

## Re: Interface freeloading on a superclass – is it good practice?

---

*Source:* <http://coding.derkeiler.com/Archive/Java/comp.lang.java.programmer/2005-08/msg02276.html>

---

- *From:* "Oliver Wong" <[owong@xxxxxxxxxxxxxxxx](mailto:owong@xxxxxxxxxxxxxxxx)>
  - *Date:* Wed, 17 Aug 2005 13:19:17 GMT
- 

"jan V" <[nul@xxxxxx](mailto:nul@xxxxxx)> wrote in message

[news:1WBMe.171245\\$7k6.9642051@xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx](mailto:news:1WBMe.171245$7k6.9642051@xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx)

>

> Who's talking about changing any method signatures? That's not what I  
> said,

> I wrote "reimplementing [methods], possibly with their functionality  
> subtly

> changed".. in other words I'm talking about the precise semantics of a  
> method being subtly changed. The method signature isn't touched at all.

>

> The problem with your scheme is that

>

> a) you pick arbitrary methods from abstract/concrete classes which are not  
> the subject of any interface contracts written by the same authors of the  
> classes

>

> b) you define an interface which contains those same methods (as  
> signatures

> only, obviously), and by doing so you "freeze" a particular \*semantic  
> contract\* for these methods

>

> c) you now subclass a third-party class (Sun's), and impose on this class  
> your interface's contract.

>

> From now on, any future changes to the \*precise semantics\* of the methods  
> you picked, as altered or enhanced or rewritten by Sun, may very likely  
> break your scheme.

>

> Why? Because Sun don't know or care about your interface's rigid semantic  
> contract, but you do... and suddenly the methods you tagged as "belonging"  
> to your interface don't match the semantics you imposed on them. The Sun  
> changes lead to a branching in semantics for these methods: one branch for  
> Sun's semantics, and one branch for your interface's semantics. Now guess  
> who's going to loose the battle of this splitting evolution? You, not Sun,  
> since Sun aren't even aware some clever guy decided to try to impose a  
> semantic straightjacket on a subset of Component methods....

>

## Re: Interface freeloading on a superclass – is it good practice?

>> So i am resting quite assured that it won't happen.

>

> Do you understand the scenario now?

I'm not sure I understand how this is primarily a "bad use of interface" problem as opposed to a "depending on undocumented behaviour" problem. I think you'd have the exact same problem if you used one of Sun's classes in composition and depended on a particular behaviour; if Sun changed the behaviour, your code wouldn't work anymore.

I'd imagine typically that if one were to create an interface for a particular set of methods from one of Sun's classes, the documentation explaining the semantics of those methods would pretty much be taken straight from Sun's javadocs. Then the interface creates no new restrictions on behaviour that weren't already publicly documented on Sun's site. If instead, the documentation for the new Interface read "int compareTo(Object o): Compares this object with the specified object for order. Additionally, computes the cube root of the hashcode of object o." then I don't think the problem lies with Interfaces per say, but rather is more of a fundamental misunderstanding of design-by-contract.

And if Sun changes documented behaviour, you can be sure a lot of people will get upset.

– Oliver

---

### • *Follow-Ups:*

- ◆ **[Re: Interface freeloading on a superclass – is it good practice?](#)**  
◇ *From: jan V*

### • *References:*

- ◆ **[Interface freeloading on a superclass – is it good practice?](#)**  
◇ *From: Chris Berg*
- ◆ **[Re: Interface freeloading on a superclass – is it good practice?](#)**  
◇ *From: Chris Berg*
- ◆ **[Re: Interface freeloading on a superclass – is it good practice?](#)**  
◇ *From: jan V*
- ◆ **[Re: Interface freeloading on a superclass – is it good practice?](#)**  
◇ *From: Chris Berg*
- ◆ **[Re: Interface freeloading on a superclass – is it good practice?](#)**  
◇ *From: jan V*

- Prev by Date: **[Re: compiler warning](#)**
- Next by Date: **[Serializing beans. Change in code makes old files unreadable.](#)**
- Previous by thread: **[Re: Interface freeloading on a superclass – is it good practice?](#)**
- Next by thread: **[Re: Interface freeloading on a superclass – is it good practice?](#)**

Re: Interface freeloading on a superclass – is it good practice?

- Index(es):

- ◆ *Date*

- ◆ *Thread*