```
1  using System;
2  using System.Collections.Generic;
3  using System.Text;
4  using System.Windows;
5  using System.Windows.Controls;
6  using System.Windows.Data;
7  using System.Windows.Documents;
8  using System.Windows.Input;
9  using System.Windows.Media;
10 using System.Windows.Media.Imaging;
11 using System.Windows.Navigation;
12 using System.Windows.Shapes;
13
14 /*
15  * Title:   Code behind PageDetails
16  * Author:  Paul McKillop
17  * Date:    230302020
18  * Purpose: Manipulate and process data
19  */
20
21 namespace SignatureGeneratorV
22 {
23     /// <summary>
24     /// Interaction logic for PageDetails.xaml
25     /// </summary>
26     public partial class PageDetails : Page
27     {
28         //-- Handler variables. Global in the module
29         internal string forename = string.Empty;
30         internal string surname = string.Empty;
31         internal string userstring = string.Empty;
32
33         //-- Validity tracking
34         internal bool forenameData;
35         internal bool surnameData;
36         internal bool userStringData;
37         internal bool stringValid = false;
38
39         //-- Demo of the use of inherited child class
40         private  PersonUser personUser = new PersonUser();
41
42         public PageDetails()
43         {
44             InitializeComponent();
45
46         }
47
48         //-- Event raised by button ClearButton
49         private void ClearButton_Click(object sender, RoutedEventArgs e)
50         {
51             ClearControls();
52         }
53
54         //-- Event raised by SummaryButton
55         private void SummaryButton_Click(object sender, RoutedEventArgs e)
56         {
```

```
57              //-- DEBUG MessageBox.Show("Summary button clicked");
58              //-- DEBUG MessageBox.Show(HarvestData().ToString());
59
60              forenameData = false;
61
62              if (HarvestData())
63              {
64
65                  //-- Implementation of child class inherited from Person
66                  personUser.Forename = forename;
67                  personUser.Surname = surname;
68                  personUser.KeyLowerLimit = UtilityZGlobals.LengthRule();
69
70                  var userData = new UserData();
71
72                  if (UtilityValidator.CharactersAllValid(userstring))
73                  {
74                      stringValid = true;
75
76                      MessageBox.Show("Characters are all valid");
77                      //-- The userstring can be processed
78
79                      userData = new UserData
80                      {
81                          Forename = forename,
82                          Surname = surname,
83                          Username = UtilityString.MakeUsername(forename,
                        surname),
84                          UserStringOriginal = userstring,
85                          UserStringReversed = UtilityString.ReverseString
                        (userstring),
86                          Score = UtilityValidator.WholeStringScore(userstring),
87                          StrengthGradeLong = UtilityValidator.StrengthGradeLong
                        (UtilityValidator.WholeStringScore(userstring))
88                      };
89
90                      MessageBox.Show(UserData.SummaryDataDisplay(userData));
91                  }
92                  else
93                  {
94                      stringValid = false;
95
96                      var invalidCharacter =
                        UtilityValidator.GetInvalidCharacter(userstring);
97                      MessageBox.Show("Invalid character " +
                        invalidCharacter.Character + " at position " +
                        invalidCharacter.Position);
98                      //-- DEBUG MessageBox.Show("Invalid characters");
99                  }
100             }
101             else
102             {
103                 MessageBox.Show("There was a problem with the data. Follow
                    message instructions");
104             }
105
```

```
106             var userDataToPass = new UserData
107             {
108                 Forename = forename,
109                 Surname = surname,
110                 Username = UtilityString.MakeUsername(forename, surname),
111                 UserStringOriginal = userstring,
112                 UserStringReversed = UtilityString.ReverseString(userstring),
113                 Score = UtilityValidator.WholeStringScore(userstring),
114                 StrengthGradeLong = UtilityValidator.StrengthGradeLong
                      (UtilityValidator.WholeStringScore(userstring))
115             };
116 
117             if (stringValid)
118             {
119                 var pageSummary = new PageSummary(userDataToPass);
120                 this.NavigationService.Navigate(pageSummary);
121             }
122             else
123             {
124                 MessageBox.Show("Cannot show summary with invalid data");
125             }
126         }
127 
128         /// <summary>
129         /// Harvest form data
130         /// </summary>
131         /// <returns></returns>
132         private bool HarvestData()
133         {
134             forenameData = false;
135 
136             //-- Object to hold data
137             FormData formData = new FormData();
138             int dataPresent = 0;
139             bool allData = false;
140 
141             // Forename
142             if (ForenameTextBox.Text != "")
143             {
144                 formData.Forename = ForenameTextBox.Text;
145                 forenameData = true;
146                 dataPresent++;
147             }
148             else
149             {
150                 MessageBox.Show("You must enter a Forename");
151                 forenameData = false;
152             }
153 
154             // Surname
155             if (SurnameTextBox.Text != "")
156             {
157                 formData.Surname = SurnameTextBox.Text;
158                 surnameData = true;
159                 dataPresent++;
160             }
```

```
161             else
162             {
163                 MessageBox.Show("You must enter a Surname");
164                 surnameData = false;
165             }
166 
167             // User string
168             if (UserStringTextBox.Text != "")
169             {
170                 //-- get what's in the text box
171                 formData.UserString = UserStringTextBox.Text;
172                 //-- Length check
173                 if (formData.UserString.Length >= UtilityZGlobals.LengthRule
                      ())
174                 {
175                     userStringData = true;
176                     dataPresent++;
177                 }
178                 else
179                 {
180                     //-- report length error
181                     MessageBox.Show("The user entered string does not meet
                        length rule");
182                     userStringData = false;
183                 }
184 
185             }
186             else
187             {
188                 MessageBox.Show("You must enter a Key String");
189                 userStringData = false;
190             }
191 
192 
193             if (dataPresent == 3)
194             {
195                 //-- All is wonderful
196                 forename = formData.Forename;
197                 surname = formData.Surname;
198                 userstring = formData.UserString;
199 
200                 allData = true;
201             }
202             else
203             {
204                 //-- There are data missing
205                 allData = false;
206             }
207 
208             return allData;
209         }
210 
211         /// <summary>
212         /// Clear controls and set focus to ForenameTextBox
213         /// </summary>
214         private void ClearControls()
```

```
215         {
216             ForenameTextBox.Text = "";
217             SurnameTextBox.Text = "";
218             UserStringTextBox.Text = "";
219 
220             ForenameTextBox.Focus();
221         }
222 
223 
224         /// <summary>
225         /// Private class for data manipulation during validation and harvest
226         /// </summary>
227         private class FormData
228         {
229             public string Forename { get; set; }
230             public string Surname { get; set; }
231             public string UserString { get; set; }
232         }
233     }
234 }
235 
```