

Re: static or not?

Source: <http://coding.derkeiler.com/Archive/Java/comp.lang.java.help/2005-05/msg00639.html>

- *From:* "hawat.thufir@xxxxxxxxxx" <hawat.thufir@xxxxxxxxxx>
 - *Date:* 27 May 2005 06:18:44 -0700
-

iamfractal@xxxxxxxxxx wrote:
Hi, Ed!

....

- > Thus, for example, controller.Driver will register its Controller
- > interface in the Registry by calling:
- > Registry.getInstance().register(this).

```
package controller;
package-private class Driver extends controller.Controller{
void register() {
Registry.getInstance().register(this);
}
}
```

is the register method correct as far as it goes?

"If I use a singleton, then I can strip away an interface from this singleton, and have it register that interface in the Registry.

If I use static methods, I cannot do this: static methods are not allowed to hide the instance methods of the interface.

This is the major reason I use a singleton, instead of a class of static methods. " -Ed

this is where you're stripping away an interface?

- > Next, a user will make some input; we needn't specify how, here, but
- > suffice it to say that when that input is made, an object will need to
- > get that input into the system. It will do this by doing something
- > like:
- > Controller controller = Registry.getRegistry().getController();
- > controller.process(input);

if this is done from class foo, what's foo's package? of MVC
I think view, but view presents data to the user, so I'm not sure.

- > The Driver object will have its process() method called, and will kick

Re: static or not?

- > the database into life, by something like:
- > Registry registry = Registry.getInstance();
- > Model model = registry.getModel();
- > model.doThis();
- > model.doThat();
- > Result result = model.getResult();
- >
- > (OK, here's an interface I'd forgotten; any guesses where we'd declare
- > it?)

heh, ir1 of course.

- > The Driver object would then output the result to the use by:
- > int viewID = registry.getCurrentViewID();
- > View view = registry.getView(viewID);
- > view.show(result);

where did result come from, please?

view.show() might trigger a SQL query and would show the outcome of that query?

you said "...portray Driver as a Controller," so it's:

```
package controller;
package-private class Driver implements SomeInterface {...
```

- > Thus we have a system in which there are only two types of
- > inter-package, concrete class dependencies:
- >
- > 1) From Start to XStart – and these XStart classes only contain enough
- > logic to kick start the business logic, in other words, these
- > dependencies do not change as business logic changes.

if everything in the start package is business logic, might that logic consist of static methods, like Math.sin(x), or is that a bit extreme?

- > 2) Dependencies on the Registry concrete class – but this class simply
- > stores and returns interfaces, and has no dependencies on the
- > methods of those interfaces, and thus is immune to changes of the
- > interface methods (although of course will change as whole
- > interfaces change, and new ones are added).

- > And **all** other inter-package dependencies are on interfaces
- > only. Thus, you can switch in any database you like, as long as it
- > conforms to the Model interface (this will, in reality, probably be
- > JDBC, or a wrapper thereof). The same goes for the Views and even the
- > Controller itself.

Re: static or not?

Re: static or not?

- >
- > Hence, "Program to an interface repository, not an implementation repository."
- >
- > Note that model package, view package, and controller packages do not
- > depend on one another (you can Google for the heated discussions on
- > which of these packages should depend on one another, and which must
- > not), and that the system is guaranteed to be free from inter-package
- > circular dependencies.

err, next time ;)

- > Of course a full system will have hundreds of packages, and there will
- > be hierarchical start/implementation repository/irX packages, but the
- > design is – and particularly the benefits listed above are –
- > wonderfully extensible.
- >
- > .ed
- >
- > www.EdmundKirwan.com – Home of The Fractal Class Composition.

this helps to justify MVC for me, thanks.

–Thufir

.

• *Follow-Ups:*

- ◆ *Re: static or not?*
 ◇ *From:* iamfractal
- ◆ *Re: static or not?*
 ◇ *From:* hawat.thufir@xxxxxxxxxx

• *References:*

- ◆ *static or not?*
 ◇ *From:* David Vanderschel
- ◆ *Re: static or not?*
 ◇ *From:* iamfractal
- ◆ *Re: static or not?*
 ◇ *From:* hawat.thufir@xxxxxxxxxx
- ◆ *Re: static or not?*
 ◇ *From:* iamfractal
- ◆ *Re: static or not?*
 ◇ *From:* hawat.thufir@xxxxxxxxxx
- ◆ *Re: static or not?*

Re: static or not?

◇ *From: iamfractal*

- Prev by Date: ***Re: Splitting Vector into smaller sub-Vectors***
- Next by Date: ***Re: static or not?***
- Previous by thread: ***Re: static or not?***
- Next by thread: ***Re: static or not?***
- Index(es):
 - ◆ ***Date***
 - ◆ ***Thread***