

## Re: Static vs. Dynamic typing (big advantage or not)---WAS: c.programming: OOP and memory management

**Source:** <http://coding.derkeiler.com/Archive/General/comp.object/2004-08/0730.html>

---

**From:** Thomas Gagne ([tgagne\\_at\\_wide-open-west.com](mailto:tgagne_at_wide-open-west.com))

**Date:** 07/27/04

Date: Mon, 26 Jul 2004 20:10:32 -0400

Michael N. Christoff wrote:

<snip>

>

>

> *This makes no sense, to me at least. How is the user of some software*  
> *'always' more knowledgeable than the person who wrote it? This implies that*  
> *before you send an object to another bit of code, you have to analyze the*  
> *receiver to the point where you know more about it than the authors*  
> *themselves. Seems like a waste of time to me, when we could simply have the*  
> *receiver specify what objects are appropriate. ie: everyone takes*  
> *responsibility for their own code. Its almost a restatement of the idea*  
> *behind encapsulation. My object has a protected state and I (the receiver)*  
> *take responsibility for protecting that state. I do not pass off that*  
> *responsibility to the users of my code. Also, there is the whole issue of*  
> *the use of commercial closed source software where there is no way the*  
> *sender could be more knowledgeable about the receiver than the receiver*  
> *itself.*

>

They don't have to know the receiver inside and out---only enough to know what it's supposed to do, then they should be able to send it anything that meets the /minimum/ requirements. In another subthread there's discussion about the effort required in most statically typed languages to define what a minimum interface is.

Don't limit consumers' knowledge of your objects to only a brief acquaintance.

They may be using it intimately for weeks, months, or years. In that amount of time they may have found more reasons and places to use it and perhaps have become more intimate than even its parents. These are typically long-term relationships and not one-night stands.

>

>> *Who is better equipped to determine the utility of a screwdriver,*  
>> *the screwdriver or the craftsman?*

comp.object: Re: Static vs. Dynamic typing (big advantage or not)---WAS: c.programming: OOP and memory management

- >
- >
- > *This is a bad analogy. It is one of those things where a physical analogy*
- > *does not map back into the actual situation: in this case software*
- > *development. Instead of a screwdriver consider a DVD player. Who is better*
- > *equipped to determine the suitability of a disk for a disk player, the user*
- > *or the player?*

The user. The DVDPlayer object may use an embedded object capable of reading CDs but place an unnecessary restriction on the disk. The user, perhaps through trial-and-error or by reading some other documentation or talking to another user, discovers CDs *should* play just as well but are frustrated by the player's insistence that only a DVD be inserted.

I like cassette players, too. They expect cassettes to be inserted, but someone invented something that looks like a cassette but is really a CD player that does some kind of magic so that the CD player's music comes through your car's stereo system via the cassette player. Cool stuff. The cassette designers probably hadn't anticipated that, but they didn't require cassettes actually have tape in them either, or that it would be an error after playing in one direction for six days without rewinding or replacing the cassette.

How about light bulb sockets? Intended, perhaps, for incandescent lamps but florescent lamps that resembled the incandescent fit as well --- because they meet /minimum/ requirements. Or I can screw-in an outlet that fits the sockets. Or I can stick my finger in and get a runtime exception (that which doesn't kill us makes us stronger).

Unless you intend to open the player up to the point where  
> *'your knowledge is more current and extensible than the designers of the DVD*  
> *player', then the receiver is in a better position to decide.*

There's an assumption here the DVD player's designers knew *everything possible*. Basically, that's impossible for humans. Kinda like the ultimate insult to reusability --- declaring a class final. National constitutions provide mechanisms for their own modification but not so Java's String class. It's creators were more inspired even than Madison, Jefferson, et al.

But just as  
> *in software development, if this level of knowledge was required before we*  
> *could operate the typical devices we use in our daily lives, no one would*  
> *have time to get anything done. That's why encapsulation and the sharing of*  
> *responsibility are thought of so highly in OO software development.*

I encapsulate so users don't /need/ to know how the internals work, not because I don't want them to know or believe my code to be perfect and beyond improvement (even as elegant and sublime as my objects are I /occasionally/ find imperfections so subtle their detection and repair is beyond that of most mortals but required nonetheless).

Re: Static vs. Dynamic typing (big advantage or not)---WAS: c.programming: OOP and memory management