/\*

\* Guess A Word (Hangman redux)

\* Create a GUI version.

\* Turn visibility of different UI elements on and off as needed. Add a counter displaying the number of submitted failed guesses (including repeated ones).

\* Use a fixed-width font for displaying a word being guessed.

\*/

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace GuessAWord

{

public partial class GuessAWordForm : Form

{

string[] words = { "TEST", "APPLES", "BASK", "NEOPHYTE", "BASTION", "I", "OSTENTATIOUS", "STALLION", "MASK", "FACADE", "MINIONS", "ELK", "FANTASTIC", "MYTH" };

const bool ON = true;

const bool OFF = false;

int i, j; // universal workhorse iterators

string word;

string guessedLetters;

bool gameOver;

int wrongGuesses;

int guesses;

string guess;

bool goodGuess;

bool isNewGuess;

bool validEntry;

string answerboard;

char[] wordArray;

char[] answerboardArray;

//

//

// form methods

//

//

public GuessAWordForm()

{

InitializeComponent();

// set label text constants

welcomeLogoLbl.Text = "\* \* \* \* \* \* \* \* \* \*";

wonLbl.Text = "CONGRATULATIONS!!";

wonLblLine4.Text = "WELL DONE!!";

}

private void startBtn\_Click(object sender, EventArgs e)

{

gameOver = false;

wrongGuesses = 0;

guesses = 0;

guess = "";

guessedLetters = "";

goodGuess = false;

validEntry = true;

feedbackLbl.Text = "";

word = SelectWord(0, words.Length - 1);

wordArray = word.ToCharArray();

answerboard = "";

for (i = 0; i < word.Length; ++i)

answerboard += "\*";

answerboardLbl.Text = answerboard;

answerboardArray = answerboard.ToCharArray();

if (welcomeLogoLbl.Visible == true)

ToggleWelcomeScreen(OFF);

if (wonLbl.Visible == true)

ToggleWonScreen(OFF);

ToggleGameUI(ON);

} // end startBtn\_Click

private void submitBtn\_Click(object sender, EventArgs e)

{

guess = guessTxtbx.Text;

goodGuess = false;

validEntry = CheckLength(guess);

if (validEntry)

validEntry = CheckForLetter(guess);

if (validEntry)

{

guess = EnsureUpper(guess);

++guesses;

isNewGuess = CheckForNewGuess(guess);

if (isNewGuess)

{

guessedLetters += guess;

// check guess

for (i = 0; i < word.Length; ++i)

{

if (guess == word.Substring(i, 1))

{

feedbackLbl.Text = "Yes!! " + guess + " is in the word!!";

for (j = 0; j < answerboard.Length; ++j)

{

if (guess == word.Substring(j, 1))

{

answerboardArray[j] = Convert.ToChar(guess);

answerboard = new string(answerboardArray);

answerboardLbl.Text = answerboard;

}

}

i = word.Length; // force exit since a correct letter found

goodGuess = true;

if (answerboard == word)

gameOver = true;

} // end the if block that checks if good letter and what to do

} // continue on to next letter in the word as needed

if (!goodGuess)

{

feedbackLbl.Text = "Sorry. " + guess + " is not in the word.";

++wrongGuesses; // increment wrong guesses counter

}

} // end if for isNewGuess

else // not a newGuess, but still incrememnt wrong guess counter

++wrongGuesses;

if (!gameOver)

{

feedbackLbl.Text += "\n\nYou have made " + wrongGuesses + " wrong guess";

if (wrongGuesses != 1)

feedbackLbl.Text += "es.";

else

feedbackLbl.Text += ".";

}

else // won

{

wonLblLine2.Text = "You got it in only " + guesses + " guess";

if (guesses != 1)

wonLblLine2.Text += "es!!";

else

wonLblLine2.Text += "!!";

wonLblLine3.Text = "(And you did it with ";

if (wrongGuesses != 0)

wonLblLine3.Text += "only ";

wonLblLine3.Text += wrongGuesses + " wrong guess";

if (wrongGuesses != 1)

wonLblLine3.Text += "es!!)";

else

wonLblLine3.Text += "!!)";

ToggleGameUI(OFF);

ToggleWonScreen(ON); //

} // end won code block

} //end if validEntry

guessTxtbx.Text = ""; // reset the textbox

guessTxtbx.Focus(); // force the focus back to the textbox as a convenience to the user

} // end submitBtn\_Click

//

//

// utility methods

//

//

private void ToggleWelcomeScreen(bool onOff)

{

welcomeLogoLbl.Visible = onOff;

}

private void ToggleGameUI(bool onOff)

{

if (onOff == ON && wordLbl.Visible == false) // keep these always on after first initialiation

{

wordLbl.Visible = onOff;

answerboardLbl.Visible = onOff;

}

startBtn.Visible = !onOff;

guessLbl.Visible = onOff;

guessTxtbx.Visible = onOff;

feedbackLbl.Visible = onOff;

submitBtn.Visible = onOff;

if (onOff == ON)

guessTxtbx.Focus(); // force the focus to the textbox as a convenience to the user

} // end ToggleGameUI()

private void ToggleWonScreen(bool onOff)

{

wonLbl.Visible = onOff;

wonLblLine2.Visible = onOff;

wonLblLine3.Visible = onOff;

wonLblLine4.Visible = onOff;

playAgainLbl.Visible = onOff;

}

private string SelectWord(int min, int max)

{

Random ranNumberGenerator = new Random();

int randomNumber;

randomNumber = ranNumberGenerator.Next(min, max);

string newWord = words[randomNumber];

return newWord;

}

private bool CheckForNewGuess(string newGuess) // check if guessed already

{

bool validGuess = true;

for (i = 0; i < guessedLetters.Length; ++i)

{

if (newGuess == guessedLetters.Substring(i, 1))

{

feedbackLbl.Text = "You already guessed " + newGuess + ".";

i = guessedLetters.Length; // result found; end loop

validGuess = false;

}

}

return validGuess;

} // end CheckForNewGuess()

private bool CheckLength(string newGuess)

{

bool isValidLength = true;

if (newGuess.Length > 1)

{

feedbackLbl.Text = "Invalid entry. Please enter only one letter.";

isValidLength = false;

}

return isValidLength;

}

private bool CheckForLetter(string newGuess)

{

bool isLetter = false;

if (String.Compare(newGuess, "a") >= 0 && String.Compare(newGuess, "z") <= 0)

isLetter = true;

else if (String.Compare(newGuess, "A") >= 0 && String.Compare(newGuess, "Z") <= 0)

isLetter = true;

else feedbackLbl.Text = "Invalid entry. Please enter only letters from A-Z";

return isLetter;

}

private string EnsureUpper(string newGuess)

{

if (String.Compare(newGuess, "a") >= 0 && String.Compare(newGuess, "z") <= 0)

newGuess = newGuess.ToUpper();

return newGuess;

}

} // end class GuessAWord form

} // end namespace GuessAWord