Jonathan Tweedle

2019/06/26

Assignment 04

Creating and Debugging my ASP.Net website

# Introduction

The goal is to demonstrate how to create and debug an ASP.Net website.

# Copy and Paste

So after creating the basic layout in the previous assignment we can start to implement some additional features from the previous assignment 02. In particular, the login request page.

Because of the change in the layout, the code needs to be copied and pasted into the reserved placeholder sections of the new login page created in Assignment 03.

After compiling and trying to test the updated login page, we run into a new error (Figure 1). This is a result of the way in which the navigation was implemented by making use of a <form> (Figure 2) which gets rendered on the server to handle the button clicks. The design now conflicts with the changes to the login page which also needs to use a form and there can only be one form defined in a page.

The solutions is to change the navigation design to instead use hyperlinks which no longer need to be contained inside of <form> tags (Figure 3). The change also requires some changes to the CSS3 styling which originally targeted the input tags (Figure 4) to instead target the hyperlink tags (Figure 5).



Figure 1: Multiple form tags on a page

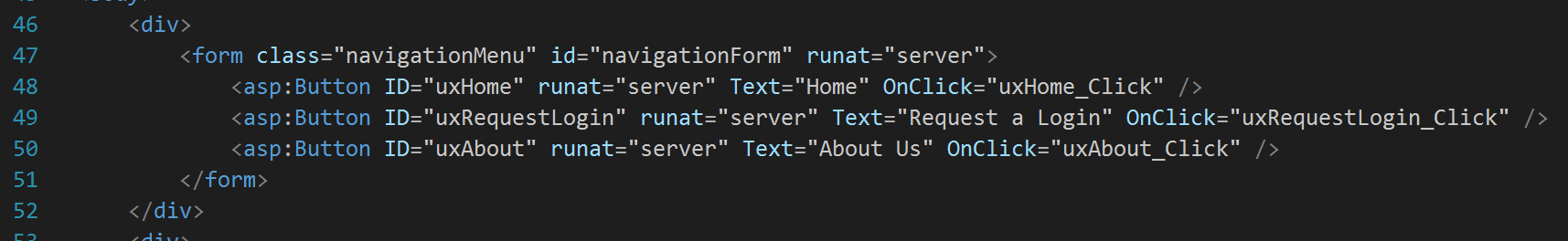


Figure 2: Original form based navigation

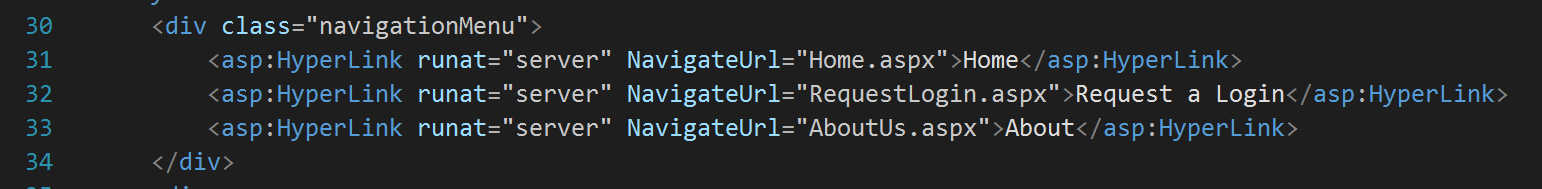


Figure 3: Hyperlink based navigation

|  |  |
| --- | --- |
| Figure 4: Old form navigation styling | Figure 5: New hyperlink navigation styling |

# Login Submission

With the changes, the form loads but fails when on the submit button click due to the target directory not existing in the project (Figure 6). It would be simple to just create the directory but since the goal is to instead store the data into a database, I refactored the submit click to break it down into 3 high level tasks (Figure 7). This makes the code design easier to understand and limits where future changes need to be made.

To make sure the code works without having any data actually stored in the database I activated trace debugging in the web config file (Figure 8). This allows me to inspect the page field values posted back when I submitted the form (Figure 9).

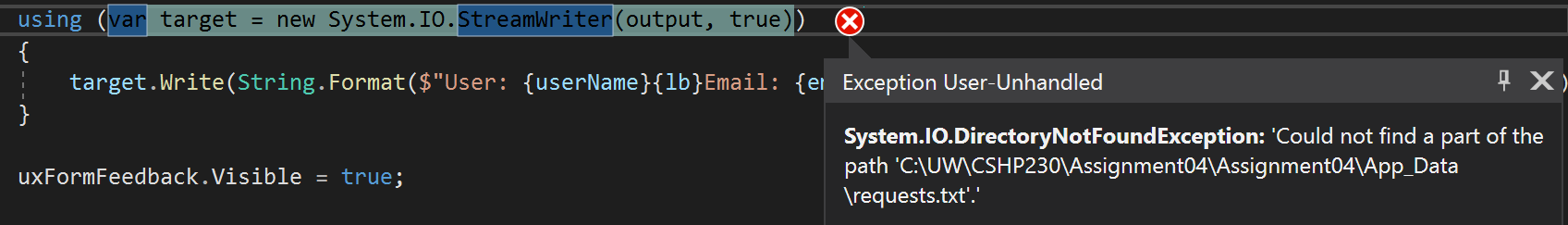


Figure 6: IO error as a result of missing directory

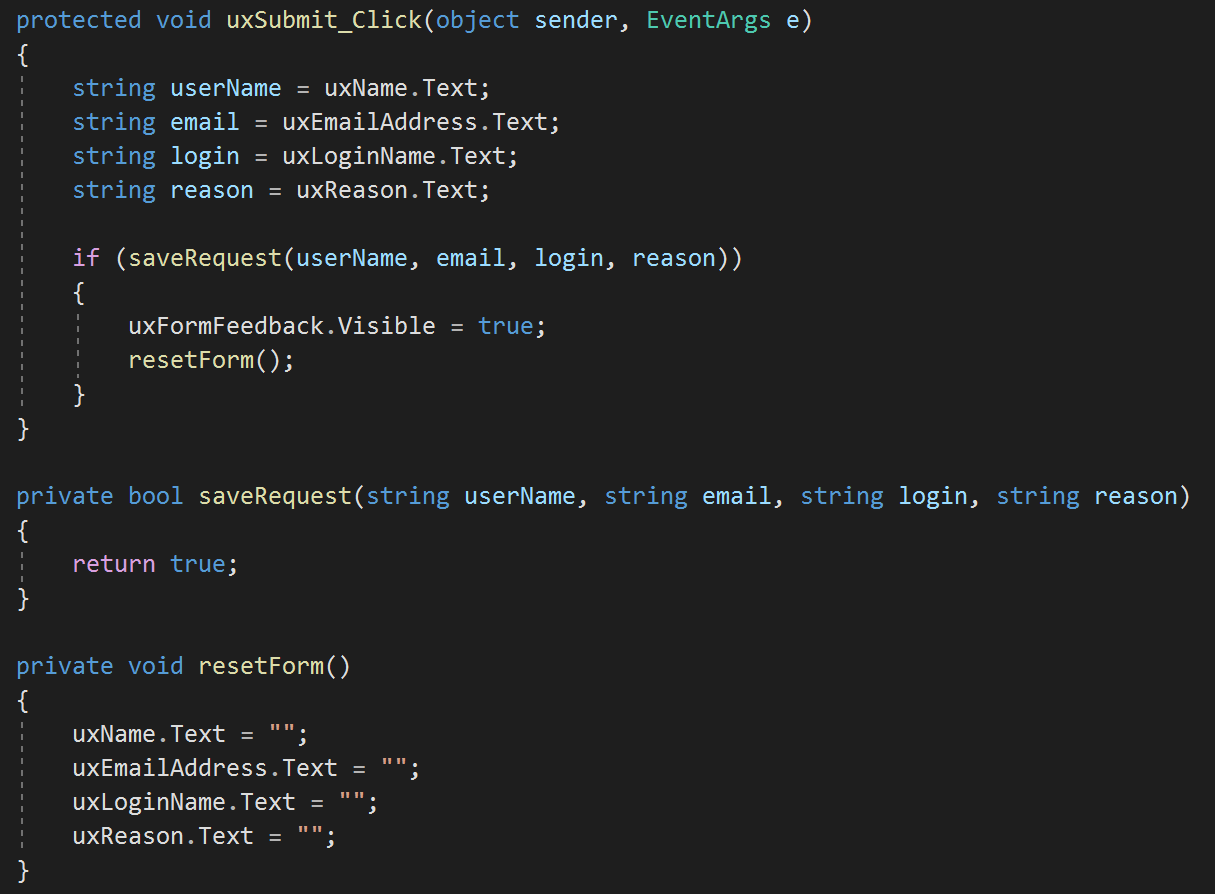


Figure 7: Refacotred server functions to handle form submission

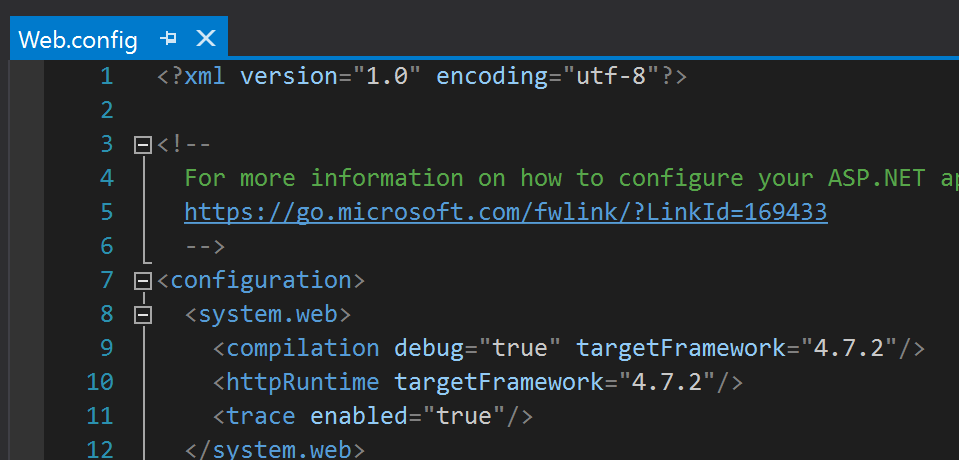


Figure 8: Enabling server page tracing

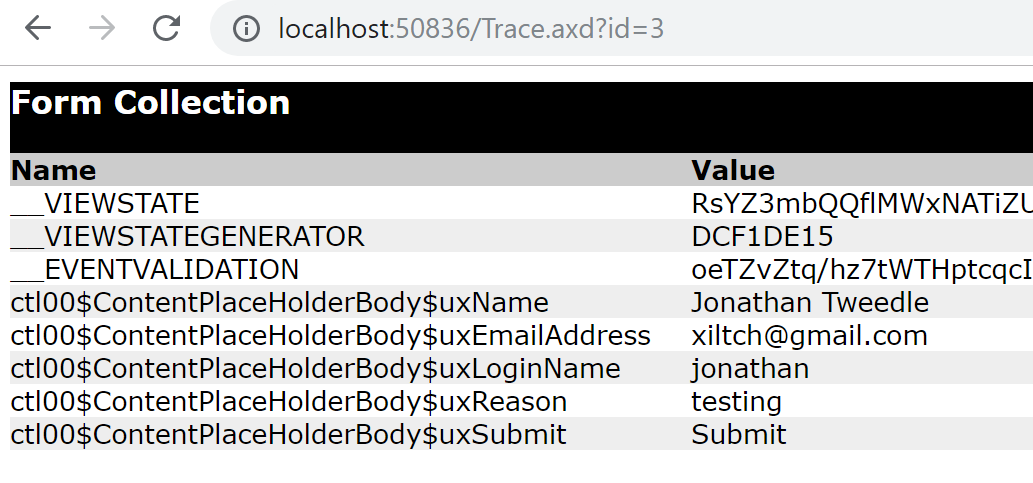


Figure 9: Tracing the POST reqest to verify information

# Persisting data to the Database

The design is now ready to persist the database back to the database. The basic concept is to create a connection and execute a stored procedure on the SQL server (Figure 11). In some configurations if the login does not execute permissions you will get a denied error message (Figure 10). You will need to update the permissions on the database before the SQL command will complete.

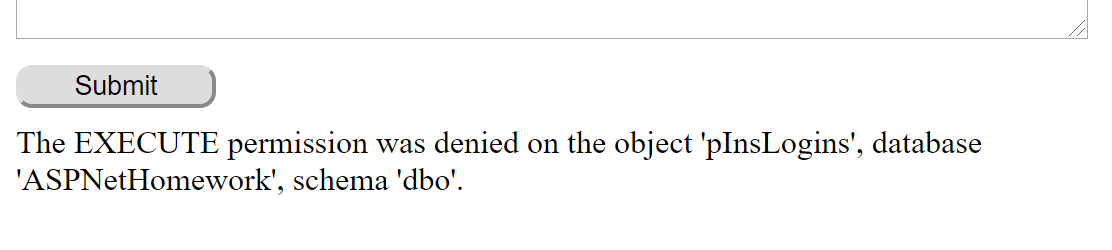


Figure 10: Error due to missing execute permissions

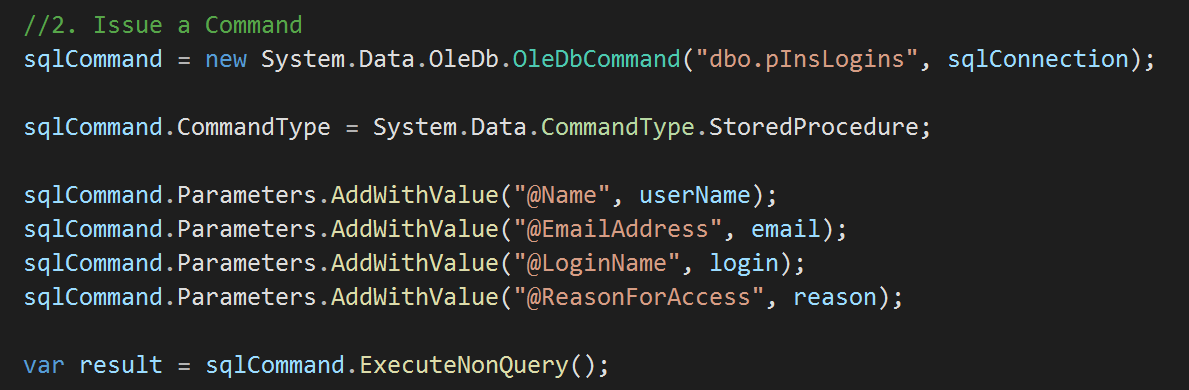


Figure 11: Basic SQL command code with parameters.

# Default Values

After resolving the permission issues due to my local setup, the code appeared to complete without issues. When checking the database though I found the values that got written to the database was incorrect (Figure 12). The parameter values were instead “Default Value”. I fired up the SQL profiler and notice that the values were not sending in the execution of the stored procedure (Figure 14) as you would expect in the correct execution (Figure 15).

I found this to a result of a missing line in the code that identifys the “CommandType” as a “StoredProcedure”. The code still worked and even though the parameters were set it failed to actually send the parameters. After adding the line I tested the code again and got the expected values to send through (Figure 13).

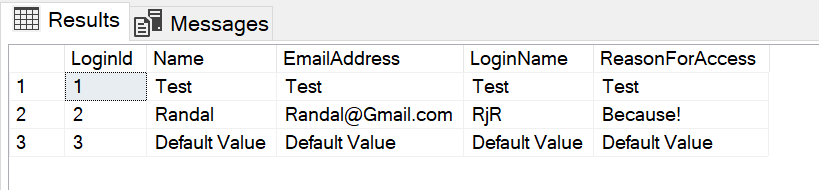


Figure 12: Default values sent through

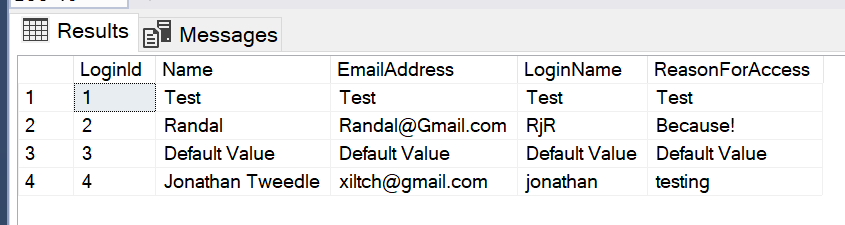


Figure 13: Correct values from the submitted form



Figure 14: SQL Profile view of the executed stored procedure missing parameters

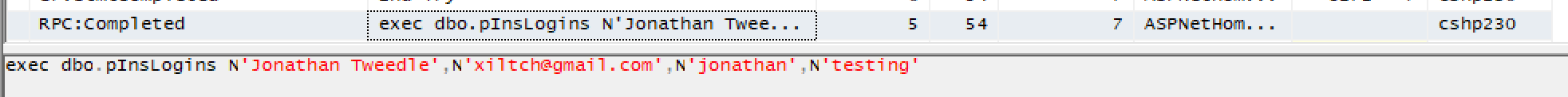


Figure 15: SQL Profile of the executed stored procedure with expected paramter values

# Final touches

I made some adjustments after the code worked to change the error handling slightly (Figure 16) and taking advantage of the feedback div to communicate with the user. This included changing from a Boolean result from saveRequest() function call to using an integer value to indicate the various states and control behaviour accordingly.

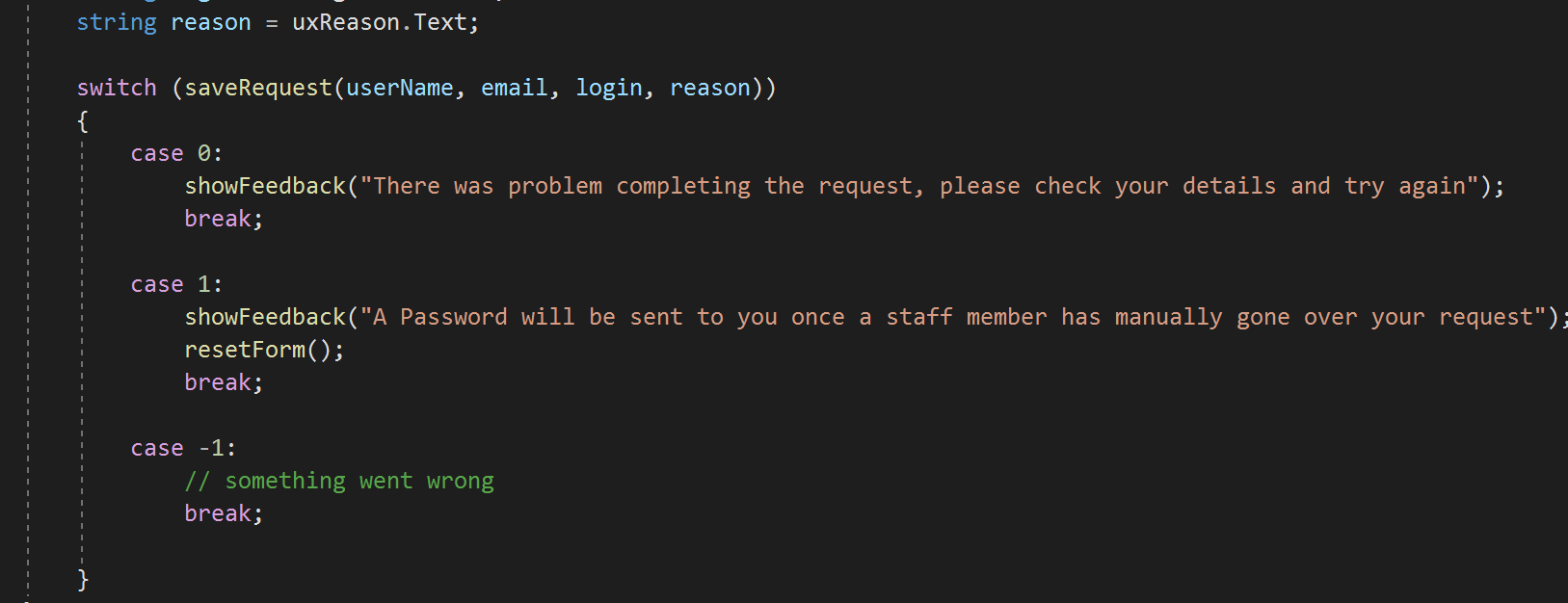


Figure 16: Change in result handling and error messages to the user

# Summary

Incremental changes to a design can be challenging that can often result in many hours of trial and error. To assist in the process of creating the site, use of various debugging tools can help to ease the process.