Best practices

Use the following guidelines to create a flexible coded UI test.

Use the **Coded UI Test Builder** whenever possible.

Do not modify the *UIMap.designer.cs* file directly. If you modify the file, the changes to the file will be overwritten.

Create your test as a sequence of recorded methods. For more information about how to record a method, see Creating coded UI tests.

Each recorded method should act on a single page, form, or dialog box. Create a new test method for each new page, form, or dialog box.

When you create a method, use a meaningful method name instead of the default name. A meaningful name helps identify the purpose of the method.

When possible, limit each recorded method to fewer than 10 actions. This modular approach makes it easier to replace a method if the UI changes.

Create each assertion using the **Coded UI Test Builder** , which automatically adds an assertion method to the *UIMap.Designer.cs* file.

If the user interface (UI) changes, re-record the test methods, or the assertion methods, or re-record the affected sections of an existing test method.

Create a separate [UIMap](https://docs.microsoft.com/en-us/previous-versions/dd580454(v%3Dvs.140)) file for each module in your application under test. For more information, see Testing a large application with multiple UI maps.

In the application under test, use meaningful names when you create the UI controls. Using meaningful names gives greater clarity and usability to the automatically generated control names.

If you are creating assertions by coding with the API, create a method for each assertion in the part of the [UIMap](https://docs.microsoft.com/en-us/previous-versions/dd580454(v%3Dvs.140)) class that is in the *UIMap.cs* file. To execute the assertion, call this method from your test method.

If you are directly coding with the API, use the properties and methods in the classes generated in the *UIMap.Designer.cs* file in your code as much as you can. These classes will make your work easier, more reliable, and will help you be more productive.

Coded UI tests automatically adapt to many changes in the user interface. If, for example, a UI element has

changed position or color, most of the time the coded UI test will still find the correct element.

During a test run, the UI controls are located by the testing framework by using a set of search properties. The search properties are applied to each control class in the definitions created by the **Coded UI Test Builder** in the *UIMap.Designer.cs* file. The search properties contain name-value pairs of property names and property values that can be used to identify the control, such as the [FriendlyName](https://docs.microsoft.com/en-us/dotnet/api/microsoft.visualstudio.testtools.uitesting.uitestcontrol.friendlyname), [Name](https://docs.microsoft.com/en-us/dotnet/api/microsoft.visualstudio.testtools.uitesting.uitestcontrol.name), and [ControlType](https://docs.microsoft.com/en-us/dotnet/api/microsoft.visualstudio.testtools.uitesting.uitestcontrol.controltype) properties of the control. If the search properties are unchanged, the coded UI test will successfully find the control in the UI. If the search properties are changed, coded UI tests have a smart match algorithm that applies heuristics to find controls and windows in the UI. When the UI has changed, you might be able to modify the search properties of previously identified elements to make sure that they are found.

If your user interface changes

User interfaces frequently change during development. Here are some ways to reduce the effect of these changes:

Find the recorded method that references this control, and use the **Coded UI Test Builder** to re-record the actions for this method. You can use the same name for the method to overwrite the existing actions.

If a control has an assertion that is no longer valid: Delete the method that contains the assertion. Remove the call to this method from the test method.

Add a new assertion by dragging the cross-hair button onto the UI control, open the UI map, and add the new assertion.

For more information about how to record coded UI tests, see Use UI automation to test your code.