Assertion Design Pattern

* Performing assertions on your test scripts is one of the most important aspects of the test-automated design.
* Assertions are useful in ensuring whether the actual outcome of your scripts matches the expected outcome in your application.
* One of the best practices in employing an assertion pattern in your test automation framework is to ensure, that in case your automation scripts fail due to a bug in SUT or a script issue, the run time messages displayed in the console/screen are readable, understandable and transparent to someone executing automated scripts.
* This will help them debug and fix the errors easily, quickly and w/o needing to find the granular details about the exact failure in scripts.
* Let’s see an example below:

Example:

We have below scenario written in Gherkin language i.e. in plain text that is easy to understand by technical and non-technical staff.

Scenario: Verify I should not see a success message if I add a product to my cart, but do not place the order.

Given I log in to a Flipkart application

When I add a product to my cart

But I do not place an order

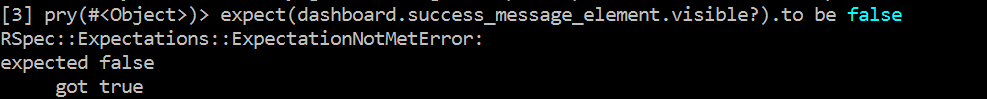
Then I should not be able to see the success message successfully

What I need to assert here is I should not be able to see a success message if am adding my product to the cart but not placing an order on Flipkart.

I have two Quality Assurance Engineers writing automated scripts for this scenario. Lets’ name them A and B.

The scripts

So once ‘A’ completes his scripts and he runs the automation script, below is the result:

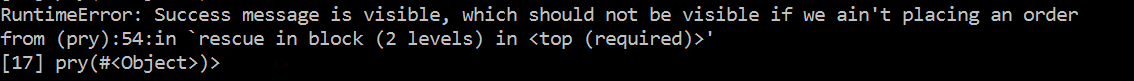


What exactly ‘expected false got true’ means? We are verifying a success message text….aren’t we?

The information we pass on the person looking into the error is just a false or a true statement.

Much better… Isn’t it?

Lets’ see what ‘B’ has in store for us:



That is Cool…This is what you call an elaborated failure message. Reading this, I do not have to think what does A mean by expected false and got true error message, whereas ‘B’ clearly defines what and where exactly the failure is occurring.

Bottom line is …always keep a self-explanatory failure messages that are easier to read, understand and do not make much fuss in itself.

Cheers!