20. Automation Code Review Guidelines

Last updated by | Warren Duffin | Jun 7, 2023 at 7:34 PM GMT+5:30

Objective

This page is to provide some help, guidance and tips to refer to when conducting a review of automation code, both scenarios and Java code.

This should help teams creating automation to mature and become more self-sufficient to perform code reviews for themselves.

Useful Links

Automation Principles and Rules: 02: CTMP Automation Principles and Rules
T-VaaS (tags and reporting): 08. Tags and Reporting

Guidelines

Below are some key items to be checking for...

Feature Files

Feature tag is present (e.g. @F12345) at the start of the file

Design tag is present (e.g. @Design_UC1234) after the feature tags at the start of the file

If no associated design ids (use cases) then @Design_NA must be used

Should have a feature description present

Scenarios
Check correct tags applied to each scenario

User story with version - @Usxxxx_Version_xx.xx
User Requirements - @UserReq_xxxx

Check scenario numbers added to each referenced requirement in qTest

Scenario - @Scenario_xxx.yyy.zzz. (the . at the end is important)

Release tag for new functionality - @Release_xx.xx
If a Scenario is updated for a Release then this should be updated to match the new Release. i.e. **There should be only one Release Tag per Scenario**

Relevant test grouping if applicable, e.g. @SmokeTest, @CoreConfig etc
@SmokeTest tag should be added to a small number of scenarios for basic checking of new functionality and should not be added to all scenarios

Common setup steps used by all scenarios should be moved to background steps at feature file level where possible

Should not be using repeated keywords as should be using **And after a Given, When or Then** keyword

Use of an identifier on data entry steps so can be referenced in later steps

When checking previously entered data use placeholders for the values

Data tables should be formatted correctly (aligned)

Java code

Check any method created isn't a duplicate of an existing method

Check if existing method could be updated rather than new 'similar' method created

UI element locators should use ID property wherever possible

if IDs don't exist they should be added to the application wherever possible

D.R.Y. - Don't Repeat Yourself (utilize or update existing methods / use existing constants - e.g. COMPANY_PERSONNEL_NO exists so don't create a new constant COMPANY_PERSONNEL_NUMBER when it is the same thing)

K.I.S.S. - Keep It Simple, Stupid (don't over complicate the solution)

Y.A.G.N.I. - You ain't Gonna Need It (don't add unnecessary/unused functionality)

Ensure the development coding standards and style are being followed

Coding Standards and Version Control

Java Programming Standards

Code Style
SonarQube

General

Read other reviewers comments which can help with your own understanding
Reply to others comments if you need to add anything extra or further explain a comment if it will help