```
 1  using Microsoft.Xna.Framework;
 2  using Microsoft.Xna.Framework.Graphics;
 3  using Microsoft.Xna.Framework.Input;
 4
 5  namespace coursework
 6  {
 7      class Textbox : InteractiveObject
 8      {
 9          private KeyboardState currentKeyboardState;
10          string dataType;
11          int maxLength;
12
13          Keys[] keysToCheckStr = new Keys[] {
14          Keys.A, Keys.B, Keys.C, Keys.D, Keys.E,
15          Keys.F, Keys.G, Keys.H, Keys.I, Keys.J,
16          Keys.K, Keys.L, Keys.M, Keys.N, Keys.O,
17          Keys.P, Keys.Q, Keys.R, Keys.S, Keys.T,
18          Keys.U, Keys.V, Keys.W, Keys.X, Keys.Y,
19          Keys.Z, Keys.Back, Keys.Space,
20          Keys.D0, Keys.D1, Keys.D2, Keys.D3, Keys.D4,
21          Keys.D5, Keys.D6, Keys.D7, Keys.D8, Keys.D9 };
22
23          Keys[] keysToCheckInt = new Keys[] {
24          Keys.D0, Keys.D1, Keys.D2, Keys.D3, Keys.D4,
25          Keys.D5, Keys.D6, Keys.D7, Keys.D8, Keys.D9,
26          Keys.Back };
27
28          private KeyboardState lastKeyboardState;
29
30          public Textbox(Rectangle rectangle, Texture2D texture, SpriteFont
              font, string text, string dataType, int maxLength)
31          {
32              this.rectangle = rectangle;
33              this.texture = texture;
34              this.font = font;
35              this.text = text;
36              this.dataType = dataType;
37              this.maxLength = maxLength;
38          }
39          public override void Update(MouseState mouseState, SpriteBatch
              spriteBatch)
40          {
41              if (rectangle.Contains(mouseState.X, mouseState.Y))
42              {
43                  if (mouseState.LeftButton == ButtonState.Pressed)
44                  {
45                      state = State.down;
46                  }
47                  else if (state == State.none)
48                  {
49                      state = State.hover;
50                  }
51                  else
52                  {
53                      state = State.up;
54                  }
```

```
55              }
56              else if (state == State.up && mouseState.LeftButton ==
                  ButtonState.Pressed)
57              {
58                  state = State.none;
59              }
60              else if (state != State.up)
61              {
62                  state = State.none;
63              }
64              Draw(spriteBatch);
65
66              if (state == State.up)
67              {
68                  Keys[] keysToCheck;
69                  currentKeyboardState = Keyboard.GetState();
70                  if (dataType == "str")
71                  {
72                      keysToCheck = keysToCheckStr;
73                  }
74                  else
75                  {
76                      keysToCheck = keysToCheckInt;
77                  }
78                  foreach (Keys key in keysToCheck)
79                  {
80                      if (CheckKey(key) && (text.Length < maxLength || key ==
                        Keys.Back))
81                      {
82                          AddKeyToText(key);
83                          break;
84                      }
85                  }
86
87
88                  lastKeyboardState = currentKeyboardState;
89              }
90          }
91          private void AddKeyToText(Keys key)
92          {
93              string newChar = "";
94
95              if (text.Length >= 20 && key != Keys.Back)
96                  return;
97
98              switch (key)
99              {
100                 case Keys.A:
101                     newChar += "a";
102                     break;
103                 case Keys.B:
104                     newChar += "b";
105                     break;
106                 case Keys.C:
107                     newChar += "c";
108                     break;
```

```
109                     case Keys.D:
110                         newChar += "d";
111                         break;
112                     case Keys.E:
113                         newChar += "e";
114                         break;
115                     case Keys.F:
116                         newChar += "f";
117                         break;
118                     case Keys.G:
119                         newChar += "g";
120                         break;
121                     case Keys.H:
122                         newChar += "h";
123                         break;
124                     case Keys.I:
125                         newChar += "i";
126                         break;
127                     case Keys.J:
128                         newChar += "j";
129                         break;
130                     case Keys.K:
131                         newChar += "k";
132                         break;
133                     case Keys.L:
134                         newChar += "l";
135                         break;
136                     case Keys.M:
137                         newChar += "m";
138                         break;
139                     case Keys.N:
140                         newChar += "n";
141                         break;
142                     case Keys.O:
143                         newChar += "o";
144                         break;
145                     case Keys.P:
146                         newChar += "p";
147                         break;
148                     case Keys.Q:
149                         newChar += "q";
150                         break;
151                     case Keys.R:
152                         newChar += "r";
153                         break;
154                     case Keys.S:
155                         newChar += "s";
156                         break;
157                     case Keys.T:
158                         newChar += "t";
159                         break;
160                     case Keys.U:
161                         newChar += "u";
162                         break;
163                     case Keys.V:
164                         newChar += "v";
```

```
165                     break;
166                 case Keys.W:
167                     newChar += "w";
168                     break;
169                 case Keys.X:
170                     newChar += "x";
171                     break;
172                 case Keys.Y:
173                     newChar += "y";
174                     break;
175                 case Keys.Z:
176                     newChar += "z";
177                     break;
178                 case Keys.D0:
179                     newChar += "0";
180                     break;
181                 case Keys.D1:
182                     newChar += "1";
183                     break;
184                 case Keys.D2:
185                     newChar += "2";
186                     break;
187                 case Keys.D3:
188                     newChar += "3";
189                     break;
190                 case Keys.D4:
191                     newChar += "4";
192                     break;
193                 case Keys.D5:
194                     newChar += "5";
195                     break;
196                 case Keys.D6:
197                     newChar += "6";
198                     break;
199                 case Keys.D7:
200                     newChar += "7";
201                     break;
202                 case Keys.D8:
203                     newChar += "8";
204                     break;
205                 case Keys.D9:
206                     newChar += "9";
207                     break;
208                 case Keys.Space:
209                     newChar += " ";
210                     break;
211                 case Keys.Back:
212                     if (text.Length != 0)
213                         text = text.Remove(text.Length - 1);
214                     return;
215             }
216             if (currentKeyboardState.IsKeyDown(Keys.RightShift) ||
217                 currentKeyboardState.IsKeyDown(Keys.LeftShift))
218             {
219                 newChar = newChar.ToUpper();
220             }
```

```
221             text += newChar;
222         }
223
224         private bool CheckKey(Keys theKey)
225         {
226             return lastKeyboardState.IsKeyDown(theKey) &&
                    currentKeyboardState.IsKeyUp(theKey);
227         }
228
229     }
230 }
231
```