**Testing and testability**

* **How do you review code?**

Below are the best practices that I feel for a code review :

1. Review few lines of code at a time - Take time to perform the proper code review by doing few lines of a code review at a time to have a efficient code in place

2.. Set goals and Capture measures and metrics - Before we start the review it is important to have the goals set as to why are we doing the code review . For example , to cut the defect percentage of defect induced in development. Also try to have the metrics in place like defect intensity, defect rate , Inspection rate

3. Look at the comments in the code or the development documentation or Code documentation to get an overall picture of what the code is doing

4. Use Checklist - Its quite obvious to do same mistakes while coding so its better to have a checklist on the common rules and practices that needs to be covered during code review

* **How do you enforce coding standards?**

Below are the coding standards that should be followed as per me when it comes to RobotFramework Test Automation scripting

Test Suite/ Script :

1. Test Script names should be less than 20 characters

2. Documentation should be updated with purpose of the script and pre-conditions if necessary

3. Proper keywords should be given for Suite Setup, Suite Teardown, Test Setup and Test Teardown.

4. Should not have too many tests (max 50) in one suite unless they are data-driven

Test Cases:

1. Test case Name should be in camel case

2. It should be easily readable and self explanatory

3. Proper keywords should be given for Suite Setup, Suite Teardown, Test Setup and Test Teardown.

4.Appropriate tags should be given for each case.

5. Hard coding of object name should be avoided.

6.Should contain many high level keywords instead of repeating steps often.

Resources:

1. Keep all resource files in single folder

2.All constants should be maintained in separate resource file

3. Keep separate resource files for application’s data

High Level Keyword / User Keyword:

1.It should be easily readable and self explanatory.

2. For easy readability, it should be in camel case

3.Can have space for better readability

4. Hard coding of object name should be avoided

5. Duplicate Functions should not be added

Variables :

1. Variable names should be less than 20 characters.

2. Variables should have meaningful words

3. For easy readability, it should be in camel case

4.Can have space. But try to restrict for minimum

Code standards in Python:

Below are some of the Key features that as a python developer or tester who is trying to write an effective Python code :

1. I follow PEP -8 style guide for coding ( import this ) which gives the output . The main factor is “Readability Counts” which means the other person should be able to understand to some extent by going through the code.

2. Indentation is important

3. Blank lines - In Python scripts, top-level function and classes are separated by two blank lines. Method definitions inside classes should be separated by one blank line

4. Imports - you should always import libraries at the start of your script.

5. Comments - Comments are used for in-code documentation in Python. They add to the understanding of the code

6. Module Level dunder Names - should be placed at the module main docstring and should be before all the import statements

7. Naming Conventions - Below are the naming conventions we need to use

Module - lowercase

Class - CapWords

Functions - lowercase

Methods - lowercase

Type variables - CapWords

Constants - UPPERCASE

Package - lowercase

* How do you plan what kind of approach you take for test automation - what libraries to use, how does it work in couple of years, how to make it easy to maintain, etc? What are the main points to consider?

Approach for a test automation is mainly based on the purpose and test coverage/functionality. If the testing is mainly focusing on the web automations then we need to import the libraries like SeleniumLibrary, RESTinstance, Django Library. If the testing is mainly focusing on testing the database connection and data analysis then we need to import Database Library ( Python or JAVA)

How does it work in couple of years ? - We might be needing to change the code a little bit based on the upgrades happening in the libraries

How to make it more easy to maintain - Try to have all the in built keywords from the libraries in a single resource file so that code changes can be easy and efficient

What are the main points to consider ?

1. Write a test plan for the test coverage with proper understanding of the Requirements

2. Mention the written steps on how are the functionality tests going to be performed

3. Look for what regression tests that need to be performed and how to build the suites

4. Data driven tests need to be covered

* **Code testability, how do you enforce it?**

One main key factor that I feel it helps in the code testability is even the smallest component of the code are independently verifiable. In order to do this each component must have its dependencies injected to it. I feel this can be highly efficient and effective way of building the code. It makes last-minute changes, the kind of changes that are often very important on a project, trivial and essentially cost-free.

* **How do you make sure that the product is testable?**

To be testable, a requirement must be clear, measurable, and complete, without any ambiguity. Requirement provides details that lead to the creation of tests . Testable requirements should not include Text that is irrelevant, a description of the problem rather than the function that solves it, Implementation details, Ambiguity