UNIT 8C – CREATING THE DESIGN

By Sophie May

Table of Contents

Assets Table

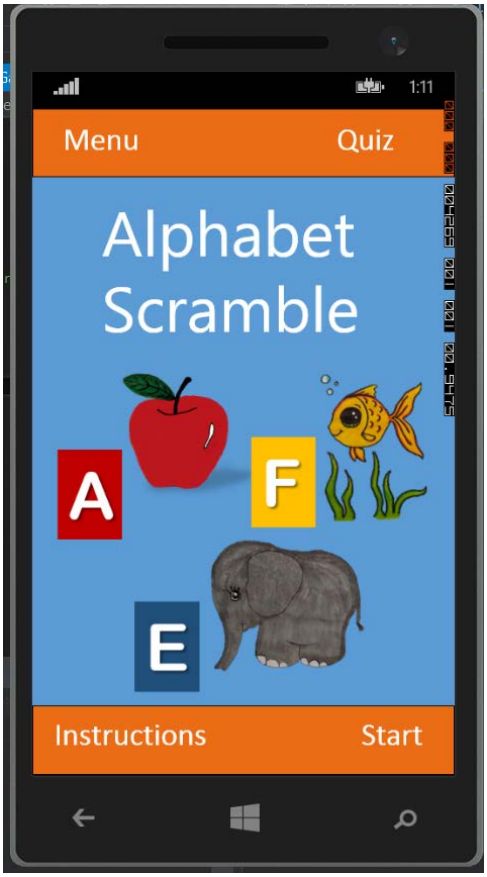
File Name	Description	Source
Apple.PNG	Hand drawn image of an apple edited using photoshop and PowerPoint.	Hand drawn by Sophie May, edited in Photoshop and PowerPoint. Optimised from jpeg to png file. I have optimised all my images using an online image compression tool from the site http://www.imagesmaller.com/ and then I saved the images and put them back into my program.
Balloon.PNG	Hand drawn image of balloons edited using photoshop and PowerPoint.	Hand drawn by Sophie May, edited in Photoshop and PowerPoint. Optimised from jpeg to png file. I have optimised all my images using an online image compression tool from the site http://www.imagesmaller.com/ and then I saved the images and put them back into my program.
Carrots.PNG	Image of happy cartoon carrots sourced from creative commons free images, on open clipart made by cyberscooty.	https://openclipart.org/detail/246055/funny-carrots https://openclipart.org/download/246055/cyberscooty-funny_carrots.svg I have optimised all my images using an online image compression tool from the site http://www.imagesmaller.com/ and then I saved the images and put them back into my program. Optimised from svg to png file.

Duck.PNG	Hand drawn image of a duck edited using photoshop and PowerPoint.	Hand drawn by Sophie May, edited in Photoshop and PowerPoint. Optimised from jpeg to png file. I have optimised all my images using an online image compression tool from the site http://www.imagesmaller.com/ and then I saved the images and put them back into my program.
Elephant.PNG	Hand drawn image of an elephant edited using photoshop and PowerPoint.	Hand drawn by Sophie May, edited in Photoshop and PowerPoint. Optimised from jpeg to png file. I have optimised all my images using an online image compression tool from the site http://www.imagesmaller.com/ and then I saved the images and put them back into my program.
Fish.PNG	Hand drawn image of a fish edited using photoshop and PowerPoint.	Hand drawn by Sophie May, edited in Photoshop and PowerPoint. Optimised from jpeg to png file. I have optimised all my images using an online image compression tool from the site http://www.imagesmaller.com/ and then I saved the images and put them back into my program.
LetterAsound1.mp3	Sound played once picture is clicked. "A is for".	Made by Sophie May using microphone on smartphone.
apple.mp3	Sound played once answer is correct. "Apple"	Made by Sophie May using microphone on smartphone.
LetterBsound1.mp3	Sound played once picture is clicked. Made using microphone on smartphone. "B is for…"	Made by Sophie May using microphone on smartphone.
Balloon.mp3	Sound played once answer is correct. "Balloon"	Made by Sophie May using microphone on smartphone.
LetterCsound1.mp3	Sound played once picture is clicked. Made using microphone on smartphone. "C is for…"	Made by Sophie May using microphone on smartphone.
Carrots.mp3	Sound played once answer is correct. "Carrot"	Made by Sophie May using microphone on smartphone.

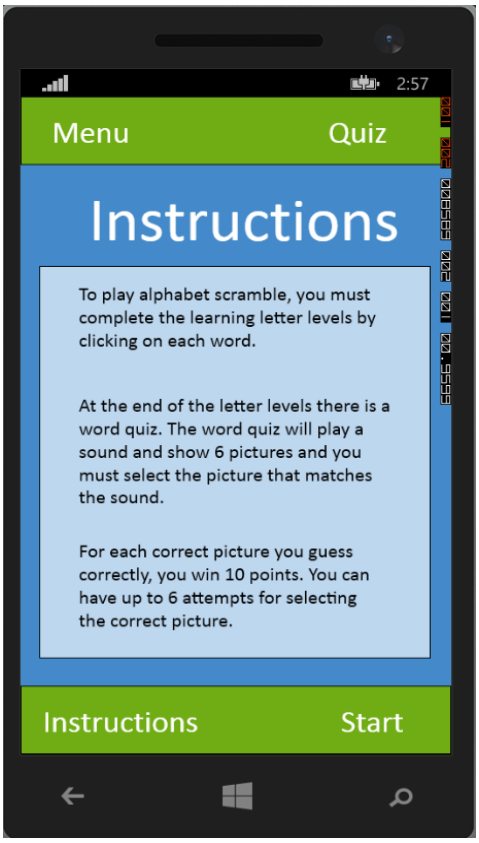
LetterDsound1.mp3	Sound played once picture is clicked. Made using microphone on smartphone. “D is for…”	Made by Sophie May using microphone on smartphone.
Duck.mp3	Sound played once answer is correct. “Duck”.	Made by Sophie May using microphone on smartphone.
LetterEsound1.mp3	Sound played once picture is clicked. Made using microphone on smartphone. “E is for…”	Made by Sophie May using microphone on smartphone.
Elephant.mp3	Sound played once answer is correct. “Elephant”	Made by Sophie May using microphone on smartphone.
LetterFsound1.mp3	Sound played once picture is clicked. Made using microphone on smartphone. “F is for..”	Made by Sophie May using microphone on smartphone.
Fish.mp3	Sound played once answer is correct. “Fish”	Made by Sophie May using microphone on smartphone.
Group_Image_Objects_Main.PNG	Group of images to be displayed on title main menu screen once it loads.	Made by Sophie May, edited in PowerPoint. I have optimised all my images using an online image compression tool from the site http://www.imagesmaller.com/ and then I saved the images and put them back into my program.
All pre-defined code	All pre-defined and altered code provided in this document.	Sourced from tutor, Simon Phillips, altered by Sophie May.

Screenshots

Main Menu Screen

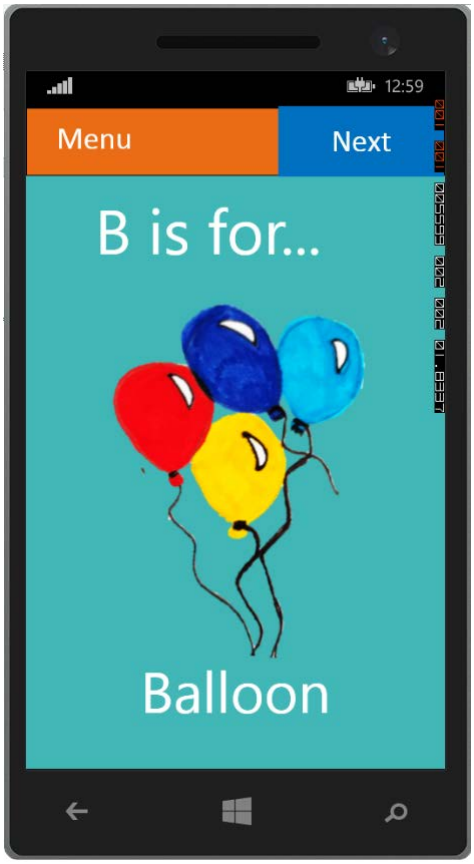


Instructions screen



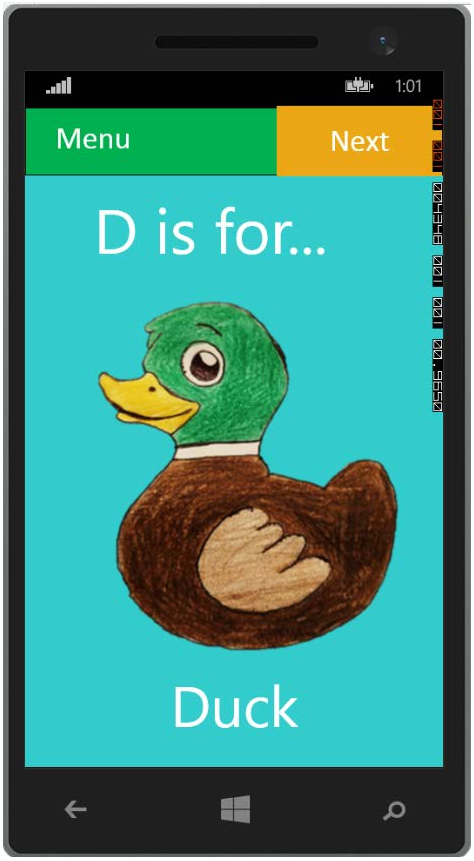
Letter A Learning Screen

Letter B Learning Screen

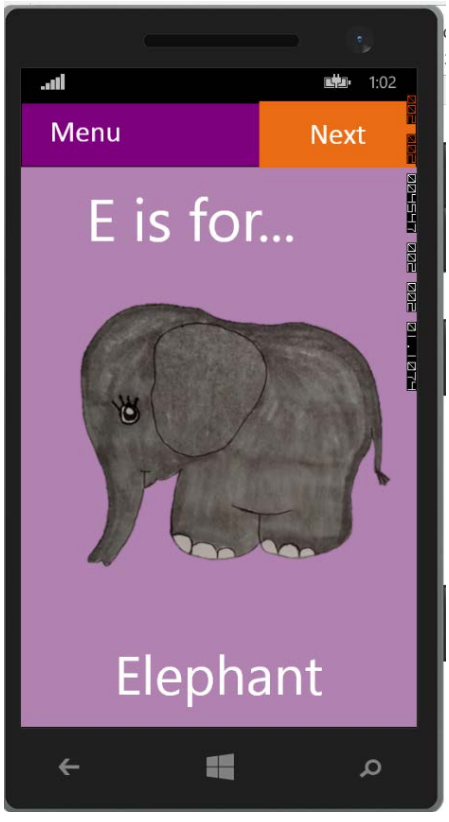


Letter C Learning Screen

Letter D Learning Screen

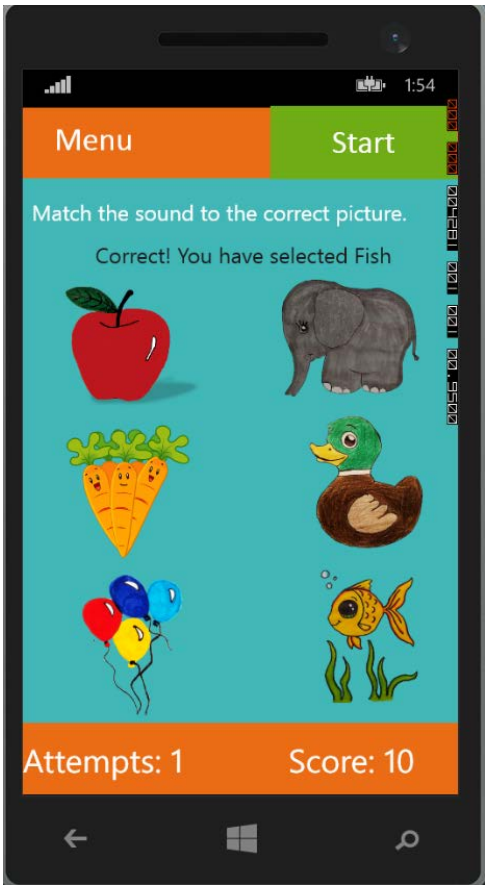


Letter E Learning Screen



Letter F Learning Screen

Quiz Screen



Pre-defined Code

```
        // example method loads list with file names for letter images and sounds

        public void loadImagesSounds()
        {

            listColoursL1.Add("A");
            listColoursL1.Add("B");
            listColoursL1.Add("C");


        }

        //example method to test maximum count has not been reached
        public void checkCount()
        {
            if (check > 4)
            {
                txtCheckCount.Text = "Game Over";
                btnPlay.IsEnabled = false;
                check = 0;

            }
            else
            {
                txtCheckCount.Text = check.ToString() + " attempts taken" + " Score: "
+score.ToString();

            }
        }




       //example method to load letters when play button is pressed
        public void loadImages()
        {
            //set the variable with the first image
            uri1 = new Uri("/Assets/" + listLettersL1[0] + ".png", UriKind.Relative);
            img1.Source = new System.Windows.Media.Imaging.BitmapImage(uri1);

            uri2 = new Uri("/Assets/" + listLettersL1[1] + ".png", UriKind.Relative);
            img2.Source = new System.Windows.Media.Imaging.BitmapImage(uri2);

            uri3 = new Uri("/Assets/" + listLettersL1[2] + ".png", UriKind.Relative);
            img3.Source = new System.Windows.Media.Imaging.BitmapImage(uri3);

            uri4 = new Uri("/Assets/" + listLettersL1[3] + ".png", UriKind.Relative);
            img4.Source = new System.Windows.Media.Imaging.BitmapImage(uri4);


        }



        private void Play_Click(object sender, RoutedEventArgs e)
```

```
        {
           //call the method to load the letters when the play button is pressed
            loadImages();

            //select a random number between 0 and 4
            number = randomIndex.Next(0, 4);

            //select a random number between 0 and 4
            soundFile = listLettersL1[number];
            //set the source code for the sound
            SoundClip.Source = new Uri("/Assets/" + soundFile + ".mp3",
UriKind.Relative);
            //set to play
            SoundClip.AutoPlay = true;
            SoundClip.Play();


        }



       //Code for each letter.  Code responds to a letter being clicked.
        private void img1_MouseLeave(object sender,
System.Windows.Input.MouseEventArgs e)
        {
            //each time the user makes an attempt increase the check variable
            check = check + 1;

            if(soundFile == listLettersL1[3])
            {
                  txtMessage.Text ="You have selected " + listLettersL1[3];

            }
            else
            {
                  txtMessage.Text ="Sorry, You have selected the wrong Letter";
            }

           //set the score on game
            txtScore.Text = "Score: " + score.ToString();

        }
```

Edited Pre-defined Code

```
public partial class Game : PhoneApplicationPage
{
    //declare a list to hold the praise sounds
    List<string> listItemsL2 = new List<string>();

    Uri sound1; //variable for first praise sound
    Uri sound2; //variable for second praise sound

    //declare a list to hold the images
    List<string> listItemsL1 = new List<string>();

    Uri uri1; //variable for first image
    Uri uri2; //variable for second image
    Uri uri3; //variable for third image
    Uri uri4; //variable for fourth image
    Uri uri5; //variable for fifth image
    Uri uri6; //variable for sixth image


    //Used to store the letter sounds. Set as empty
    string soundFile = "";
    //Used to store the praise sounds. Set as empty
    string soundFile2 = "";

    //Used to store a random number
    Random randomIndex = new Random();

    int number; // Used to store numbers
    int score;  //keep a count of the score
    int check;  //keep a count of the number of attempts
    public Game()
    {
        InitializeComponent();
        //call method to load images and sounds
        loadImagesSounds();
        praiseSounds();


    }
    //method loads list with file names for praise sounds
    public void praiseSounds()
    {
        listItemsL2.Add("TryAgain");
        listItemsL2.Add("Correct");
    }
    //method loads list with file names for letter images and sounds
    public void loadImagesSounds()
    {

        listItemsL1.Add("apple");
        listItemsL1.Add("Balloon");
        listItemsL1.Add("Carrots");
        listItemsL1.Add("Duck");
        listItemsL1.Add("Elephant");
        listItemsL1.Add("Fish");
```

```
        }
        //method to test maximum count has not been reached
        public void checkCount()
        {
               //if the number of attempts is equal to 6 and the score is equal to 60
               //then the if statement is run
            if (check == 6 & score == 60)
            {
                //code to disable the start button.
                btnStartQuiz.IsEnabled = false;
                //code to hide the start button.
                btnStartQuiz.Visibility = Visibility.Collapsed;

                //code to show the Next button to the winner screen (btnEnd1).
                btnEnd1.Visibility = Visibility.Visible;
                //code to enable the next button to the winner screen (btnEnd1).
                btnEnd1.IsEnabled = true;

                //code to disable the next button to the winner screen (btnEnd2).
                btnEnd2.IsEnabled = false;
                //code to hide the Next button to the game over screen(btnEnd2).
                btnEnd2.Visibility = Visibility.Collapsed;

                //code to output the total number of attempts, score and the a message
to the user.
                txtCheckCount.Text = "Attempts: " + check.ToString();
                txtScore.Text = "Score: " + score.ToString();
                txtMessage.Text = "Press the Next button";
                //code to disable the sounds when the max number of attempts has been
reached.
                SoundClip.AutoPlay = false;

            }

                //else if the number of attempts less than 6
                //then the else if statement is run
            else if (check < 6)
            {
                //code to show the start button.
                btnStartQuiz.Visibility = Visibility.Visible;
                //code to enable the start button.
                btnStartQuiz.IsEnabled = true;

                //code to hide the Next button to the winner screen (btnEnd1).
                btnEnd1.Visibility = Visibility.Collapsed;
                //code to disable the next button to the winner screen (btnEnd1).
                btnEnd1.IsEnabled = false;

                //code to disable the next button to the winner screen (btnEnd2).
                btnEnd2.IsEnabled = false;
                //code to hide the Next button to the game over screen(btnEnd2).
                btnEnd2.Visibility = Visibility.Collapsed;


                //code to output the total number of attempts, score and the a message
to the user.
                txtCheckCount.Text = "Attempts: " + check.ToString();
                txtScore.Text = "Score: " + score.ToString();

            }
```

```
            //if the number of attempts is more than 6 and the score is less than 60,
the else statement is run.
            else
            {
                //code to disable the start button.
                btnStartQuiz.IsEnabled = false;
                //code to hide the start button.
                btnStartQuiz.Visibility = Visibility.Collapsed;

                //code to hide the next button to the winner screen (btnEnd1).
                btnEnd1.Visibility = Visibility.Collapsed;
                //code to disable the next button to the winner screen (btnEnd1).
                btnEnd1.IsEnabled = false;

                //code to show the next button to the game over screen (btnEnd2).
                btnEnd2.Visibility = Visibility.Visible;
                //code to enable the next button to the game over screen (btnEnd2).
                btnEnd2.IsEnabled = true;

                //code to output the total number of attempts, score and the a message
to the user.
                txtCheckCount.Text = "Attempts: " + check.ToString();
                txtScore.Text = "Score: " + score.ToString();
                txtMessage.Text = "Press the Next button";
                //code to disable the sounds when the max number of attempts has been
reached.
                SoundClip.AutoPlay = false;
            }
        }



       //method to load images when play button is pressed
        public void loadImages()
        {
            //set the variable with the first image
            uri1 = new Uri("/Assets/" + listItemsL1[0] + ".png", UriKind.Relative);
            image1.Source = new System.Windows.Media.Imaging.BitmapImage(uri1);

            //set the variable with the second image
            uri2 = new Uri("/Assets/" + listItemsL1[1] + ".png", UriKind.Relative);
            image2.Source = new System.Windows.Media.Imaging.BitmapImage(uri2);

            //set the variable with the third image
            uri3 = new Uri("/Assets/" + listItemsL1[2] + ".png", UriKind.Relative);
            image3.Source = new System.Windows.Media.Imaging.BitmapImage(uri3);

            //set the variable with the fourth image
            uri4 = new Uri("/Assets/" + listItemsL1[3] + ".png", UriKind.Relative);
            image4.Source = new System.Windows.Media.Imaging.BitmapImage(uri4);

            //set the variable with the fifth image
            uri5 = new Uri("/Assets/" + listItemsL1[4] + ".png", UriKind.Relative);
            image5.Source = new System.Windows.Media.Imaging.BitmapImage(uri5);

            //set the variable with the sixth image
            uri6 = new Uri("/Assets/" + listItemsL1[5] + ".png", UriKind.Relative);
            image6.Source = new System.Windows.Media.Imaging.BitmapImage(uri6);

        }
```

```
        private void btnStartQuiz_Click(object sender, RoutedEventArgs e)
        {
            //call the method to load the images when the play button is pressed
            loadImages();


            //select a random number between 0 and 6
            number = randomIndex.Next(0, 6);

            //selects a sound at random, a Uri number between 0 and 6
            soundFile = listItemsL1[number];

            //set the source code for the sound
            SoundClip.Source = new Uri("/Assets/" + soundFile + ".mp3",
UriKind.Relative);
            //sets the sound to play
            SoundClip.AutoPlay = true;
            SoundClip.Play();
        }

          private void image1_Tap(object sender, System.Windows.Input.GestureEventArgs
e)
        {
            //each time the user makes an attempt increase the check variable
            check = check + 1;

            if (soundFile == listItemsL1[0])
            {
                //displays a message to the user
                txtMessage.Text = "Correct! You have selected " + listItemsL1[0];
                //assigns the soundfile to play the sound number1
                soundFile2 = listItemsL2[1];

                //set the source code for the sound
                SoundClip.Source = new Uri("/Assets/" + soundFile2 + ".mp3",
UriKind.Relative);
                //set to play
                SoundClip.AutoPlay = true;
                SoundClip.Play();

                //code to enable the start button.
                btnStartQuiz.IsEnabled = true;
                //code to disable the Next buttons to the winner screen (btnEnd1) and
the game over screen(btnEnd2).
                btnEnd1.IsEnabled = false;
                btnEnd2.IsEnabled = false;

                //code to show the start button.
                btnStartQuiz.Visibility = Visibility.Visible;
                //code to hide the Next buttons to the winner screen (btnEnd1) and the
game over screen(btnEnd2).
                btnEnd1.Visibility = Visibility.Collapsed;
                btnEnd2.Visibility = Visibility.Collapsed;

                //code to add 10 points to the score.
                score = score + 10;
            }
            else
            {
                //code to display a message to the user.
                txtMessage.Text = "Sorry, try again!";
                //assigns the soundfile to play the sound number 0
```

```
                soundFile2 = listItemsL2[0];

                //set the source code for the sound
                SoundClip.Source = new Uri("/Assets/" + soundFile2 + ".mp3",
UriKind.Relative);
                //sets the sound to play
                SoundClip.AutoPlay = true;
                SoundClip.Play();

            }
            //set the score on game
            txtScore.Text = "Score: " + score.ToString();
            //call the method to check how many attempts have been made
            checkCount();
        }
```

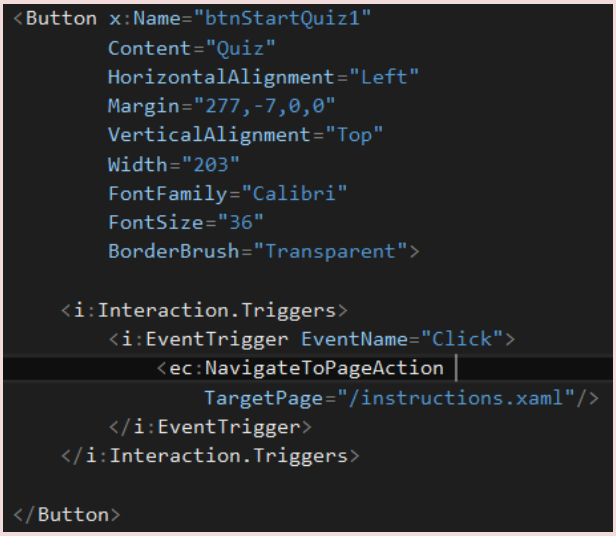
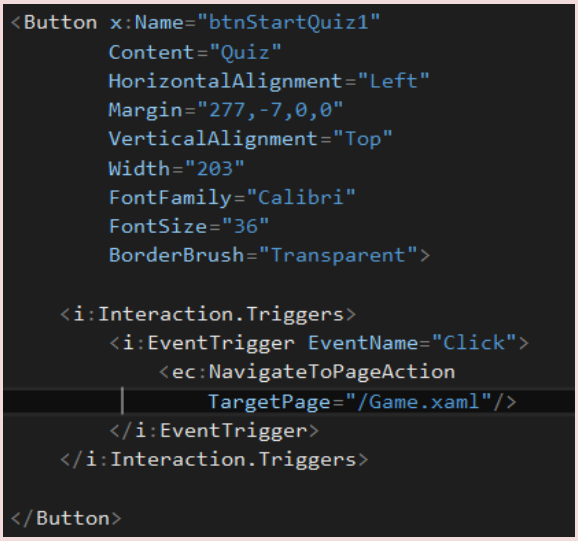
Test Plan

Test No.	Description	Test Data	Expected Result	Actual Result
1	Test that the text buttons work (such as Menu, Instructions start, quiz buttons).	Run the program and manually click on each button.	All text buttons will operate as intended once the user has clicked on them.	All buttons are functioning as I expected except for the "Quiz" button on the main menu which is navigating to the instructions screen instead of the quiz screen. I have corrected this fault and the quiz button now directs the user to the quiz screen as it should.

2	Test that the sound on each level (6 in total) works when the user clicks on the images.	Run the program and manually click on each image on all six levels to check that the sound plays as intended.	I expect that all the sounds are linked to the correct images on the right levels and the sound plays once the user clicks on the image.	All the sounds work as expected.
3	Proof-read and spellcheck the text content of the alphabet game to make sure no errors are present.	Run the program and click through each level, reading the text on every level.	The written content of the game including buttons, instructions and answers will contain zero errors.	There are no errors in the written content of the game.
4	Graphics and images should be optimised for performance and checked for quality.	Run the program and click through each level, checking the images as you progress.	The images and graphics will be optimised to reduce loading times and will be compressed to increase the quality of the graphics.	The images have been optimised.
5	Audio and sounds will be checked and tested for their quality.	Run the program and click through each image on each of the 6 levels to make sure the sound is not quiet or muted.	The sounds and audio will be audible and clear so that the audience can hear it properly and the sounds will play once the user clicks on each image on each level of the game.	The audio is audible and clear.
6	Check that each level works correctly and is able to be completed.	Run the program and complete each level.	The game will run smoothly, and each level will run in sequence (screens 1-7) and the levels will run correctly as designed.	All levels operate as anticipated.

7	Check that the total score is counted correctly.	Run the program and complete each level.	The game will count the score correctly as the user progresses through each level.	The score is totalled correctly as expected.
8	Check that the attempts are counted correctly.	Run the program and complete each level.	The game will count the number of attempts and the score correctly as the user progresses through each level.	The attempts are counted correctly as expected.
9	Check that the answer for each level is correct.	Run the program and complete each level.	The game will run smoothly, and each answer will be linked to the correct level.	The answer for each level is correct.
10	Check that the error messages display when the user inputs incorrect answers.	Run the program and complete each level incorrectly.	The game will display an error message each time the user inputs the wrong answer.	The error messages appear as expected.

Faults and Repairs

Issue	Before	After
The "Quiz" button on the main menu screen is not operating as it should, it is directing the user to the instructions screen instead of the Quiz screen.	 On the Quiz button the navigation action TargetPage is set to 'instructions.xaml' instead of Game.xaml.	 TargetPage is now set to 'Game.xaml' and the error is now fixed.

The program is not loading the images to the quiz screen.

```
//declare a list to hold the images
List<int> list = new List<int>();

Uri uri1; //variable for first image
Uri uri2; //variable for second image
Uri uri3; //variable for third image
Uri uri4; //variable for fourth image
Uri uri5; //variable for fifth image
Uri uri6; //variable for sixth image
```

```
//method loads list with file names for letter images and sounds
1 reference
public void loadImagesSounds()
{

    listItemsL1.Add("apple");
    listItemsL1.Add("Balloon");
    listItemsL1.Add("Carrots");
    listItemsL1.Add("Duck");
    listItemsL1.Add("Elephant");
    listItemsL1.Add("Fish");

}
```

The program is not loading the images as the list to hold the images is not the correct variable name for the list. The list is declared under the variable name “list” instead of “listItemsL1”.

```
//declare a list to hold the images
List<int> listItemsL1 = new List<int>();

Uri uri1; //variable for first image
Uri uri2; //variable for second image
Uri uri3; //variable for third image
Uri uri4; //variable for fourth image
Uri uri5; //variable for fifth image
Uri uri6; //variable for sixth image
```

```
//method loads list with file names for letter images and sounds
1 reference
public void loadImagesSounds()
{

    listItemsL1.Add("apple");
    listItemsL1.Add("Balloon");
    listItemsL1.Add("Carrots");
    listItemsL1.Add("Duck");
    listItemsL1.Add("Elephant");
    listItemsL1.Add("Fish");
```

The program is now loading the images correctly as the list to hold the data is now under the correct variable name. The list has been correctly declared as “listItemsL1”.

The program will not load the images.

```
//declare a list to hold the images
List<int> list = new List<int>();

Uri uri1; //variable for first image
Uri uri2; //variable for second image
Uri uri3; //variable for third image
Uri uri4; //variable for fourth image
Uri uri5; //variable for fifth image
Uri uri6; //variable for sixth image
```

```
//method loads list with file names for letter images and sounds
1 reference
public void loadImagesSounds()
{

    listItemsL1.Add("apple");
    listItemsL1.Add("Balloon");
    listItemsL1.Add("Carrots");
    listItemsL1.Add("Duck");
    listItemsL1.Add("Elephant");
    listItemsL1.Add("Fish");
```

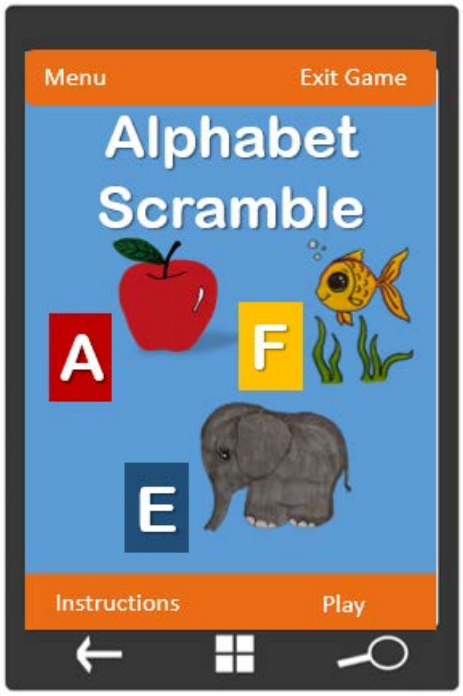
The data type for the list has been set to integer instead of string which means the program can’t locate the images and sounds as they are listed in a string format instead of an integer format.

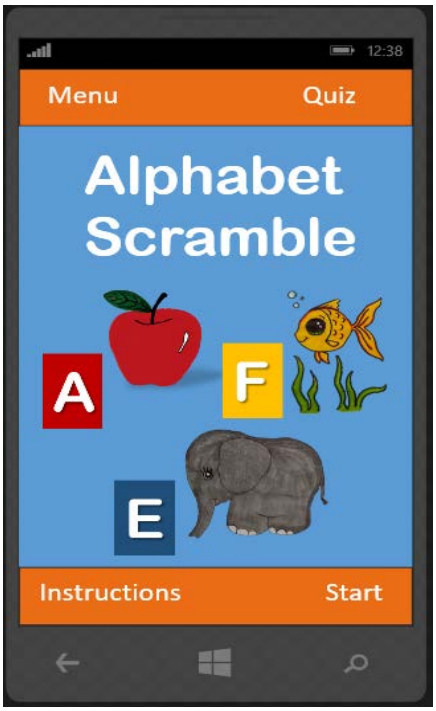
```
//declare a list to hold the images
List<string> listItemsL1 = new List<string>();

Uri uri1; //variable for first image
Uri uri2; //variable for second image
Uri uri3; //variable for third image
Uri uri4; //variable for fourth image
Uri uri5; //variable for fifth image
Uri uri6; //variable for sixth image
```

```
//method loads list with file names for letter images and sounds
1 reference
public void loadImagesSounds()
{

    listItemsL1.Add("apple");
    listItemsL1.Add("Balloon");
    listItemsL1.Add("Carrots");
    listItemsL1.Add("Duck");
    listItemsL1.Add("Elephant");
    listItemsL1.Add("Fish");
```

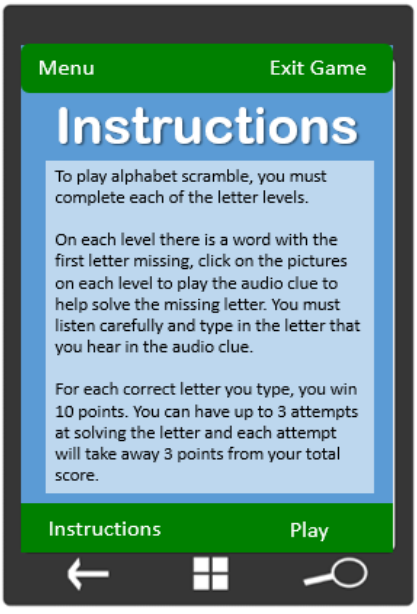
The data type for the list has been corrected to string and now the program can locate the data.

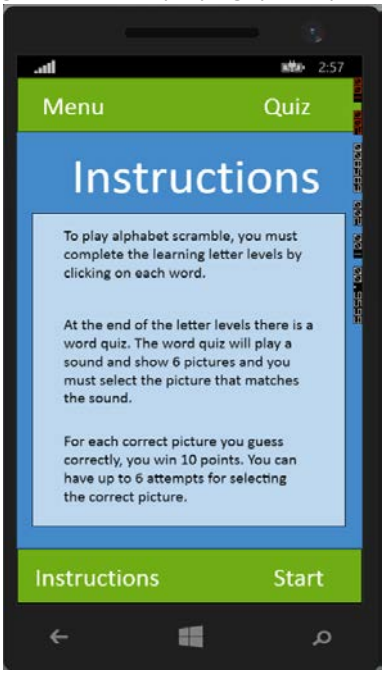
Changes



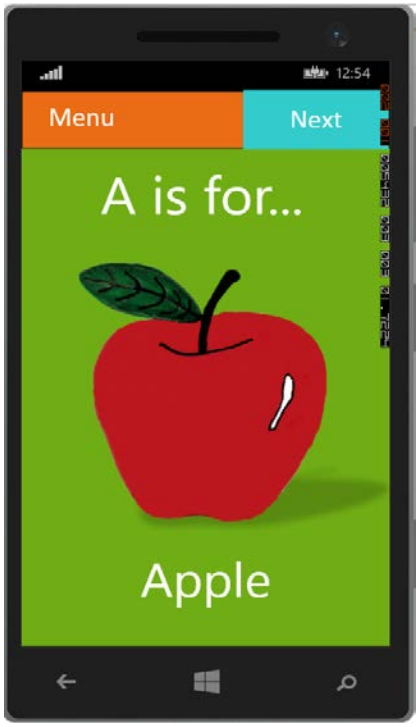


In my original design (shown left) I had an “exit game” button and a “start” button on the main menu, in my mobile application (shown right) I had to add an additional screen called “quiz” as part of the code required a list element to be included and I struggled to include it in any of the other screens so I created a new screen and the quiz button is to navigate the user directly to the quiz screen. The start button has been added to avoid confusion between the learning aspect of the game and the playing quiz aspect of the game.

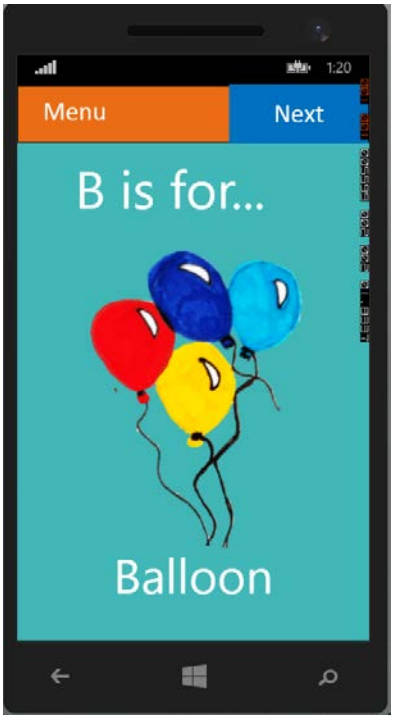




In my original design (shown left) I had an exit game button and a play button, I changed these two buttons from “play” to “start” and “exit game” to “quiz” again due to changes I had to make to my finished product. I had to include a quiz screen in my application due to difficulties adding in a list element to any of my original design screens. The quiz button now navigates the user to the quiz screen and the start button starts the learning feature screens of the game. As you can see I also altered the shade of green on the menu bars from a dark green to light green.

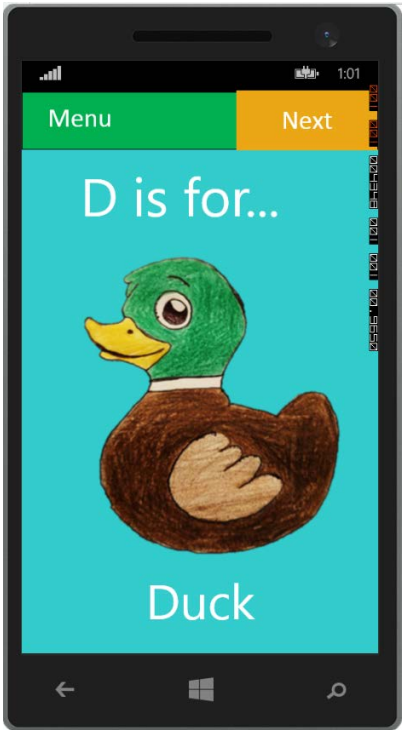


The original design (shown left) has a message prompt across the screen (click on the image to play the word), an attempts counter, a score counter and a letter input field, I found this design almost impossible to code and could figure out how to fit a list element into this design. Therefore, I created a quiz screen so that I could include the list element in my design without having to re-design the whole application and simply removed the attempts counter, score counter and letter input field from my learning screens A-F (shown right).

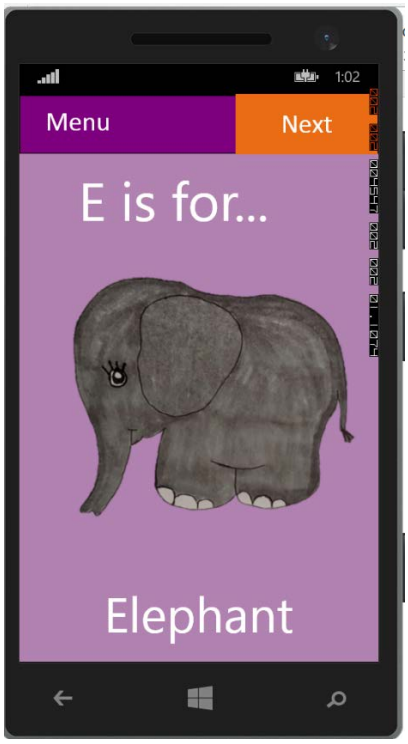


The original design (shown left) has a message prompt across the screen (click on the image to play the word), an attempts counter, a score counter and a letter input field, I found this design almost impossible to code and could figure out how to fit a list element into this design. Therefore, I created a quiz screen so that I could include the list element in my design without having to re-design the whole application and simply removed the attempts counter, score counter and letter input field from my learning screens A-F (shown right).

The original design (shown left) has a message prompt across the screen (click on the image to play the word), an attempts counter, a score counter and a letter input field, I found this design almost impossible to code and could figure out how to fit a list element into this design. Therefore, I created a quiz screen so that I could include the list element in my design without having to re-design the whole application and simply removed the attempts counter, score counter and letter input field from my learning screens A-F (shown right).

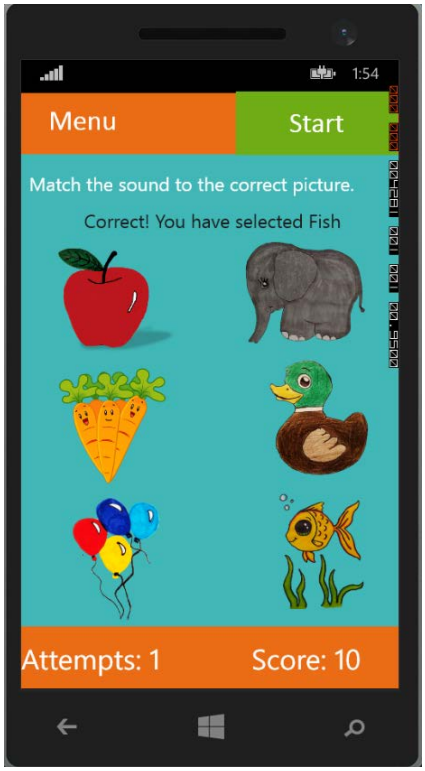


The original design (shown left) has a message prompt across the screen (click on the image to play the word), an attempts counter, a score counter and a letter input field, I found this design almost impossible to code and could figure out how to fit a list element into this design. Therefore, I created a quiz screen so that I could include the list element in my design without having to re-design the whole application and simply removed the attempts counter, score counter and letter input field from my learning screens A-F (shown right).

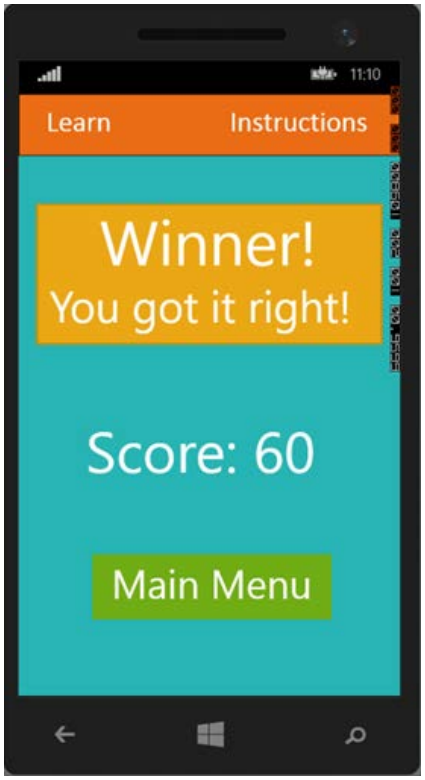


The original design (shown left) has a message prompt across the screen (click on the image to play the word), an attempts counter, a score counter and a letter input field, I found this design almost impossible to code and could figure out how to fit a list element into this design. Therefore, I created a quiz screen so that I could include the list element in my design without having to re-design the whole application and simply removed the attempts counter, score counter and letter input field from my learning screens A-F (shown right).

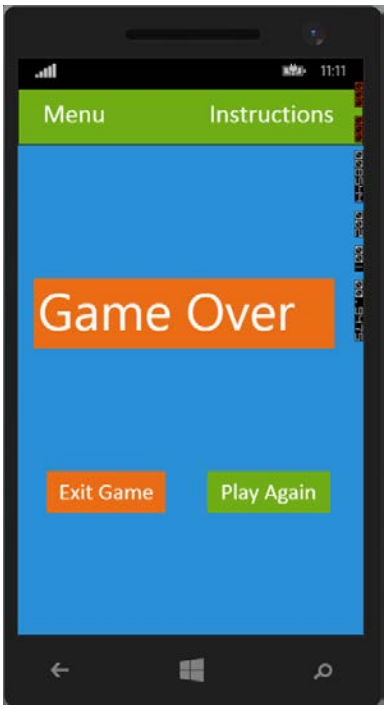
The original design (shown left) has a message prompt across the screen (click on the image to play the word), an attempts counter, a score counter and a letter input field, I found this design almost impossible to code and could figure out how to fit a list element into this design. Therefore, I created a quiz screen so that I could include the list element in my design without having to re-design the whole application and simply removed the attempts counter, score counter and letter input field from my learning screens A-F (shown right).



This is a screenshot of the new screen of my “quiz” game in my application, this is the quiz screen, it works by playing an audio file of a word such as “apple” and the user must select the correct image that corresponds with that audio clip. This screen contains the attempts counter and the score counter.



This is the winner screen that I have included as part of the refinements of this program. I originally planned to have this in my initial design but had to scrap the screen as I did not have any idea on how to program it into my game, but after the feedback was given to me by James Saunders, I decided to add the winner screen back in.



This is the game over screen that I have included as part of the refinements of this program. I originally planned to have this in my initial design but had to scrap it as I did not have any idea on how to program the screen into my game, but after the feedback was given to me by James Saunders, I decided to add the game over screen back in.

Optimised Assets

I have optimised all my images using an online image compression tool from the site http://www.imagesmaller.com/ and then I saved the images and put them back into my program.

Constructs

1. From main menu user selects instructions button and reads instructions.
2. From main menu user clicks start button to navigate to the learning screens.
3. User navigates through learning screens A-F.
4. User clicks on the quiz button from either the main menu or the last learning screen (screen F).
5. User is then directed to quiz screen where a prompt message is displayed to the user “Press start to play”.
6. Audio sound file is played to the user, such as “apple”.
7. User selects an image that correlates to the audio sound.
8. If the correct image is selected the attempts counter goes up by 1 and the score goes up by 10, an audio clip saying the user is correct is played and the user may press the start button to play the next audio clip.
9. Else if the wrong image is selected an error message is displayed to the user “Sorry, please try again!” and an audio clip is play asking the user to try again.
10. The user is allowed up to 6 attempts before a message is prompted to the user asking them to select the “next” button and the user is then directed to either the winner screen or the game over screen depending on what the user scored.
11. If they scored 60 points they will be directed to the winner screen.
12. If they scored less than 60 points (50 points or less), the user will be directed to the game over screen.

Peer Feedback

Name of Reviewer: James Saunders

Date: 14th May 2018

How easy did you find it to navigate around the app?

Yes the app is easy to navigate.

Does the app work how you would expect it to work?

Yes for the most part, however, the attempts counter needs a little more work as the program does not show the maximum number of attempts (6).

Is the audio clear?

Yes.

Does it give appropriate messages when the user has/has not succeeded?

Yes however audio clips reading these messages would also be nice.

Do you think it would be suitable for children trying to learn to read?

Yes, the audio is clear and the colours are nice.

What do you think about the design and can you make any suggestions to improve it?

On the quiz page I would suggest adding an additional start button in the centre that disappears once start has been pressed. I would also say add a completion game over screen/feedback screen.

Review of Peer Feedback

In the feedback James stated that they attempts counter was faulty and would not show the maximum number of attempts once reached, this however was not initially programmed into the application as it was not meant to display this data to the user, once the number of attempts was reached the program was only meant to display the game over message to the user. This aspect is to tell the user when they have reached the limit and cannot progress any further.

Another thing that James mentioned was that he would like to hear audio clips reading the feedback messages to the user during the quiz. I have since refined the program to add in these additional audio clips which read out the phrases “correct” and “try again” during the quiz and can be seen in my refinements table below.

James has also said he would like to see an additional start button in the centre that disappears once the start button has been pressed, however I do not like this idea as the start button is on the main menu bar at the top of the quiz and I feel as though it should not be moved from this position. The button is in an

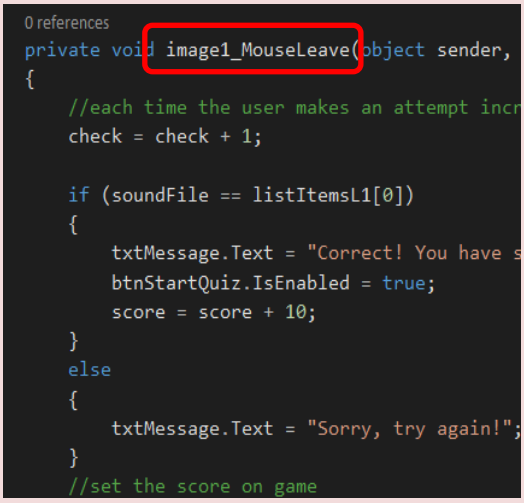
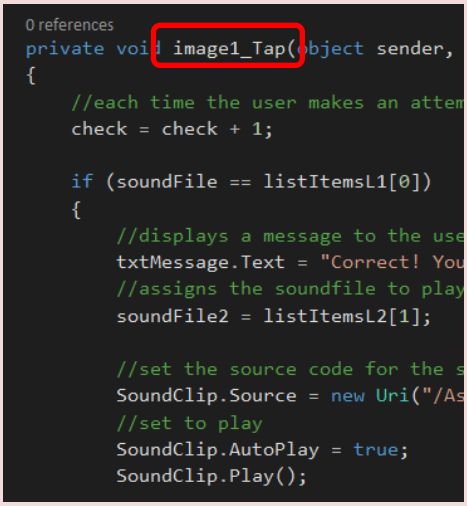
intuitive position on the menu bar, I do not wish to change this layout when the user will have already become used to the layout on the learning screens.

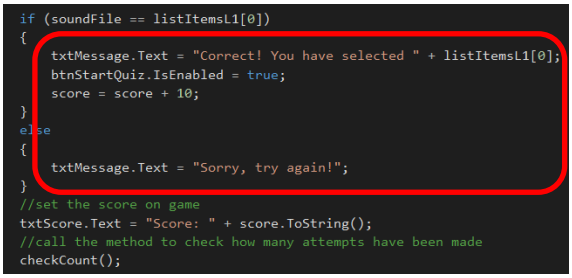
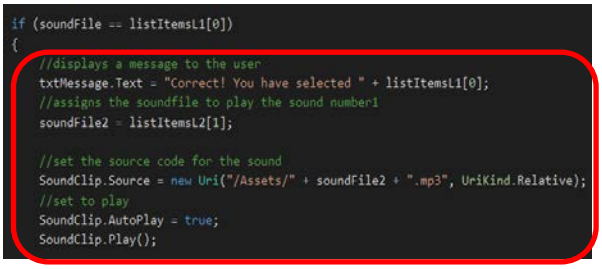
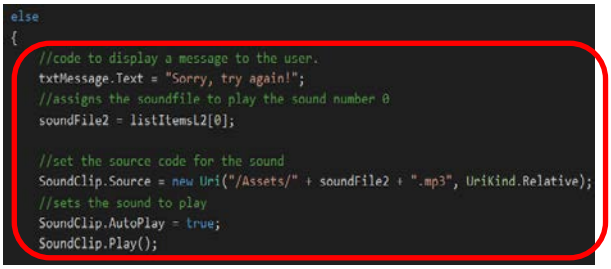
Finally, James has said that he would like additional feedback screens to display to the user upon completion of the game. Therefore, I decided to include the two feedback screens from my original design, in the refinement stage of this project and the application now directs the user to one of two feedback screens upon completion of the game.

The game directs the user to either the winner screen or the game over screen at the end of the quiz depending on the overall score that the user got during the quiz. If the user gets a perfect 60 point score in 6 attempts, the user is directed to the winner screen. If however, the user gets 50 points or less in 6 attempts, the user is then directed to the game over screen.

Refinements

No.	Description	Before	After

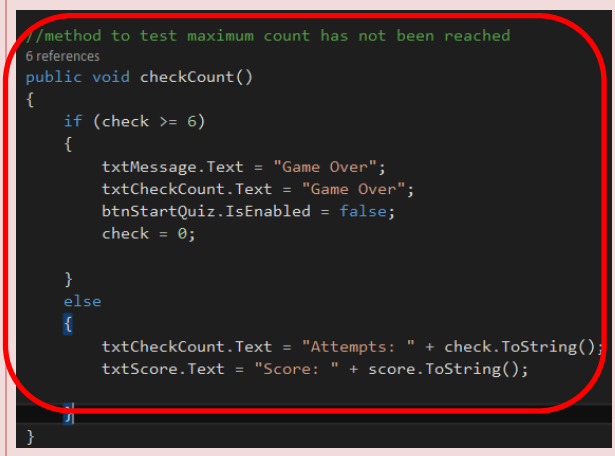
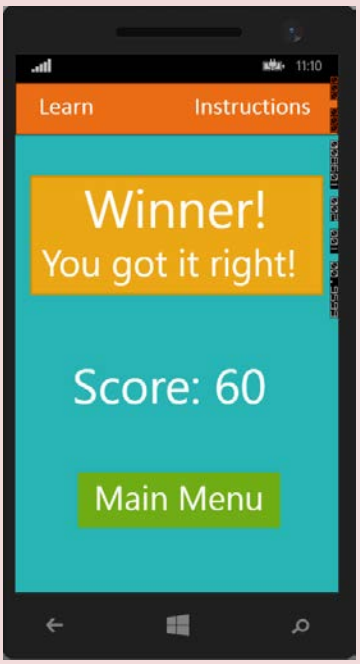
1	I refined the "mouseleave" action to a "tap" action on the images for the quiz screen.
	
	This was for two reasons, the first was that the tap action is more fitting for a mobile phone application as on a phone the user would not be able to click on the image they would need to be able to "tap" it.
	
	The second reason was that sometimes the program would not acknowledge when the mouse had left the correct image and again I think this was due to the action
		
		0 references
		private void image1_MouseLeave(object sender,
		{
		//each time the user makes an attempt incr
		check = check + 1;
		
		if (soundFile == listItemsL1[0])
		{
		txtMessage.Text = "Correct! You have s
		btnStartQuiz.IsEnabled = true;
		score = score + 10;
		}
		else
		{
		txtMessage.Text = "Sorry, try again!";
		}
		//set the score on game
			
			0 references
			private void image1_Tap(object sender,
			{
			//each time the user makes an attem
			check = check + 1;
			
			if (soundFile == listItemsL1[0])
			{
			//displays a message to the use
			txtMessage.Text = "Correct! You
			//assigns the soundfile to play
			soundFile2 = listItemsL2[1];
			
			//set the source code for the s
			SoundClip.Source = new Uri("/As
			//set to play
			SoundClip.AutoPlay = true;
			SoundClip.Play();

	being set to mouseleave instead of tap, in any case, I felt it was a better idea to change it to “tap” and the program runs much more smoothly and accurately as a result.		
2	I refined the program by adding feedback sound clips to the program so that when the user selects the correct image on the quiz screen, audio praise is played to the user which says “correct” to the user. Additionally, when the user selects the wrong image, the words “try again” are played to the user.	 In this screenshot you can see the only feedback being output to the user was a text message which says the user is correct when they guess the right image and a message saying try again when they guess the wrong image.	  In these two screenshots you can see there is a new sound file called sound file 2 which contains the two feedback sounds for correct and try again, these two sounds have been stored into a new list, called listItemsL2. The source is then listed underneath and the sound has been set to automatically play once the user has met the specified conditions.

```
if (soundFile == listItemsL1[0])
{
    txtMessage.Text = "Correct! You have selected " + listItemsL1[0];
    btnStartQuiz.IsEnabled = true;
    score = score + 10;
}
else
{
    txtMessage.Text = "Sorry, try again!";
}
//set the score on game
txtScore.Text = "Score: " + score.ToString();
//call the method to check how many attempts have been made
checkCount();
```

```
if (soundFile == listItemsL1[0])
{
    //displays a message to the user
    txtMessage.Text = "Correct! You have selected " + listItemsL1[0];
    //assigns the soundfile to play the sound number1
    soundFile2 = listItemsL2[1];

    //set the source code for the sound
    SoundClip.Source = new Uri("/Assets/" + soundFile2 + ".mp3", UriKind.Relative);
    //set to play
    SoundClip.AutoPlay = true;
    SoundClip.Play();
```

```
else
{
    //code to display a message to the user.
    txtMessage.Text = "Sorry, try again!";
    //assigns the soundfile to play the sound number 0
    soundFile2 = listItemsL2[0];

    //set the source code for the sound
    SoundClip.Source = new Uri("/Assets/" + soundFile2 + ".mp3", UriKind.Relative);
    //sets the sound to play
    SoundClip.AutoPlay = true;
    SoundClip.Play();
```

3	I refined the program by adding two extra screens to the program which are the feedback screens, I originally planned to have them in my application but struggled to program them at first so I decided to scrap them and rely on text messages to relay feedback to the user. However, I managed to work out a way in which to program these two screens into the application and after a lot of tweaks to the program, I managed to construct two screens linked to two button actions to direct the user to the feedback screens.	 This (pictured above) is the screen that would only display the text message to the user if they got the answer  wrong. This (pictured above) is the code to check if the maximum number of attempts have been reached, the if and else conditions state what the program will do if the statement conditions are met. If the if statement conditions are met then the application would simply display two messages that say game over and the start button will be disabled. If the else statement conditions are met then the application will increase the number of attempts by 1 and add it to the attempts counter and increase the score by 10 points and add it to the total score and display this data to the user.	 

```
//method to test maximum count has not been reached
6 references
public void checkCount()
{
    if (check >= 6)
    {
        txtMessage.Text = "Game Over";
        txtCheckCount.Text = "Game Over";
        btnStartQuiz.IsEnabled = false;
        check = 0;

    }
    else
    {
        txtCheckCount.Text = "Attempts: " + check.ToString();
        txtScore.Text = "Score: " + score.ToString();

    }
}
```

```
ethod to test maximum count has not been reached
erences
lic void checkCount()

   //if the number of attempts is equal to 6 and the score is equal to 60
   //then the if statement is run
if (check == 6 & score == 60)
{
   //code to disable the start button.
   btnStartQuiz.IsEnabled = false;
   //code to hide the start button.
   btnStartQuiz.Visibility = Visibility.Collapsed;

   //code to show the Next button to the winner screen (btnEnd1).
   btnEnd1.Visibility = Visibility.Visible;
   //code to enable the next button to the winner screen (btnEnd1).
   btnEnd1.IsEnabled = true;

   //code to disable the next button to the winner screen (btnEnd2).
   btnEnd2.IsEnabled = false;
   //code to hide the Next button to the game over screen(btnEnd2).
   btnEnd2.Visibility = Visibility.Collapsed;
```

```
    //else if the number of attempts less than 6
    //then the else if statement is run
else if (check < 6)
{
   //code to show the start button.
   btnStartQuiz.Visibility = Visibility.Visible;
   //code to enable the start button.
   btnStartQuiz.IsEnabled = true;

   //code to hide the Next button to the winner screen (btnEnd1).
   btnEnd1.Visibility = Visibility.Collapsed;
   //code to disable the next button to the winner screen (btnEnd1).
   btnEnd1.IsEnabled = false;

   //code to disable the next button to the winner screen (btnEnd2).
   btnEnd2.IsEnabled = false;
   //code to hide the Next button to the game over screen(btnEnd2).
   btnEnd2.Visibility = Visibility.Collapsed;
```

```
//if the number of attempts is more than 6 and the score is less than 60
//the else statement is run.
else
{
   //code to disable the start button.
   btnStartQuiz.IsEnabled = false;
   //code to hide the start button.
   btnStartQuiz.Visibility = Visibility.Collapsed;

   //code to hide the next button to the winner screen (btnEnd1).
   btnEnd1.Visibility = Visibility.Collapsed;
   //code to disable the next button to the winner screen (btnEnd1).
   btnEnd1.IsEnabled = false;

   //code to show the next button to the game over screen (btnEnd2).
   btnEnd2.Visibility = Visibility.Visible;
   //code to enable the next button to the game over screen (btnEnd2).
   btnEnd2.IsEnabled = true;
```

The three screenshots of code (pictured above) is the code to check if the attempts conditions have been reached, the if and else conditions state what the program will do if the statement conditions are met.

			If the if statement conditions are met (attempts equal 6 and the score equals 60) then the application will enable the next (End1) button for program to navigate the user to the winner screen and hide all other buttons. If the elseif statement conditions are met (the attempts equal less than 6) then the application will enable the start quiz button for the program to allow the user to keep making guesses and it will hide all other buttons. If the else statement conditions are met (attempts equal 6 but the score is 50 or less) then the application will enable the next (End2) button for the program to navigate the user to the game over screen and hide all other buttons. These statements have been created to allow the button actions to direct the user to the feedback screens which enabled me to be able to code them into the program, the code still operates very similarly to how it originally did but the buttons actions have been added and the elseif statement included.