**helloworldhelloworldhelloworldhelloworldhelloworldhelloworldhelloworldhelloworldhelloworldhelloworldhelloworldhelloworldhelloworldhelloworldhelloworldhelloworldhelloworldhelloworldhelloworldhelloworldhelloworldhelloworldhelloworldhelloworldhelloworldhelloworldhelloworldhelloworldhelloworldhelloworldhelloworldhelloworldhelloworldhelloworldhelloworldhelloworldhelloworldhelloworldhelloworldhelloworldhelloworldhelloworldhelloworldhelloworldhelloworldhelloworldhelloworldhelloworldhelloworldhelloworldhelloworldopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmrtyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmrtyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmrtyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmrtyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnm**

|  |
| --- |
| Eric Chan  Mr. Sgouraditis  Mini Actionscript Project  Eric Chan |

Idea

My concept for this game will be based on a theme of a zombie apocalypse. The player will try to survive throughout the entire level and if successful, they will have won the game. This game will be a single-player shooter where the player can control the shooter via the keyboard from a bird’s eye view. The shooter will be able to navigate in all directions on the map and the shooting mechanism will be controlled via the mouse. The player can aim and shoot in any direction, and will have to press R to reload once the ammunition reaches zero. The will be a total of four main displays on the screen; one on the bottom left corner to display the player`s health, one on the bottom right corner to display the amount of ammunition left, another one on the top left corner to display score, and one on the top center to display the time left.

The player/shooter will spawn in a large, rectangular - like “battlefield”, and a house will spawn in the center of the map for the player to refill on health and ammunition. The house is enterable and if the player walks inside, they simply walk over the health or ammunition kit to refill. However, once the player refills on either health or ammunition, both kits deactivate for 4 seconds until the player can use them again. The zombies will spawn on the outermost edges of the map. Each zombie will be eliminated with two shots and in return, reward the player with 1 point. The zombies will follow the direction of the player and will walk progressively faster and spawn much more rapidly as the timer approaches zero. When the player is attacked by a zombie, they will lose 10 points of health from a total of 100 and 1 point from their total score.

When the player’s health reaches zero, they lose the game and a total score will be displayed. If the player is able to survive until the timer reaches zero, they will have won the game and a score will be displayed as well. At the end of either ending, there will be an option for the player to replay the game or exit out of it.

Design Document – Frame 1 Main Timeline

=====Frame 1 of Main Timeline====================================

Attach the intro clip

===== end Frame 1 of Main Timeline================================

===== Intro Clip *design*===========================================

Attach the button clip

Link this clip for exporting

===== end Intro Clip *design* =============================================

===== Button Clip *design + Algorithm* =========================================

*Design:*

Link this clip for exporting

*Design & Algorithm:*

Frame 1

Draw the normal button and display

On roll over, go to play frame 2

On roll out, go to play frame 1

Frame 2

Draw the hover button and display

On press, go to play frame 3

On release, go to play frame 2 on Main Timeline

Play a sound

Frame 3

Draw the clicked button and display

===== end Button Clip design =========================================

Design Document – Frame 2 Main Timeline

=====Frame 2 of Main Timeline (MAIN *algorithm*) ====================

*Algorithm:*

Attach house clip

Attach shooter clip

Attach land clip

Attach black filter clip

Create object named player

Add a health property to player object, set to 100

Add a death animation property to player object, set to false

Add a score property to player object, set to 0

Create health display text box, display at the bottom left of the screen

Create bullet display text box, display at the bottom right of the screen

Create magazine display text box, display at the bottom right of the screen, under bullet display

Create score display text box, display at the top right of the screen

Create time display text box, display at the top center of the screen

Attach and play a zombie moaning sound

Attach and play music

Create a variable to set zombie health to 2

Create two-dimensional zombie array (First row handles movie clips, second row handles health)

Execute zombie spawner function

Use setInterval to control the spawning times of zombies

*Design of zombie generator/spawner*:

Create a zombie spawner function:

Create a variable to receive getTimer()’s data

If time is less than 2 ¼ min

Create a variable to a contain a temporary attached zombie movie clip

Create a variable to a contain a random number from 1 to 4

Execute direction spawner function and send both variables to the function

If time is more than 2 ¼ min but less than 2 ¾ min

For loop twice

Create a variable to a contain a temporary attached zombie movie clip

Create a variable to a contain a random number from 1 to 4

Execute direction spawner function and send both variables to the function

If time is more than 2 ¾ min

For loop 3 times

Create a variable to a contain a temporary attached zombie movie clip

Create a variable to a contain a random number from 1 to 4

Execute direction spawner function and send both variables to the function

End of zombie spawner function

Create a direction spawner function (it will receive a random number and a temporary movie clip)

Use a switch construct to test for random number

If random number is 1

Assign temporary zombie clip to spawn at the top of the screen

Push the temporary zombie movie clip to the first row of the zombie array

Push the zombie health variable to the second row of the zombie array

If random number is 2

Assign temporary zombie clip to spawn at the bottom of the screen

Push the temporary zombie movie clip to the first row of the zombie array

Push the zombie health variable to the second row of the zombie array

If random number is 3

Assign temporary zombie clip to spawn to the left of the screen

Push the temporary zombie movie clip to the first row of the zombie array

Push the zombie health variable to the second row of the zombie array

If random number is 4

Assign temporary zombie clip to spawn to the right of the screen

Push the temporary zombie movie clip to the first row of the zombie array

Push the zombie health variable to the second row of the zombie array

Default

Display text “no zombie spawned”

End of switch construct

End of direction spawner function

===== end Frame 2 of Main Timeline (MAIN *algorithm*) ================

=====House Clip *design + algorithm*===============================

*Design:*

Draw house

Link it for exporting

*Algorithm:*

Attach ammo kit clip

Attach health kit clip

Attach house top clip

Create a refill object

Add a ready property to refill object, set to true

On enter frame

If shooter is inside the boundaries of the house

Decrease the alpha of the house top clip

If shooter is outside the boundaries of the house

Set the alpha of the house top clip to 100

End of enter frame

=====end House Clip *design + algorithm*===========================

=====House Top Clip *design*=====================================

*Design:*

Draw the top of the house

Link it for exporting

=====end House Top Clip *design*=================================

=====Ammo kit Clip *design + algorithm*===========================

*Design:*

Draw the ammo kit

Link it for exporting

*Algorithm:*

Declare a variable to contain the instantiation of the color object

Create two objects, one containing properties for tinting the colours of the ammo kit, the other to revert it back to its normal colours

On Enter frame

If the shooter hits this clip &&ready property of refill object is true&&bullet array, second row < 3

Set bullet array, second row with a value of 3

Set the remaining bullets property of pistol object to true

Execute countdown function

setInterval for countdown function to 4 seconds, and pass a value of true

Update the magazine display

Update the bullet display

Change the colour using the tinting object

Attach pickup sound

End of Enter frame

Create a countdown function (It will receive a Boolean value)

Set the ready property of the refill object to the Boolean value of the variable received

If ready property of the refill object is true

Clear the countdown setInterval

Change the colour using the normal colours object

End of countdown function

=====end Ammo kit Clip *design + algorithm*=======================

=====Health kit Clip *design + algorithm*===========================

*Design:*

Draw health kit

Link it for exporting

*Algorithm:*

Declare a variable to contain the instantiation of the color object

Create two objects, one containing properties for tinting the colours of the health kit, the other to revert it back to its normal colours

On Enter frame

If the shooter hits this clip &&ready property of refill object is true&&health property of player object is <100

Add 20 points to health property of player object

If health property > 100

Set health property to 100

Change the colour using the tinting object

Attach pickup sound

Update health display

Execute countdown function

setInterval for countdown function to 4 seconds, and pass a value of true

End of Enter frame

Create a countdown function (It will receive a Boolean value)

Set the ready property of the refill object to the Boolean value of the variable received

If ready property of the refill object is true

Clear the countdown setInterval

Change the colour using the normal colours object

End of countdown function

=====end Health kit Clip *design + algorithm*=======================

===== gun clip *design* =========================================

*Design:*

Draw the gun

Link it for exporting

===== end gun clip *design* =====================================

===== bullet clip *design* ========================================

*Design:*

Draw the bullet clip

Link it for exporting

===== end bullet clip *design* ====================================

===== legs *design + algorithm* ===================================

*Design:*

Link it for exporting

*Algorithm:*

Create speed variable

On Enter frame

If ‘w’ key is pressed

Decrement shooter clip’s y value by speed

If ‘s’ key is pressed

Increment shooter clip’s y value by speed

If ‘a’ key is pressed

Decrement shooter clip’s x value by speed

If ‘d’ key is pressed

Increment shooter clip’s x value by speed

End of enter frame

===== end legs *design + algorithm* ===============================

===== head and arms *design + algorithm* ==========================

*Design:*

Draw the head and arms of shooter

Link it for exporting

*Algorithm:*

Attach gun movie clip

===== end head and arms *design + algorithm* =======================

=====Shooter Clip *algorithm*====================================

Frame 1

*Algorithm:*

Attach legs clip

Attach head and arms clip

Create two-dimensional bullet array (First row handles bullet movie clips, second row handles number of bullets, third row handles number of pistol magazines)

Create bullets variable, set value to 9

Create magazine variable, set value to 3

Push row 1 of bullet array with bullets variable

Push row 2 of bullet array with magazine variable

Update bullet display

Update magazine display

Create mouse clicked variable, set to false

Create speed variable, set to 40

Create reload speed variable, set to 250

Create a pistol object

Create a remaining bullets property, set to true

Create a reload complete property, set to true

Create a time elapsed string variable

Create a clock time function (receive a number value in milliseconds)

Create seconds variable

Create minutes variable

Divide the received number value by 1000, round it off, and assign it to the seconds variable

Divide seconds variable by 60, round it off, and assign it to the minutes variable

Subtract seconds variable by minutes variable x 60

Return a string value containing the minutes and seconds variable

End of clock time function

Root on Enter Frame

Head and arms coordinates == legs coordinates

Create radians variable

Create degrees variable

Calculate in radians, the angle of the mouse while moving, assign it to the radians variable

Calculate the angle of the mouse in degrees, using the radians variable, assign to the degrees variable

Assign degrees variable to the head and arms variable’s rotation

Execute fire weapon function

Execute update bullet function

Input getTimer into the clock time function

Assign the returned value of the clock time function to the time elapsed variable

If time elapsed != 3 min

Update time display with the time elapsed variable

Else if time elapsed == 3 min

Set time display == 3 min

Go to main timeline and stop at frame 4

End of Root on Enter Frame

Create fire weapon function

If reload complete property of pistol object&& mouse clicked variable && remaining bullets == true

Execute create bullet function

End of fire weapon function

Create a create bullet function

If row 1 on bullet array >=1

Create a temporary bullet clip to contain attached bullet clip

Assign temporary bullet clip coordinates == shooter coordinates

Assign temporary bullet clip rotation == degrees variable

Calculate vertical speed; assign value to a newly created vertical property of the temporary bullet clip

Calculate horizontal speed; assign value to a newly created horizontal property of the temporary bullet clip

Use setInterval to execute destroy bullet function and to send the temporary bullet clip to it

Assign the value of setInterval to a newly created life timer property of the temporary bullet clip

Push temporary bullet clip to bullet array, row 1

Subtract a value of 1 from bullet array, row 2

Execute start reloading function

Update bullet display

Attach sound

Else if bullet array == 0

Set remaining bullet property to false

End of create bullet function

Create update bullet function

For loop the length of bullet array, row 1

Create a temporary bullet clip

Using the for loop counting variable to access each value of bullet array, row 1; assign each new bullet array, row 1 value to temporary bullet clip

X coordinate of temporary bullet clip += temporary bullet clip vertical property

Y coordinate of temporary bullet clip += temporary bullet clip horizontal property

Execute check enemy hit function, input temporary bullet clip

Execute check house hit function, input temporary bullet clip

End of update bullet function

Create check enemy hit function (receive a bullet clip)

For loop the length of zombie array, row 1

Create temporary zombie clip

Using the for loop counting variable to access each value of zombie array, row 1; assign each new zombie array, row 1 value to temporary zombie clip

If temporary zombie clip hitTest received bullet clip

Execute destroy zombie function, input temporary zombie clip

Execute destroy bullet hit function, input received bullet clip

Attach zombie hit sound

End of check enemy hit function

Create destroy zombie function (receive a zombie clip)

For loop the length of zombie array, row 1

Create temporary zombie clip

Using the for loop counting variable to access each value of zombie array, row 1; assign each new zombie array, row 1 value to temporary zombie clip

If temporary zombie clip == received zombie clip

(Using the for loop counting variable to access the specific zombie array, row 2)

If zombie array, row 2 == 2

Decrement a value of 1, from zombie array, row 2

Play frame 2 of temporary zombie clip

Else if zombie array, row 2 == 1

Play frame 6 of zombie array, row 1

Splice zombie array, row 1, once

Splice zombie array, row 2, once

Add 1 point to the score property of the player object

Update score display

End of destroy zombie function

Create destroy bullet hit function (receive a bullet clip)

For loop the length of bullet array, row 1

Create temporary bullet clip

Using the for loop counting variable to access each value of bullet array, row 1; assign each new bullet array, row 1 value to temporary bullet clip

If temporary bullet clip == received bullet clip

(Using the for loop counting variable to access the specific bullet array)

Splice bullet array, row 1, once

Remove temporary bullet clip

End of destroy bullet hit function

Create destroy bullet function (receive a bullet clip)

For loop the length of bullet array, row 1

Create temporary bullet clip

Using the for loop counting variable to access each value of bullet array, row 1; assign each new bullet array, row 1 value to temporary bullet clip

If temporary bullet clip == received bullet clip

(Using the for loop counting variable to access the specific bullet array)

Splice bullet array, row 1, once

clearInterval on life timer property of the temporary bullet clip

Remove temporary bullet clip

End of destroy bullet function

Create start reloading function

Assign a false value to reload complete property of the pistol object

Create a reload timer property on the pistol object

Assign setInterval (with the gun reloaded function and reload speed variable inputted) to reload timer property

End of start reloading function

Create gun reloaded function

clearInterval on the reload timer property of the pistol object

Assign a true value to reload complete property of pistol object

End of gun reloaded function

Create check house hit function (receive a bullet clip)

If received bullet clip hitTest house clip

Execute destroy bullet hit function, with received bullet clip inputted

End of check house hit function

Create new key object

On key down

If r key is pressed

If bullet array, row 2 < 9 && bullet array, row 2 >=0 && bullet array, row 3 > 0

Decrement a value of 1, from bullet array, row 3

Assign bullet array, row 2, with a value of 9

Assign a value of true, to remaining bullets property of the pistol object

Update magazine and bullet displays

Attach a reload sound

Create new mouse object

On mouse down

Set mouse clicked variable to true

If remaining bullets property of pistol object == false

Attach a dry fire sound

Else if health property of player object == false

Set mouse clicked variable to false

On mouse up

Set mouse clicked variable to false

Add mouse listener

Add key listener

Frame 2

Set dead animation property of player object to true

Attach dead shooter clip

Set dead shooter clip’s coordinates and rotation == head and arms clip’s coordinates and rotation

Set head and arms clip’s alpha to 0

Set legs clip’s alpha to 0

Play Frame 3 of the main timeline

=====end Shooter Clip *design + algorithm*===========================

=====land Clip *design* =================================================

*Design:*

Draw a background

Link it for exporting

=====end land Clip *design*==============================================

=====lamp Clip *design* =================================================

*Design:*

Draw a lamp

Manually attach light clip

Link it for exporting

=====end lamp Clip *design*==============================================

=====light Clip *design* =================================================

*Design:*

Animate a glowing light using tweens

=====end light Clip *design*==============================================

=====black filter Clip *design* ============================================

*Design:*

Draw a rectangular black filter

Manually attach lamp clips

Link it for exporting

=====end black filter Clip *design* ========================================

===== zombie clip *design + algorithm* =====================================

*Design:*

Draw the zombie clip

Link it for exporting

*Algorithm:*

Frame 1

Attach alive zombie clip

Create a distance x variable

Create distance y variable

Create radians variable

Create degrees variable

Create velocity x variable

Create velocity y variable

Create a speed variable, with a value of 2

Create two objects, one containing properties to change colour for a hurt shooter, another for the normal colours

Create a change normal colour function (Accepts a Boolean value)

If accepted Boolean value == false

Change colour using normal colour object

clearInterval on colour interval changer

End change normal function

On enter frame

Calculate horizontal distance from shooter to zombie; Assign value to distance x variable

Calculate vertical distance from shooter to zombie; Assign value to distance y variable

Calculate in radians, the angle pointing towards the shooter clip; Assign value to radians variable

Convert radians to degrees; Assign value to degrees variable

Assign this clip’s rotation to the degrees variable

Calculate horizontal velocity using the speed variable and radians variable; Assign value to velocity x variable

Calculate vertical velocity using the speed variable and radians variable; Assign value to velocity y variable

Assign this clip’s x value to the value of velocity x

Assign this clip’s y value to the value of velocity y

If this clip hitTest the shooter clip

If health property of player object is != 0

Attach shooter hit sound

Execute change normal function

Set colour changer interval to execute change normal function, while passing a false value to it

If health property of player object == 10 && dead animation property of player object == false

Go play Frame 2 of shooter clip

Else if health property of player object is >0

Execute change colour function, with hurt object inputted

Decrement 10 points to health property of player object

Update health display

Add distance x value to shooter clip’s x value

Add distance y value to shooter clip’s y value

If score property of player object is >0

Decrement a value of 1 to score property of player object

Update score display

Else if dead animation of player object == true

Decrement this clip’s x value by the value of velocity x

Decrement this clip’s y value by the value of velocity y

If this clip hitTest house clip

Decrement this clip’s x value by the value of velocity x

Decrement this clip’s y value by the value of velocity y

End of enter frame

Frame 2 – 5

End of Frame 5

Go to play Frame 1 of this clip

Frame 6

Attach dead zombie clip

Assign dead zombie clip’s x value to the zombie clip’s x value

Assign dead zombie clip’s y value to the zombie clip’s y value

Assign dead zombie clip’s rotation to zombie clip’s rotation

Remove zombie clip

===== end zombie clip *design + algorithm* ==================================

===== dead zombie clip *design + algorithm* =================================

*Design:*

Draw the animation of dying zombie

Link it for exporting

*Algorithm:*

(On the last frame)

Create a vanish speed variable

On enter frame

Decrement dead zombie clip’s alpha by the vanish speed variable

If dead zombie clip’s alpha == 0

Remove dead zombie clip

End enter frame

===== end dead zombie clip *design + algorithm* =============================

Design Document – Frame 3 Main Timeline

=====Frame 3 of Main Timeline *algorithm*===========================

*Algorithm:*

Remove health display text box

Remove bullet display text box

Remove magazine display text box

Remove score display text box

Remove time display text box

clearInterval on zombie spawner

Attach the ending clip

===== end Frame 3 of Main Timeline *algorithm*=======================

Design Document – Frame 4 Main Timeline

=====Frame 4 of Main Timeline *algorithm* ==========================

*Algorithm:*

Remove health display text box

Remove bullet display text box

Remove magazine display text box

Remove score display text box

Remove time display text box

Remove shooter movie clip

clearInterval on zombie spawner

Attach the ending clip

===== end Frame 4 of Main Timeline *algorithm* ======================