TEST CLASSES

Test Classes are the automated test scripts used to maintain the quality of the code (business logic).

While deploying the code to the Production, we have to maintain minimum 75% of the Code Coverage which can be implemented by using "Test Classes".

Test Classes should be used to test the below code components.

1. Apex classes.

2. Triggers

3. Batch Classes

4. Future Methods

5. Queueable Classes

7. Schedule Classes

8. Web Services and API's.

We can verify the Code Coverage of the Organization as below.

Click on "Setup" menu.

1. Search for the option "Apex Classes".
2. Click on the link "Estimate the Organization's Code Coverage".
3. View the Code Coverage.

(OR)

Click on "Setup ---> Developer Console".

1. Go to the Developer Console Editor.
2. Expand the Tab Bar and Click on "Tests" tab.
3. View the Code Coverage in right panel.

While preparing the test Classes follow the below Steps.

Step 1: Prepare a test class which should be pre-fixed with the annotation "@isTest()'.

Syntax: @isTest()

```
[Private/Public] Class <Class Name>
    {
        // Write the Testing Code
    }
```

Step 2: Prepare the Test Method inside the Test Class.

Syntax: @isTest()

```
[Private/Public] Class <ClassName>
        {
                [Private / Public] testmethod static void <methodName>()
        {
                        // Write the Testing Code
        }
        }
```

Step 3: Invoke the Test Class.

Case i: By using "Run Test" button.

1. Open the Test Class, which has been prepared.
2. Click on "Run Test" button.

Case ii: By using "Apex Test Execution".

1. Click on "Setup" menu.
2. Search for the option "Apex Test Execution" from Quick Find box.
3. Select the "Test Classes" to be get Tested by using "Select Test" button.
4. Click on "Run Button".

Case iii: By using "Run All Test" button.

1. Click on "Setup" menu.
2. Search for the option "Apex Classes" in Quick Find box.
3. Click on "Run All Test" button.

<u>Example 1</u>

Business Logic Class:

```
public class MathOperationsHandler
{
        Public static void Addition(Integer fNumber, Integer sNumber)
   {
      System.debug('Addition Result is...: '+ (fNumber + sNumber));
   }

   Public static void Subtraction(Integer fNumber, Integer sNumber)
```

```
    {
        System.debug('Subtraction Result is...: '+ (fNumber - sNumber));
    }

    Public static void Multiply(Integer fNumber, Integer sNumber, Integer tNumber)
    {
        System.debug('Multiplication Result is...: '+ (fNumber * sNumber * tNumber));
    }

    Public static void Division(Integer fNumber, Integer sNumber)
    {
        System.debug('Division Result is...: '+ (fNumber / sNumber));
    }
}
```

Test Class:

```
@isTest()
Private class MathOperationsHandlerTest
{
        Public testmethod static void TestMathOperations()
    {
        MathOperationsHandler.Addition(34500, 7800);
        MathOperationsHandler.Subtraction(12000, 4922);
        MathOperationsHandler.Multiply(129, 231, 12);
        MathOperationsHandler.Division(45000, 1247);
    }
}
```

Example 2

Batch Class

```
Global class CalculateTotalRevenueBatch implements Database.Batchable<SObject>,
Database.Stateful
{
    Global Decimal totalAnnualRevenue = 0.0;
```

```
        Global Database.QueryLocator Start(Database.BatchableContext bContext)
    {
        String accountsQuery = 'Select id, name, rating, industry, annualrevenue from Account
Where annualRevenue != Null';


        return Database.getQueryLocator(accountsQuery);
    }
    Global void Execute(Database.BatchableContext bContext, List<SObject> accountRecords)
    {
        if(! accountRecords.isEmpty())
        {
            for(SObject obj : accountRecords)
            {
                Account accRecord = (Account) obj;


                totalAnnualRevenue += accRecord.AnnualRevenue;
            }
        }
    }


    Global void Finish(Database.BatchableContext bContext)
    {
        System.debug('Batch Job Id is.....: '+ bContext.getJobId());
        AsyncApexJob jobDetails = [ Select id, status, totalJobItems, jobItemsProcessed,
                                    numberOFErrors, CreatedBy.Email
                                    from AsyncApexJob     Where id =: bContext.getJobId()];


        MessagingHelperUtility.SendBatchJobStatusNotificationEmail(jobDetails,
'CalculateTotalRevenueBatch', totalAnnualRevenue);
    }
}
```

MessagingHelperUtility class

```
public class MessagingHelperUtility
{
        Public static void SendBatchJobStatusNotificationEmail(AsyncApexJob jobDetails,
String jobName, Decimal totalRevenue)
   {
      if(jobDetails.Id != Null)
      {
          Messaging.SingleEmailMessage sEmail = new Messaging.SingleEmailMessage();


  String[] toAddress = new String[]{jobDetails.CreatedBy.Email, 'srinivaskon.1986@gmail.com'};
   sEmail.setToAddresses(toAddress);
  sEmail.setSenderDisplayName('DELL Customer Support Center.');
  sEmail.setReplyTo('support@dell.com');
    String emailSubject = 'Weekly Customers Total Annual Revenue Job Status : '+jobName+ '
('+ jobDetails.Id+ ').';
    sEmail.setSubject(emailSubject);
    String emailContent = 'Dear Customer Support Team, <br/><br/> '+
     We are pleased to Inform you, that Weekly Customers Total Annual Revenue Batch Job has
been Processed Successfully. <br/><br/>'+
                    'Here are the Batch Job Results...: <br/><br/>'+
                    'Batch Job Id is.....: '+ jobDetails.Id+
                    '<br/>Batch Job Name is.....: '+jobName +
                    '<br/>Batch Job Status is.....: '+ jobDetails.Status+
                    '<br/>Total Number Of Batches Available...: '+ jobDetails.TotalJobItems+
                    '<br/>Number of Batches Processed.....: '+ jobDetails.JobItemsProcessed+
                    '<br/>Number of Batches Failed.....: '+ jobDetails.NumberOfErrors+
                    '<br/>Batch Job Owner Email Address.....: '+ jobDetails.CreatedBy.Email+
                    '<br/>Total Annual Revenue Value is.....: '+ totalRevenue +
                    '<br/><br/> Please Contact on the below address, if any queries. <br/><br/>'+
                    '***<i>This is a System-Generated Email. Please Do Not Reply.</i><br/><br/>'+
```

```
                'Thanks & Regards, <br/>Customer Support Center, <br/>DELL Inc.';
                sEmail.setHtmlBody(emailContent);
        Messaging.SendEmailResult[] results = Messaging.sendEmail(new
Messaging.SingleEmailMessage[] {sEmail});
    }
  }
}
```

Test Class

```
@isTest()
public class CalculateTotalRevenueBatchTest
{
        Public testmethod static void TestRevenueBatch()
  {
    // Insert the Account Records
       CreateAccountRecords();
    // Invoke the Batch Class
    Test.startTest();
       CalculateTotalRevenueBatch revenueBatch = new CalculateTotalRevenueBatch();
       Database.executeBatch(revenueBatch);
    Test.stopTest();
  }
  Public static void CreateAccountRecords()
  {
    List<Account> lstAccounts = new List<Account>();
       for(integer counter =1; counter <= 50; counter++)
        {
           Account acc = new Account();
              acc.Name = 'Tata Motors Inc.';
              acc.Rating = 'Hot';
              acc.Industry = 'Manufacturing';
              acc.AnnualRevenue = 25000000;
              acc.Phone = '9900445577';
```

```
            acc.Fax = '9900004455';
            acc.Ownership = 'Private';
            acc.Type = 'Customer - Direct';
            acc.Website = 'www.tatamotors.com';
            acc.CustomerPriority__C = 'High';
            acc.Active__C = 'Yes';
            acc.BillingCity = 'Hyderabad';
            acc.BillingState = 'Telangana';
        lstAccounts.Add(acc);
    }
    if(! lstAccounts.isEmpty())
        Insert lstAccounts;
  }
}
```

Note: Governor limits are reset when the Test.startTest() appears and the code between Test.startTest() and Test.stopTest() executes in fresh governor limits. Also when the test.stoptest() appears, the context will be moved back to the original code.

Example 3:

Trigger Code

```
trigger AutoLeadConvertTrigger on Lead (After Update)
{
  if(Trigger.isAfter && Trigger.isUpdate)
  {
    AutoLeadConvertTriggerHandler.AfterUpdate(Trigger.New);
  }
}
```

Handler Class

```
public class AutoLeadConvertTriggerHandler
{
    Public static void AfterUpdate(List<Lead> lstLeads)
  {
```

```
        LeadStatus lStatus = [Select id, MasterLabel, isConverted  from LeadStatus
                                                    Where isConverted = true];

      List<Database.LeadConvert> leadRecordsToConvert = new List<Database.LeadConvert>();

      for(Lead ldRecord : lstLeads)
      {
         if(ldrecord.IsConverted == False && ldRecord.Status == 'Closed - Converted')
         {
            // Write the Code to Convert the Lead
            Database.LeadConvert lConvert = new Database.LeadConvert();
                lConvert.setLeadId(ldrecord.Id);
                lConvert.setDoNotCreateOpportunity(false);
                lConvert.setSendNotificationEmail(true);
                lConvert.setConvertedStatus(lStatus.MasterLabel);
            // Add the record to Collection
            leadRecordsToConvert.Add(lConvert);
         }
      }
      // Convert the Lead Records as Customers
Database.LeadConvertResult[] results = Database.convertLead(leadRecordsToConvert, false);
   }
}
```

Test Class

```
@isTest()
public class AutoLeadConvertTriggerTest
{
        Public testmethod static void TestLeadConversion()
   {
      Lead ldRecord = new Lead();
        ldRecord.FirstName = 'Praveen';
```

```
            ldrecord.LastName = 'Kumar';
            ldRecord.Rating = 'Hot';
            ldRecord.Industry = 'Finance';
            ldRecord.AnnualRevenue = 2300000;
            ldRecord.Company = 'Honeywell Technologies Inc.';
            ldRecord.Status = 'Open - Not Contacted';
            ldRecord.leadSource = 'Phone';
            ldRecord.Phone = '9900444477';
            ldRecord.Fax = '900333333';
            ldRecord.Email = 'praveen@gmail.com';
            ldRecord.Street = '#401, Sai Residency';
            ldRecord.City = 'Hyderabad';
            ldrecord.State = 'Telangana';
            ldRecord.country = 'India';
            ldRecord.title = 'Sales Manager';
        Insert ldRecord;
        if(ldrecord.Id != Null)
        {
            ldRecord.Status = 'Closed - Converted';
            Update ldRecord;
        }
    }
}
```

Example 4

Class Code

```
public class FutureMethodHelper
{
  Public static void DoDMLOperations()
    {
        // De-Activate the User --- Setup Object.
```

```
    User userToDeActivate = [Select id, firstname, lastname, username, isActive
                                                    from User
                                    Where userName = 'manageruser@dl.com and isActive = true];

    if(userToDeActivate.id != Null)
      {
          userToDeActivate.IsActive = false;
          Update userToDeActivate;
           InsertAccountRecord();
      }
}
@future()
Public static void InsertAccountRecord()
{
   // Create a New Account Record --- Non-Setup Object.
    Account acc = new Account();
      acc.Name = 'Krishna Teja Prasad';
          acc.Rating = 'Hot';
          acc.Industry = 'Manufacturing';
          acc.AnnualRevenue = 2900000;
          acc.Phone = '9900334455';
          acc.Fax = '8899445566';
          acc.Website = 'www.gmail.com';
          acc.Type = 'Installation Partner';
          acc.Ownership = 'Public';
          acc.BillingCity = 'Hyderabad';
          acc.BillingState = 'Telangana';
          acc.BillingCountry = 'India';
          acc.BillingStreet = '#306, Srinivasa Apartments, Kukatpally.';
          acc.BillingPostalCode = '500021';
          acc.CustomerPriority__C = 'High';
```

```
            acc.Active__C = 'Yes';
        Insert acc;
         If (acc.Id != Null)
            System.debug('Account Record Inserted Successfully. Account Id is...: '+ acc.Id);
    }
}
```

Test Class Code

```
@isTest
public class FutureMethodHelperTest
{
  Public static testmethod void TestFutureMethods()
    {
       FutureMethodHelper.DoDMLOperations();
    }
}
```