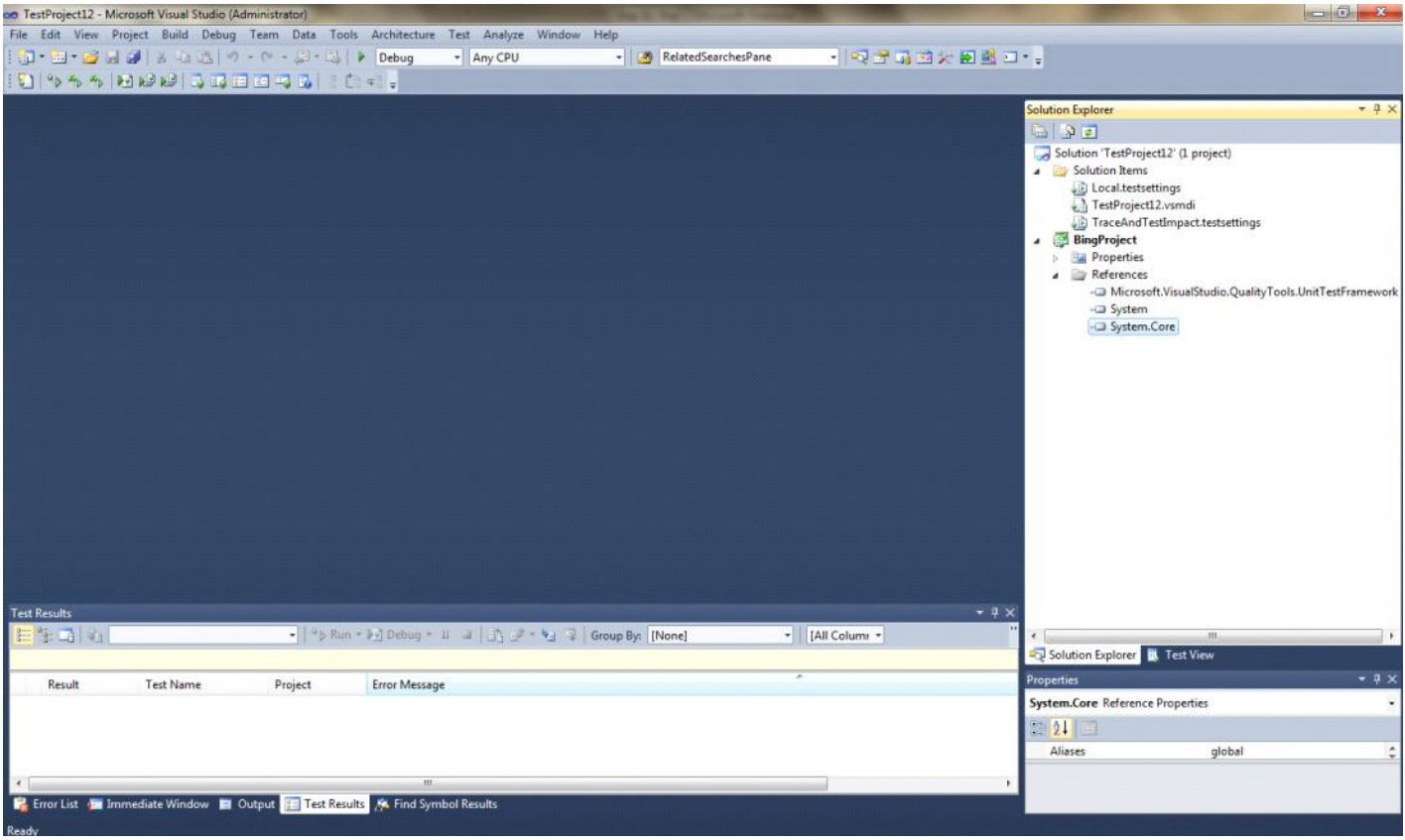
Framework - Step By Step

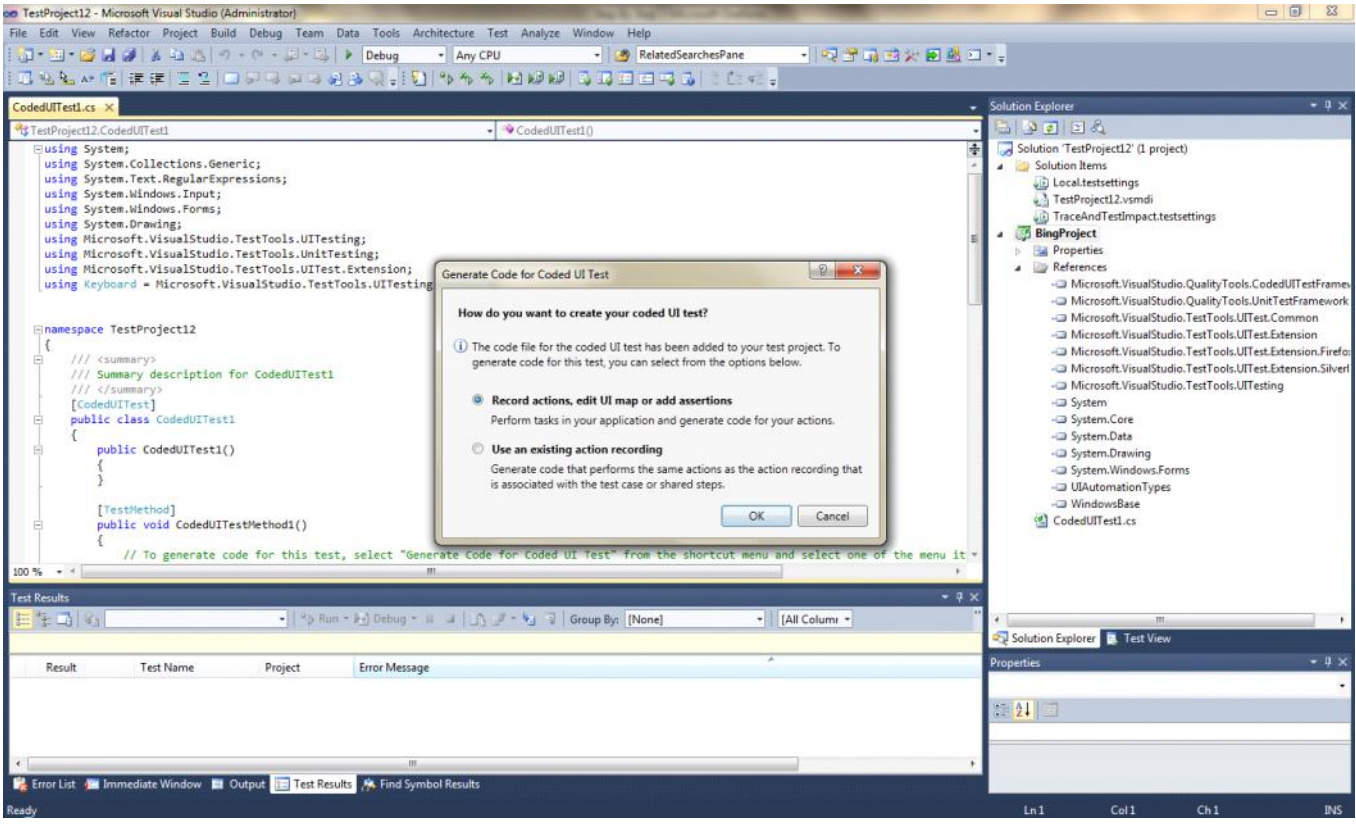
Monday, January 10, 2011
3:57 PM

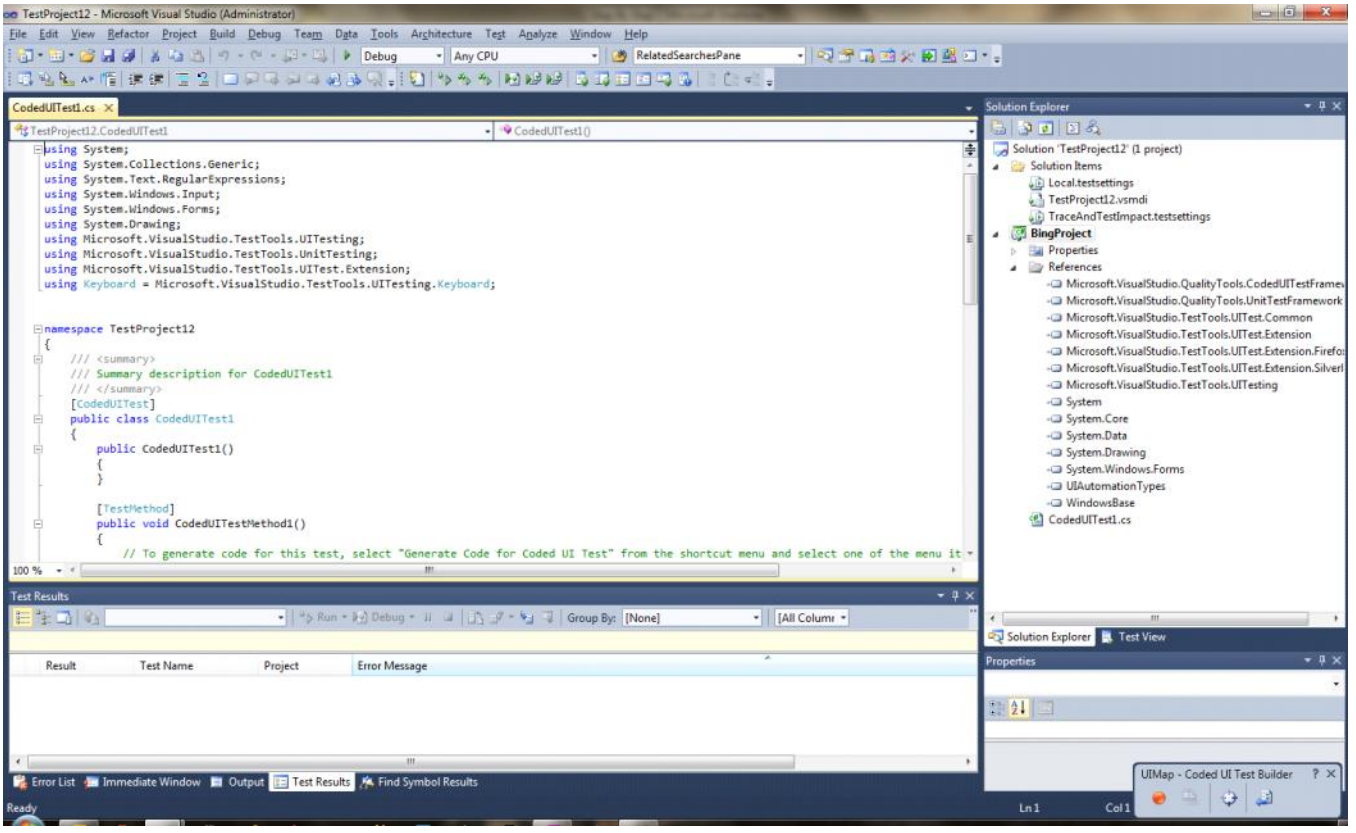
- This Document will provide Guidance to design a Framework on top of CodedUI for implementing and managing your Test Automation Suite
- This Framework was designed with main focus on Usability, Reusability, Maintenance, Management, Platform Independence etc
- Also, added some sample end to end flows for reference

1. **Create a New Test Project**

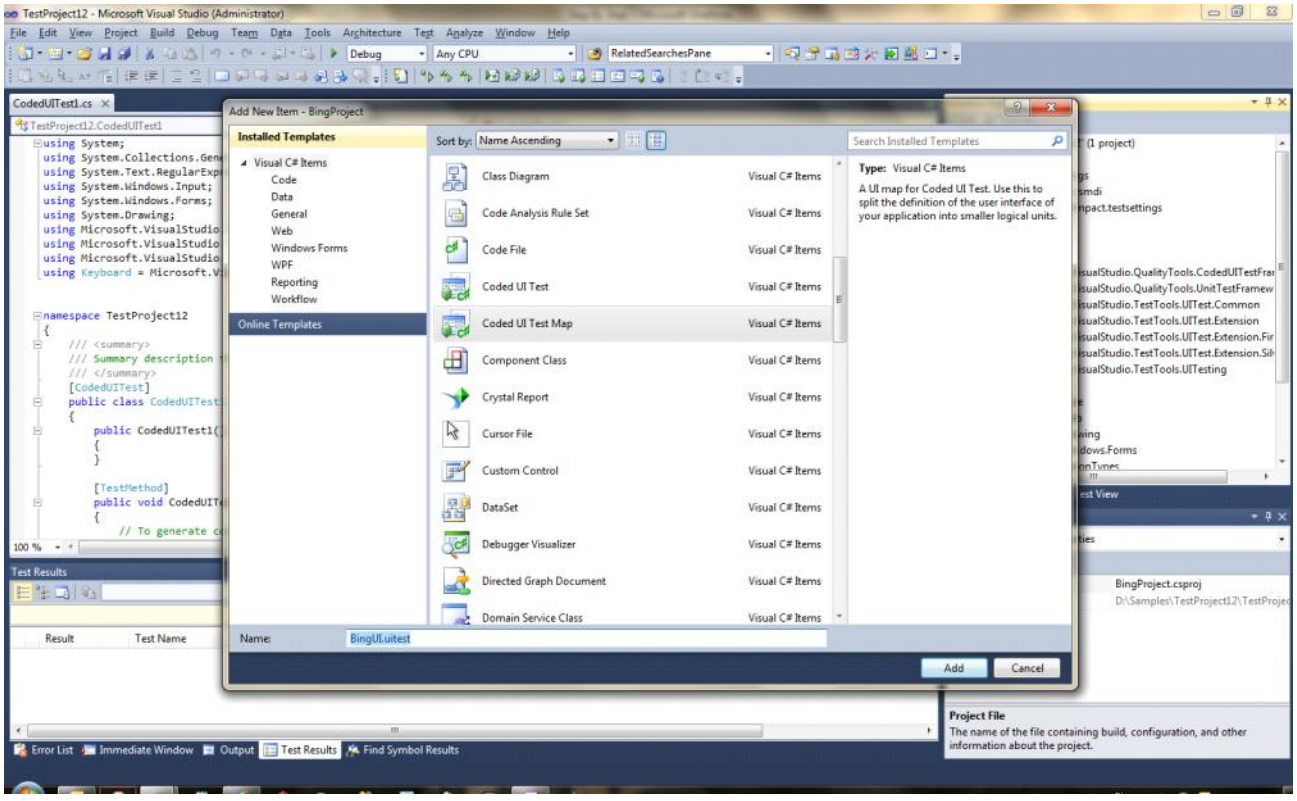


2. **Add a New Coded UI Test File**
   a. Select Record actions, edit UI map or add assertions and Click on OK button

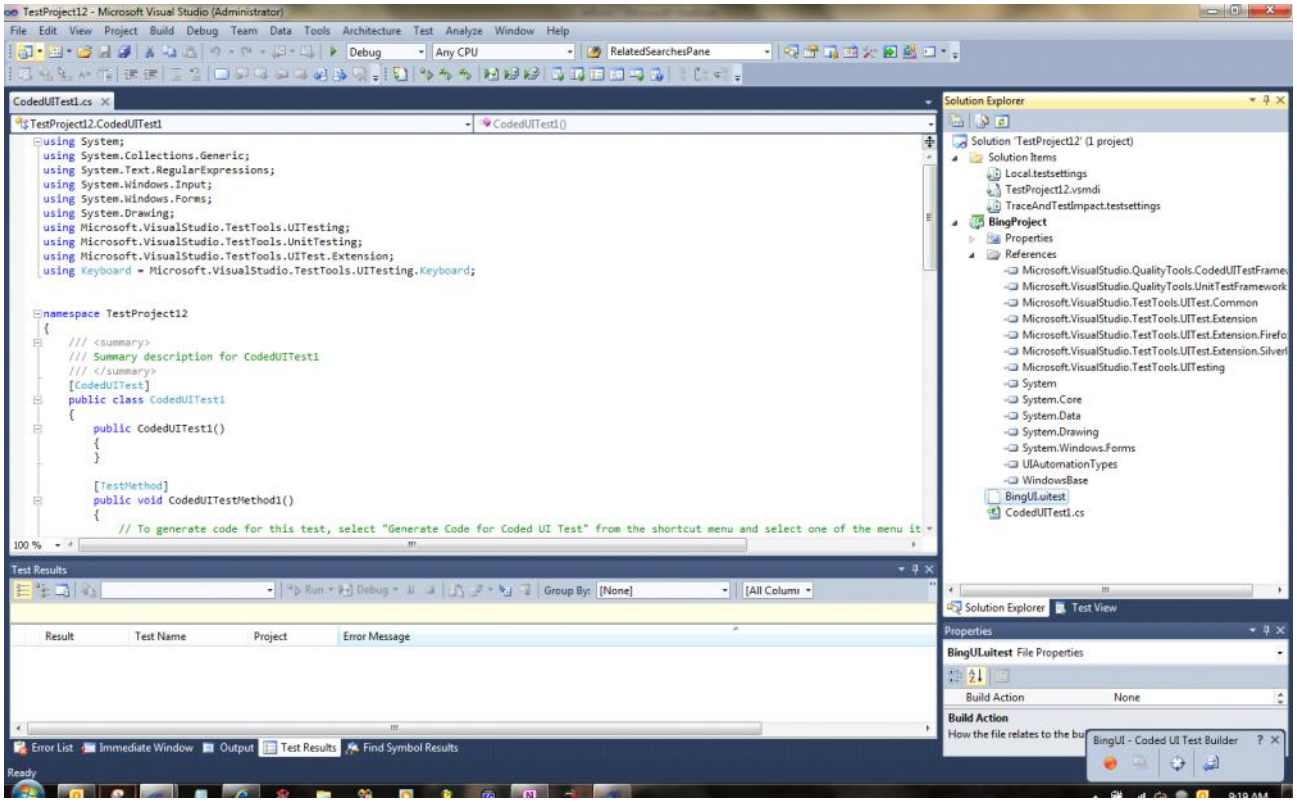




b. Not advisable to modify the name of the default UIMap added with the Script file
   i. Delete the default UIMap file
   ii. Add UIMap explicitly through path Add --> New Item --> CodedUI Test Map
   iii. Provide Meaningful name to the new UIMap file



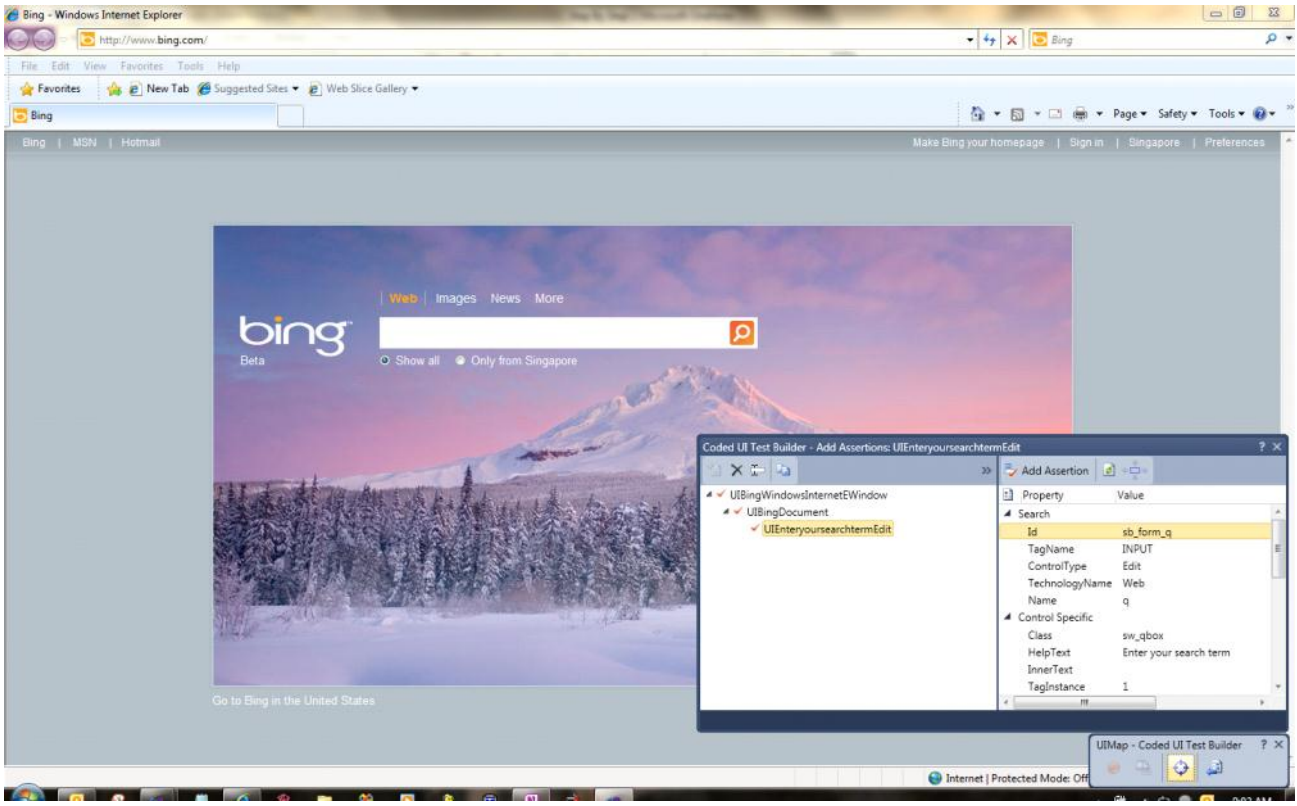
3. **Edit UIMap**
   a. **UIMap - CodedUI Test Builder Wizard** will be displayed on the bottom right corner of screen
   b. Anytime you can edit a UIMap
      i. Right click on the UIMap
      ii. Select the option **Edit With Coded UI Test Builder**



4. **Add the required objects to the UIMap**
   a. Use the **UIMap - CodedUI Test Builder Wizard**

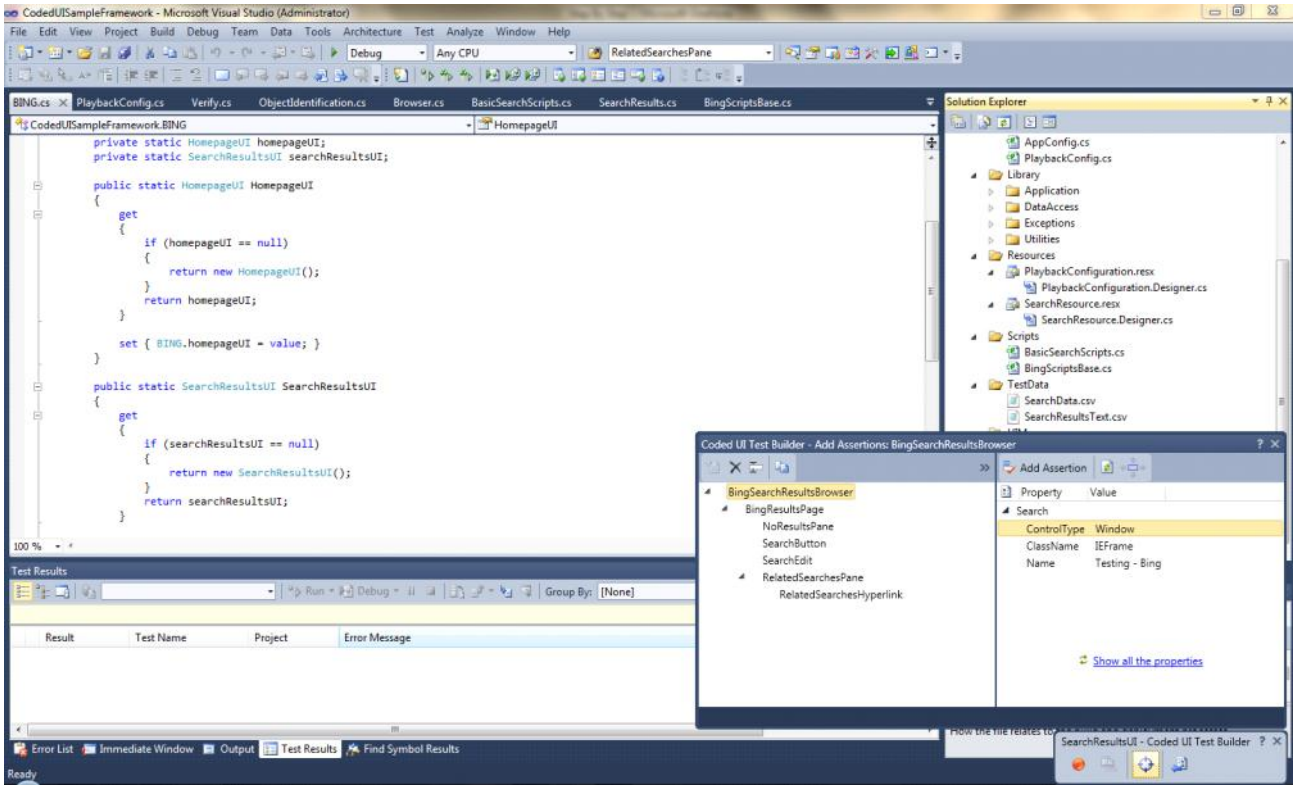
b. Steps

i. Add Objects - Drag and drop Crosshair on the required control and add the control to UIMap



ii. Rename Objects

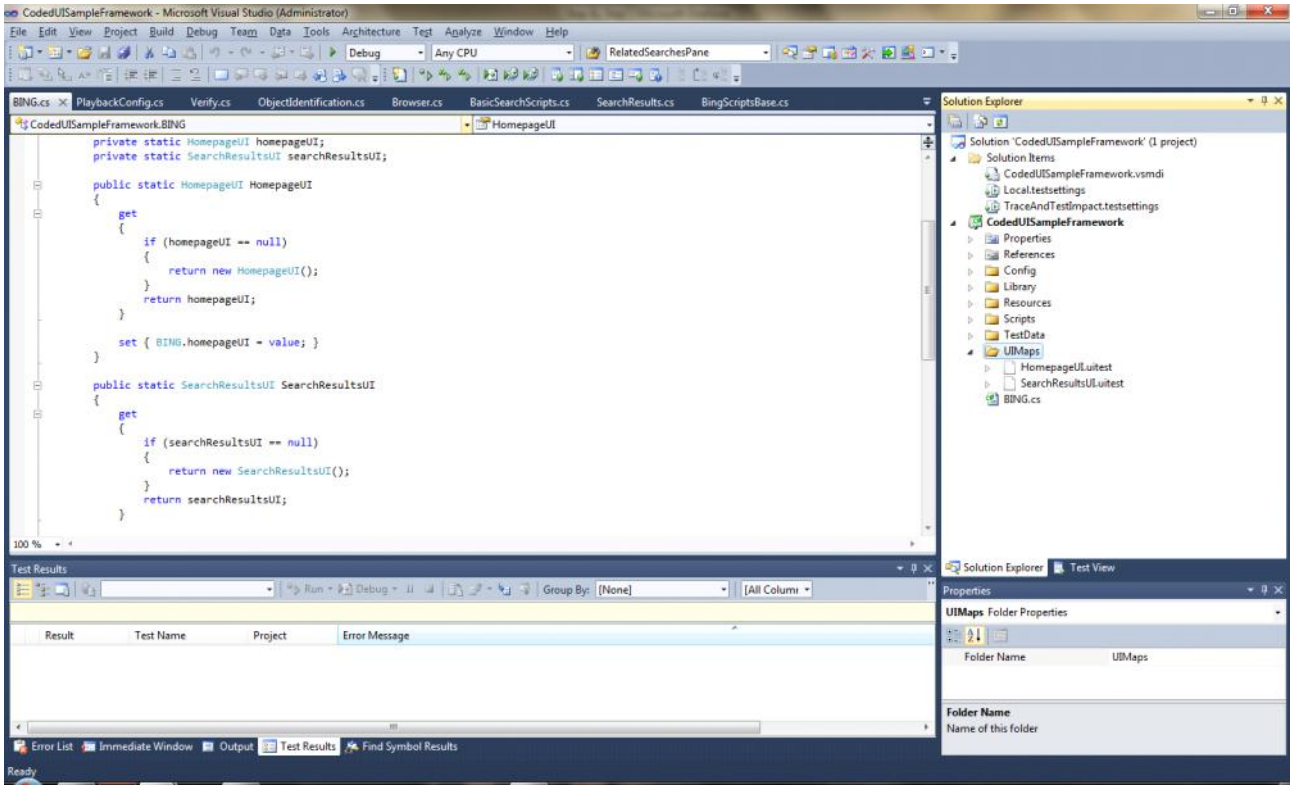
1) Provided meaningful names to each object
2) Suffix the Control Type at the end of Object Name



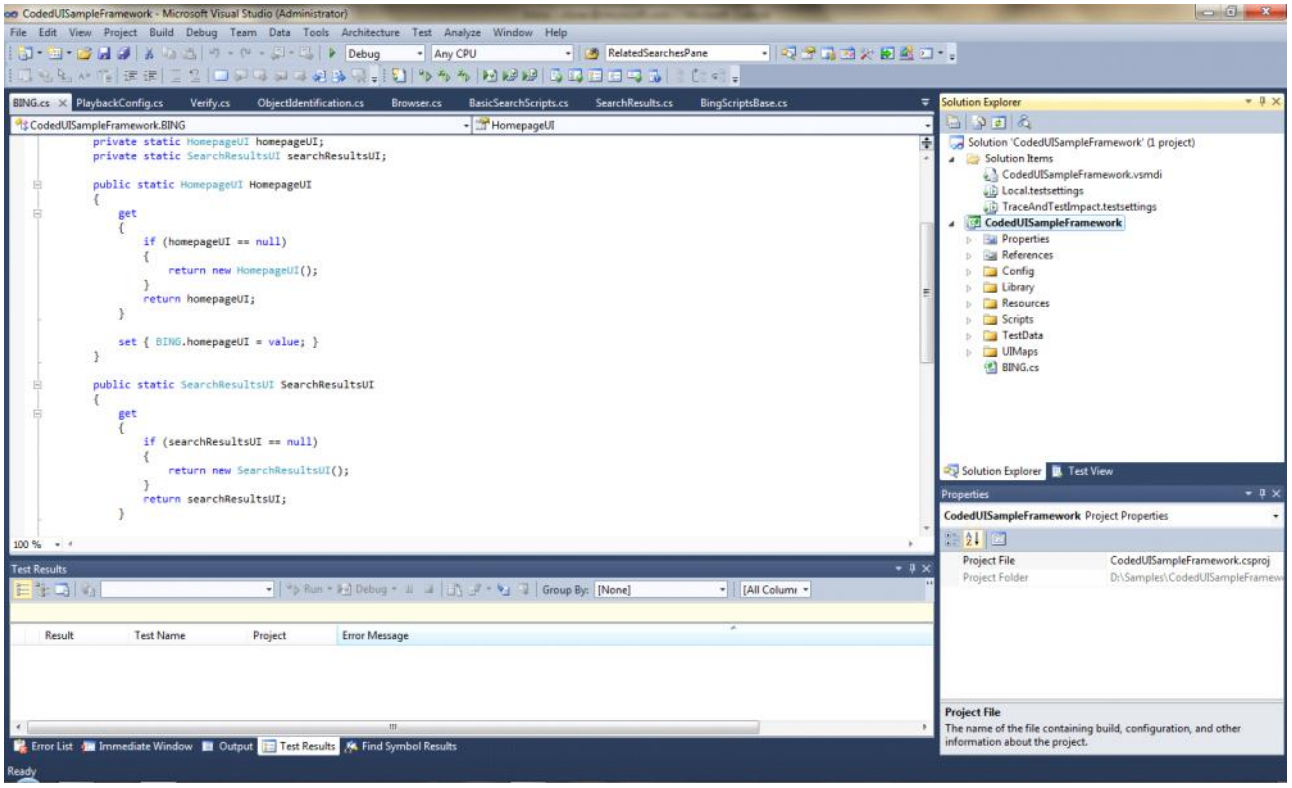
iii. Update Search Properties/Configurations for each object

iv. Add multiple pages and controls - repeat steps 1 to 3

5. Add more UIMaps based on requirement- repeat Step 4



6. Create the following Folder Structure to Divide the various responsibilities of the Framework



- **Config**
- **Library**
- **Resources**
- **Scripts**
- **TestData**

- **UIMaps**

a. **Config**
    i. Contains all files related to various Configuration Components
        1) App.config
            a) Can configure many global configurations like Environment information
        2) AppConfig.cs
            a) Wrapper API's to manipulate App.config entries

b. **Library**
    i. Applications
        1) Contains reusable Application specific libraries
    ii. DataAccess
        1) Wrapper API's to access data from TestData files
    iii. Exceptions
        1) Custom Exceptions
    iv. Utilities
        1) Generic Utilities which can be used across Projects. No applications specific contents

c. **Resources**
    i. Many reusable data used across testcases can be kept in Resource files, like SearchResource.resx
    ii. Configuration data can be kept in resource files, like PlaybackConfiguration.resx

d. **Scripts**
    i. Create one script file, with multiple scripts, covering one area of the Application
        1) BasicSearchScripts.cs
    ii. All the Script files should be inherited from the base script file
        1) BingScriptsBase.cs

e. **TestData**
    i. Keep your TestData under this folder
    ii. Create multiple sub folders if required to effectively manage the data

f. **UIMaps**
    i. Create all the UIMaps under this folder

7. Components Explained
    a. **BING.cs**
        i. Contains the entry point for accessing all the UIMaps and hence the objects added under them
        ii. This will help in globally accessing UIMap objects in various libraries and script files, rather than instantiating the object multiple times
    b. **HomepageUI.cs**
        i. Define all the controls added under this hierarchy as separate fields and associate them with the Object Hierarchy
    c. **BingScriptsBase.cs**
        i. Contains the Global Initialization and Cleanup methods
        ii. Other Global Routines like PlaybackSettings can be managed from here
        iii. Also contains the Static definition for the TestContext Instance. This will help in easily using the Testcontext instance globally
    d. **BasicSearchScripts.cs**
        i. Sample script file containing multiple scripts
        ii. Scripts should be simple
        iii. Specific Initialization and Cleanup methods
    e. **PlaybackConfiguration.resx**
        i. Easily manage the Playback Settings in this Resource file
        ii. Create similar resource files to easily manage configuration and test data
    f. **Browser.cs**
        i. Generic Browser related API's
    g. **ObjectIdentification.cs**
        i. Generic Object Identification API's. Generic API's to add update Search/Filter properties
    h. **Verify.cs**
        i. Wrapper over Assert Statements
    i. **ZeroSearchResultsException.cs**
        i. Custom Exception to throw when Zero results are displayed on screen

ii. Write similar reusable custom exceptions

j. **Data.cs**
   i. Reusable API's to easily extract data from the TestData files

k. **Homepage.cs and SearchResults.cs**
   i. Application specific reusable functions
   ii. These reusable API's will be called from Script
   iii. All complex logic should go in to the reusable API's and the Scripts should be simple

l. **App.config**
   i. Can configure many global configurations like Environment information