4/18/2016

Simon Light

UTC Reading

Assignment 2

Unit 6

Contents

[P6 – story board of UI 2](#_Toc448744186)

[Title Page 2](#_Toc448744187)

[Game scenarios 3](#_Toc448744188)

[Start-up 3](#_Toc448744189)

[Data Dictionary 4](#_Toc448744190)

[D2 5](#_Toc448744191)

[Pseudo Code 5](#_Toc448744192)

# P6 – story board of UI

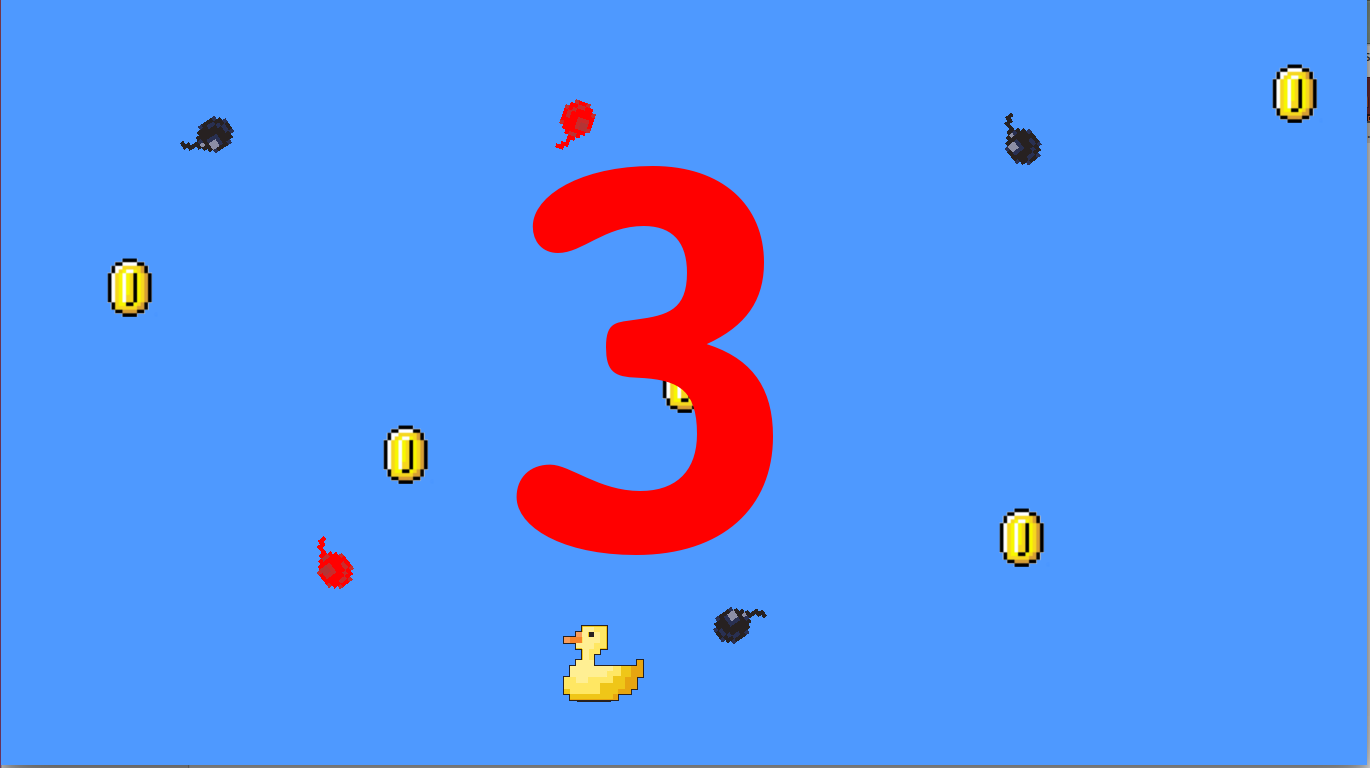
In this section I will be detailing the goals that I want my game to reach. This will include the title page, menu and the games different scenarios.

## Title Page

Above is my title page. I am also going to use it as the menu page for when players pause the game. The background shall be running a simulation of the game. This could either be in the form of a video of someone playing the game or it could be created live giving a more diverse result. The Title and menu pane will not interfere with the game mechanics (the bombs will not bounce off of the Title.

# Game scenarios

## Start-up

When the game begins all objects will be randomly generated. As shown above a countdown from 3 will begin to give the player some time to ready themselves for the beginning of the game. The Duck character will be generated in the centre of the screen but will instantly be placed in-line with the mouse x-coordinate. This Character can then be moved by the mouse. All other sprites will not be able to move until the countdown has finished, then the game will commence.

# Data Dictionary

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Purpose | Data Type  & Ranges | Reasoning |
| frame\_width  frame\_height | Store the width/height of the users screen. | Integer  -2,147,483,648 to 2,147,483,647 | Is a small enough data type that I am not worried about running the application on a computer with low amounts of RAM, big enough to store the largest/multi screen displays. |
| colour\_depth | Store the amount of colour bits required. | Integer  -2,147,483,648 to 2,147,483,647 | Small enough to keep RAM usage low, has the best compatibility with library programs without changing to a different data type. |
| frame\_resizeable  frame\_fullscreen | Stores whether the user wants the screen to be resizeable/full screen | Boolean  True/False | The answer is only going to be true/false and boolean is the best and smallest way to store this. |
| use\_opengl  use\_hardware | Stores whether the user wants the game to use their graphics card/openGL | Boolean  True/False | The answer is only going to be true/false and boolean is the best and smallest way to store this. |
| Background  sprites | Stores the sprite sheet and background going to be used. | Surface  Is dependent on image size | Very good at adjusting the data to be minimal as well as adjustable (i.e. cutting sprites from the sprite sheet). It is also very good at interfacing with SDLdotNET (the graphics library I am using) |
| Video | Store the output to the user, comes from the combination of background, sprites and the physics engine. | Surface  Is dependent on image size | Very good at adjusting the data to be minimal as well as adjustable (i.e. cutting sprites from the sprite sheet). It is also very good at interfacing with SDLdotNET (the graphics library I am using) |

## D2

### Pseudo Code

class program:

rnd = Random

ss (SpriteSheet)

duck (Duck)

Bomb[10] bombs

Coin[10] coins

scoreWord(scoreWord)

tens(TensScore)

units(UnitsScore)

gameOver(GameOver)

void onInit:

new Duck

new ScoreCard

new TensScore

new UnitsScore

new GameOver

for i is less than amount of bombs:

new bomb[i]

for i is less than amount of coins:

new coin [i]

end void

bool checkBombsIntercept takes i(integer):

if any intercections:

return true

else

return false

end bool

void newScore takes score(integer):

change score counter to score(integer)

end void

bombStepper(integer) = 0

coinStepper(integer) = 0

delayer(integer) = 0

score(integer) = 0

secDelayer(integer) = 200

if secDelayer is bigger than 150 && secDelayer is less than 200:

quit aplication

end if

for i(integer is less than amount of bombs:

if checkBombsIntercept(i)

show gameover

secDelayer = 0;

end if

end for loop

for i(integer)is less than amount of coins:

if checkCoinsIntercept(i):

if delayer is greater than 10:

score++

newScore(score)

coins[i] set Y = 0

coins[i] set X = rnd.Next(FRAME\_WIDTH)

coins[i] set DY = rnd.Next(1, 4)

delayer = 0

end if

end if

end for loop

if bombStepper is greater than 60:

for i(integer)is less than amount of bombs:

if bombs[i] get Sprite is equal to 2:

bombs[i] set Sprite = 1

end if

else

bombs[i] set Sprite = 2

end else

bombStepper = 0

end for loop

end if

if coinStepper is greater than 2:

for i(integer) is less than amount of coins:

if coins[i] get Sprite is equal to 10:

coins[i] set Sprite = 4

end if

else:

coins[i] set Sprite = (coins[i] get Sprite +1)

end else

coinStepper = 0

end for loop

end if

if duck get Y is less than 0:

duck setY = FRAME\_HEIGHT

end if

else if duck get Y is greater than FRAME\_HEIGHT:

duck set Y = 0

end else if

for i(integer) is less than amount of bombs:

bombs[i] move

if bombs[i] get Y is greater than FRAME\_HEIGHT:

bombs[i] set Y = 0

bombs[i] set X = rnd.Next(FRAME\_WIDTH)

bombs[i] set DY = rnd.Next(1,4)

end if

end for

for i(integer) is less than amount of coins:

coins[i] move

if coins[i] get Y is greater than FRAME\_HEIGHT:

coins[i] set Y = 0

coins[i] set X = rnd.Next(FRAME\_WIDTH)

coins[i] set DY = rnd.Next(1, 4)

end if

end for

drawBackground();

drawSprite(duck get Sprite,duck get X,duck get Y,duck get Direction)

drawSprite(scoreWord get Sprite, scoreWord get X, scoreWord get Y,0)

drawSprite(tens get Sprite, tens get X, tens get Y ,0)

drawSprite(units get Sprite, units get X, units get Y,0)

if sow game over is true:

drawSprite(gameOver get Sprite, gameOver get X, gameOver get Y, 0)

end if

for i(integer) is less than amount of bombs:

drawSprite(bombs[i] get Sprite, bombs[i] get X, bombs[i] get Y, bombs[i] get Direction)

end for loop

for i(integer) is less than amount of coins

drawSprite(coins[i] get Sprite, coins[i] get X , coins[i] get Y, coins[i] get Direction)

end for loop

update image

end class

## Real code

(Behind this page)

### 

### 