```cpp

#include <iostream>
#include "MainMenu.h"
#include "Game.h"


/// <summary>
/// @author Peter Lowe
/// @version 1.0
/// @date May2016
///
/// </summary>

MainMenu::MainMenu()
{
}


MainMenu::~MainMenu()
{
}


/// <summary>
/// @brief setup font and message.
///
/// Used to get a refrence to the font and
/// load texture and setup sprite and text
/// for each button and positions on screen
/// </summary>
/// <param name="font">Refrence to font loaded in main game</param>
void MainMenu::initialise(sf::Font& t_font)
{
    m_topOffset = 50;
    m_verticalSpacing = 100;
    m_buttonWidth = 200;
    m_leftOffset = (Game::s_screenWidth - m_buttonWidth)/2;
    m_buttonHeight = 50;
    int textDropOffset = 10;
    sf::String m_menuTexts[] = { "Play", "Help", "Exit" };

    m_font = t_font;
    if (!m_buttonTexture.loadFromFile("ASSETS/IMAGES/button.png"))
    {
        std::cout << "error with button file";
    }
    for (int i = 0; i < m_optionCount; i++)
    {
        m_buttonSprites[i].setTexture(m_buttonTexture);
        m_buttonSprites[i].setPosition(m_leftOffset, m_verticalSpacing * i +
          m_topOffset);
        sf::Vector2u textureSize = m_buttonTexture.getSize();
```

```
52          m_buttonSprites[i].setScale(m_buttonWidth/textureSize.x, m_buttonHeight/
              textureSize.y);
53
54          m_buttonTexts[i].setFont(m_font);
55          m_buttonTexts[i].setString(m_menuTexts[i]);
56          m_buttonTexts[i].setFillColor(sf::Color::White);
57          m_buttonTexts[i].setCharacterSize(24);
58          sf::FloatRect textSize = m_buttonTexts[i].getGlobalBounds();
59          float textOffset = (m_buttonWidth - textSize.width) / 2;
60          m_buttonTexts[i].setPosition(m_leftOffset + textOffset, m_verticalSpacing
              * i + m_topOffset + textDropOffset);
61      }
62  }
63
64
65  /// <summary>
66  /// check current status of the mouse for intersection with
67  /// location of buttons using locations and offsets rather than rectangles
68  /// for intersection
69  ///
70  /// Mouse button down will be true at this location on the next screen /
      gamestate
71  /// might cause a problem if so use event polling loop to detect mouse button
      down event
72  /// </summary>
73  /// <param name="time">update delta time</param>
74  /// <param name="window">refrence to main render window</param>
75  void MainMenu::update(sf::Time t_deltaTime, sf::Window& t_window)
76  {
77      if (sf::Mouse::isButtonPressed(sf::Mouse::Button::Left))
78      {
79          sf::Vector2i mouseLocation;
80          mouseLocation = sf::Mouse::getPosition(t_window);
81          if (mouseLocation.x > m_leftOffset && mouseLocation.x < m_leftOffset +
              m_buttonWidth )
82          {
83              if (mouseLocation.y > m_topOffset && mouseLocation.y &&
                    mouseLocation.y <  m_topOffset + m_buttonHeight)
84              {
85                  Game::currentState = GameState::Game;
86              }
87              if (mouseLocation.y > m_topOffset + m_verticalSpacing &&
                    mouseLocation.y <  m_topOffset + m_verticalSpacing +
                    m_buttonHeight)
88              {
89                  Game::currentState = GameState::Help;
90              }
91              if (mouseLocation.y > m_topOffset + m_verticalSpacing * 2 &&
                    mouseLocation.y <  m_topOffset + m_verticalSpacing * 2 +
                    m_buttonHeight)
92              {
93                  t_window.close();
```

```
94              }
95          }
96      }
97  }
98
99  /// <summary>
100 ///  draw menu text over buttons
101 /// /// clear and display methods will be called in game.cpp
102 /// </summary>
103 /// <param name="window">refrence to main render window</param>
104 void MainMenu::render(sf::RenderWindow& t_window)
105 {
106     for (int i = 0; i < m_optionCount; i++)
107     {
108         t_window.draw(m_buttonSprites[i]);
109         t_window.draw(m_buttonTexts[i]);
110     }
111 }
```