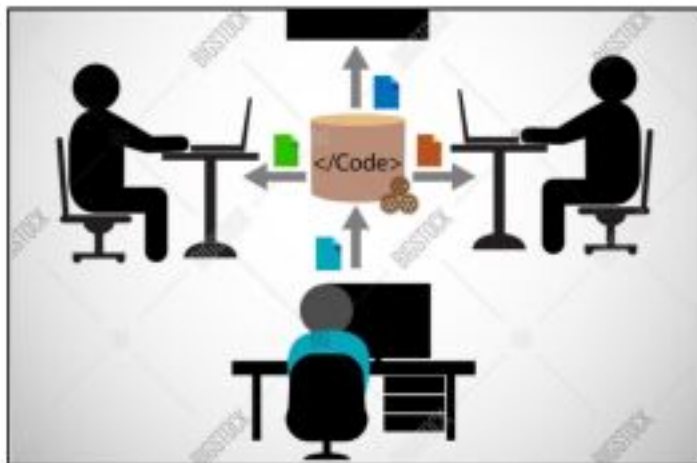


CRIMSON ZOMBIE



1. Create a project called 'Level Editor'

2. Add a file called 'Source.cpp'

3. Add a file called 'App.h'

4. Add a file called 'App.cpp'

ADD A
'SOURCE.CPP'
FILE ...



AND SET THE PROJECT
PROPERTIES!

```
// main function, place in 'Source.cpp'


#include "App.h"

int main()
{
    App game("Level Editor", 800, 600, 32);

    if(!game.Init())
    {
        printf("Game could not be started!");
        return 1;
    }
    else {
        game.Run();
    }
    return 0;
}
```

```
// App class definition, place in 'App.h'
```

```
#pragma once
```



ENSURES FILE IS
INCLUDED ONLY ONCE

```
#include "SFML/Graphics.hpp"
```

```
class App
```

```
{
```

```
    private:
```

```
        sf::Event          event;
```

```
        sf::View           view;
```

```
        sf::RenderWindow   window;
```

```
        // other data members here
```

```
// continued on next page
```

```
// continued from previous page
```

```
public:
```

```
    App(const char* title,    int screenWidth,  
         int screenHeight, int screenBpp);
```

```
    ~App();
```

```
    bool Init();
```

```
    void HandleEvents()
```

```
    void Draw();
```

```
    void Update();
```

```
    void Run();
```

```
};
```

```
// end of App class definition
```

```
// App method definitions, place in 'App.cpp'

#include "App.h"

// constructor
App::App(const char* title, int screenWidth,
         int screenHeight, int screenBpp)
{
    window.create(
        sf::VideoMode(screenWidth,
                        screenHeight,
                        screenBpp),
        title);

    window.setFrameLimit(0);
    view = window.getDefaultView();
}
```

```
// App method definitions, place in 'App.cpp'
```

```
// destructor
```

```
App::~App()
```

```
{
```

```
    // release memory
```

```
}
```

```
bool App::Init()
```

```
{
```

```
    // initialise App data members
```

```
    return true;
```

```
}
```



```
// App method definitions, place in 'App.cpp'
```

```
void App::Update()  
{  
    // update  
}
```

```
void App::Draw()  
{  
    window.clear();  
    window.setView(view);  
    // draw  
    window.display();  
}
```

```
// App method definitions, place in 'App.cpp'

void App::HandleEvents()
{
    if(event.type == sf::Event::Closed)
        window.close();

    // other keyboard, mouse events
}
```

```
// App method definitions, place in 'App.cpp'
```

```
void App::Run()  
{  
    while(window.isOpen()) {  
        while(window.pollEvent(event)) {  
            HandleEvents();  
        }  
        Update();  
        Draw();  
    }  
}
```

```
// end of App method definitions
```

1. task description

4. task description

2. task description

5. task description

3. task description

6. task description