Software Testing

#05 - Software Testing Techniques

Herman Kabetta

Software Testing Techniques

- Boundary Value Analysis (BVA)
- Equivalence Class Partitioning
- Decision Table based testing.
- State Transition
- Error Guessing

Boundary Value Analysis (BVA)

- Boundary value analysis is based on testing at the boundaries between partitions.
- Includes maximum, minimum, inside or outside boundaries, typical values and error values

Boundary Value Analysis (BVA)

- If an input condition is restricted between values x and y, then the test cases should be designed with values x and y as well as values which are above and below x and y.
- If an input condition is a large number of values, the test case should be developed which need to exercise the minimum and maximum numbers. Here, values above and below the minimum and maximum values are also tested.
- Apply guidelines 1 and 2 to output conditions. It gives an output which reflects the minimum and the maximum values expected. It also tests the below or above values.

Boundary Value Analysis (BVA)

- Input condition is valid between 1 to 10
- Boundary values 0,1,2 and 9,10,11

Equivalence Class Partitioning

- Equivalent Class Partitioning allows you to divide set of test condition into a partition which should be considered the same.
- This method divides the input domain of a program into classes of data from which test cases should be designed.

Equivalence Class Partitioning

Input conditions are valid between

```
1 to 10 and 20 to 30
```

Hence there are three equivalence classes

```
--- to 0 (invalid)
1 to 10 (valid)
11 to 19 (invalid)
20 to 30 (valid)
31 to --- (invalid)
```

You select values from each class, i.e.,

```
-2, 3, 15, 25, 45
```

Decision Table Based Testing

- Also known as to Cause-Effect table.
- This technique is used for functions which respond to a combination of inputs or events.
- For example, a submit button should be enabled if the user has entered all required fields.

Decision Table Based Testing

Steps :

- Enlist the inputs in rows
- Enter all the rules in the column
- Fill the table with the different combination of inputs
- In the last row, note down the output against the input combination.

Decision Table Based Testing

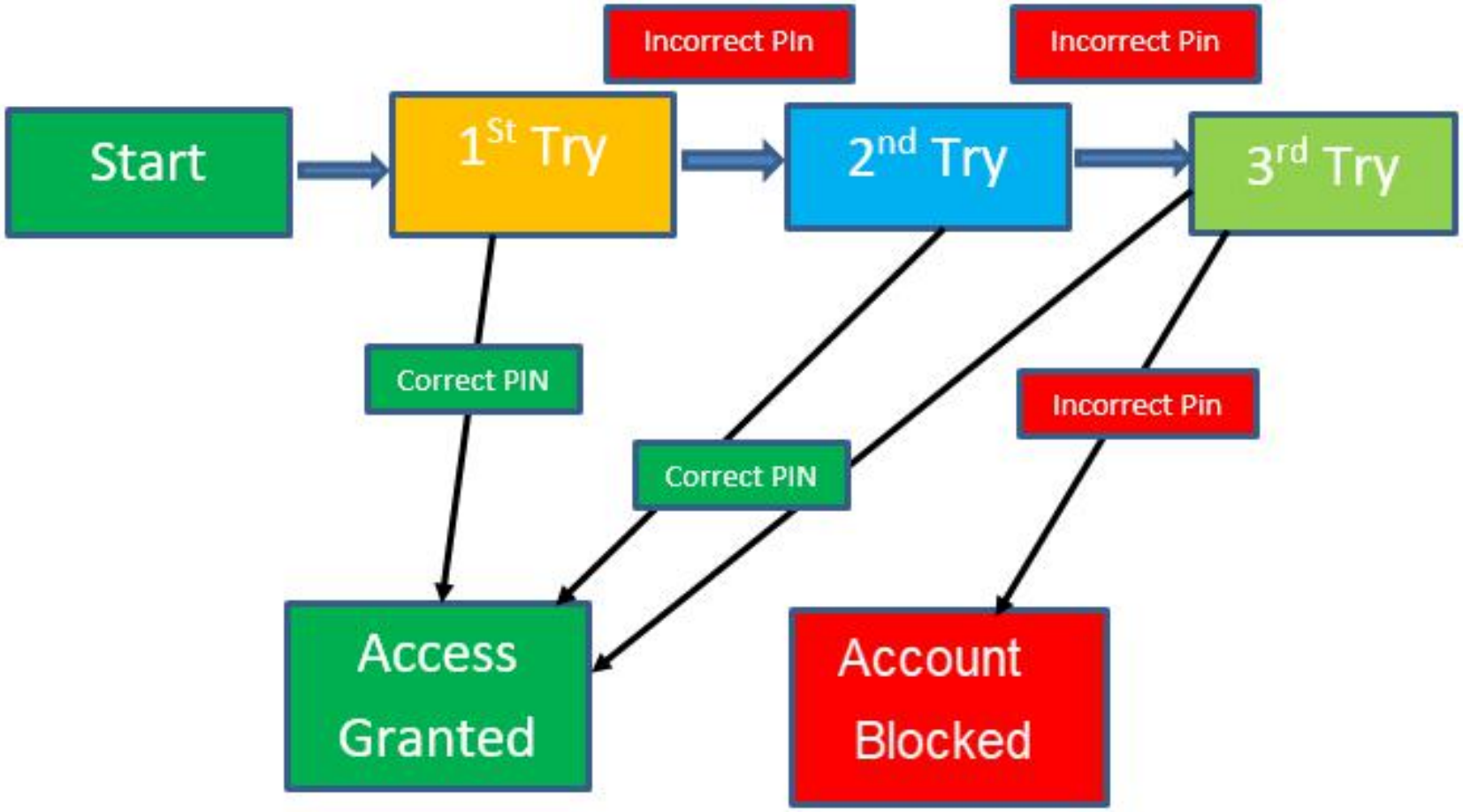
	Rule 1	Rule 2	Rule 3	Rule 4	Rule 5	Rule 6	Rule 7	Rule 8
Input								
Name	F	T	F	T	F	T	F	T
Email	F	F	T	T	F	F	T	T
Message	F	F	F	F	T	T	T	T
Output								
Submit	F	F	F	F	F	F	F	T

State Transition

- Changes in input conditions change the state of the Application Under Test (AUT).
- Allows the tester to test the behavior of an AUT.
- Can perform this action by entering various input conditions in a sequence.
- Testing team provides positive as well as negative input test values for evaluating the system behavior.

State Transition

- Diagram



State Transition

- Table

	Correct PIN	Incorrect PIN
S1) Start	S5	S2
S2) 1st attempt	S5	S3
S3) 2nd attempt	S5	S4
S4) 3rd attempt	S5	S6
S5) Access Granted	-	-
S6) Account blocked	-	-

Error Guessing

- Based on guessing the error which can prevail in the code.
- An experience-based technique where the test analyst uses his/her or experience to guess the problematic part of the testing application.
- Counts a list of possible errors or error-prone situations, Then tester writes a test case to expose those errors.

Manual Testing VS Automated Testing

- Manual testing is testing of the software where tests are executed manually by a QA Analysts.
- In Automated Software Testing, testers write code/test scripts to automate test execution. Testers use appropriate automation tools to develop the test scripts and validate the software.

Manual Testing

Pros of Manual Testing:

- Get fast and accurate visual feedback
- It is less expensive as you don't need to spend your budget for the automation tools and process
- Human judgment and intuition always benefit the manual element
- While testing a small change, an automation test would require coding which could be time-consuming. While you could test manually on the fly.

Cons of Manual Testing:

- Less reliable testing method because it's conducted by a human. Therefore, it is always prone to mistakes & errors.
- The manual testing process can't be recorded, so it is not possible to reuse the manual test.
- In this testing method, certain tasks are difficult to perform manually which may require an additional time of the software testing phase.

Automated Testing

Pros of automated testing:

- Automated testing helps you to find more bugs compare to a human tester
- As most of the part of the testing process is automated, you can have a speedy and efficient process
- Automation process can be recorded. This allows you to reuse and execute the same kind of testing operations
- Automated testing is conducted using software tools, so it works without tiring and fatigue unlike humans in manual testing
- It can easily increase productivity because it provides fast & accurate testing result
- Automated testing support various applications
- Testing coverage can be increased because of automation testing tool never forget to check even the smallest unit

Cons of Automated Testing:

- Without human element, it's difficult to get insight into visual aspects of your UI like colors, font, sizes, contrast or button sizes.
- The tools to run automation testing can be expensive, which may increase the cost of the testing project.
- Automation testing tool is not yet full proof. Every automation tool has their limitations which reduces the scope of automation.
- Debugging the test script is another major issue in the automated testing. Test maintenance is costly.

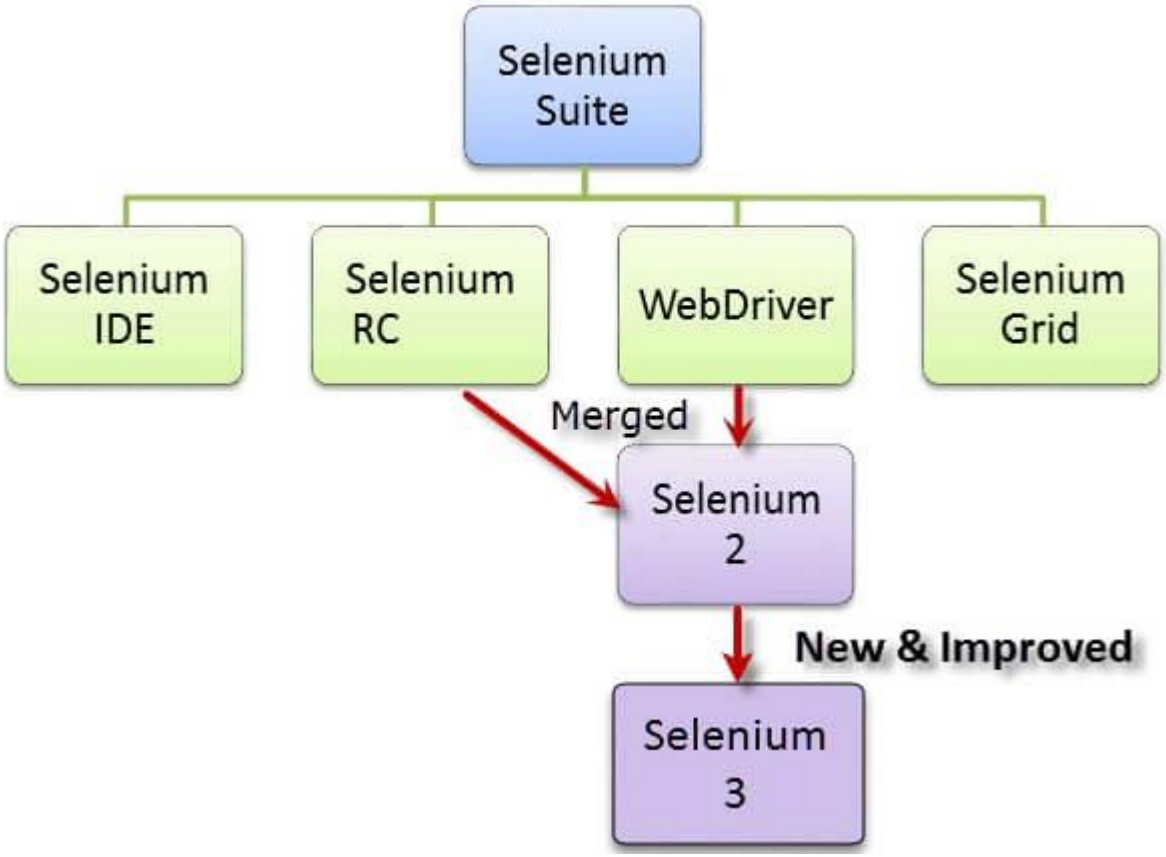
Automation Testing Tools

- Ranorex Studio
- Mabl
- Selenium
- Robot Framework
- Appium
- etc.

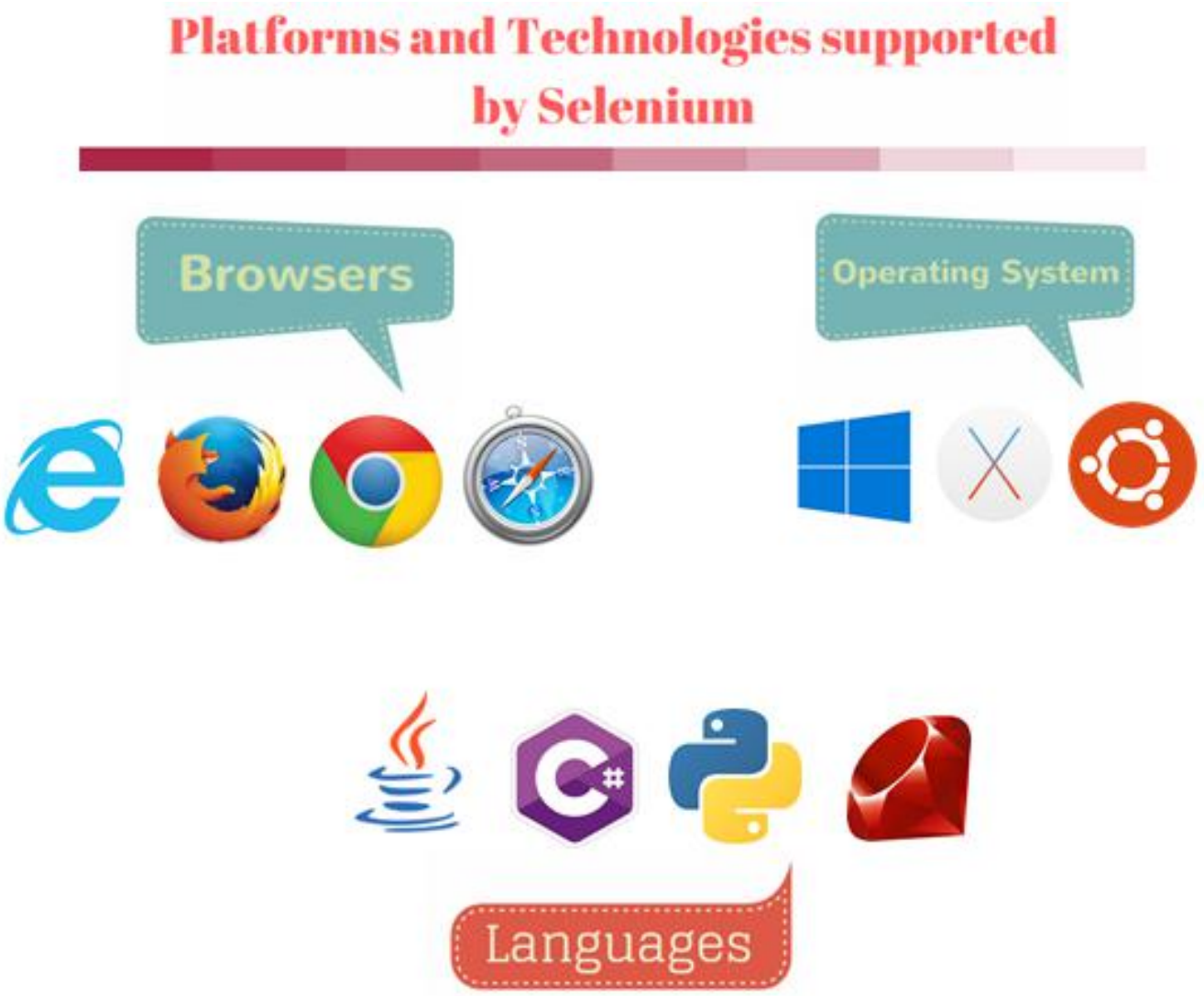
What is Selenium?

Selenium is a free (open source) automated testing suite for web applications across different browsers and platforms

What is Selenium?



What is Selenium?



What is Selenium?

Advantages :

- Free and Open Source
- Supports a wide variety of programming languages
- Support for cross browser testing

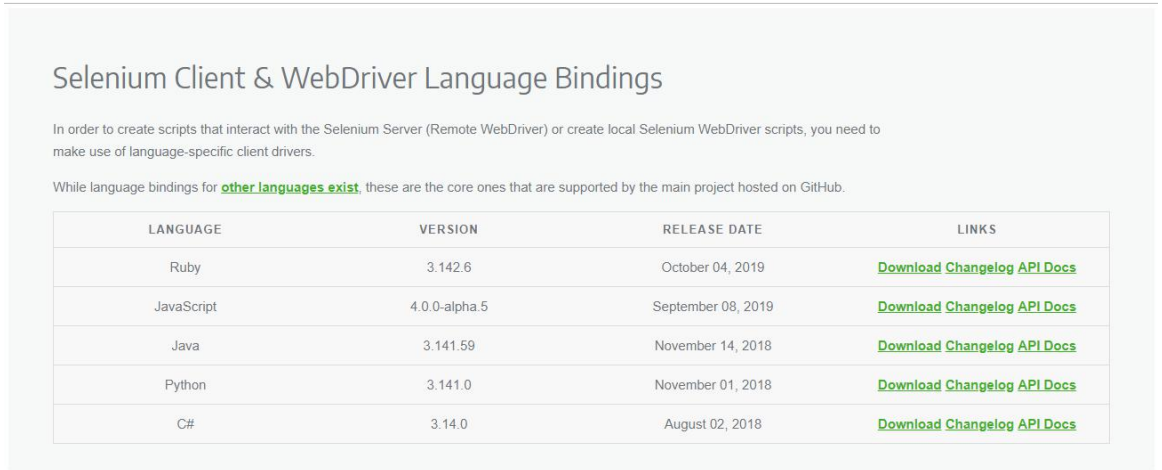
Disadvantages :

- Only for Web Based App
- Lack of professional customer support

Selenium Requirements

- Java v8
- Java IDE (Netbeans, Eclipse, InteliJ, etc.)
- Selenium Client
- WebDriver (ChromeDriver, GeckoDriver, etc.)

https://selenium.dev/downloads/



Selenium Client & WebDriver Language Bindings

In order to create scripts that interact with the Selenium Server (Remote WebDriver) or create local Selenium WebDriver scripts, you need to make use of language-specific client drivers.

While language bindings for other languages exist, these are the core ones that are supported by the main project hosted on GitHub.

LANGUAGE	VERSION	RELEASE DATE	LINKS
Ruby	3.142.6	October 04, 2019	Download Changelog API Docs
JavaScript	4.0.0-alpha.5	September 08, 2019	Download Changelog API Docs
Java	3.141.59	November 14, 2018	Download Changelog API Docs
Python	3.141.0	November 01, 2018	Download Changelog API Docs
C#	3.14.0	August 02, 2018	Download Changelog API Docs

Selenium First Script

```
public static void main(String[] args) {
    System.setProperty("webdriver.chrome.driver", "path to chromedriver.exe");
    WebDriver driver = new ChromeDriver();
    String baseUrl = "http://google.com";
    String expectedTitle = "Google";
    String actualTitle = "";
    driver.get(baseUrl);
    actualTitle = driver.getTitle();
    if (actualTitle.contentEquals(expectedTitle)){
        System.out.println("Test Passed!");
    } else {
        System.out.println("Test Failed");
    }
    driver.close();
}
```