GUI Automation testing - Window handle questions

Asked 16 years, 4 months ago Modified 15 years ago Viewed 3k times

3

Our company is currently writing a GUI automation testing tool for compact framework applications. We have initially searched many tools but none of them was right for us.

By using the tool you can record test-cases and group them together to test-suites. For every test-suite there is generated an application, which launches the application-under-test and simulates user-input.

In general the tool works fine, but as we are using **window handles** for simulation user input, you can't do very many things. For example it is impossible for us to get the name of a control (we just get the caption).

Another problem using window handles is checking for a change. At the moment we simulate a click on a control and depending on the result we know if the application has gone to the next step.

Is there any other (simpler) way for doing such things (for example the message queue or anything else)?

ui-automation window-handles gui-testing

Share

Improve this question

Follow

edited Dec 4, 2009 at 18:13

Anthony Mastrean

22.4k ● 22 ● 115 ● 190

asked Aug 20, 2008 at 20:03

ollifant

8,606 ● 10 ● 37 ● 45

5 Answers

Sorted by: Highest score (default)

2

Interesting problem! I've not done any low-level (think Win32) Windows programming in a while, but here's what I would do.

Use a named pipe and have your application listen to it. Using this named pipe as a communication medium, implement a real simple protocol whereby you can query the application for the name of a control given its HWND, or other things you find useful. Make sure the protocol is rich enough so that there is sufficient information exchanged between your application and the test framework. Make sure that the test framework does not yield too much "special behavior" from the app, because then you wouldn't really be testing the features, but rather your test framework.

There's probably way more elegant and cooler ways to implement this, but this is what I remember from the top of my head, using only simple Win32 API calls.

Another approach, which we have implemented for our product at work, is to record user events, such as mouse clicks and key events in an event script. This should be rich enough so that you can have the application play it back, artificially injecting those events into the message queue, and have it behave the same way it did when you first recorded the script. You basically simulate the user when you play back the script.

In addition to that, you can record any important state (user's document, preferences, GUI controls hierarchy, etc.), once when you record the script, and once when you play it back. This gives you two sets of data you can compare, to make sure for instance that everything stays the same. This solution gives you tests that not easy to modify (you have to re-record if your GUI changes), but that provide awesome regression testing.

(EDIT: This is also a terrific QA tool during beta testing, for instance: just have your users record their actions, and if there's a crash, you have a good chance of easily reproducing the problem by just playing back the script)

Good luck!

Carl

answered Aug 20, 2008 at 20:33

Carl Seleborg

13.3k ● 11 ● 59 ● 70

1

If the Automated GUI testing tool has knowledge about the framework the application is written in it could use that information to make better or more advanced scripts. TestComplete for example knows about Borland's VCL and WinForms. If you test applications build using Windows Presentation Foundation has advanced support for this build in.

Share Improve this answer

Follow

answered Aug 20, 2008 at 20:17

Lars Truijens

43.6k ● 6 ● 130 ● 145

1

use NUnitForms. I've used them with great success for single and multi threading apps and you don't have to worry about handles and stuff like that

Here are some posts about NUnitForms worth reading

NUnitForms and failed DragDrop registration - problem of MTA vs STA

Compiled application exe GUI testing with NUnitForms

Share Improve this answer

Follow

answered Nov 12, 2008 at 9:42

Mladen Prajdic

15.7k ● 2 ● 45 ● 51

I finally found a solution to communicate between the testing-application and the application-under-test:

0

Managed Spy. It's basically a .NET application build on top of ManagedSpyLib.

ManagedSpyLib allows programmatic access to the Windows Forms controls of another process. For this it uses Window Hooks and memory-mapping files.

Thanks for all who helped me to get to this solution!

Share Improve this answer

Follow

answered Aug 25, 2008 at 16:14

ollifant

8,606 ● 10 ● 37 ● 45

0

Managed Spy does not provide a solution for compact framework applications.

The company Jamo Solutions (www.jamosolutions.com) meets the requirements for automation testing on mobile devices, including .net compact framework applications.

Share Improve this answer

Follow

answered Nov 12, 2008 at 9:23

ksp