

User Interface - Bug #2837

Bug # 2677 (New): fix drawing and functional differences between P2J GUI and 4GL GUI

GUI combo-box drop-down needs to be able to draw outside of it's containing top-level window

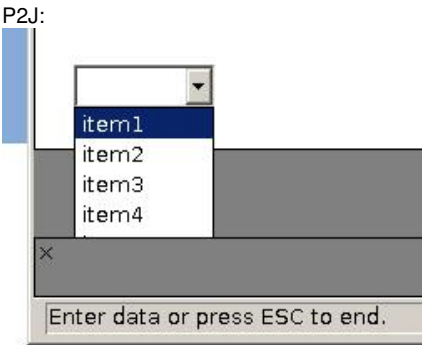
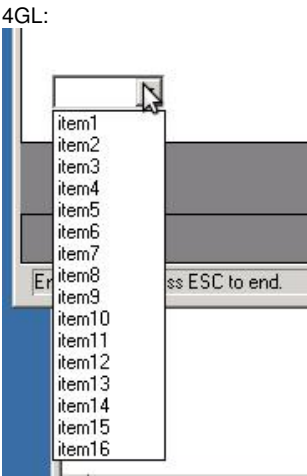
11/09/2015 09:38 PM - Hynek Cihlar

Status:	Closed	Start date:	
Priority:	Normal	Due date:	
Assignee:	Eugenie Lyzenko	% Done:	100%
Category:		Estimated time:	0.00 hour
Target version:	GUI Support for a Complex ADM2 App	case_num:	
billable:	No	version:	
vendor_id:	GCD		
Description			
Related issues:			
Related to User Interface - Bug #2970: rework menus to use OverlayWindow			Closed
Related to User Interface - Bug #2971: rework tooltip support to use OverlayW...			New
Related to User Interface - Bug #2982: in Web client, overlay window must not...			Closed

History

#1 - 11/09/2015 09:44 PM - Hynek Cihlar

- File Drop-down 4GL.png added
- File Drop-down P2J.png added



#2 - 11/09/2015 09:46 PM - Hynek Cihlar

- File deleted (Drop-down 4GL.png)

#3 - 11/09/2015 09:46 PM - Hynek Cihlar

- File deleted (Drop-down P2J.png)

#4 - 11/09/2015 09:47 PM - Hynek Cihlar

- File Drop-down_4GL.png added

- File Drop-down_P2J.png added

#5 - 11/10/2015 05:24 AM - Greg Shah

This issue seems related to #2816, since the size of the combo-box drop-down will be affected by the font/line height that is wrong.

However, the core issue here is that we will have to implement a kind of overlay "window" that can extend its drawing area outside of the containing 4GL window. This would be a special canvas element in the web client and some other kind of JFrame or equivalent in Swing GUI.

This will probably also have to be used for menus, popup menus and tooltips. I guess all off these things can draw outside of the containing window.

#6 - 11/10/2015 05:29 AM - Hynek Cihlar

Greg Shah wrote:

This will probably also have to be used for menus, popup menus and tooltips. I guess all off these things can draw outside of the containing window.

Yes, good point.

#7 - 11/24/2015 09:25 AM - Greg Shah

- Target version set to Milestone 12

- Start date deleted (11/09/2015)

- Subject changed from GUI combo-box drop-down won't leave the top-level window to GUI combo-box drop-down needs to be able to draw outside of it's containing top-level window

#8 - 12/13/2015 01:38 AM - Eugenie Lyzenko

The task 2837a has been created based on trunk revision 10959.

#9 - 12/15/2015 06:32 PM - Eugenie Lyzenko

After investigation the I have something to present as implementation plan.

1. I guess we need to have the driver independent solution, so simple JFrame or JPanel implementation does not fit.
2. Using WindowGuilImpl or subclass is too heavy to work within DropDownGuilImpl because the WindowGuilImpl has many features we do not need to present opened drop-down.
3. Possible solution can be the introducing new class on the same level as ModalWindow, say OverlayWindow extending TopLevelWindow to have good starting point and be able to have separate draw area outside the main window.
4. When DropDownGuilImpl is creating we create the overlay window and attach DropDownGuilImpl to the overlay window instead of the main window the combo-box belongs to.
5. This modified logic will work only in GUI code and because no direct using of Swing - we will have common solution for both Swing and Web

clients.

If there are some points I've missed that prevent this plan to work properly let me know.

#10 - 12/16/2015 06:04 AM - Hynek Cihlar

Eugenie Lyzenko wrote:

After investigation the I have something to present as implementation plan.

1. I guess we need to have the driver independent solution, so simple JFrame or JPanel implementation does not fit.
2. Using WindowGuilImpl or subclass is too heavy to work within DropDownGuilImpl because the WindowGuilImpl has many features we do not need to present opened drop-down.
3. Possible solution can be the introducing new class on the same level as ModalWindow, say OverlayWindow extending TopLevelWindow to have good starting point and be able to have separate draw area outside the main window.
4. When DropDownGuilImpl is creating we create the overlay window and attach DropDownGuilImpl to the overlay window instead of the main window the combo-box belongs to.
5. This modified logic will work only in GUI code and because no direct using of Swing - we will have common solution for both Swing and Web clients.

If there are some points I've missed that prevent this plan to work properly let me know.

This sounds like a good plan. TopLevelWindow was meant for the purpose of top-level windows like in this case.

#11 - 12/16/2015 08:32 AM - Greg Shah

I like the plan.

Have you checked if these windows can stay open while you are moving/minimizing/restoring/maximizing/resizing the parent window? Please test these cases and make sure our plan can handle any needed behavior.

Make sure to test menus and tooltips too. I suspect that for menus and tooltips, the window is dismissed when these events occur. But for the combo drop-down, I wonder.

#12 - 12/16/2015 09:04 AM - Eugenie Lyzenko

Greg Shah wrote:

Have you checked if these windows can stay open while you are moving/minimizing/restoring/maximizing/resizing the parent window? Please test these cases and make sure our plan can handle any needed behavior.

Make sure to test menus and tooltips too. I suspect that for menus and tooltips, the window is dismissed when these events occur. But for the combo drop-down, I wonder.

4GL investigations:

Any mouse press outside drop-down causes the drop-down to dismiss. No further event generations. For example if to press on minimize button while drop-down is opened, no minimize event is happening, only drop-down close.

The tooltip dismisses when mouse leaves the widget, so tooltip also not have a chance to handle such events.

The menu also dismissing, but unlike drop-down the extra event generation is possible, when for example to press main window control buttons(min, max, close).

#13 - 12/16/2015 09:12 AM - Greg Shah

OK, go ahead with your plan. Make sure to get all of these behaviors correct in the implementation.

#14 - 12/18/2015 03:29 PM - Eugenie Lyzenko

- File *cbb_test21_p2j_gui_20151218.jpg* added

This is the very first picture of the overlay window with opened drop-down widget. And there is one point to discuss. Consider shadow effect around the drop-down. Unfortunately this is the OS decoration feature for every window in Ubuntu and I doubt we can control this appearance from java code.

The good news is looks like in Swing client this is not happening.

What do you think, how critical it is? Do we have to fix it somehow or can live with it?

#15 - 12/18/2015 03:46 PM - Eugenie Lyzenko

Rebased task branch 2837a from P2J trunk revision 10960, new branch revision is 10960 to(no updates yet).

#16 - 12/18/2015 07:09 PM - Eugenie Lyzenko

Task branch 2837a updated to revision 10961.

This is the early draft version, so lot of commented out code is left. And some issues. But some basic functionality is working. The drop-down is opening in right place, drawing outside main window is possible, selection is possible, drop-down is closing on selection.

Looks like widget physical Y location is not matched the actual one if the owning frame has title. Had to add this value to combo widget to get right coordinate for drop-down starting point.

Continue working.

#17 - 12/21/2015 03:12 PM - Greg Shah

This is the very first picture of the overlay window with opened drop-down widget. And there is one point to discuss. Consider shadow effect around the drop-down. Unfortunately this is the OS decoration feature for every window in Ubuntu and I doubt we can control this appearance from java code.

This only happens in Ubuntu? If you run this in another OS, this does not occur?

If so, then I think it is OK.

The good news is looks like in Swing client this is not happening.

Did you mean that it doesn't occur in the web client?

#18 - 12/21/2015 03:15 PM - Greg Shah

By the way, it looks good so far. Nice work!

#19 - 12/21/2015 04:06 PM - Eugenie Lyzenko

Greg Shah wrote:

The good news is looks like in Swing client this is not happening.

Did you mean that it doesn't occur in the web client?

Yes I meant Web client, sorry for confusing, the Web client has no shadow for window so this effect does not work.

#20 - 12/21/2015 08:27 PM - Eugenie Lyzenko

Task branch 2837a updated to revision 10962.

Fixed size/location issues. Still having uncleaned code due to active development is in progress. Continue working.

#21 - 12/22/2015 11:41 AM - Eugenie Lyzenko

Rebased task branch 2837a from trunk revision 10961, new branch revision is 10963.

#22 - 12/22/2015 07:59 PM - Eugenie Lyzenko

Task branch 2837a updated to revision 10964.

This is almost ready first release. The only remaining issue is inability to dismiss drop-down by pressing mouse on the owning window title. The root cause is strange. Looks like the window title does not receive/generate mouse press event. Investigating. If someone knows why the ThinClient does not receive mouse press for window title(but receive mouse click) please share.

Also a bit more testing is require for new drop-down implementation of course.

Continue working.

#23 - 12/23/2015 04:50 PM - Eugenie Lyzenko

- File *cbb_test21_p2j_web_20151223.jpg* added

Task branch 2837a updated to revision 10965.

This is the fix for web client NPE issue related to creating child window with no title. And in web client there is no shadow effect. The sample picture is attached.

#24 - 12/23/2015 09:38 PM - Eugenie Lyzenko

Task branch 2837a for review updated to revision 10966.

The combo-box testcases updated to revision 1443. The test for overlapped drop-down is `./combo_box/combo_box21.p`

This update has the fix for issue when drop-down is not closing when pressing on window title. The idea is the window has no mouse click event when pressing on title. It has window activation event instead. So handling window activation we can get the current overlay window from `WindowManager` and properly terminate it. To get rid of the explicit calling to `exitDropDown()` it is possible to introduce new interface for `OverlayWindow` children, so all of them will be implementing some common `dismiss()` method. This will allow us to use not only drop-down widget to handle the event like this.

The other issue has been found for web client. The drop-down is opening and displaying behind the main window. Looks like the overlay window is moving to the bottom of the window list. Investigating to provide possible solution.

#25 - 12/25/2015 04:59 PM - Eugenie Lyzenko

Task branch 2837a for review updated to revision 10967.

Some improvements for overlay window focus/activation management. The dismiss issue is still here. Not possible to close drop-down pressing the window title and not always possible to do is by activating other non-P2J OS window.

Continue working. The issue is more complicated than I have expected. So unfortunately the detailed investigation and solution will be ready in a new year.

#26 - 01/05/2016 01:20 AM - Eugenie Lyzenko

Task branch 2837a for review updated to revision 10968.

This is the fixes for swing client issue related to ability to dismiss drop-down by pressing mouse on the main window title. Selecting another OS window is also properly works. The idea is to use main window activation/deactivation to control the drop down dismissing via sending deactivation not only moving focus to not p2j window. Commented out code still exists because the web client is not yet working.

Continue working.

#27 - 01/05/2016 09:25 AM - Eugenie Lyzenko

Web client issues with combo-box:

1. When drop-down is opening the main window gets activated and this moves the drop-down window to the bottom.
2. While drop-down is opened the mouse press on main window title does not generate de-activation message for overlay window. We can not use mouse click because it is too late, need to capture mouse press(for example if to make window moving by mouse the click event will be after window moved). In swing client we have window activation listener to keep track on main window activation. And we need the same API for web client.
3. The `screenPhysicalLocation()` returns the same coord for web client not depending on where the main window is located inside the web browser. We need to have the ability to get window coordinates relative to the browser workspace start. Otherwise the drop-down will always be opened in the same place not depending on current window location.

Another point is: how to start the web client in debug mode? I need to use jSwat to attach client session to debug or at least to be able to make some debug output by `System.out.println()` to some log channel. What is the debug possibility you used for this purpose?

#28 - 01/05/2016 01:57 PM - Sergey Ivanovskiy

Hi Eugenie,

3. The `screenPhysicalLocation()` returns the same coord for web client not depending on where the main window is located inside the web browser. We need to have the ability to get window coordinates relative to the browser workspace start. Otherwise the drop-down will always be opened in the same place not depending on current window location.

I think that `displayPhysicalLocation()` returns the absolute coordinates with respect to the screen and it takes into account the location of the parent window that is the location of the canvas.

In 1811t I have fixed the location of the canvas if the window is moving. JS module `p2j.mouse.js` should be changed to send correct window location coordinates

```
function processMouseDown(mThis, mouseOp)
{
    return function(evt)
    {
        if (mThis.dragOwner != wid || !mThis.mouseDrag || !p2j.screen.canProcessOsEvent(evt, win))
        {
            return;
        }

        var mousePos = win.getMousePos(evt);

        var dx = evt.clientX - mThis.lastMouseEvent.clientX;
        var dy = evt.clientY - mThis.lastMouseEvent.clientY;

        var rect = win.canvas.getBoundingClientRect();

        var x = dx + rect.left;
```

```

var y = dy + rect.top;

win.canvas.style.left = x.toString() + "px";
win.canvas.style.top = y.toString() + "px";
// !!! set the current location in order to send the actual data to java web client side.
mThis.screenLoc = { "x" : x, "y" : y };

mThis.lastMouseEvent = evt;

// event was processed, cancel other listeners
evt.preventDefault();
};
};

```

Another point is: how to start the web client in debug mode? I need to use jSwat to attach client session to debug or at least to be able to make some debug output by `System.out.println()` to some log channel. What is the debug possibility you used for this purpose?

I use this configuration in `directory.xml`

```

<node class="container" name="clientConfig">
  <node class="string" name="spawner">
    <node-attribute name="value" value="../../../1811t/build/native/spawn"/>
  </node>
  <node class="string" name="workingDir">
    <node-attribute name="value" value="/home/sbi/projects/testcases/simple/server"/>
  </node>
  <node class="string" name="jvmArgs">
    <node-attribute name="value" value="-Xmx512m -XX:MaxPermSize=64m -Djava.awt.headless=true -Xdebug
-Xnoagent -Djava.compiler=NONE -agentlib:jdwp=transport=dt_socket,address=9999,server=y,suspend=n"/>
  </node>
</node>

```

and it helps to attach the debugger (Eclipse) to the web client on the 9999 port, but you can change this value fitted to your needs.

#29 - 01/05/2016 02:57 PM - Sergey Ivanovskiy

1. When drop-down is opening the main window gets activated and this moves the drop-down window to the bottom.

If my understanding is correct then z-order is important here, the created child window should be on the top of the parent window. May be `void moveToTop(int windowId)` of `GuiDriver` can help to move the child window on the top position.

#30 - 01/06/2016 09:43 AM - Greg Shah

While drop-down is opened the mouse press on main window title does not generate de-activation message for overlay window. We can not use mouse click because it is too late, need to capture mouse press (for example if to make window moving by mouse the click event will be after window moved). In swing client we have window activation listener to keep track on main window activation. And we need the same API for web client.

In the web client, there is a javascript (JS) listener (`processWindowFocus()`) registered in `ui/client/gui/driver/web/res/p2j.mouse.js`. Each window (which is backed by an HTML5 canvas element) has a separate listener. When that canvas get a click, it requests the focus and moves itself to the top of the z-order (unless it is already at the top of the z-order). This is all done on the JS side up until this point.

At that point, it sends a `WebClientMessageTypes.MSG_WINDOW_ACTIVATED` message to the Java client using the websocket. That message has the window ID as the parameter.

In the Java client the code `GuiWebSocket.processBinaryMessage()` will be invoked to handle the message. It will execute a registered callback which is the `GuiWebDriver.windowActivated()` method, passing in the window ID as a parameter. That code will post the window activated event.

#31 - 01/06/2016 10:27 AM - Greg Shah

Code Review Task Branch 2837a Revision 10968

The changes are shaping up nicely.

Hynek: please do a code review.

1. Please add javadoc to `DropDownGuiImpl.dismiss()`.
2. The comments at the top of `OverlayWindow` suggest that one can make an instance modal, but `isModal()` always returns false.
3. `OverlayWindow.show()` should not have code that is specific to the web-client.

Greg Shah wrote:

Code Review Task Branch 2837a Revision 10968

The changes are shaping up nicely.

Hynek: please do a code review.

1. In sake of robustness the OverlayWindow, when constructed, should be passed the owner top-level window explicitly as returned by `combobox.topLevelWindow()`.
2. I think the reference to `DropDownGuiImpl` in `OverlayWindow.ctor()` is not necessary. Child widget is added to `OverlayWindow.widgets` and so `addWidgetsRecursive(this)` should properly register the child widget as well.
3. Similarly in `OverlayWindow.destroy()`, better to call `exitDropDown()` from `DropDown.destroy()`.
4. There is some commented out code in `OverlayWindow.processEvent()`. Also why not use the `destroy()` instead of `DropDownGuiImpl.dismiss()`? With `destroy()` no need to reference `DropDownGuiImpl` and `OverlayWindow` can be kept generic.
5. In `DropDownGuiImpl.findMouseSource()`

```
if (evtID == MouseEvent.MOUSE_CLICKED)
{
    mousePressedSource = null;
}
```

is never executed.

6. `OverlayWindow` has a lot in common with `ModalWindow`, it would probably make sense to derive a common parent for them. A suggestion for the future development.
7. `TC.waitForWorker()`, line 10792. What is the use case for the overlay window invalidation?
8. `TopLevelWindow`, line 699.

```
OverlayWindow ow = WindowManager.findOverlayWindow();
if (ow != null)
{
    WindowManager.moveToTop(ow);
}
```

Is the above needed? When a window receives an activation event, the drop down should be closed. Or am I wrong?

9. `TopLevelWindow`, line 740. A dead commented out piece of code.
10. Dead commented out code in `WindowManager`.

#33 - 01/06/2016 08:27 PM - Eugenie Lyzenko

- File `cbb_test21_p2j_web_20160106.jpg` added

Task branch 2837a for review updated to revision 10969.

Some of the notes have been resolved. The explanation for Web mouse events specifics helped significantly, thanks. Continue testing/working. For:

3. `OverlayWindow.show()` should not have code that is specific to the web-client.

The issue here has been presented on the picture attached. The main window was moved by mouse first, then the drop-down was opened. As we can see the overlay window has the same initial coordinates that was before window move. The method `screenPhysicalLocation()` for main window does not work properly for web client.

The idea behind the dismiss by main window activation is when `GuiWebDriver.windowActivated()` gets control after mouse press on window title we check if there is any overlay exists and deactivate it. The remaining issue here is when the client just starts it causes the drop-down to close just after opening. Not happening if to activate another OS window and re-activate web client back. Looks like we have some issue in initial window activation logic for web client. The main issue here is to debug this it is required another machine to preserve original web client window activation path because as soon as we switch to the debugger the activation become right.

Continue working.

#34 - 01/07/2016 11:54 AM - Eugenie Lyzenko

Rebased task branch 2837a from trunk revision 10962, new branch revision is 10970.

#35 - 01/07/2016 12:25 PM - Eugenie Lyzenko

I think that `displayPhysicalLocation()` returns the absolute coordinates with respect to the screen and it takes into account the location of the parent window that is the location of the canvas.

In 1811t I have fixed the location of the canvas if the window is moving. JS module `p2j.mouse.js` should be changed to send correct window location coordinates

This fix does not help for me.

The only change is the adding:

```
...
    // !!! set the current location in order to send the actual data to java web client side.
    mThis.screenLoc = { "x" : x, "y" : y };
...
```

into `p2j.mouse.js`, `processMouseDrag(mThis, mouseOp)` right? Or I missed for something?

3. OverlayWindow.show() should not have code that is specific to the web-client.

The issue here has been presented on the picture attached. The main window was moved by mouse first, then the drop-down was opened. As we can see the overlay window has the same initial coordinates that was before window move. The method screenPhysicalLocation() for main window does not work properly for web client.

I understand you have a problem that needs a solution. My request is simply that we find a different way to resolve the issue. The approach should not have any code specific to the web-client in OverlayWindow (or any other "common code"). It seems that the web-client driver itself either needs to provide a new interface or a fix is needed in its code to allow it to respond the same way as the Swing driver.

This fix does not help for me.

The only change is the adding:

...

into p2j.mouse.js, processMouseDrag(mThis, mouseOp) right? Or I missed for something?

I think Sergey is referring to this change from task branch 1811t revision 10970:

```
bzr diff -c10970 src/com/goldencode/p2j/ui/client/gui/driver/web/WebMouseHandler.java
=== modified file 'src/com/goldencode/p2j/ui/client/gui/driver/web/WebMouseHandler.java'
--- src/com/goldencode/p2j/ui/client/gui/driver/web/WebMouseHandler.java      2015-12-01 13:37:14 +0000
+++ src/com/goldencode/p2j/ui/client/gui/driver/web/WebMouseHandler.java      2016-01-04 08:14:16 +0000
@@ -13,6 +13,7 @@
 ** 002 CA 20150925 Fixed mouse entry/exit: explicit source is passed to the MouseEvt.
 ** 003 SBI 20151201 Fixed to deliver mouse events to their target widgets (JS client has
 **                      no information about child-parent relations, only widgets z-order.)
+** 004 SBI 20160104 Fixed the cursor positioning within the editor.
 */

package com.goldencode.p2j.ui.client.gui.driver.web;
@@ -51,10 +52,10 @@
    Widget w = tc.getWidget(widgetId);
    if (w instanceof AbstractContainer)
    {
-        NativePoint origin = w.displayPhysicalLocation();
+        // It needs to get the widget's location with respect to the parent window.
+        NativePoint origin = w.screenPhysicalLocation();
        Widget target = ((AbstractContainer) w).findMouseSource(
            new NativePoint(event.getX() - origin.x, event.getY() - origin.y));
-
        if (target != null && target.getId() != null)
```

```
{  
    widgetId = target.getId().asInt();
```

#38 - 01/07/2016 03:01 PM - Sergey Ivanovskiy

Eugenie Lyzenko wrote:

I think that `displayPhysicalLocation()` returns the absolute coordinates with respect to the screen and it takes into account the location of the parent window that is the location of the canvas.
In 1811t I have fixed the location of the canvas if the window is moving. JS module `p2j.mouse.js` should be changed to send correct window location coordinates

This fix does not help for me.

The only change is the adding:

[...]

into `p2j.mouse.js`, `processMouseDown(mThis, mouseOp)` right? Or I missed for something?

Before you wrote that `screenPhysicalLocation()` doesn't work correctly and that for your problem it is required to get the widget's absolute location within the browser, but according to the `screenPhysicalLocation()` code it returns the widget's location within the parent window. I need some time to resolve the proposed 2837a changes in order to be helpful.

#39 - 01/07/2016 03:53 PM - Eugenie Lyzenko

Before you wrote that `screenPhysicalLocation()` doesn't work correctly and that for your problem it is required to get the widget's absolute location within the browser, but according to the `screenPhysicalLocation()` code it returns the widget's location within the parent window. I need some time to resolve the proposed 2837a changes in order to be helpful.

Actually what I need is the absolute coordinates meaning the position when the (0,0) is the left top pixel of the OS display screen.

#40 - 01/07/2016 04:14 PM - Eugenie Lyzenko

I understand you have a problem that needs a solution. My request is simply that we find a different way to resolve the issue. The approach should not have any code specific to the web-client in OverlayWindow (or any other "common code"). It seems that the web-client driver itself either needs to provide a new interface or a fix is needed in its code to allow it to respond the same way as the Swing driver.

I see. And of course the common code will not have such GUI specific code. It is temporary fix until the real solution is ready.

#41 - 01/07/2016 07:29 PM - Eugenie Lyzenko

- File *cbb_test21_p2j_web_fix_20160107.jpg* added

Task branch 2837a for review updated to revision 10971.

This is the fix for incorrect drop-down overlay location for web client. Also the web related code removed from "common" calls. The picture for fixed overlay is attached here.

The activation issue is still happening sometimes, need additional investigation, continue working.

#42 - 01/08/2016 07:36 AM - Eugenie Lyzenko

Hynek Cihlar wrote:

5. In `DropDownGuiImpl.findMouseSource()`

```
if (evtID == MouseEvent.MOUSE_CLICKED)
{
    mousePressedSource = null;
}
```

is never executed.

Do you mean `OverlayWindow.findMouseSource()`, right?

#43 - 01/08/2016 08:17 AM - Hynek Cihlar

Eugenie Lyzenko wrote:

Hynek Cihlar wrote:

5. In `DropDownGuiImpl.findMouseSource()`

[...]
is never executed.

Do you mean `OverlayWindow.findMouseSource()`, right?

Yes, sorry, I meant `OverlayWindow`.

#44 - 01/08/2016 08:40 AM - Eugenie Lyzenko

- File *cbb_test21_p2j_web_initial_20160108.jpg* added

Another question for web client. Consider the picture attached. This is how the main window is initially located. The `physicalLocation()` returns (0,0) but as you can see there is a (X,Y) shift (I marked in red) related to the left, top canvas point in the browser. If we move the window inside browser the coordinate become in sync with what `physicalLocation()` returns. So the question is why initially the window is not located in exact top-left corner and where the code that places the window in this initial location? This is the bug or expected behavior?

#45 - 01/08/2016 08:55 AM - Sergey Ivanovskiy

Sergey Ivanovskiy wrote:

Eugenie Lyzenko wrote:

Another question for web client. Consider the picture attached. This is how the main window is initially located. The `physicalLocation()` returns (0,0) but as you can see there is a (X,Y) shift (I marked in red) related to the left, top canvas point in the browser. If we move the window inside browser the coordinate become in sync with what `physicalLocation()` returns. So the question is why initially the window is not located in exact top-left corner and where the code that places the window in this initial location? This is the bug or expected behavior?

For the Firefox it is available Inspector developer tool. I advise you to inspect your page with this tool. It can help to find the actual sizes and locations of html elements within the browser. For my example `widgets_demo.p` has correct location with respect to the report produced by Inspector, the canvas has location 0x0 but is displayed several pixels to the right from the browser's left side. Thus I think it is correct.

Yes, you are right, the initial location doesn't set, I have checked that certainly it is a bug.

#46 - 01/08/2016 08:59 AM - Greg Shah

Yes, you are right, the initial location doesn't set, I have checked that certainly it is a bug.

Eugenie: does this bug block you?

Sergey: please go ahead and fix this. If Eugenie doesn't need it for this task, then put the fix into 1811t. Otherwise it should go into 2837a.

#47 - 01/08/2016 10:08 AM - Eugenie Lyzenko

...For my example widgets_demo.p has correct location with respect to the report produced by Inspector, the canvas has location 0x0 but is displayed several pixels to the right from the browser's left side.

Sergey,

So this shift must be the initial coordinates, correct?

#48 - 01/08/2016 10:13 AM - Eugenie Lyzenko

Eugenie: does this bug block you?

Not really. I can make the web related code assuming the fix already implemented. And when it will be ready - everything become OK.

#49 - 01/08/2016 10:17 AM - Sergey Ivanovskiy

Sergey,

So this shift must be the initial coordinates, correct?

I am not sure because I have only started this bug and will do it as soon as possible.

#50 - 01/08/2016 10:47 AM - Sergey Ivanovskiy

Yes, you are correct, this bug is due to this code in the GuiWebEmulatedWindow

```
case SET_LOCATION:
    if (x != ps.x || y != ps.y)
    {
        x = ps.x;
        y = ps.y;
        websock.setWindowLocation(x, y);
    }
    break;
```

Thus the initial location that is usually (0,0) is not set. Does it need to commit the fix for this bug into 2837a or 1811t?

#51 - 01/08/2016 10:58 AM - Sergey Ivanovskiy

Eugenie, we can add boolean initLocation flag to set the initial location

```
case SET_LOCATION:
    if (x != ps.x || y != ps.y || initLocation)
    {
        initLocation = false;
        x = ps.x;
        y = ps.y;
        websock.setWindowLocation(x, y);
    }
    break;
```

#52 - 01/08/2016 11:02 AM - Eugenie Lyzenko

Another point to discuss here. Assume the web client window with opened drop-down. Then we click somewhere outside the web browser. What is the correct reaction for drop-down in this case? It must be closed or remain opened?

Asking because if we need the drop-down to be closed - we need the ability to notify WebGuiDriver when the activation target is outside the p2j web client. Currently we only have activation method that is called when activation/focus is moved from one p2j web client window to another. The question: is it possible to track this event? Like we are trying to implement this for swing client.

#53 - 01/08/2016 11:14 AM - Greg Shah

Does it need to commit the fix for this bug into 2837a or 1811t?

1811t is fine.

#54 - 01/08/2016 11:16 AM - Greg Shah

It must be closed or remain opened?

I think it can stay open.

#55 - 01/08/2016 11:25 AM - Sergey Ivanovskiy

- *File initLocation_1.txt added*

The incorrect initial window location is fixed in revision 10975 (1811t).

#56 - 01/08/2016 01:09 PM - Eugenie Lyzenko

Task branch 2837a for review updated to revision 10972. Initial window location fix is not included.

I think it can stay open.

So this is the release candidate for review. If the window location fix is working all other issues are seems to be OK. Or you need the window location fix to be included into 2837a?

#57 - 01/08/2016 04:57 PM - Eugenie Lyzenko

Tested the fix from 1811t with overlay window changes. Confirm the issue resolution for drop-down initial location.

#58 - 01/08/2016 05:22 PM - Eugenie Lyzenko

Task branch 2837a for review updated to revision 10973.

Minor comment syntax fix and removed dead drop-down related code from WindowGuiImpl. So if there are no notes it is ready for regression testing.

#59 - 01/09/2016 10:42 AM - Eugenie Lyzenko

Hynek Cihlar wrote:

7. TC.waitForWorker(), line 10792. What is the use case for the overlay window invalidation?

When it is active and "current".

#60 - 01/09/2016 05:24 PM - Eugenie Lyzenko

Task branch 2837a for review updated to revision 10974.

This is additional comments cleanup, removing not required method isModal() and changing the call to OverlayWindow constructor.

6. OverlayWindow has a lot in common with ModalWindow, it would probably make sense to derive a common parent for them. A suggestion for the future development.

Do we need this change now? These two types of windows(OverlayWindow and ModalWindow) have more differences than commons. In the code that really working the behavior is different.

#61 - 01/10/2016 08:42 AM - Hynek Cihlar

Eugenie Lyzenko wrote:

Hynek Cihlar wrote:

7. TC.waitForWorker(), line 10792. What is the use case for the overlay window invalidation?

When it is active and "current".

Since the overlay window is always active, this doesn't narrow the use case where the TC invalidation would be useful. When the call to setInvalidate() for the overlay window is removed, the window redraws itself just fine. Hence I think it is not needed and the call should be removed.

I found two issues with the combo simple mode.

(1)

An exception when scroll bar up button is quickly pressed multiple times:

```
Caused by: java.lang.ArrayIndexOutOfBoundsException: -1
  at java.util.ArrayList.elementData(ArrayList.java:418)
  at java.util.ArrayList.get(ArrayList.java:431)
  at com.goldencode.p2j.ui.client.model.ListModel.get(ListModel.java:96)
  at com.goldencode.p2j.ui.client.ScrollableList.getItemText(ScrollableList.java:154)
  at com.goldencode.p2j.ui.client.gui.ScrollableSelectionListGuiImpl.drawItem(ScrollableSelectionListGuiImpl.java:628)
  at com.goldencode.p2j.ui.client.gui.ScrollableSelectionListGuiImpl.access$5(ScrollableSelectionListGuiImpl.java:604)
  at com.goldencode.p2j.ui.client.gui.ScrollableSelectionListGuiImpl$2.run(ScrollableSelectionListGuiImpl.java:226)
  at com.goldencode.p2j.ui.client.gui.driver.AbstractGuiDriver.draw(AbstractGuiDriver.java:1988)
  at com.goldencode.p2j.ui.client.gui.ScrollableSelectionListGuiImpl.draw(ScrollableSelectionListGuiImpl.java:187)
  at com.goldencode.p2j.ui.client.widget.Viewport.draw(Viewport.java:64)
```

(2)

Simple list doesn't scroll when scroll bar buttons are clicked. This can be seen in demo/demo_widgets.p when the combo there is changed to simple mode. When demo_widgets.p is run, try to scroll the list in the lower frame. Or in the upper frame click into the list and then try to scroll with the scroll bar buttons. In both cases the list won't scroll.

a correction. The above is not combo specific, it can be reproduced in other widgets as well.

Please clarify. Other like what? What widgets are problematic?

The problem seems to be in ScrollBarGuiImpl or some of its sub-components. Thus any widget showing scroll bars is infected. I'll look at it.

For the issue (2) this seems to be a regression from some of the recent changes. Because the issue exists in combo_box/combo_box10.p and worked fine when I've committed the simple mode fixes.

I was able to reproduce the problem with the trunk revision containing the simple mode fixes. Maybe this will help you pinpoint the problem.

Task branch 2837a for review updated to revision 10975.

This is the fix for issue(2) above.

#63 - 01/10/2016 01:51 PM - Eugenie Lyzenko

Hynek,

When it is active and "current".

Since the overlay window is always active, this doesn't narrow the use case where the TC invalidation would be useful. When the call to `setInvalidate()` for the overlay window is removed, the window redraws itself just fine. Hence I think it is not needed and the call should be removed.

The idea of this change was to protect the opened overlay window from accidental activation of the main window if for some reasons the main window invalidation and repaint generate main window activation(which in turn cause the overlay window to dismiss). Do you think this can never be happen?

#64 - 01/10/2016 02:47 PM - Hynek Cihlar

Eugenie Lyzenko wrote:

Hynek,

When it is active and "current".

Since the overlay window is always active, this doesn't narrow the use case where the TC invalidation would be useful. When the call to `setInvalidate()` for the overlay window is removed, the window redraws itself just fine. Hence I think it is not needed and the call should be removed.

The idea of this change was to protect the opened overlay window from accidental activation of the main window if for some reasons the main window invalidation and repaint generate main window activation(which in turn cause the overlay window to dismiss). Do you think this can never be happen?

I see. To my best knowledge, window invalidation/repaint should not cause the window to get active.

#65 - 01/10/2016 03:13 PM - Eugenie Lyzenko

I see. To my best knowledge, window invalidation/repaint should not cause the window to get active.

OK. Task branch 2837a for review updated to revision 10976. I have removed ThinClient from update set.

#66 - 01/10/2016 05:14 PM - Sergey Ivanovskiy

Eugenie Lyzenko wrote:

I found two issues with the combo simple mode.

(1)

An exception when scroll bar up button is quickly pressed multiple times:

```
Caused by: java.lang.ArrayIndexOutOfBoundsException: -1
    at java.util.ArrayList.elementData(ArrayList.java:418)
    at java.util.ArrayList.get(ArrayList.java:431)
    at com.goldencode.p2j.ui.client.model.ListModel.get(ListModel.java:96)
    at com.goldencode.p2j.ui.client.ScrollableList.getItemText(ScrollableList.java:154)
    at com.goldencode.p2j.ui.client.gui.ScrollableSelectionListGuiImpl.drawItem(ScrollableSelectionListGuiImpl
.java:628)
    at com.goldencode.p2j.ui.client.gui.ScrollableSelectionListGuiImpl.access$5(ScrollableSelectionListGuiImpl
.java:604)
    at com.goldencode.p2j.ui.client.gui.ScrollableSelectionListGuiImpl$2.run(ScrollableSelectionListGuiImpl.ja
va:226)
    at com.goldencode.p2j.ui.client.gui.driver.AbstractGuiDriver.draw(AbstractGuiDriver.java:1988)
    at com.goldencode.p2j.ui.client.gui.ScrollableSelectionListGuiImpl.draw(ScrollableSelectionListGuiImpl.jav
a:187)
    at com.goldencode.p2j.ui.client.widget.Viewport.draw(Viewport.java:64)
```

This exception looks like [#2924](#) change-36125 (LE: GES what is this referencing?).
and if this is the same bug, then it has been fixed in 1811t.

#67 - 01/10/2016 05:28 PM - Eugenie Lyzenko

Sergey Ivanovskiy wrote:

Eugenie Lyzenko wrote:

An exception when scroll bar up button is quickly pressed multiple times:

[...]

This exception looks like [#2924](#) change-36125
and if this is the same bug, then it has been fixed in 1811t.

Yes, the bug is the same and your change fixes it, thanks for help.

#68 - 01/11/2016 03:56 PM - Greg Shah

Code Review Task Branch 2837a Revision 10976

The changes look good.

You can regression test these. Please note that if Hynek's 2875a passes testing, it will be merged to trunk first, so you might have to re-do testing if you go ahead now.

#69 - 01/11/2016 05:22 PM - Eugenie Lyzenko

Greg Shah wrote:

Code Review Task Branch 2837a Revision 10976

You can regression test these. Please note that if Hynek's 2875a passes testing, it will be merged to trunk first, so you might have to re-do testing if you go ahead now.

OK. The testing is in progress.

#70 - 01/11/2016 06:15 PM - Eugenie Lyzenko

Task branch 2837a for review updated to revision 10977.

The only change is adding unnecessary activation of the already active window for web client GuiWebDriver. In a rare cases on fast enough systems we have double call to windowActivated() for already active window. And if the overlay is opened it will incorrectly dismiss overlay.

If no objection I will replace 2837a with 10977 in regression testing cycle.

#71 - 01/12/2016 03:58 AM - Eugenie Lyzenko

The CTRL-C tests are OK (with 3-way additional standalone run). The main part has some badge reader failures, starting another main round.

#72 - 01/12/2016 09:16 AM - Greg Shah

Code Review Task Branch 2837a Revision 10977

I'm fine with the change.

You don't have to restart testing for this change, since this code cannot be reached in MAJIC regression testing.

#73 - 01/12/2016 01:24 PM - Eugenie Lyzenko

Greg Shah wrote:

Code Review Task Branch 2837a Revision 10977

I'm fine with the change.

You don't have to restart testing for this change, since this code cannot be reached in MAJIC regression testing.

Testing completed. No regressions.

I'm going to merge into trunk. Is it OK?

#74 - 01/12/2016 01:31 PM - Greg Shah

I'm going to merge into trunk. Is it OK?

Sorry, no. Recall this from note 68:

Please note that if Hynek's 2875a passes testing, it will be merged to trunk first, so you might have to re-do testing if you go ahead now.

2875a has passed testing and is being merged to trunk right now.

Please rebase when you see the merge notification and then re-run regression testing. Your update will be merged to trunk next.

#75 - 01/12/2016 01:38 PM - Eugenie Lyzenko

Greg Shah wrote:

2875a has passed testing and is being merged to trunk right now.

Please rebase when you see the merge notification and then re-run regression testing. Your update will be merged to trunk next.

OK. The current result is 2837a_10977_de19d84_20160112_evl.zip.

#76 - 01/12/2016 03:53 PM - Eugenie Lyzenko

Rebasing to 10963 from 2875a causes the serious regressions with overlay window implementation. I need a time to debug.

Moreover in the web client 10963 gives the main window to be initially inactive(according to window title background color). I have found it in 2837a rebased code. Not sure if this issue in the trunk now or not.

#77 - 01/12/2016 03:55 PM - Hynek Cihlar

Eugenie Lyzenko wrote:

Rebasing to 10963 from 2875a causes the serious regressions with overlay window implementation. I need a time to debug.

What is the nature of the problem?

Moreover in the web client 10963 gives the main window to be initially inactive(according to window title background color). I have found it in 2837a rebased code. Not sure if this issue in the trunk now or not.

What is the test where you see this?

#78 - 01/12/2016 04:08 PM - Eugenie Lyzenko

- File `gui_btn_test5_p2j_20160112.jpg` added

Hynek Cihlar wrote:

What is the nature of the problem?

The overlay window does not receive window deactivation event anymore when clicking on main window title for both web and swing clients. The same is true when activating any other non-P2J window for swing client.

Moreover in the web client 10963 gives the main window to be initially inactive (according to window title background color). I have found it in 2837a rebased code. Not sure if this issue is in the trunk now or not.

What is the test where you see this?

`gui_btn_test5.p` for example.

#79 - 01/12/2016 04:27 PM - Hynek Cihlar

Eugenie Lyzenko wrote:

Hynek Cihlar wrote:

What is the nature of the problem?

The overlay window does not receive window deactivation event anymore when clicking on main window title for both web and swing clients. The same is true when activating any other non-P2J window for swing client.

Try the Swing client first, check that `windowGainedFocus()` and `windowLostFocus()` are invoked in `SwingEmulatedWindow` and trace how are the calls dispatched.

#80 - 01/12/2016 05:12 PM - Eugenie Lyzenko

Rebased task branch 2837a from trunk revision 10963, new branch revision is 10978.

Merge verification completed. The debugging and regression fixing is in progress.

#81 - 01/12/2016 05:34 PM - Eugenie Lyzenko

Another regression has been encountered, swing client:

```
[01/13/2016 01:13:32 MSK] (StandardServer.invoke:SEVERE) {00000001:00000009:bogus} Abnormal end!
java.lang.reflect.InvocationTargetException
    at sun.reflect.NativeMethodAccessorImpl.invoke0(Native Method)
    at sun.reflect.NativeMethodAccessorImpl.invoke(NativeMethodAccessorImpl.java:62)
    at sun.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAccessorImpl.java:43)
    at java.lang.reflect.Method.invoke(Method.java:497)
    at com.goldencode.p2j.util.Utills.invoke(Utills.java:1289)
    at com.goldencode.p2j.main.StandardServer$MainInvoker.execute(StandardServer.java:1767)
    at com.goldencode.p2j.main.StandardServer.invoke(StandardServer.java:1270)
    at com.goldencode.p2j.main.StandardServer.standardEntry(StandardServer.java:427)
    at sun.reflect.NativeMethodAccessorImpl.invoke0(Native Method)
    at sun.reflect.NativeMethodAccessorImpl.invoke(NativeMethodAccessorImpl.java:62)
    at sun.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAccessorImpl.java:43)
    at java.lang.reflect.Method.invoke(Method.java:497)
    at com.goldencode.p2j.util.MethodInvoker.invoke(MethodInvoker.java:76)
    at com.goldencode.p2j.net.Dispatcher.processInbound(Dispatcher.java:705)
    at com.goldencode.p2j.net.Conversation.block(Conversation.java:319)
    at com.goldencode.p2j.net.Conversation.run(Conversation.java:163)
    at java.lang.Thread.run(Thread.java:745)
Caused by: java.lang.RuntimeException: No renderer is registered for id = -57
    at com.goldencode.p2j.ui.client.gui.driver.GuiPrimitivesImpl.getWindowEmulator(GuiPrimitivesImpl.java:
208)
    at com.goldencode.p2j.ui.client.gui.driver.GuiPrimitivesImpl.selectWindow(GuiPrimitivesImpl.java:275)
    at com.goldencode.p2j.ui.client.gui.driver.AbstractGuiDriver.selectWindow(AbstractGuiDriver.java:1708)
    at com.goldencode.p2j.ui.client.gui.driver.GuiOutputManager.selectWindow(GuiOutputManager.java:284)
    at com.goldencode.p2j.ui.chui.ThinClient.eventDrawingBracket(ThinClient.java:13846)
    at com.goldencode.p2j.ui.chui.ThinClient.independentEventDrawingBracket(ThinClient.java:13734)
    at com.goldencode.p2j.ui.client.gui.DropDownGuiImpl.hide(DropDownGuiImpl.java:272)
    at com.goldencode.p2j.ui.client.ComboBox.exitDropDown(ComboBox.java:1022)
    at com.goldencode.p2j.ui.client.gui.ComboBoxGuiImpl.exitDropDown(ComboBoxGuiImpl.java:902)
    at com.goldencode.p2j.ui.client.ComboBox$2.run(ComboBox.java:1120)
    at com.goldencode.p2j.ui.chui.ThinClient.eventBracket(ThinClient.java:13920)
    at com.goldencode.p2j.ui.chui.ThinClient.eventDrawingBracket(ThinClient.java:13861)
    at com.goldencode.p2j.ui.chui.ThinClient.independentEventDrawingBracket(ThinClient.java:13734)
    at com.goldencode.p2j.ui.client.ComboBox.activate(ComboBox.java:1115)
    at com.goldencode.p2j.ui.client.gui.ComboBoxGuiImpl.mousePressed(ComboBoxGuiImpl.java:423)
    at com.goldencode.p2j.ui.client.gui.driver.MouseHandler.applyMouseEvent(MouseHandler.java:323)
    at com.goldencode.p2j.ui.client.gui.driver.MouseHandler.handleMouseEvent(MouseHandler.java:238)
    at com.goldencode.p2j.ui.client.gui.driver.AbstractGuiDriver.handleMouseEvent(AbstractGuiDriver.java:2
412)
    at com.goldencode.p2j.ui.client.TopLevelWindow.processEvent(TopLevelWindow.java:680)
    at com.goldencode.p2j.ui.client.gui.WindowGuiImpl.processEvent(WindowGuiImpl.java:1334)
    at com.goldencode.p2j.ui.chui.ThinClient.processProgressEvent(ThinClient.java:15308)
    at com.goldencode.p2j.ui.chui.ThinClient.processEventsWorker(ThinClient.java:14931)
    at com.goldencode.p2j.ui.chui.ThinClient.pop(ThinClient.java:13960)
    at com.goldencode.p2j.ui.chui.ThinClient.eventBracket(ThinClient.java:13943)
    at com.goldencode.p2j.ui.chui.ThinClient.eventDrawingBracket(ThinClient.java:13861)
    at com.goldencode.p2j.ui.chui.ThinClient.applyWorker(ThinClient.java:13617)
    at com.goldencode.p2j.ui.chui.ThinClient.waitForWorker(ThinClient.java:10940)
    at com.goldencode.p2j.ui.chui.ThinClient.waitFor(ThinClient.java:10448)
    at com.goldencode.p2j.ui.chui.ThinClient.waitFor(ThinClient.java:10402)
    at sun.reflect.NativeMethodAccessorImpl.invoke0(Native Method)
    at sun.reflect.NativeMethodAccessorImpl.invoke(NativeMethodAccessorImpl.java:62)
    at sun.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAccessorImpl.java:43)
    at java.lang.reflect.Method.invoke(Method.java:497)
    at com.goldencode.p2j.util.MethodInvoker.invoke(MethodInvoker.java:76)
    at com.goldencode.p2j.net.Dispatcher.processInbound(Dispatcher.java:705)
    at com.goldencode.p2j.net.Conversation.block(Conversation.java:319)
    at com.goldencode.p2j.net.Conversation.waitForMessage(Conversation.java:257)
    at com.goldencode.p2j.net.Queue.transactImpl(Queue.java:1128)
    at com.goldencode.p2j.net.Queue.transact(Queue.java:585)
    at com.goldencode.p2j.net.BaseSession.transact(BaseSession.java:223)
```

```
at com.goldencode.p2j.net.HighLevelObject.transact (HighLevelObject.java:163)
at com.goldencode.p2j.net.RemoteObject$RemoteAccess.invokeCore (RemoteObject.java:1425)
at com.goldencode.p2j.net.InvocationStub.invoke (InvocationStub.java:97)
at com.sun.proxy.$Proxy4.standardEntry (Unknown Source)
at com.goldencode.p2j.main.ClientCore.start (ClientCore.java:281)
at com.goldencode.p2j.main.ClientCore.start (ClientCore.java:102)
at com.goldencode.p2j.main.ClientDriver.start (ClientDriver.java:201)
at com.goldencode.p2j.main.CommonDriver.process (CommonDriver.java:398)
at com.goldencode.p2j.main.ClientDriver.process (ClientDriver.java:95)
at com.goldencode.p2j.main.ClientDriver.main (ClientDriver.java:267)
```

#82 - 01/12/2016 06:17 PM - Eugenie Lyzenko

Hynek,

For your current activation change I need to exactly know:

1. What are the cases when window gets activation, explicit or implicit?
2. What are the cases when window loses activation, explicit or implicit?
3. What is the complete message set sending to window that become active?
4. What is the complete message set sending to window that become inactive?
5. What is the order of the messages in point 3 and 4 above? I need exact sequence.

I do not need the perfect model of what the activation engine should be. I need the current state that was changed in the 10963 update. Or difference with what was before this update.

#83 - 01/12/2016 06:49 PM - Eugenie Lyzenko

Hynek,

Another questions:

What was the reason to remove focus monitoring synchronization approach in SwingEmulatedWindow? How the incoming Swing focus events are now synchronized? The SwingEmulatedWindow now starts running to handle windowLostFocus() while the P2J TopLevelWindow handles WindowActivation to make the main window active. How this thread racing is now resolved?

All these are critical for OverlayWindow with DropDownGuiImpl because we have to immediately react to main window activation event when drop-down is opened.

#84 - 01/12/2016 08:38 PM - Eugenie Lyzenko

The call in SwingEmulatedWindow constructor:

```
...
    window.setAutoRequestFocus(false);
...
```

terminates the last opportunity to close the drop-down overlay window by clicking the main window title. Because no focus gaining event is generated. We need to turn it on again to be able to work with GUI combo-boxes.

#85 - 01/13/2016 04:09 AM - Hynek Cihlar

- *File let_driver_post_the_activated_event.diff added*

You should not post the WindowActivated event yourself, let the driver do it for you. I modified your changes and I am attaching the diff here.

#86 - 01/13/2016 04:14 AM - Hynek Cihlar

Eugenie Lyzenko wrote:

Hynek,

For your current activation change I need to exactly know:

1. What are the cases when window gets activation, explicit or implicit?
2. What are the cases when window loses activation, explicit or implicit?
3. What is the complete message set sending to window that become active?
4. What is the complete message set sending to window that become inactive?
5. What is the order of the messages in point 3 and 4 above? I need exact sequence.

Window is activated when (1) user focuses the window (mouse, keyboard) or (2) when the app makes a request through WindowManager.activateWindow(). If you need to see the exact event sequence I suggest you to log the WindowActivated constructor and TopLevelWindow.processEvent().

#87 - 01/13/2016 04:18 AM - Hynek Cihlar

Eugenie Lyzenko wrote:

Hynek,

Another questions:

What was the reason to remove focus monitoring synchronization approach in SwingEmulatedWindow? How the incoming Swing focus events are now synchronized? The SwingEmulatedWindow now starts running to handle windowLostFocus() while the P2J TopLevelWindow handles WindowActivation to make the main window active. How this thread racing is now resolved?

Yes, proper event synchronization is important. 2875a brings a lot of improvements to this. For example all Swing "write" operations are performed on the Swing event dispatching thread, events are properly dispatched through event queues, etc.

#88 - 01/13/2016 01:46 PM - Eugenie Lyzenko

Task branch 2837a for review updated to revision 10979.

Fixed TopLevelWindow source corruption in branch.

#89 - 01/13/2016 03:26 PM - Hynek Cihlar

Please review the various window-activation improvements checked in to 2837a revision 10980.

#90 - 01/13/2016 05:38 PM - Eugenie Lyzenko

Please review the various window-activation improvements checked in to 2837a revision 10980.

On the first look I do not have any objections. I need to integrate them into my current code to see if something is changing or not.

Do you have any particular reasons to make these changes? Something was broken before is now working properly? Or this is just a kind of optimization?

I can explain the situation I'm currently struggling with. When the drop-down is become active and shown the SwingEmulatedWindow the OverlayWindow receives the deactivation and terminates itself. The reason is the SwingEmulatedWindow is calling from Swing with sequential gaining/losing/gaining focus messages.

And this is not we can directly control, right? Generally speaking we have asynchronous message processing here for activation/focus. On the one hand there is external event producer(Swing thread) and another thread is consumer(P2J event processing loop). As soon as the SwingEmulatedWindow get called from Swing it posts the OS event and go further. Some events processing order should be preserved here. But the consumer thread also can generate the events and push them to the consumer thread. And it is possible to have interference in consumer's event processing. That can cause the wrong behavior sometimes.

Previously(before recent trunk changes) this resolved by locking the Swing producer thread for some time and disabling the SwingEmulatedWindow code from pushing messages for several P2J operations(like moving window to the top). Currently we do not ignore any incoming Swing events and need to handle this focus "bouncing". Again the only way I can dismiss the drop-down for pressing mouse on window title is to react to the deactivation message. So we have to deactivate the overlay only when we want to close it. The code like focus-unfocus-focus in SwingEmulatedWindow is not acceptable for OverlayWindow.

#91 - 01/13/2016 06:47 PM - Eugenie Lyzenko

Hynek Cihlar wrote:

Please review the various window-activation improvements checked in to 2837a revision 10980.

I'm fine with the update, no any behavior changes for overlay window issues.

Although I'm still waiting for clarification about what these updates are really fix/change.

Please let's append any changes in this area with some explanation for idea behind. This is very sensitive task and I think it will be good for both of us if we will explain what we are doing.

#92 - 01/14/2016 12:02 AM - Eugenie Lyzenko

Task branch 2837a for review updated to revision 10981.

Fixes for issues from recent rebase. Still containing debug calls. Because the Web client is not always working properly. The drop-down overlay is closing just after opening. Looks main window gaining the activation back. Debugging to get a fix.

- The main idea of the fixes is to introduce new API to GuiDriver to turn the extended focus generation back for overlay drop-down window. The main window keeps the current off value for this `window.setAutoRequestFocus(false)`; call.
- Also the window title is repainting if activation change does not work with overlay, otherwise we have the confused title background working with combo-box.
- Initial focus request for GUI drop-down is now disabled because it is not necessary due to overlay activation.

The swing client is now working properly.

For web client the interesting fact is the overlay opened properly on mouse press but mouse release event causes the overlay to deactivate and then incorrectly dismisses the drop-down.

#93 - 01/14/2016 03:56 AM - Hynek Cihlar

Eugenie as I have mentioned in note #85 you should not post the WindowActivated events yourself but let the GUI driver to do this job. I have also provided you with a diff in the note so you could see how to handle the window activated events properly. Anyway, I am fixing the latest revision 10981, meanwhile please do not check in anything into the branch.

#94 - 01/14/2016 08:51 AM - Hynek Cihlar

Revision 10982 fixes Overlay window activation for Swing GUI. There are some issues that must be addressed though. These issues arise from the additional special behavior of Overlay window.

Overlay window is a top-level window with special properties:

- It doesn't deactivate its owner window, the title bar of the owner is drawn as active,
- there can be maximum of one overlay window at any given time,
- the overlay window must not appear on the task bar.

The key point here in respect to the current state of the code is that the overlay window should not deactivate the owner window, this is needs to be implement, among others. I believe that for this feature we need a little help from the GUI driver. The driver needs to know whether a window being activated is an overlay window or not and send activation events accordingly.

The sequence of window activation events should be as follows: Main window A is switched to by the user, the driver sends window-activated event

for A, user clicks on the combo box icon, the app logic shows the overlay window B with the drop-down, the driver sends window-activated event for B (note that no window-deactivated for A), user clicks outside of B but inside of A, the driver sends window-deactivated event to B and again no window-activated for A because A was already active. Window A in this case is active all the time - even when an overlay window is active.

In terms of implementation, I think this is what we need:

- WindowManager.isWindowActive must be extended to properly handle the case with overlay window on,
- Swing GUI driver needs to be taught not to send deactivation for the owner of an overlay window,
- Make the Web GUI driver compatible with Swing driver.

Eugenie, did I miss anything?

#95 - 01/14/2016 10:33 AM - Eugenie Lyzenko

Task branch 2837a for review updated to revision 10983.

Eugenie, did I miss anything?

One important thing:

1. The only event we can have from the mouse press on window title - window activation. We can get mouse click but it is too late and dragging window does not generate click.
2. We can not have two active windows at the same time.
3. If we do not activate overlay we have no ability to know when the main window gets the mouse press and have no ability to close overlay.

Note these considerations are about only window title. Pressing inside the window works fine.

Also I have restored the conditional title repaint because inactive window title while drop-down opened confuses the user.

Working on web client stable combo opening.

#96 - 01/14/2016 10:57 AM - Hynek Cihlar

Eugenie Lyzenko wrote:

Task branch 2837a for review updated to revision 10983.

Eugenie, did I miss anything?

One important thing:

1. The only event we can have from the mouse press on window title - window activation. We can get mouse click but it is too late and dragging window does not generate click.

We should only handle window activation events. WindowActivated is sent regardless the point where you click in the window.

2. We can not have two active windows at the same time.

Yes, while we cannot have two activated windows at the driver level (at least this is the case for Swing UI), on the application level (where we implement P2J widgets) we need to act as if the two windows are active. This is how the native 4GL works - when you open the drop down, the main window doesn't deactivate and no LEAVE message is sent to the 4GL program.

3. If we do not activate overlay we have no ability to know when the main window gets the mouse press and have no ability to close overlay.

Again, no need to care about mouse. We need to care about window events.

Also I have restored the conditional title repaint because inactive window title while drop-down opened confuses the user.

The conditional title repaint is not needed and should be removed, instead the main window should not be deactivated, see above.

#97 - 01/14/2016 11:25 AM - Eugenie Lyzenko

Hynek Cihlar wrote:

3. If we do not activate overlay we have no ability to know when the main window gets the mouse press and have no ability to close overlay.

Again, no need to care about mouse. We need to care about window events.

But we have to care about mouse press, not click, while overlay is opened.

Pressing mouse on window title must close opened drop-down, correct?

If the window is active pressing mouse on window title does not generate window activation event, correct?

So in this case we do not have the ability to close overlay by pressing mouse on the window title, correct?

#98 - 01/14/2016 11:56 AM - Hynek Cihlar

Eugenie Lyzenko wrote:

Hynek Cihlar wrote:

3. If we do not activate overlay we have no ability to know when the main window gets the mouse press and have no ability to close overlay.

Again, no need to care about mouse. We need to care about window events.

But we have to care about mouse press, not click, while overlay is opened.

Pressing mouse on window title must close opened drop-down, correct?

Yes.

If the window is active pressing mouse on window title does not generate window activation event, correct?

Yes. But the main window will not be active on the driver's level. The active window will be the overlay. Hence clicking outside anywhere will cause the driver to send window-deactivated to the overlay window and this what we handle.

#99 - 01/14/2016 12:36 PM - Eugenie Lyzenko

Yes. But the main window will not be active on the driver's level. The active window will be the overlay. Hence clicking outside anywhere will cause the driver to send window-deactivated to the overlay window and this what we handle.

And this is how it currently implemented.

#100 - 01/14/2016 12:55 PM - Hynek Cihlar

Eugenie Lyzenko wrote:

Yes. But the main window will not be active on the driver's level. The active window will be the overlay. Hence clicking outside anywhere will cause the driver to send window-deactivated to the overlay window and this what we handle.

And this is how it currently implemented.

The important part is that when you click outside the overlay window (regardless whether window content, title, etc.) the overlay window must receive WindowActivated event. No need to handle mouse events.

#101 - 01/14/2016 02:36 PM - Eugenie Lyzenko

Task branch 2837a for review updated to revision 10984.

This update includes web client overlay window compatibility support. The idea is to disable focus listener for click event. It is enough to have this logic on press event. Having in click too causes another window activation and this breaks overlay window.

So if there are no objection the update is ready to be tested and committed.

And yes, the window title is not repainting when drop-down overlay window is opening.

#102 - 01/14/2016 03:02 PM - Greg Shah

Code Review Task Branch 2837a Revisions 10980 through 10984

I'm fine with the accumulated changes. Eugenie: go ahead and start testing.

Hynek: any objections or concerns?

#103 - 01/14/2016 03:56 PM - Eugenie Lyzenko

Greg Shah wrote:

Code Review Task Branch 2837a Revisions 10980 through 10984

I'm fine with the accumulated changes. Eugenie: go ahead and start testing.

OK. The main cycle has been started.

- File drop-down_inverted.png added

Greg Shah wrote:

Code Review Task Branch 2837a Revisions 10980 through 10984

I'm fine with the accumulated changes. Eugenie: go ahead and start testing.

Hynek: any objections or concerns?

Yes, I have some concerns.

(1) Consider this case. User activates the drop down and then directly switches to another main window. The main window containing the activated dropdown should receive a LEAVE 4GL event. Currently it doesn't because it is not active and thus doesn't receive WindowActivated event to indicate window deactivation, LEAVE/POST 4GL events are managed by the WindowActivated handler in TopLevelWindow.

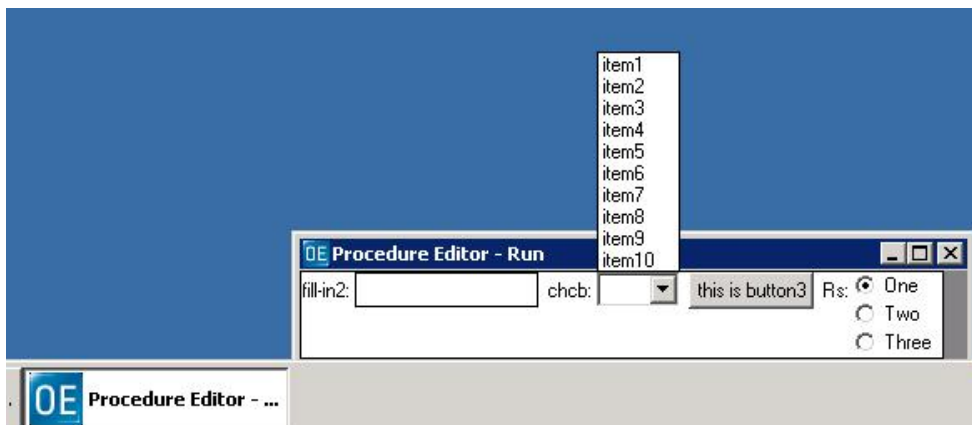
(2) Focus cycling is missing. When TAB is typed in the drop-down the focus should move to the next widget in the top-level window of the combo box widget.

(3) Occasionally the app abends when combo box drop-down button is clicked. I was able to reproduce it with two combos and clicking the drop down buttons back and forth.

```
Caused by: java.lang.RuntimeException: No renderer is registered for id = -269
    at com.goldencode.p2j.ui.client.gui.driver.GuiPrimitivesImpl.getWindowEmulator(GuiPrimitivesImpl.java:208)
    at com.goldencode.p2j.ui.client.gui.driver.GuiPrimitivesImpl.selectWindow(GuiPrimitivesImpl.java:275)
    at com.goldencode.p2j.ui.client.gui.driver.AbstractGuiDriver.selectWindow(AbstractGuiDriver.java:1708)
    at com.goldencode.p2j.ui.client.gui.driver.GuiOutputManager.selectWindow(GuiOutputManager.java:284)
    at com.goldencode.p2j.ui.chui.ThinClient.eventDrawingBracket(ThinClient.java:13853)
    at com.goldencode.p2j.ui.chui.ThinClient.independentEventDrawingBracket(ThinClient.java:13741)
    at com.goldencode.p2j.ui.client.gui.DropDownGuiImpl.hide(DropDownGuiImpl.java:273)
    at com.goldencode.p2j.ui.client.ComboBox.exitDropDown(ComboBox.java:1023)
    at com.goldencode.p2j.ui.client.gui.ComboBoxGuiImpl.exitDropDown(ComboBoxGuiImpl.java:902)
    at com.goldencode.p2j.ui.client.ComboBox$2.run(ComboBox.java:1120)
    at com.goldencode.p2j.ui.chui.ThinClient.eventBracket(ThinClient.java:13927)
    at com.goldencode.p2j.ui.chui.ThinClient.eventDrawingBracket(ThinClient.java:13868)
    at com.goldencode.p2j.ui.chui.ThinClient.independentEventDrawingBracket(ThinClient.java:13741)
    at com.goldencode.p2j.ui.client.ComboBox.activate(ComboBox.java:1115)
    at com.goldencode.p2j.ui.client.gui.ComboBoxGuiImpl.mousePressed(ComboBoxGuiImpl.java:423)
    at com.goldencode.p2j.ui.client.gui.driver.MouseHandler.applyMouseEvent(MouseHandler.java:323)
    at com.goldencode.p2j.ui.client.gui.driver.MouseHandler.handleMouseEvent(MouseHandler.java:238)
    at com.goldencode.p2j.ui.client.gui.driver.AbstractGuiDriver.handleMouseEvent(AbstractGuiDriver.java:2412)
    at com.goldencode.p2j.ui.client.TopLevelWindow.processEvent(TopLevelWindow.java:683)
    at com.goldencode.p2j.ui.client.gui.WindowGuiImpl.processEvent(WindowGuiImpl.java:1344)
```

(4) When a drop-down of a combo A is activated clicking on the drop-down button of combo B doesn't bring its drop-down on.

(5) Drop-down position must take up the available space. See the screenshot.



(1) Consider this case. User activates the drop down and then directly switches to another main window. The main window containing the activated dropdown should receive a LEAVE 4GL event...

No it should not. Because the main window containing opened drop-down is not active.

(2) Focus cycling is missing. When TAB is typed in the drop-down the focus should move to the next widget in the top-level window of the combo box widget.

This is not a regression, I think can be fixed in next 2837b.

(3) Occasionally the app abends when combo box drop-down button is clicked. I was able to reproduce it with two combo and clicking the drop down buttons back and forth.

Yes, I think this abend must be fixed within current 2837a branch before commit.

(4) When a drop-down of a combo A is activated clicking on the drop-down button of combo B doesn't bring its drop-down on.

Yes, this is critical and must be fixed within current 2837a.

(5) Drop-down position must take up the available space. See the screenshot.

This is a kind of new feature for drop-down, let's decide when to implement this.

Another point. I have found the issue with new activation. Sometimes initially the main window does not react to keyboard after show(). Try to use combo_box21.p to reproduce. This is not related to overlay/dropdown implementation because detected before overlay code become active.

#106 - 01/15/2016 08:49 AM - Eugenie Lyzenko

The current testing completed without regression. I have stopped the servers.

The next plan is to fix issues (3) and (4) above and restart testing, commit if success. Then we can resolve the rest notes in 2837b. Is is acceptable?

#107 - 01/15/2016 09:17 AM - Greg Shah

(1) Consider this case. User activates the drop down and then directly switches to another main window. The main window containing the activated dropdown should receive a LEAVE 4GL event...

No it should not. Because the main window containing opened drop-down is not active.

I think Hynek is correct here. From the perspective of the 4GL, a secondary overlay window doesn't exist. That is an implementation detail that the Windows OS hides inside the combo-box control. The containing window is definitely activated and in our new implementation, we must not deactivate/activate the containing window when the overlay is shown/dismissed.

However, if the overlay is dismissed in a case that would make the containing window deactivate, then it must deactivate. And when the 4GL window deactivates, it will get a LEAVE event, right?

#108 - 01/15/2016 09:24 AM - Greg Shah

The next plan is to fix issues (3) and (4) above and restart testing, commit if success. Then we can resolve the rest notes in 2837b. Is is acceptable?

Yes.

Hynek also has [#2958](#) which he may include here if it can be done in time.

#109 - 01/15/2016 09:46 AM - Eugenie Lyzenko

I think Hynek is correct here. From the perspective of the 4GL, a secondary overlay window doesn't exist. That is an implementation detail that the Windows OS hides inside the combo-box control. The containing window is definitely activated and in our new implementation, we must not deactivate/activate the containing window when the overlay is shown/dismissed.

However, if the overlay is dismissed in a case that would make the containing window deactivate, then it must deactivate. And when the 4GL window deactivates, it will get a LEAVE event, right?

Right. But in p2j this is true if the window containing opened drop-down is active. So to make this happen we really have to have the main window active when drop-down is opened. This faces us with known constraint: the only way to track mouse press on window title is to see when the window get activated. And when the window is active(and `window.setAutoRequestFocus(false);`) as in Hynek's offered approach - we do not have this message for mouse press. So no ability to close drop-down for mouse press on window title.

#110 - 01/15/2016 11:12 AM - Hynek Cihlar

Eugenie Lyzenko wrote:

I think Hynek is correct here. From the perspective of the 4GL, a secondary overlay window doesn't exist. That is an implementation detail that the Windows OS hides inside the combo-box control. The containing window is definitely activated and in our new implementation, we must not deactivate/activate the containing window when the overlay is shown/dismissed.

However, if the overlay is dismissed in a case that would make the containing window deactivate, then it must deactivate. And when the 4GL window deactivates, it will get a LEAVE event, right?

Right. But in p2j this is true if the window containing opened drop-down is active. So to make this happen we really have to have the main window active when drop-down is opened. This faces us with known constraint: the only way to track mouse press on window title is to see when the window get activated. And when the window is active(and `window.setAutoRequestFocus(false);`) as in Hynek's offered approach - we do not have this message for mouse press. So no ability to close drop-down for mouse press on window title.

We should be ok here. When the overlay window is on it is also active at the driver level. Clicking outside of the overlay window will cause the driver to send a window-deactivate for the overlay, and the handler of this will close it.

#111 - 01/15/2016 11:19 AM - Greg Shah

And when the window is active (and `window.setAutoRequestFocus(false);`) as in Hynek's offered approach - we do not have this message for mouse press. So no ability to close drop-down for mouse press on window title.

I thought you had already implemented his recommendation. Are you saying that the current revision does not implement it? Have you tried it?

#112 - 01/15/2016 11:26 AM - Eugenie Lyzenko

We should be ok here. When the overlay window is on it is also active at the driver level.

What is the "driver level"? What code implements now this level? This is not Swing events handler - `SwingEmulatedWindow` because here we can not have two active windows at the same time, correct?

Clicking outside of the overlay window will cause the driver to send a window-deactivate for the overlay, and the handler of this will close it.

1. Click handling is not an issue and not acceptable because too late, we need to handle mouse press.
2. For this to work we need to receive window-activate first for the window that become active. If the window is active - it will not receive this event.

#113 - 01/15/2016 11:36 AM - Eugenie Lyzenko

I thought you had already implemented his recommendation. Are you saying that the current revision does not implement it? Have you tried it?

Yes, the current revision activates `OverlayWindow` after `show()`. If we do not activate `OverlayWindow` - it is not closing by mouse press on main window title.

Eugenie Lyzenko wrote:

We should be ok here. When the overlay window is on it is also active at the driver level.

What is the "driver level"? What code implements now this level? This is not Swing events handler - SwingEmulatedWindow because here we can not have two active windows at the same time, correct?

The driver level means SwingGuiDriver and GuiWebDriver and the related classes. At this level we will only have one active window at a time. In case of an open overlay window, this active window will be the overlay window. So when you click outside the driver will always give you the window-deactivated message. For the driver the main window and overlay window are just regular windows.

But at the widgets level (TopLevelWindow, WindowGuiImpl, @OverlayWindow) we still want to consider the main window (WindowGuiImpl) as active even when the overlay window is on. This is independent on the driver.

Clicking outside of the overlay window will cause the driver to send a window-deactivate for the overlay, and the handler of this will close it.

1. Click handling is not an issue and not acceptable because too late, we need to handle mouse press.

When I mentioned clicking outside of the overlay window I didn't mean to handle a mouse click but instead a user action leading to a window-deactivate event.

2. For this to work we need to receive window-activate first for the window that become active. If the window is active - it will not receive this event.

See above.

#115 - 01/15/2016 11:52 AM - Hynek Cihlar

Hynek Cihlar wrote:

Eugenie Lyzenko wrote:

We should be ok here. When the overlay window is on it is also active at the driver level.

What is the "driver level"? What code implements now this level? This is not Swing events handler - SwingEmulatedWindow because here we can not have two active windows at the same time, correct?

The driver level means SwingGuiDriver and GuiWebDriver and the related classes. At this level we will only have one active window at a time. In case of an open overlay window, this active window will be the overlay window. So when you click outside the driver will always give you the window-deactivated message. For the driver the main window and overlay window are just regular windows.

But at the widgets level (TopLevelWindow, WindowGuiImpl, @OverlayWindow) we still want to consider the main window (WindowGuiImpl) as active even when the overlay window is on. This is independent on the driver.

For the above I was assuming the driver wouldn't know about the special properties of overlay windows, it would consider overlay and main window equal. Which is how the drivers are implemented now.

#116 - 01/15/2016 04:21 PM - Eugenie Lyzenko

Yes.

Hynek also has [#2958](#) which he may include here if it can be done in time.

OK.

Task branch 2837a for review updated to revision 10985.

The update has fixes for notes (3) and (4). Drop-down activation logic has changed a bit. The DB restore is in progress. If there are no objection I will put this into regression testing to planned commit. Is it OK.

#117 - 01/15/2016 05:02 PM - Eugenie Lyzenko

Eugenie Lyzenko wrote:

Is it OK.

Sorry, that was a question, I'm going to start testing if you do not mind. OK?

#118 - 01/15/2016 06:02 PM - Eugenie Lyzenko

Hynek Cihlar wrote:

The driver level means SwingGuiDriver and GuiWebDriver and the related classes. At this level we will only have one active window at a time. In case of an open overlay window, this active window will be the overlay window. So when you click outside the driver will always give you the window-deactivated message. For the driver the main window and overlay window are just regular windows.

But at the widgets level (TopLevelWindow, WindowGuiImpl, @OverlayWindow) we still want to consider the main window (WindowGuiImpl) as active even when the overlay window is on. This is independent on the driver.

Where in this concept the WindowManager class is located? Driver level or widget level?

#119 - 01/16/2016 04:56 AM - Hynek Cihlar

Eugenie Lyzenko wrote:

Yes.

Hynek also has [#2958](#) which he may include here if it can be done in time.

OK.

Task branch 2837a for review updated to revision 10985.

The update has fixes for notes (3) and (4). Drop-down activation logic has changed a bit. The DB restore is in progress. If there are no objection I will put this into regression testing to planned commit. Is it OK.

I can confirm I don't see the abend for (3) anymore.

For (4) the drop-down of combo-box B still doesn't open occasionally. I think the code should work. It seems that sometimes the click event doesn't arrive to the target widget, probably a timing issue. I am looking into it.

I have a little problem with the logic fixing (4) though. The OverlayWindow mechanism should be generic so that it can be used in other cases, like the popup menu. But we are hard-wiring the logic into a combo box widget which is not correct. It is essential to get the eventing right so please let me do

the check above first.

#120 - 01/16/2016 06:43 AM - Eugenie Lyzenko

Hynek Cihlar wrote:

For (4) the drop-down of combo-box B still doesn't open occasionally. I think the code should work. It seems that sometimes the click event doesn't arrive to the target widget, probably a timing issue. I am looking into it.

I've never seen it on my system.

I have a little problem with the logic fixing (4) though. The OverlayWindow mechanism should be generic so that it can be used in other cases, like the popup menu. But we are hard-wiring the logic into a combo box widget which is not correct. It is essential to get the eventing right so please let me do the check above first.

The problem was combo-box mouse events processing specific. So the fix was combo-box specific. Actually I doubt it will happen for other possible overlay window child.

#121 - 01/16/2016 06:46 AM - Eugenie Lyzenko

The main part of 10985 testing is OK. Continue with CTRL-C one.

#122 - 01/16/2016 08:10 AM - Greg Shah

Code Review Task Branch 2837a Revision 10985

It is OK to test this version. We may want to go ahead and merge this to trunk as well, if it passes testing, just because there are some useful fixes in this branch.

However, there is a hard-coded reference to DropDownGuiImpl in OverlayWindow.dismiss(), which definitely should not be there. As Hynek notes, when we use this in menus and tooltips, we are going to have the same problem to fix. Perhaps we can fix that in 2837b where several other changes are pending too.

#123 - 01/16/2016 09:31 AM - Eugenie Lyzenko

Greg Shah wrote:

Code Review Task Branch 2837a Revision 10985

It is OK to test this version. We may want to go ahead and merge this to trunk as well, if it passes testing, just because there are some useful fixes in this branch.

OK.

However, there is a hard-coded reference to `DropDownGuiImpl` in `OverlayWindow.dismiss()`, which definitely should not be there. As Hynek notes, when we use this in menus and tooltips, we are going to have the same problem to fix. Perhaps we can fix that in 2837b where several other changes are pending too.

OK. I have already thought about making the overlay window client be generic. We need all possible widgets inside overlay to implement some predefined set of methods I guess.

#124 - 01/16/2016 04:31 PM - Eugenie Lyzenko

Greg Shah wrote:

Code Review Task Branch 2837a Revision 10985

It is OK to test this version. We may want to go ahead and merge this to trunk as well, if it passes testing, just because there are some useful fixes in this branch.

The testing completed. The results - 2837a_10985_de19d84_20160116_evl.zip. No regressions found.

Can I merge this update into trunk?

#125 - 01/16/2016 09:09 PM - Eugenie Lyzenko

However, there is a hard-coded reference to `DropDownGuiImpl` in `OverlayWindow.dismiss()`, which definitely should not be there. As Hynek notes, when we use this in menus and tooltips, we are going to have the same problem to fix. Perhaps we can fix that in 2837b where several other changes are pending too.

Task branch 2837a for review updated to revision 10986.

The hardcoding has been removed. No testing required because only GUI code is changed and can be committed due to only cosmetic update.

#126 - 01/17/2016 10:12 AM - Greg Shah

The change is not exactly what I had in mind. The problem is that the code still has a reference to the drop-down widget. It should not have any such reference. We will need an interface that must be implemented by all classes that get contained in an overlay window.

However, for now please do merge 2837a revision 10986 to trunk. Then open 2837b for the remaining changes.

#127 - 01/17/2016 04:41 PM - Eugenie Lyzenko

Greg Shah wrote:

The change is not exactly what I had in mind. The problem is that the code still has a reference to the drop-down widget. It should not have any such reference. We will need an interface that must be implemented by all classes that get contained in an overlay window.

Yes, I understand. This is just a fix to avoid considering all overlay child as drop-down.

However, for now please do merge 2837a revision 10986 to trunk. Then open 2837b for the remaining changes.

OK. Will be done in a hour I guess.

#128 - 01/17/2016 04:52 PM - Eugenie Lyzenko

Branch 2837a was merged to trunk as revno 10964 then it was archived.

#129 - 01/17/2016 05:21 PM - Eugenie Lyzenko

Branch 2837b has been created.

#130 - 01/18/2016 12:44 PM - Eugenie Lyzenko

Greg Shah wrote:

The change is not exactly what I had in mind. The problem is that the code still has a reference to the drop-down widget. It should not have any such reference. We will need an interface that must be implemented by all classes that get contained in an overlay window.

Task branch 2837b for review updated to revision 10965.

This update implements generic approach to the methods to be implemented with every widget plugged inside the OverlayWindow. There should be the set of method to be implemented for every widget, the all of them are located in new OverlayChild interface.

#131 - 01/18/2016 01:38 PM - Hynek Cihlar

Eugenie Lyzenko wrote:

Greg Shah wrote:

The change is not exactly what I had in mind. The problem is that the code still has a reference to the drop-down widget. It should not have any such reference. We will need an interface that must be implemented by all classes that get contained in an overlay window.

Task branch 2837b for review updated to revision 10965.

This update implements generic approach to the methods to be implemented with every widget plugged inside the OverlayWindow. There should be the set of method to be implemented for every widget, the all of them are located in new OverlayChild interface.

Eugenie, I am wondering whether OverlayChild is needed at all. We already have Widget.hide() and Widget.destroy() and you can also send events to the overlay child widgets directly.

#132 - 01/18/2016 08:32 PM - Hynek Cihlar

Hynek Cihlar wrote:

Eugenie Lyzenko wrote:

Yes.

Hynek also has [#2958](#) which he may include here if it can be done in time.

OK.

Task branch 2837a for review updated to revision 10985.

The update has fixes for notes (3) and (4). Drop-down activation logic has changed a bit. The DB restore is in progress. If there are no objection I will put this into regression testing to planned commit. Is it OK.

I can confirm I don't see the abend for (3) anymore.

For (4) the drop-down of combo-box B still doesn't open occasionally. I think the code should work. It seems that sometimes the click event doesn't arrive to the target widget, probably a timing issue. I am looking into it.

Yes, this is definitely a timing issue and the way combo processes events for the drop downs. The combo B should open its drop-down on mouse-pressed event. But when the event is enqueued with EventManager.postOSEvent() during ComboBox.activate() and the call to

TC.waitForWorker(), it will not be processed and the drop-down won't open. The mouse event being on the OS queue is applied to the regular queue (EventManager.postEvent()) during TC.waitForWorker() and then popped (without being processed) before ComboBox.activate() returns. This makes the event linger in the queue until it is trashed with ThinClient.popTrash().

I am not sure yet, whether the problem is caused with the mouse-pressed event being popped from the OS and pushed to the regular queue (TC.postEvent()) or that the event is not processed before ComboBox.active() returns.

Constantin, do you have any thoughts on this?

#133 - 01/19/2016 03:40 AM - Eugenie Lyzenko

Hynek Cihlar wrote:

Eugenie, I am wondering whether OverlayChild is needed at all. We already have Widget.hide() and Widget.destroy() and you can also send events to the overlay child widgets directly.

None of these methods do serve the facility required and implemented in exit() or dismiss(). Maybe we could leave these methods empty for other OverlayWindow child but drop-down has some specific. It is under control from related combo-box and this produces some predefined set of actions to exit from.

#134 - 01/19/2016 11:36 AM - Hynek Cihlar

Hynek Cihlar wrote:

I am not sure yet, whether the problem is caused with the mouse-pressed event being popped from the OS and pushed to the regular queue (TC.postEvent()) or that the event is not processed before ComboBox.active() returns.

I think the correct solution is to process any pending events when popping the event queue in ComboBox.activate().

And the solution I believe is to execute TC.waitForWorker() in ComboBox.active() in an event bracket:

```
tc.eventBracket(() ->
{
    tc.waitForWorker(list, fid, -1, null, true, false, false,
                    BlockingOperation.NO_BLOCKING_OPERATION);
});
```

Constantin, can you please confirm the fix?

#135 - 01/19/2016 02:05 PM - Eugenie Lyzenko

Task branch 2837b for review updated to revision 10967.

This is the fix for TAB/BACK-TAB key pressing inside opened drop-down. Now the focus is properly moving to next widget in appropriate tab key direction. The idea is to defer TAB/BACK-TAB keys to special buffer and initiate processing after drop-down is closed.

#136 - 01/19/2016 02:23 PM - Eugenie Lyzenko

One more issue has been found for drop-down. Open it, select any item and close. On the next opening it is not possible to walk through items by keyboard. It looks very strange, debugging.

#137 - 01/19/2016 07:19 PM - Eugenie Lyzenko

Task branch 2837b for review updated to revision 10968.

This is the fix for navigation keys issue recently found. The idea is to route events coming to opened drop-down to the internal selection-list widget. Also the generated beep() sound is turned off as a reaction of the selection list to TAB/BACK-TAB keys processing.

#138 - 01/19/2016 07:59 PM - Eugenie Lyzenko

Clarifications for some of the remaining TODO items:

About overlay activation.

- We need to have input focus inside overlay to be able to get a keyboard events for drop-down internals.

- I think we can have two layers for activation representation as initially suggested by Hynek. In this case from the low-level point of view(Swing side event producer) the main window will loose activation opening overlay window. But from the application point of view(everything evolving around ThinClient event processing) the main window will behaves as if it will keep active when OverlayWindow is opening. Is it correct understanding of the requirements?

- To implement such two layer window activation handler we need to strictly separate these layers and define as simple as possible tool to communicate one level with another, right? I'm still thinking what is the place for WindowManager class in this concept? Can it be such conductor delivering activation events from one layer to another?

- Ideally there must be no difference between window activation moving when we shift from one P2J main window to another P2J or from one P2J main window to non-P2J native window, correct?

- On the P2J application high layer it is not possible to have two active windows and two opened overlay windows at the same time, correct?

- If the main window title has active color - this means the main window must properly accept and handle keyboard event, correct? This is important for current new P2J window behavior that sometimes(not always) does not do this.

#139 - 01/20/2016 03:19 PM - Eugenie Lyzenko

- File *cbb_test21_1_reverse_p2j_gui_20160120.jpg* added

Task branch 2837b for review updated to revision 10969.

(5) Drop-down position must take up the available space. See the screenshot.

This is the fix for Y drop-down positioning when there is no room space to roll down. The swing client screenshot is attached.

Testing on the web client is in progress.

#140 - 01/20/2016 03:30 PM - Eugenie Lyzenko

Testcase combo_box/combo_box21_1.p has been added to bzt repo. Revision 1456.

#141 - 01/20/2016 03:50 PM - Eugenie Lyzenko

- File *cbb_test21_1_reverse_p2j_gui_web_20160120.jpg* added

The web client is working good for 10969. See the picture attached.

#142 - 01/20/2016 08:02 PM - Eugenie Lyzenko

Regarding the activation subsystem rework for OverlayWindow the plan is:

1. Investigate in details how the current approach is working.
2. Separate the two layer players. Just because the internal rule could be a bit different for each part in this approach. We need to define which classes are in the low layer and which are on the high one, what exactly is the connection tool between them.
3. Determine all requirements for OverlayWindow against activation engine.
4. Implement OverlayWindow part and possible make changes into general subsystem to make it working.
5. Document all considerations to take in mind for all further modifications of the window activation subsystem or related code. This is required to avoid regressions.

Any comments or notes or objection for this plan?

Hynek, I would like to know about what you are going to do in the activation subsystem to avoid interference.

#143 - 01/21/2016 07:28 AM - Greg Shah

The plan seems reasonable.

#144 - 01/21/2016 08:56 AM - Hynek Cihlar

Eugenie Lyzenko wrote:

Hynek, I would like to know about what you are going to do in the activation subsystem to avoid interference.

Eugenie, what interference do you mean?

#145 - 01/21/2016 09:36 AM - Eugenie Lyzenko

Hynek Cihlar wrote:

Eugenie Lyzenko wrote:
Eugenie, what interference do you mean?

I mean the changes we both making for same classes can conflict with each other. That causes the code to be undone/fix to make the final key to work. I want to minimize this effect. To do this I need to know your planned strategy(and want you to know about what I'm going to do) for activation related code.

#146 - 01/22/2016 05:26 AM - Hynek Cihlar

Eugenie Lyzenko wrote:

Hynek Cihlar wrote:

Eugenie Lyzenko wrote:
Eugenie, what interference do you mean?

I mean the changes we both making for same classes can conflict with each other. That causes the code to be undone/fix to make the final key to work. I want to minimize this effect. To do this I need to know your planned strategy(and want you to know about what I'm going to do) for activation related code.

I see. I only have the pending fix for the problem with the drop-down not showing up, this will be a trivial fix and should not cause conflicts with your changes.

#147 - 01/22/2016 07:37 AM - Constantin Asofiei

Hynek Cihlar wrote:

Hynek Cihlar wrote:

I am not sure yet, whether the problem is caused with the mouse-pressed event being popped from the OS and pushed to the regular queue (TC.postEvent()) or that the event is not processed before ComboBox.active() returns.

I think the correct solution is to process any pending events when popping the event queue in ComboBox.activate().

And the solution I believe is to execute TC.waitForWorker() in ComboBox.active() in an event bracket:
[...]

Constantin, can you please confirm the fix?

I think this is OK for GUI (as triggers are seen and executed while the drop-down is active), but is not OK in ChUI. See this test:

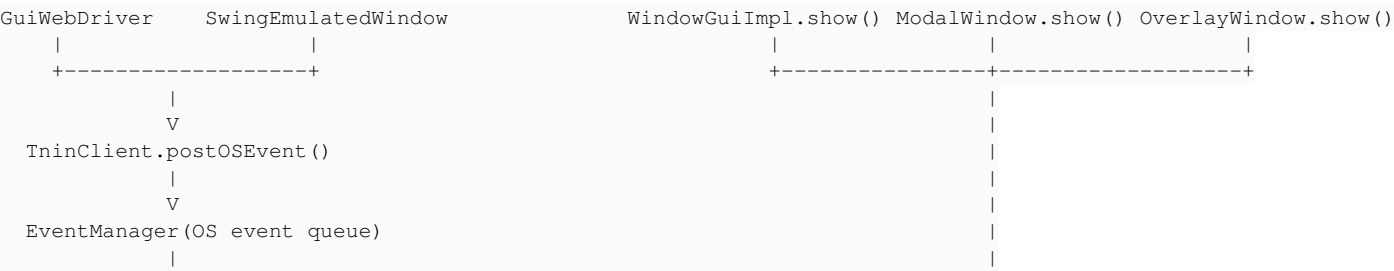
```
DEF VAR ch AS CHAR VIEW-AS combo-box LIST-ITEMS "1" , "2" , "3" , "4".
    DEF VAR i AS INT.
ON 'a':U ANYWHERE
DO:
    MESSAGE "a" STRING(i) .
    i = i + 1.
    RETURN.
END.

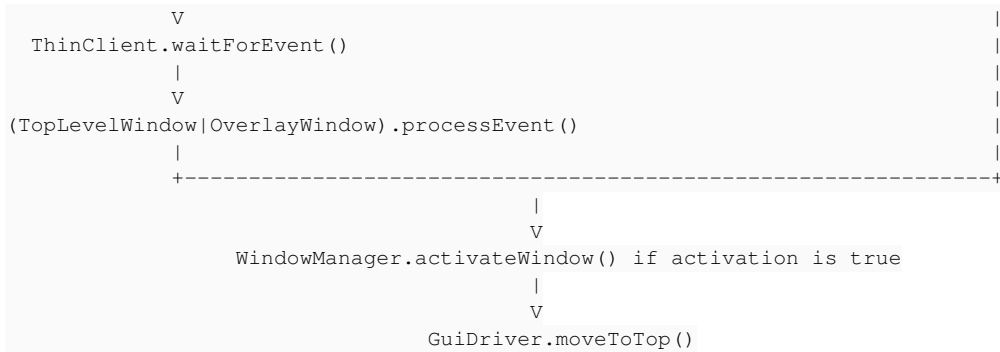
UPDATE ch.
```

In GUI, after the drop-down is opened, pressing "a" will execute the trigger. In ChUI, it will not.

#148 - 01/22/2016 10:08 AM - Eugenie Lyzenko

Activation sources schema:





Getting info about if the given window is active:

```
Window <- GuiDriver.isWindowActive() <- (GuiWebDriver|SwingEmulatedWindow).check_If_The_Window_Is_Active
```

The new implementation idea is to make WindowManager the point where the low layer activation is transforming to high layer. This class become a gate between two layers of the activation subsystem.

1. We have only one active window per client session(or none)
2. Change window check for activation from GuiDriver to WindowManager in all application level calls.
3. The WindowManager.activeWnd will store the currently active window or null in none is active.
4. OverlayWindow will not be active from WindowManager point of view.

The rules:

1. There must be no direct calls from widget code to GuiDriver to find out active window status. Only WindowManager.isActiveWindow()
2. Every call from Window to change activation status(including moveToTop() calls) must change the WindowManger active window cached variable to avoid out of sync.

If any comments/notes/objections please let me know.

Eugenie Lyzenko wrote:

Activation sources schema:

[...]

Getting info about if the given window is active:

[...]

I like the idea of moving more of the window-related logic to WindowManager.

The new implementation idea is to make WindowManager the point where the low layer activation is transforming to high layer. This class become a gate between two layers of the activation subsystem.

1. We have only one active window per client session(or none)
2. Change window check for activation from GuiDriver to WindowManager in all application level calls.
3. The WindowManager.activeWnd will store the currently active window or null in none is active.

Note that WindowManager.activeWnd is used to implement ACTIVE-WINDOW 4GL system handle. It has different semantics from the OS-level active window, ACTIVE-WINDOW and currently active OS window may not be equal.

4. OverlayWindow will not be active from WindowManager point of view.

The rules:

1. There must be no direct calls from widget code to GuiDriver to find out active window status. Only WindowManager.isActiveWindow()
2. Every call from Window to change activation status(including moveToTop() calls) must change the WindowManager active window cached variable to avoid out of sync.

IMHO asking the GUI driver for an active window is more robust than caching the value in WindowManager. The gui driver may choose whether to cache the value itself or whether to forward the call to the UI backend (Swing/OS for example).

#150 - 01/22/2016 11:53 AM - Eugenie Lyzenko

IMHO asking the GUI driver for an active window is more robust than caching the value in WindowManager. The web driver may choose whether to cache the value itself or whether to forward the call to the UI backend (Swing/OS for example).

I see. I would like to make the P2J activation handling GUI driver independent. We have Web and Swing drivers, the internal activation logic can be different inside these drivers. That's why I'm inclining to make the WindowManager the main player here.

#151 - 01/22/2016 12:01 PM - Eugenie Lyzenko

Note that WindowManager.activeWnd is used to implement ACTIVE-WINDOW 4GL system handle. It has different semantics from the OS-level active window, ACTIVE-WINDOW and currently active OS window may not be equal.

Another word in P2J the variable that implements ACTIVE-WINDOW can never be NULL for P2J client session, correct?

#152 - 01/22/2016 12:27 PM - Hynek Cihlar

Eugenie Lyzenko wrote:

Note that WindowManager.activeWnd is used to implement ACTIVE-WINDOW 4GL system handle. It has different semantics from the OS-level active window, ACTIVE-WINDOW and currently active OS window may not be equal.

Another word in P2J the variable that implements ACTIVE-WINDOW can never be NULL for P2J client session, correct?

It seems that in GUI ACTIVE-WINDOW always returns a valid handle, even in batch mode. I didn't check ChUI though.

#153 - 01/22/2016 01:05 PM - Greg Shah

Eugenie Lyzenko wrote:

Note that WindowManager.activeWnd is used to implement ACTIVE-WINDOW 4GL system handle. It has different semantics from the OS-level active window, ACTIVE-WINDOW and currently active OS window may not be equal.

Another word in P2J the variable that implements ACTIVE-WINDOW can never be NULL for P2J client session, correct?

More importantly, the ACTIVE-WINDOW in the 4GL may be a window that is INACTIVE from an operating system perspective. For this reason, we must not use the same state for both things.

#154 - 01/22/2016 01:07 PM - Greg Shah

I would like to make the P2J activation handling GUI driver independent. We have Web and Swing drivers, the internal activation logic can be different inside these drivers. That's why I'm inclining to make the WindowManager the main player here.

I tend to agree with this. If we do find that there is a specific need for differences at the driver level, then we should limit that to a specific API and still have as much of the rest of the activation processing in common code.

#155 - 01/22/2016 04:03 PM - Eugenie Lyzenko

Task branch 2837b for review updated to revision 10970.

Implemented new activation approach related to OverlayWindow. New variable was introduced to keep track the current window state, not involving the ACTIVE-WINDOW handler related code. The window title repaint code moved to place after the current active window value is changed. If any notes or objections - please let me know.

Tested with Swing client. Going to test against the Web client as well.

#156 - 01/22/2016 05:37 PM - Eugenie Lyzenko

Task branch 2837b for review updated to revision 10971.

Adding the change to make the web client working with new activation approach. Not possible to make OverlayWindow active in Web client too. Now the Web client also working.

The next step is to create more tests to check more complex event/window/overlay combination.

#157 - 01/22/2016 09:32 PM - Eugenie Lyzenko

So the only issue I can see in with current drop-down/overlay implementation is the issue when we have two combo-boxes and try to open one of them(combo A) when another(combo B) is already opened. Need to investigate the issue and fix. The complexity is the issue is happening not always, sometimes it works fine. Agree, looks a kind of timing problem or message handling incorrect order.

Hynek, do you have any findings in this area I need to know?

#158 - 01/23/2016 05:09 AM - Hynek Cihlar

Eugenie Lyzenko wrote:

So the only issue I can see in with current drop-down/overlay implementation is the issue when we have two combo-boxes and try to open one of them(combo A) when another(combo B) is already opened. Need to investigate the issue and fix. The complexity is the issue is happening not always, sometimes it works fine. Agree, looks a kind of timing problem or message handling incorrect order.

Hynek, do you have any findings in this area I need to know?

See notes 132, 134, 147. You will find the analysis of the issue there and a solution which is, as pointed out by Constantin, not completely OK.

#159 - 01/23/2016 07:28 AM - Greg Shah

Code Review Task branch 2837b Revision 10971

I think the changes are moving in the right direction.

My only concern is that the code added to `OverlayWindow.processEvent()` is something that should be handled in a centralized place. This same problem exists whenever we switch from any P2J window to a non-P2J window. The problem is not specific to the `OverlayWindow` and the code actually doesn't affect the `OverlayWindow` itself. So the code should not be there, but in some more appropriate (and common/shared) location.

Hynek: please do a code review.

Also note that I have checked in some minor comment cleanup.

#160 - 01/23/2016 04:56 PM - Hynek Cihlar

Code Review Task branch 2837b Revision 10972.

(1) `TopLevelWindow.processEvent()` sends LEAVE event to the main window when overlay window is activated, it should not. Similarly, when `WindowActivated` is sent to the main window as a result of overlay window closed, no ENTRY event should be sent.

(2) The comment `// set up drop-down location just below the combo-box` in `DropDownGuiImpl` is misleading.

- (3) The comment // the rest should go through scrollable list in DropDownGuiImpl implies that TAB should not be routed to the scrollable list, but it is.
- (4) In WindowManager.activateWindow() should not set activeWnd and currentActiveWnd. These should be set only after WindowActivated message is sent by the driver. The reason for this is that the OS may happily ignore the request to activate a window.
- (5) When you moved caching of active window to WindowManager it is no longer needed in GuiWebDriver. GuiWebDriver.activeWindow can be removed as well as GuiDriver.isWindowActive().
- (6) GuiWebDriver doesn't send window-activated event for overlay window, while SwingGuiDriver does. We should keep this consistent.

#161 - 01/23/2016 05:18 PM - Eugenie Lyzenko

Task branch 2837b for review updated to revision 10973.

The event processing code for deactivation has been moved to common place in TopLevelWindow. However the required actions to do are related to overlay window(that's why it was implemented in OverlayWindow before). So we have to consider this in common code to make a difference between regular top-level window and overlay window.

#162 - 01/23/2016 06:01 PM - Eugenie Lyzenko

- (5) When you moved caching of active window to WindowManager it is no longer needed in GuiWebDriver. GuiWebDriver.activeWindow can be removed...

This method is called when the mouse is pressing on window title in web client and it is the only way to close the overlay window in such conditions for web client. We can not remove it.

#163 - 01/23/2016 06:19 PM - Hynek Cihlar

Eugenie Lyzenko wrote:

- (5) When you moved caching of active window to WindowManager it is no longer needed in GuiWebDriver. GuiWebDriver.activeWindow can be removed...

This method is called when the mouse is pressing on window title in web client and it is the only way to close the overlay window in such conditions for web client. We can not remove it.

In revision 10973 I don't see any references to GuiDriver.isWindowActive(). What code are you referring to?

#164 - 01/23/2016 06:32 PM - Eugenie Lyzenko

Hynek Cihlar wrote:

In revision 10973 I don't see any references to `GuiDriver.isWindowActive()`. What code are you referring to?

Yes, you are right. I wrongly thought about `GuiWebDriver.activateWindow()`.

#165 - 01/23/2016 06:53 PM - Eugenie Lyzenko

(1) `TopLevelWindow.processEvent()` sends LEAVE event to the main window when overlay window is activated, it should not.

When the overlay is activated no `TopLevelWindow.processEvent()` is called, it overrides with `OverlayWindow.processEvent()` and does not generate any LEAVE event.

#166 - 01/23/2016 07:15 PM - Hynek Cihlar

Eugenie Lyzenko wrote:

(1) `TopLevelWindow.processEvent()` sends LEAVE event to the main window when overlay window is activated, it should not.

When the overlay is activated no `TopLevelWindow.processEvent()` is called, it overrides with `OverlayWindow.processEvent()` and does not generate any LEAVE event.

Right. So the only problem for (1) is when the main window is activated after its overlay window is closed. In this case LEAVE is sent to the overlay and ENTRY sent to the main window, if I interpret the event bracket correctly in `TopLevelWindow` line 751.

#167 - 01/23/2016 09:04 PM - Eugenie Lyzenko

Task branch 2837b for review updated to revision 10974.

Notes resolution.

(4) In `WindowManager.activateWindow()` should not set `activeWnd` and `currentActiveWnd`. These should be set only after `WindowActivated` message is sent by the driver. The reason for this is that the OS may happily ignore the request to activate a window.

I need this code to properly work the Web client. So if you have the sample to illustrate your consideration - let me know.

#168 - 01/24/2016 05:21 AM - Hynek Cihlar

Eugenie Lyzenko wrote:

Task branch 2837b for review updated to revision 10974.

Notes resolution.

(4) In `WindowManager.activateWindow()` should not set `activeWnd` and `currentActiveWnd`. These should be set only after `WindowActivated` message is sent by the driver. The reason for this is that the OS may happily ignore the request to activate a window.

I need this code to properly work the Web client. So if you have the sample to illustrate your consideration - let me know.

See [https://msdn.microsoft.com/en-us/library/windows/desktop/ms633539\(v=vs.85\).aspx](https://msdn.microsoft.com/en-us/library/windows/desktop/ms633539(v=vs.85).aspx), especially the "Remarks" section.

In any case I believe we should keep the behavior of all drivers consistent. If Web GUI driver doesn't send a window activation message when it should (and I know it sometimes doesn't, see [#2956](#)) we should fix the driver and not the app logic above.

#169 - 01/24/2016 08:11 AM - Greg Shah

Code Reivew Task Branch 2837b Revision 10973

This is certainly better.

However, instead of putting `instanceof OverlayWindow` everywhere, I would rather see a method (returning boolean) added to `TopLevelWindow` to differentiate certain conditions. For example, `hasTitlebar()` could be used to determine if titlebar drawing is needed. By using `@instanceof OverlayWindow` we are hard coding references to that specific class everywhere. Even if we only use some logic in that one case, the `instanceof` approach makes the code more tightly coupled, unpredictable and harder to understand.

Please look at these places and see if a cleaner/better approach can be made:

```

./WindowManager.java:      if (!(window instanceof OverlayWindow))
./WindowManager.java:          if (tw instanceof OverlayWindow)
./WindowManager.java:      if (window instanceof OverlayWindow)
./TopLevelWindow.java:          if (this instanceof OverlayWindow && tlw != null)
./TopLevelWindow.java:          if (!(oldActive instanceof OverlayWindow) ||
./TopLevelWindow.java:              (this instanceof OverlayWindow && activated.withNativeWindow()))
./TopLevelWindow.java:      else if ((oldActive != this && !(oldActive instanceof OverlayWindow)) ||
./TopLevelWindow.java:      else if (!(this instanceof OverlayWindow))
./TopLevelWindow.java:      if (this instanceof GuiWindow && !(this instanceof OverlayWindow))
./gui/OverlayWindow.java:      if (currFocusedWnd instanceof OverlayWindow &&

```

#170 - 01/24/2016 07:23 PM - Eugenie Lyzenko

Task branch 2837b for review updated to revision 10975.

Notes resolution for instanceof OverlayWindow usage and WindowManager.activateWindow() setting activeWnd and currentActiveWnd.

#171 - 01/25/2016 07:39 AM - Greg Shah

Code Review Task Branch 2837b Revision 10975

I'm fine with the changes.

#172 - 01/25/2016 12:13 PM - Eugenie Lyzenko

Task branch 2837b for review updated to revision 10976.

This update fixes the issue for two combo-boxes opening for the case when trying open one combo while opened another.

Going to rebase with 10965, so

Hynek,

please do not update anything to 2837b until further note. I'll notify when I finish.

#173 - 01/25/2016 12:34 PM - Eugenie Lyzenko

Rebased task branch 2837b from trunk revision 10965, new branch revision is 10977.

Hynek, the branch is now unlocked.

So on my systems this version properly handles drop-down opening for two combo-box widget.

And if there are no objections I would like to put this version into testing to commit with trunk if there will be no regressions.

#174 - 01/25/2016 12:48 PM - Greg Shah

Code Review Task Branch 2837b Revision 10977

I'm fine with the change.

Please go ahead with regression testing. Post your results here. Wait for my confirmation before merging.

#175 - 01/25/2016 12:49 PM - Greg Shah

Hynek: If I understand correctly, your only remaining concern is the following:

In any case I believe we should keep the behavior of all drivers consistent. If Web GUI driver doesn't send a window activation message when it should (and I know it sometimes doesn't, see [#2956](#)) we should fix the driver and not the app logic above.

I think this is a valid concern, but I suspect we should fix this as part of [#2956](#). What do you think?

#176 - 01/25/2016 12:52 PM - Eugenie Lyzenko

Greg Shah wrote:

In any case I believe we should keep the behavior of all drivers consistent. If Web GUI driver doesn't send a window activation message when it should (and I know it sometimes doesn't, see [#2956](#)) we should fix the driver and not the app logic above.

I think this is a valid concern, but I suspect we should fix this as part of [#2956](#). What do you think?

I made the changes to resolve this in 10977. At least in a manner the GuiWebDriver can send anything to TopLevelWindow.

#177 - 01/25/2016 02:28 PM - Eugenie Lyzenko

Greg Shah wrote:

Please go ahead with regression testing. Post your results here. Wait for my confirmation before merging.

OK. The testing will be started in a hour after DB is restored.

#178 - 01/25/2016 04:51 PM - Hynek Cihlar

Greg Shah wrote:

Hynek: If I understand correctly, your only remaining concern is the following:

Yes.

In any case I believe we should keep the behavior of all drivers consistent. If Web GUI driver doesn't send a window activation message when it should (and I know it sometimes doesn't, see [#2956](#)) we should fix the driver and not the app logic above.

I think this is a valid concern, but I suspect we should fix this as part of [#2956](#). What do you think?

I think you are right.

#179 - 01/25/2016 04:52 PM - Greg Shah

Hynek: Please add a note to [#2956](#) with your concern and your recommended approach.

#180 - 01/25/2016 05:37 PM - Hynek Cihlar

Hynek Cihlar wrote:

Greg Shah wrote:

Hynek: If I understand correctly, your only remaining concern is the following:

Yes.

A correction. The concern (note 160, point 4) has been addressed in 2837b revision 10976. Both drivers send window-activated message as a result of WindowManager.activateWindow() call, the fields WindowManager.activeWnd and WindowManager.currentActiveWnd are set as part of dispatch of the window-activated event.

In any case I believe we should keep the behavior of all drivers consistent. If Web GUI driver doesn't send a window activation message when it should (and I know it sometimes doesn't, see [#2956](#)) we should fix the driver and not the app logic above.

I think this is a valid concern, but I suspect we should fix this as part of [#2956](#). What do you think?

I think you are right.

#181 - 01/25/2016 05:37 PM - Hynek Cihlar

Greg Shah wrote:

Hynek: Please add a note to [#2956](#) with your concern and your recommended approach.

The concern has been addressed, see note 180.

#182 - 01/26/2016 07:33 AM - Eugenie Lyzenko

The main part completed without regressions. Running the CTRL-C tests.

#183 - 01/26/2016 11:13 AM - Eugenie Lyzenko

The testing completed without regression. Had to run CTRL-C 3-way tests in separate session. The results - 2837b_10977_de19d84_20160126_evl.zip.

So it is ready to be committed, waiting for the confirmation to merge into trunk.

#184 - 01/26/2016 11:20 AM - Greg Shah

You can merge to trunk.

#185 - 01/26/2016 11:29 AM - Eugenie Lyzenko

Greg Shah wrote:

You can merge to trunk.

Branch 2837b was merged to trunk as revno 10966 then it was archived.

#186 - 01/26/2016 11:46 AM - Greg Shah

I have created [#2970](#) for implementing OverlayWindow in menus and popup menus.

Before we close this task, would you please check if tooltips also need this support? I hope not, but we should know now if it is possible for a tooltip to ever draw outside of the containing window.

#187 - 01/26/2016 11:50 AM - Eugenie Lyzenko

Greg Shah wrote:

Before we close this task, would you please check if tooltips also need this support? I hope not, but we should know now if it is possible for a tooltip to ever draw outside of the containing window.

OK. I'll test this shortly.

#188 - 01/26/2016 12:17 PM - Eugenie Lyzenko

- File tooltip_overlay_4gl_gui_20160126.jpg added

Greg Shah wrote:

Before we close this task, would you please check if tooltips also need this support? I hope not, but we should know now if it is possible for a tooltip to ever draw outside of the containing window.

I think it is possible to have tooltip outside of the main window. Please see the picture attached. So looks like we need to enclose tooltip inside overlay window too. What do you think?

#189 - 01/26/2016 01:20 PM - Greg Shah

Yes, I was worried that might be the case. I'll make a new task for it.

#190 - 01/26/2016 01:21 PM - Greg Shah

I've created [#2971](#) for reworking the tooltip support to use OverlayWindow.

#191 - 01/26/2016 01:22 PM - Greg Shah

- Status changed from New to Closed

- Assignee set to Eugenie Lyzenko

- % Done changed from 0 to 100

#192 - 11/16/2016 12:12 PM - Greg Shah

- Target version changed from Milestone 12 to GUI Support for a Complex ADM2 App

Files				
Drop-down_P2J.png	1.78 KB	11/10/2015	Hynek Cihlar	
Drop-down_4GL.png	2.92 KB	11/10/2015	Hynek Cihlar	
cbb_test21_p2j_gui_20151218.jpg	119 KB	12/18/2015	Eugenie Lyzenko	
cbb_test21_p2j_web_20151223.jpg	97.7 KB	12/23/2015	Eugenie Lyzenko	
cbb_test21_p2j_web_20160106.jpg	112 KB	01/07/2016	Eugenie Lyzenko	
cbb_test21_p2j_web_fix_20160107.jpg	134 KB	01/08/2016	Eugenie Lyzenko	
cbb_test21_p2j_web_initial_20160108.jpg	96.9 KB	01/08/2016	Eugenie Lyzenko	
initLocation_1.txt	1.28 KB	01/08/2016	Sergey Ivanovskiy	
gui_btn_test5_p2j_20160112.jpg	47.6 KB	01/12/2016	Eugenie Lyzenko	
let_driver_post_the_activated_event.diff	2.38 KB	01/13/2016	Hynek Cihlar	
drop-down_inverted.png	5.54 KB	01/14/2016	Hynek Cihlar	
cbb_test21_1_reverse_p2j_gui_20160120.jpg	80.6 KB	01/20/2016	Eugenie Lyzenko	
cbb_test21_1_reverse_p2j_gui_web_20160120.jpg	104 KB	01/20/2016	Eugenie Lyzenko	
tooltip_overlay_4gl_gui_20160126.jpg	67.2 KB	01/26/2016	Eugenie Lyzenko	