



Fakultät für Informatik

Studiengang Informatik

Automatisierte UI Tests für Windows Qt-Anwendungen

Software Qualitätssicherung

von

Dominik Sieberer

Stand der Version: 09.05.2022

Abstract

TODO

Inhaltsverzeichnis

1	Automatisierte UI Tests	1
1.1	Einleitung	1
1.2	Anwendung	2
1.3	Anforderungen	4
2	Open HMI Tester	6
2.1	Einführung Open HMI Tester	6
2.2	Durchführung der Tests	6
2.3	Tests	6
3	Qt Test	7
3.1	Einführung Qt Test	7
3.2	Durchführung der Tests	7
3.3	Tests	7
4	Vergleich	8
4.1	Open HMI Tester	8
4.2	Qt Test	8
4.3	Vergleich	8
5	Fazit	9
	Literaturverzeichnis	10

Abbildungsverzeichnis

1.1 Anwendung	3
-------------------------	---

1 Automatisierte UI Tests

1.1 Einleitung

Automatisiertes UI-Testing bietet zahlreiche Vorteile in der Softwareentwicklung. Einige davon sind:

- Automatisierte Tests haben vergleichbare Ergebnisse und sind weniger anfällig für menschliche Fehler
- Höhere Testabdeckungsrate der Software
- Erhöhte Testabdeckung fördert debugging
- Erstellter Testcode kann wiederverwendet werden, wodurch das Testen leicht skalierbar wird
- Automatisierte Tests sind im Vergleich zu manuellen Tests viel schneller
- Sie sind Kosten- und Zeiteffizienter als manuelle Tests

Der Global Quality Report zeigt, dass mehr als 60 % der Unternehmen aufgrund der höheren Testabdeckung durch Testautomatisierung in der Lage sind, Fehler schneller zu erkennen. Darüber hinaus stellten 57 % der Befragten fest, dass die Wiederverwendung von Testfällen durch den Einsatz von Automatisierung zunahm. Automatisiertes Testing gilt als de facto Standard in der Software Entwicklung [Cap].

Es äußerst wichtig, das richtige Gleichgewicht zwischen manuellen und automatisierten Tests zu finden. Jedes Projekt ist einzigartig, und es gilt, verschiedene Aspekte wie wirtschaftliche Machbarkeit, zeitliche Beschränkungen und die Art der durchzuführenden Tests zu berücksichtigen. Hier muss jedes Teams fundierte Entscheidungen treffen. Dieses Dokument dient als Entscheidungsgrundlage um ein Framework zu wählen, mit dem automatisierte UI Tests für Windows Qt-Anwendungen durchgeführt werden. Die Zielgruppe dieses Dokuments sind Softwareentwickler die mit dem C++ Qt Framework vertraut sind.

1.2 Anwendung

Die Anwendung, die zum Vergleich der Frameworks verwendet wird, ist eine stark vereinfachte Version der Anlagensteuerungssoftware der Ambright GmbH.

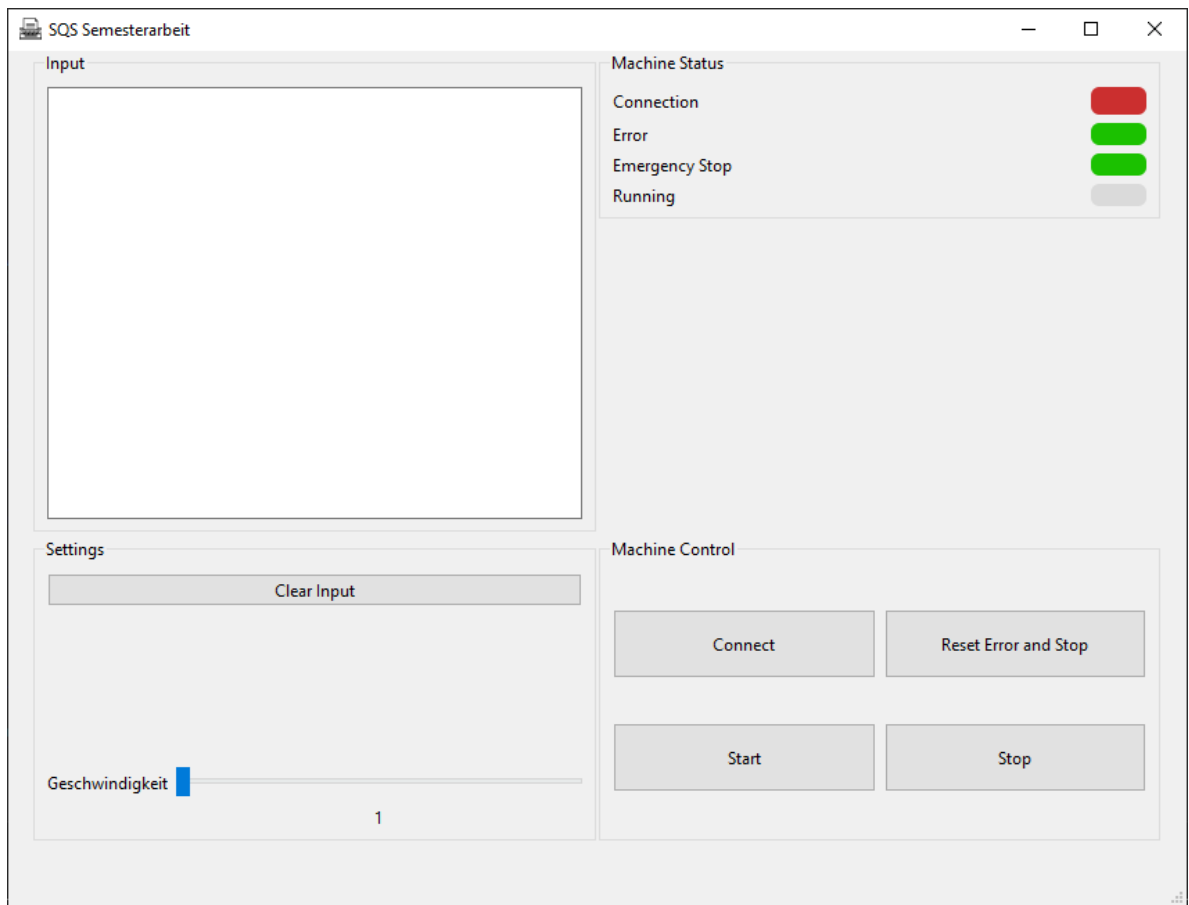


Abbildung 1.1 Anwendung an der die Tests Evaluiert werden

Die Anwendung (Abbildung 1.1) lässt sich in 4 Bereiche Unterteilen. Der erste Bereich, links oben, ist ein Text Feld. Dieses Text Feld kann mit User Input gefüllt werden. Der Input wird durch drücken des Start Buttons im vierten Bereich, rechts unten, evaluiert. Wenn der Input 'disconnect' oder 'error' beinhaltet werden entsprechende Funktionen ausgeführt und der Status der Maschine verändert.

Im zweiten Bereich, rechts oben, sind 4 Statusanzeigen zu sehen. Die Statusanzeigen Signalisieren durch Farben den Status der Maschine. Wenn eine aktive Verbindung besteht ist die Anzeige der Connection grün, andernfalls rot. Falls beim ausführen des Inputs aus dem ersten Bereich ein Fehler auftritt wird die Error Anzeige rot. Wenn kein Fehler auftritt ist sie grün eingefärbt. Die Emergency Stop Anzeige zeigt an wenn der Nothalt Schalter gedrückt wurde mit roter Farbe. Falls der Nothalt Schalter nicht gedrückt wurde ist die Anzeige grün. Die letzte Anzeige ist grün wenn gerade ein Script aus dem Inputfeld ausgeführt wird, und bleibt grün sofern keine Fehler auftreten oder Stop gedrückt wird.

Im dritten Bereich, links unten, befinden sich Settings zur Maschine. Mit dem Button Clear Input werden alle Zeichen aus dem Input Feld aus dem ersten Bereich entfernt. Durch den Slider kann die Geschwindigkeit innerhalb des Wertebereichs 1 bis 10 eingestellt werden. Der eingestellte Geschwindigkeitswert wird durch ein Label unter dem Slider angezeigt.

Im letzten Bereich, rechts unten, befinden sich 4 Buttons. Der erste Button stellt eine Verbindung her. Der zweite Button setzt den Error Status zurück. Die letzten beiden Buttons sind zum Starten und Stoppen der Maschine.

Tabelle 1.1 Anforderungstabelle

ID	Anforderung	Metrik
01	Kosten	Kosten sollen so gering wie möglich sein
02	Lizenz	Bestenfalls GPL oder vergleichbar
03	Open Source	Source-Code öffentlich verfügbar
04	Integration in vorhandene Pipeline	Wie viel Zeit benötigt es die Tests in die Pipeline zu integrieren
05	Lesbarkeit	Kann der Testcode von Entwicklern gelesen werden
06	Aufwand	Benötigte Zeit zum erstellen eines Tests
07	Änderbarkeit	Benötigte Zeit zum ändern eines Tests
08	Support / Community	Aktivität und Größe der Community (Stackoverflow, eigenes Forum, ...)
09	Updates	Update-Zyklus des Frameworks
10	Reife / Robustheit	Stabilität des Frameworks
11	In Qt Editor ausführbar	Sind die Tests im Qt Editor ausführbar
12	Abdeckung der Qt Features	Mögliche Abdeckung der mit Qt erstellbaren Widgets
13	Laufzeit	Wie viel Zeit beansprucht die Ausführung eines Tests
14	Dokumentation	Umfang und Qualität der Dokumentation der Tests
15	Auswertbarkeit der Ergebnisse	Lassen sich die Ergebnisse automatisiert auswerten

1.3 Anforderungen

Die Folgende Tabelle gibt einen Überblick über die Anforderungen, welche an die UI Testing Frameworks gestellt werden. Anhand dieser Anforderungen werden die gewählten Frameworks verglichen und eine Entscheidung getroffen. Jede Anforderung ist mit einer ID versehen. Diese IDs werden im Dokument zur Referenzierung verwendet.

Die Kosten die für das UI Testing Framework anfallen sollen so gering wie möglich sein. Bestenfalls sollten keine Kosten anfallen. Dies ist direkt mit der Lizenz verknüpft. Eine GPL Lizenz oder vergleichbares wäre daher vorteilhaft. Wenn der Source Code des UI Testing Frameworks frei verfügbar ist bestünde die Möglichkeit, Änderungen oder Ergänzungen am Framework vorzunehmen, sofern die Lizenz dies erlaubt. Wichtig ist, wie viel Aufwand das Schreiben oder Aufnehmen eines Tests ist, sowie die Zeit die benötigt wird um mit dem Framework familiär zu werden sollte gering sein. Dabei wäre es ebenso von Vorteil wenn man den Testcode lesen, nachvollziehen und verstehen kann. Ebenso wird verglichen wie viel Aufwand eine Änderung eines Tests ist, wenn sich beispielsweise die Position eines Buttons verändert. Wenn das Framework nicht in die Verwendete Pipeline integriert werden kann, kann es nicht automatisiert verwendet werden und würde damit ausgeschlossen werden. Der beste Fall hierfür wäre die Möglichkeit die Tests im Qt Editor ausführen zu können. Ein weiteres Kriterium ist der Umfang des Frameworks. Dabei wird gemessen wie weit sich alle durch Qt bereit gestellten Widgets testen lassen und ob es die Möglichkeit gibt custom Widgets zu testen. Die Laufzeit der Tests ist ebenso ein Kriterium. Wenn sich 10 Tests innerhalb von 2 Minuten ausführen und auswerten lassen. Dabei wird gleichzeitig die Anforderung der automatisierten Testauswertung in Betracht gezogen. Zuletzt wird untersucht

1 Automatisierte UI Tests

wie groß die Community der Frameworks auf seiten wie Stackoverflow, Github oder einem eigenem Forum ist, sowie die die aussicht auf zukünftige Updates

2 Open HMI Tester

2.1 Einführung Open HMI Tester

Open HMI Tester (OHT) ist ein Framework für die Entwicklung von GUI Testing Tools. Es verwendet Echtzeit GUI Beobachtung um reale Nutzerinteraktionen aufzunehmen und abzuspielen. Dies soll Robustheit und Toleranzen für Änderungen während der Testphase bringen.

OHT bietet ein plattformübergreifendes, offenes Design zur Unterstützung von ereignisbasierten GUI-Plattformen. Da es nicht in den Code eingreift kann es in laufende und ältere Entwicklungen integriert werden. Als Ergebnis bietet das Framework eine anpassbare, erweiterbare, skalierbare und robuste Basis zur Unterstützung der Automatisierung von GUI-Testprozessen. Die Tests können einzeln oder alle hintereinander ausgeführt werden.

Eine fertiger Build für Windows Qt-Anwendungen wird von OHT Angeboten. Dieser Build wird in dieser Arbeit verwendet.

2.2 Durchführung der Tests

2.3 Tests

3 Qt Test

3.1 Einführung Qt Test

Qt Test ist ein Teil des Qt Frameworks. Es bietet Klassen für Unit-Tests von Qt-Anwendungen und -Bibliotheken. Alle öffentlichen Methoden befinden sich im QTest-Namensraum. Darüber hinaus bietet die Klasse QSignalSpy eine einfache Introspektion der Signale und Slots von Qt, und der QAbstractItemModelTester ermöglicht das nicht-destruktive Testen von Elementmodellen. Dadurch können auch UIs getestet werden. Es lassen sich Maus und Tastatur Eingaben durch ein von der Anwendung unabhängiges Eventsystem simulieren [Doc22].

Beispiel für einen Test:

```
void TestGui::testGui ()
{
    QLineEdit lineEdit;

    QTest::keyClicks(&lineEdit , "hello_world");

    QCOMPARE(lineEdit.text() , QString("hello_world"));
}
```

Die Funktion TestGui::testGui() erstellt ein Eingabefeld. Durch die Funktion QTest::keyClicks() wird das als Parameter übergebene QWdiget fokussiert und der als Parameter übergebene String als Tastatur Eingabe dem QWdiget übergeben. Mit dem Makro QCOMPARE() werden zwei Werte verglichen. Wenn die beiden übergebenen Werte den selben Inhalt haben wird das Programm fortgeführt. Sollte dies nicht der Fall ist wird der eine Fehlermeldung im Log ausgegeben und die Funktion beendet. Für Mauseingaben gibt es mehrere Funktionen. Unter anderem die Funktionen QTest::mouseMove() und QTest::mouseClick(). QTest::mouseMove() bewegt den Mauszeiger auf die übergebene Position, ausgehend von dem Übergebenen QWdiget. QTest::mouseClick() führt einen Mausklick auf die übergebene Position für die übergebene Zeit aus.

3.2 Durchführung der Tests

3.3 Tests

4 Vergleich

4.1 Open HMI Tester

4.2 Qt Test

4.3 Vergleich

5 Fazit

Literaturverzeichnis

[Cap] S. Capgemini. Microfocus, "World Quality Report 2021". URL: <https://www.capgemini.com/research/world-quality-report-wqr-2021-22>.

[Doc22] Q. Documentation. Qt Test, 2022. [Zugegriffen am 10.06.2022].