



Test Automation FX

UI test automation for .NET developers

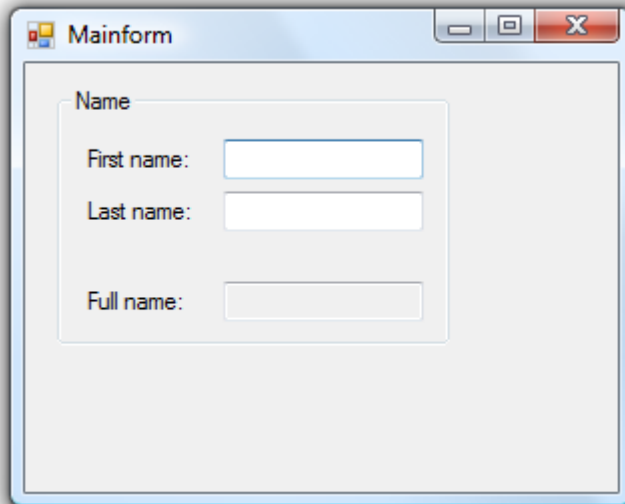
Access properties and methods in the tested application

Invoking methods

When testing applications that are developed with .NET you can use the features built into Test Automation FX allowing you to call methods and set properties for the controls in the AUT. This can be used to combine API testing with UI testing. It might also be a convenient way to set up the AUT and its objects in a known state for the UI tests.

The ability to call methods on objects in the AUT might also be a great workaround for automating objects that are hard to control with mouse clicks and keystrokes in a reproducible manner.

Let's consider an application that looks like this:



Picture 1. The Mainform of the EnterNameApp

It might be nice to have a method to clear the fields when we perform the UI tests. Then we can add a ClearFields method to the MainForm class.

```
public void ClearFields()  
{  
    m_FirstName.Text = "";  
    m_LastName.Text = "";  
    m_FullName.Text = "";  
}
```

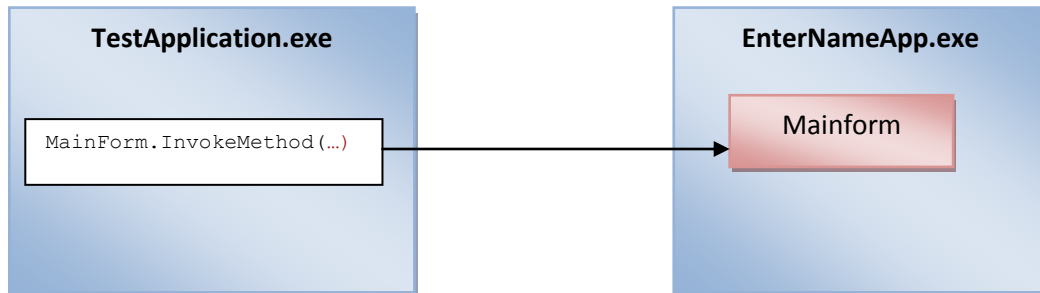
Now if we add the MainForm to the UIMap we can call the ClearFields method remotely simply by writing code in our test application.

```
MainForm.InvokeMethod("ClearFields");
```



Test Automation FX

UI test automation for .NET developers



Picture 2. Methods in the AUT can be called from the TestApplication with the InvokeMethod command.

If the method has parameters they can be added in the InvokeMethod call. Consider the following method:

```
public void SetFields(string firstName, string lastName)
{
    m_FirstName.Text = firstName;
    m_LastName.Text = lastName;
}
```

The SetFields method can be called with parameters.

```
MainForm.InvokeMethod("ClearFields", "Tom", "Jones");
```

or

```
MainForm.InvokeMethod("ClearFields", firstName, lastName);
```

If the method throws an exception it will be caught in the AUT and rethrown in our application. So if we expect the method to fail with an exception we have to include it in a try-catch statement.

Setting and getting properties

Another way to interact with the objects of the AUT is to set properties on them. This is done by using the Properties property of the UIObjects.

For instance if we want to set the BackColor of the MainForm in EnterNameApp to 'red' we do that by writing code like this.

```
MainForm.Properties["BackColor"] = Color.Red;
```

In the same manor properties can be read. This will show the text properties:

```
MessageBox.Show(FullName.Properties["Text"].ToString());
```

The VerifyProperty method will also access managed properties if they are available.

```
MainForm.VerifyProperty("BackColor", Color.Red);
```

