Five Move Murder Fest

The RPG Manager Application
Developed by Starshiplad

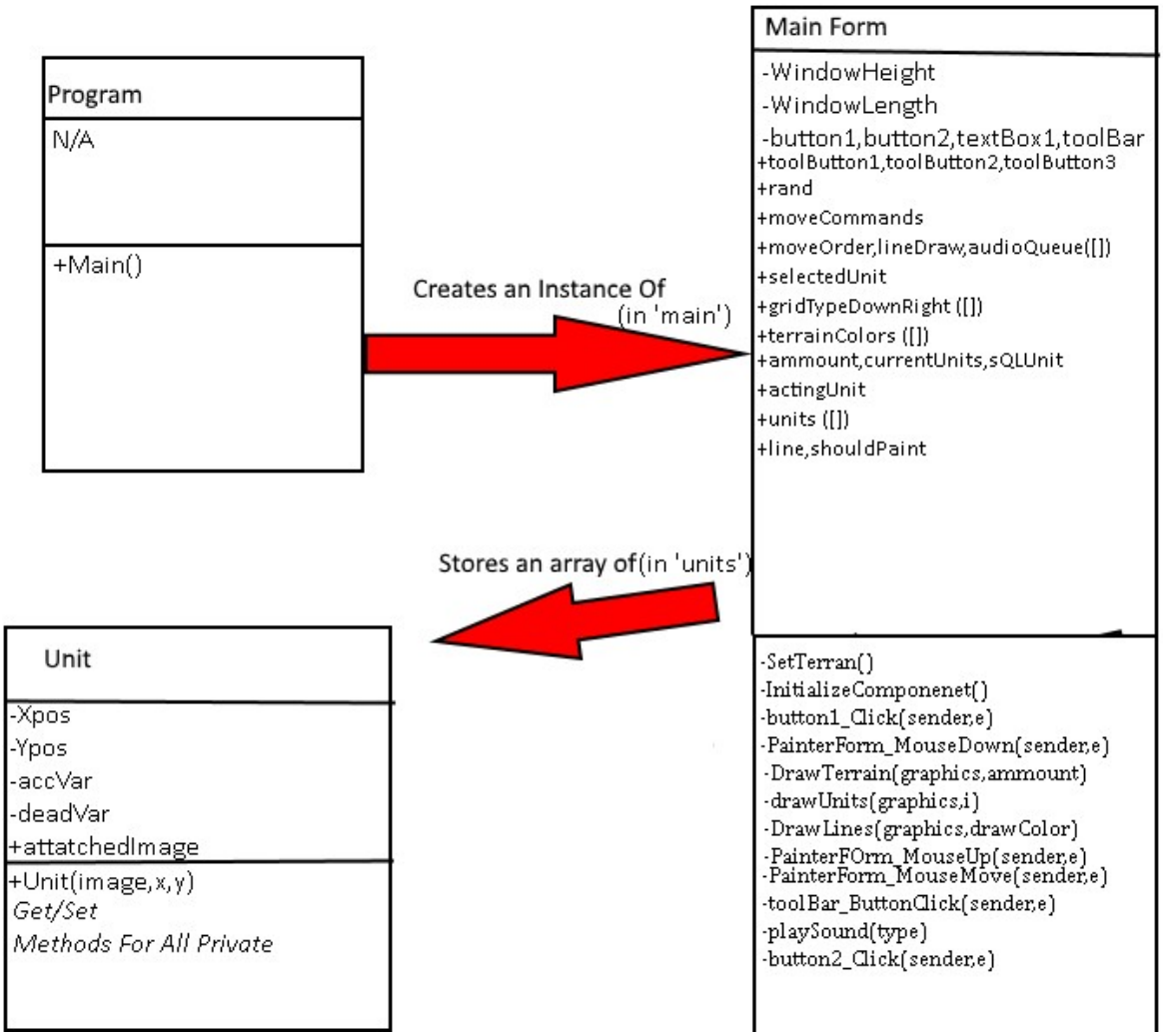
Update as of: 27/09/19

Contents:

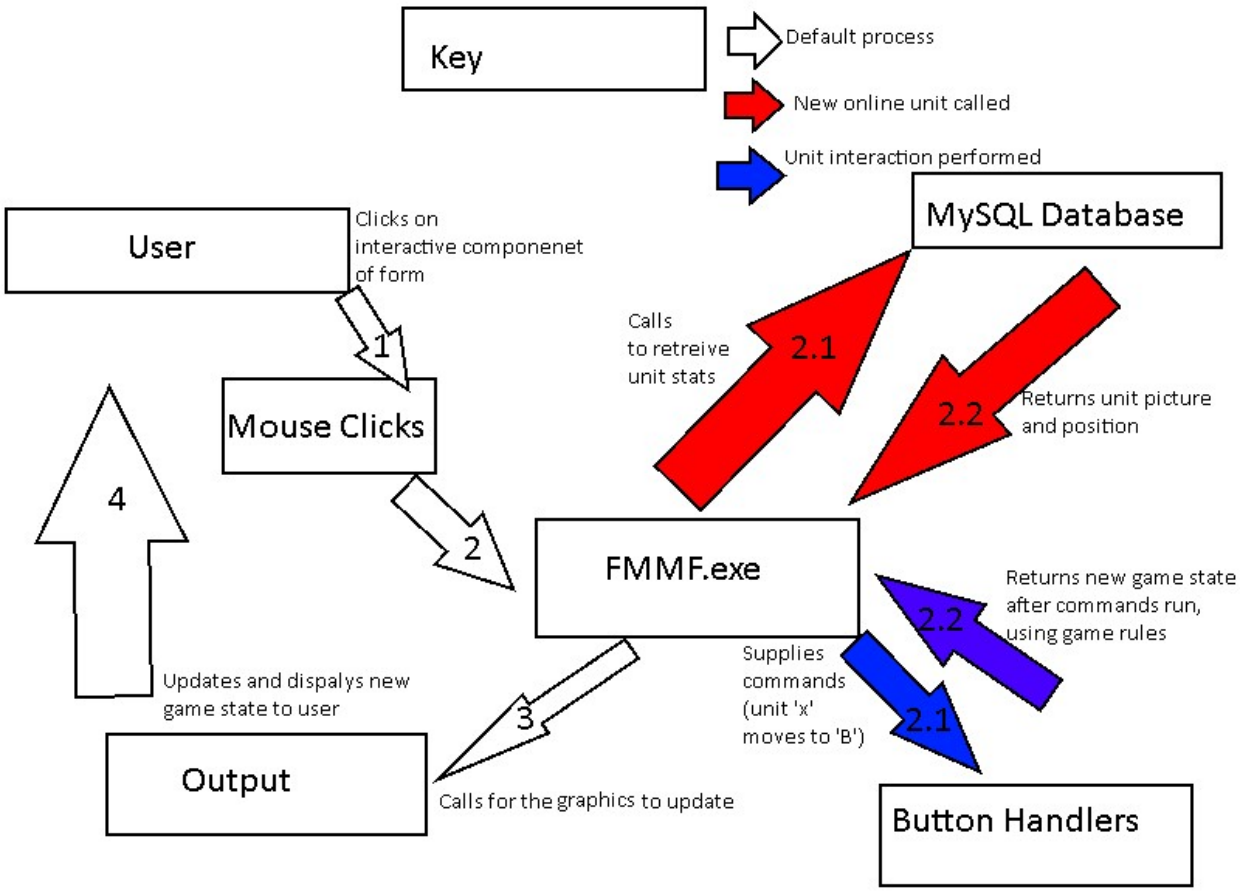
Overview:

Five Move Murder Fest is an RPG system developed by Starshiplad for at-home games http://www.starshiplad.com/FMMF.pdf .
This program will, in its final form, be able to automate a campaign for one computer (Either one palyer playing multiple characters or multiple people taking turns playing each of their own characters). The program will generate events the players can interact wth, randomly generate combat encounters that use the acctual stats to resolve 'rolls', and will perform NPC and enemy actions using simplistic decision making.
The program will feature sound and several menu bar items.
The program will run on windows 7 - > windows 10. If time allows it will be able to be multiple resolutions, however the default is 800x450 px.
The program will run via an executable, and although it will have the ability to connect to a remote database, it will be entirely runnable offline.

UML Diagram:



Application Process Flow:



Method Descriptions:

Unit:

Unit(BufferedImage Image,int x,int y):Initializes a new instance of the Unit class, alive,with an acc of 50 and unacted.
Unit has given image, xpos and ypos.

MainForm:

Dispose(bool disposing) : Clean up any resources being used.
InitializeComponent(): Runs set-up operations and intializes relevant componenets on *MainForm* start up
---
button1_Click(object sender, EventArgs e):
button1_Click deals with the SQL loading of several variables from an SQL database. It then dispalys output text to a console and textbox and draws

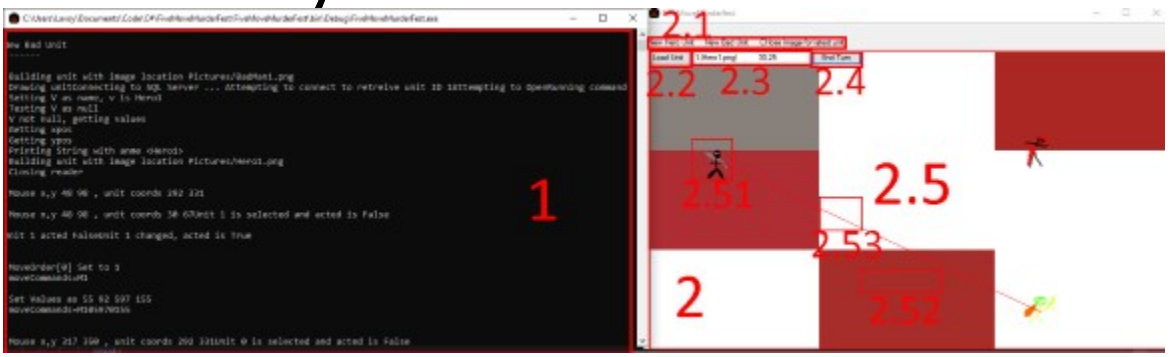
---
PainterForm_MouseDown(object sender, MouseEventArgs e):
Handles any mouse down events performed on MainForm.
If a unit is clicked or already selected, it handles the logic for adding a command string to be run in order when *button_2 Click* is run
---
DrawTerrain(Graphics graphics,int ammount):
Utility method to draw all terrain as their respective colros based on terrain type
---
DrawUnits(Graphics graphics, int i):
Utility method to iterativly call all units currently initialized
---
DrawLines(Graphics graphics,Color drawColor):
Utility method to draw the lines between all actioned units and their action target
---
PainterForm_MouseUp(object sender, MouseEventArgs e):
When using the 'paint on mouse' mosue down enviroment, tells the program to stop drawing
---
PainterForm_MouseMove(object sender, MouseEventArgs e):
Animates the potential target of selected unit if a unit is selected.
Draws continously if mouse down and 'paint on mouse' trues
---
toolBar_ButtonClick
(object sender, ToolBarButtonClickEventArgs e):
Utility method to handle any clicks on the toolbar. Checks aganst the button's 'Name' value
---
playSound(int type):
A utility function to paly a sound.
As of this build:
The type of sound:
1=move,2=shoot
---

button2_Click(object sender,EventArgs e):
Listner class that handles the stored 'Command strings', and the changes they produce, drawing any required.CommandStrings are built in *Mouse down*

Program:

*Main():*Runs when the .exe is first run, creates an instance of *MainForm*

Window Layout:



1:
Console- This is for the debug build only. Displays any significant callback messages or changes to the program that occur.
2:
MainForm- The main application program that the user can interact with and that any important changes are drawn to.
2.1:
(Note: Toolbar's height must be taken into account when drawing to mainform)
Toolbar- Useful features not specificaly part of the game will be drawn at the top of the window.
2.2:
(Note: 2.2-2.4 will probably be moved to toolbar in future builds)
Load unit- Makes a call to a mySQLserver to retreve the next available unit data, then draws that unit onto the mainform.
2.3:
Textbox- Textbox is currently a redundent debug reporter repalced

by 'Console' [1]. It will eventually be the output of narrative interactions relevant to the palyer (E.G instead of "unit 3 fired at coords 123,456 and got a score of 45.7", it will say "Billy Bob fired at a bandit but missed!")

2.4:

End Turn- Most important part of the game logic componenet of the progrma. On click performs all recorded commands in order of initiative, updates game state once all actions are performed, then updates UI and resets unit states.

2.5:

MainForm Drawing Space-

The area of the MainForm where the 'game' will be drawn and interactable

2.51:

Unit-

A catch all term for interactible objects in the game. Currently all units can be clicked, then assigned orders (See actionLine [2.53]), however it will be updated so only the palyer's units can be interacted with by the player. Unit's contain individual images that are drawn to the MainForm Drawing Space at the coordinates the unit exsists in.

2.52:

Terrain-

The MainFormDrawing Space is segmented into areas of terrain. These represent the various aspects of a combat area and impac some game logic based on if a targeted unit is 'in' the terrain

2.53:

actionLine-

When a unit is selected and given an order, the order will not be performed until the end of turn (See End Turn [2.4]). Until then, a line between the unit and their target(Either to move to or shoot at) will dispaly on the MainForm Drawing Space

Development Log:

27/09/19-

Added a documentation log, added and icon to program, build .exe

---

Changes:

>Began this developer documentation, Added UML diagram Application process flow diagram, method descriptors and Window layout diagram.
>Added the FMMF_icon.ico file, changed prgram icon to said .ico.
>Removed the developer Console before publishing an .exe of current program.
>Release 0.01 build to Github

To Do:
-Implement basic enemy Decision Making
-Implement initative system
-Make terrain more asthetic