Kepler - Bug #4282

Duplicate of ptolemy.gui.Top

08/05/2009 02:28 PM - Christopher Brooks

Status:	Resolved	**Start date:**	08/05/2009
Priority:	Immediate	**Due date:**	
Assignee:	Chad Berkley	**% Done:**	0%
Category:	build system	**Estimated time:**	0.00 hour
Target version:	2.0.0	**Spent time:**	0.00 hour
Bugzilla-Id:	4282		

Description

I'm creating a bug for this so that it does not get lost and because
this will block bug# 3801.

I don't have the time right now to do a merge between the fix for 3801
and the duplicate code. I don't think the functionality provided
by the duplicate copy of Top is worth the overhead of a duplicate class
and an entire module (ptII).

David wrote:

> Hi Christopher,
>
> I am happy to consider any alternative methods that you would like to
> achieve the same result. What I want is a menu item that allows you to
> invoke the Module Manager from Ptolemy. This allows you to switch from
> Ptolemy to Kepler and back via the Module Manager without using the
> build system. This has proven to be highly convenient on more than one
> occasion.
>
> This functionality probably does not make sense in the context of
> Ptolemy run without Kepler. That is why it is in the Kepler repository
> and I have not discussed with you the possibility of integrating it
> into Ptolemy.
>
> The reason I use Top is so that the option to invoke the Module
> Manager will exist in every possible context where Ptolemy is run
> using the Kepler build system. That way, a user exploring Ptolemy
> functionality always has an easy way to invoke the Module Manager to
> get back into Kepler.
>
> David

On Aug 5, 2009, at 11:48 AM, Christopher Brooks wrote:

> Hi David,
> It looks like you checked in ptII/src/ptolemy/gui/Top.java,
> which is a duplicate of ptolemy/src/ptolemy/gui/Top.java.
>
> The comment for ptII/src/ptolemy/gui/Top.java is:
>
> r19082 | welker | 2009-06-05 14:41:40 -0700 (Fri, 05 Jun 2009) | 2
> lines
>
> Adding a Tools-->Module Manager... menu item into ptII so that users
> who switch to Ptolemy from the Module Manager can go back to Kepler.
>
> I think we should remove ptII/src/ptolemy/gui/Top.java because it
> is a duplicate and because the functionality it provides is used very
> little, if at all.

> It looks like the ptII module is only present to support the duplicate
> Top class. Thus, if we get rid of the duplicate Top class, we can
> get rid of the ptII module, which reduces complexity in the module
> system.
>
> If you would like to come up with a patch to Top.java that provides
> a hook for the menu choice that you would like to add, then we could
> consider that. One problem is that Top does not use the
> Configuration,
> so this might entail setting a property and reading it or something.
>
> The reason this comes up is because I'm tracking down the white boxes
> under Windows problem at
> http://bugzilla.ecoinformatics.org/show_bug.cgi?id=3801
> that occurs when the background is set to white.
>
> My fix worked fine in ptolemy outside of Kepler, but failed inside
> Kepler
> because of the duplicate Top class.
>
> _Christopher

Related issues:

Blocked by Kepler - Bug #3801: open dialog, common places pane has white box ...	**Resolved**	**01/29/2009**

History

#1 - 08/26/2009 03:12 PM - Chad Berkley

Get rid of override in ptII

#2 - 08/26/2009 07:16 PM - David Welker

(In reply to comment #1)

> Get rid of override in ptII

I am not sure what the substance of this reply is. I think the override should stay. It allows one to switch from Ptolemy to Kepler to Ptolemy to Kepler, all from the GUI. There are no real disadvantages, except the concern that Top will change and we will not get the latest version. But Top.java does not change very often.

#3 - 08/26/2009 11:27 PM - Matt Jones

The functional changes needed in Top should just be refactored as extensions to the existing Top so that they can coexist in the Ptolemy tree. This eliminates the need for someone to maintain the class against changes in the Ptolemy tree.   We have 5 years of evidence that people do not maintain the differences even for classes that change infrequently. We've inserted extension hooks in ptolemy before -- its just not that hard to find a way to do these things that is maintainable without overrides but still provides the needed functionality.

#4 - 08/27/2009 05:14 AM - David Welker

A hook allowing the extension of ptolemy rather than an override is fine.

I think I was more responding to the view expressed by Christopher that this feature just isn't important enough to justify an override.

#5 - 08/27/2009 09:00 AM - Christopher Brooks

The point is that the duplicate version of Top is a bug that was added
for functionality that no one uses, which is to invoke the module manager
from vergil that is invoked using the Kepler build system.

Has anyone other than David used this? I think I'm likely the only other
person that even tries to invoke vergil using the Kepler build system. Why
spend time working on functionality that is virtually unused?

Bug #3801 (white boxes in the open file dialog) is blocked by this.
A workaround for #3801 would be to merge the
changes from the original version of Top into this copy of Top.

#6 - 08/27/2009 09:33 AM - David Welker

Christopher -

An override is not a bug. That you entered it into Bugzilla does not change that.

You may be right (or not) about the value of being able to switch between Kepler to Ptolemy. I should note, however, that if we take the ability to switch from Ptolemy to Kepler out of the module manager, then the ability to switch from Kepler to Ptolemy must be taken out as well. It would be an untenable situation for a user to switch to Ptolemy and not ever be able to get back out.

It is my view that you have decided to take a personal preference of yours, and make it into a bug. Which is fine, since the way we use Bugzilla does, as you know, accommodate feature requests. The point is, the desire to run Ptolemy from the Kepler build system may be small, but so is the importance or level of concern we should devote to this issue. Whether Top is overridden or not is an implementation detail that is of concern to developers and not users. Furthermore, I am sure that developers could easily fix any issues that arose due to future changes in the functionality of Top.

That said, I am fine with the solution of removing Ptolemy as something that can be invoked from the Kepler build system. It would allow me to refactor the build system code and make it somewhat cleaner. I am also fine with the solution of having a hook written into Ptolem

However, I vaguely remember Bertram using this feature to explore some functionality that was implemented in Ptolemy but not Kepler. I am not sure how important it is to him.

A final point. Upon reflection, the solution that Matt suggests of writing a hook into Top is probably not a technically feasible solution. After all, we are not talking about extending Ptolemy. We are talking about changing the behavior of Ptolemy so that the module manager can be invoked from the menu bar of any JFrame that extends Top. Therefore, I think the two feasible solutions are to either (1) keep this override or (2) remove the ability to run Ptolemy from the build.

#7 - 08/27/2009 09:47 AM - Christopher Brooks

Being able to invoke vergil from the build system is separate functionality
from being able to invoke the module manager from vergil.

Two other solutions:
3) Remove the copy of Top - which means that invoking vergil from the kepler
build system would not be able to invoke the build system.
4) Updating the copy of Top to fix bug 3801 and then moving this bug to 2.X.Y
seems like a quick workaround.

#8 - 08/27/2009 10:21 AM - Chad Berkley

Though overrides are not technically "bugs," they do represent the potential for many true bugs in the future. We have been dealing with this for 5 years. Each of us has created an override and said "we'll make sure we maintain that" but eventually it is forgotten until something really bad starts happening then it takes forever to debug the problem. So even though overrides are not technically bugs, personally, I would prefer if they did not exist in any core module of Kepler. They have enough potential to create future bugs that they should be avoided and in their place, functionality added to make the feature extendable in a configurable manner.

#9 - 08/27/2009 10:28 AM - David Welker

"Being able to invoke vergil from the build system is separate functionality
from being able to invoke the module manager from vergil."

First, you are the one that originally suggested that invoking Vergil from the build system was not important. I was going along with your conflation of the two issues.

Second, these things are in fact linked, at least as the module manager is currently implemented. Since the module manager uses the build system, it is possible to invoke Vergil from it. In fact, it is even easy to do so. Therefore, your proposed third solution is not workable. It is completely unacceptable in an installed system where Kepler users cannot be expected to even know about the build system, much less use it, to have a one-way trip to Vergil and never be able to return except through use of the build system from the command-line. But that is precisely what would happen if we removed the override to Top.

Now, I happen to be working on the module manager once again. So, it is possible for me to simply disable the ability to switch to Vergil from the module manager by detecting the situation where Vergil would be invoked and disabling the buttons that would allow you to proceed.

This assumes of course, that for ordinary users, switching to Ptolemy from the module manager GUI is not desirable.

I am not sure that this issue of overriding Top, which in my view is a trivial technical issue, is worth removing functionality from the GUI based on the assumption that the functionality is not useful.

In a more general sense, I think that the issue of overrides is one where worries are more theoretical than practical. I am not saying that problems never happen. I am saying they rarely happen and when they do they have a fairly evident solution. Let us assume you refactored Top such that Ptolemy was rendered incompatible with Kepler until the override of Top was changed. It would be very straightforward to fix such a problem, which one would expect to occur rarely. Yes, module incompatibilities can occur due to overrides. But even more insidious (because such incompatibilities are not as detectable) incompatibilities can occur due to the mixing of incompatible jars.

In general, we should look at overrides from a perspective of cost-benefit analysis where primarily speculative concerns are not given disproportionate weight. In my whole time working on Kepler, I do not recall a serious bug that I have had to deal with due to overrides (although I do in fact such bugs to appear).

Of course, all things being equal, developing hooks, when feasible (here it is not), is better than overriding. Why increase the probability of incompatibilities, even if only slightly, if it could be avoided? Also, as the number of overrides increased, one could expect that the probability of incompatibilities would become more significant.

However, things are rarely equal. In many cases, developing a hook can be quite expensive. It isn't always worth devoting developer time to such tasks merely to avert speculative but real and rather small probabilities of incompatibilities. Also, issues of timing are very important. If a developer is working on a related project anyway, it might be more feasible to develop hooks that render certain overrides unnecessary.

All I am saying is that as I think discussion of Top has illustrated, concern about overrides is given too much weight by some people. It can be very expensive to develop hooks or alternative solutions. I think it would rarely be worth it to remove existing functionality, even if we think that functionality is relatively rarely invoked, merely to remove an override.

#10 - 08/29/2009 02:18 AM - Bertram Ludaescher

For instructional and educational purposes I found it useful to be able to switch to Ptolemy from Kepler, not least because of the plethora of nice demos, illustrating different models of computations etc.
Having said that, I don't think it's a terribly important use case, so I'm not having strong opinions on this issue. I did find it neat though to be able to run Ptolemy from Kepler without having a separate Ptolemy installation (as I used have previously).

BL

#11 - 08/29/2009 12:57 PM - Daniel Crawl

FYI, I've updated the loader so you can start Ptolemy from the command line:

```
./kepler.sh -vergil [model.xml]
```

#12 - 09/10/2009 01:05 PM - Chad Berkley

I added a configuration parameter called _alternateTopHandler into the configuration. This defines a class to override the open(), saveAs() and pack() methods in ptolemy.gui.Top. The class is currently set to ptolemy.gui.KeplerTopHandler which is located in the gui module. KeplerTopHandler needs to be in the ptolemy.gui because much of what it changes in Top is protected.

This bug is now closed, unless Christopher objects to something I did in Top. I did have to add two imports to Top:

```
import ptolemy.actor.gui.Configuration;
import ptolemy.kernel.util.StringAttribute;
```

I'm not sure if this will be ok with Christopher or not. If not, we'll have to figure out a new way to do this override.

Christopher, please re-open this bug if you want me to do it differently or don't agree with my solution.

#13 - 09/10/2009 01:45 PM - Christopher Brooks

ptolemy.gui cannot depend on ptolemy.actor because Ptplot uses
ptolemy.gui.Top and does not ship with ptolemy.actor.

Could you back out the changes to Top in the ptII svn repository?

I don't have a good workaround that will provide the functionality of adding
a menu choice to Vergil/Ptolemy.

If we remove ptII/src/ptolemy/gui/Top.java, then the entire ptII module can
go away. The addition to Top is only necessary to provide access to the module
system if someone is running Vergil/Ptolemy (not Kepler) from the kepler build
system.
This functionality is not part of Vergil/Ptolemy, it is not documented and is
little used.

The module system seems useful, and Top should be capable of having menu items
added to it. Reworking this using Eclipse or Netbeans RCP is the right way
to go. It seems to me that removing the ptII module would increase programmer productivity. It is confusing that we have a ptolemy module and a ptII module.

#14 - 09/10/2009 02:53 PM - Chad Berkley

I've removed my changes to Top.java in ptolemy and commented out the configuration option in kepler. There is no way to pass in the needed configuration items to Top without using the ptolemy.actor package which is inappropriate for ptolemy.

Should probably just leave Top.java out of kepler for now until the new configuration system is up and running when we can revisit this.

#15 - 09/10/2009 07:05 PM - David Welker

Absent any objections, I will implement the following solution:

Let's remove the ability to change to Ptolemy from the GUI. This involves some changes to the code, because currently the module manager can change to any configuration that the build can change to.

If the need to change to Ptolemy from the GUI is removed, then there will be no need to change to Kepler from Ptolemy. Then, the override of Top.java will be unnecessary.

This change to the module manager is not a high priority. However, I will get to it soon, because I am in the process of making a new version of the module manager that supports patching.

#16 - 09/11/2009 09:48 AM - Chad Berkley

That sounds good to me.

#17 - 09/11/2009 10:42 AM - Chad Berkley

I've removed all of my changes from yesterday and I have not re-added Top.java to the ptII module, so this bug is effectively closed. I will create a new enhancement request linked to this bug for the changes David plans to make.

#18 - 03/27/2013 02:26 PM - Redmine Admin

Original Bugzilla ID was 4282