to the factory. I did not see where you
used INTERFACES...*

Remember that an Interface is not a Class they are both distinct constructs
in .NET. An Interface only has the definition of Members ever, while a Class
has the implementation of the members. Of course an abstract class can have
the definition of members (see the MustInherit & MustOverride keywords)...

> – *Also, was quite dissappointed that the System.Configuration classes
> do not allow section entries to be placed in a separate file :(*
Yes they do ;-) even without the Configuration Management application
block...

Look at the NameValueFileSectionHandler, which of course is labled as "not
intended to be used directly from your code", although I remember reading
one or two articles on using it, possible the links I gave.

Of course you could always create your own SectionHandler that does support
loading entries from a separate file! If you have enough apps that need the
custom section, you can always put it in the machine.config file instead of
the app.config file.

Hope this helps

Jay

"R. Raghuraman" <raghuraman@infosys.com> wrote in message
news:a72db9f8.0403121112.33d60689@posting.google.com...

> *Jay,*

>

> *Am staying with the solution using reflection. I do see your point
> that reflection is limited to loading the type but then on it's
> straight no-reflection code (and the idea does comfort me) – It was*

> *just that initially I was queasy about using reflection.*
>
> *Thought you'd be interested in some more of it...so here goes:*
>
> *Idea of separating interfaces and having the rest of the app depend*
> *only on the interfaces is the reason I've put all my interfaces in a*
> *separate project and localized the object creation to the factory.*
>
> *As you've mentioned I too wanted to keep the configuration section*
> *separate. In addition, I will have multiple executables that need to*
> *use the class factory so I did not want for each executable's config*
> *file to have all the settings – even if it's a separate section.*
>
> *– Also, was quite dissappointed that the System.Configuration classes*
> *do not allow section entries to be placed in a separate file :(*
>
> *So, I'm using Configuration Management application block. That allows*
> *me to put all the configuration settings for the class factory in a*
> *separate configuration file with the app's configuration file pointing*
> *to the class factory's configuration file.*
>
> *And coming to the book, I'm running right now to get a copy of it :)*
> *Cheers,*
> *Raghu*
>
>
>
> *"Jay B. Harlow [MVP – Outlook]" <Jay_Harlow_MVP@msn.com> wrote in message*
news:<Opts8EECEHA.464@TK2MSFTNGP11.phx.gbl>...
> > *Raghu,*
> > *Note I normally put the IDBWrapper & Employee setting in a custom*
> > *configuration section & not appSettings, I am only using appSettings in*
this
> > *example. For example a custom DictionarySectionHandler section that you*
> > *change "key" to "domainObject" and "value" to "type".*
> >
> > *<configuration>*
> >
> > *<configSections>*
> > *<section name="domainObjects"*
> > *type="Project1.DomainObjectSectionHandler, Project1" />*
> > *</configSections>*
> >
> > *<appSettings>*
> > *<add key="key1" value="value1" />*
> > *</appSettings>*
> >
> > *<domainObjects>*
> > *<add domainObject="Employee" type="Project2.Employee, Project2"*
/>
> > *</domainObjects>*

microsoft.public.dotnet.languages.vb: Re: Circular references not possible?

>>

>> </configuration>

>>

>>

>> *See the following on how to create new sections via the configSections*

>> *section.*

>>

>>

<http://msdn.microsoft.com/library/default.asp?url=/library/en-us/cpguide/html/cpconconfigurationsectionhandlers.asp>

>>

>> *and:*

>>

<http://msdn.microsoft.com/library/default.asp?url=/library/en-us/cpgenref/html/gngrfconfigurationsectionsschema.asp>

>>

>> *Also read about the System.Configuration.ConfigurationSettings class and*

>> *other classes in the System.Configuration namespace.*

>>

>> *Hope this helps*

>> *Jay*

>>

>> *"Jay B. Harlow [MVP – Outlook]" <Jay_Harlow_MVP@msn.com> wrote in message*

>> *news:%23ngXI7DCEHA.3784@TK2MSFTNGP10.phx.gbl...*

>>> *Raghu*

>>> *I've reverted to using reflection but am not really happy about it.*

Do

>>>> *let me know if you've come up with something*

>>>> *As I stated use the Separated Interface pattern!*

>>>>

>>>> <http://www.martinfowler.com/eaCatalog/separatedInterface.html>

>>>>

>>>> *Remember that your ClassFactory will effectively be implementing the*

>>>> *Plugin*

>>>> *pattern to dynamically load types.*

>>>>

>>>> <http://www.martinfowler.com/eaCatalog/plugin.html>

>>>>

>>>> *Note you may need to read the entire book "Patterns Of Enterprise*

>>>> *Application Architecture" by Martin Fowler from Addison–Wesley rather*

then

>>>> *the above pages to get the full "gist" of what the above patterns are*

>>>> *doing*

>>>> *for you. As the book goes into further details & examples.*

>>>>

>>>> <http://www.martinfowler.com/books.html#ea>

>>>>

>>>>> *Thanks for the response, but that does not solve the problem*

(atleast

>>>>> *without using reflection)*

>>>>> *Yes! it does solve the problem & reflection is not involved per se as*

you

Re: Circular references not possible?

microsoft.public.dotnet.languages.vb: Re: Circular references not possible?

> > > are early binding to the interface. Some may consider reflection is
> > involved
> > > as you are dynamically loading the class, however you are only
dynamically
> > > loading the class, you are still early binding to the interface. Hence
no
> > > reflection is involved in execution.
> > >
> > > > You see, Project1 will still need to refer to Project2 because even
> > > > though it returns reference to the interface, it still needs to call
> > > > teh ctor of the concrete classes that are defined in Project2
> > > The Activator.CreateInstance will call the ctor of the concrete class,
the
> > > name of the concrete class will be in the app.config. You will early
bind
> > to
> > > the interface!
> > >
> > > ' app.config file
> > > <configuration>
> > > <appSettings>
> > > <add key="IDBWrapper" value="Project2.DBWrapper, Project2" />
> > > </appSettings>
> > > </configuration>
> > >
> > >
> > > ' various files in the ClassFactory project
> > >
> > > Public Interface IDBWrapper
> > > Sub Method1()
> > > Sub Method2()
> > > End Interface
> > >
> > > > public shared function createDBWrapper as IDBWrapper
> > > Dim typeName As String
> > > typeName =
> > > Configuration.ConfigurationSettings.AppSettings("IDBWrapper")
> > > Dim t As Type
> > > t = Type.GetType(typeName)
> > > Dim value As Object
> > > value = Activator.CreateInstance(t)
> > > Return DirectCast(value, IDBWrapper)
> > > > end function
> > >
> > > ' Then in some project that references the Class Factory Project
> > > Option Strict On
> > >
> > > Dim dbw As IDBWrapper
> > > dbw = ClassFactory.CreateDBWrapper()
> > > dbw.Method1()
> > > dbw.Method2()

Re: Circular references not possible?

> > >
> > > *As the name Separated Interface suggests, the interface to your DBWrapper*
> > *is*
> > > *separated from the implementation, two major benefits include (but are not*
> > > *limited to):*
> > >
> > > *1. You can easily change the implementation without recompiling your app.*
> > > *You could include any of the following lines you your app config & your*
> > *app*
> > > *is suddenly using SQL Server, Odbc, Ole DB, or Oracle as the database.*
> > *(The*
> > > *plug in pattern)*
> > > *<add key="IDBWrapper" value="Project2.SqlDBWrapper, Project2"*
> > */>*
> > > *<add key="IDBWrapper" value="Project2.OdbcDBWrapper, Project2"*
> > */>*
> > > *<add key="IDBWrapper" value="Project2.OleDbDBWrapper, Project2" />*
> > > *<add key="IDBWrapper" value="Project3.OracleDBWrapper, Project3"*
> > */>*
> > >
> > > *2. You can have circular references between assemblies without resorting*
> > *to*
> > > *Reflection or Late Binding.*
> > >
> > > *The key is that your app only knows about the Interface & uses the*
> > *Interface*
> > > *for early binding, meanwhile back in the liar (I mean Class Factory)*
> > *you*
> > *use*
> > > *Activator.CreateInstance to dynamically load the concrete class that*
> > *offers*
> > > *the implementation for the interface.*
> > >
> > > *Remember that using an Interface (Public Interface MyInterface) allows*
> > *you*
> > > *to have Early binding while keeping the Implementation (Public Class*
> > > *MyClass) hidden out of the way.*
> > >
> > > *Hope this helps*
> > > *Jay*