GUI Testing tools and feedbacks [closed]

Asked 16 years, 3 months ago Modified 2 years, 8 months ago

Viewed 6k times

15

Closed. This question needs to be more focused. It is not currently accepting answers.

Want to improve this question? Update the question so it focuses on one problem only by editing this post.

Closed 10 years ago.

Improve this question

I am working on the issue of testing my GUI and I am not entirely sure of the best approach here. My GUI is built using a traditional MVC framework so I am easily able to test the logic parts of the GUI without bringing up the GUI itself. However, when it comes to testing the functionality of the GUI, I am not really sure if I should worry about individually testing GUI components or if I should mainly just focus on functional testing the system. It is a pretty complex system in which testing the GUI frequently involves sending a message to the server and then observing the response on the GUI. My initial thoughts are that functional testing is the way to go here since I need a whole system running to really test the UI. Comments on this issue would be appreciated.

unit-testing user-interface functional-programming

Share

Improve this question

Follow

edited Mar 31, 2022 at 19:51

vvvvv

31.3k ● 19 ● 62 ● 98

asked Sep 18, 2008 at 12:26

user17601

201 ● 4 ● 7

13 Answers

Sorted by: Highest score (default)

7

Other GUI-testing tools I can offer are: [Thoughtworks White](), [PyWinAuto](), [AutoIt](), [AutoHotKey]().

One thing to keep in mind when trying to automate GUIs is that the only way you can do that is to build the GUI with automation in mind. Crush devs that think their GUIs should not support testability early on in the project and happily expose all the hooks that can help in automation on demand as your testing needs require that.

Share Improve this answer

Follow

answered Sep 18, 2008 at 17:15

Hristo Deshev

907 ● 6 ● 10

4

You have (at least) 2 issues - the complexity of the environment (the server) and the complexity of the GUI.

There are many tools for automating GUI testing. All of them are more or less fragile and require pretty much constant maintenance in the face of changing layout. There is benefit to be gained from using them, but it's a long term benefit.

The environment, on the other hand, is an area that can be tamed. If your application is architected using the Dependency Injection/Inversion technique (where you 'inject' the server component into the application), then you can use a 'mock' of the relevant server interfaces to enable you to script test cases.

Combining these two techniques will allow you to automate GUI testing.

Share Improve this answer Follow

edited Mar 31, 2022 at 19:51
vvvvv
31.3k ● 19 ● 62 ● 98

answered Sep 18, 2008 at 12:34
Seb Rose
3,666 ● 20 ● 29

3

Depending on where in the spectrum of MVC (that's an overused term) you sit, testing the view could be a mechanical process of ensuring that the correct model methods are called in response to the correct inputs to the view to testing some client side validation to who knows.

A lot of the patterns that have been evolved out of MVC (I'm thinking passive view, supervising controller) are striving to make the view require very little testing because it's really just wiring user inputs to the presenter or model (depending on the exact variant of the pattern you're using).

"testing the GUI frequently involves sending a message to the server and then observing the response on the GUI" This statement worries me.

I'm immediately thinking that the GUI should be tested using a mock or stub of the server to test that the correct interactions are occurring and the GUI responds appropriately.

If you need automated functional tests of the server, I don't see the need to have the GUI involved in those.

answered Sep 18, 2008 at 12:38

Hamish Smith
8,181 ● 1 ● 38 ● 49

2

Mercury QuickTest Pro, Borland SilkTest, and Ranorex Recorder are some GUI testing tools.

Share Improve this answer Follow

answered Sep 18, 2008 at 12:36

Nick
13.4k ● 17 ● 66 ● 100

Only "quick" thing about Mercury is the part in its name. Would you like to hear a story about the choice of name for Selenium? – Esko Sep 25, 2009 at 21:55

2

If your application is web-based you can write tests using tools like [WatiN]() or [Selenium]().

If your application is Windows .NET based, you could try [White]().

Share Improve this answer Follow

answered Sep 18, 2008 at 12:37

andypike
551 ● 2 ● 9 ● 16

1

My advice: forget the traditional GUI testing. It's too expensive. Coding the tests takes a lot of time, the tools aren't really stable so you will get unreliable test results. The coupling between the code and the test is very strong and you'll spend a lot of time with the maintenance.

The new trend is to ignore the GUI tests. See the ModelViewPresenter pattern from Fowler as a guideline [link text]

Share Improve this answer Follow

answered Sep 18, 2008 at 12:39

Karl
3,216 ● 1 ● 23 ● 29

Could You elaborate more on that? Reference some sources for the trend? – Rekin Apr 4, 2011 at 13:26

1

The clearest way I can say this is:

Don't waste your time writing automated GUI tests.

Especially when your working with an MVC app - in your case, when you send a message to the server, you can make sure the right message number comes back and be done. You can add some additional cases - or another test completely to make sure that the GUI is converting the message id's into the right strings, but you just need to run that test once.

Share Improve this answer Follow

answered Sep 18, 2008 at 17:03

Alan
1,045 ● 7 ● 14

We do incorporate GUI testing in our project, and it has its side effects. The developers however have one critical

1

design principle: *Keep the GUI layer as thin as possible!*

That means *no logic* in the GUI classes. Separate this in presentation models responsible for input validation etc.

For testing on a Unix machine we use the Xvfb server as the DISPLAY when running the tests.

Share Improve this answer Follow

answered Sep 18, 2008 at 17:07
Asgeir S. Nilsen
1,137 ● 9 ● 13

1

Try the hallway usability test. It's cheap and useful: go to the nearest hallway, grab the first person that passes, make them sit at your computer and use your software. Watch over their shoulder, you will see what they try to do, what frustrates them, and so on. Do this a few times and notice the patterns.

Share Improve this answer Follow

answered Sep 25, 2009 at 21:43
Emilio M Bumachar
2,613 ● 4 ● 27 ● 31

1 This is good for testing the GUI design, but it doesn't allow to easily and repeatedly check if the GUI (as it was designed) was broken in a recent commit or if it still works well. – Joanis Nov 24, 2010 at 21:47

What you're looking for is "acceptance testing." How you do it depends on the frameworks you're using, what type

0

of application you are creating and in what language. If you google your particular technology and the above phrase, you should find some tools you can use.

Share Improve this answer Follow

answered Sep 18, 2008 at 12:28

user1228

0

I've found [WinTask](#) to be a very good way to do GUI testing. Provided you don't constantly change the way the OS refers to each element of the UI, WinTask addresses the UI elements by name, so even if the layout changes, the UI elements can still be pressed / tweaked / selected.

Share Improve this answer Follow

answered Sep 23, 2008 at 11:24

Peter K.
8,128 ● 4 ● 51 ● 75

0

Don't miss the 'U' in 'GUI'

I mean: if what you're trying to test is all works right and works as it was planned to work, then you may follow [Seb Rose's answer](#).

But please, don't forget a **USER interface has to be made thinking about USERS**, and not ANY user but the TARGET USER the application was made for. So, after you are sure all works like it have to work, put every single view/screen/form in a test with a team made of users representing every group of different users that may use your application: advanced users, administrators, MS Office users, low computer profile users, high computer profile users... and then, get the critiques of every user, make a mix, re-touch your GUI if it's neccesary and back again to GUI user's test.

Share Improve this answer Follow

edited May 23, 2017 at 11:54
Community Bot
1 ● 1

answered Sep 19, 2008 at 7:25
joseluisgv

0

For SIMPLE Web based GUI testing try iMacros (a simple Firefox plug-in , has a cool feature to send the entire test to another person) Note that SIMPLE was spelled with Initials ...

Share Improve this answer Follow

edited May 2, 2009 at 19:39

answered May 1, 2009 at 17:56
Yordan Georgiev
5,420 ● 2 ● 59 ● 56