

Re: Abstract class or interface?

Source:

<http://www.tech-archive.net/Archive/DotNet/microsoft.public.dotnet.languages.csharp/2005-05/msg00445.html>

- *From:* "Brett" <no@xxxxxxxx>
 - *Date:* Tue, 3 May 2005 07:10:50 -0400
-

"ozbear" <ozbear@xxxxxxxx> wrote in message

news:4277546e.81458468@xxxxxxxxxxxxxxxx

> On Sun, 1 May 2005 19:35:19 -0400, "Brett" <no@xxxxxxxx> wrote:

>

>>I'm still trying to figure out concrete reasons to use one over the other.

>>I understand the abstract class can have implementation in its methods and

>>derived classes can only inherit one abstract class. The interface has

>>implied abstract methods/properties and derived classes can inherit

>>multiple

>>interfaces. The interface properties/methods have no implementation.

>>

>>Besides definitions of the two, what are some conceptual reasons to use

>>one

>>over the other? Perhaps examples of this project uses an abstract classes

>>vs. this one uses an interface...for these reasons.

>>

>>I have one project that uses an interface. I chose the interface over an

>>abstract class because the derived classes all do the same thing, they

>>just

>>go about doing it in slightly different ways. Their results are different

>>but conceptually what they do is the same thing. I don't ever see the

>>need

>>for adding more methods or properties to the interface. If there is every

>>such a need, I can encapsulate this variability into one of the derived

>>classes, since it will be "one" that has such a need rather than all.

>>That

>>make sense?

>>

>>Thanks,

>>Brett

>

> I have a 3-D package that under user control needs to be able to move

> and rotate "things" along all three X, Y, and Z coordinate axis.

>

> The "things" could be simple points (Vertex class), lines (Line

> class), planar faces (Faces class), or solid things (Solids class).

>

Re: Abstract class or interface?

- > All these things are defined as supporting the I3D interface which
- > means they must implement methods such as Rotate(x angle, y angle,
- > z angle), Translate(some axis), Scale(some percentage), and so forth.
- >
- > Rotating a Vertex is different than a Line, but the interface doesn't
- > know nor care what the differences are. Furthermore, I can place all
- > the different things in an ArrayList by their interface and then run
- > a simple...
- >
- > foreach (I3D i3dobj in arraylist)
- > i3dobj.Rotate(xangle,yangle,zangle);
- >
- > to rotate all the objects in the scene.
- >
- > In this example it would be possible to define an abstract class
- > called, say, Ab3D, and have Vertex, Line, Face, etc., derive from
- > Ab3D and override the Rotate, Translate, etc. methods in each.
- > This implies some closer coupling between these classes than I
- > would like, furthermore...
- >
- > What if I want to include rotated text or something else where
- > the bulk of the work is already defined in an existing class which
- > does not derive from Ab3D? If I don't have control of that existing
- > class' definition, or if it itself is a derived class I cannot make it
- > derive from Ab3D to override those methods.
- >
- > With interfaces however, I can derive a new class from that already
- > existing one, state that it supports the I3D interface, and then
- > write the supporting Rotate, Translate, etc., methods for it. Then I
- > can include it in the arraylist above and perform the 3D actions on
- > it with no changes to manipulation code. I do not have to change
- > the interface to make new things 3D manipulatable.
- >
- > Oz
- > --
- > A: Because it fouls the order in which people normally read text.
- > Q: Why is top-posting such a bad thing?
- > A: Top-posting.
- > Q: What is the most annoying thing on usenet and in e-mail?

Thanks. Great example. Say your interface has three methods – X, Y, Z. You want to add a forth – Time or T. What now? This is completely unexpected. I know you could allow your implementation to inherit from another interface, perhaps called ITime. What is best?

Brett

.

Re: Abstract class or interface?

- *Follow-Ups:*
 - ◆ *Re: Abstract class or interface?*
 - ◇ *From:* Jon Shemitz

- *References:*
 - ◆ *Abstract class or interface?*
 - ◇ *From:* Brett
 - ◆ *Re: Abstract class or interface?*
 - ◇ *From:* ozbear

- Prev by Date: *Re: Abstract class or interface?*
- Next by Date: *Starting Service*
- Previous by thread: *Re: Abstract class or interface?*
- Next by thread: *Re: Abstract class or interface?*
- Index(es):
 - ◆ *Date*
 - ◆ *Thread*