

---

# Login Manager

---

## Contents

Introduction .....	2
Login Management Framework.....	2
Windows Form Application Sample .....	2
Password Management .....	2
Reset password – Create reset code.....	3
ResetCode property .....	3
Create reset code .....	4
Show error .....	7
Abort on error .....	10
Get email address .....	11
Set email address .....	14
Include reset code in email .....	17
Send email.....	20
Show notification .....	21
Remove email address .....	22
Protect the form .....	24
Test.....	25
Reset password – set new password .....	27
Verify inputs .....	27
Reset password .....	28
Show error message .....	34
Show success.....	37
Test.....	40
Permission Level .....	42
Custom Permission Control .....	46
Disable user level .....	47
Check permission .....	47

Show “permission denied” .....	51
Close form if permission denied .....	54
Test.....	56
Web Project Sample.....	58
PHP Web Project Sample .....	58
ASPX Web Project Sample .....	58

## Introduction

This is part B2 of Login Manager document. The whole document consists of following files.

- Part A – It is a reference to the Login Management Framework. It can be downloaded from <http://www.limnor.com/support/LoginManagerPartA.PDF>
- Part B1 – It contains the first part of a Windows Form Application sample. It can be downloaded from <http://www.limnor.com/support/LoginManagerPartB1.PDF>
- Part B2 – This file. It contains the second part of a Windows Form Application sample.
- Part C – It contains a PHP web application sample. It can be downloaded from <http://www.limnor.com/support/LoginManagerPartC.PDF>
- Part D – It contains an ASPX web application sample. It can be downloaded from <http://www.limnor.com/support/LoginManagerPartD.PDF>

The files for sample projects are long but with little text. Most of contents are screenshots showing step by step programming procedures of sample projects. Most steps look similar and repetitive. For beginners, please pay attention to the highlighting in the screenshots, especially the highlighting in the Expression Editor: the result of each operation depends on which parts of an expression are highlighted.

## Login Management Framework

See part A from <http://www.limnor.com/support/LoginManagerPartA.PDF>

## Windows Form Application Sample

Please read Part B1 before reading this document.

<http://www.limnor.com/support/LoginManagerPartB1.PDF>

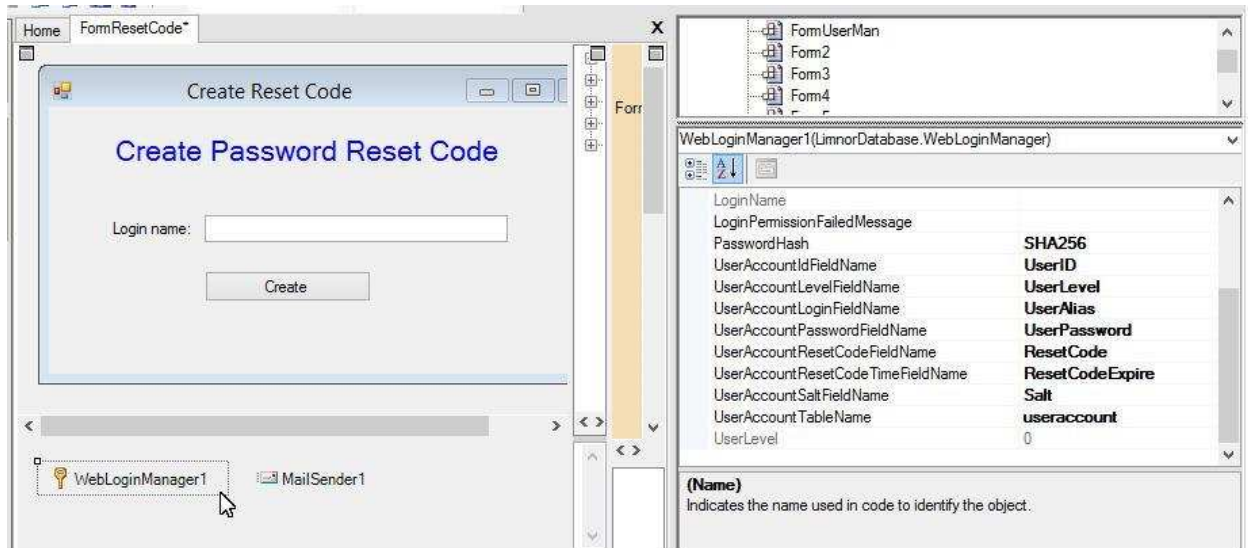
## Password Management

Part B1 contains samples for setting initial password and changing password. This part contains samples for resetting password and permission controls.

## Reset password – Create reset code

For a user forgetting his/her password, there is not a way to retrieve the password from the computer. We may create a reset code and send the code to the user. The user may use the code as a credential and change the password.

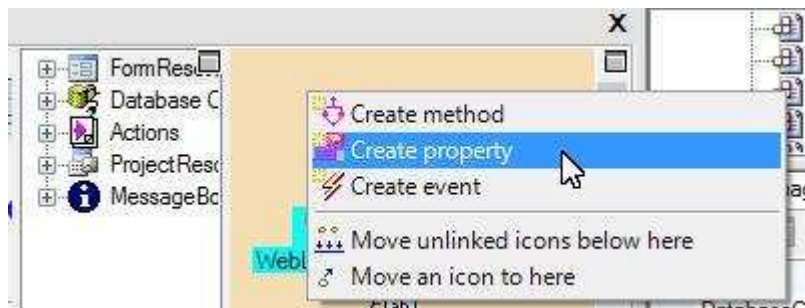
We'll create a form for generating and sending reset code. The form will be protected to restrict access.



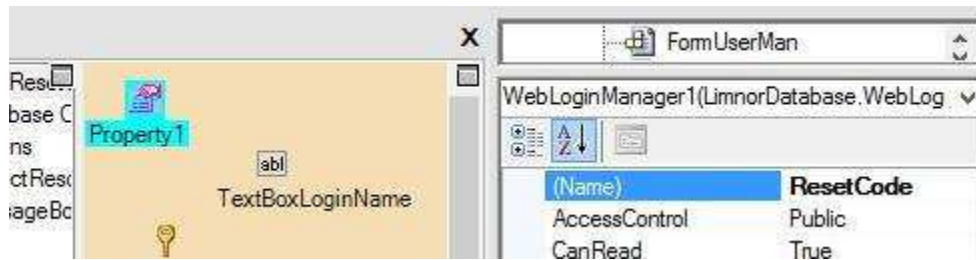
Note that the properties of the WebLoginManager component must be the same as those for the WebLoginManager component in the login form.

## ResetCode property

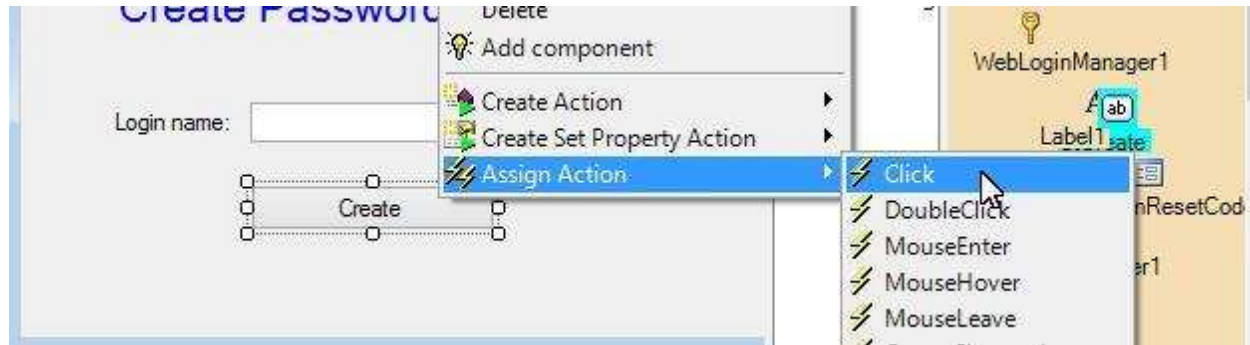
Create a ResetCode property to hold the reset code to be created:



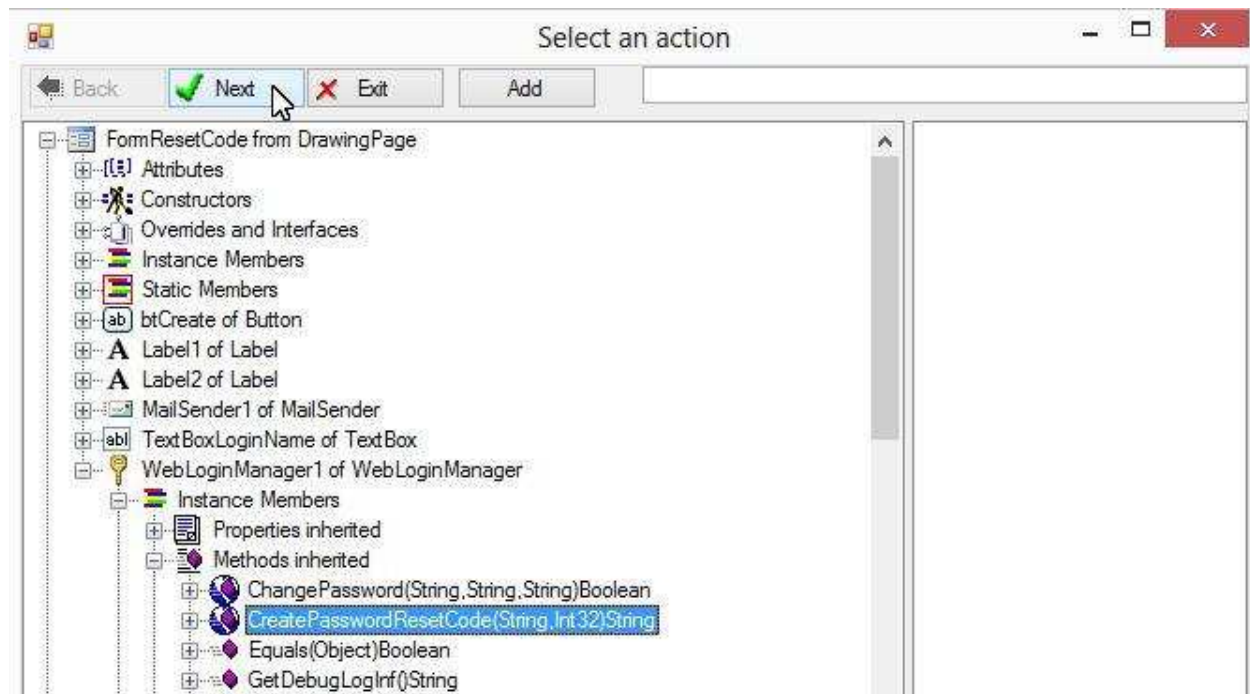
Rename the new property to ResetCode:



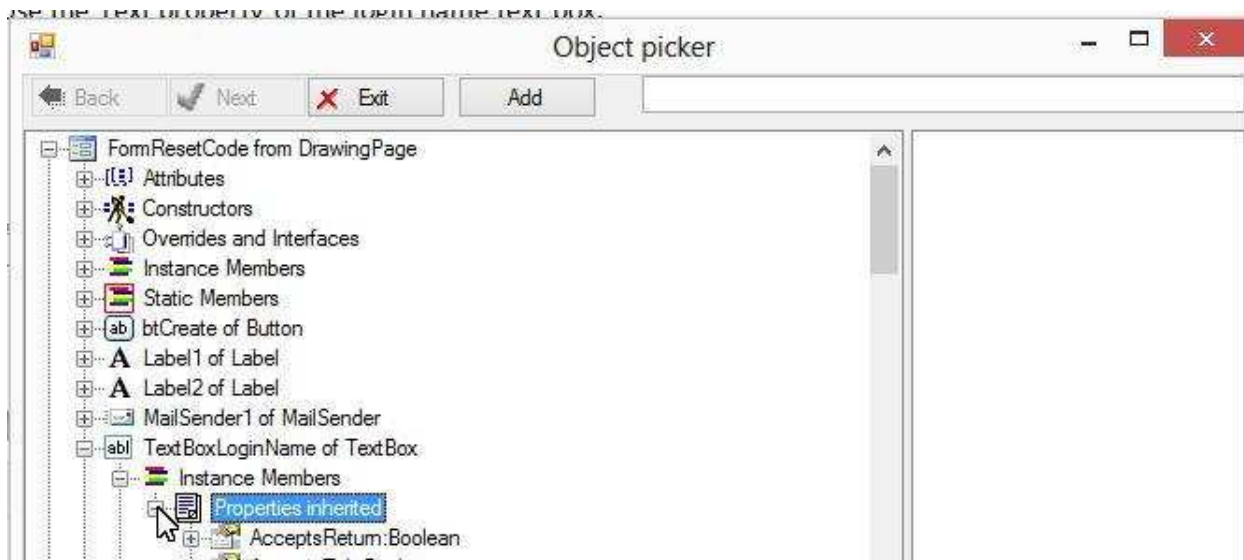
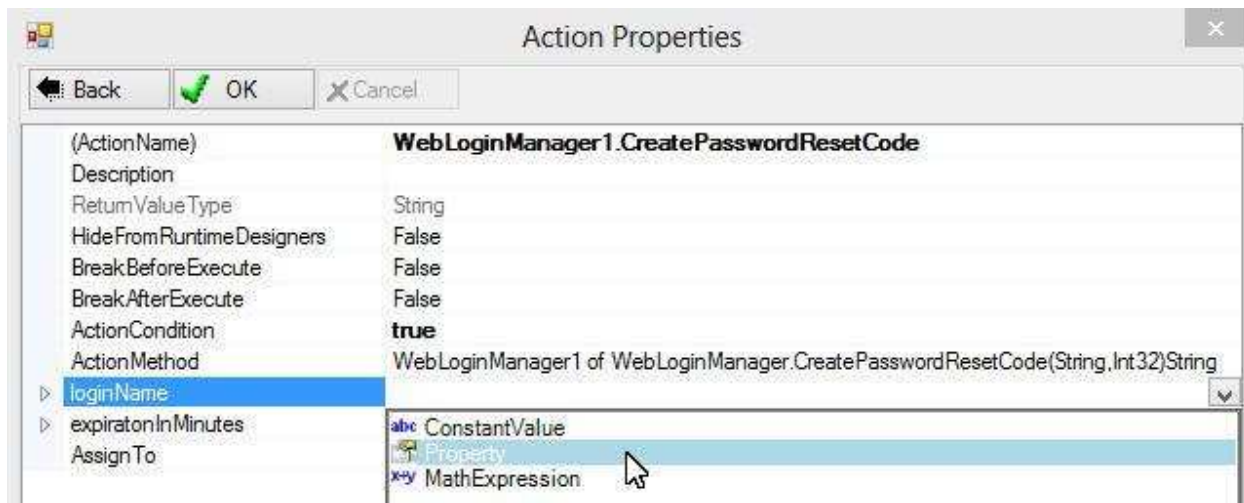
## Create reset code



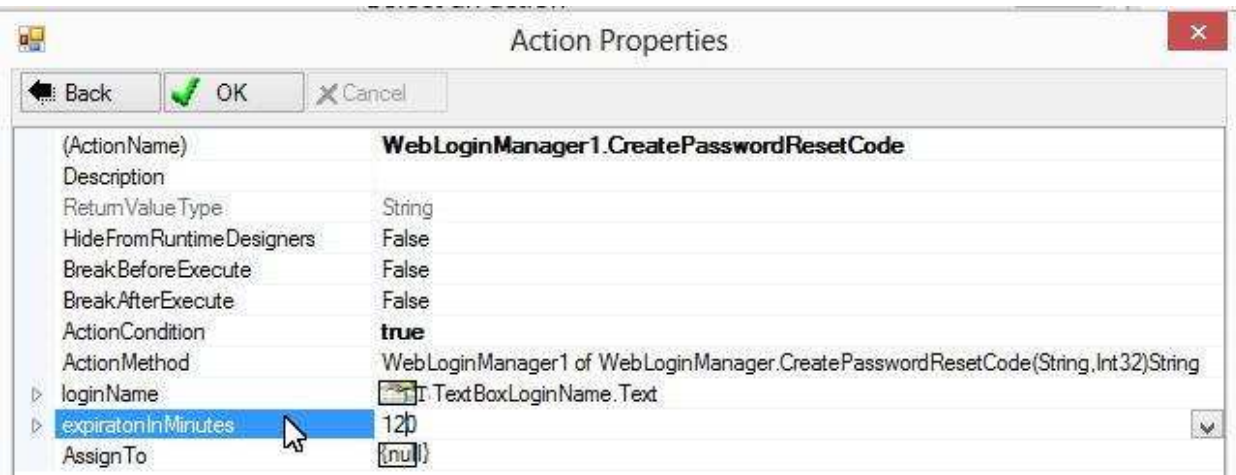
Choose CreatePasswordResetCode method:



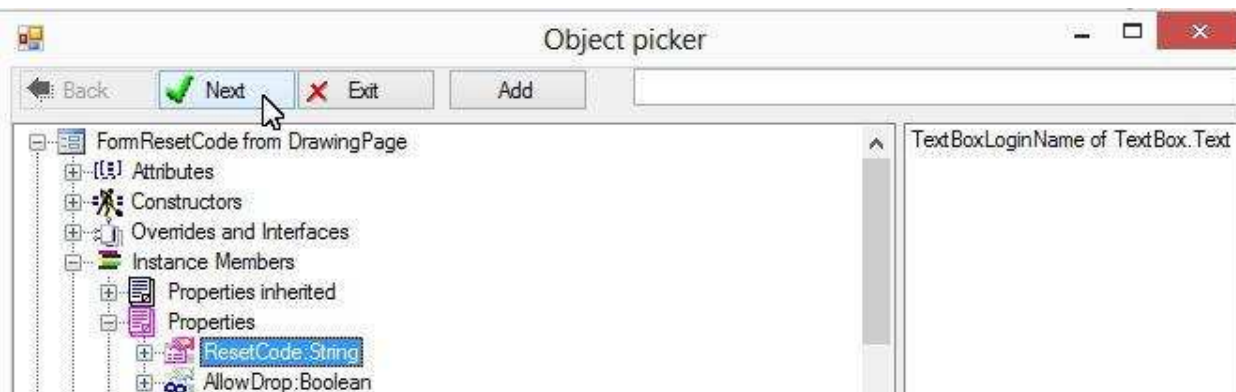
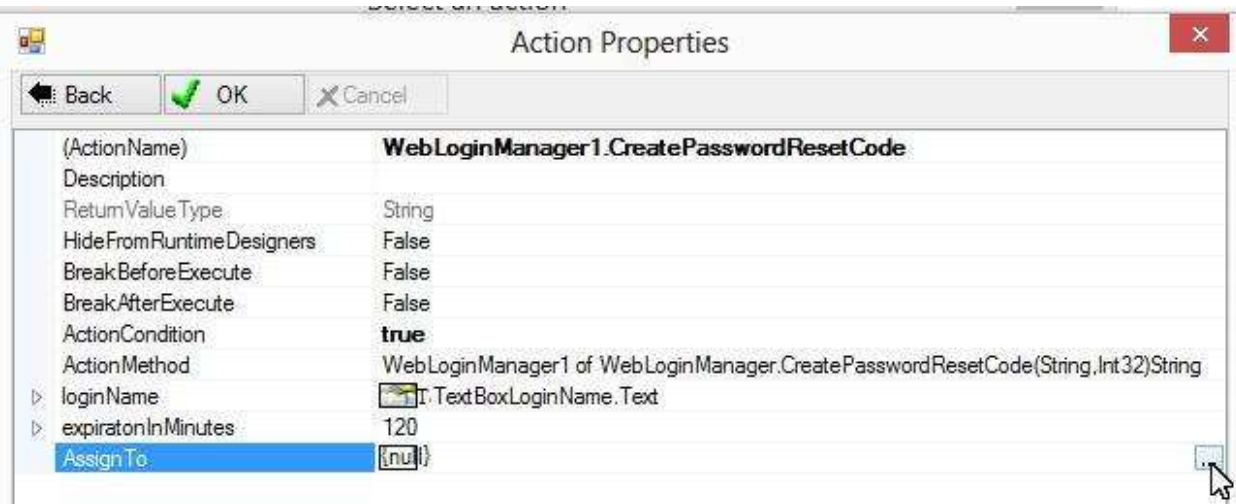
Use the Text property of the login name text box:



“expirationInMinutes” indicates how long the reset code will stay valid. We use 120 so that the reset code will expire in 2 hours:

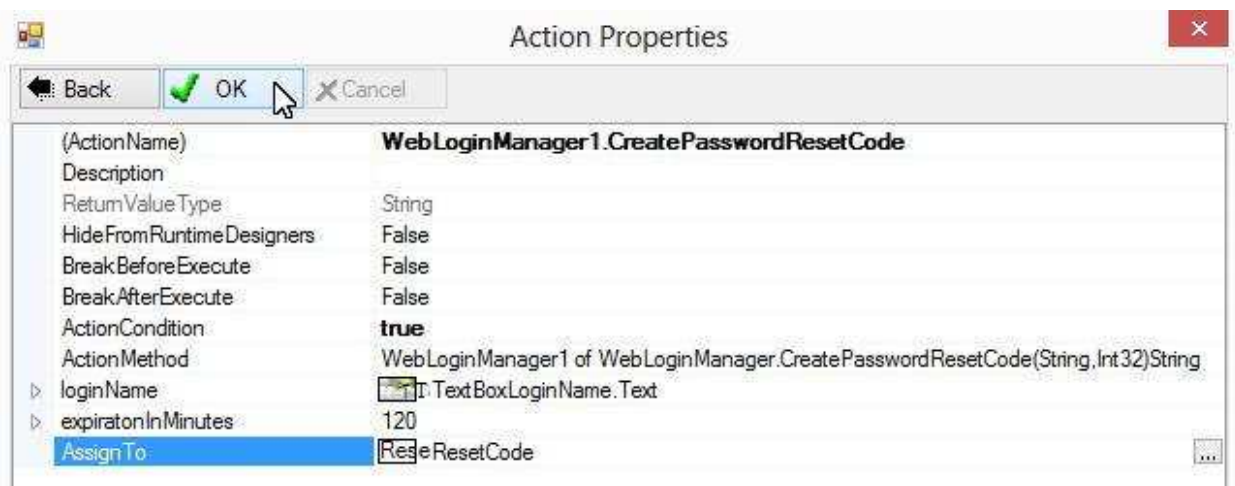


This action will return the reset code. Set "AssignTo" to property ResetCode:

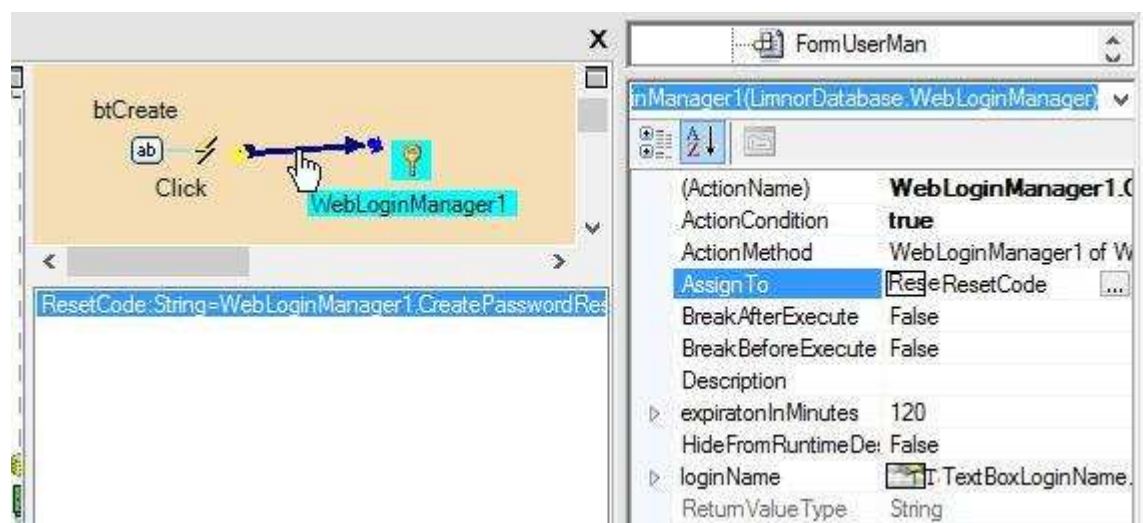


Click OK:



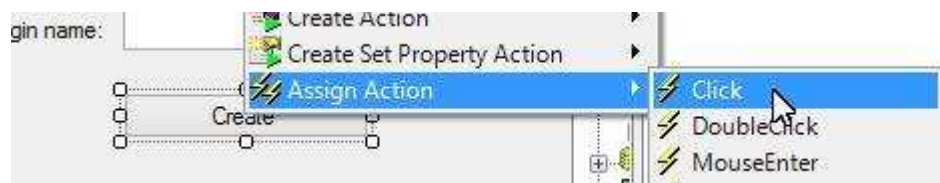


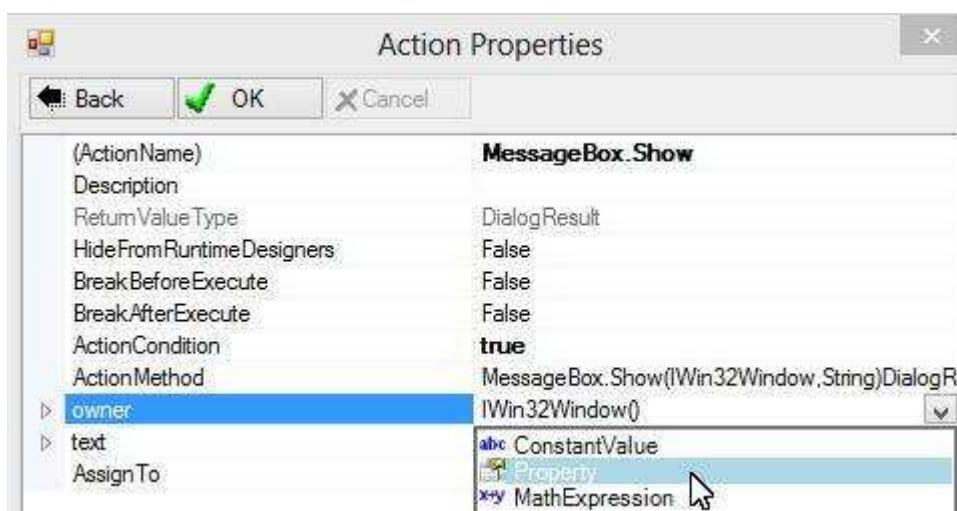
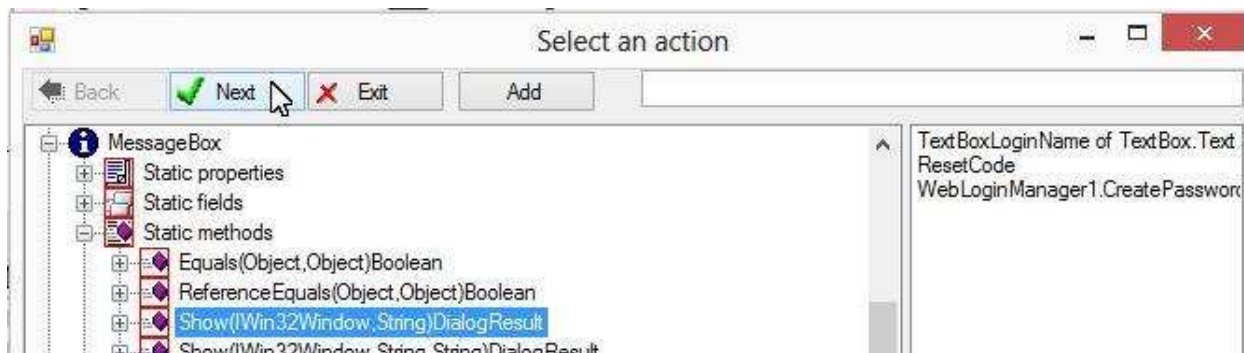
The action is created and assigned to the button:



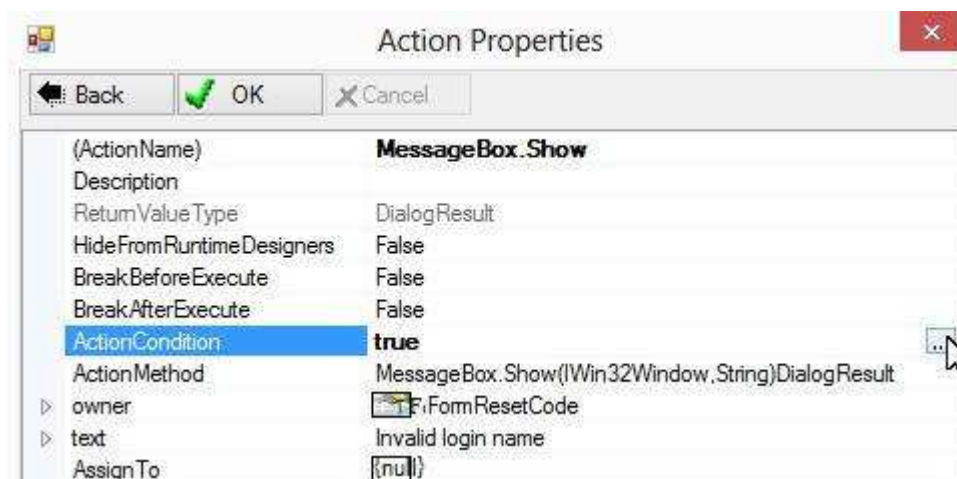
## Show error

If the above action fails then ResetCode property is empty. We may show an error message.

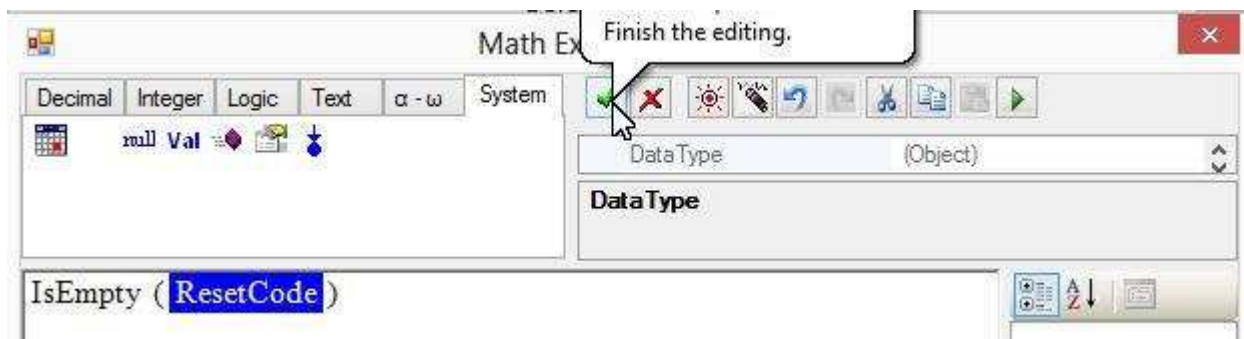
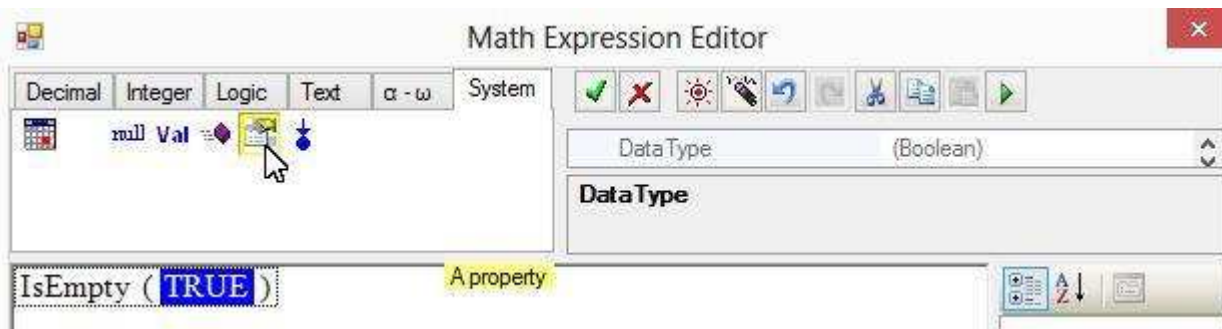
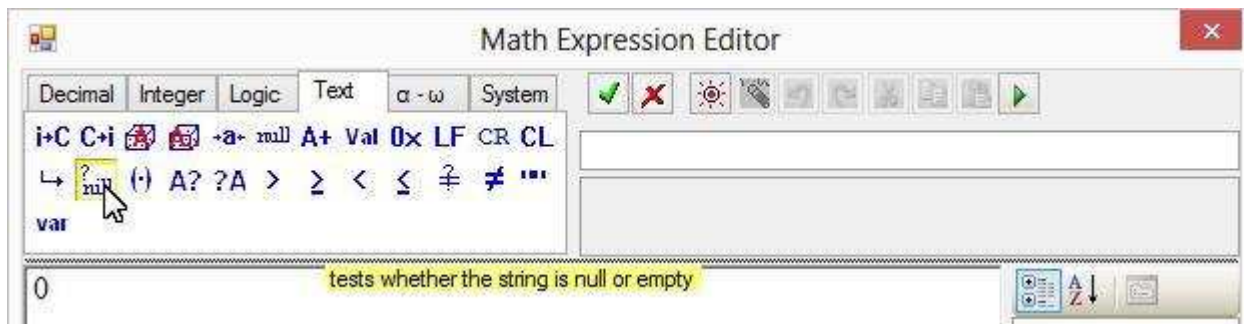


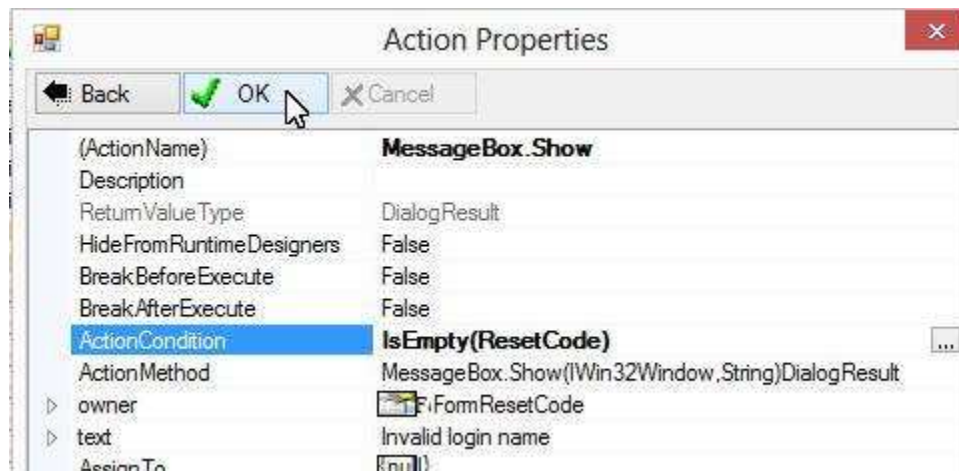


Set the message text and ActionCondition:

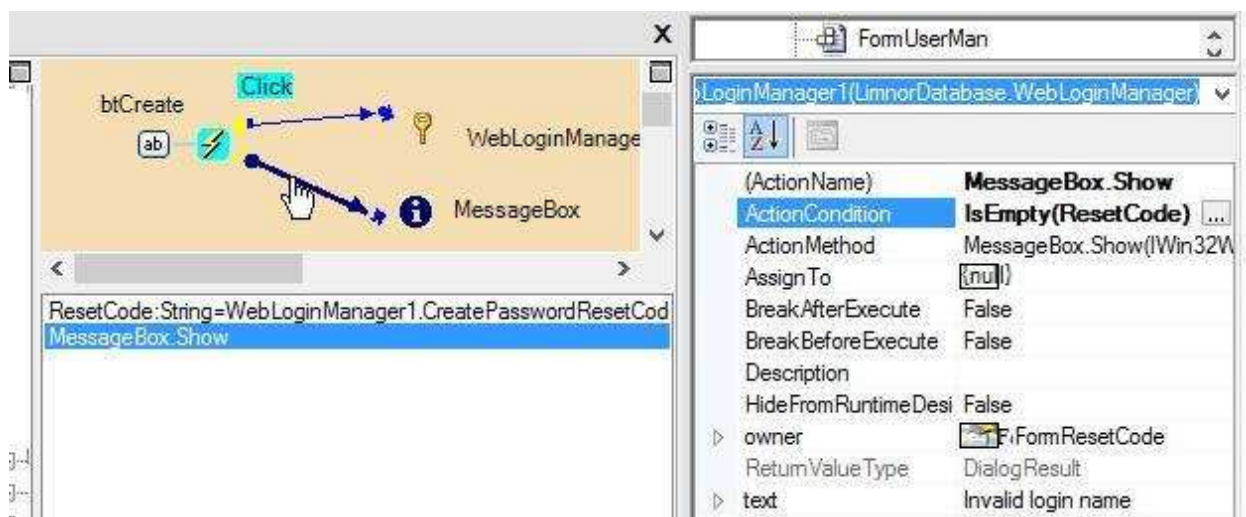




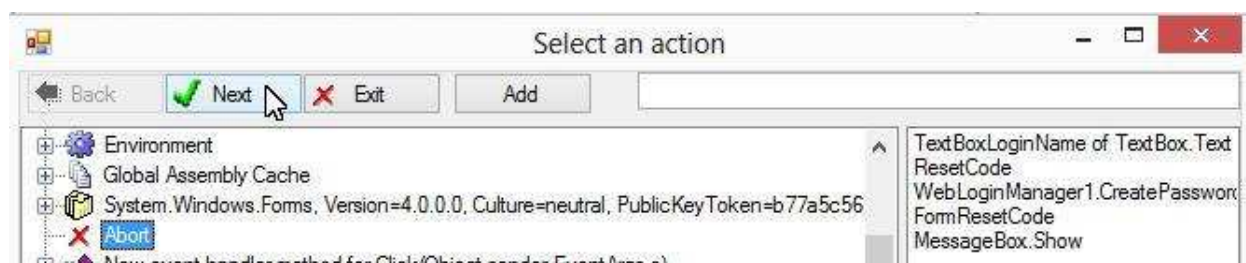
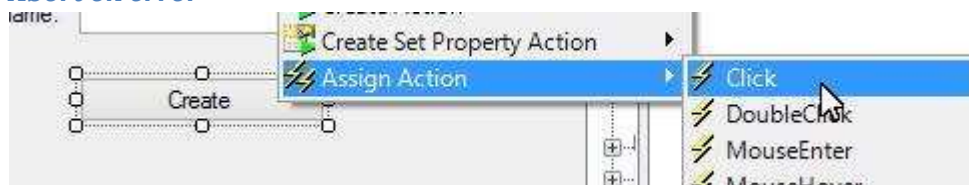




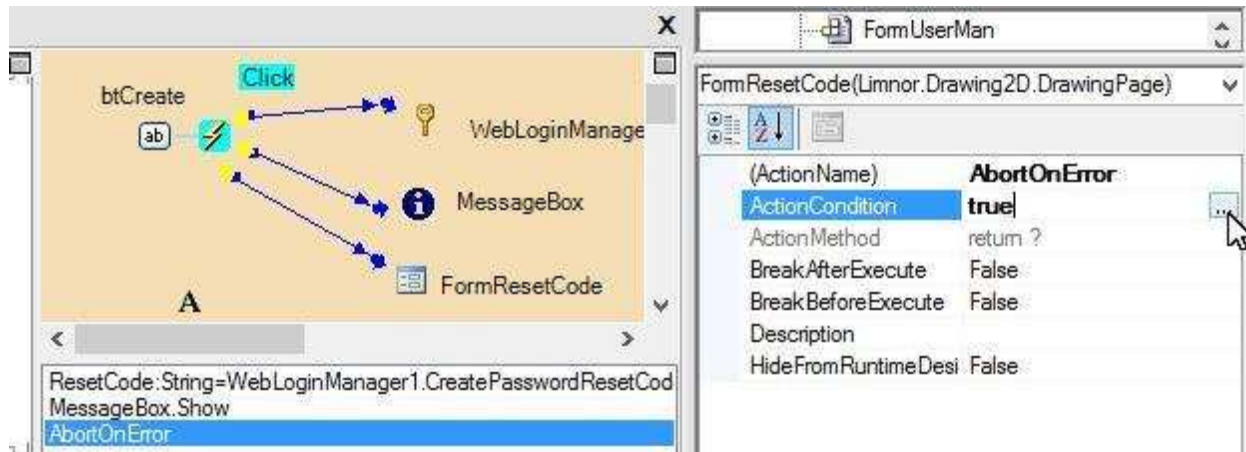
The action is created and assigned to the button:



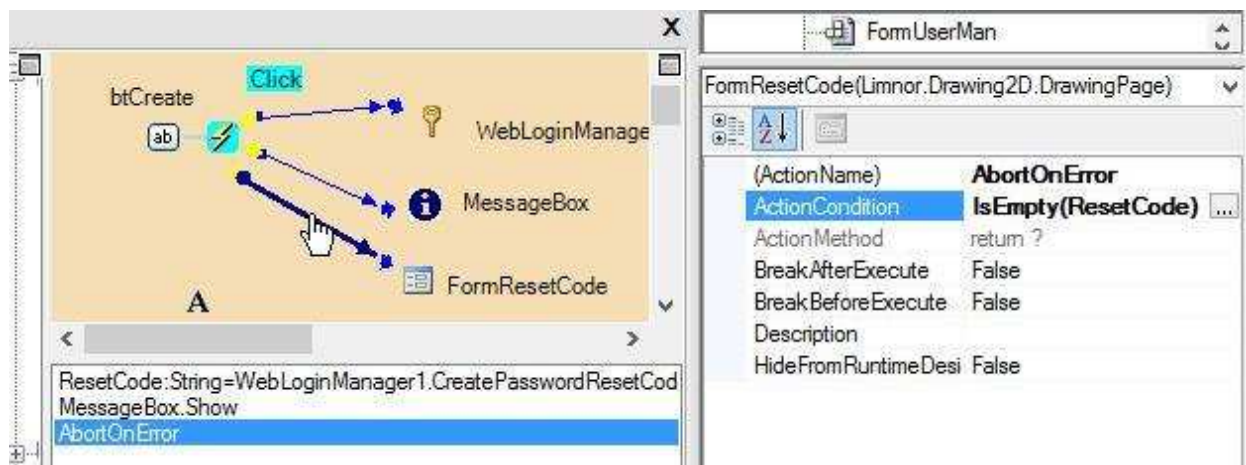
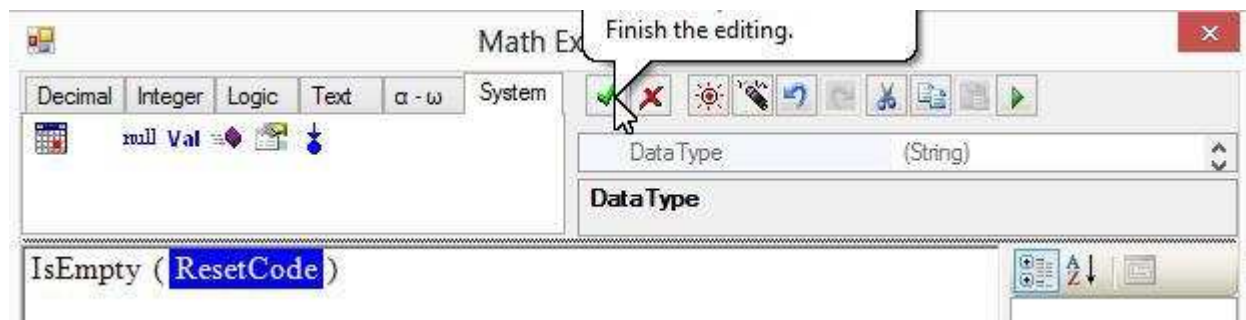
## Abort on error



Rename it and set its ActionCondition:

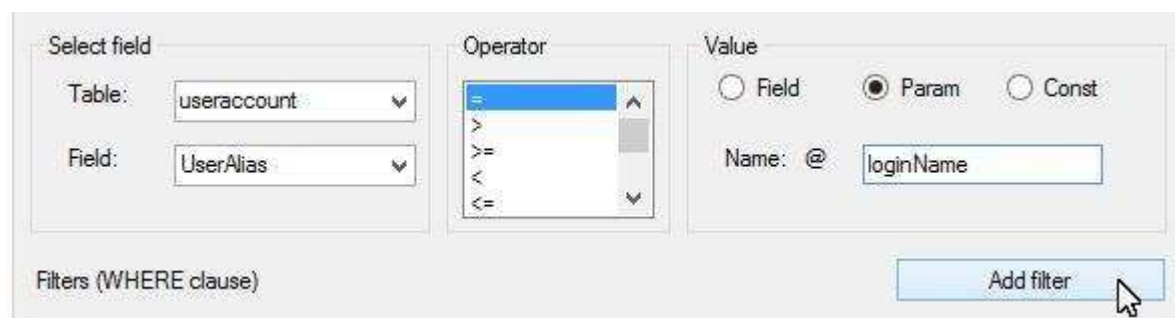
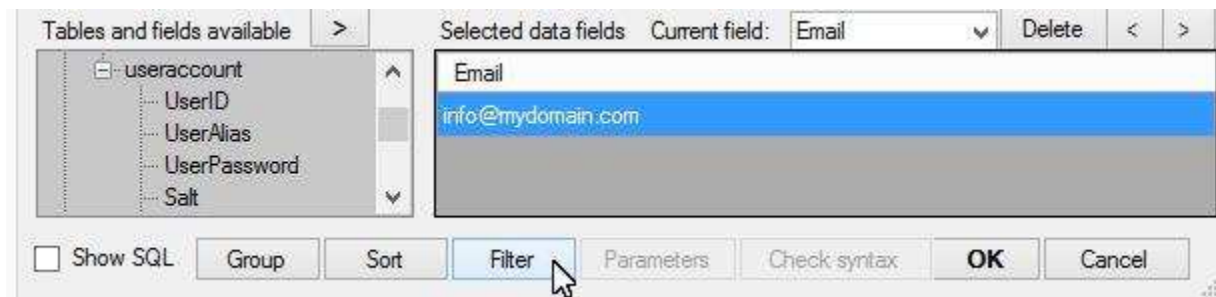
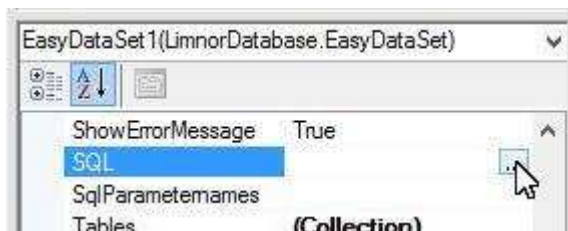


Use the same condition as for showing the error message box:



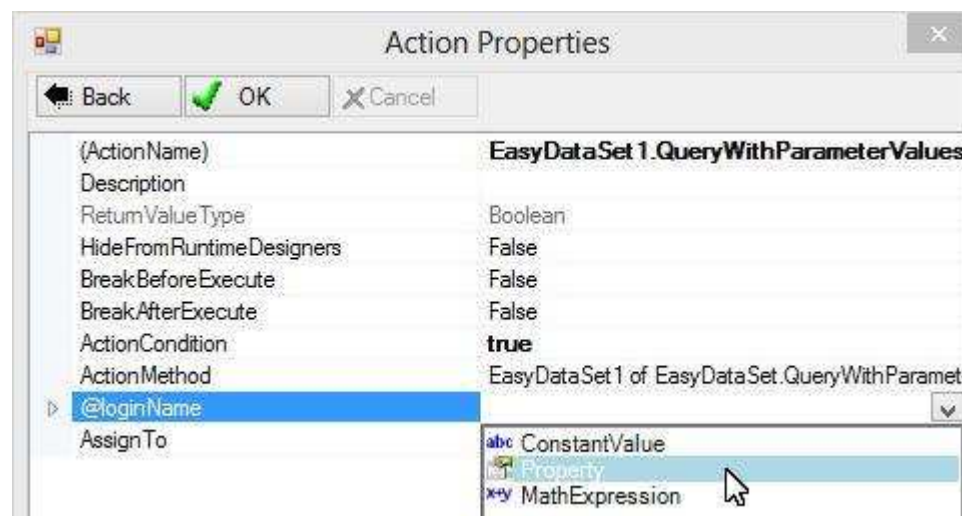
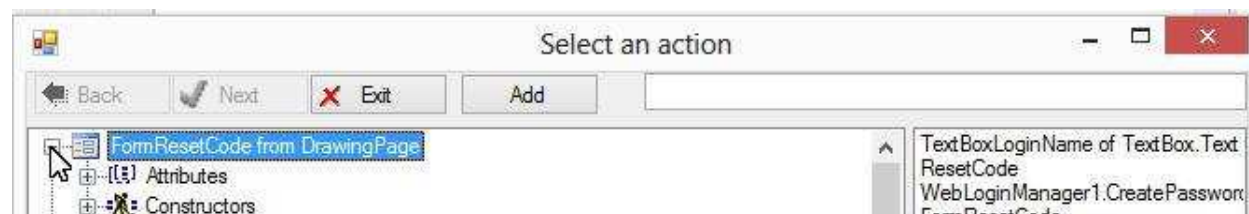
### Get email address

We may use an EasyDataSet to get email address:

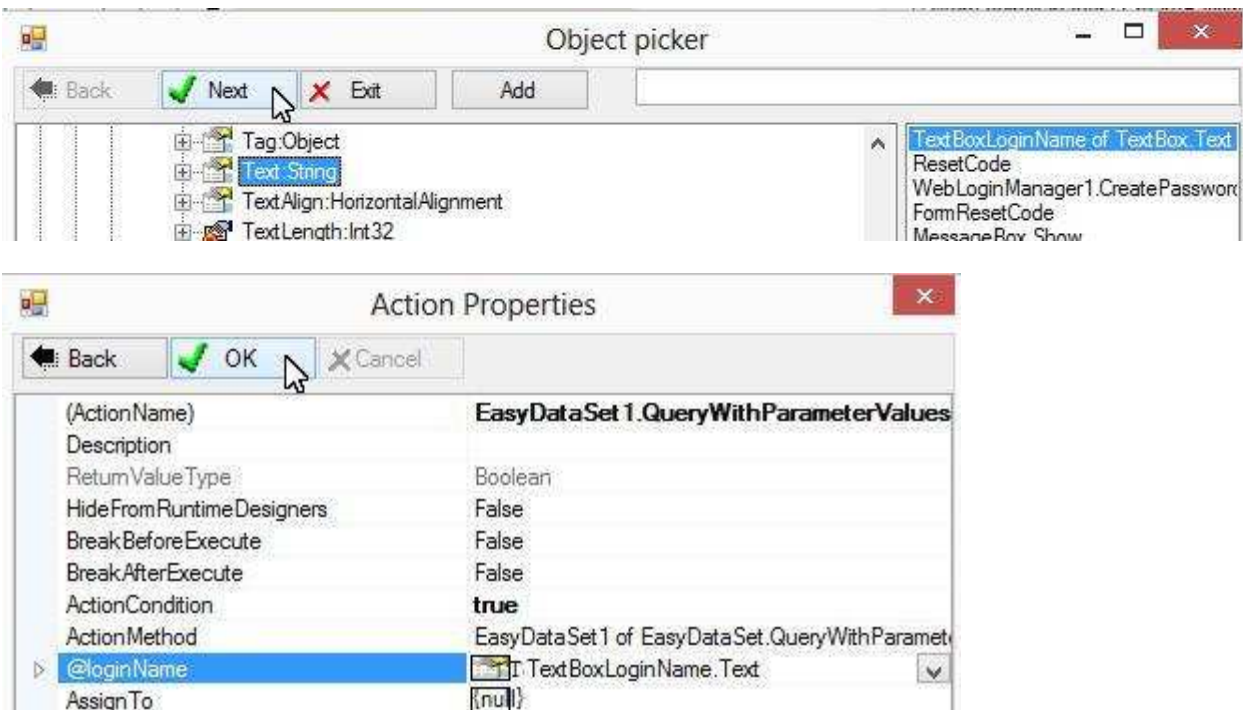




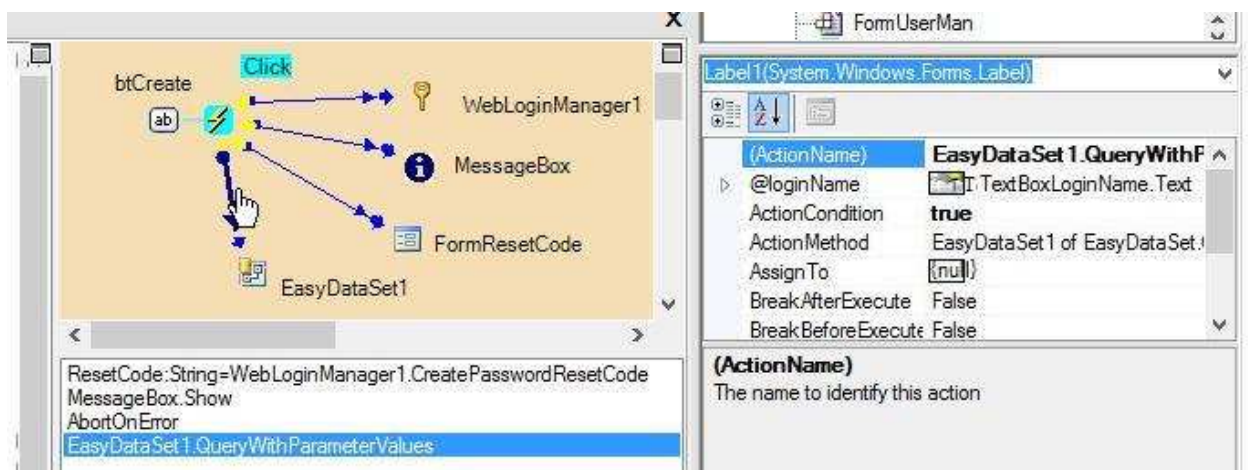
We may execute a QueryWithParameterValues action to get email address:



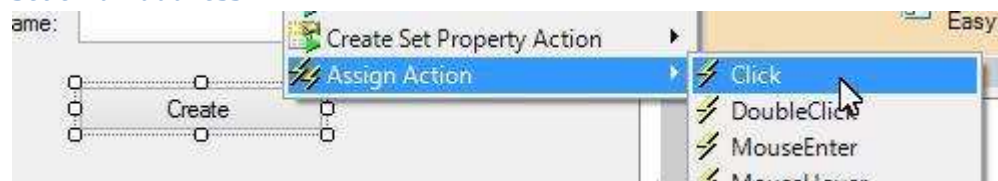


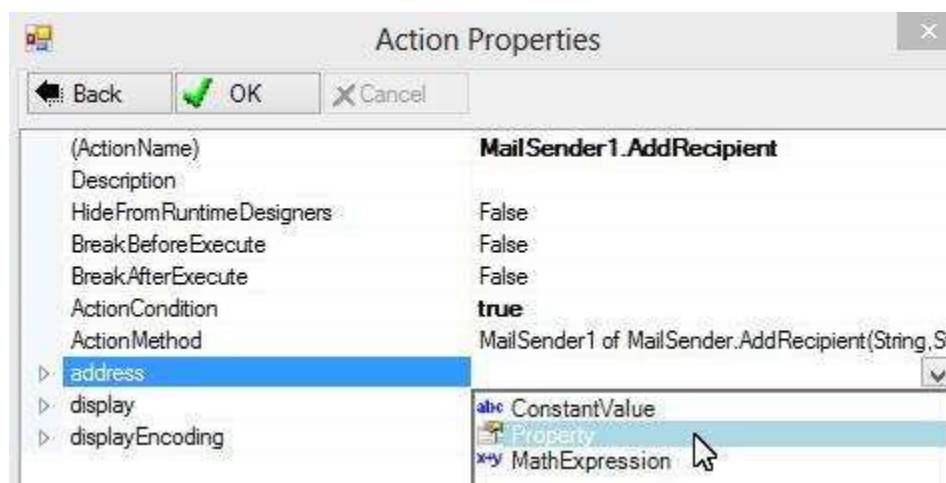
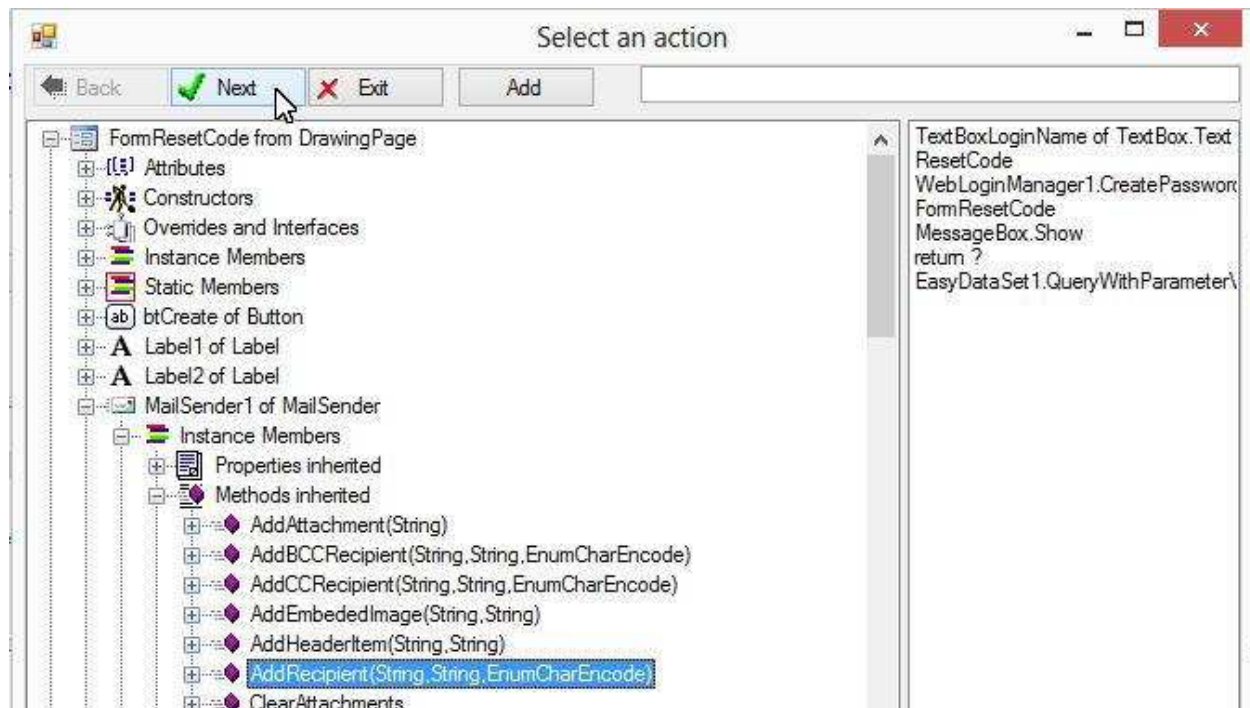


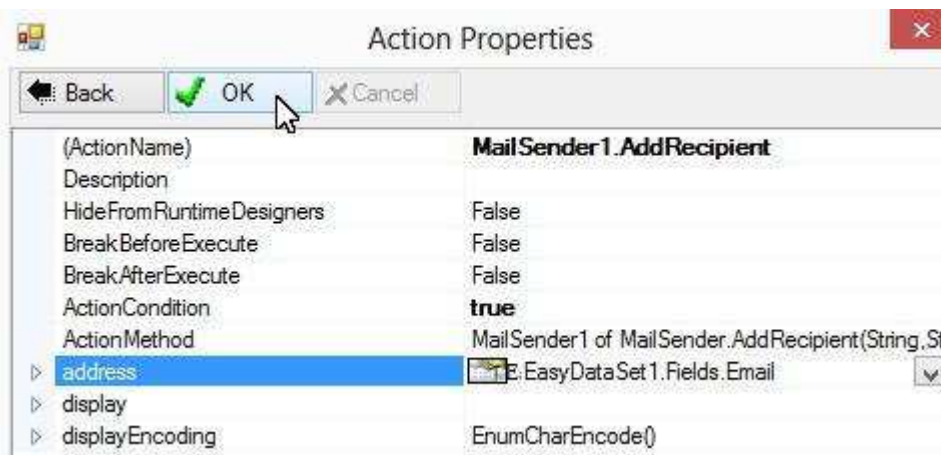
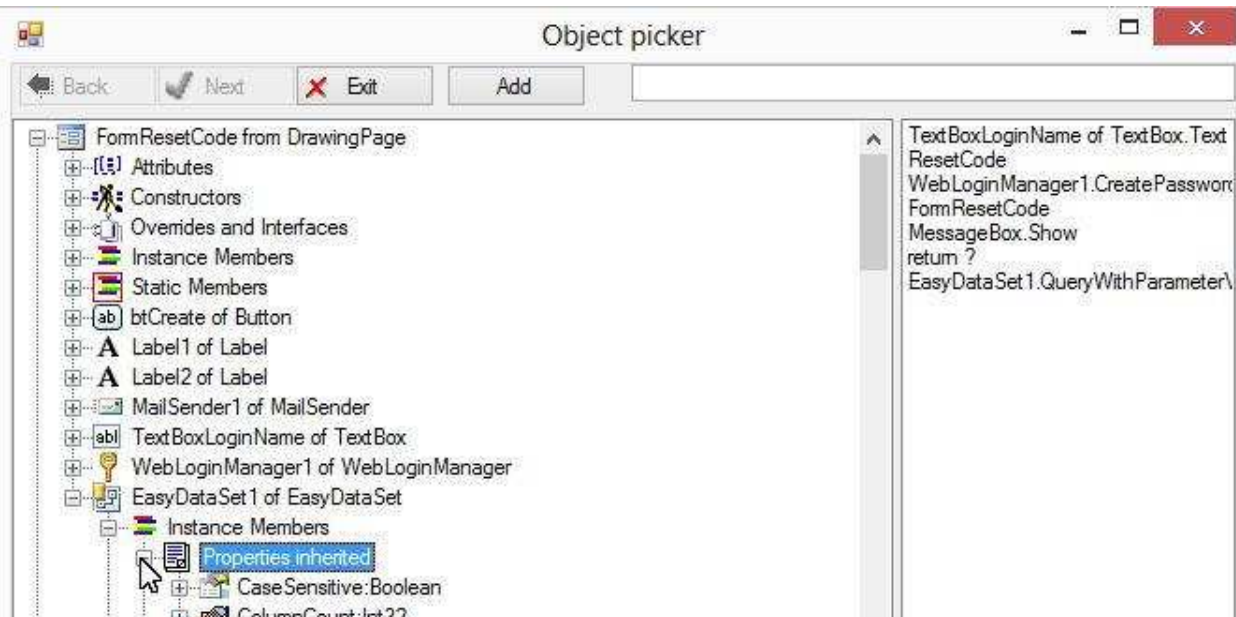
The action is created and assigned to the button:



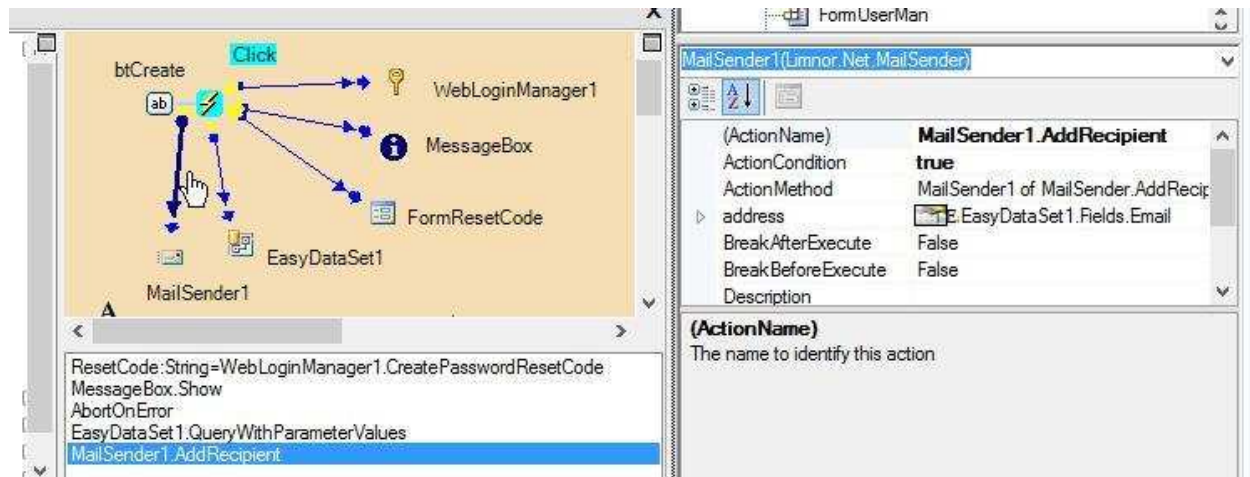
## Set email address



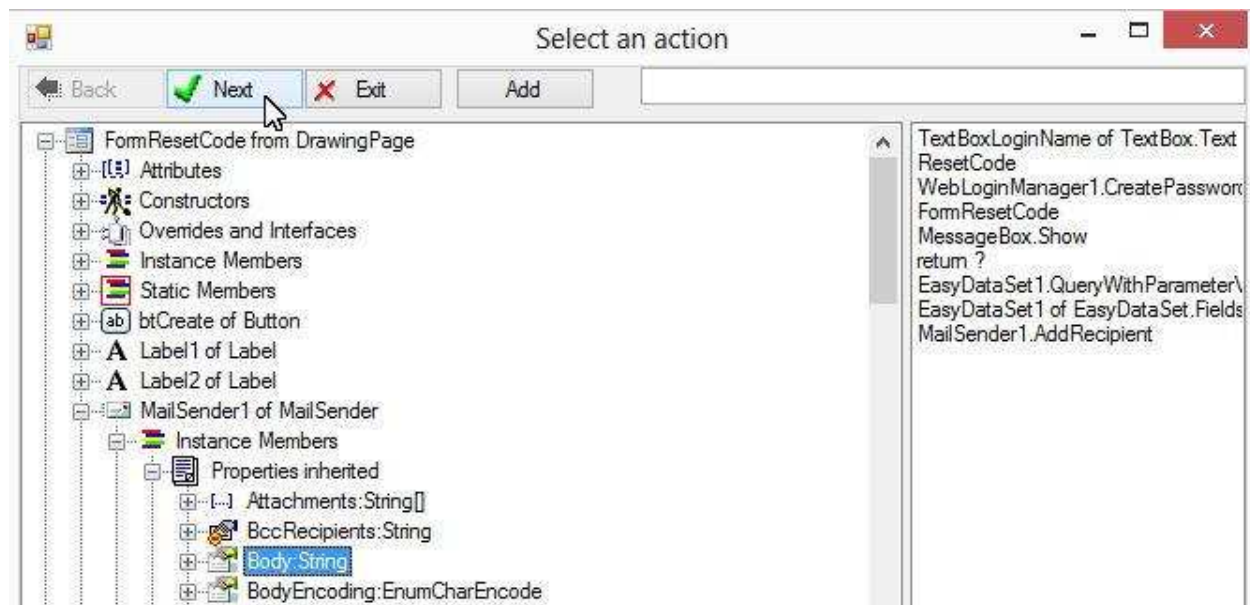
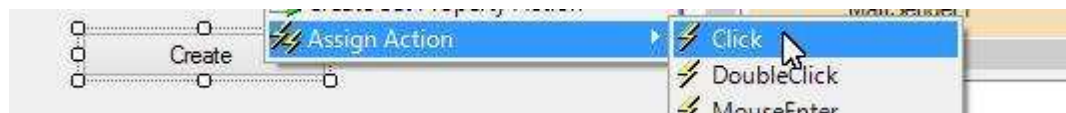




The action is created and assigned to the button:

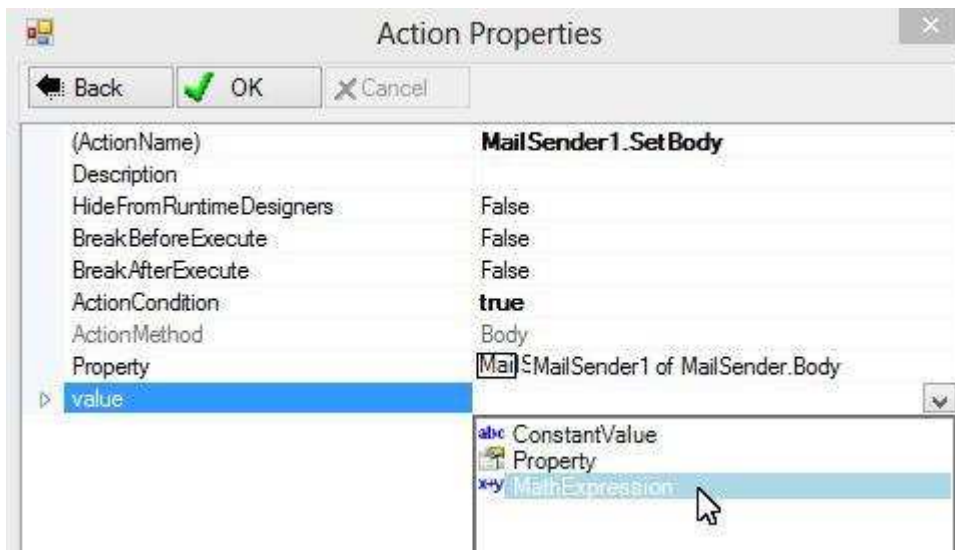


## Include reset code in email



Use an expression to form email message:





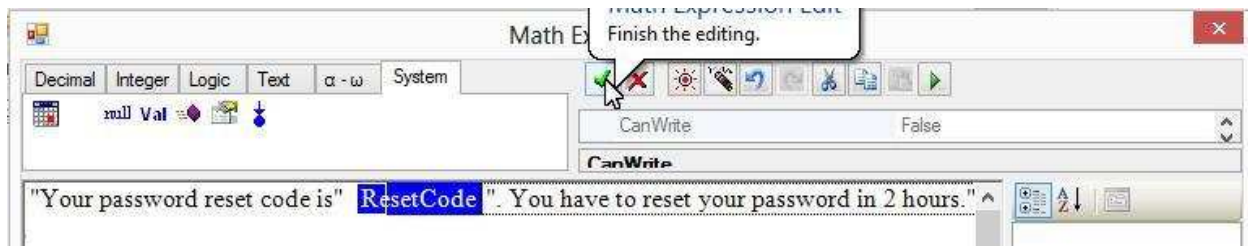
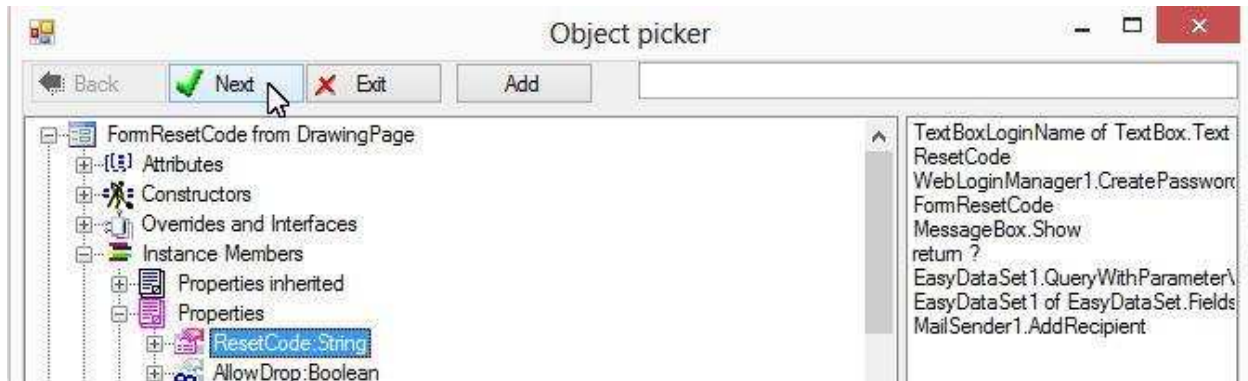
Click “A+” several times to create several string parts:



One part of the string is the reset code:



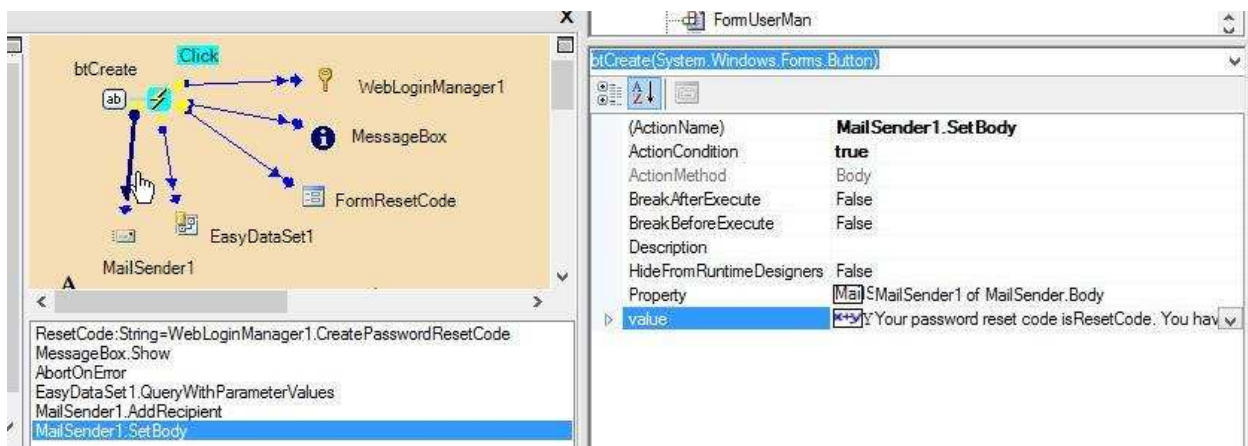




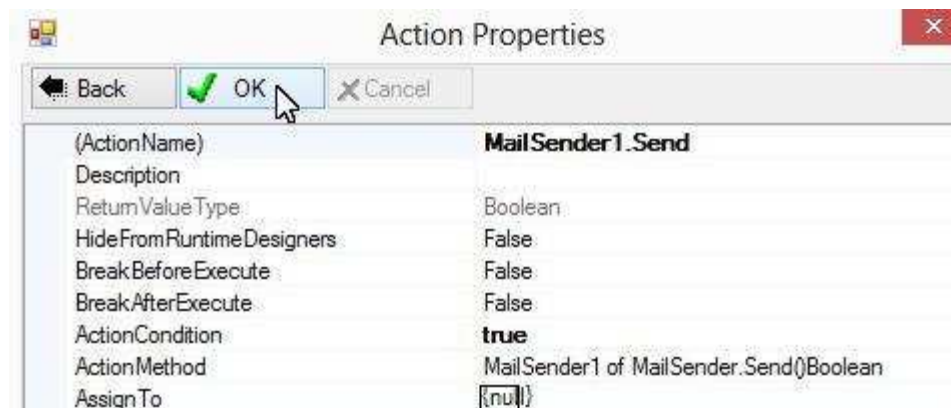
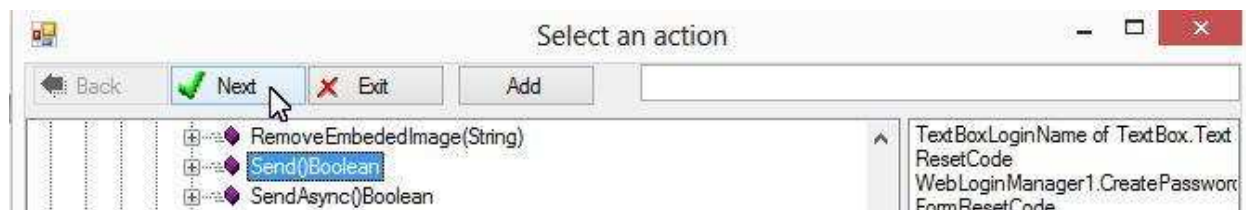
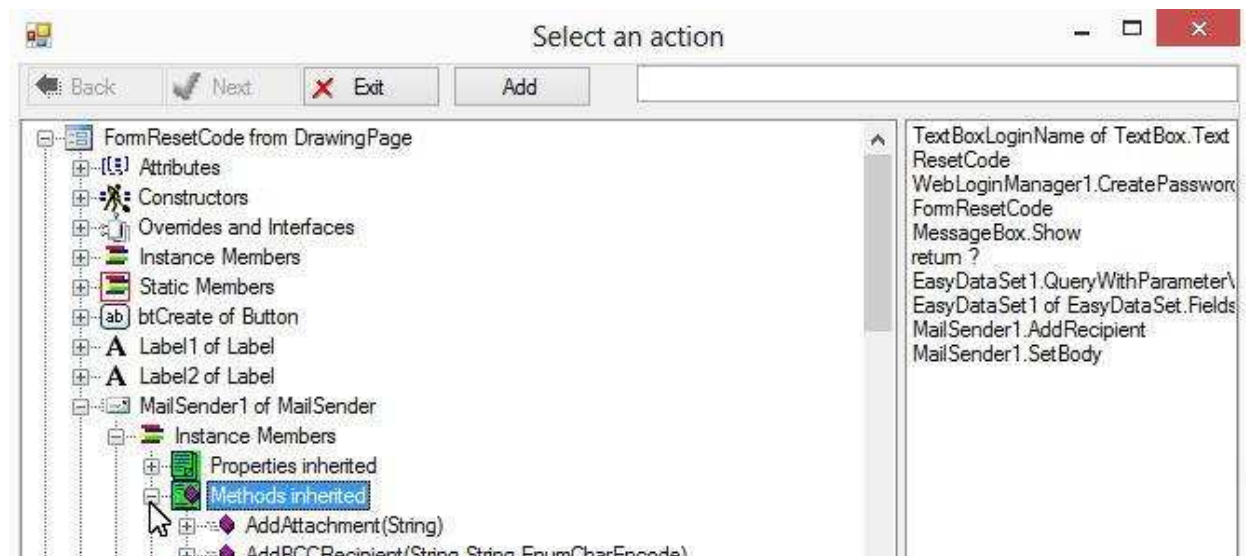
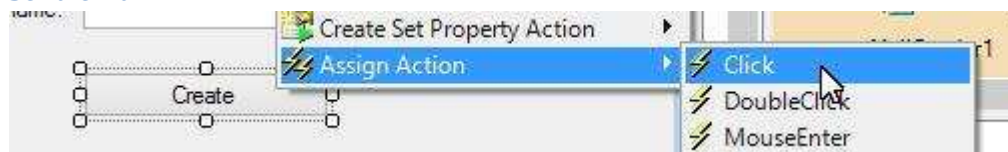
Click OK to finish creating this action:



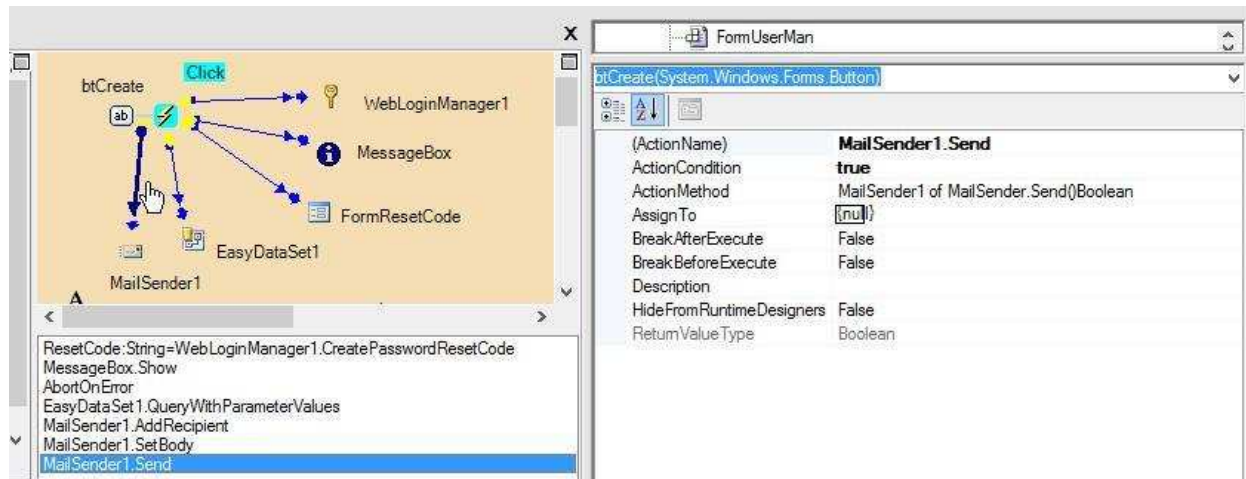
The action is created and assigned to the button:



## Send email

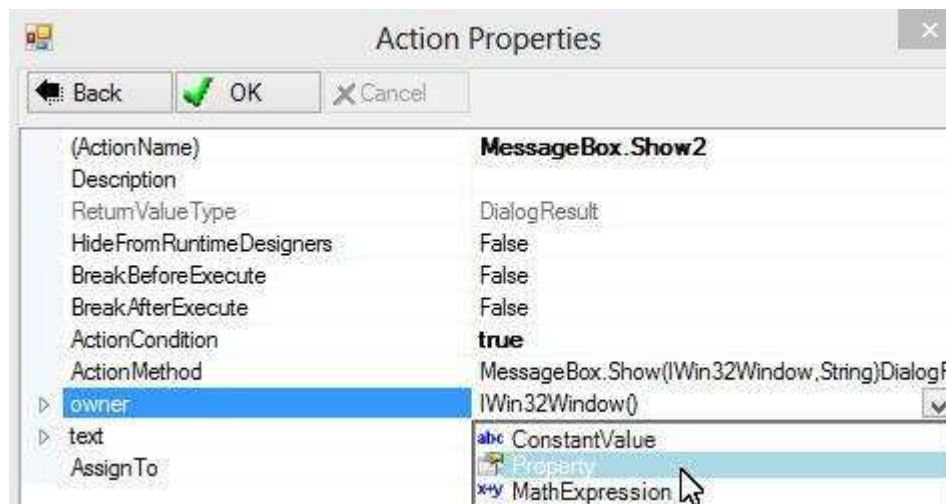
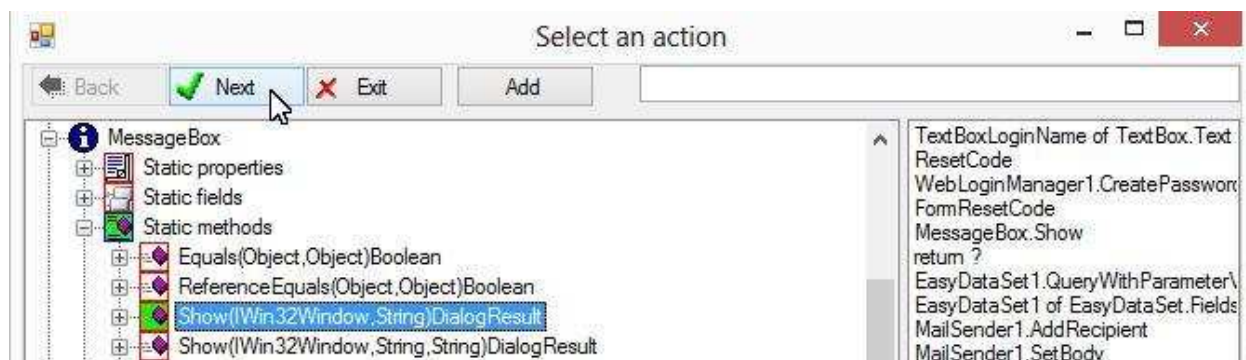


The action is created and assigned to the button:

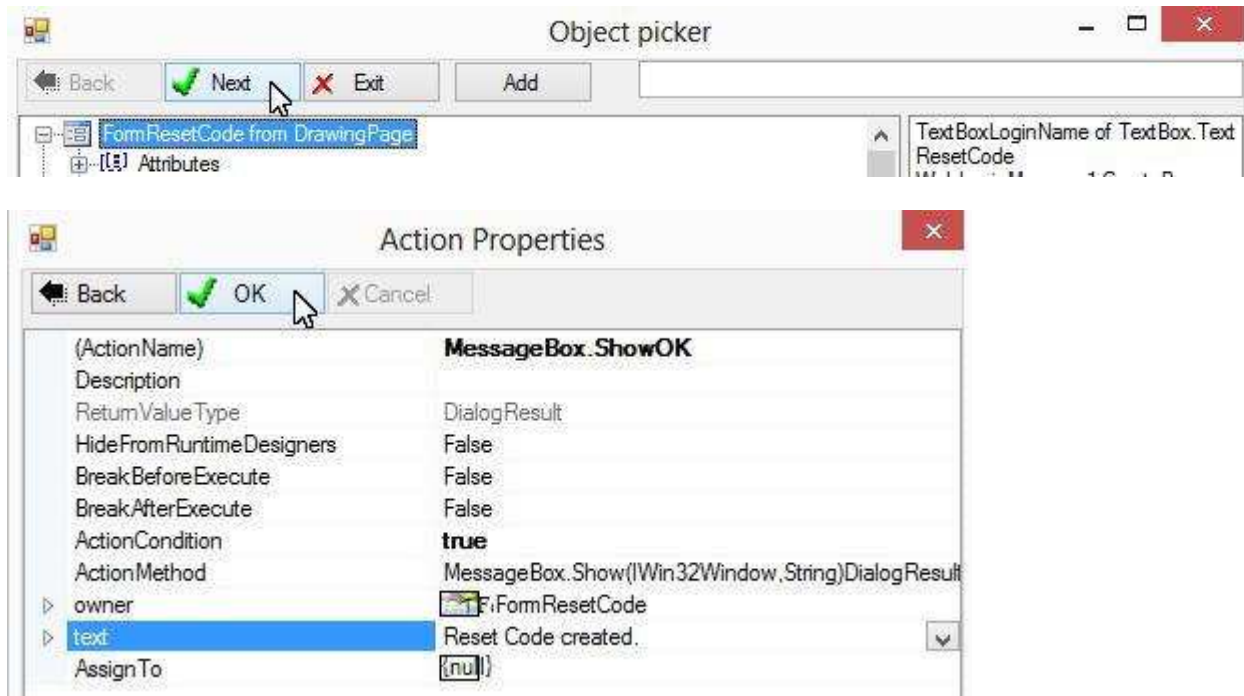


## Show notification

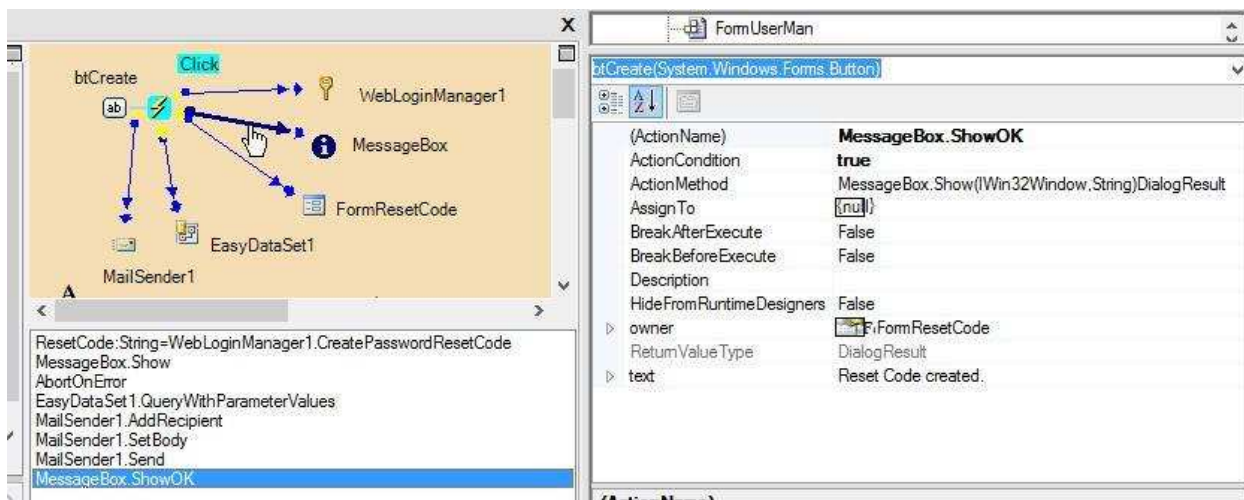
We may show a message box saying “Reset code created”:





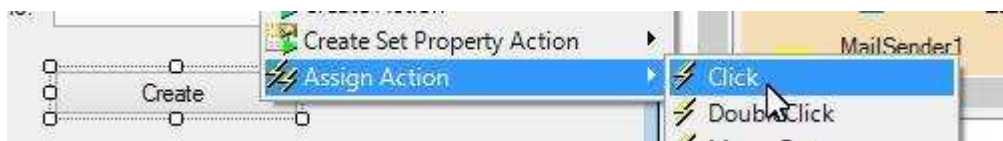


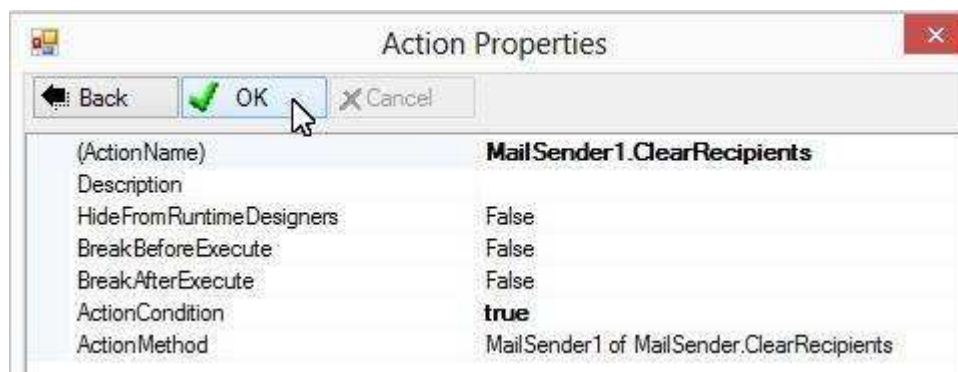
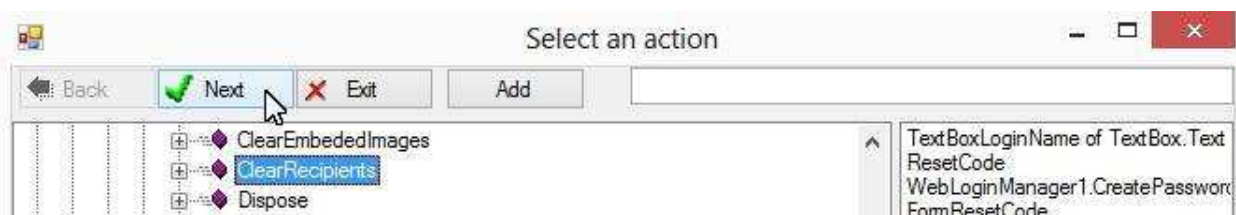
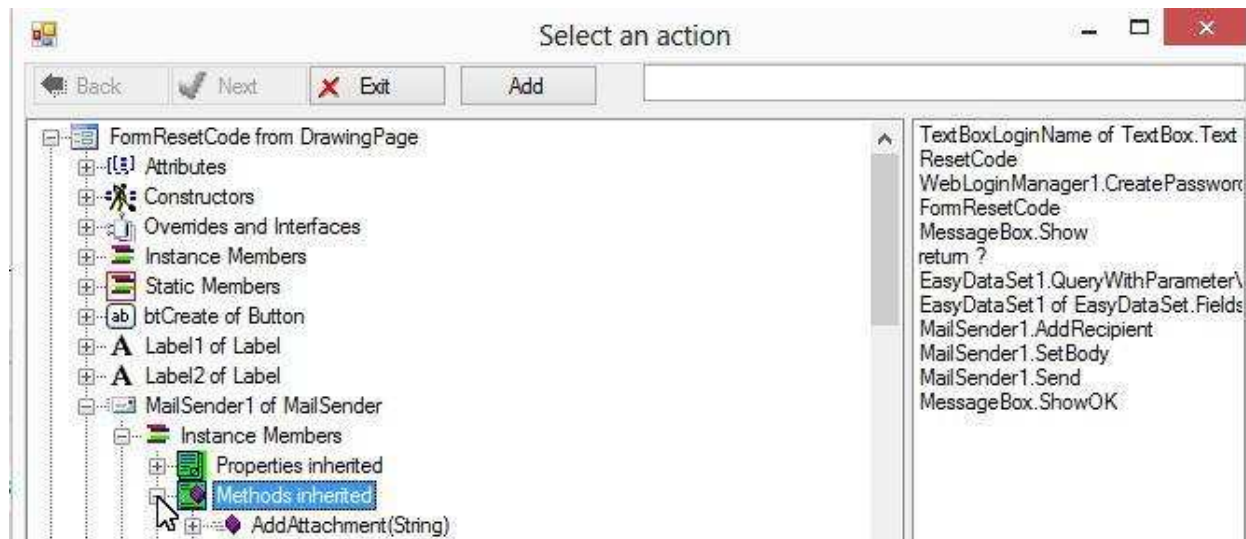
The action is created and assigned to the button:



## Remove email address

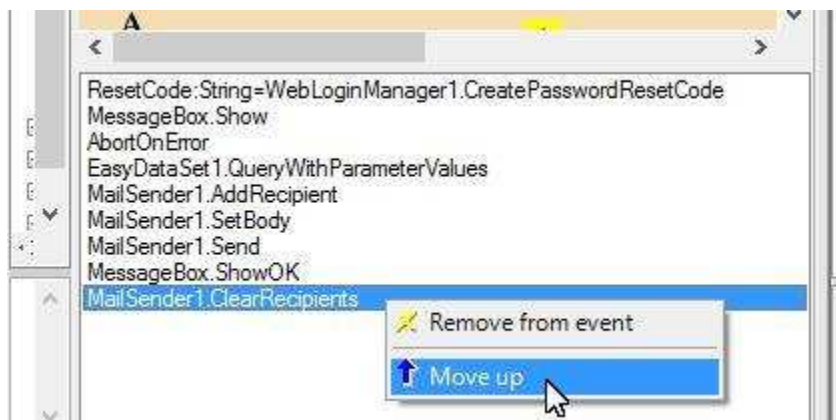
This is to prevent sending reset code to a wrong person.



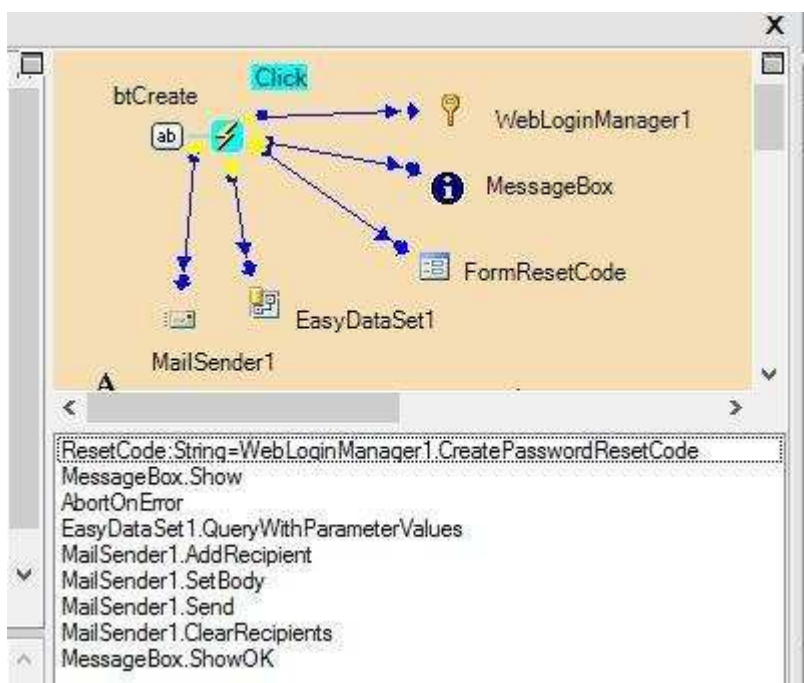


The action is created and assigned to the button. We may move it up to let it execute before the message box appears:





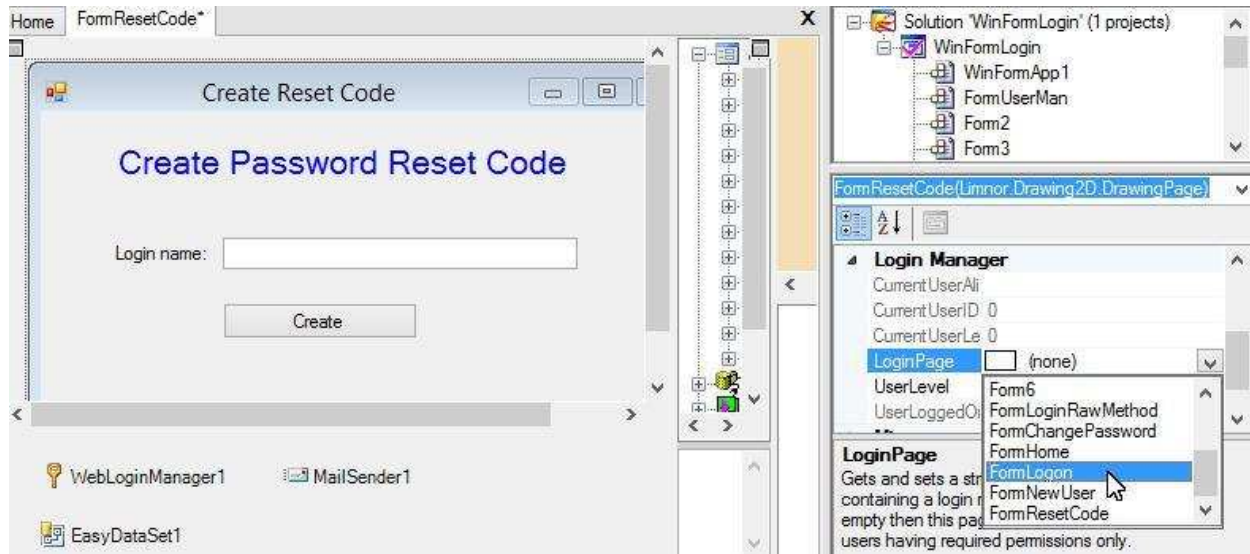
Our programming for the button is complete.



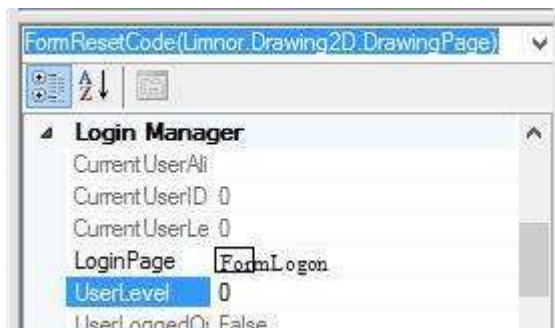
It executes a CreatePasswordResetCode action, and save the new reset code to property ResetCode; it shows an error message if ResetCode is empty; it aborts if the ResetCode is empty; it searches the database to get user email address; it sets the email address; it sets the reset code in email message; it sends the email; it removes email address; it shows a message box.

## *Protect the form*

Set LoginPage of this form:

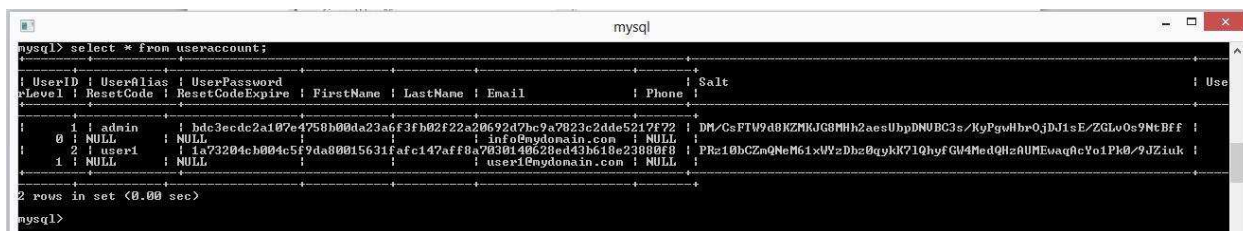


Keep its UserLevel 0 to require maximum permission for this form:



## Test

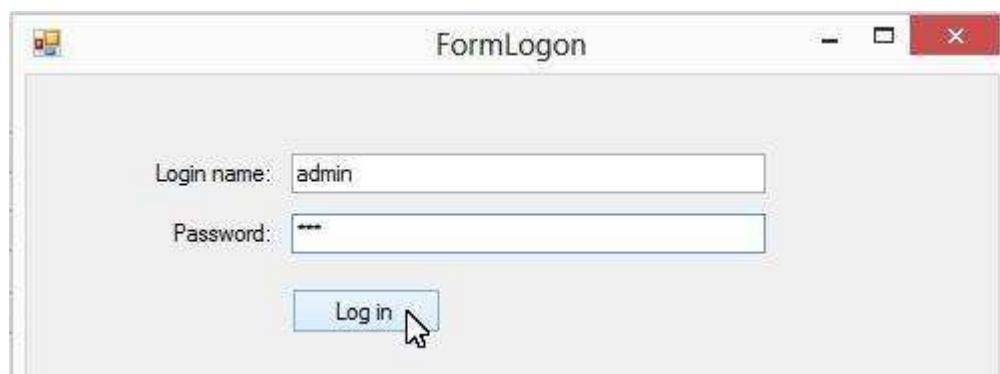
Let's create a new user with login name "user1" and a valid email address.



Suppose the user request to reset his/her password. Suppose user "admin" is going to create the password reset code for this user.



"admin" needs to log in to access the form:



The form appears. Enter login name "user1" and click "Create":



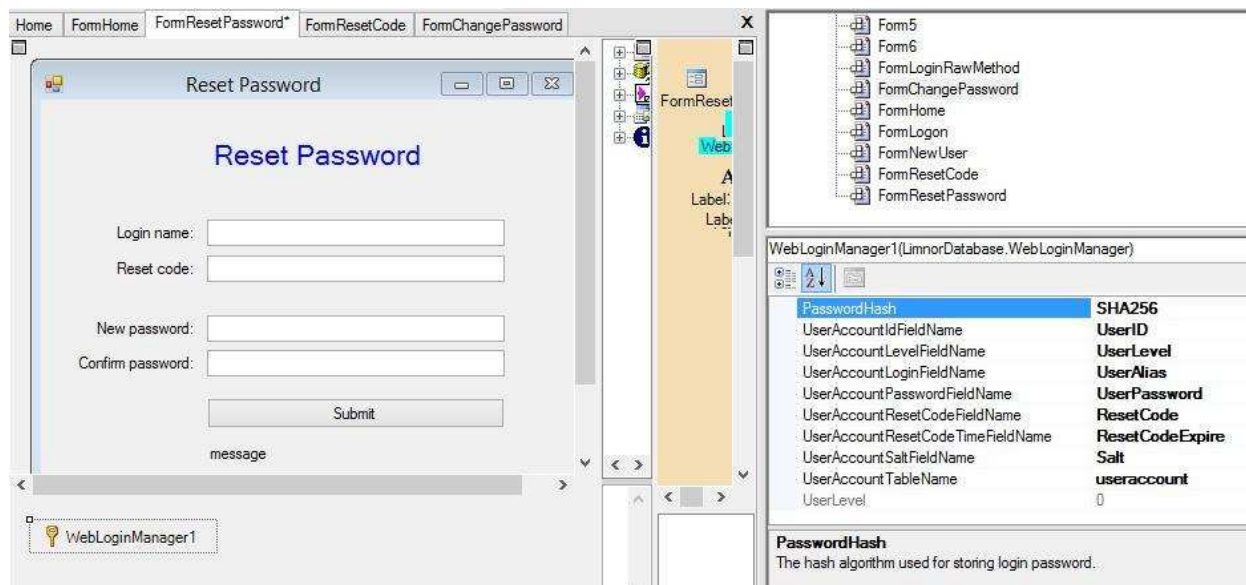
A message box appears:



“user1” will get the reset code via email, and use the reset code to change his/her email. That will be done in another form as described below.

## Reset password – set new password

We create a new form to allow users to do it. The form uses a text box to allow the user to enter password reset code. It allows the user to enter new password.



Note that the properties of the WebLoginManager must be set to the same as we did for the login form.

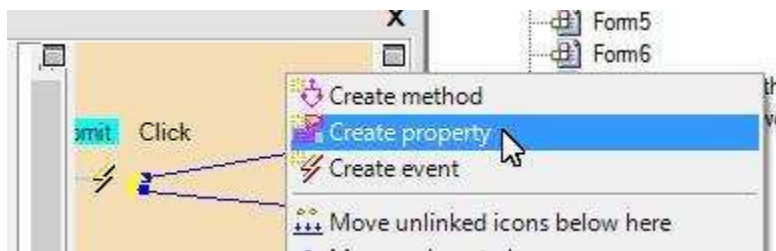
## Verify inputs

The new password cannot be empty and must match confirm password. If the verification fails, we display a message box and abort. We did such programming before. See “Verify input” for “Change password” section.

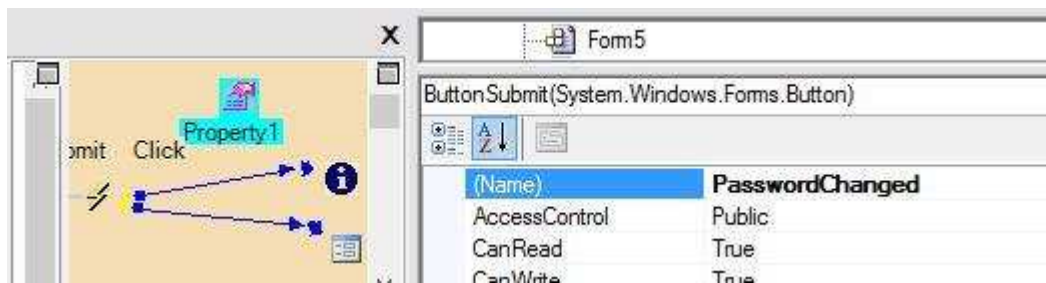


## Reset password

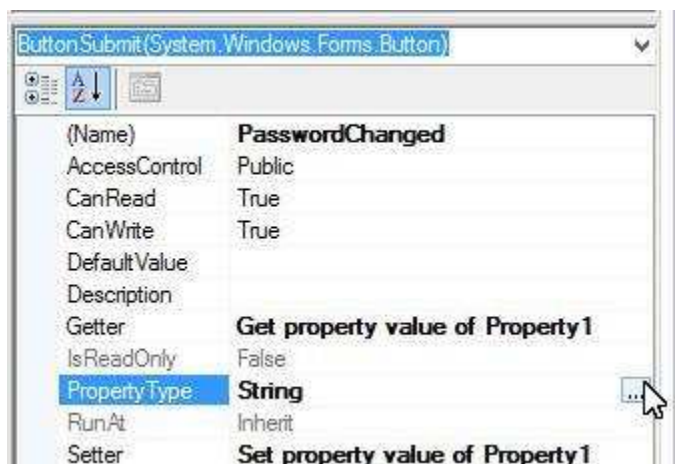
We use a property to indicate whether password reset is successful:



Rename the new property to “PasswordChanged”:



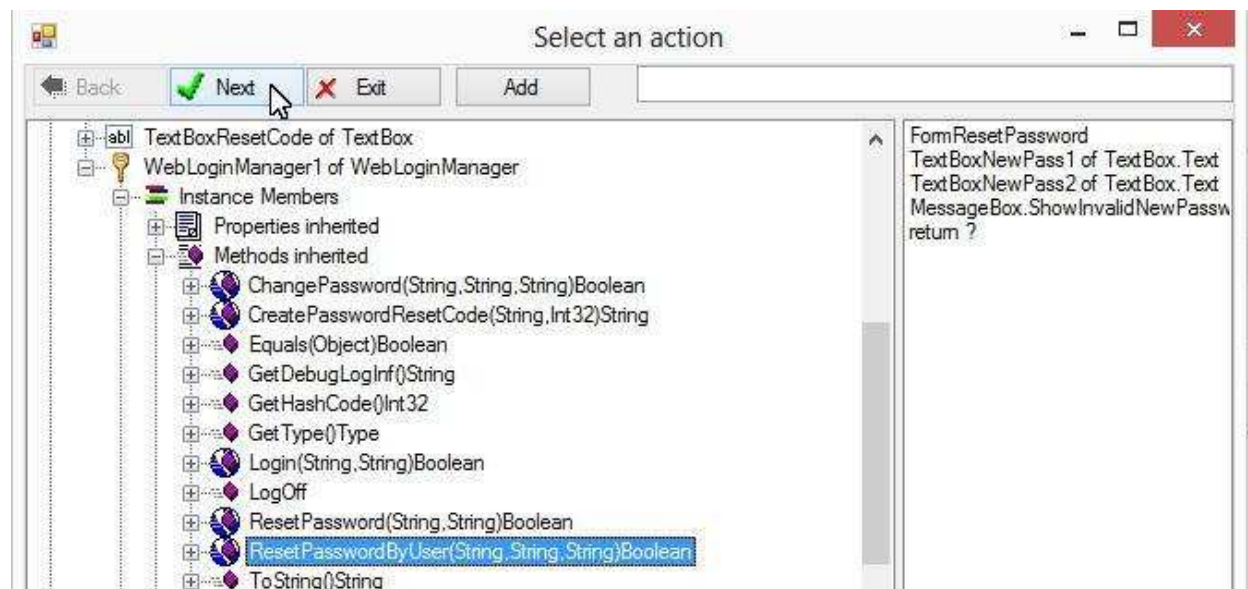
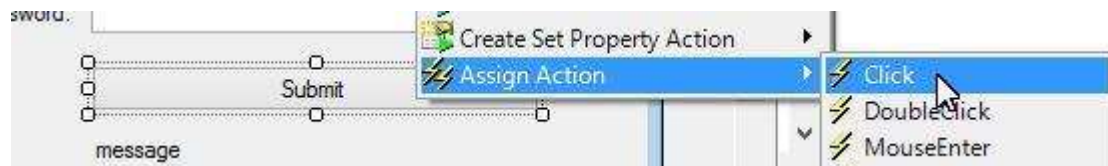
Change property type to Boolean:



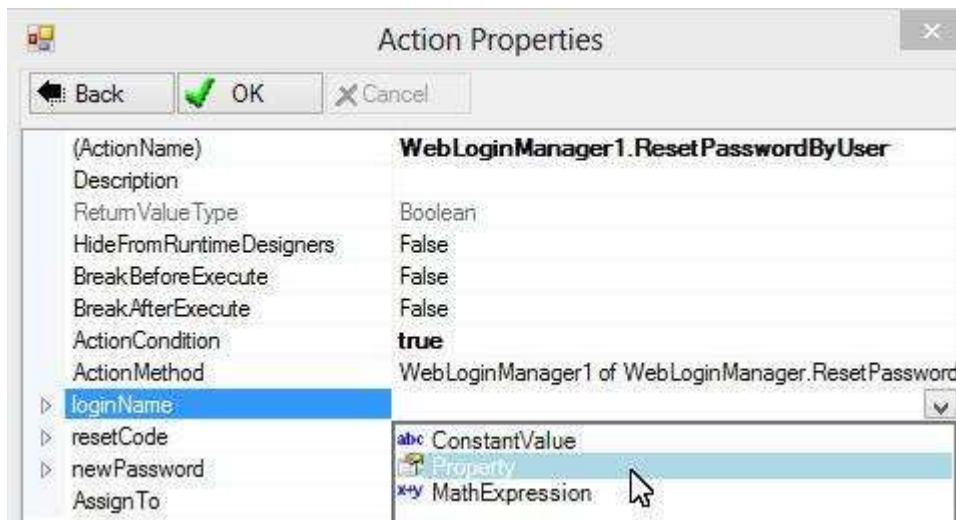


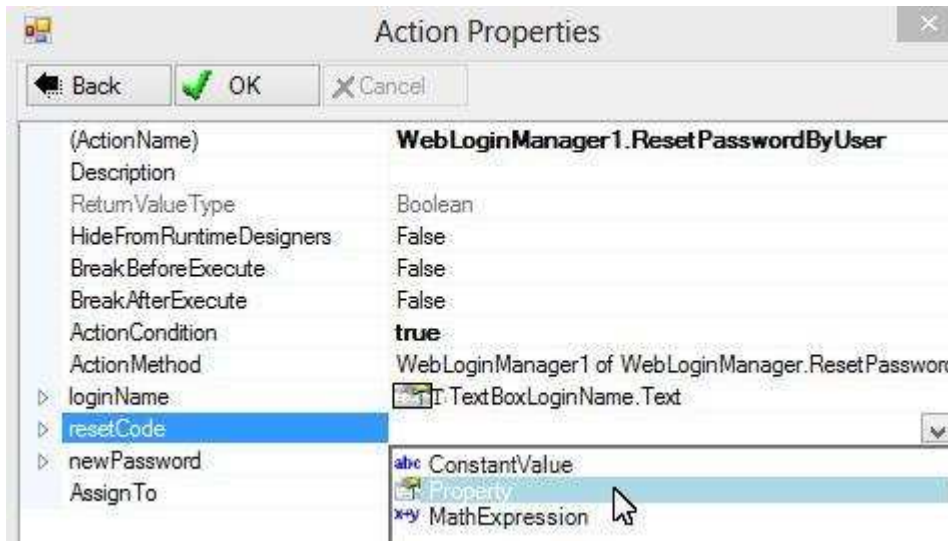


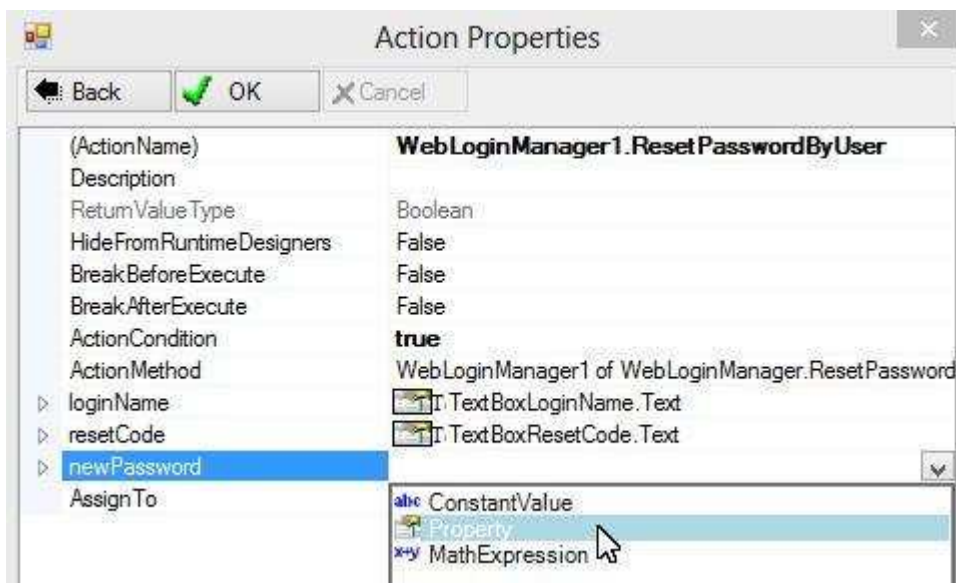
Now we may create password reset action.



Pass the user inputs to the action:

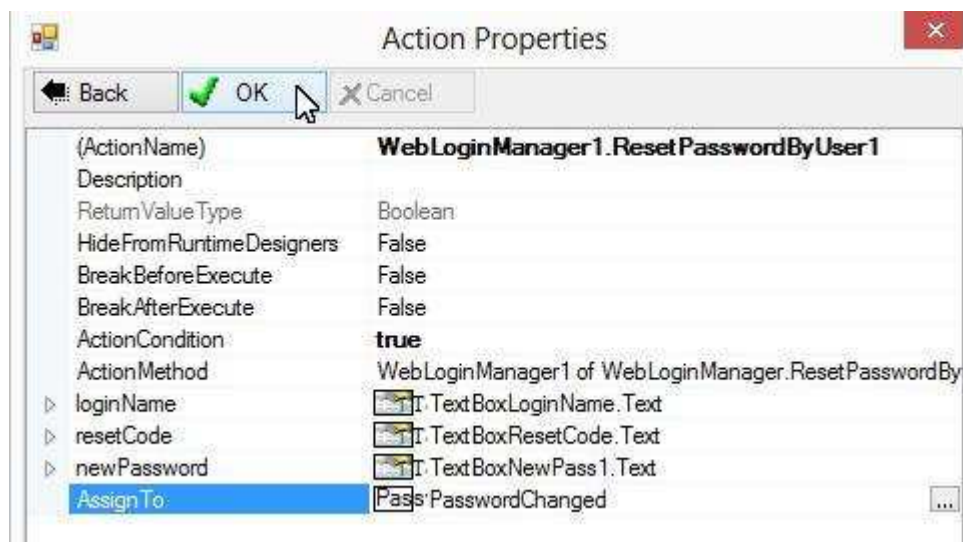
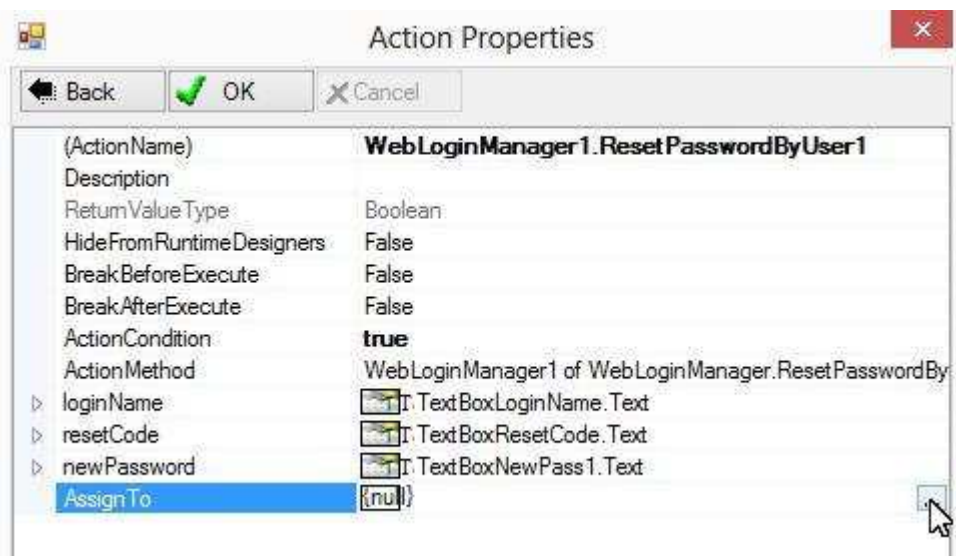




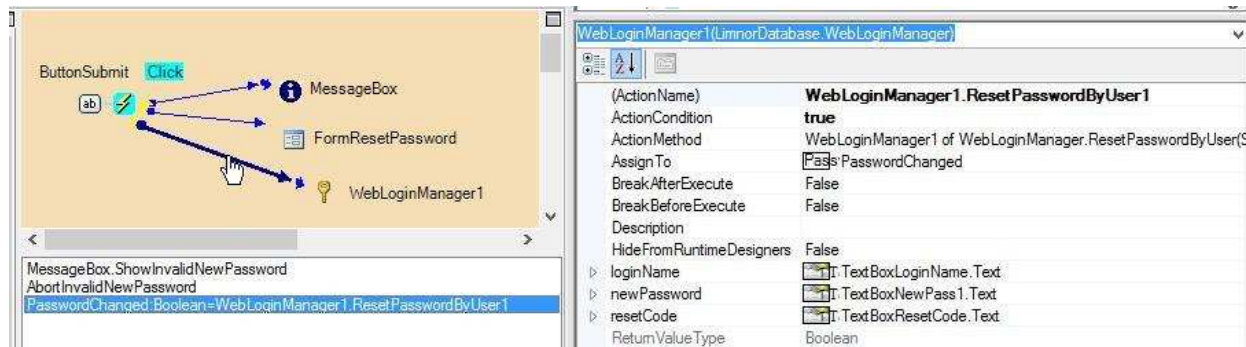


Set "AssignTo" to property PasswordChanged:



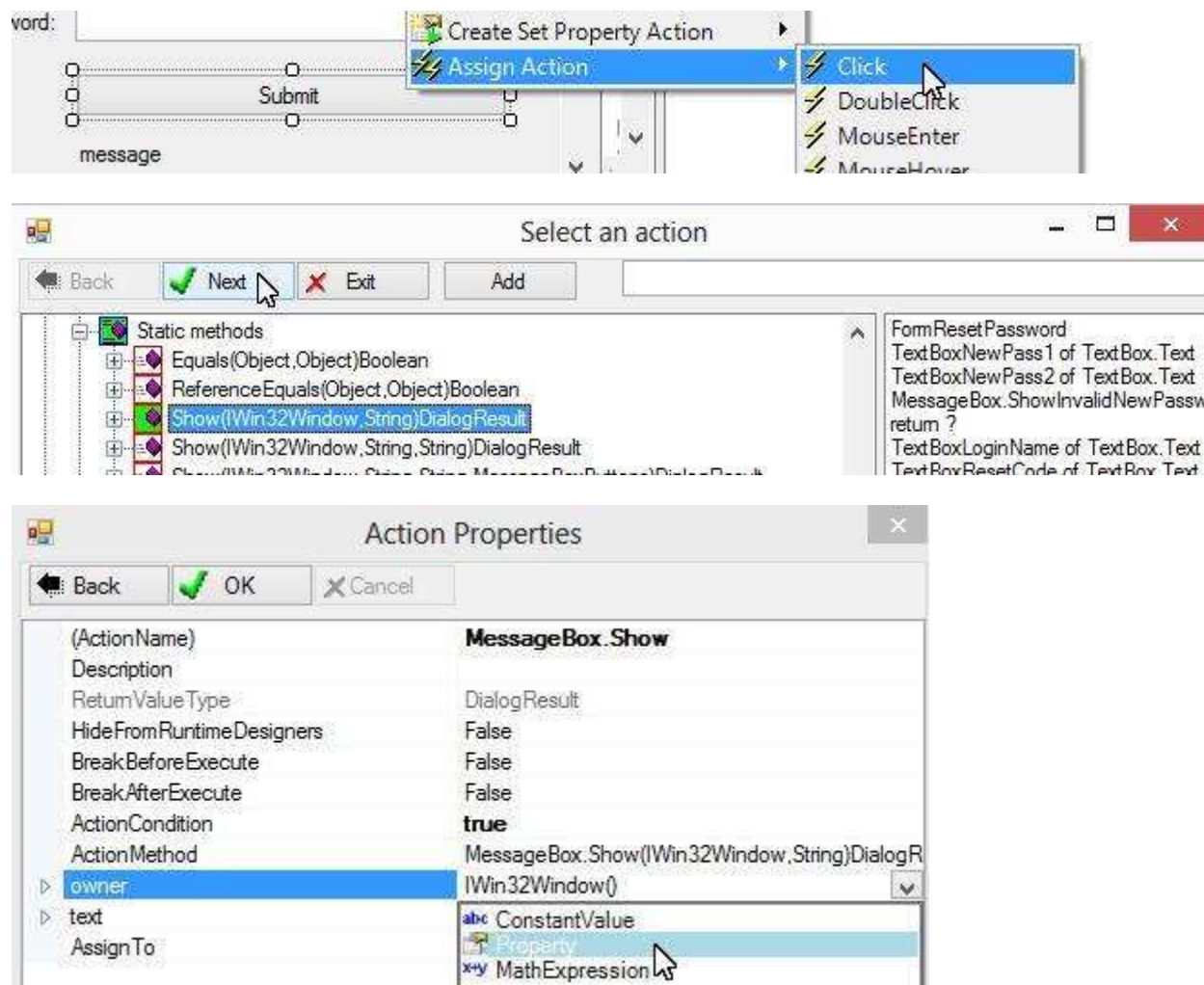


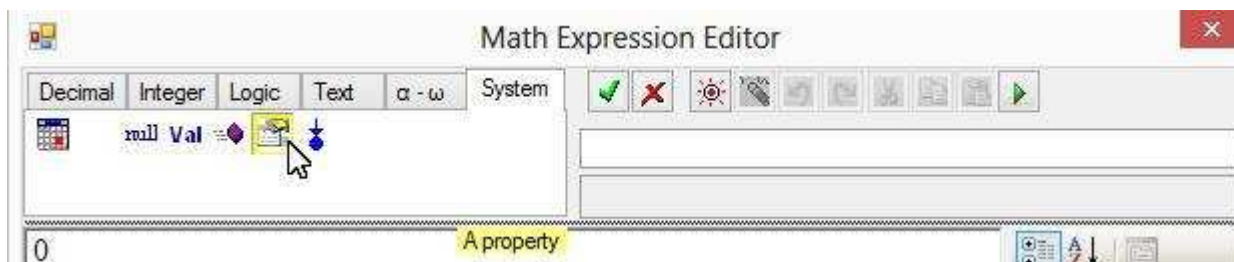
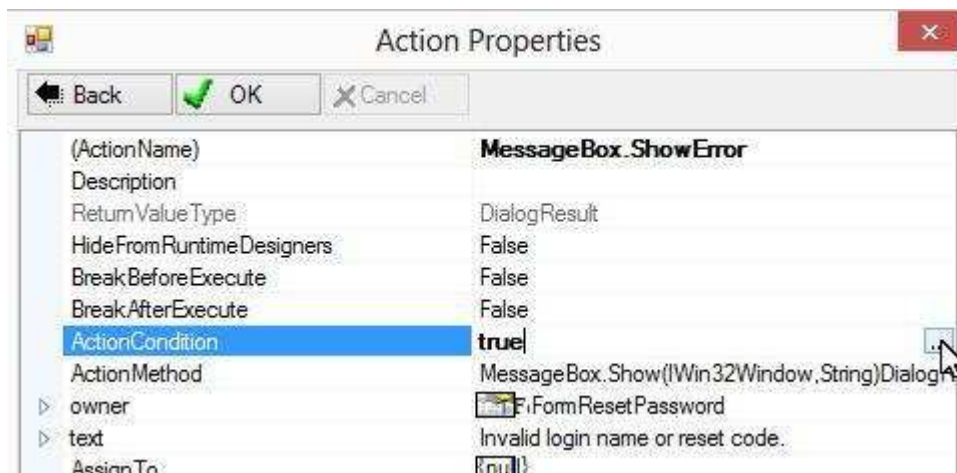
The action is created and assigned to the button:

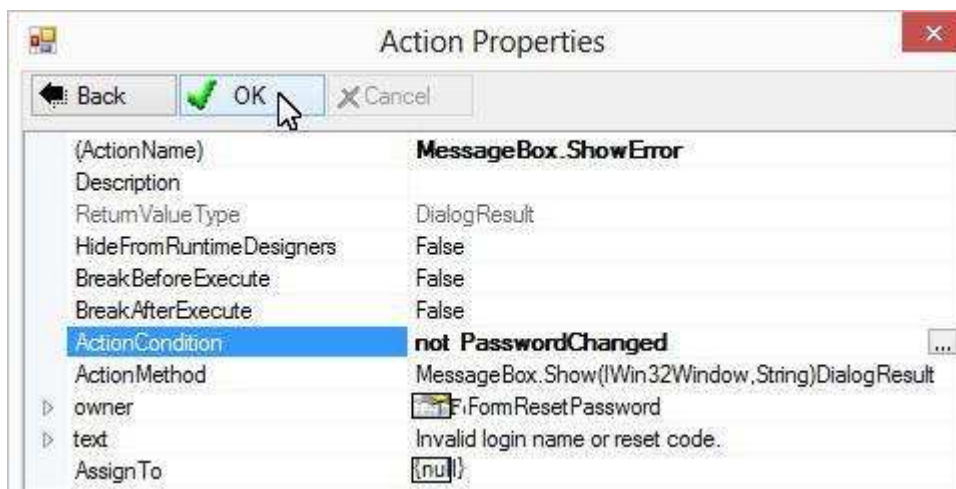
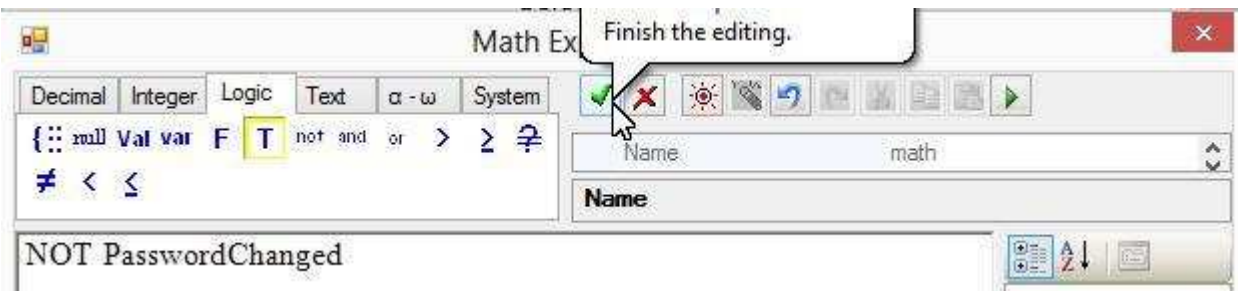
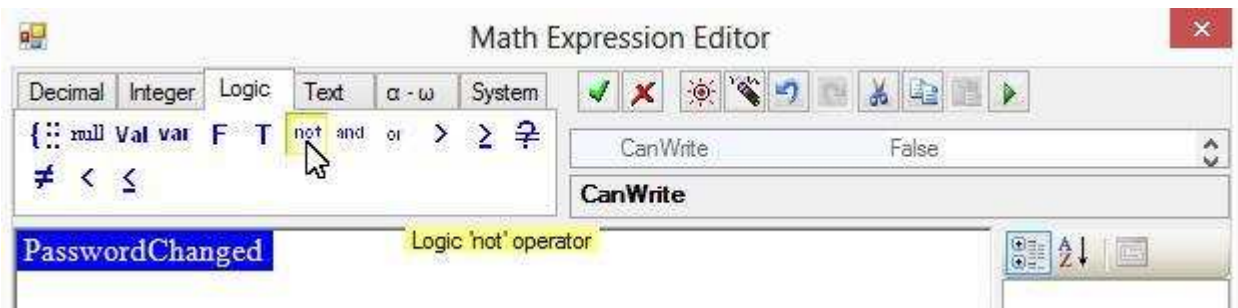


## Show error message

If PasswordChange is false then we want to show an error message:

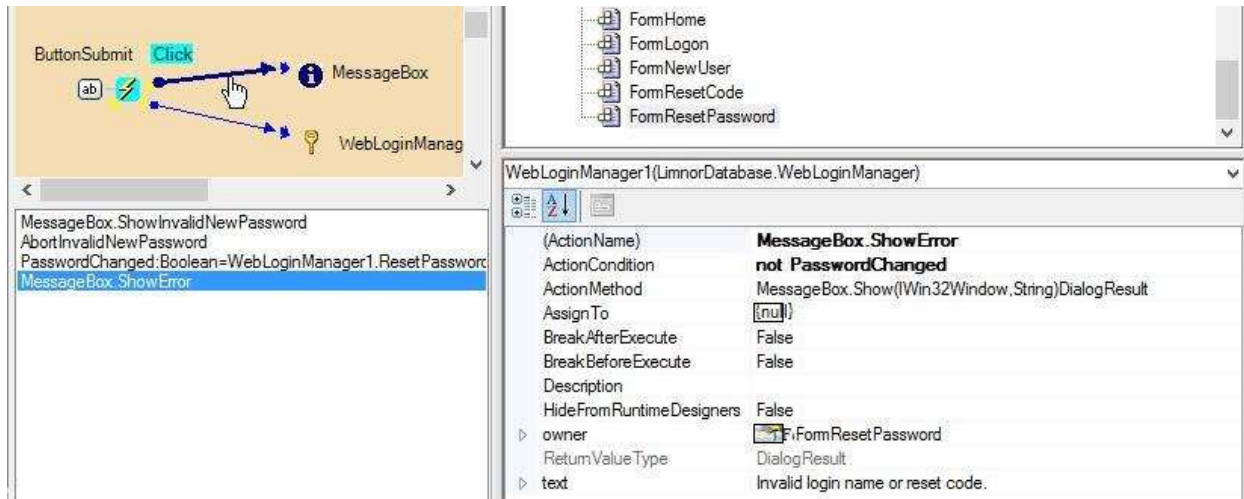






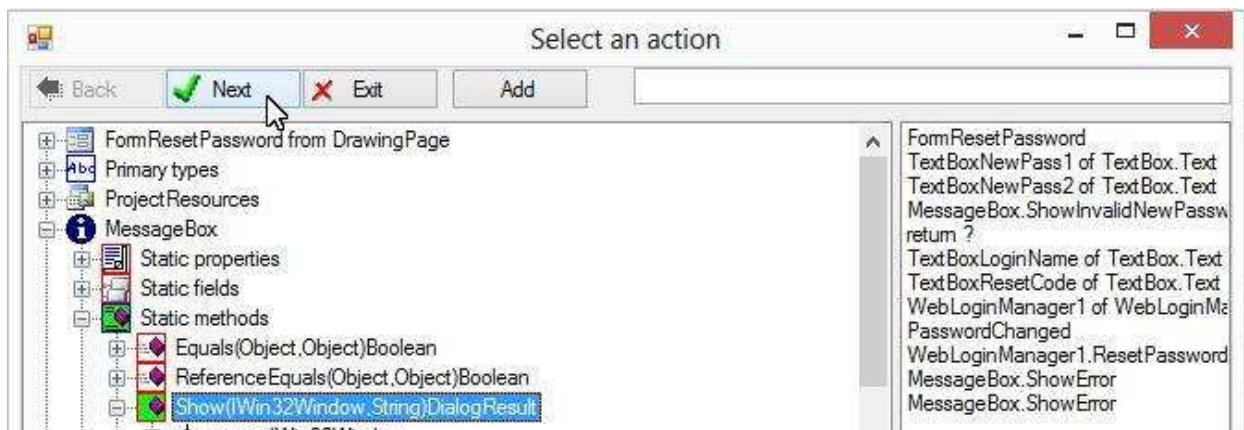
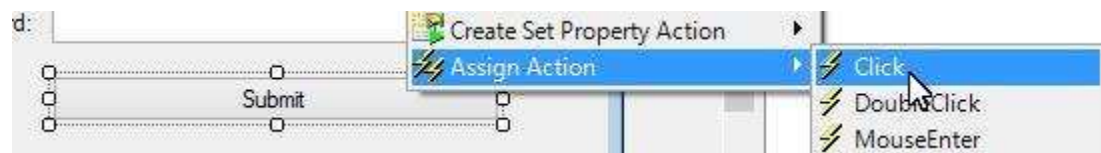
The action is created and assigned to the button:

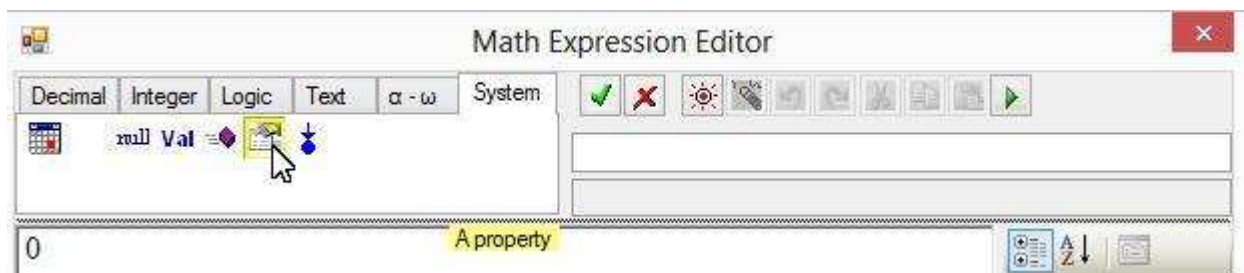
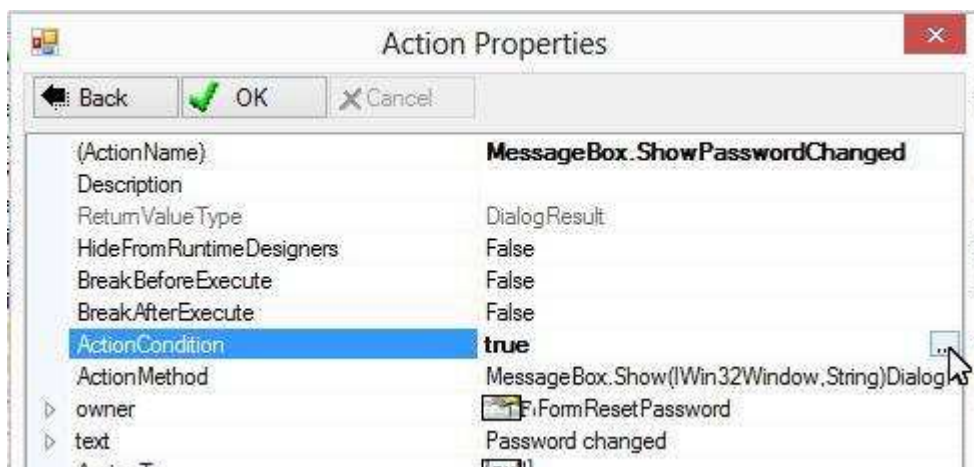
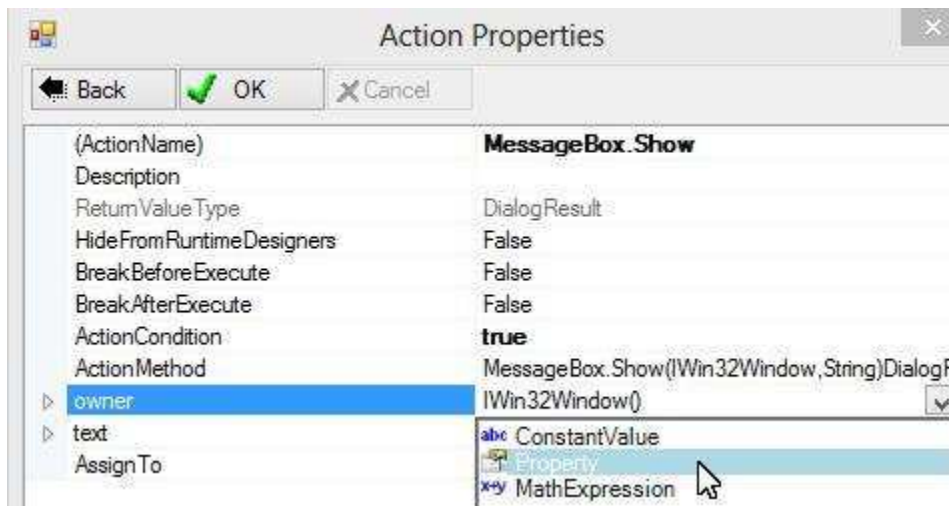


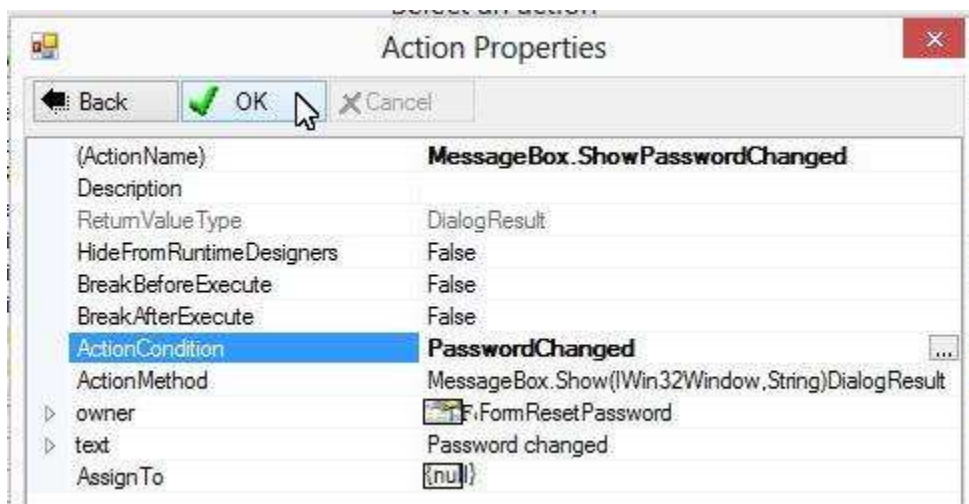
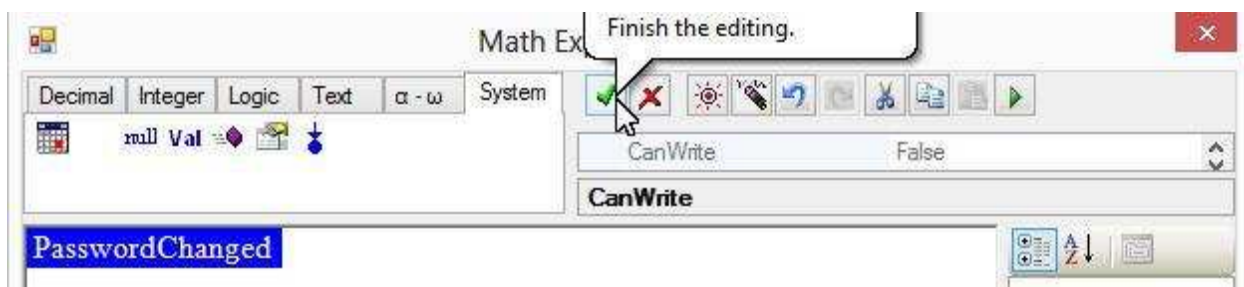
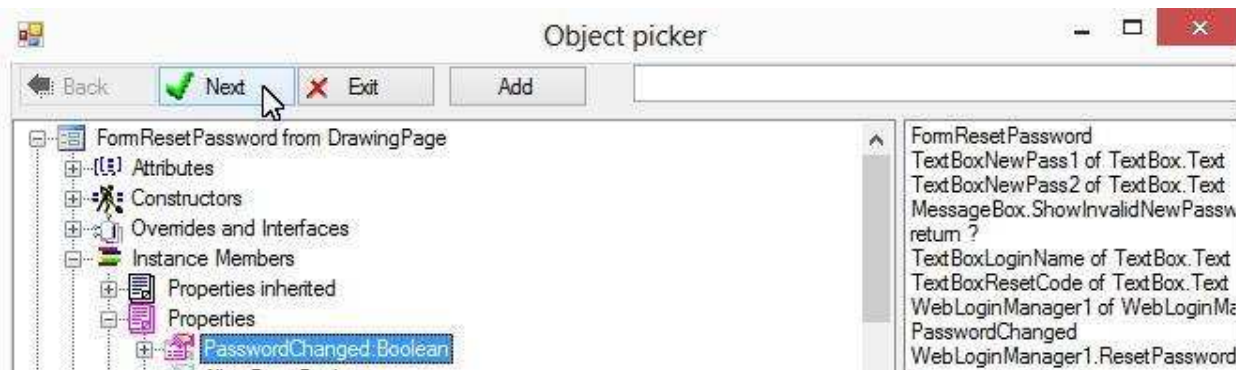


## Show success

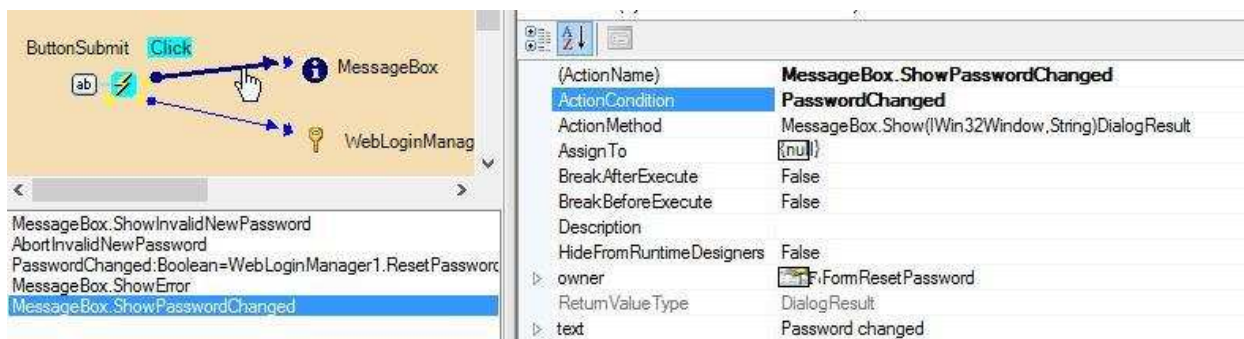
If PasswordChanged is true then we want to display a message box saying “Password changed”.





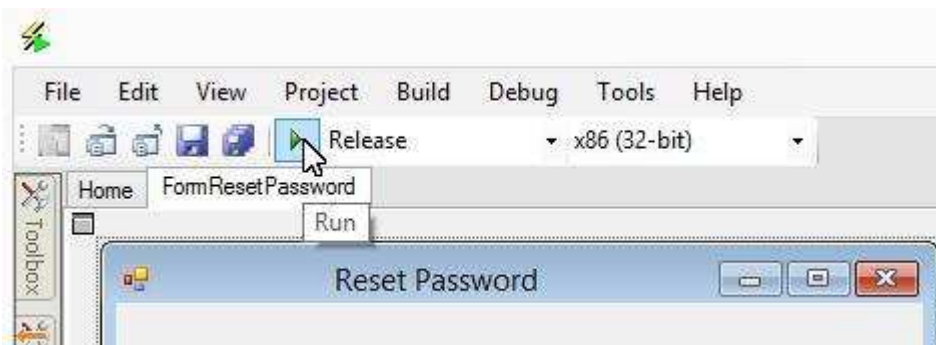


The action is created and assigned to the button:

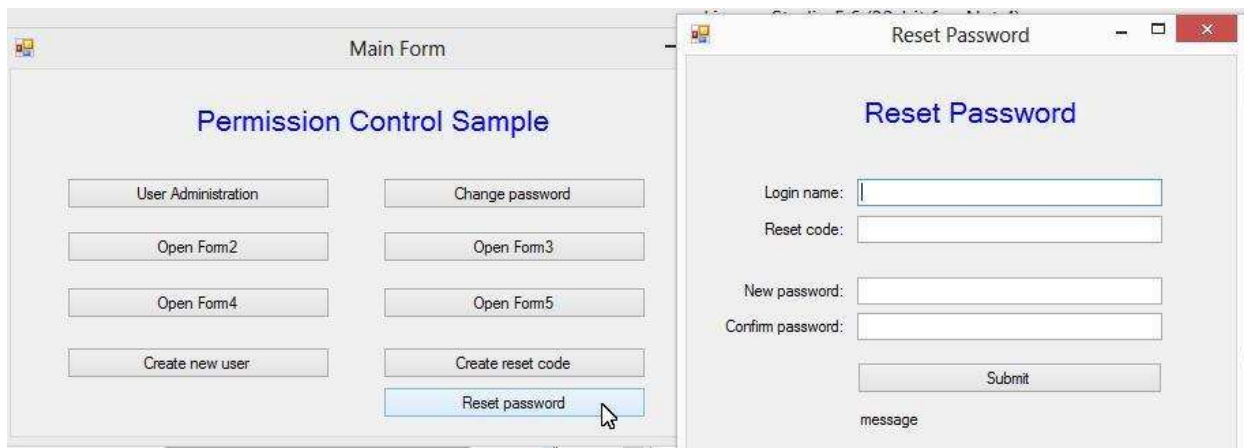


## Test

Let's use the reset code, "X2qL6wP2", for "user1" to change password for "user1".



Click "Reset password". The form appears because it is not protected:



Enter login name, reset code, and new password. Click Submit:





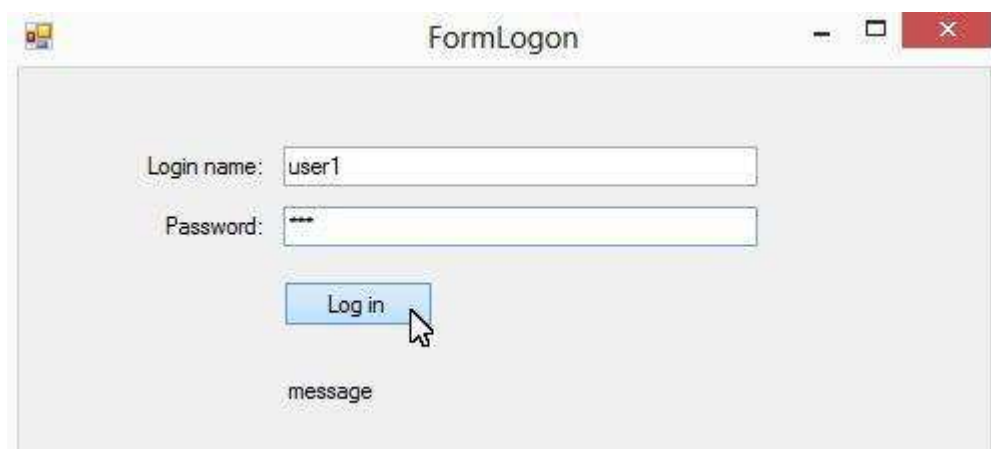
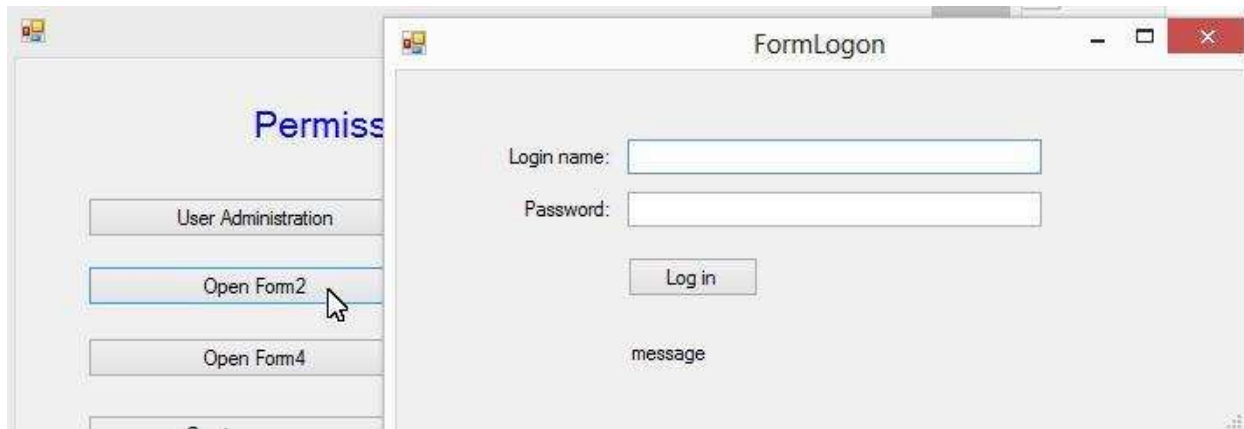
The screenshot shows a Windows Forms application window titled "Reset Password". The window has a light gray background and a title bar with standard Windows window controls. The main content area contains the following elements:

- A title "Reset Password" in a large, bold, blue font.
- A "Login name:" label followed by a text box containing "user1".
- A "Reset code:" label followed by a text box containing "\*\*\*\*\*".
- A "New password:" label followed by a text box containing "\*\*\*".
- A "Confirm password:" label followed by a text box containing "\*\*\*".
- A blue "Submit" button.
- A "message" label below the Submit button.



This screenshot shows the same "Reset Password" window as the previous one, but with a small dialog box overlaid on top. The dialog box is titled "Password changed" and contains an "OK" button. The "Submit" button in the background window is now highlighted by a mouse cursor. The "message" label is still visible below the Submit button.

We may try to use the new password to access protected forms:



The form does not appear. We get a “Permission denied” message. It indicates that the new password works:

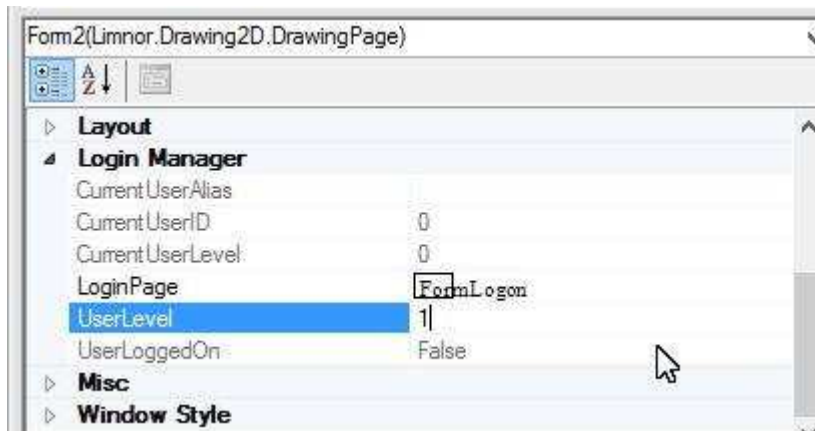


We get “Permission denied” because the UserLevel of Form2 is 0 and the user level for “user1” is 1.

## Permission Level

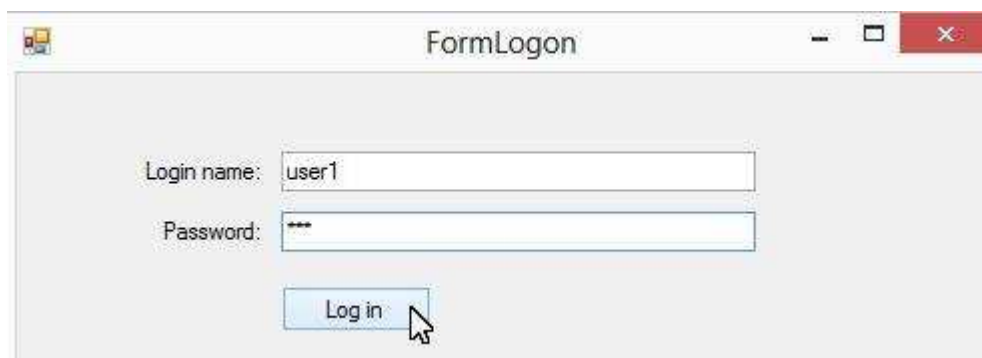
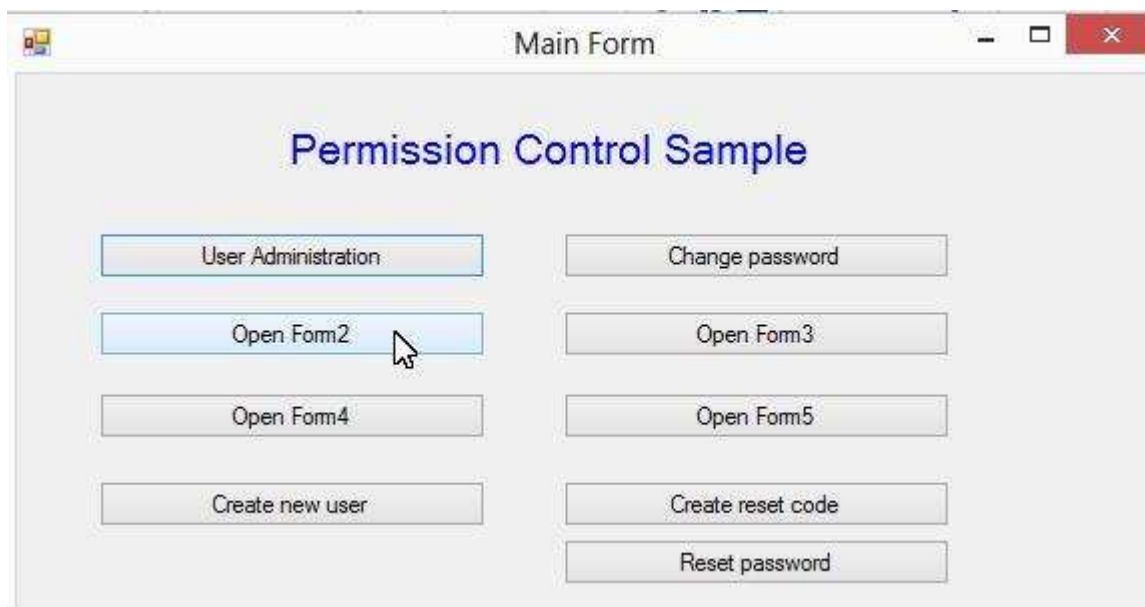
We set permission level of “user1” to 1; therefore, “user1” cannot access a form with UserLevel 0.

Suppose we want Form2 and Form3 to be accessible by users of level 1, we need to set UserLevel of Form2 and Form3 to be 1:

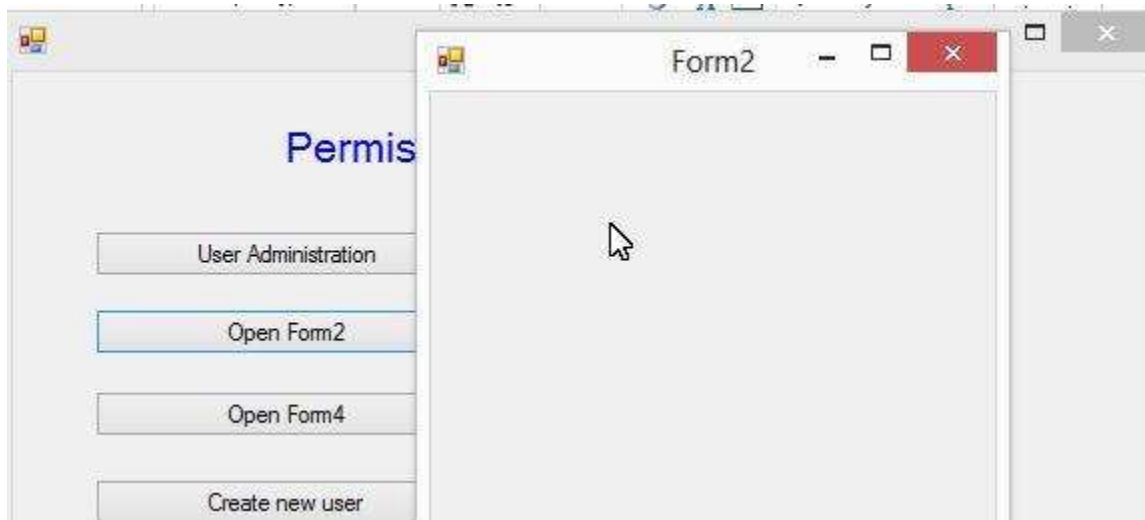


Do the same for **Form3**.

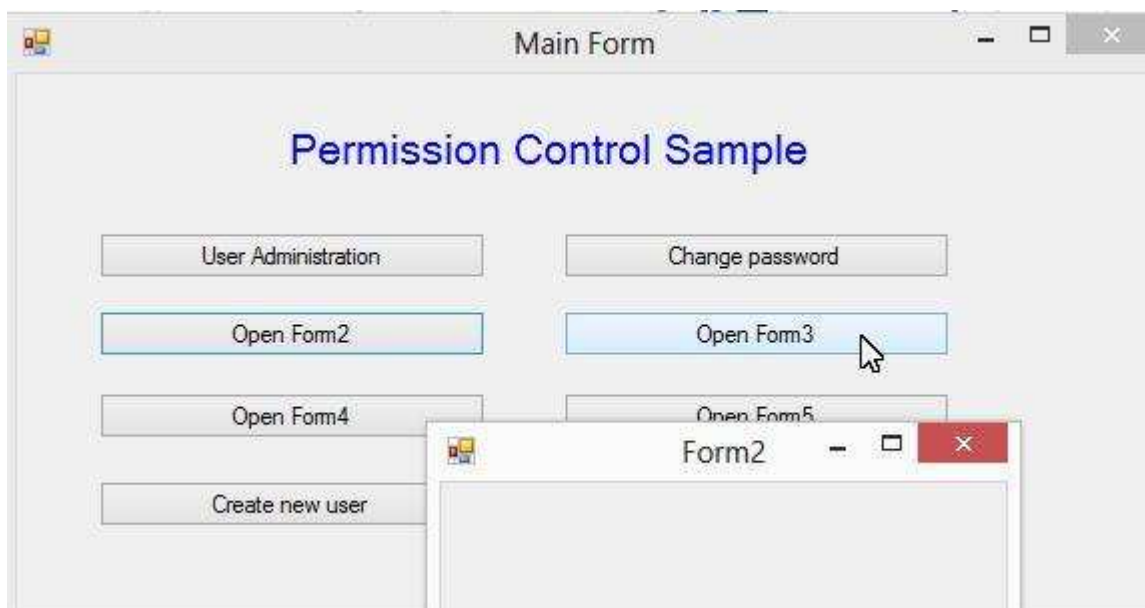
Now let's access Form2 with user "user1":



Form2 appears:

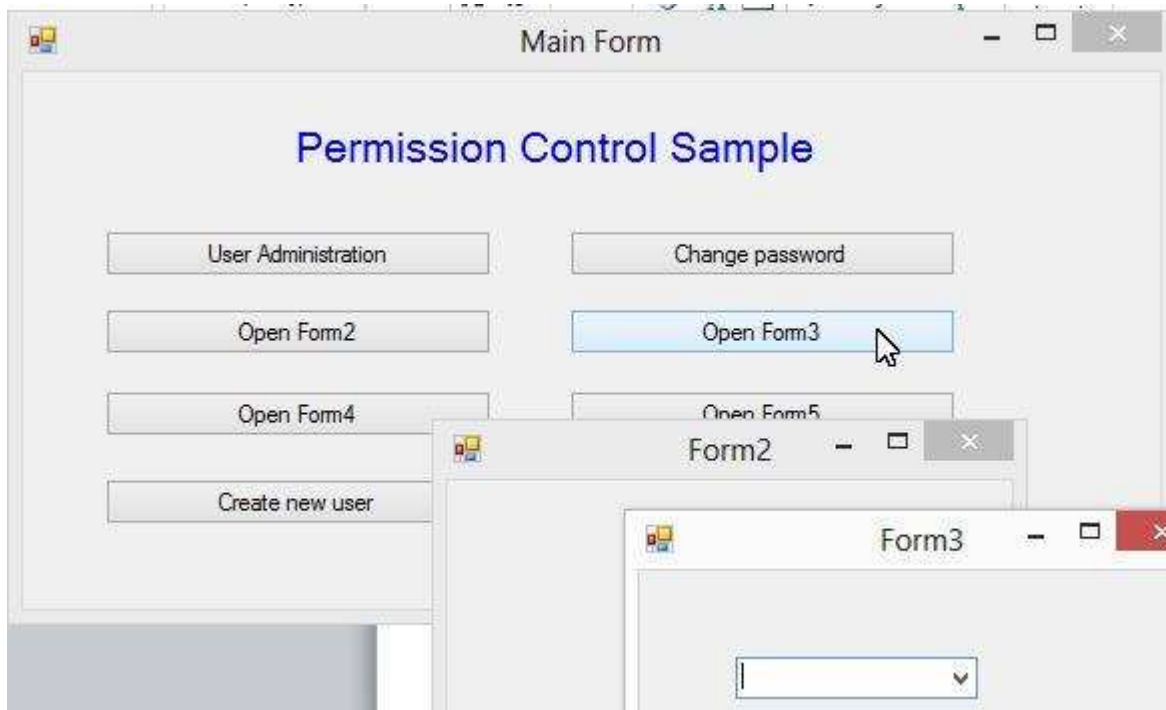


Click “Open Form3”:

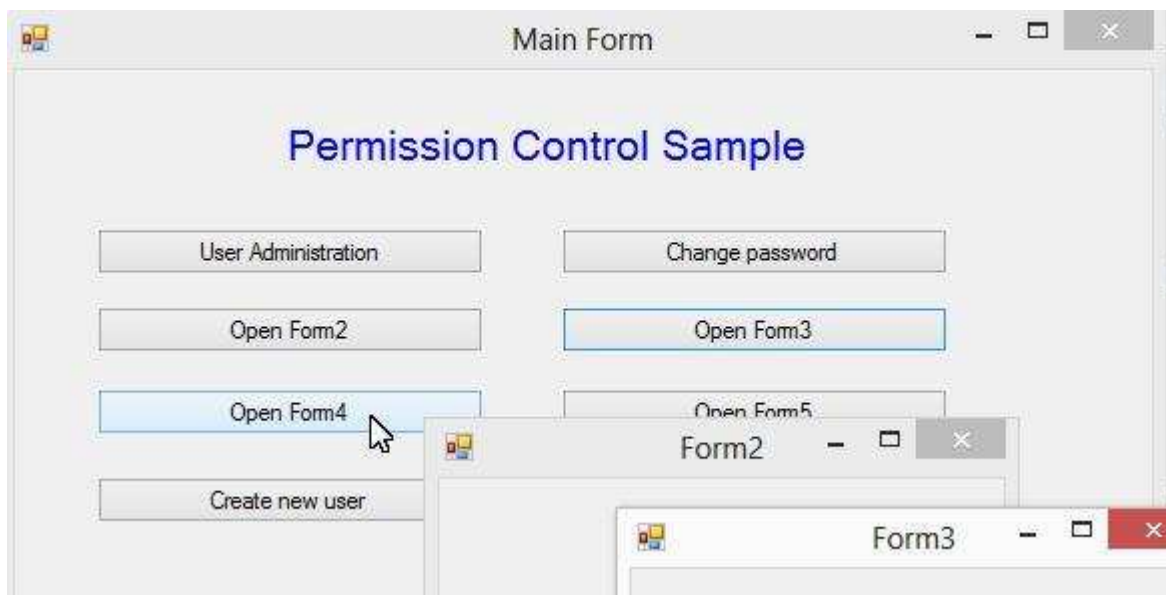


Form3 appears:

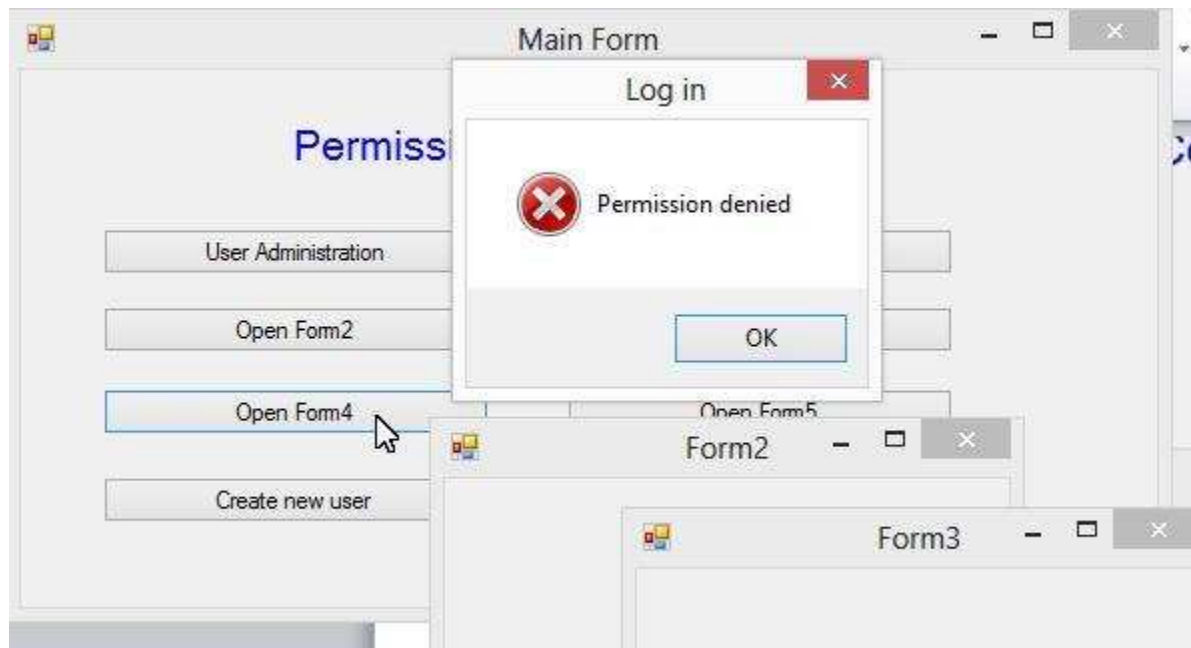




Click "Open Form4":



We get "Permission denied":



### Custom Permission Control

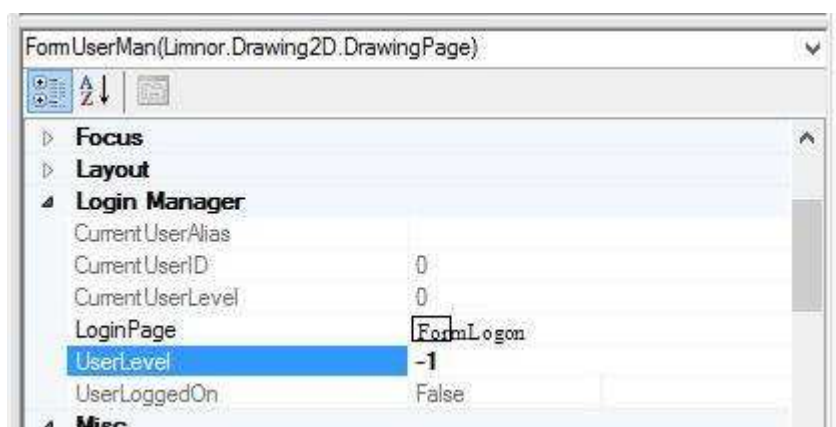
We have shown how the user level works for permission control. If this built in permission control does not meet your business requirements then you may add your own permission controls. We use a sample to show one way of doing it.

Suppose we have a form for modifying user records. We do not want to use user level to control permissions. We want to only allow specific users to use it. For this purpose, we add a Boolean field in the user table. The field name is `CanModifyUsers`. If this field is true for a user then the user may open the form; if this field is NULL or false then the user cannot open the form.



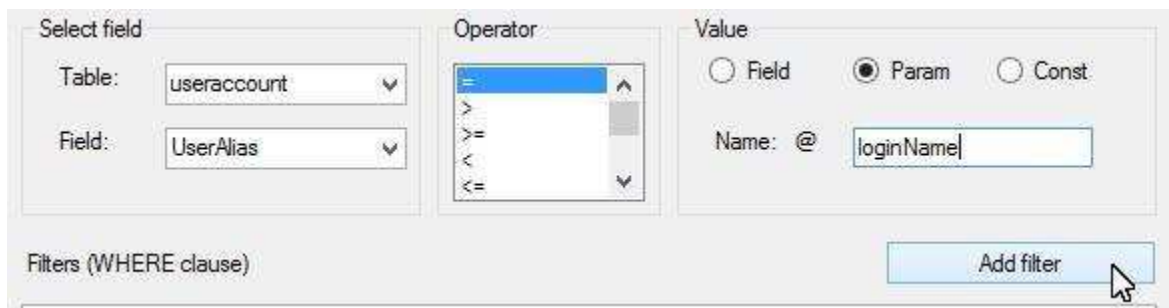
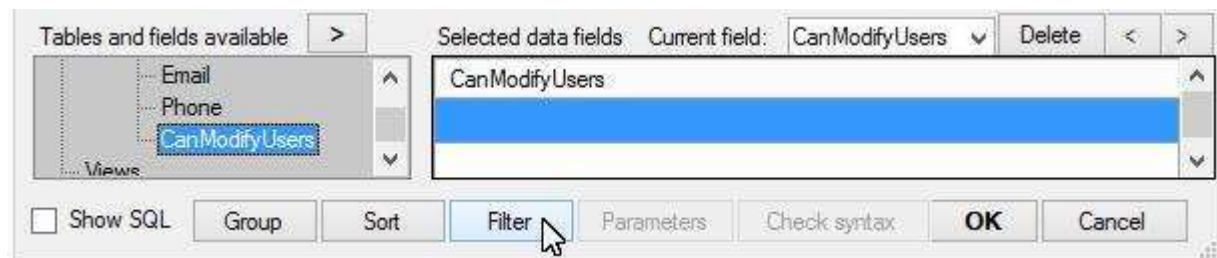
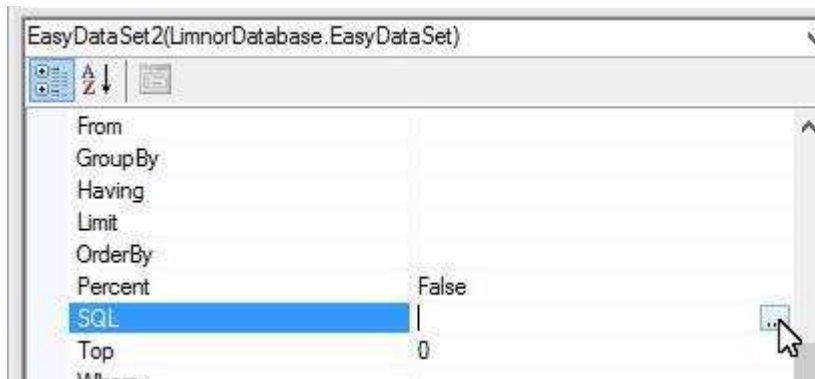
### Disable user level

Set UserLevel to -1 so that user level checking will not be performed by the Login Management Framework:

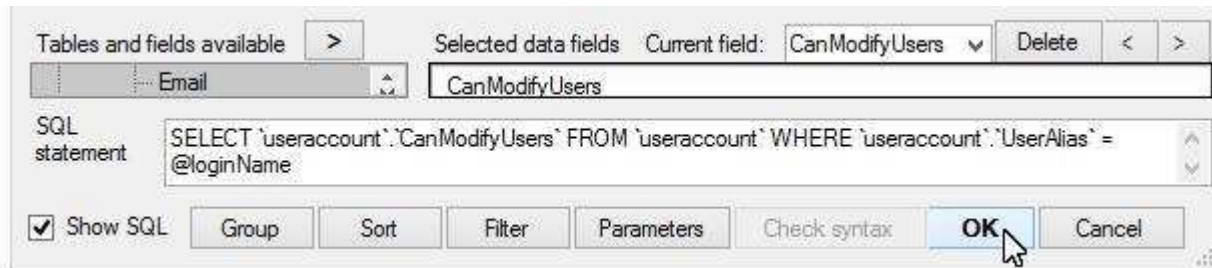


### Check permission

We use an EasyDataSet to read field CanModifyUsers to check the permission.



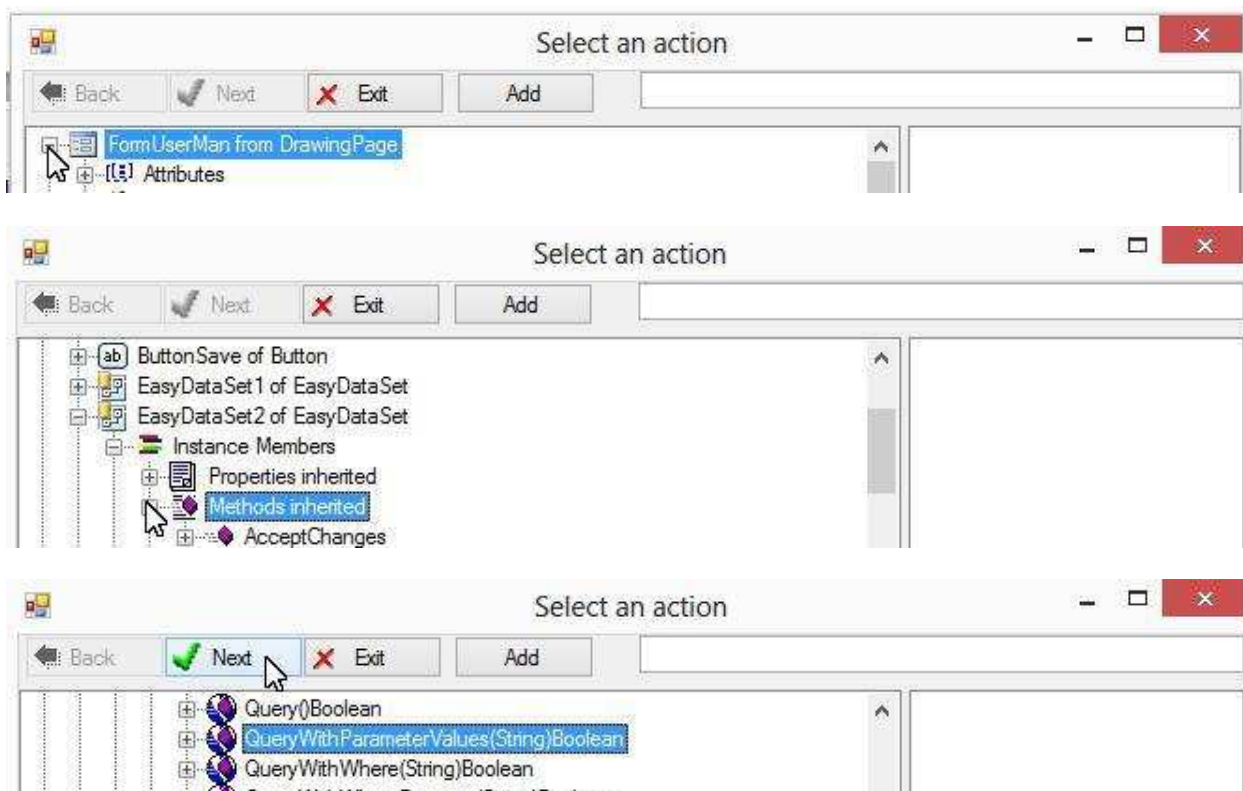




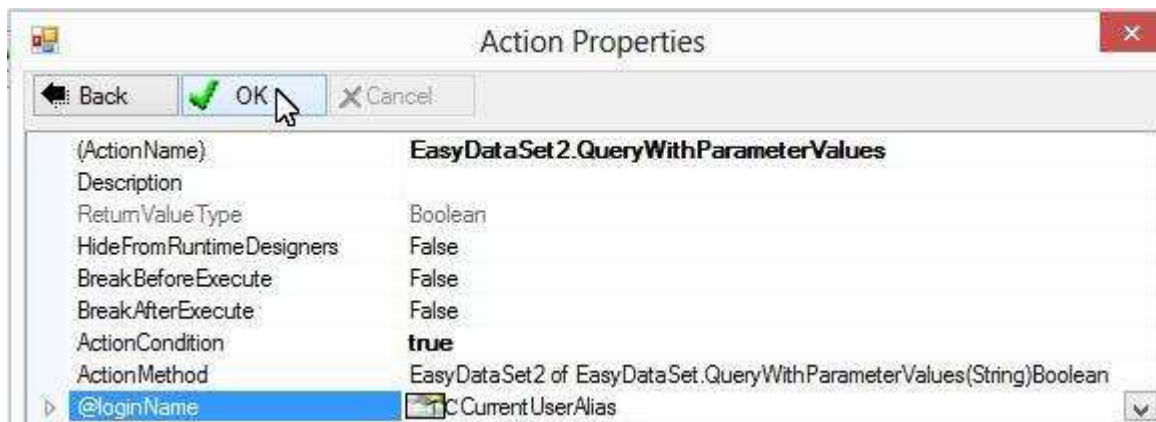
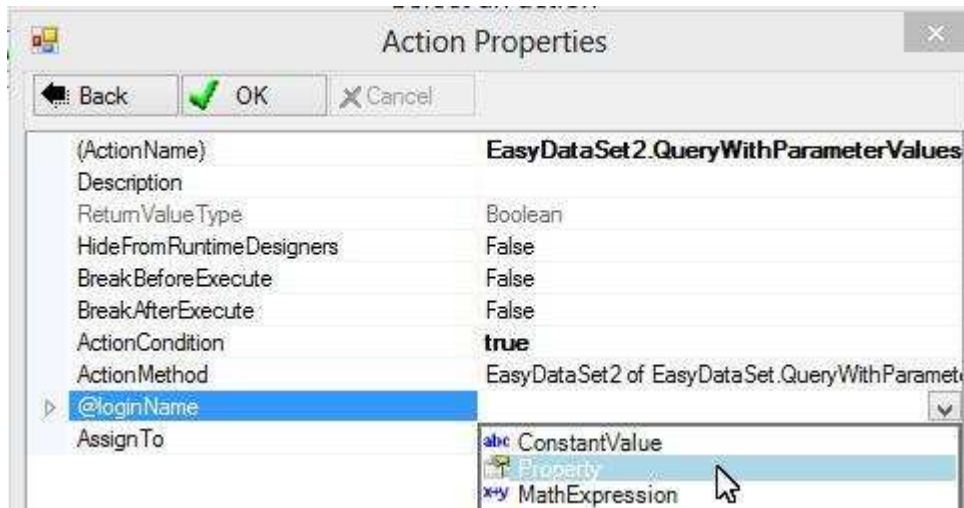
We may do the permission checking at the event Load. Right-click the form; choose “Assign Action”; choose “Load” event:



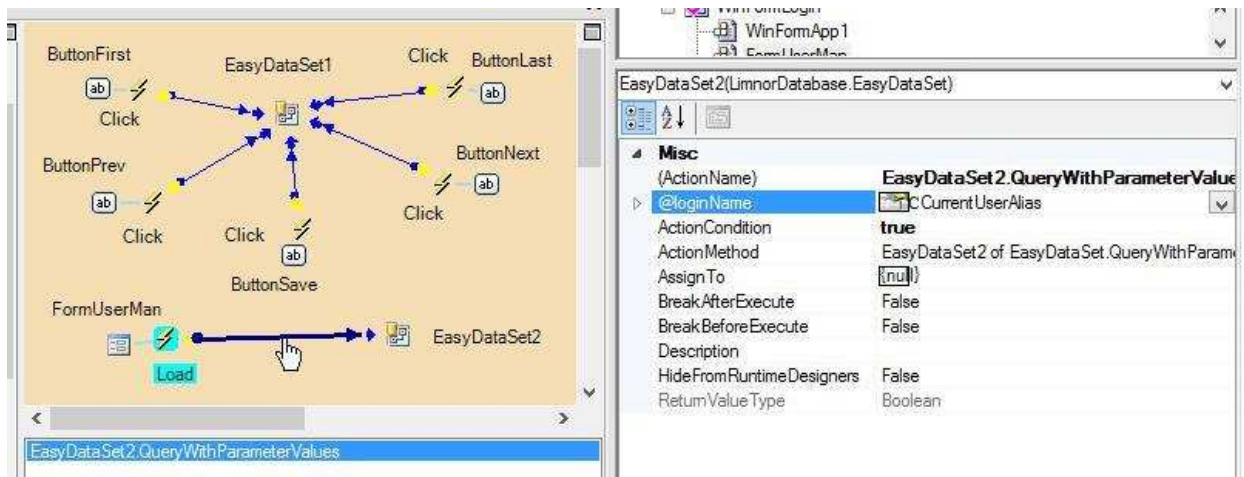
Select QueryWithParameterValues of EasyDataSet2:



Use the CurrentUserAlias for @loginName:

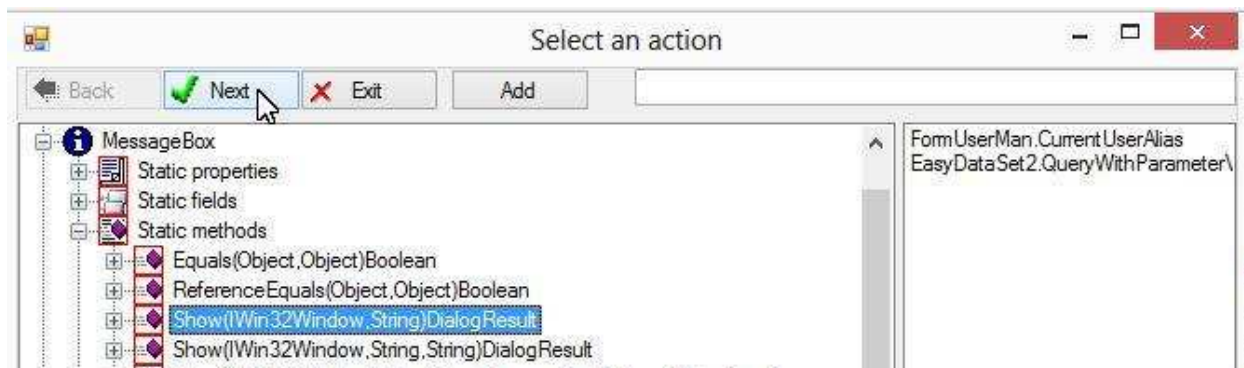


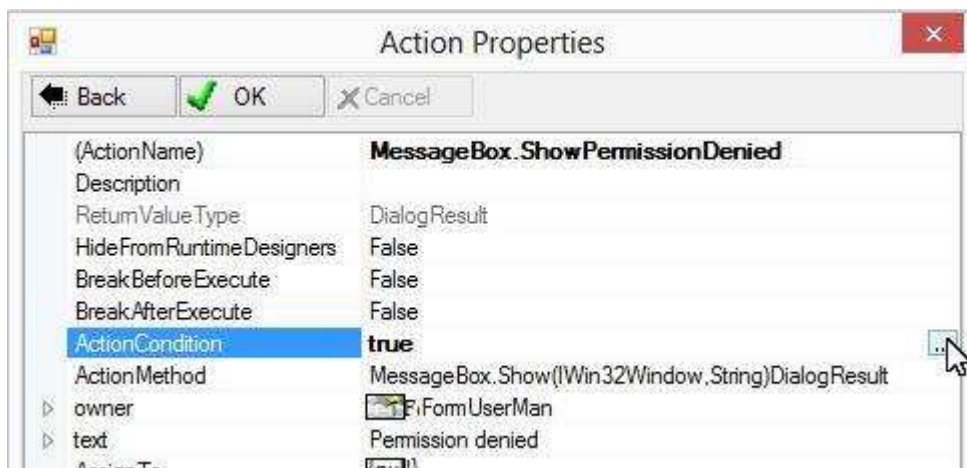
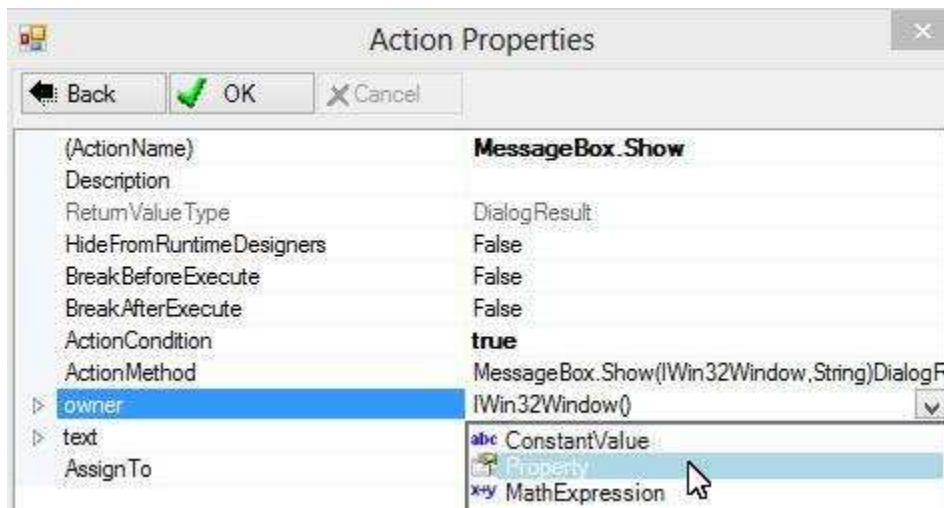
The action is created and assigned to Load event:



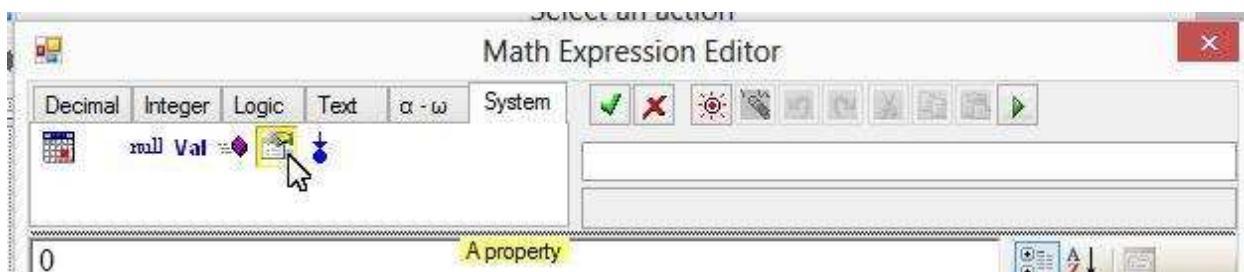
## Show “permission denied”

We may show a message box if cannot get a true for CanModifyUsers:

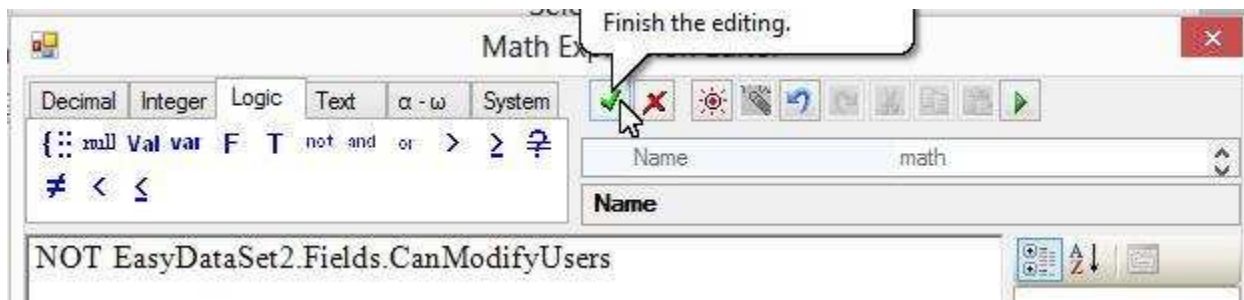
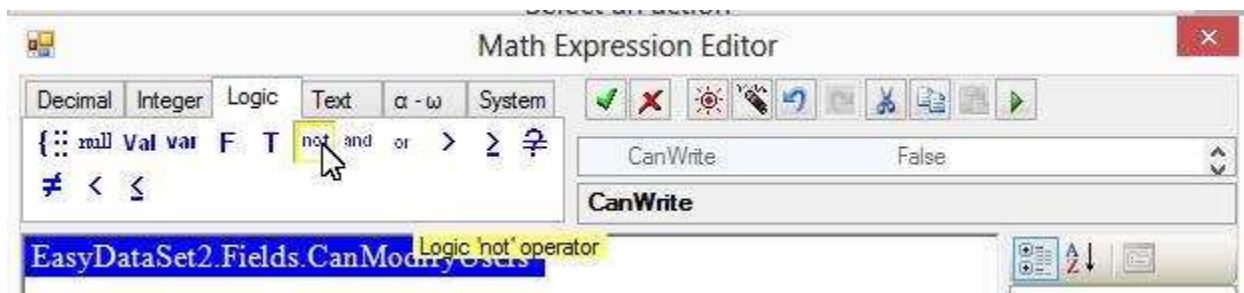


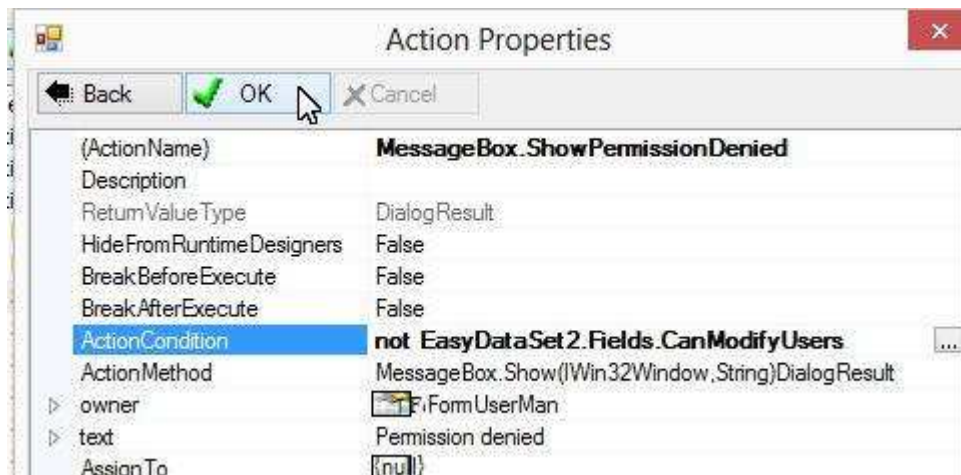


Select CanModifyUsers from EasyDataSet2:

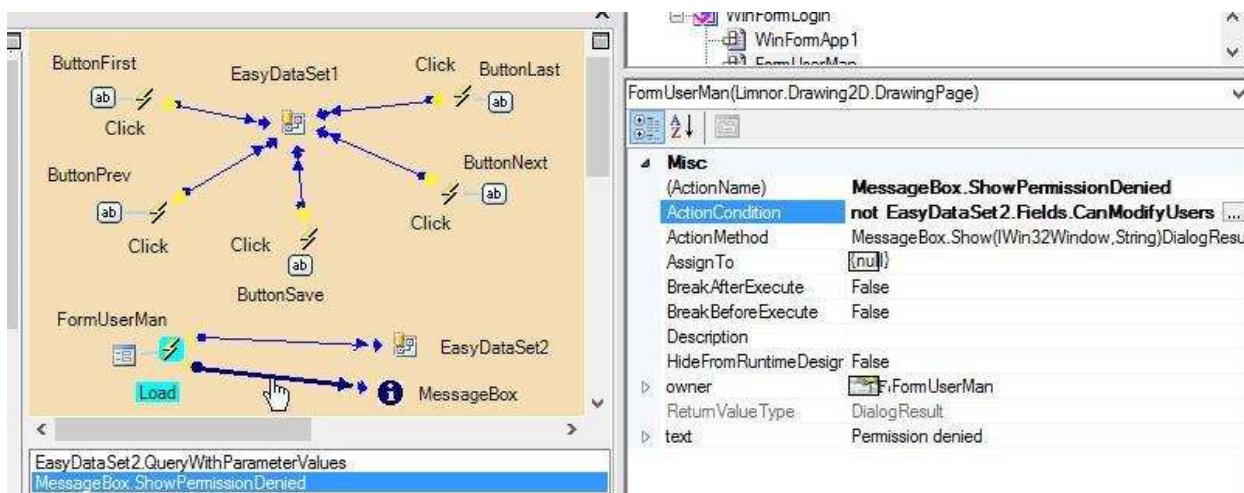




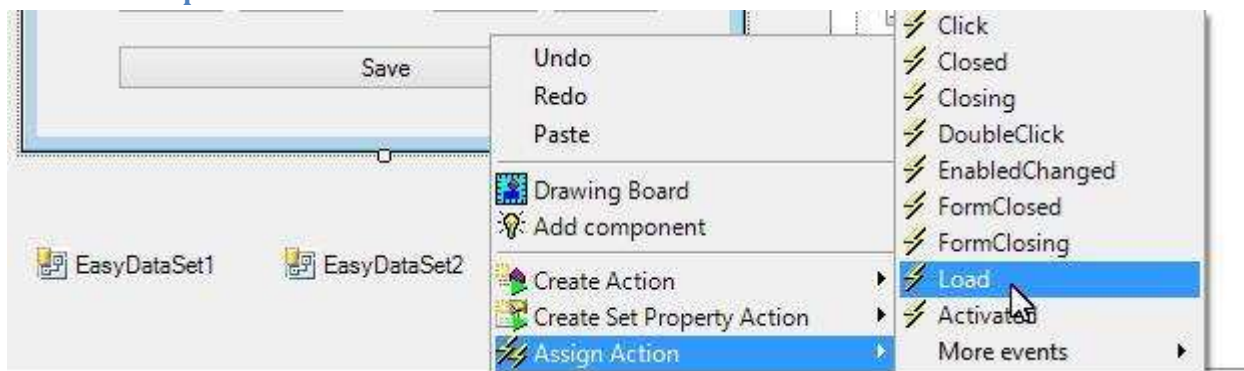


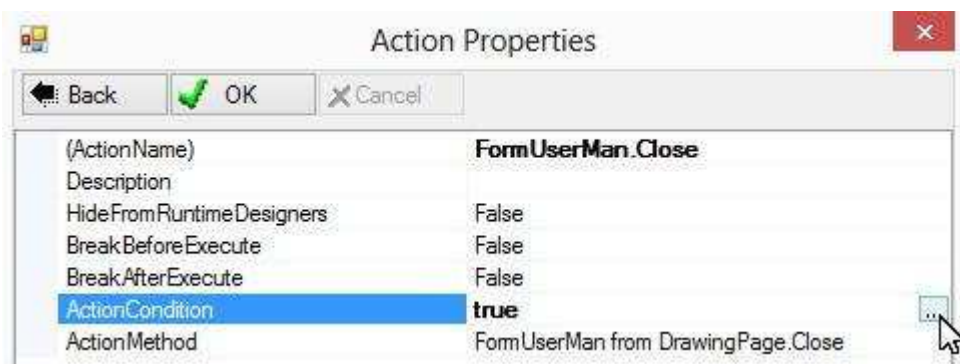


The action is created and assigned to event Load:

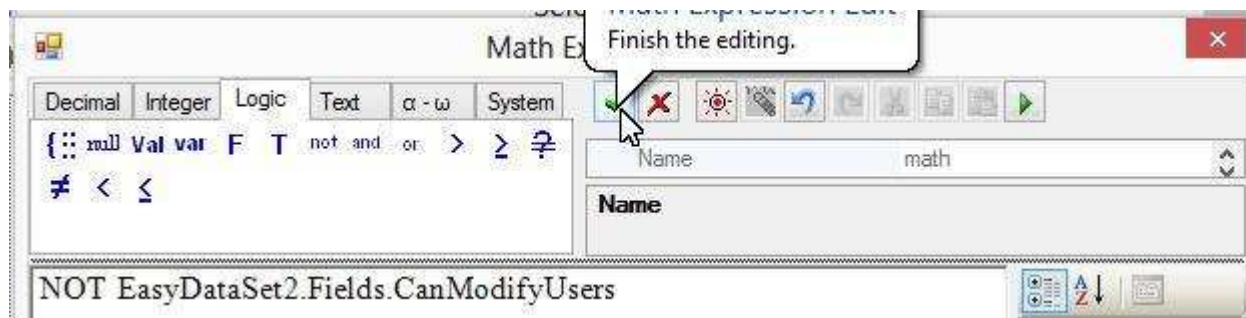


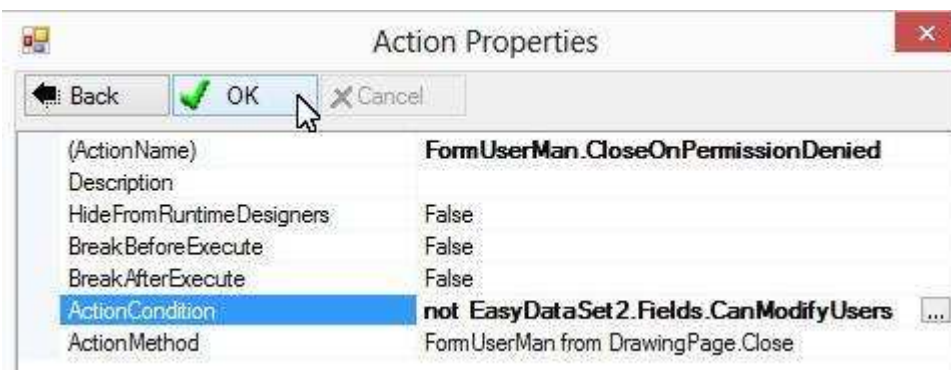
Close form if permission denied



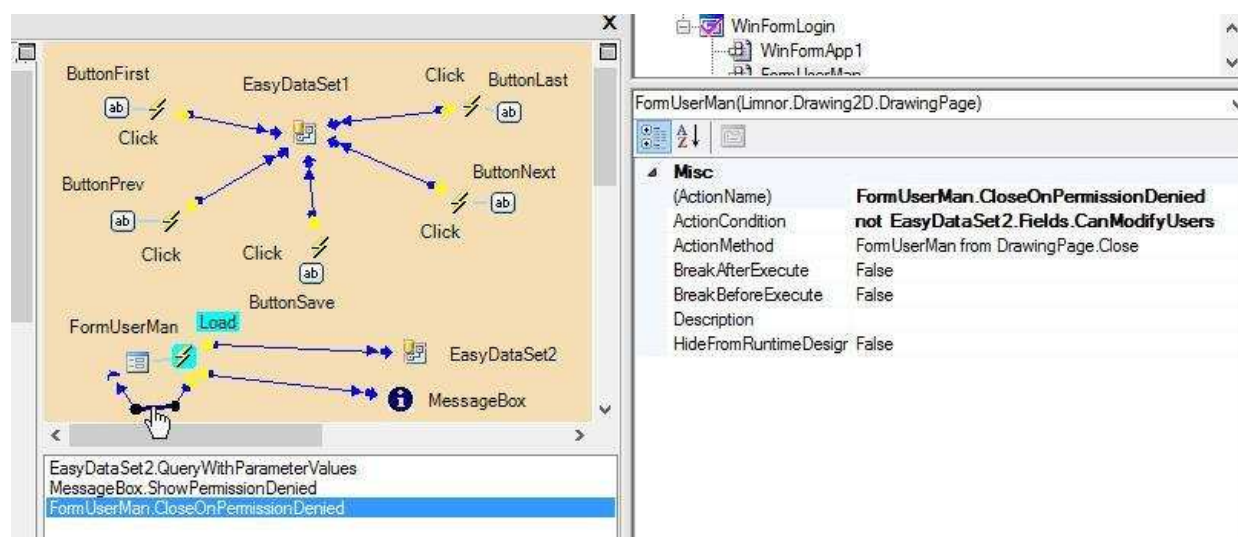


The condition is the same as the message box:





The action is created and assigned to event Load:

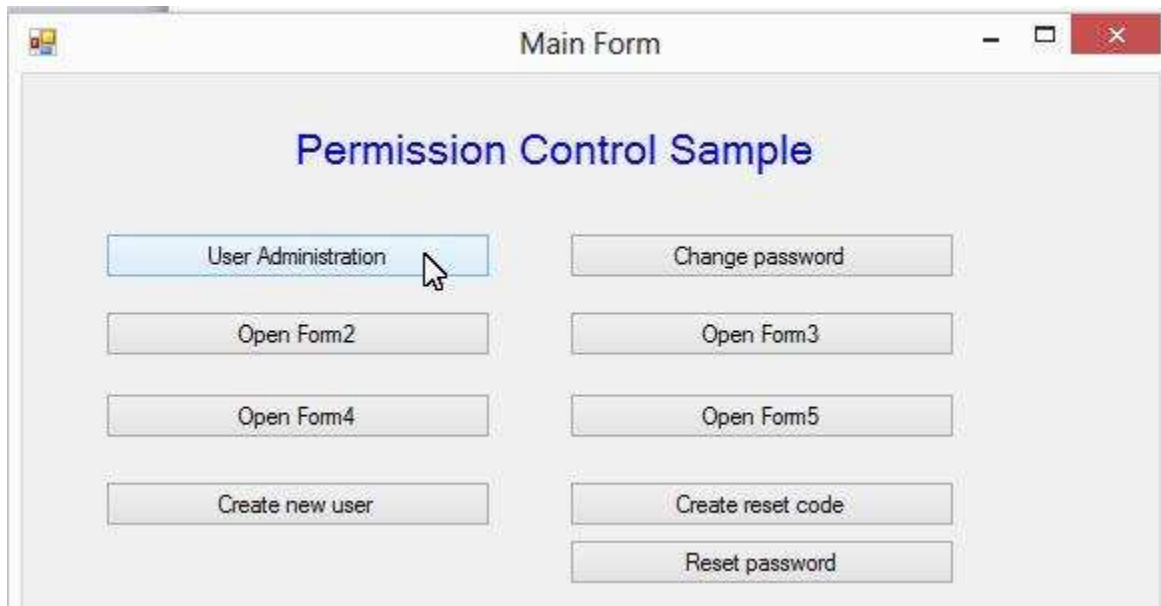


## Test

We set CanModifyUsers to true for user "user1" for our testing purpose. Now let's see how this field affects form permissions.

Run the application. Click "User Administration":





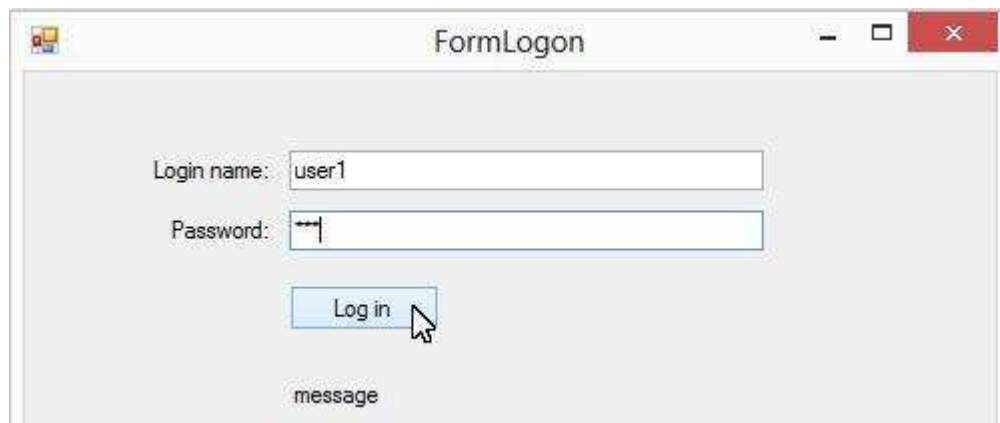
Log in with “admin”:



We get “Permission denied”:



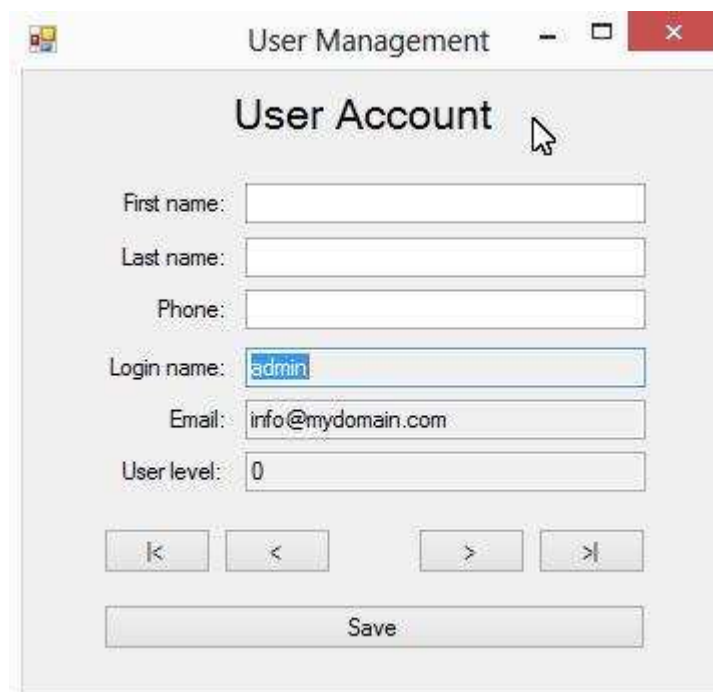
Re-start the application; click “User Administration” again. This time, log in as “user1”:



The screenshot shows a Windows Forms application window titled "FormLogon". Inside the window, there is a login form with the following elements:

- A "Login name:" label followed by a text box containing the text "user1".
- A "Password:" label followed by a password text box with masked characters (asterisks).
- A "Log in" button below the password field.
- A "message" label at the bottom of the form.

This time the form appears:



The screenshot shows a Windows Forms application window titled "User Management". Inside the window, there is a "User Account" form with the following elements:

- A title "User Account" at the top.
- Fields for "First name:", "Last name:", and "Phone:".
- A "Login name:" field with the text "admin" entered and highlighted.
- An "Email:" field with the text "info@mydomain.com" entered.
- A "User level:" field with the text "0" entered.
- Navigation buttons at the bottom: "<|", "<", ">", and ">|".
- A "Save" button at the bottom.

## Web Project Sample

### PHP Web Project Sample

See Part C from <http://www.limnor.com/support/LoginManagerPartC.PDF>

### ASPX Web Project Sample

See Part D from <http://www.limnor.com/support/LoginManagerPartD.PDF>