Attachment 1: Assessment Task project documentation

Contents

[1.“program name” Design Specifications 1](#_Toc134534618)

[2.“program name” Log book 1](#_Toc134534619)

[3.‘program name” Gantt chart 4](#_Toc134534620)

[4. “program name” Context Diagram 4](#_Toc134534621)

[5.“program name” Structure Chart 5](#_Toc134534622)

[6.“program name” Storyboard 5](#_Toc134534623)

[7.“program name” Test Report of final game: 6](#_Toc134534624)

[8.“program name” Evaluation : 6](#_Toc134534625)

[9.“program name” Maintenance: 6](#_Toc134534626)

[10.‘program code’ 6](#_Toc134534627)

# 1.“Get Away” Design Specifications

|  |  |
| --- | --- |
| Developers’ perspective | Users’ perspective |
| Documentation  Approach towards developing the software  Data types – integers – display\_width and display\_height  Data structures  Variables | Appropriate messages  Interface design – what the screen looks like  Appropriate icons  Relevant data formats for display  Social and ethical issues of the program  Gameplay and functionality |

# 2.“Get Away” Log book

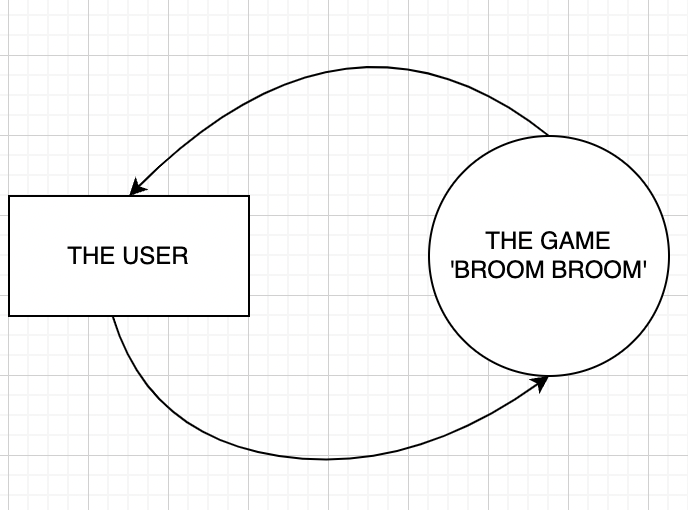
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Date /Time | Description of progress | Tasks achieved | Issues- stumbling blocks | references |
| 3/6/23  Time – 30 minutes | Planning and designing how the game will look and function. Gathering images. | Got a plan for the coding stage. |  | <https://www.google.com/search?q=top+view+pixel+art+car+png&source=lnms&tbm=isch&sa=X&ved=2ahUKEwjhtpmd6q__AhV9wjgGHeNmBsQQ_AUoAXoECAEQAw&biw=1440&bih=821&dpr=2> |
| 4/6/23  Time – 1 hour | Work on documentation, added – Gantt chart, context diagram, structure chart and story board. | Most of documentation is completed | Forgot the meaning of symbols for the different diagram and charts. Referred to coarse specifications. | <https://educationstandards.nsw.edu.au/wps/wcm/connect/44325629-51c6-4330-8bf8-662d5cfbe5fb/software-design-development-course-specs.pdf?MOD=AJPERES&CVID=> |
| 5/6/23  Time – 2 hours | Changing how the game looked using new images such as, the background, user’s car and rectangles | The game looks more like a game. | Got stuck when images were really small or big, found the solution online. | <https://stackoverflow.com/questions/43046376/how-to-change-an-image-size-in-pygame> |
| 7/6/23  Time – 1 hour | Reworking the main home screen with new and different button.  Created a separate program from the main to be ran from. | Reworked main screen  Separate program to start on. | Did not know how to call from a separate file, using the link provided found the solution | <https://www.geeksforgeeks.org/python-call-function-from-another-file/> |
| 8/6/23  Time – 1 hour | Created the help window and a button on the main screen for the help.  When paused instead of quitting you have the option to go back to the main screen. | Help menu created.  Pause function improved. | No issues |  |
| 9/6/23  Time – 1 hour | Improving the death screen with a background image and changing quit to return the user back to the menu.  Added stage counter to see what wanted Level going up to max of 5. | Reworked death screen.  Added wanted level counter on the top left. |  |  |
| 10/6/23  Time – 1 hour | Attempt on fixing the collision boundaries of the enemy car.  Attempt on creating animation, example tilting the car left when turn left. | Nothing achieved.  Code Finished. | Was unable to figure out how to created and fix these features. |  |
| 11/6/23  Time – 1 hour | Finalising the documentation, such as the test reporting with user feedback, along with other documentation. | Finished documentation. | Couldn’t update the whole table of contents as it would add all the images to it, so title remain old, but page numbers are updated. | Course Specifications:  <https://educationstandards.nsw.edu.au/wps/wcm/connect/44325629-51c6-4330-8bf8-662d5cfbe5fb/software-design-development-course-specs.pdf?MOD=AJPERES&CVID=>  Text book:  <https://camdenhaven.instructure.com/courses/1897/pages/links-and-resources> |

# 3.”Get Away” Gantt chart



# 

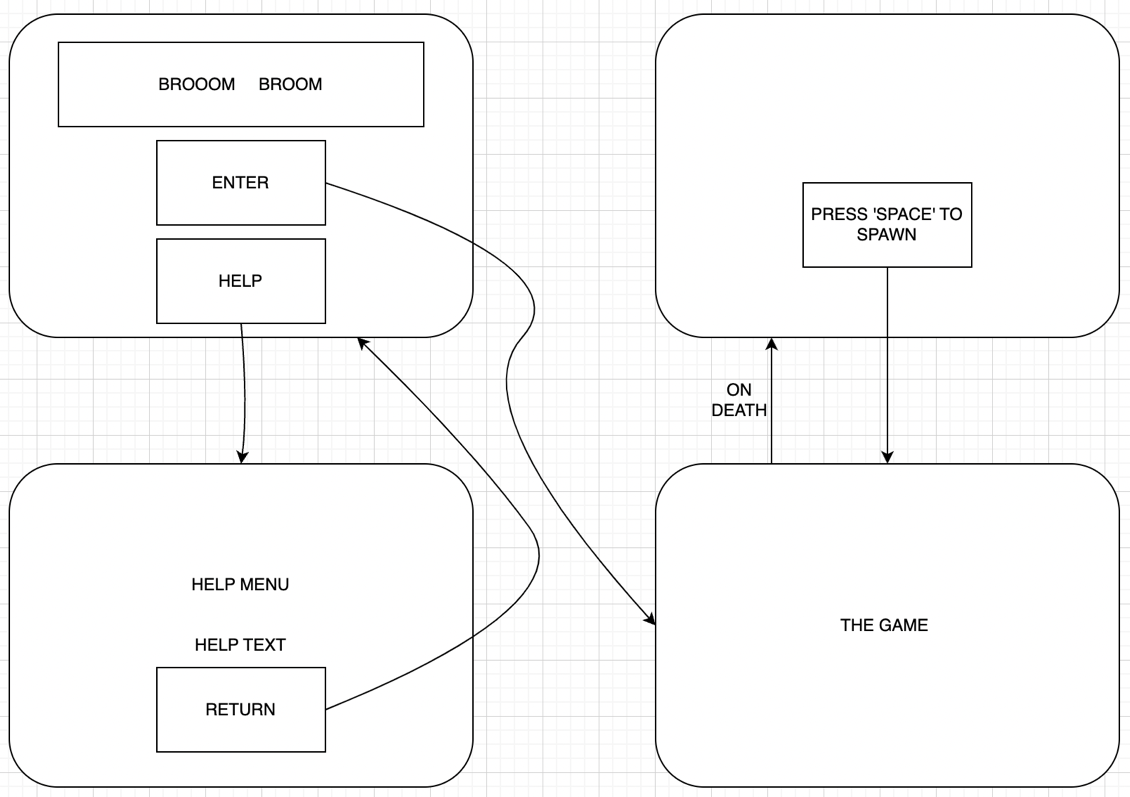
# 4. “Get Away” Context Diagram



# 5.“Get Away” Structure Chart

# 

# 6.“Get Away” Storyboard



7.“Get Away” Test Report of final game:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **User level of IT experience** | **Computer specs** | **Users feedback** | **Developer’s observations of user** | **Performance/ Errors** |
| **high** | **MacBook Air** | **Very good play well, only glitched with the boundaries once in the many plays.** | **Could navigate and use the program on their own and with ease.** | **Program continued to run but the boundaries of the enemy car are incorrect.** |
| **Moderate** | **MacBook Air** | **Found the game very catchy, great little game** | **Went to testing all the button, finding the game fun and understandable.** | **No errors found** |
| **low** | **MacBook Air** | **Very easy to understand game play. Instructions were easy to understand. Game played well and ran smoothly** | **When opening the game when straight to the help and then knew what to do finding it easy to understand.** | **No errors found.** |

8.“Get Away” Evaluation:

**The final result to the software solution is very effective as the game has gone from a plan looking game to something that looks like a real game, comparing it to the marking guidelines specifications the new program has ticked most nearly all of what is being ask of it.**

9.“Get Away” Maintenance:

**The current state of the program is stable and works well enough. As for the future of the program I would fix the boundaries of the cars / hitboxes, as a way that the user’s car can shoot out the cop cars to get them out of the way especially when the cars get faster and create a turning animation – I made a rotated version of the car but could get it to rotate when moving left or right.**

# 10.”Get Away” ‘program code’

AT3 – mainScreen:

# Jake Johnson, 10/6/23, separate file to open the main game

#Imports

import pygame

from AT3V4 import \*

#Used to initialize all imported pygame modules

pygame.init()

#make window size

display\_width = 800

display\_height = 600

#Colours

black = (0,0,0)

white = (255,255,255)

red = (255,0,0)

green = (0,200,0)

dark\_green = (1, 50, 32)

bright\_red = (255,0,0)

bright\_green = (0,255,0)

#Car size

car\_width = 75

#Sets game window and name

gameDisplay = pygame.display.set\_mode((display\_width,display\_height))

pygame.display.set\_caption('BROOM BROOM')

clock = pygame.time.Clock()

def text\_objects(text, font):

textSurface = font.render(text, True, black)

return textSurface, textSurface.get\_rect()

def button(msg, x, y, w, h, ic, ac, action=None):

mouse = pygame.mouse.get\_pos()

click = pygame.mouse.get\_pressed()

if x + w > mouse[0] > x and y + h > mouse[1] > y:

pygame.draw.rect(gameDisplay, ac, (x, y, w, h))

if click[0] == 1 and action != None:

if action == "play":

game\_loop()

elif action == "quit":

pygame.quit()

quit()

elif action == "continue":

unpause()

elif action == "Menu":

game\_intro()

elif action == "help":

game\_help()

else:

pygame.draw.rect(gameDisplay, ic, (x, y, w, h))

smallText = pygame.font.Font("freesansbold.ttf", 20)

textSurf, textRect = text\_objects(msg, smallText)

textRect.center = ((x + (w / 2)), (y + (h / 2)))

gameDisplay.blit(textSurf, textRect)

def game\_intro():

intro = True

while intro:

for event in pygame.event.get():

if event.type == pygame.QUIT:

pygame.quit()

quit()

# Background colour

gameDisplay.fill(dark\_green)

# Car

gameDisplay.blit(car\_size, car\_rect)

largeText = pygame.font.Font('freesansbold.ttf', 100)

TextSurf, TextRect = text\_objects("GET AWAY", largeText)

TextRect.center = ((display\_width / 2), (50))

gameDisplay.blit(TextSurf, TextRect)

button("GO!", 350, 300, 100, 50, green, bright\_green, "play")

button("Quit", 350, 500, 100, 50, red, bright\_red, "quit")

button("Help", 350, 400, 100, 50, blue, bright\_blue, "help")

pygame.display.update()

clock.tick(15)

game\_intro()

AT3V4:

# Jake Johnson, 10/6/23, to create a car game

# Imports

import pygame

import random

# Used to initialize all imported pygame modules

pygame.init()

# Make window size

display\_width = 800

display\_height = 600

# Colours

black = (0, 0, 0)

white = (255, 255, 255)

red = (153, 0, 0)

green = (0, 200, 0)

blue = (0, 0, 255)

bright\_blue = (0, 150, 255)

dark\_green = (1, 50, 32)

bright\_red = (255, 0, 0)

bright\_green = (0, 255, 0)

block\_colour = (53, 115, 255)

# Car size

car\_width = 75

# Sets game window and name

gameDisplay = pygame.display.set\_mode((display\_width, display\_height))

pygame.display.set\_caption('GET AWAY')

clock = pygame.time.Clock()

# Images

carImg = pygame.image.load('Images/car.png').convert\_alpha()

car\_size = pygame.transform.scale(carImg, (100, 150)).convert\_alpha()

car\_rect = car\_size.get\_rect(center = (400,200))

car\_normal = car\_size, car\_rect

# Was going to be used to turn the are left when moving left

car\_left = pygame.transform.rotate(car\_size, (20))

roadImg = pygame.image.load('Images/road.png').convert\_alpha()

road\_Big = pygame.transform.scale(roadImg, (800, 600)).convert\_alpha()

enemy = pygame.image.load('Images/enemy.png').convert\_alpha()

enemy\_size = pygame.transform.scale(enemy, (100,150)).convert\_alpha()

bars = pygame.image.load('Images/bars.png').convert\_alpha()

bars\_size = pygame.transform.scale(bars, (800,600))

bars\_rect = bars\_size.get\_rect(topleft = (0,0))

# Icon

pygame.display.set\_icon(carImg)

pause = False

# Will count the amount of times dodged

def things\_dodged(count):

font = pygame.font.SysFont(None, 50)

text = font.render("Dodged: " + str(count), True, black)

gameDisplay.blit(text, (0, 0))

return count

# Blocks/Enemy/Police

def things(thingx, thingy, thingw, thingh, colour):

gameDisplay.blit(enemy\_size, (thingx, thingy, thingw, thingh))

# places the car on screen

def car(x, y):

gameDisplay.blit(car\_size, (x, y))

def text\_objects(text, font):

textSurface = font.render(text, True, white)

return textSurface, textSurface.get\_rect()

# Quit function

def quitgame():

pygame.quit()

quit()

# Used to unpause game

def unpause():

global pause

pause = False

# Used to pause game

def paused():

largeText = pygame.font.SysFont("comicsansms", 115)

TextSurf, TextRect = text\_objects("Paused", largeText)

TextRect.center = ((display\_width / 2), (display\_height / 2))

gameDisplay.blit(TextSurf, TextRect)

while pause:

for event in pygame.event.get():

if event.type == pygame.QUIT:

pygame.quit()

quit()

button("Continue", 150, 450, 100, 50, green, bright\_green, 'continue')

button("Menu", 550, 450, 100, 50, red, bright\_red, 'Menu')

pygame.display.update()

clock.tick(15)

# Displays Crashed message on screen

def crash(count):

gameDisplay.fill(dark\_green)

gameDisplay.blit(bars\_size, bars\_rect)

# Big text after death

largeText = pygame.font.SysFont("comicsansms", 110)

TextSurf, TextRect = text\_objects("You Got Caught", largeText)

TextRect.center = ((display\_width / 2), (display\_height / 2))

gameDisplay.blit(TextSurf, TextRect)

# Attempting to show score on death screen

largeText = pygame.font.SysFont("comicsansms", 50)

TextSurf, TextRect = text\_objects("Dodged: " + str(count), largeText)

TextRect.center = ((display\_width / 2), (400))

gameDisplay.blit(TextSurf, TextRect)

while True:

for event in pygame.event.get():

if event.type == pygame.QUIT:

pygame.quit()

quit()

# Buttons

button("Escape", 150, 450, 100, 50, green, bright\_green, 'play')

button("Menu", 550, 450, 100, 50, red, bright\_red, 'Menu')

pygame.display.update()

clock.tick(15)

def game\_help():

helps = True

while helps:

for event in pygame.event.get():

if event.type == pygame.QUIT:

pygame.quit()

quit()

# Background colour

gameDisplay.fill(dark\_green)

largeText = pygame.font.Font('freesansbold.ttf', 80)

TextSurf, TextRect = text\_objects("GET AWAY: HELP", largeText)

TextRect.center = ((display\_width / 2), (50))

gameDisplay.blit(TextSurf, TextRect)

largeText = pygame.font.Font('freesansbold.ttf', 40)

TextSurf, TextRect = text\_objects("CONTROLS", largeText)

TextRect.center = ((display\_width / 2), (120))

gameDisplay.blit(TextSurf, TextRect)

text = pygame.font.SysFont("comicsansms", 20)

TextSurf, TextRect = text\_objects("To move the car left or right you can either use:", text)

TextRect.center = ((display\_width / 2), (150))

gameDisplay.blit(TextSurf, TextRect)

TextSurf, TextRect = text\_objects("'a''d' or the left and right arrows keys", text)

TextRect.center = ((display\_width / 2), (170))

gameDisplay.blit(TextSurf, TextRect)

TextSurf, TextRect = text\_objects("Pause the game by pressing 'ESC', can be used to return to the menu", text)

TextRect.center = ((display\_width / 2), (200))

gameDisplay.blit(TextSurf, TextRect)

TextSurf, TextRect = text\_objects("AIM OF THE GAME", largeText)

TextRect.center = ((display\_width / 2), (260))

gameDisplay.blit(TextSurf, TextRect)

TextSurf, TextRect = text\_objects("Your on the run from the police", text)

TextRect.center = ((display\_width / 2), (290))

gameDisplay.blit(TextSurf, TextRect)

TextSurf, TextRect = text\_objects("Moving your car left and right to dodge the police driving towards you", text)

TextRect.center = ((display\_width / 2), (310))

gameDisplay.blit(TextSurf, TextRect)

TextSurf, TextRect = text\_objects("You will lose and get caught if you run off the road or get hit by the police", text)

TextRect.center = ((display\_width / 2), (330))

gameDisplay.blit(TextSurf, TextRect)

########

# REACHING DIFFERENT STAGES THE LONGER YOU LAST

button("Menu", 350, 450, 100, 50, red, bright\_red, 'Menu')

pygame.display.update()

clock.tick(15)

# Menu screen buttons and actions

def button(msg, x, y, w, h, ic, ac, action=None):

mouse = pygame.mouse.get\_pos()

click = pygame.mouse.get\_pressed()

if x + w > mouse[0] > x and y + h > mouse[1] > y:

pygame.draw.rect(gameDisplay, ac, (x, y, w, h))

if click[0] == 1 and action != None:

#Opens the different screens

if action == "play":

game\_loop()

elif action == "quit":

pygame.quit()

quit()

elif action == "continue":

unpause()

elif action == "Menu":

game\_intro()

elif action == "help":

game\_help()

else:

pygame.draw.rect(gameDisplay, ic, (x, y, w, h))

smallText = pygame.font.Font("freesansbold.ttf", 20)

textSurf, textRect = text\_objects(msg, smallText)

textRect.center = ((x + (w / 2)), (y + (h / 2)))

gameDisplay.blit(textSurf, textRect)

# Intro screen

def game\_intro():

intro = True

while intro:

for event in pygame.event.get():

if event.type == pygame.QUIT:

pygame.quit()

quit()

# Background colour

gameDisplay.fill(dark\_green)

# Car

gameDisplay.blit(car\_size, car\_rect)

# Title

largeText = pygame.font.Font('freesansbold.ttf', 100)

TextSurf, TextRect = text\_objects("GET AWAY", largeText)

TextRect.center = ((display\_width / 2), (50))

gameDisplay.blit(TextSurf, TextRect)

# Buttons

button("GO!", 350, 300, 100, 50, green, bright\_green, "play")

button("Quit", 350, 500, 100, 50, red, bright\_red, "quit")

button("Help", 350, 400, 100, 50, blue, bright\_blue, "help")

# Updates display

pygame.display.update()

clock.tick(15)

# Main game loop

def game\_loop():

global pause

# Cars start position

x = (display\_width \* 0.45) # 0.45

y = (display\_height \* 0.76) # 0.85

# Used to change the cars position

x\_change = 0

# Random starting position

thing\_startx = random.randrange(200, 550)

# Rectangle size and speed

thing\_starty = -600

thing\_speed = 4

thing\_width = 60

thing\_height = 60

score = 1

dodge = 0

gameExit = False

while not gameExit:

# closes game

for event in pygame.event.get():

if event.type == pygame.QUIT:

pygame.quit()

quit()

# arrow keys move car

if event.type == pygame.KEYDOWN:

if event.key == pygame.K\_LEFT:

x\_change = -4

elif event.key == pygame.K\_RIGHT:

x\_change = 4

elif event.key == pygame.K\_ESCAPE:

pause = True

paused()

# Stops the car once key is lifted

if event.type == pygame.KEYUP:

if event.key == pygame.K\_LEFT or event.key == pygame.K\_RIGHT:

x\_change = 0

# Using 'a' and 'd' to move the car left and right

if event.type == pygame.KEYDOWN:

if event.key == pygame.K\_a:

x\_change = -4

elif event.key == pygame.K\_d:

x\_change = 4

# Stops the car once key is lifted

if event.type == pygame.KEYUP:

if event.key == pygame.K\_a or event.key == pygame.K\_d:

x\_change = 0

# Cars postion

x += x\_change

# Background colour

gameDisplay.fill(dark\_green)

# Road / Where the car can go

gameDisplay.blit(road\_Big, (0, 0))

things(thing\_startx, thing\_starty, thing\_width, thing\_height, block\_colour)

thing\_starty += thing\_speed

# car position

car(x, y)

things\_dodged(dodge)

# Crashes if car runs off the road

if x > 630 - car\_width or x < 150:

crash(dodge)

# checks dodges and makes rectangles faster

if thing\_starty > display\_height:

thing\_starty = 0 - thing\_height

thing\_startx = random.randrange(200, 550)

dodge += 1

thing\_speed += 0.5

# Progressively enemy gets faster

if thing\_speed >= 10 and thing\_speed < 10:

thing\_speed += 1

if thing\_speed >= 15:

thing\_speed += 1.5

# Adds Score

if thing\_speed == 5:

score += 1

if thing\_speed == 10:

score += 1

if thing\_speed == 20:

score += 1

if thing\_speed == 30:

score += 1

# Displaying the stage number on the screen

font = pygame.font.SysFont(None, 50)

display\_score = font.render("Wanted: " + str(score), True, black)

gameDisplay.blit(display\_score, (630, 0))

# Detects collisions for crashing

if y < thing\_starty+thing\_height:

if x > thing\_startx and x < thing\_startx + thing\_width or x+car\_width > thing\_startx and x+car\_width < thing\_startx+thing\_width:

crash(dodge)

# Updating the display by 60 ticks a second

pygame.display.update()

clock.tick(60)

# Running the main functions

game\_intro()

game\_loop()

game\_help()

pygame.quit()

quit()