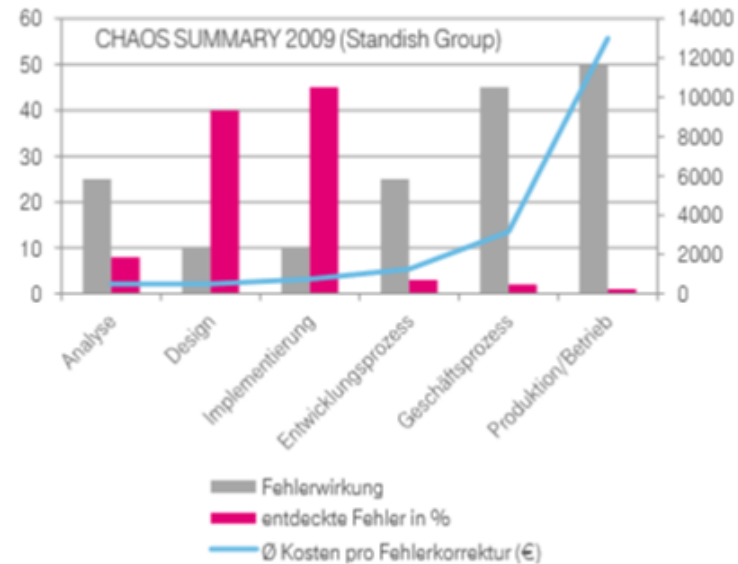




Selenium meets Browserstack:
Cross-Browser-Tests von Websites leicht gemacht.



- „Irren ist menschlich.“
- Menschen erstellen Software.
- Software enthält deshalb Fehler
- „Testen minimiert Fehler.“
- Frühzeitiges Testen spart exponentiell ansteigende Fehlerfolgekosten.
- Testen erhöht Kundenzufriedenheit.
- Testen bedeutet größeren Erfolg.



Testautomatisierung ist die Verwendung von Software:

- um die Test-Preconditions festlegen
- um die Ausführung der Tests zu steuern
- um tatsächliche Ergebnisse mit vorhergesagten Ergebnisse zu vergleichen
- um letztendlich den Ausführungsstatus zu melden

Üblicherweise umfasst Testautomatisierung die Automatisierung von bereits vorhandenen manuellen Prozessen, die ein formalisiertes Testverfahren verwenden.

Warum und wann sollte man automatisieren?

- Bei häufige Regressionstests
- Wiederholte Testfallausführung erforderlich
- User Acceptance Tests
- Smoke Test
- Schnellerer Feedback zu den Entwicklern
- Reduzieren der menschlichen Aufgaben
- Test der gleichen Anwendung auf mehrere Umgebungen

The screenshot shows the Selenium IDE 1.9.1 interface. The window title is "Valid_login.html - Selenium IDE 1.9.1". The menu bar includes File, Edit, Actions, Options, and Help. The Base URL bar shows "http://newtours.demoaut.com/". The toolbar includes a speed slider (Fast to Slow), play, pause, and other icons. The Test Case pane on the left lists "Invalid_login" and "Valid_login". The Table pane shows a list of commands and their targets/values. The Log pane at the bottom shows the execution log with errors.

Annotations:

- Menu bar
- Base URL bar
- Toolbar
- Test Case pane
- Log/Reference/UI-Element/Rollup pane
- Start and Stop Recording
- Accessor Area
- Clear Log

Command	Target	Value
open	/	
type	userName	tutorial
type	password	cause_of_failure
clickAndWait	login	
verifyTitle	Find a Flight: Mercu...	
clickAndWait	link=SIGN-OFF	
verifyTitle	Sign-on: Mercury T...	
clickAndWait	link=Home	
verifyTitle	Welcome: Mercury ...	

Runs: 2
Failures: 1

Log/Reference/UI-Element/Rollup

Info Clear

[info] Changed test case

[info] Executing: |open | / | |

[info] Executing: |type | userName | tutorial |

[info] Executing: |type | password | cause_of_failure |

[info] Executing: |clickAndWait | login | |

[info] Executing: |verifyTitle | Find a Flight: Mercury Tours: | |

[error] Actual value 'Sign-on: Mercury Tours' did not match 'Find a Flight: Mercury Tours:'

[info] Executing: |clickAndWait | link=SIGN-OFF | |

[error] Element link=SIGN-OFF not found

The screenshot displays the Selenium IDE 2.9.1 interface. The top bar shows the title 'TestALittleBitMeinHeine (untitled suite) - Selenium IDE 2.9.1 *'. Below it, the 'Base URL' is set to 'http://www.heine.de/'. The main area is divided into three panes. The left pane shows the 'Test Case' list with 'TestALittleBitMeinHeine *'. The middle pane shows a table of commands and their targets. The right pane shows the details of the selected command, 'verifyElementPresent', with target 'id=login' and value 'id=login'.

Command	Target	Value
open	/	//www.heine.de
clickAndWait	link=Anmelden	An
assertTitle	Heine - Immer etwas Besonderes	//Hello Heine
verifyElementPresent	id=login	
verifyText	//div[@id='txtLogin']/p[2]	Melden Sie sich hier mit Ihrer E-Mail-Adresse un...
type	id=loginId	
type	id=password	
assertText	css=#rememberMe > label	Angemeldet bleiben
verifyValue	name=rememberMe	on
click	name=rememberMe	//die Checkbox anklicken
verifyValue	name=rememberMe	off
waitForElementPresent	css=a.go.js_submitform > span.btGo	//nur mit diesem Befehl funktioniert der clickAnd...
clickAndWait	css=a.go.js_submitform > span.btGo	//Anmeldung
assertText	css=h1	Willkommen Ec Commerce Test B
verifyText	css=span.customerNumber	exact:Kdnr: 25695241
clickAndWait	css=span.ui-btn-text > span	//Bestellungen angeklickt
selectAndWait	id=delivery	label=Artikel noch nicht versendet
verifyText	css=div.information	Es sind keine Bestellungen vorhanden.
clickAndWait	link=zurück zur Übersicht	//selbsterklärend
clickAndWait	//div[@id='main']/div/a[2]/span/span/span	//Meine Buchungen geklickt

Command: verifyElementPresent
Target: id=login
Value: id=login

Runs: 1
Failures: 0

Log [Reference](#) Expert UI-Element Rollup

verifyElementPresent(locator)
Generated from **isElementPresent(locator)**

Arguments:

- locator - an element locator

Returns:

true if the element is present, false otherwise
Verifies that the specified element is somewhere on the page.

Selenium Expert by Samit Badle. Click the inspect button to run inspections.
Check my [blog](#) for help and news on this and my other Selenium IDE plugins.

New Test		
open	/	//www.heine.de
clickAndWait	link=Anmelden	//Anmeldung
assertTitle	Heine - Immer etwas Besonderes	//Hello Heine
verifyElementPresent	id=login	//Wir sind im Login
verifyText	//div[@id='txtLogin']/p[2]	Melden Sie sich hier mit Ihrer E-Mail-Adresse und Ihrem Passwort an, um in den "Mein Heine"-Bereich zu gelangen.
type	id=logonId	
type	id=password	
assertText	css=#rememberMe > label	Angemeldet bleiben
verifyValue	name=rememberMe	on
click	name=rememberMe	//die Checkbox anklicken
verifyValue	name=rememberMe	off
waitForElementPresent	css=a.go.js_submitform > span.btGo	//nur mit diesem Befehl funktioniert der clickAndWait danach
clickAndWait	css=a.go.js_submitform > span.btGo	//Anmeldung
assertText	css=h1	Willkommen Ec Ecommerce Test B
verifyText	css=span.customerNumber	exact:Kdnr: 25695241
clickAndWait	css=span.ui-btn-text > span	//Bestellungen angeklickt
selectAndWait	id=delivery	label=Artikel noch nicht versendet
verifyText	css=div.information	Es sind keine Bestellungen vorhanden.
clickAndWait	link=zurück zur Übersicht	//selbsterklärend
clickAndWait	//div[@id='main']/div/a[2]/span/span/span	//Meine Buchungen geklickt
verifyText	css=h1	Meine Buchungen
verifyText	css=div.overviewContent.dotted > span	Mein Kontostand:
verifyText	css=span.green	EUR 4,00
assertText	css=div.overviewContent.noBorder > span	Zu zahlender Betrag:
verifyText	//div[@id='bankData']/div[2]/span[2]	EUR 0,00
assertText	css=div.bHeaderTitle.pdt10 > span	Buchungsart
select	id=bookingTypeIdFilter	label=Rechnungen
assertText	css=td.center > strong	Zur Zeit liegen keine aktuellen Buchungen vor. Ihr Gesamtkontostand bildet Beträge aus älteren Buchungen ab.
clickAndWait	link=zurück zur Übersicht	
clickAndWait	link=Abmelden	

1.Actions: Commands which change the state of the application are classified as actions, like click on some link, select some options on the page, select a value from drop down etc. When action is performed on page the test will fail if the action is not successful. Most common action commands of selenium are:

- a. Open
- b. ClickAndWait
- c. Click
- d. Type

2.Accessors: These commands check the state of the application and stores the application state in some variable. Some examples of accessors are as follows.

- a. storeTitle
- b. storeText

Accessors are also used to evaluate whether the desired result is present on the page and store the result in the variable.

Examples:

- a. storeTextPresent: If text is found on the page then it stores the boolean value as true else store false.
- b. storeElementPresent: If UI element is present on the page then it stores boolean value as true else stores false.

3.Assertions: Assertions are used like the checkpoints or verification points in automation. Assertion verifies the state of the application conforms to the expected state.

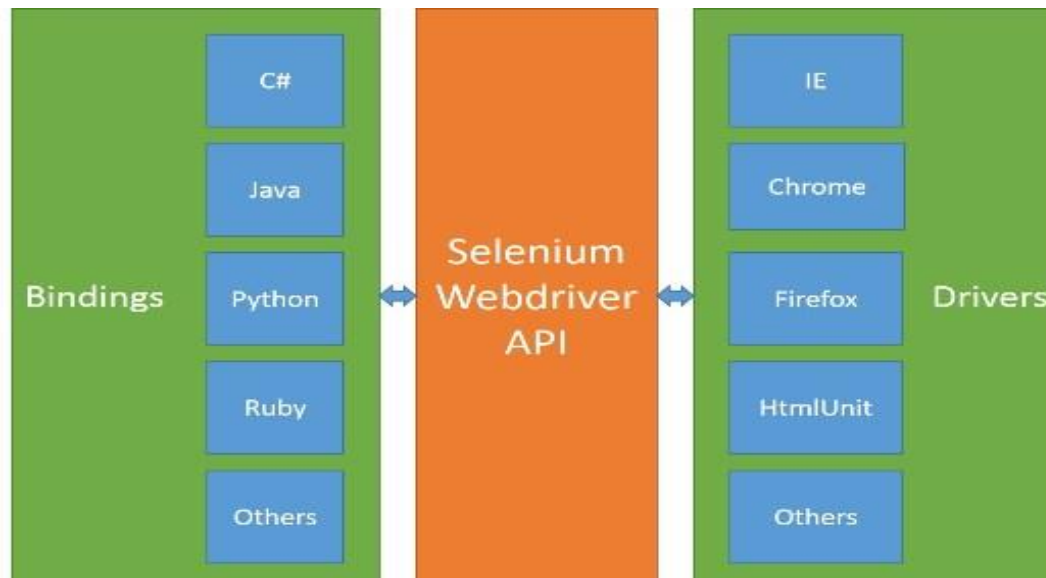
Examples:

- a. verifyElementPresent
- b. verifyText
- c. verifyTitle
- d. assertText

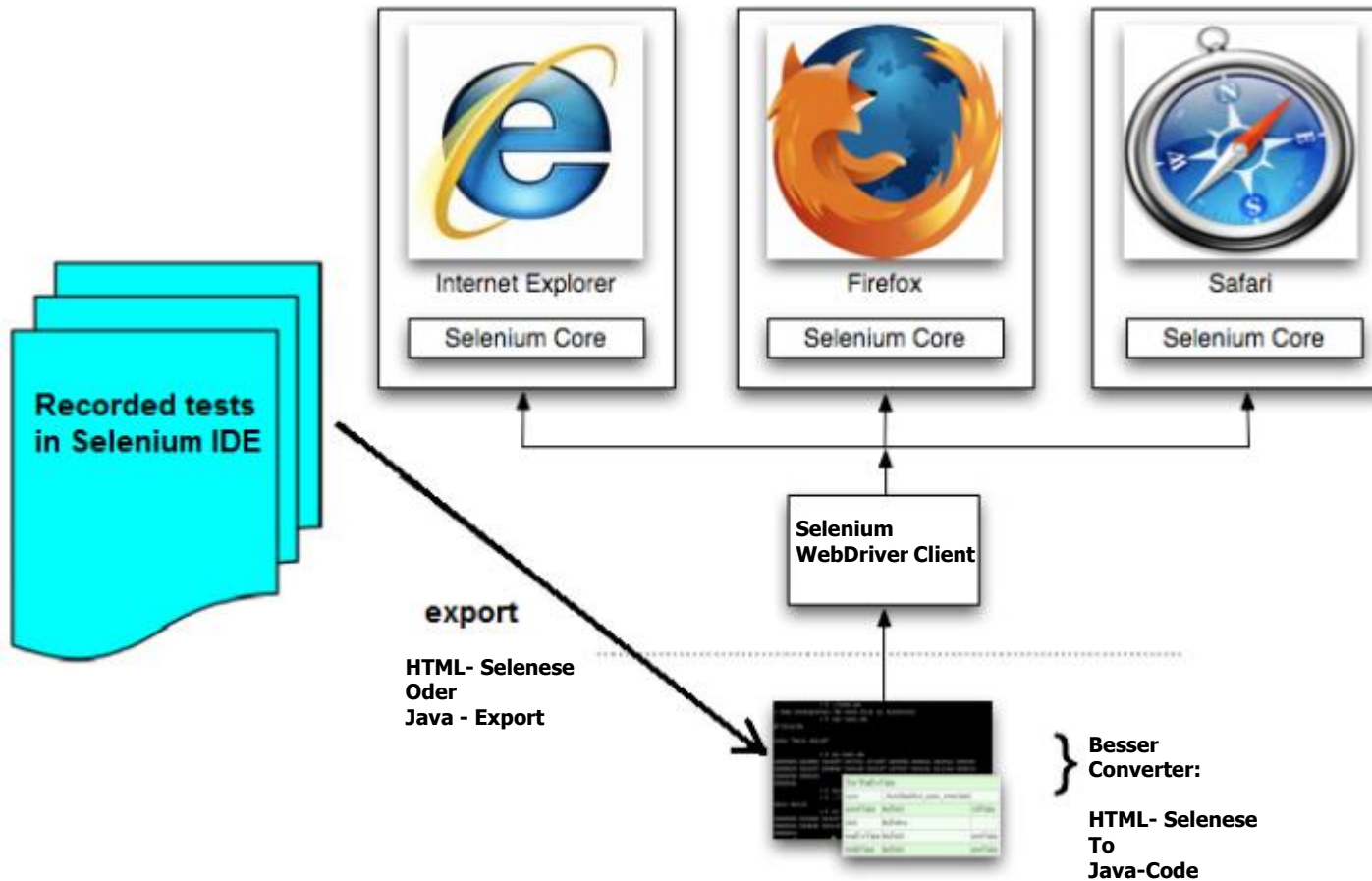
- Macro-Recorder: für Aufzeichnungen der Browser auch von Nicht-Programmierern bedienbar
- Automatischer Test des Firefox-Browsers mit Hilfe von selbstangelegten Test-Suites und Scheduler möglich
- HTML-Test-Scripting übersetzbar in die geläufigsten Webserver-Sprachen u.a. Java, C#, Python (Java unsere Wahl hier)
- Selenese Scripting mit Assertion-und Verify-Anweisungen bieten eine effiziente Möglichkeit des Vergleichens von erwarteten und den tatsächlichen Ergebnissen.
- Primitive Berichterstattung (Ampel und Log)

- Unterstützt aktuell kein Cross-Browser Testing. Der veraltete Selenium RC-Server ist keine Option mehr.
- Programmierung (wie Schleifen, Bedingungen) kann auf der IDE nur mit weiteren Addons angewandt werden
- Selenium IDE kann die Skripte in nur Selenese erstellt auszuführen.
- Es ist schwierig, Selenium IDE für die Überprüfung komplexer Testfälle mit dynamischen Inhalten benutzen
- Keine Integration in DevOps-Umgebung möglich

- Die Selenium-Tests sollen mit allen gängigen Browsern und Devices ausgeführt werden
- Adaption von JUnit und TestNG und mit allen Java-Libraries möglich machen
- Daher Einsatz der Selenium Webdriver API



- Umwandlung der Selenes - Testcases in hochwertige Programmiersprache und in die Selenium Webdriver API notwendig, wir nehmen im weiteren Beispiel Java-Code
- Die IDE leistet dies mit einer Konvertierung, jedoch ist langfristig eine selber geschriebene Konvertierung qualitativ besser



```
package de.heine.testing.clazz.creation;

import java.io.File;

public class HtmlParser {
    private Pattern testTitleTag;
    private Pattern testTbodyTag;
    private Pattern testTdefTag;
    private Pattern testcaseTag;
    private Pattern tagPattern;
    private String clazzTitle;

    public HtmlParser() {
        testTitleTag = Pattern.compile("<title>(.*?)</title>");
        testTbodyTag = Pattern.compile("<tbody>(.*?)</tbody>");
        testTdefTag = Pattern.compile("<td>(.*?)</td>");
        testcaseTag = Pattern.compile("<tr><td><a href=\"(.*?)\">(.*?)</a></td></tr>");
    }

    public Iterator<ClazzInfo> readTestcaseTags(String html, String path) {

        String stringToParse = html;

        // Find class name
        tagPattern = testTitleTag;
        Matcher tagmatch = tagPattern.matcher(stringToParse);
        if (tagmatch.find()) {
            clazzTitle = tagmatch.group().replace("<title>", "").replace("</title>", "");
        }

        // Find method body
        tagPattern = testTbodyTag;
        tagmatch = tagPattern.matcher(stringToParse);
        if (tagmatch.find()) {
            String tBody = tagmatch.group().replace("<tbody>", "").replace("</tbody>", "");
            stringToParse = tBody;
        }

        // Add methods to the TreeMap
        TreeMap<Integer, ClazzInfo> map = new TreeMap<Integer, ClazzInfo>();
        tagPattern = testTdefTag;
        int i = 0;
        int j = 0;
        tagmatch = tagPattern.matcher(stringToParse);

        // Create class info to fill;
        ClazzInfo ci = new ClazzInfo();
        while (tagmatch.find()) {
            i++;
            j++;
            ci.clazzTitle = path;
            String tDef = tagmatch.group().replace("<td>", "").replace("</td>", "");
```

New Test		
open	/	//www.heine.de
clickAndWait	link=Anmelden	//Anmeldung
assertTitle	Heine - Immer etwas Besonderes	//Hello Heine
verifyElementPresent	id=login	//Wir sind im Login
verifyText	//div[@id='txtLogin']/p[2]	Melden Sie sich hier mit Ihrer E-Mail-Adresse und Ihrem Passwort an, um in den "Mein Heine"-Bereich zu gelangen.
type	id=logonId	
type	id=password	
assertText	css=#rememberMe > label	Angemeldet bleiben
verifyValue	name=rememberMe	on
click	name=rememberMe	//die Checkbox anklicken
verifyValue	name=rememberMe	off
waitForElementPresent	css=a.go.js_submitform > span.btGo	//nur mit diesem Befehl funktioniert der clickAndWait danach
clickAndWait	css=a.go.js_submitform > span.btGo	//Anmeldung
assertText	css=h1	Willkommen Ec Ecommerce Test B
verifyText	css=span.customerNumber	exact:Kdnr: 25695241
clickAndWait	css=span.ui-btn-text > span	//Bestellungen angeklickt
selectAndWait	id=delivery	label=Artikel noch nicht versendet
verifyText	css=div.information	Es sind keine Bestellungen vorhanden.
clickAndWait	link=zurück zur Übersicht	//selbsterklärend
clickAndWait	//div[@id='main']/div/a[2]/span/span/span	//Meine Buchungen geklickt
verifyText	css=h1	Meine Buchungen
verifyText	css=div.overviewContent.dotted > span	Mein Kontostand:
verifyText	css=span.green	EUR 4,00
assertText	css=div.overviewContent.noBorder > span	Zu zahlender Betrag:
verifyText	//div[@id='bankData']/div[2]/span[2]	EUR 0,00
assertText	css=div.bHeaderTitle.pdt10 > span	Buchungsart
select	id=bookingTypeIdFilter	label=Rechnungen
assertText	css=td.center > strong	Zur Zeit liegen keine aktuellen Buchungen vor. Ihr Gesamtkontostand bildet Beträge aus älteren Buchungen ab.
clickAndWait	link=zurück zur Übersicht	
clickAndWait	link=Abmelden	

```

import java.util.regex.Pattern;
import java.util.concurrent.TimeUnit;
import org.junit.*;
import static org.junit.Assert.*;
import static org.hamcrest.CoreMatchers.*;
import org.openqa.selenium.*;
import org.openqa.selenium.firefox.FirefoxDriver;
import org.openqa.selenium.support.ui.Select;

public class Test1 {
    private WebDriver driver;
    private String baseUrl;
    private boolean acceptNextAlert = true;
    private StringBuffer verificationErrors = new StringBuffer();

    @Before
    public void setUp() throws Exception {
        driver = new FirefoxDriver();
        baseUrl = "http://www.heine.de/";
        driver.manage().timeouts().implicitlyWait(30, TimeUnit.SECONDS);
    }

    @Test
    public void test1() throws Exception {
        driver.get(baseUrl + "/");
        driver.findElement(By.linkText("Anmelden")).click();
        assertEquals("Heine - Immer etwas Besonderes", driver.getTitle());
        try {
            assertTrue(isElementPresent(By.id("login")));
        } catch (Error e) {
            verificationErrors.append(e.toString());
        }
        try {
            assertEquals("Melden Sie sich hier mit Ihrer E-Mail-Adresse und Ihrem Passwort an, um in \n den \n \"Mein Heine\"-Be");
        } catch (Error e) {
            verificationErrors.append(e.toString());
        }
        driver.findElement(By.id("logonId")).clear();
        driver.findElement(By.id("logonId")).sendKeys(" ");
        driver.findElement(By.id("password")).clear();
        driver.findElement(By.id("password")).sendKeys(" ");
        assertEquals("Angemeldet bleiben", driver.findElement(By.cssSelector("#rememberMe > label")).getText());
        try {
            assertEquals("on", driver.findElement(By.name("rememberMe")).getAttribute("value"));
        } catch (Error e) {
            verificationErrors.append(e.toString());
        }
        driver.findElement(By.name("rememberMe")).click();
        try {
            assertEquals("off", driver.findElement(By.name("rememberMe")).getAttribute("value"));
        } catch (Error e) {
            verificationErrors.append(e.toString());
        }
        for (int second = 0;; second++) {
            if (second >= 60) fail("timeout");
            try { if (isElementPresent(By.cssSelector("a.go.js_submitform > span.btGo"))) break; } catch (Exception e) {}
            Thread.sleep(1000);
        }
        driver.findElement(By.cssSelector("a.go.js_submitform > span.btGo")).click();
        assertEquals("Willkommen Ec Ecommerce Test B", driver.findElement(By.cssSelector("h1")).getText());
        try {
            assertEquals("Kdnr: 25695241", driver.findElement(By.cssSelector("span.customerNumber")).getText());
        } catch (Error e) {
    
```

```
package de.heine.test.ui.tests.steps.Bestellstrecke2;

/* Created by aSchuette on 13.03.2017. */

import de.heine.test.ui.core.utils.WebDriverUtils;

@Generated(value = { "de.heine.testing.clazz.creation.HtmlParser" })
public class DirektBestellung_FalscheEingaben {

    public static void desktop (WebDriver driver, TestParameter task, HashMap<Integer, String[]> IDS) {
        WebDriverUtils.println("...starting DirektBestellung_FalscheEingaben");
        WebDriverUtils.waitForElementPresent(By.xpath("//div[@id='addButton']/a/span"));
        WebDriverUtils.findElement(By.id("artNo_1")).clear();
        WebDriverUtils.sendKeys(By.id("artNo_1"), "");
        WebDriverUtils.findElement(By.xpath("//div[@id='addButton']/a/span")).click();
        WebDriverUtils.sleep(); // leere Eingabe



























        assertTrue(WebDriverUtils.isElementPresent(By.id("productErrorHeadline")));
        assertTrue(WebDriverUtils.isElementPresent(By.id("productError")));
        assertTrue(WebDriverUtils.isElementPresent(By.id("artNo_1"))); // error: highlight red ArtikelNo.












        WebDriverUtils.findElement(By.id("artNo_1")).clear();
        WebDriverUtils.sendKeys(By.id("artNo_1"), "123456"); // 123456
        WebDriverUtils.findElement(By.xpath("//div[@id='addButton']/a/span")).click();
        WebDriverUtils.sleep(); // ungültige ArtNo.

        WebDriverUtils.waitForElementPresent(By.id("productErrorHeadline"));
        assertTrue(WebDriverUtils.isElementPresent(By.id("productErrorHeadline")));
        assertTrue(WebDriverUtils.isElementPresent(By.id("productError")));
        assertTrue(WebDriverUtils.isElementPresent(By.id("artNo_1"))); // error: highlight red ArtikelNo.

        WebDriverUtils.findElement(By.id("artNo_1")).clear();
        WebDriverUtils.sendKeys(By.id("artNo_1"), "00258086"); // 00258086
        WebDriverUtils.findElement(By.id("size_1")).clear();
        WebDriverUtils.sendKeys(By.id("size_1"), "99"); // 99
        WebDriverUtils.findElement(By.xpath("//div[@id='addButton']/a/span")).click();
        WebDriverUtils.sleep(); // Falsche Größenangabe

        WebDriverUtils.waitForElementPresent(By.id("productErrorHeadline"));
        assertTrue(WebDriverUtils.isElementPresent(By.id("productErrorHeadline")));
        assertTrue(WebDriverUtils.isElementPresent(By.id("productError")));
        assertTrue(WebDriverUtils.isElementPresent(By.cssSelector("input#size_1.directOrderUnitInputSize.pliError"))); // error: hi
```


Quick Launch	Samsung		Google		Others	
 Android	 Galaxy S7	6	 Pixel	O Dev	LG	
 iOS	 Galaxy S6	5	 Pixel XL	7.1	 G5	6
	 Galaxy S5	4.4	 Pixel	7.1	Motorola	
 Windows Phone	 Galaxy S4	4.4	 Nexus 6P	7	 Moto X 2nd Gen	6
 Windows	 Galaxy Note 4	4.4	 Nexus 5X	7	 Moto X 2nd Gen	5
	 Galaxy Note 3	4.3	 Nexus 6	6	 Moto G 2nd Gen	5
	 Galaxy Tab 4 10.1	4.4 Tablet	 Nexus 6	5	HTC	
	Galaxy S3	4.1	 Nexus 5	4.4	One M8	4.4
	Galaxy S2	2.3	 Nexus 9	5.1 Tablet	Amazon	
	Galaxy S	2.2	 Nexus 7	6 Tablet	Kindle Fire HDX 7	4.3 Tablet
	Galaxy Note 2	4.1	 Nexus 7	4.4 Tablet	Kindle Fire HD 8.9 *	4 Tablet
	Galaxy S5 Mini	4.4	Nexus 4	4.2	Sony	
	Galaxy Note 10.1	4 Tablet			Xperia Tipo *	4
	Galaxy Tab 2 10.1	4 Tablet				
Drag a browser here to add to Quick Launch						

Quick Launch	iPhone		iPad	
 Android	 iPhone 6S Plus	9	 iPad Air 2	8
 iOS	 iPhone 6S	9	 iPad Air	7
 Windows Phone	 iPhone 6 Plus	8	 iPad 4	7
 Windows	 iPhone 6	8	 iPad Mini 3	8
 10	 iPhone 5S	7	 iPad Mini 2	7
 8.1	 iPhone 7 Plus	10	 iPad Pro	10
 8	 iPhone 7	10	 iPad Pro	9.3
 7	 iPhone SE	10	 iPad Air 2	9.3
 XP	 iPhone 5	6	 iPad Air	8.3
 Mac	 iPhone 4S	6	 iPad Mini 4	9.3
	 iPhone 4S	5.1	 iPad Mini 2	8.3
	 iPhone 4	4	 iPad Mini	7
	 iPhone 3GS	3		
	Drag a browser here to add to Quick Launch			

FEATURES

- Browserstack erleichtert Entwicklern die Arbeit erheblich, weil nicht jedes Gerät und jede Browserversion manuell getestet werden muss. Stattdessen können mit dem Tool Screenshots für die Darstellung in verschiedenen Browsern erstellt oder die Anzeige auf mobilen Geräten, die mit iOS, Android oder Opera laufen, emuliert werden.
- Zudem können verschiedene Browser **durch eine virtuelle Maschine im Browserfenster live getestet** werden. Auch können unterschiedliche Auflösungen ausgewählt werden, mit denen getestet werden soll. Ebenfalls praktisch ist die Möglichkeit, das **Responsive Design** der Website auf verschiedenen Geräten testen zu lassen.
- Weitere Features von Browserstack sind beispielsweise die Optionen, mit den **Developer-Tools** des virtuellen Browsers Probleme zu beheben oder Screenshots der Website zu erfassen und zu kommentieren.
- Weiterhin können inzwischen auch Apps getestet werden

Browserstack hilft damit Entwicklern bei der Analyse von Websites. Wie demonstriert, hält es zahlreiche Browser auf verschiedenen Plattformen bereit, auf denen sich sowohl öffentliche als auch lokale Webangebote ausprobieren lassen; Automatisierung über eine JavaScript-API möglich; Testzugang erfordert Registrierung

heine

Mein Konto Merkzettel Warenkorb

Suchbegriff/Artikel...

SWITCH

1920 x 1042

AKTIONSCOD 10817

MUTTERTAG 10817% AUF ALLES

Jetzt einlösen

Pantolette

★★★★★

Erste Bewertung schreiben >

- Aus hochwertigem Feinsynthetik in Nappa-Optik
- Besondere Optik durch Zipperlement und Nieten
- Kelabsatz mit trendy Plateau in Kork-Optik
- Moderes Design - sommerlich luftig
- Feminin, verleiht jedem Outfit den besonderen Touch

...mehr Artikeldetails >

Artikel-Nr: 078083Y

EUR 59,90

Preise inkl. gesetzl. MwSt. und zzgl. Service- und Versandkosten >

✓ lieferbar in 1-3 Werktagen

In den Warenkorb >

Auf den Merkzettel >

Unsere Moderation empfiehlt:

Miele. Für das, was wir besonders lieben.

Inspector Console Debugger Style Editor Performance Memory Network

Search HTML

Rules Computed Animations Fonts

Filter Styles

Pseudo-elements

This Element

element { inline

body { _unsortable.less:4018

margin: 0;

padding: 0;

color: #333;

width: 100%;

It looks like you haven't started Firefox in a while. Do you want to clean it up for a fresh, like-new experience? And by the way, welcome back!

Refresh Firefox...

```
{
  "server": "hub-cloud.browserstack.com",
  "user": "browserstackhein1",
  "key": "XXXXXXXXXXXXXXXXXXXX",
  "capabilities": {
    "browserstack.debug": true,
    "browserstack.video": true,
    "browserstack.local": false,
    "browserstack.selenium_version": "2.45.0"
  },
  "environments": [
    {
      "browser": "Firefox",
      "browser_version": "51.0",
      "os": "Windows",
      "os_version": "10",
      "resolution": "2048x1536"
    },
    {
      "browser": "Chrome",
      "os": "Windows",
      "os_version": "10",
      "resolution": "2048x1536"
    },
    {
      "os": "Windows",
      "os_version": "10",
      "browser": "IE",
      "resolution": "2048x1536"
    },
    {
      "os": "OS X",
      "os_version": "El Capitan",
      "browser": "Safari",
      "browser_version": "9.1",
      "resolution": "1600x1200"
    },
    {
      "browserName": "android",
      "platform": "ANDROID",
      "device": "Google Nexus 5"
    },
    {
      "browserName": "iPhone",
      "platform": "MAC",
      "device": "iPhone 6S Plus"
    }
  ]
}
```

Here is a sample test case written for running with JUnit.

```
public class SingleTest extends BrowserStackJUnitTest {

    @Test
    public void test() throws Exception {
        driver.get("https://www.google.com/ncr");
        WebElement element = driver.findElement(By.name("q"));
        element.sendKeys("BrowserStack");
        element.submit();
        Thread.sleep(5000);







        assertEquals("BrowserStack - Google Search", driver.getTitle());
    }
}
```

Here is a sample test case written using Selenide for TestNG.

```
public class SingleTest extends BrowserStackTest {

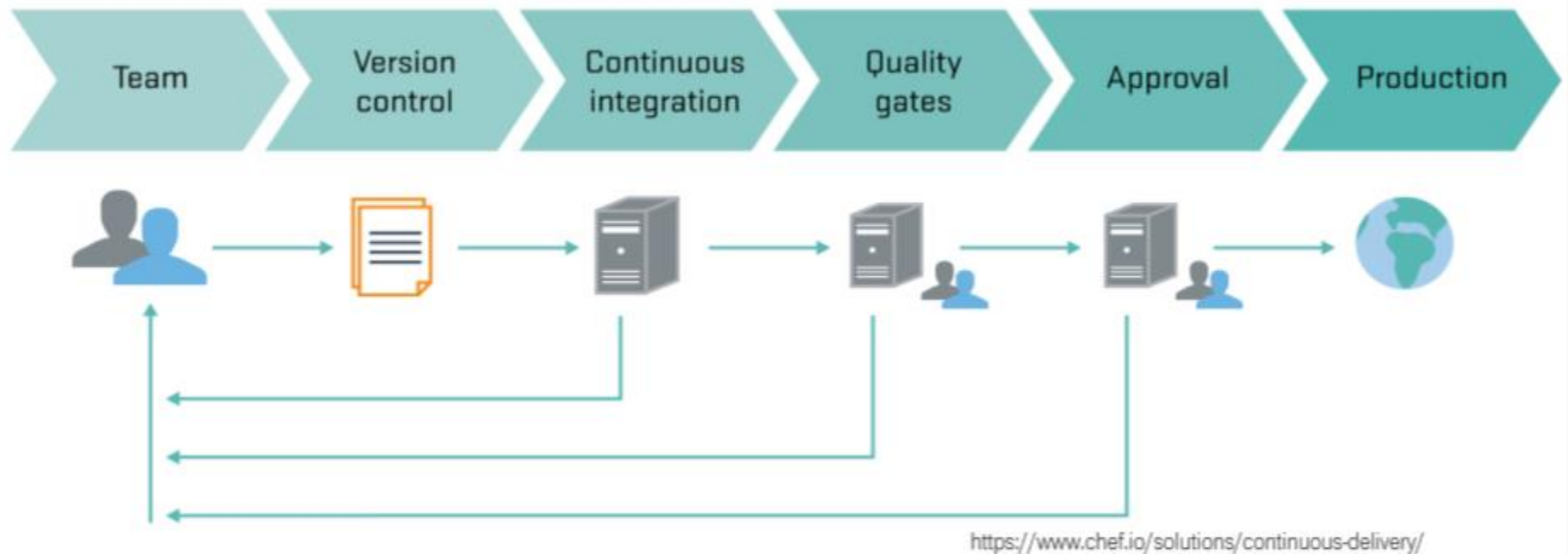
    @Test
    public void test() throws Exception {
        open("http://www.google.com");
        $(By.name("q")).setValue("BrowserStack").pressEnter();
        sleep(2000);
        Assert.assertEquals(title(), "BrowserStack - Google Search");
    }
}
```

Languages and Frameworks

 Java Gauge JBehave JUnit Selenide Serenity TestNG	 Node JS Cucumber JS Intern Nightwatch Protractor WD WebdriverIO	C# MBUnit NUnit PUnit Specflow	 PHP Behat Codeception PHPUnit
 Python Behave Lettuce	 Ruby Capybara RSpec	 Perl	

Continuous Integration

 Jenkins Plugin	 Travis CI add-on	 TeamCity Plugin	 Bamboo Plugin
---	---	--	--

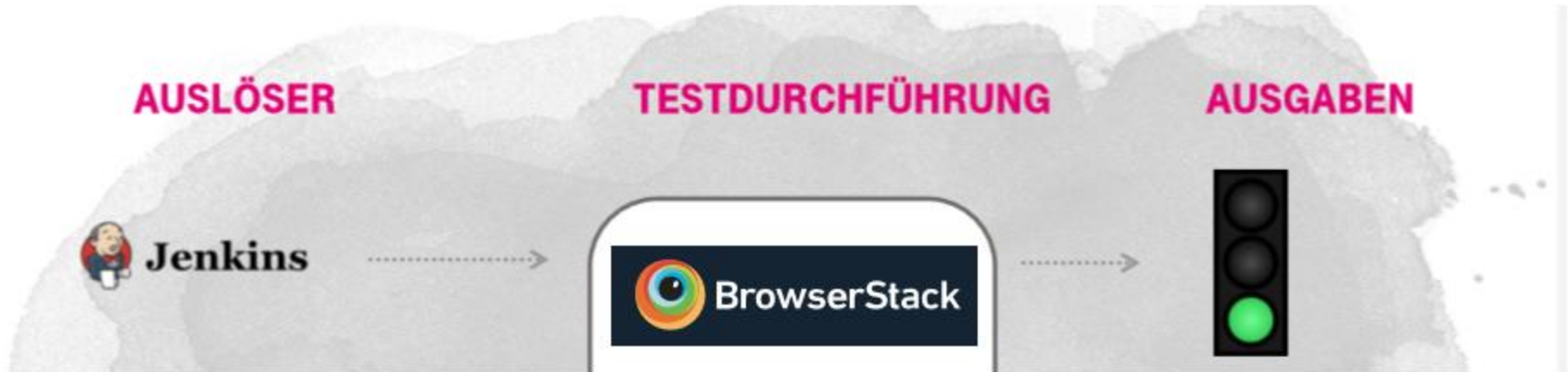


Schnelle Aussage über SW-Qualität



Fundierte Aussage über SW-Qualität





Automatisierung auf Basis eines
Automatisierungsframeworks

Testläufe tracken

Testergebnisreporting per Mail

BrowserStack

[View this on BrowserStack Automate Dashboard](#)

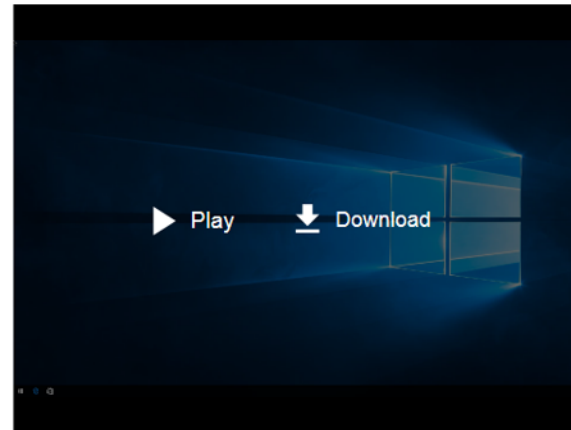


Note: Test on real mobile devices! Pass the capabilities realMobile = true and device = Google Pixel

Session: QA1 instance

Project: HomePageGTM_desktop, Build: [IE010140_12.04.2017 14:44:55](#)

Started 12:42 UTC 12 Apr 2017
Duration 26 secs
Status ✔ Completed
Platform Windows 10
Browser Chrome 57.0
Local testing False
User name Browserstack Heine (browserstackheint)
[Input Capabilities](#) [Browser Capabilities](#)



Text Logs

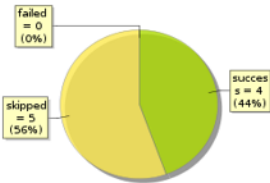
Visual Logs

[Raw Logs](#)

Start	Duration	Action	Expand all ▼
00:00	12	Starting Browser	
00:12	1	Maximize window	

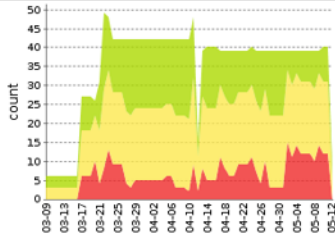
Job statistics

Test Statistics Chart



Latest builds

Test Trend Chart



Build statistics

Status of the build	Description	Number of builds	Percentage of total builds
!	Fehigeschlagen	19	86.36
!	Instabil	0	
✓	Erfolgreich	2	9.09
—	Bevorstehend	0	
—	Deaktiviert	0	
—	Abgebrochen	1	4.55
—	Nicht gebaut	0	
Total builds	All builds	22	

[Wiki](#)
[IBM KnowBase](#)
[Redmine](#)
[GitLab](#)
[Jenkins](#)
[Wichtige URLs](#)
[Fernwartung](#)
[Mailinglisten](#)
[DEV](#)
[Cache QA1](#)
[Cache local](#)
[SAC](#)
[CMC QA1](#)
[Integrated Solutions C...](#)
[CMC DEV](#)
[Routine Review Neue...](#)
[Agiles Board Tickets B...](#)
[WebPagetest - Websit...](#)
[WebPagetest - Websit...](#)

heine

Suchen

ANMELDEN

[Jenkins](#)
[BrowserStack](#)
[GTM_TEST](#)
[#13](#)
[Testergebnis](#)
[de.heine.test.ui.tests.gtm](#)
[HomePageGTM](#)
[desktop\[QA1 instance_Chrome_Windows_10\]](#)

AUTO-AKTUALISIERUNG EINSCHALTEN

[← Zurück zum Projekt](#)

Status

Änderungen

Konsolenausgabe

Build-Informationen anzeigen

Verlauf

Environment Variables

Git Build Data

Testergebnis

Vorheriger Build

Erfolg

de.heine.test.ui.tests.gtm.HomePageGTM.desktop[QA1 instance_Chrome_Windows_10] (from de.heine.test.ui.testsuite.GoogleTagManagerTestSuit)

Dauer: 29 Sekunden.

BrowserStack

View this on BrowserStack Automate Dashboard

BrowserStack

Note: Test on real mobile devices! Pass the capabilities realMobile = true and device = Google Pixel

Session: QA1 instance

Project: HomePageGTM_desktop, Build: [IE010140_12.04.2017 14:44:55](#)

Started

12:42 UTC 12 Apr 2017

Duration

26 secs

Status

Completed

Platform

Windows 10

Browser

Chrome 57.0

Local testing

False

User name

Browserstack Heine (browserstackheint)

[Input Capabilities](#)

[Browser Capabilities](#)

Text Logs

Visual Logs

[Raw Logs](#)

Start	Duration	Action	Expand all
00:00	12	Starting Browser	
00:12	1	Maximize window	

15:25

12.05.2017



Vielen Dank!

Fragen??