

```
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 using FlooringCalculator.Models;
14
15 /*
16  * Title:   PageDataEntry
17  * Author:  Paul McKillop
18  * Date:    November 2020
19  * Purpose: Code behind for page
20  */
21
22 /* *****
23  * COMPLETION SEQUENCE
24  * *****
25  *
26  * This is the most detailed of all the processes.
27  * The order is important because of the dependency of
28  * some methods on others that must already be created.
29  *
30  * Video 24
31  * 00. Check all Gui controls have names
32  * 01. Directive for models
33  * 02. Gui control methods
34  *     a) Clear button
35  *     b) Calculate button
36  *     c) Combo OnLoaded
37  *     d) Combo OnSelectionChanged
38  * 03. Handler data variables
39  * 04. Assignment data for testing
40  * 05. Call assignment data in Page constructor
41  * Video 25
42  * 06. Create a list of tiles for Combo control
43  * 07. Complete combo OnLoaded method
44  * 08. Complete combo OnSelectionChanged method
45  * 09. Create ResetControls method
46  * 10. Call ResetControls from Clear button Click
47  * Video 26
48  * 11. Create method GetSelectedTile
49  * 12. Create method ControlHasValueCheck
50  * 13. Gui Help Button Click method
51  * 14. Implement Help button method
52  * Video 27
53  * 15. HarvestData method
54  * 16. Prepare PageSummary to receive data
55  * Video 28
56  * 17. Implement Calculate Button Click method
57  * 18. Test all
58  */
```

```
59
60 namespace FlooringCalculator
61 {
62     /// <summary>
63     /// Interaction logic for PageDataEntry.xaml
64     /// </summary>
65     public partial class PageDataEntry : Page
66     {
67
68         // -- variables for management of data in the module
69         private string selectedTileName = string.Empty;
70         private Room room = new Room();
71         private Tile tile = new Tile();
72         private DataSummary dataSummary = new DataSummary();
73
74         public PageDataEntry()
75         {
76             InitializeComponent();
77             AssignmentRoomData();
78         }
79
80
81         // -- Clears the form's controls
82         private void ClearButton_OnClick(object sender, RoutedEventArgs e)
83         {
84             ResetControls();
85         }
86
87         private void CalculateButton_OnClick(object sender, RoutedEventArgs e)
88         {
89
90         }
91
92         private void TileComboBox_OnLoaded(object sender, RoutedEventArgs e)
93         {
94             var combo = (ComboBox)sender;
95             if (combo == null) return;
96             combo.ItemsSource = Tiles();
97             combo.SelectedIndex = 0;
98         }
99
100        private void TileComboBox_OnSelectionChanged(object sender,           ↗
101            SelectionChangedEventArgs e)
102        {
103            var combo = (ComboBox)sender;
104
105            try
106            {
107                if (combo != null) selectedTileName =           ↗
108                    combo.SelectedItem.ToString();
109            }
110            catch (Exception exception)
111            {
112                Console.WriteLine(exception);
113                throw;
114            }
115        }
116        private void LaunchHelpButton_OnClick(object sender, RoutedEventArgs ↗
117            e)
118        {
119            var pageHelp = new PageHelp();
120        }
121    }
122 }
```

```
117         if (NavigationService != null) NavigationService.Navigate
118             (pageHelp);
119     }
120
121     private void AssignmentRoomData()
122     {
123         RoomWideATextBox.Text = "6.50";
124         RoomLongBTextBox.Text = "7.20";
125         Cutout1WideCTextBox.Text = "1.60";
126         Cutout1LongDTextBox.Text = "2.30";
127         Cutout2WideETextBox.Text = "0.6";
128         Cutout2LongFTextBox.Text = "0.3";
129     }
130
131     /// <summary>
132     /// Make a list of tile types
133     /// </summary>
134     /// <returns></returns>
135     private List<string> Tiles()
136     {
137         List<string> tiles = new List<string>();
138         tiles.Add("60 x 60");
139         tiles.Add("75 x 75");
140
141         return tiles;
142     }
143
144     /// <summary>
145     /// Reset data entry controls
146     /// </summary>
147     private void ResetControls()
148     {
149         RoomWideATextBox.Text = "0";
150         RoomLongBTextBox.Text = "0";
151         Cutout1WideCTextBox.Text = "0";
152         Cutout1LongDTextBox.Text = "0";
153         Cutout2WideETextBox.Text = "0";
154         Cutout2LongFTextBox.Text = "0";
155     }
156
157     /// <summary>
158     /// In production, this would search a database for stored
159     /// Tile information
160     /// </summary>
161     /// <returns>Tile object</returns>
162     private Tile GetSelectedTile()
163     {
164         var tempTile = new Tile();
165
166         switch (selectedTileName)
167         {
168             case "60 x 60":
169                 tempTile.TileWide = 0.60m;
170                 tempTile.TileLong = 0.60m;
171                 break;
172             case "75 x 75":
173                 tempTile.TileWide = 0.75m;
```

```
175         tempTile.TileLong = .75m;
176         break;
177         // -- we must provide a default case allowing for
178         // -- not in list
179         default:
180             tempTile.TileWide = 1;
181             tempTile.TileLong = 1;
182             break;
183     }
184
185     return tempTile;
186 }
187
188
189 /// <summary>
190 /// /// Check that all text box controls have a value in the
191 /// to work with
192 /// </summary>
193 /// <returns>bool true if all have data</returns>
194 private bool ControlHasValueCheck()
195 {
196     return !string.IsNullOrEmpty(RoomWideATextBox.Text) &&
197            !string.IsNullOrEmpty(RoomLongBTextBox.Text) &&
198            !string.IsNullOrEmpty(Cutout1WideCTextBox.Text) &&
199            !string.IsNullOrEmpty(Cutout1LongDTextBox.Text) &&
200            !string.IsNullOrEmpty(Cutout2WideETextBox.Text) &&
201            !string.IsNullOrEmpty(Cutout2LongFTextBox.Text);
202 }
203
204 }
205 }
206
```