Bestehende Testautomatisierung analysieren und modernisieren

Gewachsene Testlösungen systematisch bewerten, stabilisieren und weiterentwickeln

01 INTRO

Testautomatisierung altert wie jede andere Software. Über Jahre entstehen duplizierter Testcode, fragile Selektoren, lange Laufzeiten, unklare Abstraktionen, Flaky Tests und Abhängigkeiten von veralteten Technologien. Ein kompletter Neubau ist trotzdem häufig weder notwendig noch wirtschaftlich sinnvoll.

Der Kurs behandelt die Modernisierung bestehender Testautomatisierung als Engineering- und Migrationsaufgabe. Die Teilnehmer analysieren technische und methodische Schwachstellen, entwickeln ein Zielbild und planen eine schrittweise Verbesserung, ohne das vorhandene Regressionstestnetz leichtfertig aufzugeben.

DAUER

3 Tage

FORMAT

Präsenz · Live Online · Inhouse

VORKENNTNISSE

Praktische Erfahrung mit Testautomatisierung und Programmierkenntnisse

02 TRAININGSPROGRAMM

Inhalte

Warum Testautomatisierung altert

- Änderungen am Testobjekt
- Frameworkentwicklung
- technische Schulden
- Copy-and-Paste
- fehlende Architektur
- instabile Tests
- wachsender Wartungsaufwand
- Vertrauensverlust

Bestehende Lösung erfassen

- Technologien
- Frameworks
- Testumfang
- Testebenen
- Projektstruktur
- Abhängigkeiten
- Ausführungswege
- CI/CD
- Ownership

Testbestand analysieren

- relevante Tests
- Duplikate
- obsolete Tests
- Testabdeckung
- Testebenen
- lange Tests
- unklare Assertions
- fachlicher Wert

Codequalität untersuchen

- Struktur
- Kopplung
- Kohäsion
- Duplikate
- Abstraktionen
- Helper
- Page Objects
- technische Schulden
- Testcode als Produktionscode behandeln

Flaky Tests systematisch analysieren

- Timing
- Synchronisation
- Daten
- Umgebungen
- Reihenfolgen
- Parallelität
- externe Abhängigkeiten
- Ursachen klassifizieren

Laufzeit und Feedback untersuchen

- Suite-Laufzeiten
- langsame Tests
- Testpyramide
- unnötige UI-Tests
- Parallelisierung
- Testselektion
- Pipeline-Struktur
- Feedbackzeit

Architektur bewerten

- Verantwortlichkeiten
- Schichten
- fachliche und technische Abstraktion
- Daten
- Konfiguration
- Diagnosefähigkeit
- Erweiterbarkeit
- Zielarchitektur

Framework- und Technologiewechsel beurteilen

- bestehende Technologie weiterführen
- Upgrade
- Migration
- Selenium zu Playwright als mögliches Beispiel
- Sprache oder Framework wechseln
- Kosten
- Nutzen
- Risiken
- kein Rewrite aus Prinzip

Refactoring-Strategie

- sichere kleine Schritte
- Characterization
- Duplikate entfernen
- Abstraktionen verbessern
- Selektoren stabilisieren
- Daten entkoppeln
- Testverhalten erhalten

Migration ohne Verlust des Sicherheitsnetzes

- Parallelbetrieb
- alte und neue Tests
- schrittweise Migration
- Coverage vergleichen
- kritische Regression erhalten
- Abschaltkriterien
- Rollback
- Fortschritt messen

CI/CD und Betrieb modernisieren

- Pipeline-Struktur
- Container
- Parallelisierung
- Reports
- Tracing
- Artefakte
- Quality Gates
- Ownership und Wartungsprozess

Modernisierungsplan entwickeln

- Ist-Zustand bewerten
- Probleme priorisieren
- Quick Wins identifizieren
- Zielarchitektur definieren
- Migrationsschritte planen
- Risiken berücksichtigen
- Erfolgskriterien festlegen

03 DURCHFÜHRUNG

Rahmen und Organisation

Dauer	3 Tage
Durchführung	Präsenz oder Live Online, vorzugsweise als Inhouse-Schulung
Schwerpunkt	Inhouse-Schulungen für Unternehmen und Teams
Sprache	Deutsch, englische Fachbegriffe und Dokumentation
Unterlagen	Kursunterlagen, Beispiele und Übungsaufgaben
Technik	Eigener Rechner mit Entwicklungsumgebung und Zugriff auf die zu analysierende Testlösung

> **Inhouse-Schulungen**
> Eine vorhandene Lösung oder anonymisierte Ausschnitte daraus eignen sich besonders gut als Grundlage für Analyse und Modernisierungsplanung.

04 ZIELGRUPPE

Für wen ist der Kurs gedacht?

Erfahrene Testautomatisierer, Quality Engineers, Test Architects, Entwickler und QA Leads, die bestehende Automatisierungslösungen übernehmen, sanieren oder technologisch weiterentwickeln müssen.

> **Voraussetzungen**
> Praktische Erfahrung mit Testautomatisierung und grundlegende Software-Engineering-Kenntnisse.

05 LERNZIELE

Was Sie nach dem Kurs können

Nach dem Kurs können die Teilnehmer:

- bestehende Testautomatisierung technisch und methodisch analysieren
- Flakiness, Wartungsprobleme und Architekturdefizite systematisch identifizieren
- den tatsächlichen Wert eines vorhandenen Testbestands beurteilen
- Refactoring, Upgrade und Migration gegeneinander abwägen
- eine schrittweise Modernisierungsstrategie entwickeln
- bestehende Regression während der Modernisierung erhalten
- Verbesserungen anhand nachvollziehbarer Kriterien bewerten

06 FAQ

Häufige Fragen

Geht der Kurs davon aus, dass alte Testautomatisierung ersetzt werden sollte?

Nein. Ein Rewrite ist nur eine mögliche Option und häufig nicht die beste. Zuerst wird untersucht, welche Teile wertvoll sind und mit vertretbarem Aufwand verbessert werden können.

Ist eine Migration von Selenium zu Playwright Bestandteil?

Sie eignet sich als mögliches Szenario, ist aber nicht das eigentliche Thema. Die Methodik gilt ebenso für andere Framework-, Sprach- und Architekturwechsel.

Was unterscheidet den Kurs von der Testautomatisierungsarchitektur?

Der Architekturkurs fragt: **Wie entwerfe ich eine gute Testautomatisierungsarchitektur?** Hier lautet die Frage: **Wie komme ich von einer bereits existierenden, problematischen Lösung kontrolliert zu einer besseren?**

Kann mit einer eigenen Testautomatisierungslösung gearbeitet werden?

Ja. Eine vorhandene Lösung oder anonymisierte Ausschnitte daraus eignen sich besonders gut als Grundlage für Analyse und Modernisierungsplanung.

UC IT Service · Ulrich Cuber · kontakt@uc-it.de
https://www.uc-it.de/coaching/testautomatisierung-modernisieren/