CISC 110, Fall 2012, Final Project User Manual

Name(s): _____

Student Number(s): _____

Project Name: _____

Description (what the project does and how to use it)

The concept of this game is to fly the helicopter using the up and down arrow keys and collect the different gems, while avoiding the birds. If a bird is hit the game is over. The player can replay the game as many times as he or she would like trying to beat their high score. The project is made within Adobe Flash CS5.5 and all files and images (symbols) are located within the library so no additional files or programs must be opened for the game to work.

Required Flash Animation and ActionScript Features

(Symbol, Variable, and Function names, Locations, Actions, Purposes)

- **Motion Tweens**
  - **Location:** instructions folder on the main timeline on the instructions layer and button layer.
    - **Action:** The instructions image alpha starts at 0% and increase to 100% as it moves up the stage. The button (main menu) starts with its alpha at 0% and increases to 100%.
  - **Location:** on the main timeline in the gameoverpage folder on the title layer.
    - **Action:** The title “Game Over” rotates on the page when the player reaches the page by hitting a bird in the game.
  - **Location:** the credit page on the main timeline.
    - **Action:** The credit image’s alpha starts at 0% and increases to 100% as it moves up the stage and the main menu button’s alpha increase from 0% to 100%.
  - **Location:** In the library there are two more symbols that contain Motion Tweens in the game folder
    - **Action:** In the helicopter symbol the tail blades and blades layer both have Motion Tweens rotating them so they appear to be spinning when the game runs
    - **Action:** The skyline symbol also contains Motion Tweens through

the clouds. Layers cloud1, clouds2, and cloud3 move across the sky and return to their original position, so that during the game the clouds are constantly floating through the sky.

- **Shape Tweens**
  - **Location: i**n the library under the game folder in the cloud symbols
    - **Action:** Symbols cloud1, cloud2, and cloud3 each change shape to indicate that like real clouds they are constantly changing.

- **Layers and Symbols**
  - **MovieClip Symbol Names:** cloud1, cloud2, cloud3, skyline, helicopter
  - **Button Symbol Names:** play_btn, instruction_btn, credit_btn, main_btn, main2_btn, replay_btn
  - **Locations of Layers - which timelines:** in the main timeline and in the timelines for cloud1, cloud2, cloud3, skyline, and helicopter

- **Composite MovieClips and MovieClips with their Own Timelines**
  - **Folder Name and List of MovieClips in it with their Own Timeslines:** In the game folder, cloud1, cloud2, cloud3, skyline, and helicopter symbols all have their own timelines.
  - **Folder Name and List of MovieClips in it with their Own Timeslines:** In the gameoverpage folder, the gameoverpage symbol contains a timeline
  - **Folder Name and List of MovieClips in it with their Own Timeslines:** In the homepage folder, the homepage symbol contains a timeline.
  - **Composite MovieClip Name and MovieClips Within It:** the helicopter symbol is composed of a body and tailBlade movie clips
  - **Composite MovieClip Name and MovieClips Within It:** the skyline symbol is composed of a background and cloud movie clips

- **MouseEvents**
  - **Button Name & Location(s):** play_btn on Frame 1
    - **Action:** takes the player to the game
  - **Button Name & Location(s):** instruction_btn on Frame 1
    - **Action:** takes the player to the instructions page
  - **Button Name & Location(s):** credit_btn on Frame 1
    - **Action:** takes the player to the credit page
  - **Button Name & Location(s):** main_btn on Frame 75
    - **Action:** takes the player to the homepage
  - **Button Name & Location(s):** replay_btn on Frame 120
    - **Action:** takes the player back to the game
  - **Button Name & Location(s):** main2_btn on Frame 186
    - **Action:** takes the player to the homepage

- **A KeyboardEvent**
  - **Handler Function Name:** onKeyDown
    - **How it is Used and What it Does:** A KeyboardEvent is used to allow the helicopter to move up and down on the stage to collect gems and avoid Birds. This event also calls the checkForHit function, which uses the hitTestObject method to check if the helicopter has hit the gems or birds. The checking is done by going through the array of gems and birds.
    - When the helicopter hits a gem, the gem is removed from its array and from the display list, and points are added to the "Score" variable. The number of points added differs depending on the color. Highest to lowest: red, green, blue, yellow
    - When the helicopter hits a bird, the timeline then goes to label3, the game over page. The gameOver function is called to remove listeners and objects before going to label4, the replay page

- **If-else statements**
  - **In Which Function:** in the onKeyDown function
    - **Action:** An if else statement is used for the movement of the helicopter up and down the stage, so it's able to collect gems and avoid birds. If the up key is pressed the helicopter moves in the y direction -10 pixels. Else if the down key is pressed the helicopter moves in the y direction +10 pixels.

- **Variables and assignment statements**
  - **List of Variables and how they are Used:**
    - Score : changes as gems are picked up by the helicopter, where each color gem has a different point value.
    - greengemlist, yellowgemlist, redgemlist, bluegemlist: for the gem arrays
    - greengemTimer, yellowgemTimer, redgemTimer, bluegemTimer : for the timer loops in which the arrays of gems are produced.
    - greengem, yellowgem, redgem, bluegem : indicates which gem symbol from the library is taken for that array
    - birdlist : for the bird array
    - birdTimer : for the timer loop in which the array of birds is produced.
    - bird : indicates which bird symbol from the library is taken for that array
    - helicopter : the helicopter object the user moves with arrow keys

- **Many function definitions that require you to write a program call to use them. In other words, not listener functions that have an Event parameter.**

    - **Function Header**: function initializeGame( )
        - **Action**: sets up the variables at the start of a game when it is replayed. It also calls for the barloop2 frame loop to stop. At this frame there is also a call for the score to be visible through scoreBox.text = "Your Score: " + Score.
    - **Function Header**: function gameOver(score: uint)
        - **Action**: stops the greengemTimer, yellowgemTimer, redgemTimer, bluegemTimer, birdTimer, the stage EventListener(Event.ENTER_FRAME, moveObjects), the stage EventListener(KeyboardEvent.KEY_DOWN, moveObject), and it removes the Child(helicopter) and gem and bird objects. Then it goes to the gameOver frame, label3, and displays the score.

- **At least one function that returns a result value**

    - **Function Header**: function checkForHit(score: uint): uint
    - **Action / Calculation**: detects if the helicopter comes into contact with any of the gem array or bird array objects. Within this function there are for loops. A return statement returns the updated value for the score.

- **An array**
    - **Array Name(s):** greengemlist, yellowgemlist, redgemlist, bluegemlist
        - **Purpose:** the gem arrays, to organize the gems that will be moved across the screen for the helicopter to hit.
    - **Array Name(s):** birdlist
        - **Purpose:** the bird array, to organize the birds that will be moved across the screen for the helicopter to hit.

- **Timeline control via gotoAndStop, gotoAndPlay, etc.**

    - **List of buttons that use gotoAndStop, gotoAndPlay, etc.:** play_btn, instruction_btn, credit_btn, main_btn, main2_btn, replay_btn
    - **Other Uses of gotoAndStop, gotoAndPlay, etc.:** There is a gotoAndPlay when a bird is hit in that game. This signifies that the game is over. So this takes the player to the gameoverpage where the player's score is displayed.

Additional Features (beyond the basic requirements)

- **For Loops**
  - For Loops are used inside the moveObjects function to go through the gem and bird arrays to move the objects

- **Timers**
  - There are four TimerEvents, one for each colour of gems, which start the production of different gems in different locations along the y-axis (with Math.random) on the right side of the stage.
  - There is a TimerEvent for the birds, which starts the production of different birds in different locations along the y-axis (with Math.random) on the right side of the stage.

- **Dynamic Arrays of Objects**
  - Each of the gems and birds is created as a dynamic object by the timers and added to a dynamic array of objects

- **Frame Loops**
  - There is a frame loop that uses Event.ENTER_FRAME to control the barloop, so that the radar bar goes around in a circle like a real radar screen.
  - There is a frame loop whose handler function calls the moveObjects function, which contains for loops to move the gem arrays and bird array along the x-axis of the screen by -5 pixels.

- **Hit Test**
  - The hitTestObject method is used to check whether the helicopter has hit a bird or gem

- **Sound**
  - A radar sound is used on the homepage, instruction page, credit page, and game over page where the radar bar goes around in a circle

- **ColorTransform Class**
  - A ColorTransform object is used to change the color of the helicopter when it hits something

Credits

Credit for Concept: The concept of this game came from a game I used to play called simply the helicopter game found at the website:

http://www.helicoptergame.net/. However, instead of being in a cave I decided to bring the helicopter outside in front of a city skyline. I also changed the obstacles and how points are gained. In the helicopter game the obstacles are blocks floating on the stage, but in my game the player must avoid the birds flying across the stage. Points in my game are gained by picking up the different colored gems floating across the screen, whereas in the helicopter game the score is calculated by the distance traveled before the helicopter crashes.

Credit for Image and Sound Files:

- The images of the skyline and bird were found on a public website
  - http://www.clker.com/clipart-city-skyline-1.html - for the skyline
  - http://www.clker.com/clipart-3303.html - for the bird
  -
- The radar sound was also taken from a youtube video of an actual radar screen.
  - http://www.youtube.com/watch?v=iSdHHx0bxpM&feature=related

- I created all of the other images

Credit for Code Ideas (not including examples from class):

All of the code is my own, adapted from what I learned in the course.