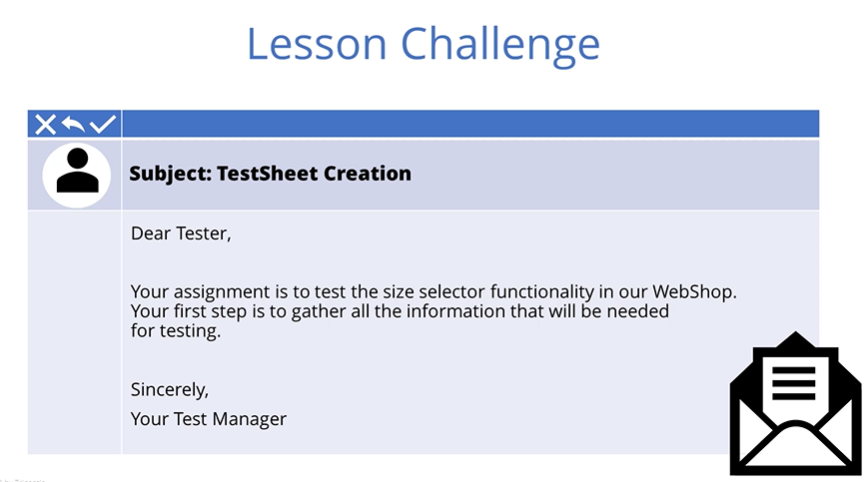
TOSCA AS2 →

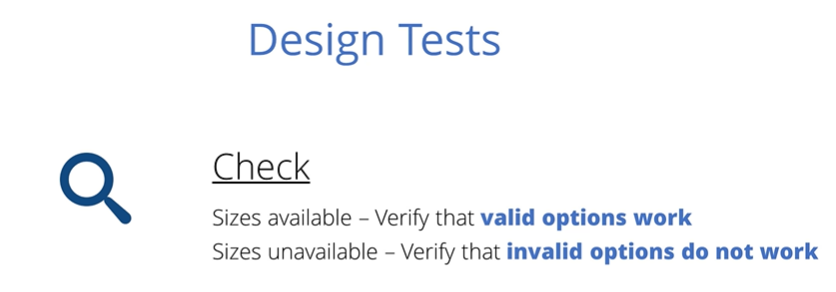
* We have created libraries and reusable test step blocks so that whatever changes happen can be made at one place and reflected everywhere
* But what if the changes take place at verification level or where we cannot use libraries
* For that we need to use different concepts → tcd, instances, attributes etc ………...

# Lesson 1 - TestSheet Creation: →



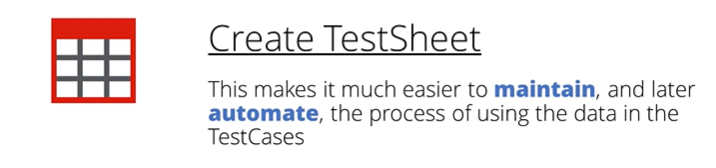
SUT functionality to test → add shirt to shopping cart but for that we need to select the size

For this we need to design test to check if all the sizes are available



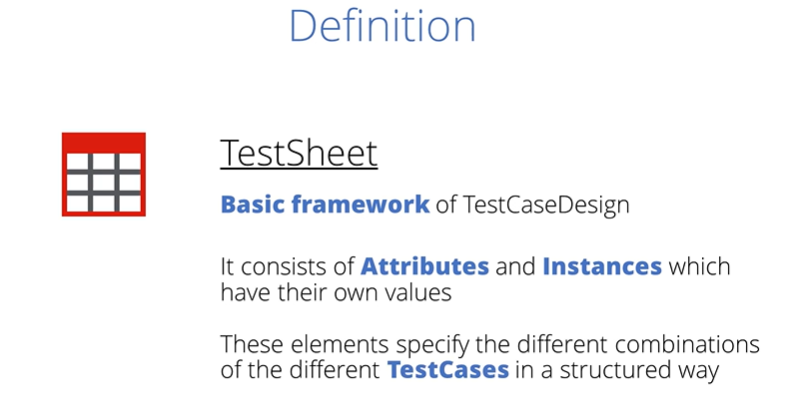
Invalid → gives error message

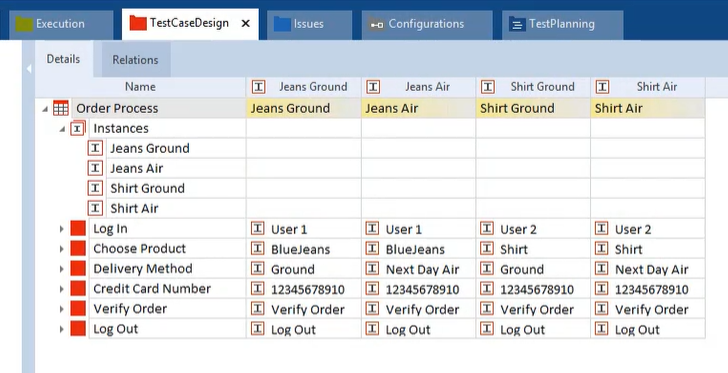
Order size not available → check what is result

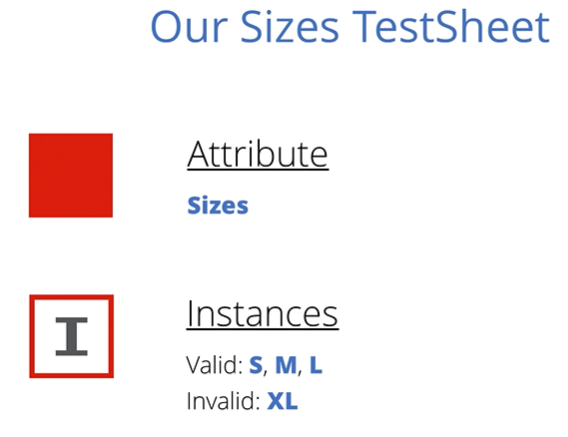


Topics →

1. Understanding concept to TestSheet
2. Identify the difference between attributes and instances
3. Create TestSheet



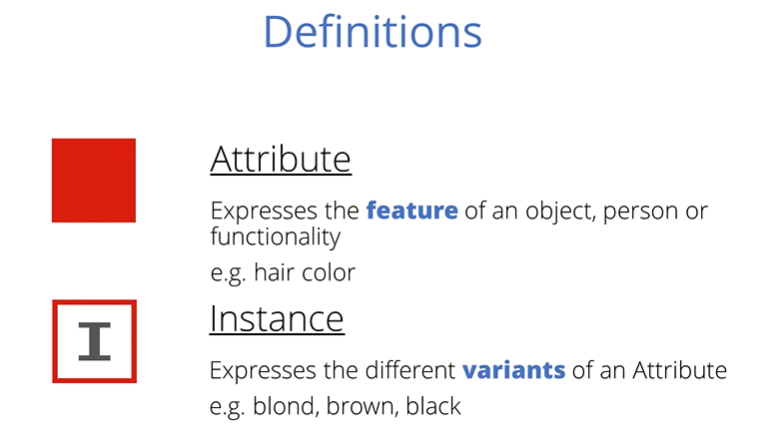


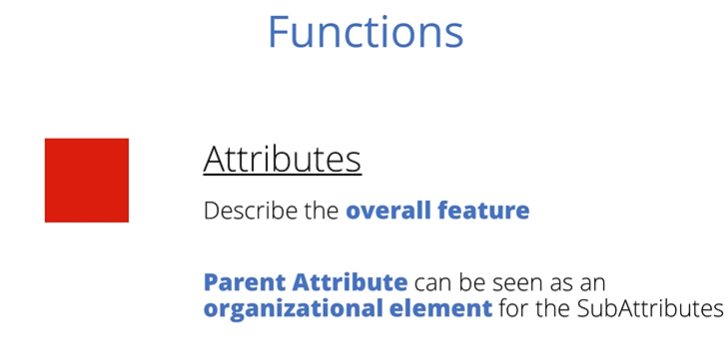


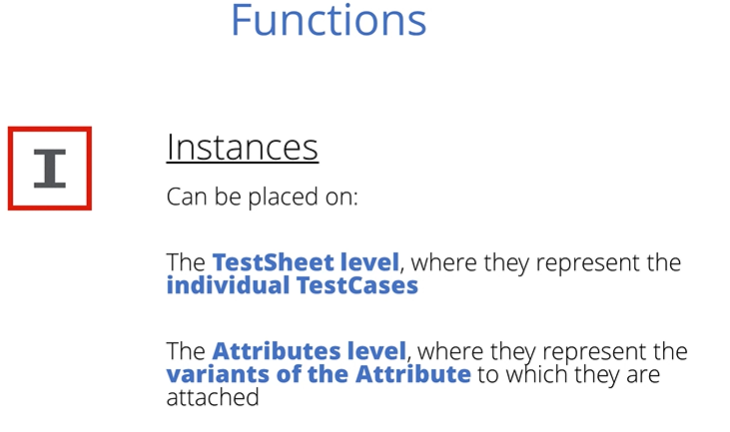
To create TESTSHEET →

1. RC on TCD folder > Select create TestSheet (need to be manually completed by us this work is done by test design specialist with the values given by business department)

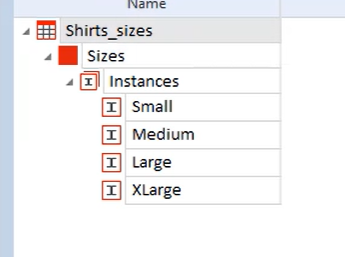
→ Empty TestSheet is not useful > Add attributes to TestSheet and then add instances to the Attributes



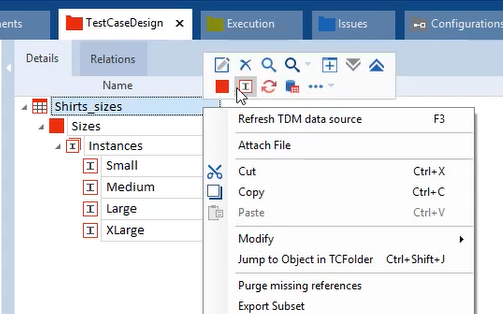


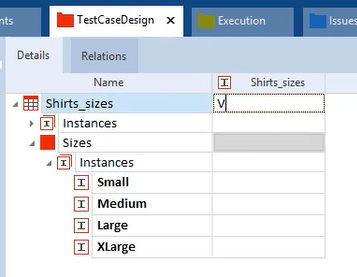


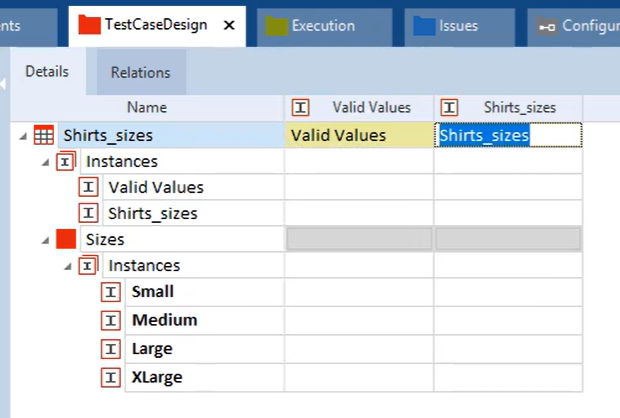
Add instances to Attribute level →



1. Add instances on the TestSheet level to identify the actual testcase that we need to create later



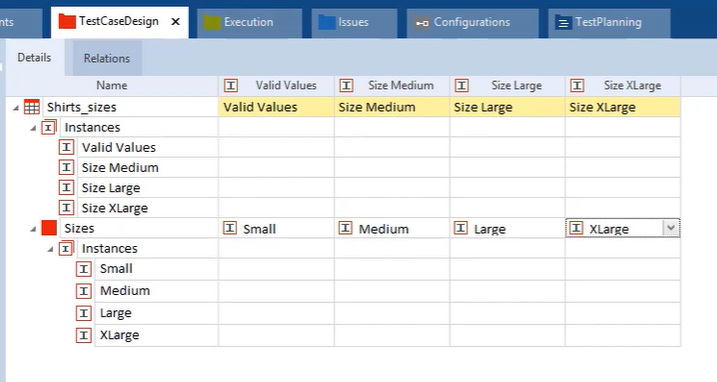




File these instances with the data to indicate the values to be entire at various stages of the test

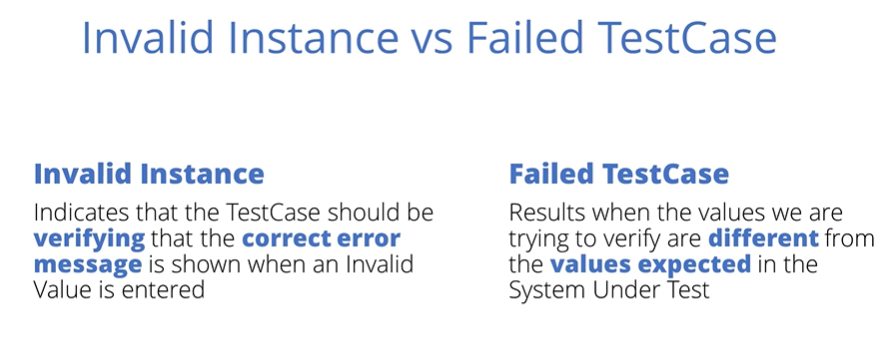
Instance values can be entered manually on the instance it self

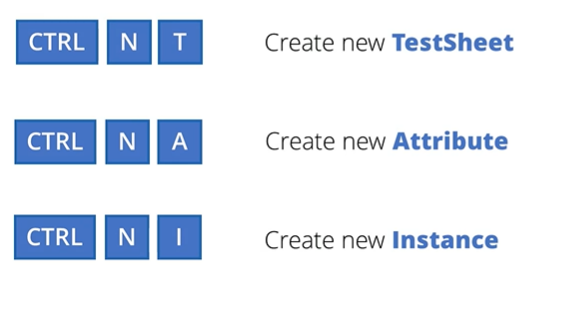
→ But on attribute level, possible values can select by drop down menu



Valid values column is considered as the most important, most frequently used or easiest path with the least dependencies. (happy path)

This instance column will be used to create test cases





# Lesson 2 - Templates: →

We could create individual test cases to cover each possibility in the testsheet

Even if we use libraries there will be a down side bcz of repetitive process

Also, each test cases require Individual maintenance

ex. When developer changes the area of SUT not suitable for a reusable test step block

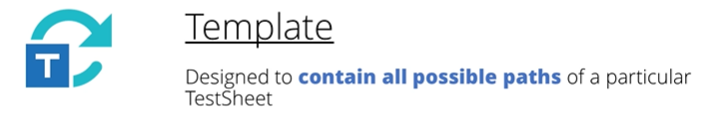
To avoid such problems, we will create a Testcase Template which is used to automatically create the number of test cases using test data contained to the TestSheet

Majority of cars are designed same way and each car differs only for certain attributes

Blueprint → original car design acts as a blueprint & it can be used for different variations



Ex. different color and transmission

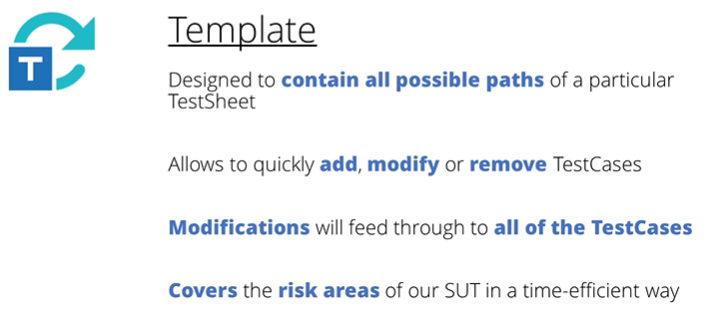


By linking data from the testsheet with testcase, we can quickly create number of test cases

Allows to add, modify or remove test cases when there are changes to our SUIT

When the template is modified to reflect changes, these modifications will feed through to all of the test cases.

That’s why templates are great help when trying to cover all the risk areas of our SUT in a time efficient way



STEPS

1. Resolve all the references to the test steps from library
2. Rc on testcase > convert to template

→ now testcase is a template

→ we can convert the template back to testcase (rc > convert to testcase)

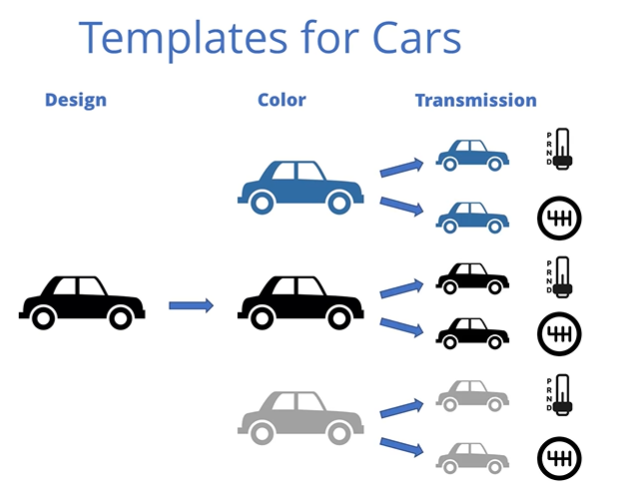
1. 2a login process → create a folder and add testcase with login process teststeps
2. 2b resolve reference → resolve all the references
3. 2c create template → convert testcase to template

# Lesson 3 - How to use Templates: →

Template cannot run as a test itself, template lacks some of the specific info needed for a test to successfully complete

Template will create a series of slightly different test cases that use the template test cases as a starting point but to do this is requires information





We have all this info in our testsheet so now just link the testsheet to the testcase

Topics: →

1. Link a testsheet to template
2. Link the testsheetvalues to teststepsvalues in the template
3. Check template and use the jump to schema function
4. Instantiate the template

Steps →

1. Link

→ select testsheet > drag and drop it to the template

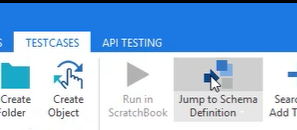
→ link testsheet to testcase as soon as it is created to void confusion as which testcase related to which testsheet

→ tosca allows to link testsheet to more than one testcase but it is not a best practice as it makes it hard to maintain and keep an overview of several templates

→ if same data is being required for multiple test cases recommended use classes

1. Once testsheet is linked to testcase

→ under the testcase section on ribbon > jump to schema definition option has arrived.



→ helps to check which testsheet is linked to which template

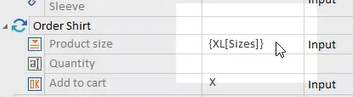
Another way (select template)

1. Ctrl + j
2. Rc > select jump to schema definition
3. Under test cases section on ribbon
4. Check the name of testsheet link to template in property tab
5. Link values from the testsheet to the relevant test step values in the template.

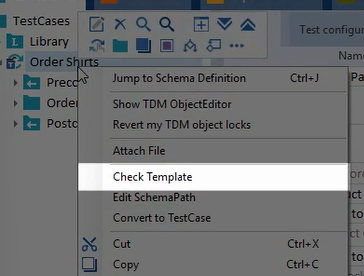
Without this we would not be able to tell tosca which parts of the testsheet are used for the different teststeps .

→ drag and drop on to the relevant teststep from the testsheet

* Drag and drop the attribute sizes from testsheet on to the teststep in the testcase > which is the point at the testcase where size will be chosen
* This will automatically create an {XL[Sizes]} reference



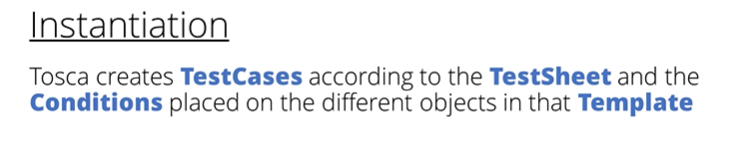
* XL reference tells that values for the teststep will be taken from the testsheet attribute
* XL reference itself contains the path for tosca to follow in order to find the value from the testsheet
* Once linking is done , search for the error in the XL reference
* We can do this by RC on template and select Check Template



* This will run through the links to the testsheets and will validate their existence
* It will not check for the functionality error or the formulas
* Check template function will highlight the test step values containing error, not the XL reference itself

Now, our workstate should be set to complete

1. Use Workstate to create test cases > through INSTANTIATION



1. This process will create 4 test cases

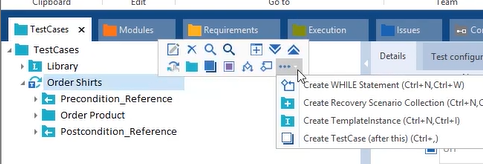
→ one testcase for each for each of the four test case instances in our testsheet

**Ways : →**

1. To instantiate, select the template and then click on the instantiate button in the ribbon in the testcase tab



1. RC template > select create template instance



1. Select template > ctrl + N + I
2. The template remains the same but below a new folder called template instance is created.

→ after expanding the folder > all test cases that were created based on our template with the correct values for the test focus.

→ it is important to now check the test cases

Instances → test cases

→ values are used from the testsheet instead of references

# **Lesson 4 - Modifications to templates : →**

SUT is developed using Agile Methodology

In this sprint, the developers introduced a new feature to add the cart button



→ after adding the number of shirts and clicking the add to cart button then a message appears

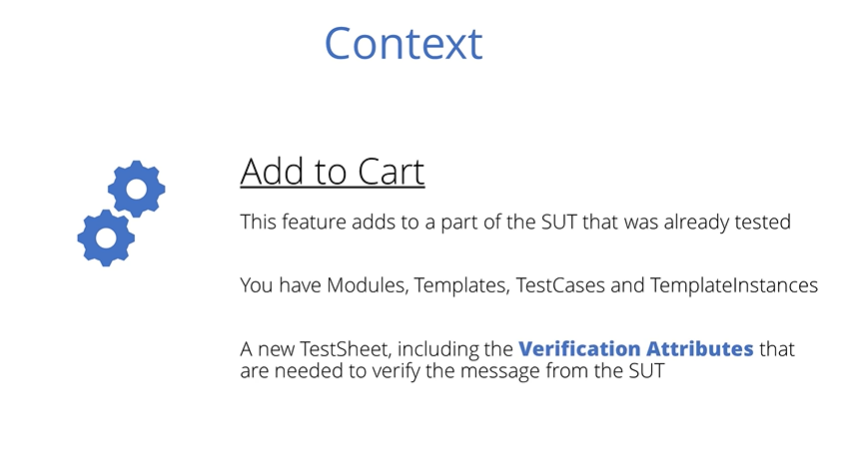
→ shows user if they interred valid quantity or an invalid one

Our task is to test new functionality is working correct or not

1. If the quantity entered is a positive whole number bigger than zero - valid entry
2. Negative whole number - invalid entry

invalid → error message with red background

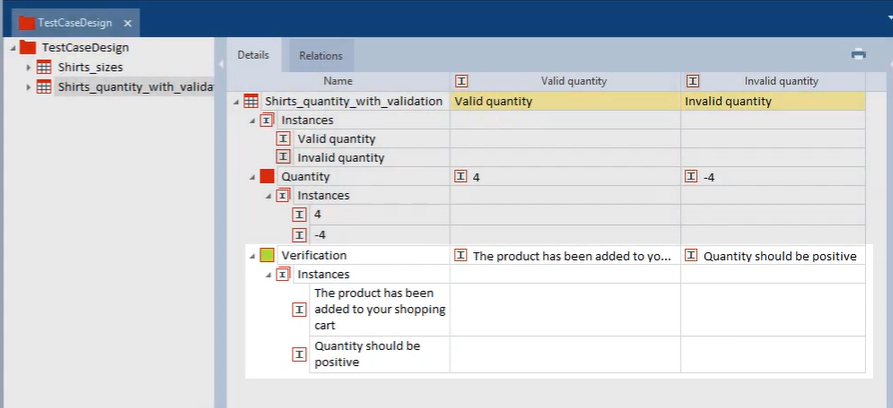
Valid → valid message with green background

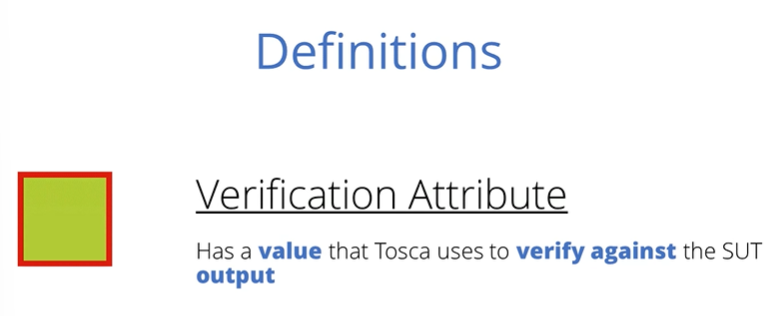


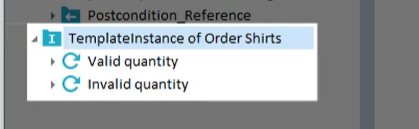
→ You received a new test sheet from the test design specialist including the verification attributes needed to verify the message from the SUT

Topics →

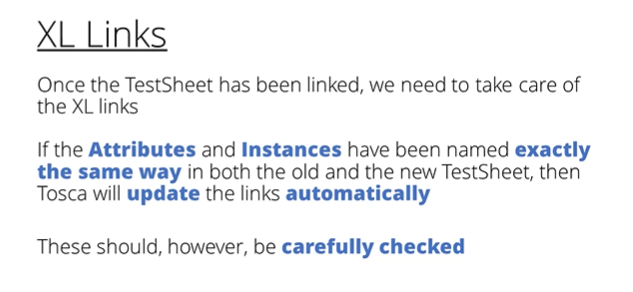
1. Define what a verifications attributes is
2. Modify an existing template
3. Reinstantiate a modified template



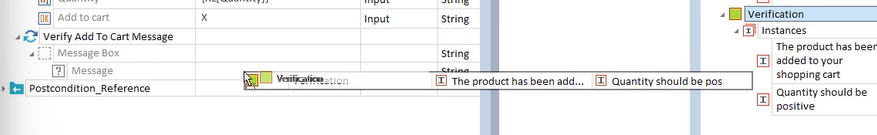


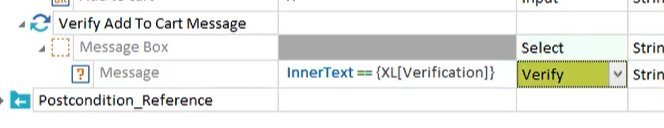


1. Resolve any references before proceeding to modify it , prevents any changes being made, made to our library
2. All TestSteps that requires modification need to be identified
3. If any of them have references to reusable teststep blocks, these references will need to be resolved
4. Add new test steps to verify the message
5. Link testsheet to the template > automatically old linked testsheet is removed
6. Once testsheet is linked > take care of XL Links

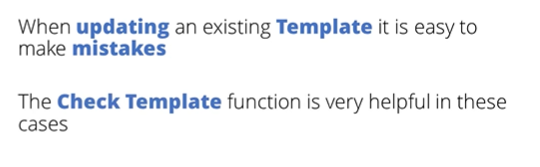


1. Drag and drop the new links to the teststep manually





1. Check for the errors.

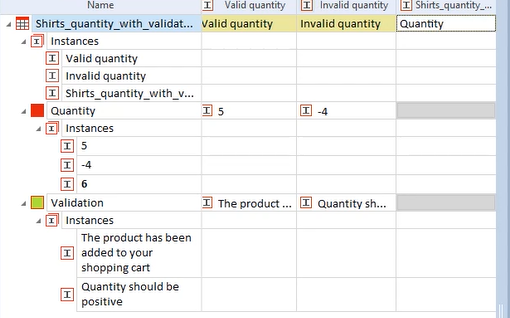


1. Delete the old template instances folder previously created

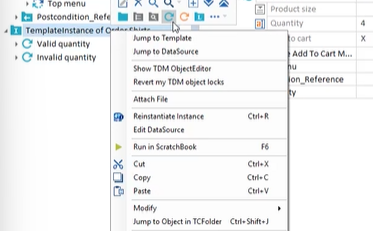
Rc on testcase and click on create template instance

If test design specialist had amended the existing test sheet instead of providing a new one?

→ make changes in the existing testsheet

****

**→ reinstantiate**

****



# **Lesson 5 - Conditions : →**

With other, slightly more complex templates, if we were to run testcases instances at this stage, most of them are likely to fail.

Bcz , tc contains contradictory validation steps for the same values.

Some validations can only be verified if specific conditions are fulfilled

