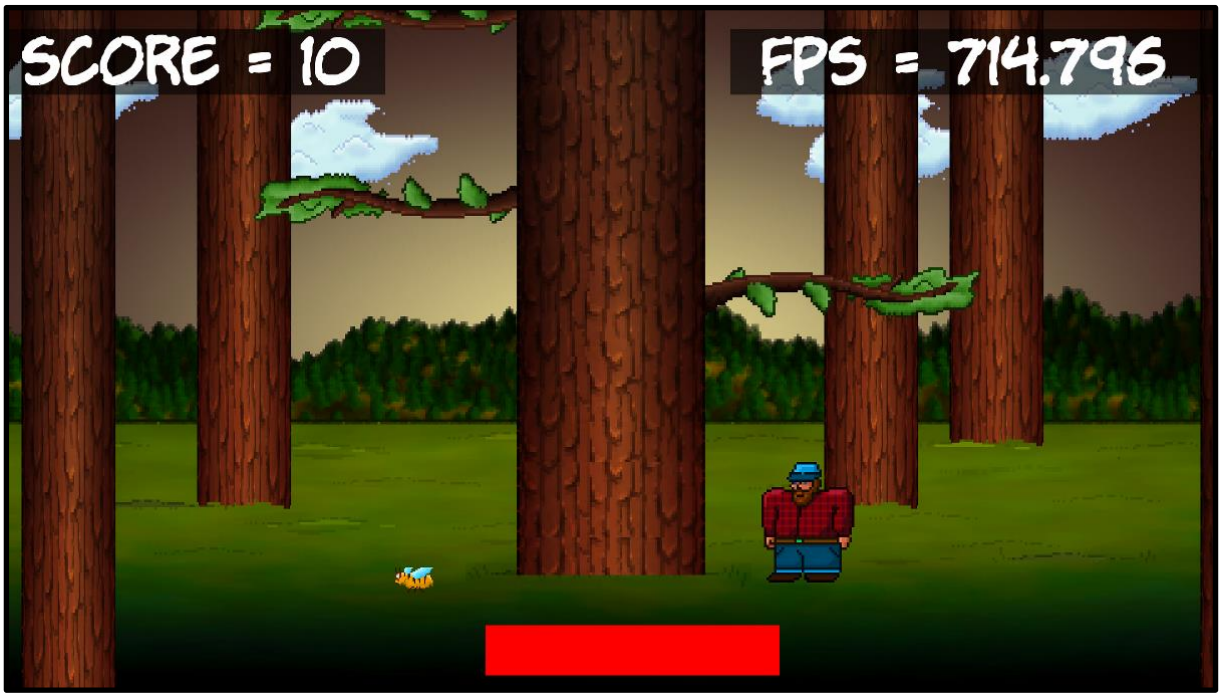
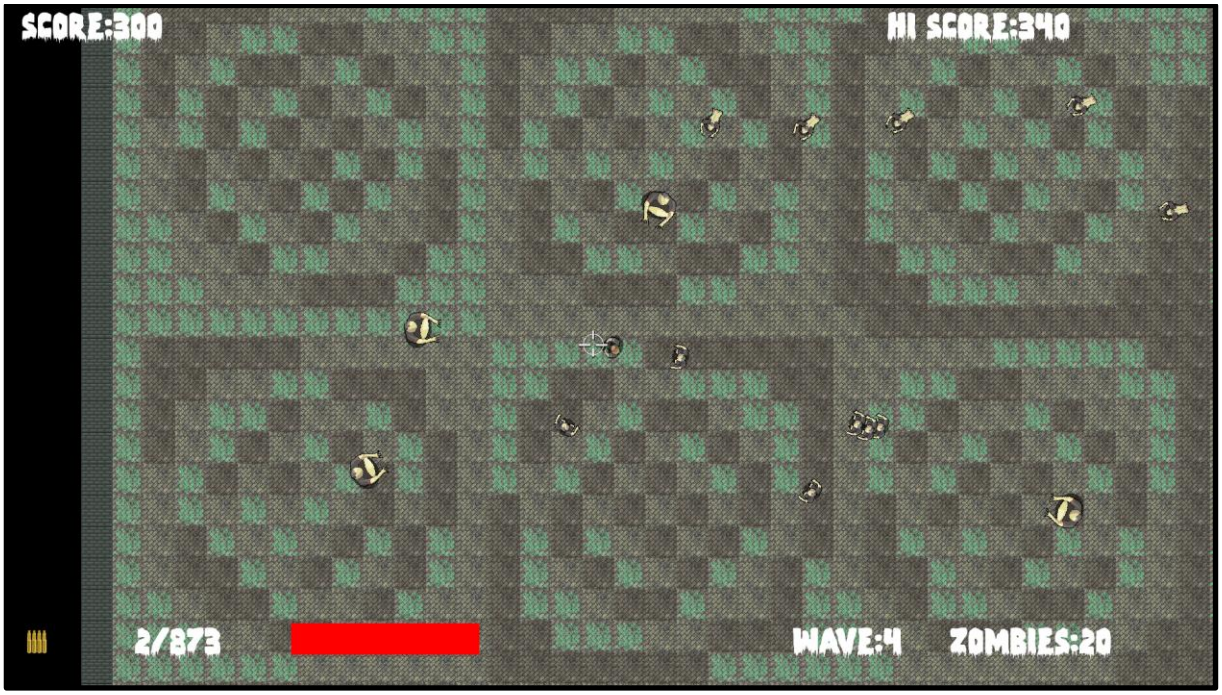
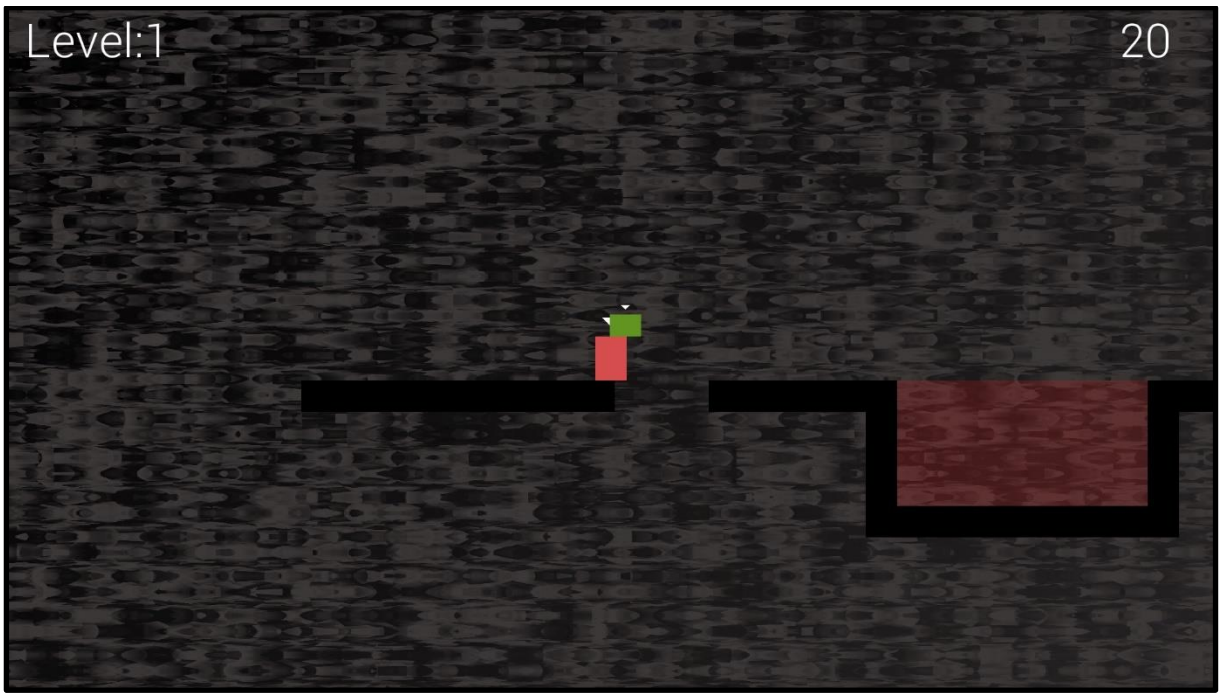
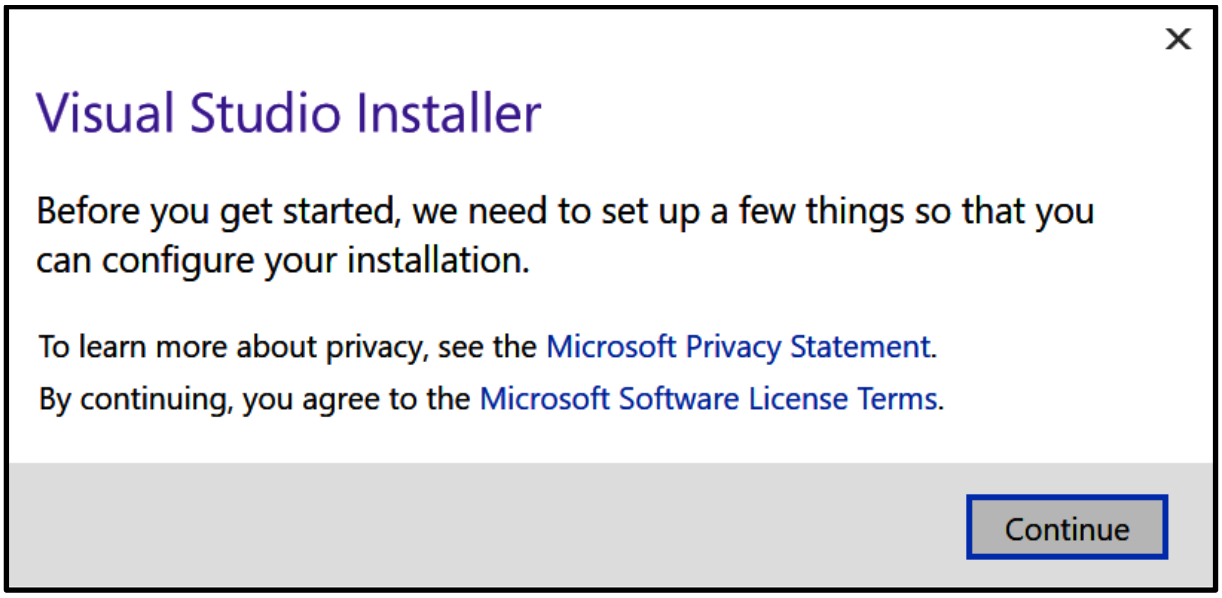
Chapter 1: C++, SFML, Visual Studio, and Starting the First Game

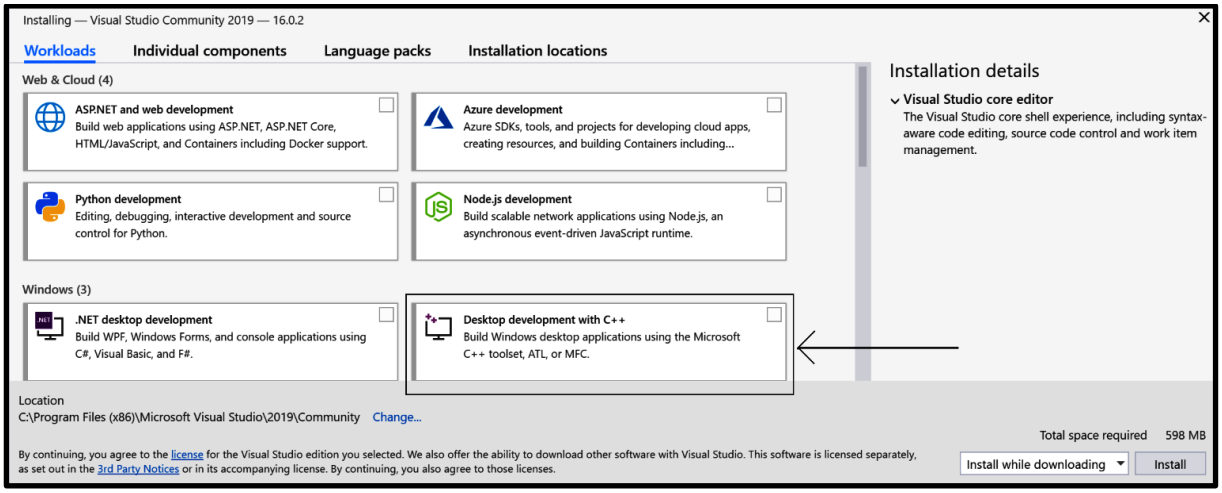



SCORE:300
HI SCORE:340
2/873
WAVE:4
ZOMBIES:20


Level:1
20


Visual Studio 2019
Code faster. Work smarter. Create the future with the best-in-class IDE.
Download Visual Studio
Community 2019
Professional 2019
Enterprise 2019
Tools
Test
Analyze
Window
Help
Attach...
Quick


Visual Studio Installer
Before you get started, we need to set up a few things so that you can configure your installation.
To learn more about privacy, see the Microsoft Privacy Statement.
By continuing, you agree to the Microsoft Software License Terms.
Continue


Installing — Visual Studio Community 2019 — 16.0.2
Workloads
Individual components
Language packs
Installation locations
Web & Cloud (4)
ASP.NET and web development
Build web applications using ASP.NET, ASP.NET Core, HTML/JavaScript, and Containers including Docker support.
Azure development
Azure SDKs, tools, and projects for developing cloud apps, creating resources, and building Containers including...
Python development
Editing, debugging, interactive development and source control for Python.
Node.js development
Build scalable network applications using Node.js, an asynchronous event-driven JavaScript runtime.
Windows (3)
.NET desktop development
Build WPF, Windows Forms, and console applications using C#, Visual Basic, and F#.
Desktop development with C++
Build Windows desktop applications using the Microsoft C++ toolset, ATL, or MFC.
Installation details
Visual Studio core editor
The Visual Studio core shell experience, including syntax-aware code editing, source code control and work item management.
Location
C:\Program Files (x86)\Microsoft Visual Studio\2019\Community
Change...
Total space required
598 MB
By continuing, you agree to the license for the Visual Studio edition you selected. We also offer the ability to download other software with Visual Studio. This software is licensed separately, as set out in the 3rd Party Notices or in its accompanying license. By continuing, you also agree to those licenses.
Install while downloading
Install

Download SFML 2.5.1

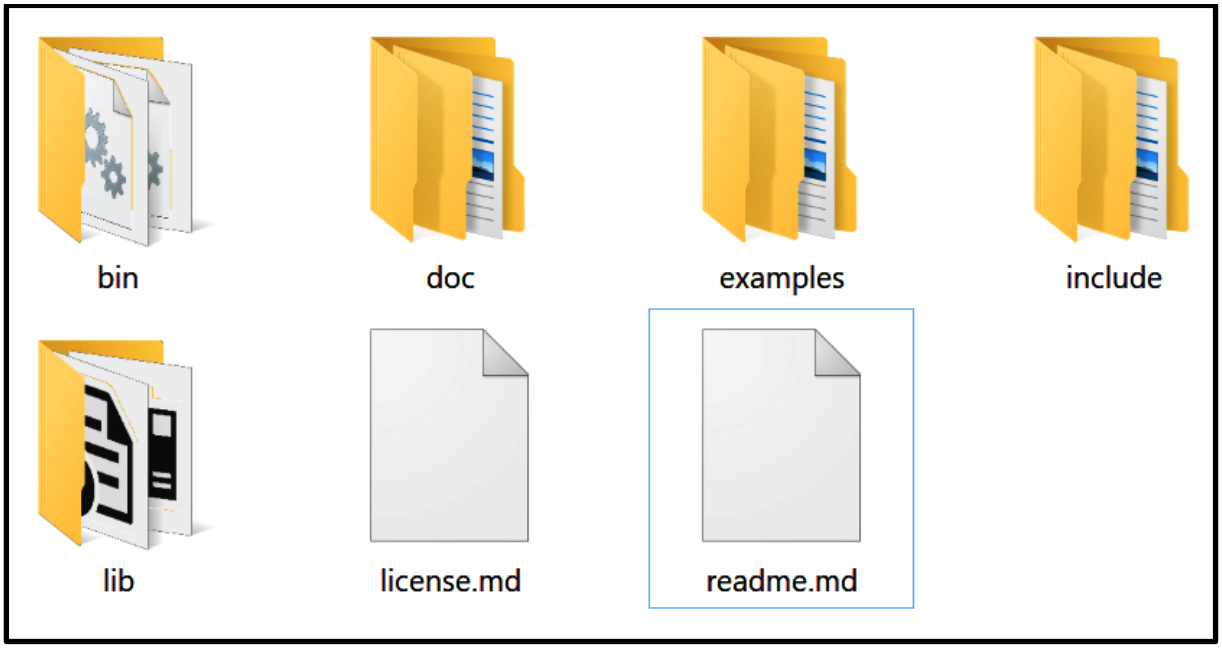
On Windows, choosing 32 or 64-bit libraries should be based on which platform you want to compile for, not which OS you have. Indeed, you can perfectly compile and run a 32-bit program on a 64-bit Windows. So you'll most likely want to target 32-bit platforms, to have the largest possible audience. Choose 64-bit packages only if you have good reasons.

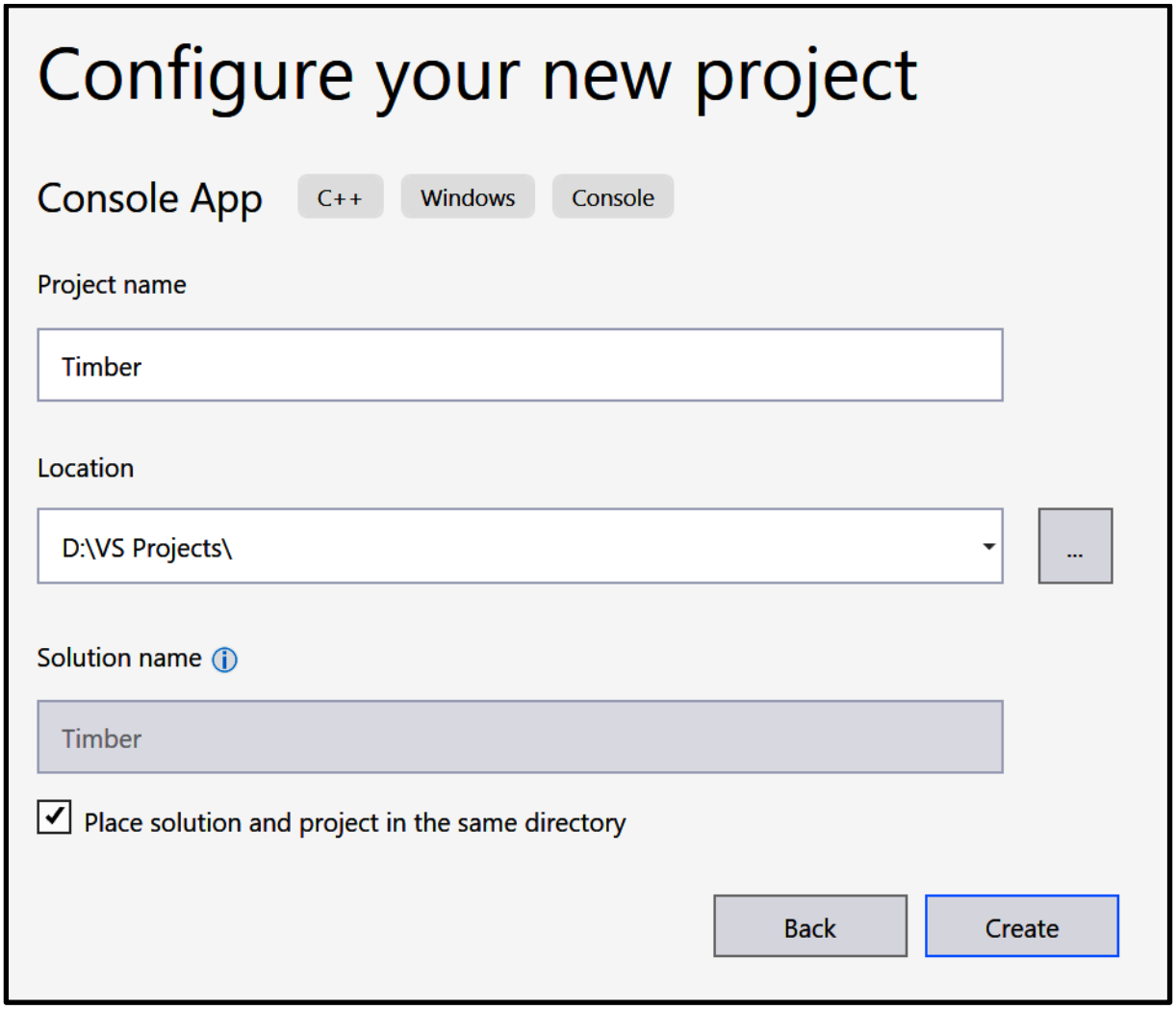
The compiler versions have to match 100%!

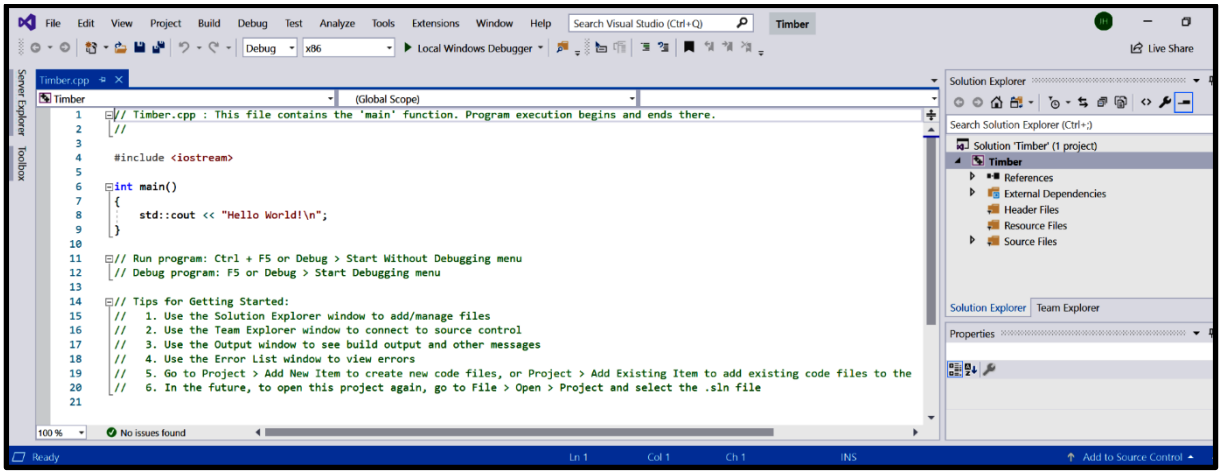
Here are links to the specific MinGW compiler versions used to build the provided packages:

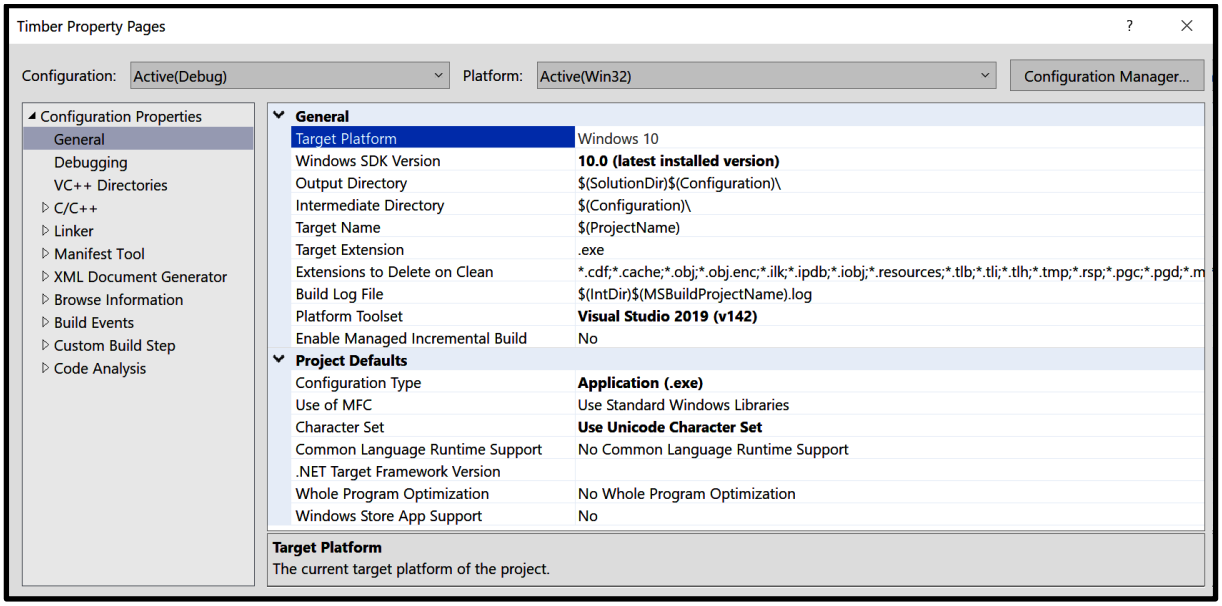
TDM 5.1.0 (32-bit), MinGW Builds 7.3.0 (32-bit), MinGW Builds 7.3.0 (64-bit)

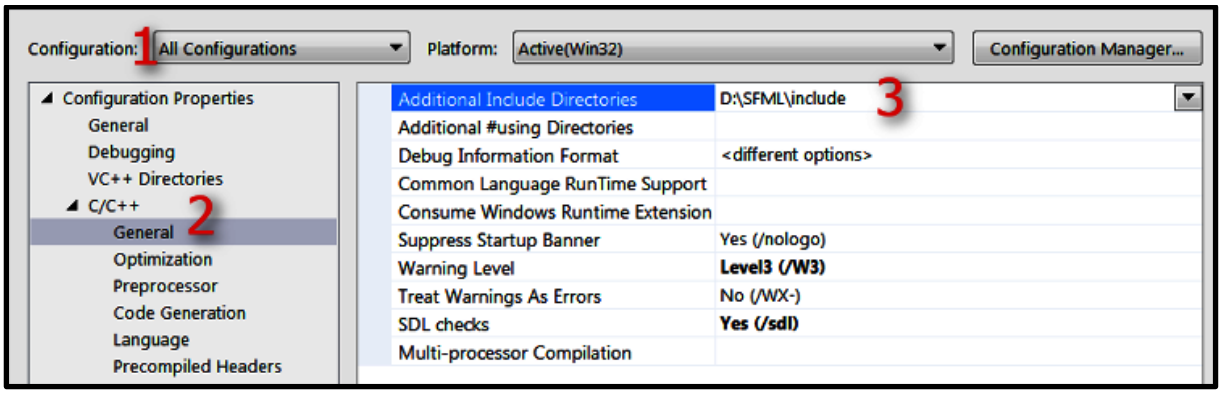
Visual C++ 15 (2017) - 32-bit	Download | 16.3 MB	Visual C++ 15 (2017) - 64-bit	Download | 18.0 MB
Visual C++ 14 (2015) - 32-bit	Download | 18.0 MB	Visual C++ 14 (2015) - 64-bit	Download | 19.9 MB
Visual C++ 12 (2013) - 32-bit	Download | 18.3 MB	Visual C++ 12 (2013) - 64-bit	Download | 20.3 MB
GCC 5.1.0 TDM (SJLJ) - Code::Blocks - 32-bit	Download | 14.1 MB		
GCC 7.3.0 MinGW (DW2) - 32-bit	Download | 15.5 MB	GCC 7.3.0 MinGW (SEH) - 64-bit	Download | 16.5 MB

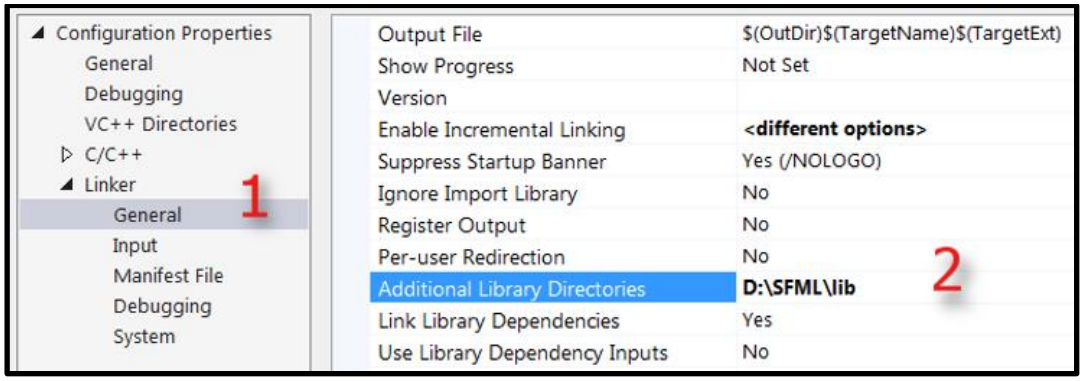

bin
doc
examples
include
lib
license.md
readme.md

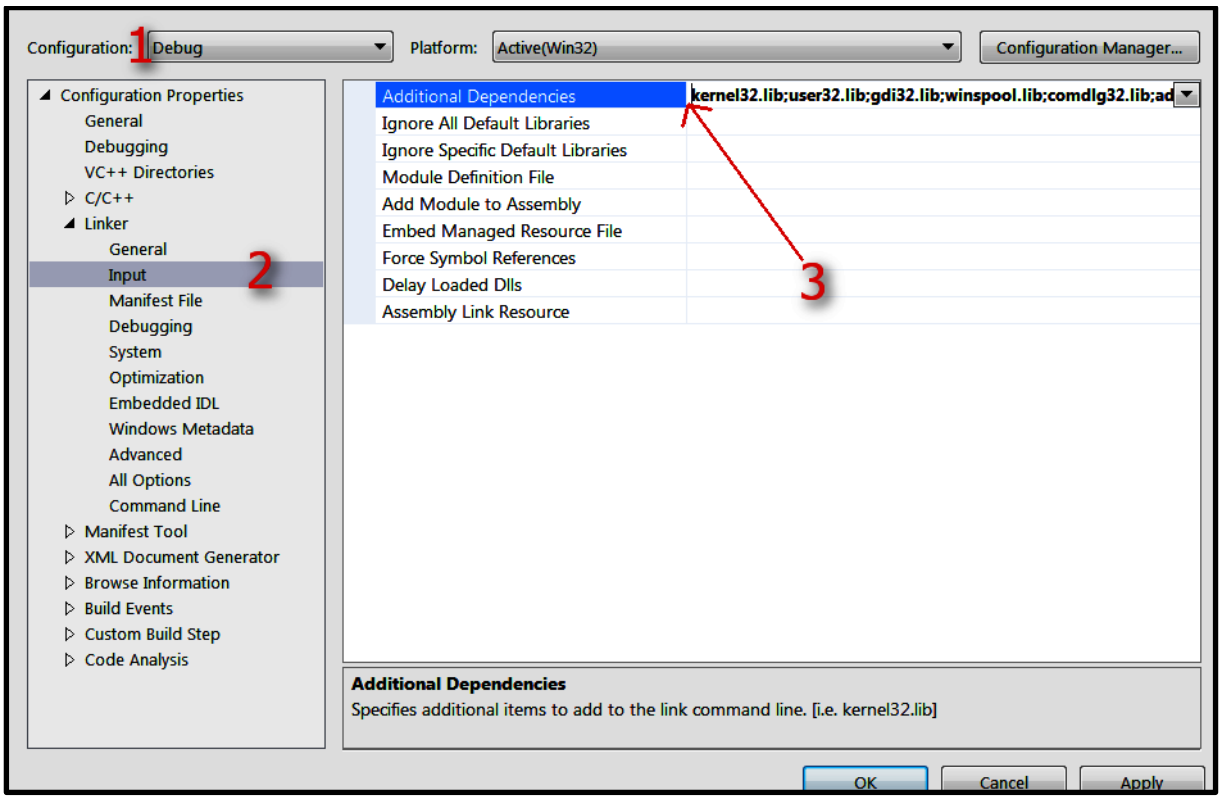

Configure your new project
Console App
C++
Windows
Console
Project name
Timber
Location
D:\VS Projects\
...
Solution name
Timber
Place solution and project in the same directory
Back
Create

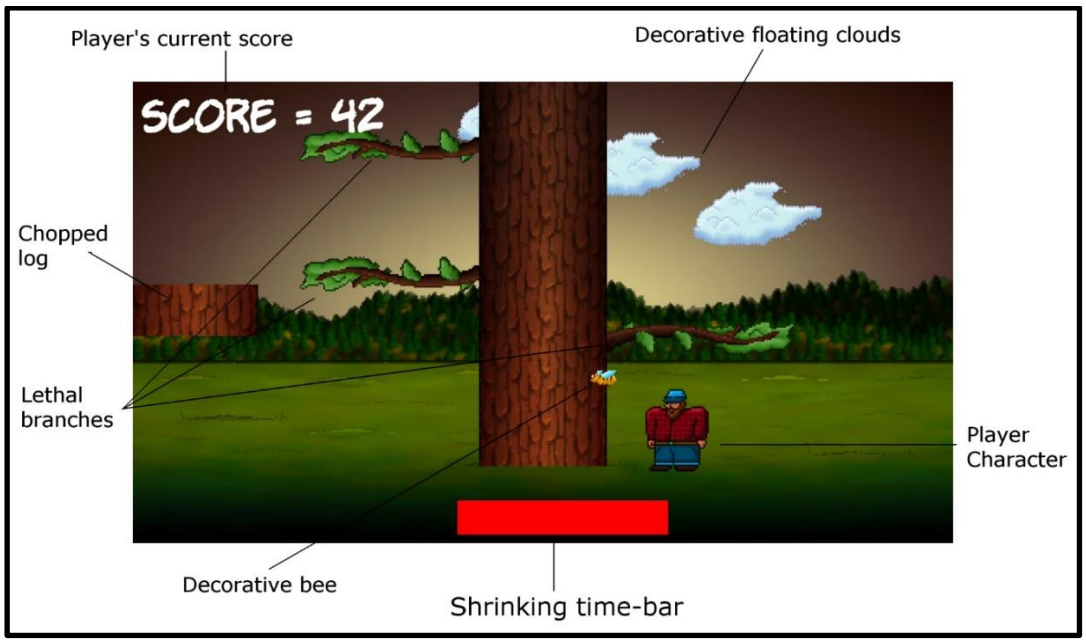

File Edit View Project Build Debug Test Analyze Tools Extensions Window Help
Search Visual Studio (Ctrl+Q)
Timber
Debug
x86
Local Windows Debugger
Live Share
Timber.cpp
(Global Scope)
// Timber.cpp : This file contains the 'main' function. Program execution begins and ends there.
//
#include <iostream>
int main()
{
std::cout << "Hello World!\n";
}
// Run program: Ctrl + F5 or Debug > Start Without Debugging menu
// Debug program: F5 or Debug > Start Debugging menu
// Tips for Getting Started:
// 1. Use the Solution Explorer window to add/manage files
// 2. Use the Team Explorer window to connect to source control
// 3. Use the Output window to see build output and other messages
// 4. Use the Error List window to view errors
// 5. Go to Project > Add New Item to create new code files, or Project > Add Existing Item to add existing code files to the
// 6. In the future, to open this project again, go to File > Open > Project and select the .sln file
Solution Explorer
Search Solution Explorer (Ctrl+;)
Solution 'Timber' (1 project)
Timber
References
External Dependencies
Header Files
Resource Files
Source Files
Solution Explorer
Team Explorer
Properties
100 %
No issues found
Ready
Ln 1
Col 1
Ch 1
INS
Add to Source Control

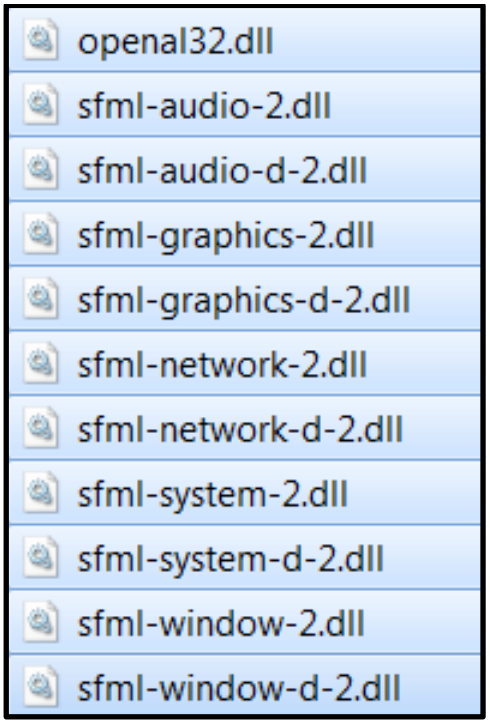

Timber Property Pages
Configuration: Active(Debug)
Platform: Active(Win32)
Configuration Manager...
Configuration Properties
General
Debugging
VC++ Directories
C/C++
Linker
Manifest Tool
XML Document Generator
Browse Information
Build Events
Custom Build Step
Code Analysis
General
Target Platform Windows 10
Windows SDK Version 10.0 (latest installed version)
Output Directory $(SolutionDir)$(Configuration)\
Intermediate Directory $(Configuration)\
Target Name $(ProjectName)
Target Extension .exe
Extensions to Delete on Clean *.cdf;*.cache;*.obj;*.obj.enc;*.ilk;*.ipdb;*.iobj;*.resources;*.tlb;*.tli;*.tlh;*.tmp;*.rsp;*.pgc;*.pgd;*.m
Build Log File $(IntDir)$(MSBuildProjectName).log
Platform Toolset Visual Studio 2019 (v142)
Enable Managed Incremental Build No
Project Defaults
Configuration Type Application (.exe)
Use of MFC Use Standard Windows Libraries
Character Set Use Unicode Character Set
Common Language Runtime Support No Common Language Runtime Support
.NET Target Framework Version
Whole Program Optimization No Whole Program Optimization
Windows Store App Support No
Target Platform
The current target platform of the project.

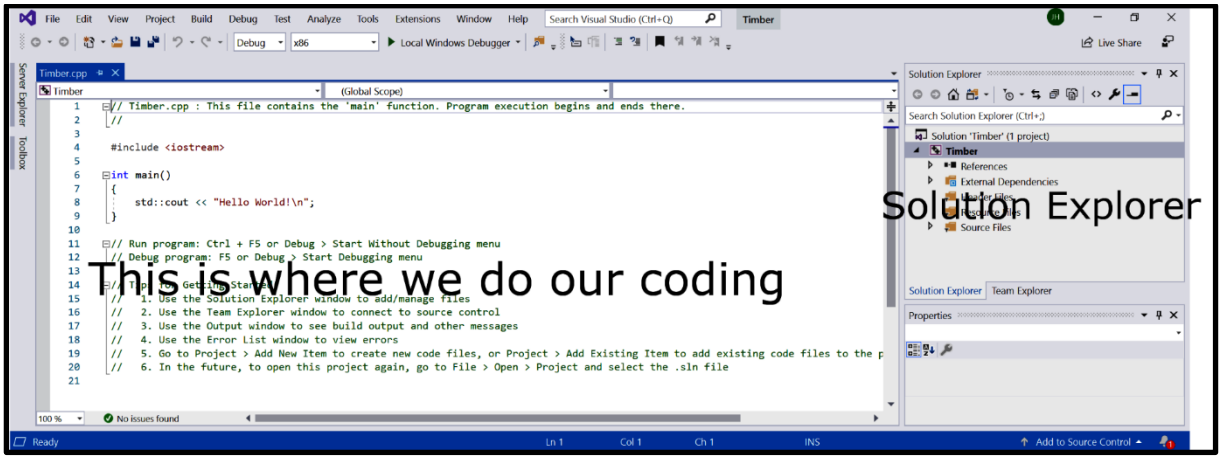

Configuration: 1 All Configurations
Platform: Active(Win32)
Configuration Manager...
Configuration Properties
General
Debugging
VC++ Directories
C/C++
2 General
Optimization
Preprocessor
Code Generation
Language
Precompiled Headers
Additional Include Directories D:\SFML\include 3
Additional #using Directories
Debug Information Format <different options>
Common Language RunTime Support
Consume Windows Runtime Extension
Suppress Startup Banner Yes (/nologo)
Warning Level Level3 (/W3)
Treat Warnings As Errors No (/WX-)
SDL checks Yes (/sdl)
Multi-processor Compilation


Configuration Properties
General
Debugging
VC++ Directories
C/C++
Linker
General
Input
Manifest File
Debugging
System
1
Output File
$(OutDir)$(TargetName)$(TargetExt)
Show Progress
Not Set
Version
Enable Incremental Linking
<different options>
Suppress Startup Banner
Yes (/NOLOGO)
Ignore Import Library
No
Register Output
No
Per-user Redirection
No
Additional Library Directories
D:\SFML\lib
Link Library Dependencies
Yes
Use Library Dependency Inputs
No
2

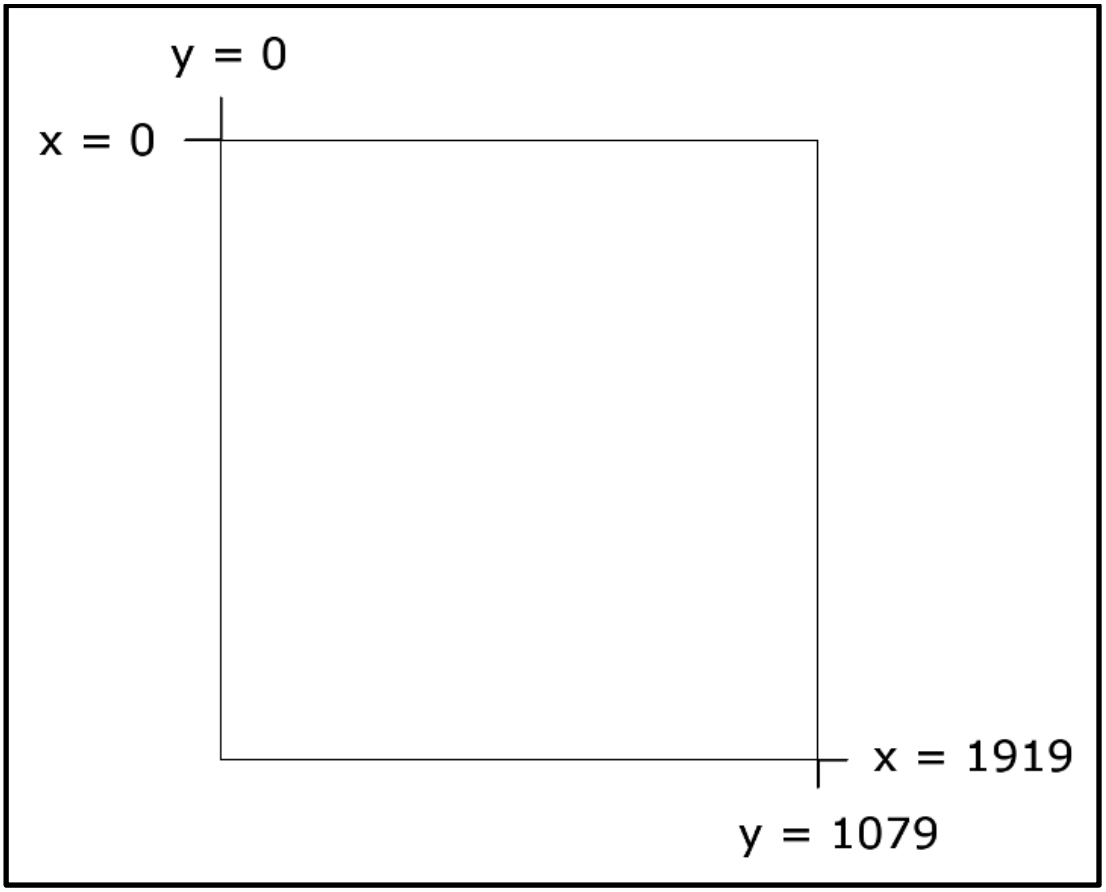

Configuration:
1
Debug
Platform:
Active(Win32)
Configuration Manager...
Configuration Properties
General
Debugging
VC++ Directories
C/C++
Linker
General
Input
2
Manifest File
Debugging
System
Optimization
Embedded IDL
Windows Metadata
Advanced
All Options
Command Line
Manifest Tool
XML Document Generator
Browse Information
Build Events
Custom Build Step
Code Analysis
Additional Dependencies
kernel32.lib;user32.lib;gdi32.lib;winspool.lib;comdlg32.lib;ad
Ignore All Default Libraries
Ignore Specific Default Libraries
Module Definition File
Add Module to Assembly
Embed Managed Resource File
Force Symbol References
Delay Loaded Dlls
Assembly Link Resource
3
Additional Dependencies
Specifies additional items to add to the link command line. [i.e. kernel32.lib]
OK
Cancel
Apply

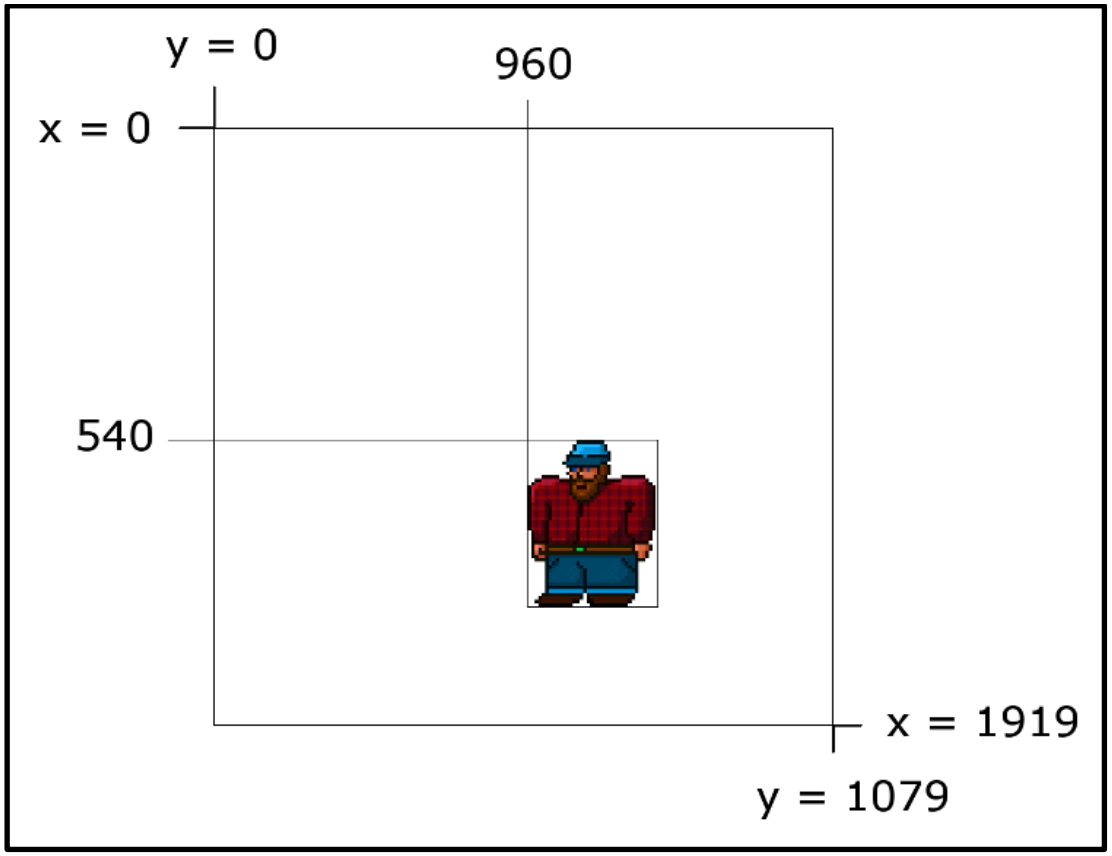

Player's current score
Decorative floating clouds
SCORE = 42
Chopped log
Lethal branches
Player Character
Decorative bee
Shrinking time-bar

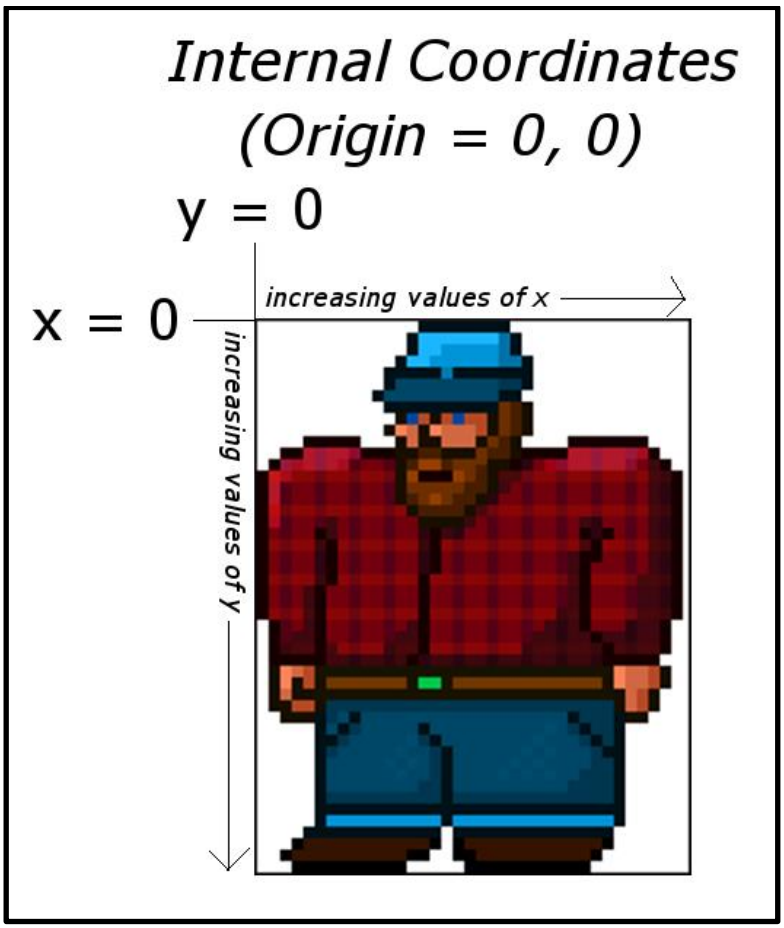

openal32.dll
sfml-audio-2.dll
sfml-audio-d-2.dll
sfml-graphics-2.dll
sfml-graphics-d-2.dll
sfml-network-2.dll
sfml-network-d-2.dll
sfml-system-2.dll
sfml-system-d-2.dll
sfml-window-2.dll
sfml-window-d-2.dll


This is where we do our coding
Solution Explorer

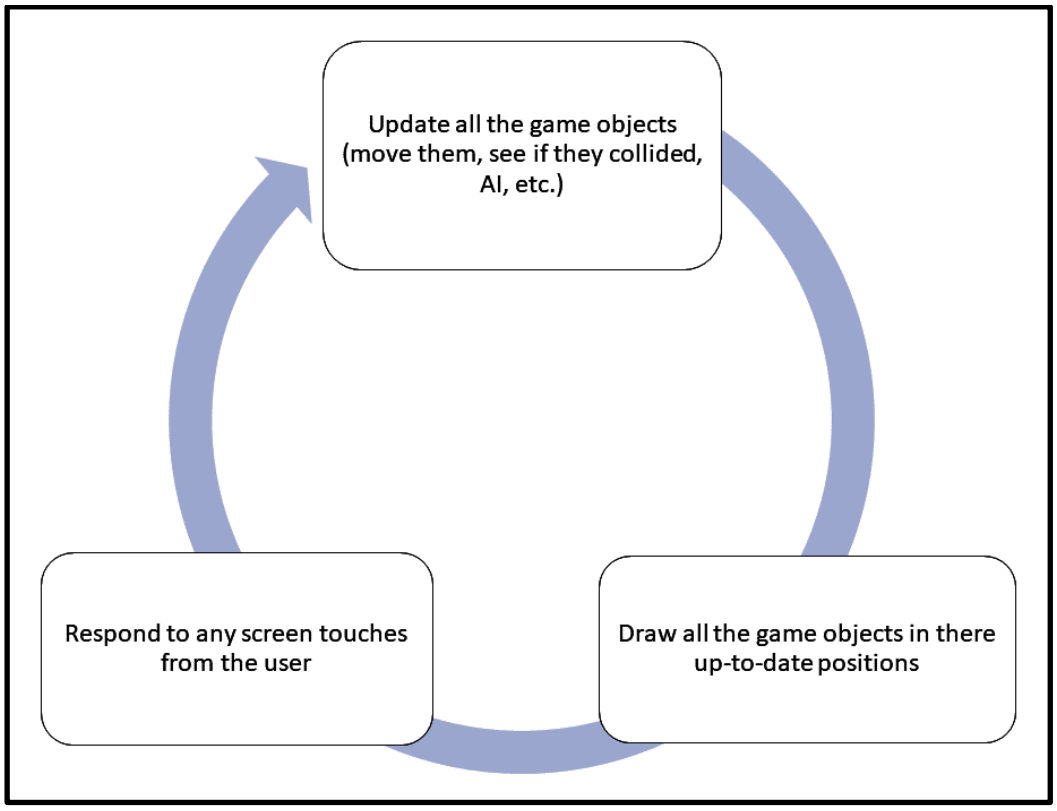

axe
background
bee
branch
cloud
log
player
rip
tree


y = 0
x = 0
x = 1919
y = 1079


y = 0
960
x = 0
540
x = 1919
y = 1079


Internal Coordinates
(Origin = 0, 0)
y = 0
x = 0
increasing values of x
increasing values of y

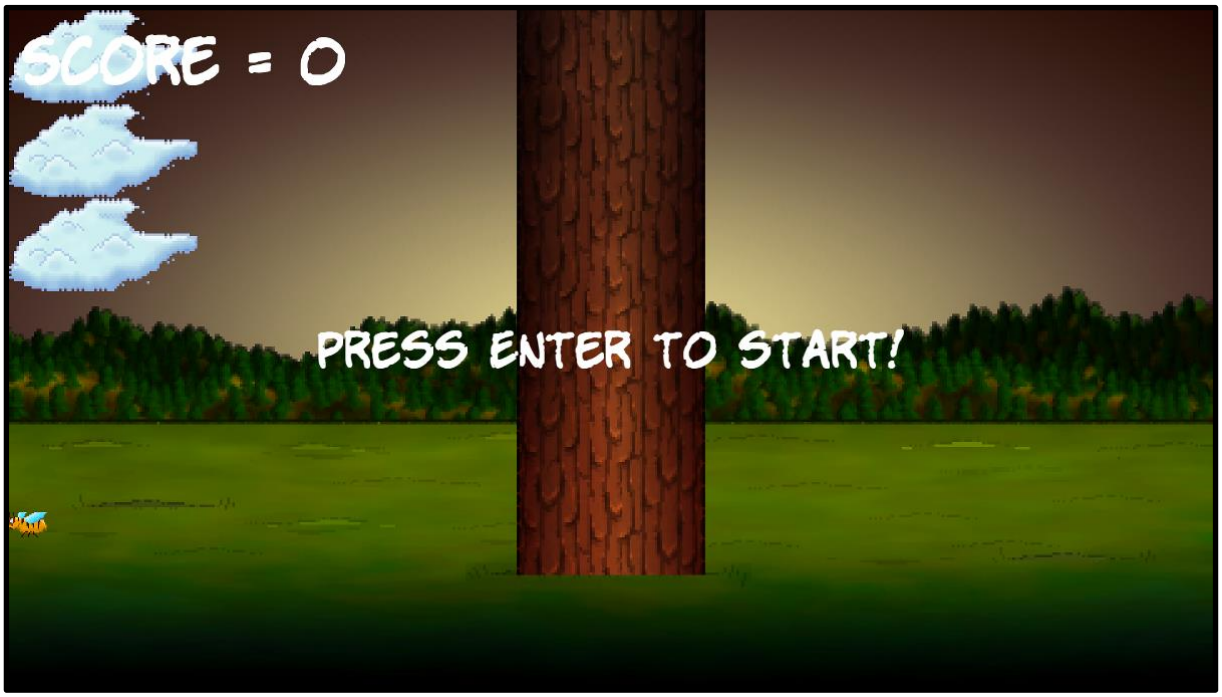

Local Windows Debugger

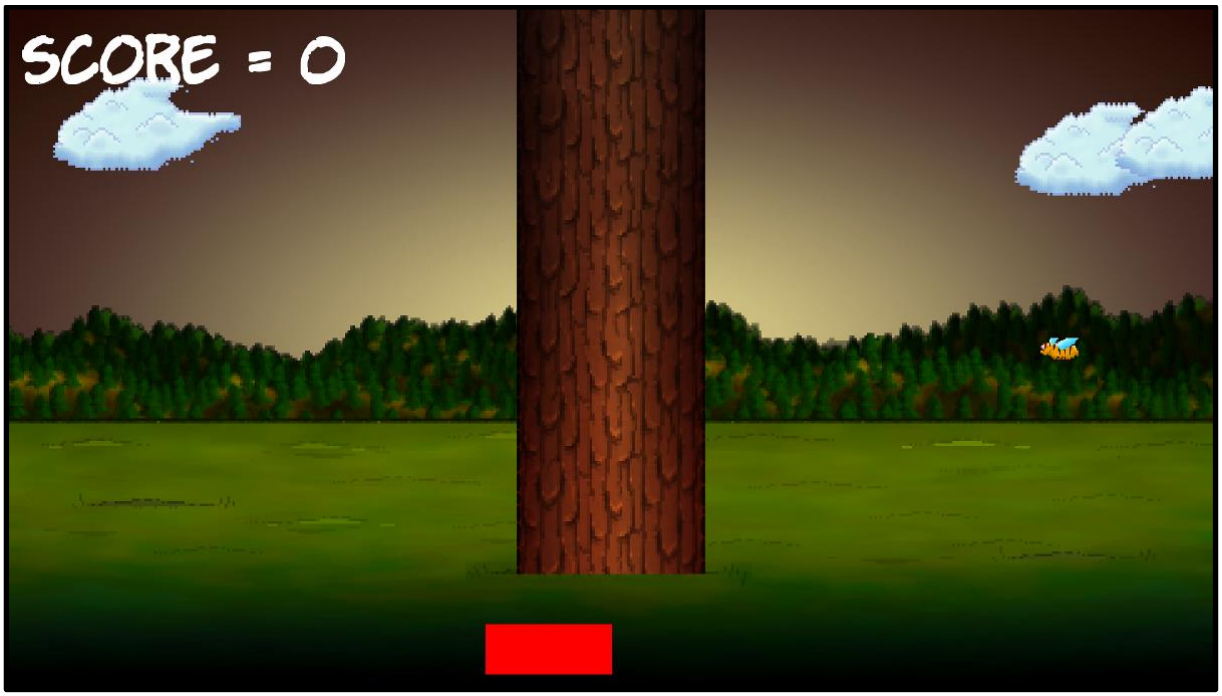

Update all the game objects (move them, see if they collided, AI, etc.)
Draw all the game objects in there up-to-date positions
Respond to any screen touches from the user


Score: 14 Lives: 3 Wave: 1

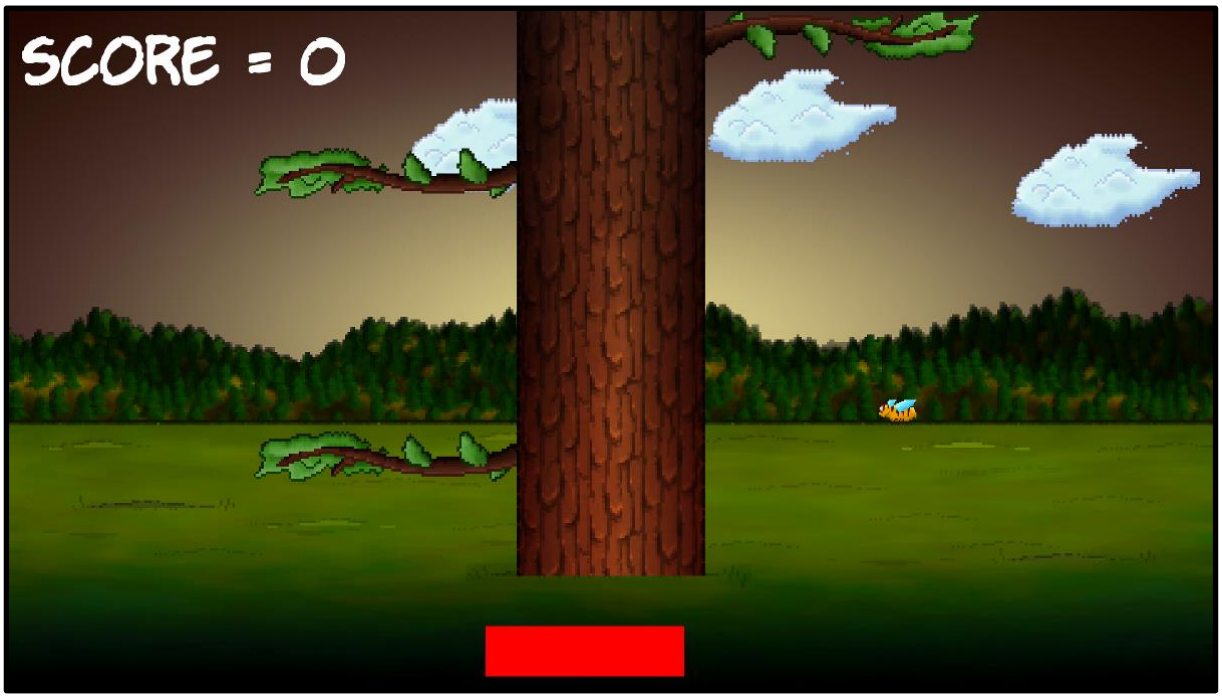
Chapter 2: Variables, Operators, and Decisions – Animating Sprites

Chapter 3: C++ Strings and SFML Time – Player Input and HUD



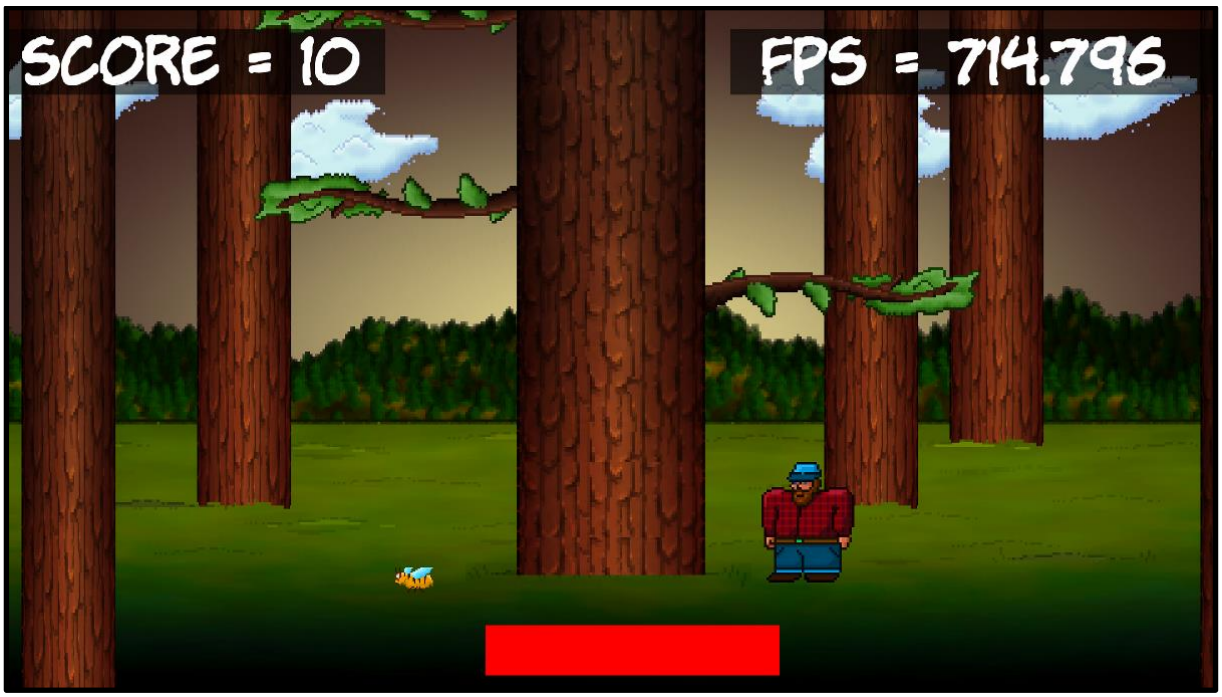


Chapter 4: Loops, Arrays, Switches, Enumerations, and Functions – Implementing Game Mechanics



Chapter 5: Collisions, Sound, and End Conditions – Making the Game Playable

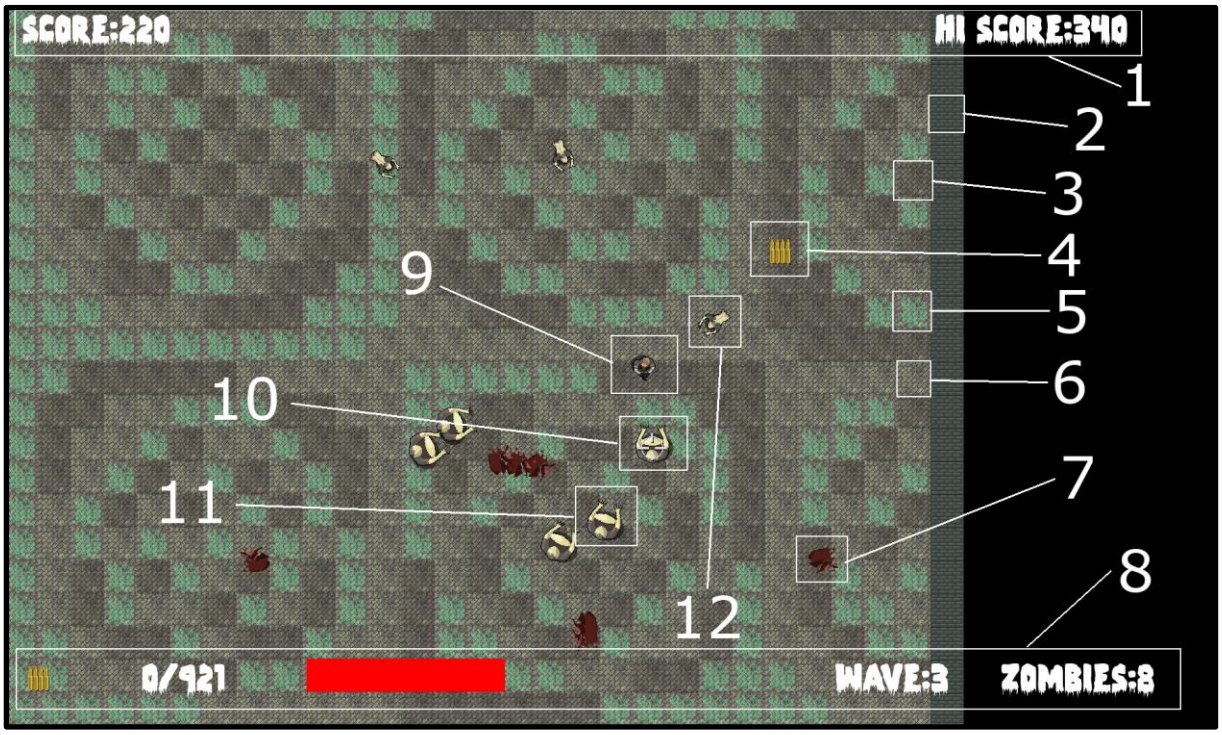

SCORE = 5
SQUISHED!!
RIP

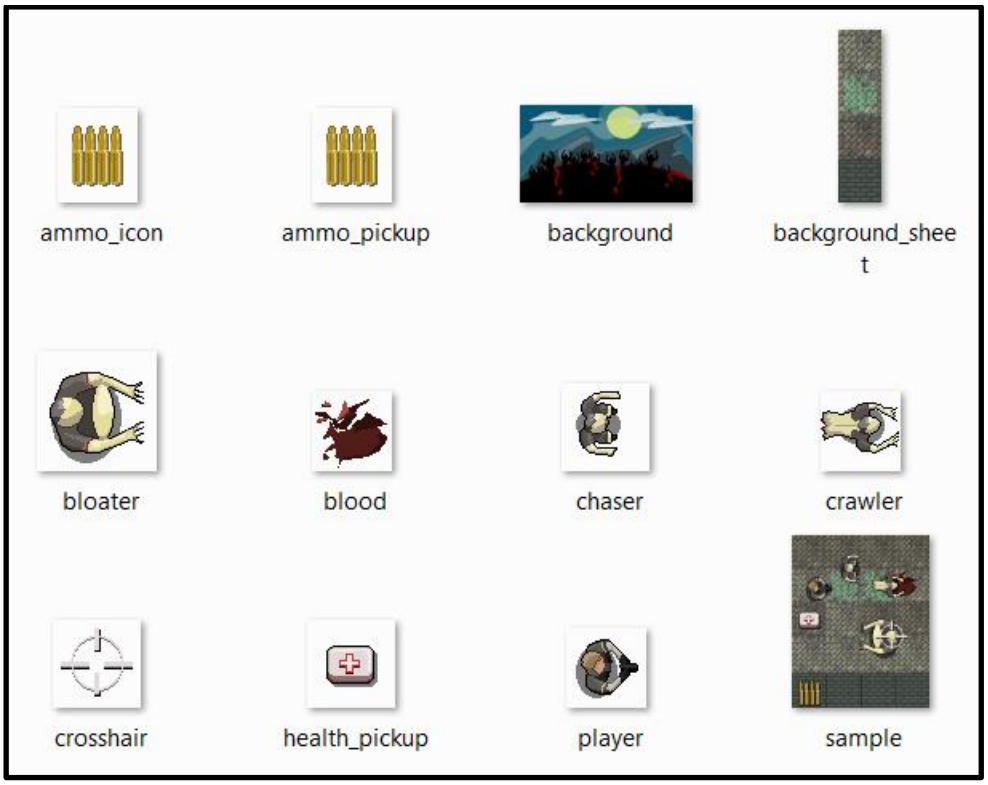

SCORE = 10
FPS = 714.796

Chapter 6: Object-Oriented Programming – Starting the Pong Game

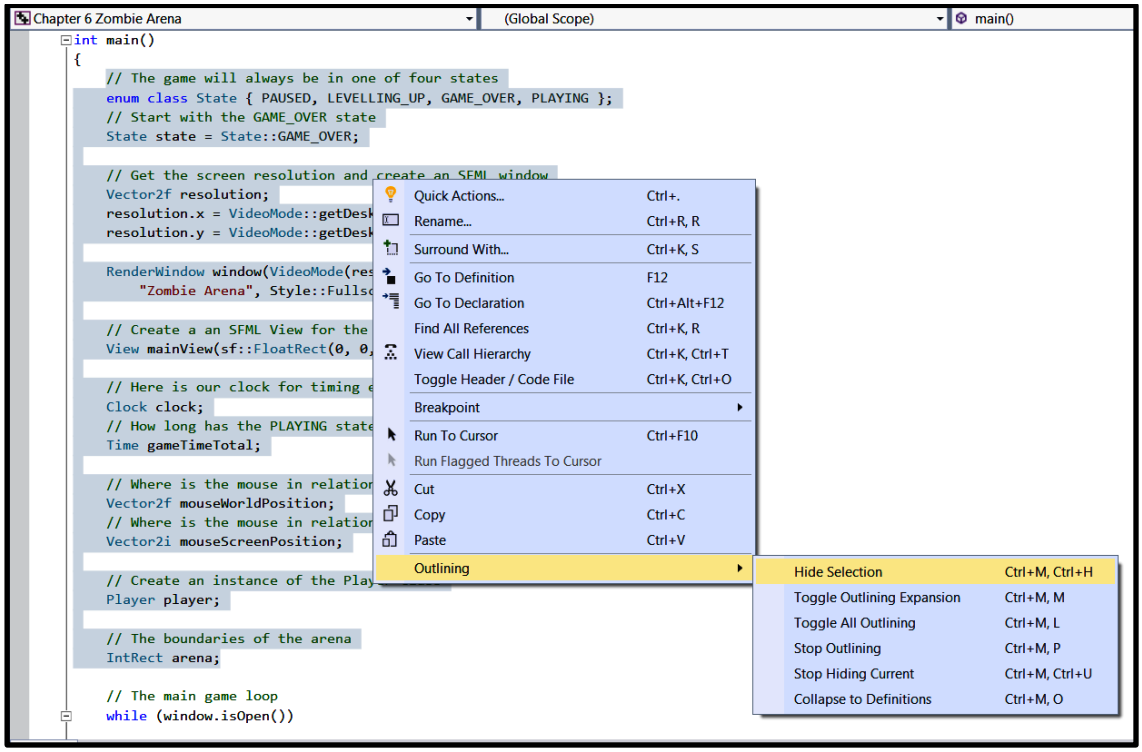
Chapter 7: Dynamic Collision Detection and Physics – Finishing the Pong Game

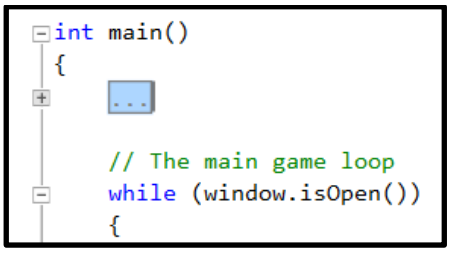
Chapter 8: SFML Views – Starting the Zombie Shooter Game

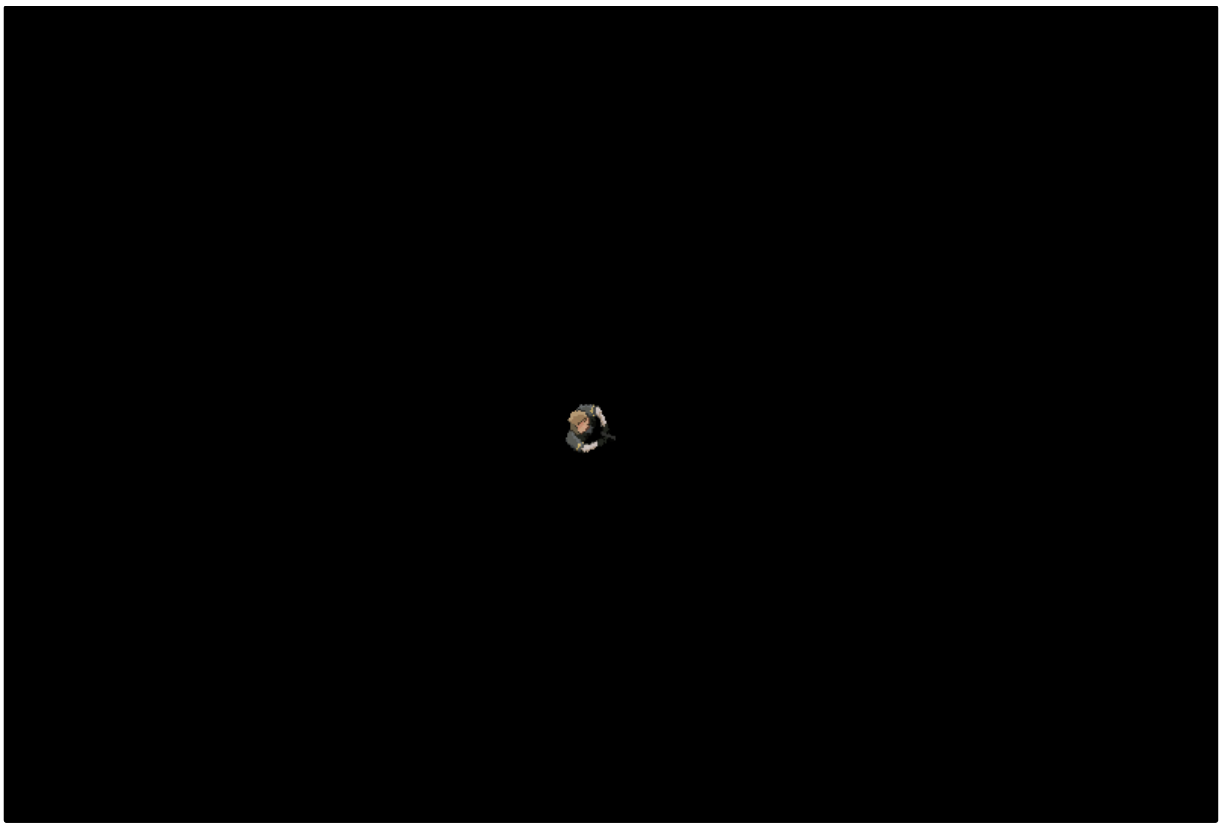




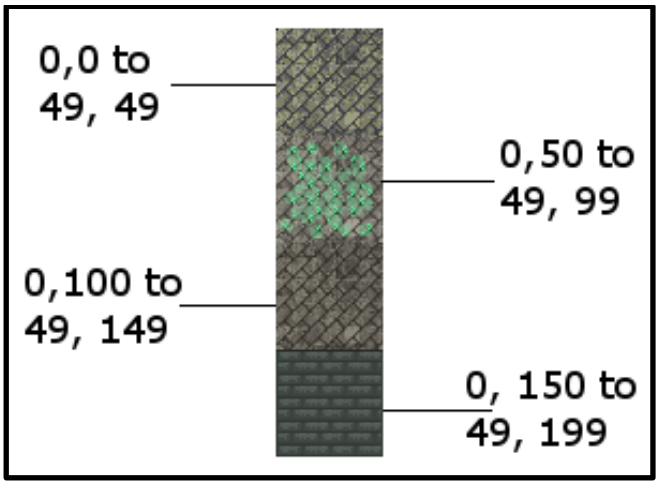

```
// The main game loop
while (window.isOpen())
{
```

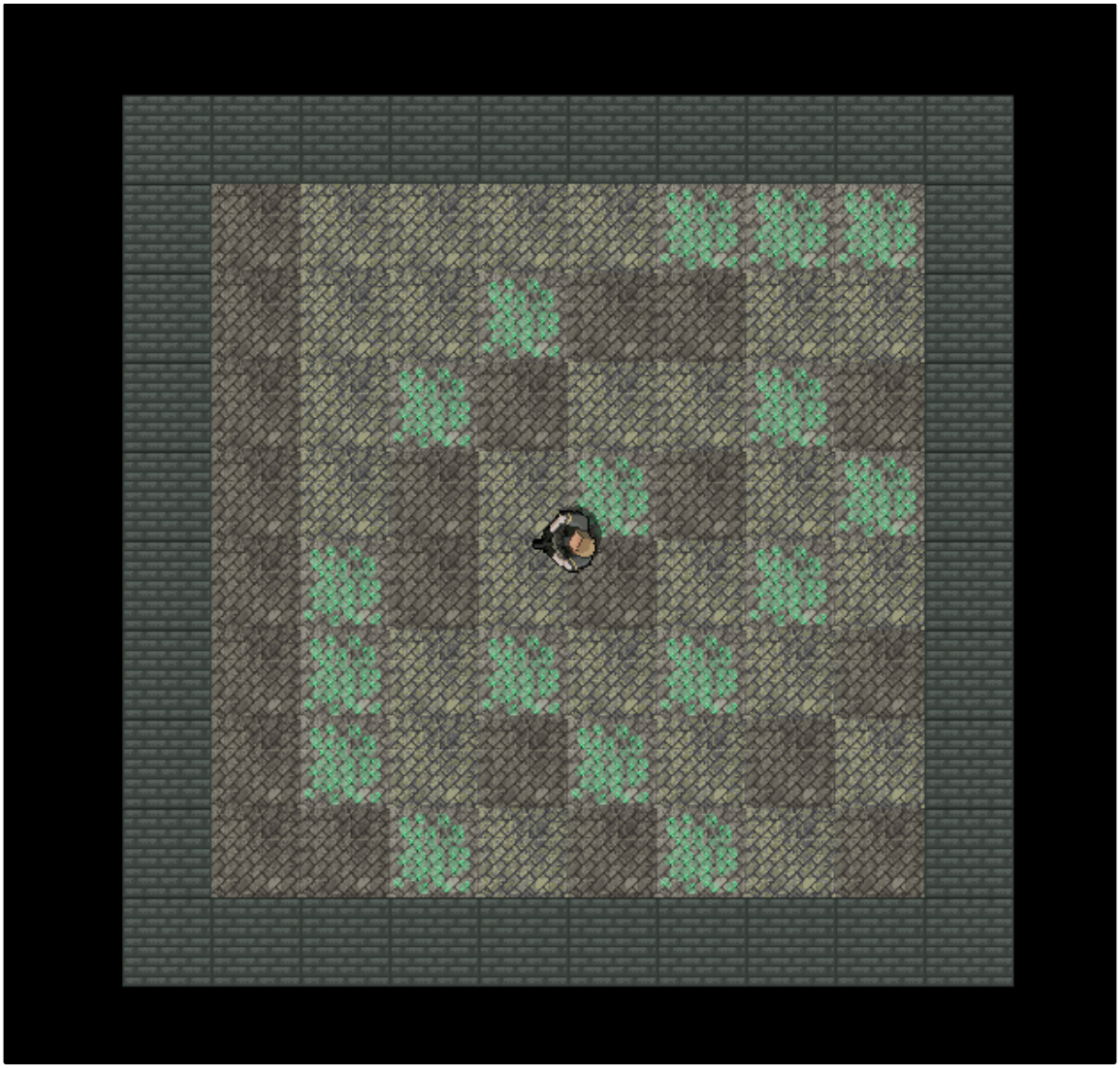

Chapter 6 Zombie Arena
(Global Scope)
main()
int main()
{
// The game will always be in one of four states
enum class State { PAUSED, LEVELLING_UP, GAME_OVER, PLAYING };
// Start with the GAME_OVER state
State state = State::GAME_OVER;
// Get the screen resolution and create an SFML window
Vector2f resolution;
resolution.x = VideoMode::getDesk
resolution.y = VideoMode::getDesk
RenderWindow window(VideoMode(res
"Zombie Arena", Style::Fullsc
// Create a an SFML View for the
View mainView(sf::FloatRect(0, 0,
// Here is our clock for timing e
Clock clock;
// How long has the PLAYING state
Time gameTimeTotal;
// Where is the mouse in relation
Vector2f mouseWorldPosition;
// Where is the mouse in relation
Vector2i mouseScreenPosition;
// Create an instance of the Play
Player player;
// The boundaries of the arena
IntRect arena;
// The main game loop
while (window.isOpen())
Quick Actions... Ctrl+.
Rename... Ctrl+R, R
Surround With... Ctrl+K, S
Go To Definition F12
Go To Declaration Ctrl+Alt+F12
Find All References Ctrl+K, R
View Call Hierarchy Ctrl+K, Ctrl+T
Toggle Header / Code File Ctrl+K, Ctrl+O
Breakpoint
Run To Cursor Ctrl+F10
Run Flagged Threads To Cursor
Cut Ctrl+X
Copy Ctrl+C
Paste Ctrl+V
Outlining
Hide Selection Ctrl+M, Ctrl+H
Toggle Outlining Expansion Ctrl+M, M
Toggle All Outlining Ctrl+M, L
Stop Outlining Ctrl+M, P
Stop Hiding Current Ctrl+M, Ctrl+U
Collapse to Definitions Ctrl+M, O


int main()
{
...
// The main game loop
while (window.isOpen())
{

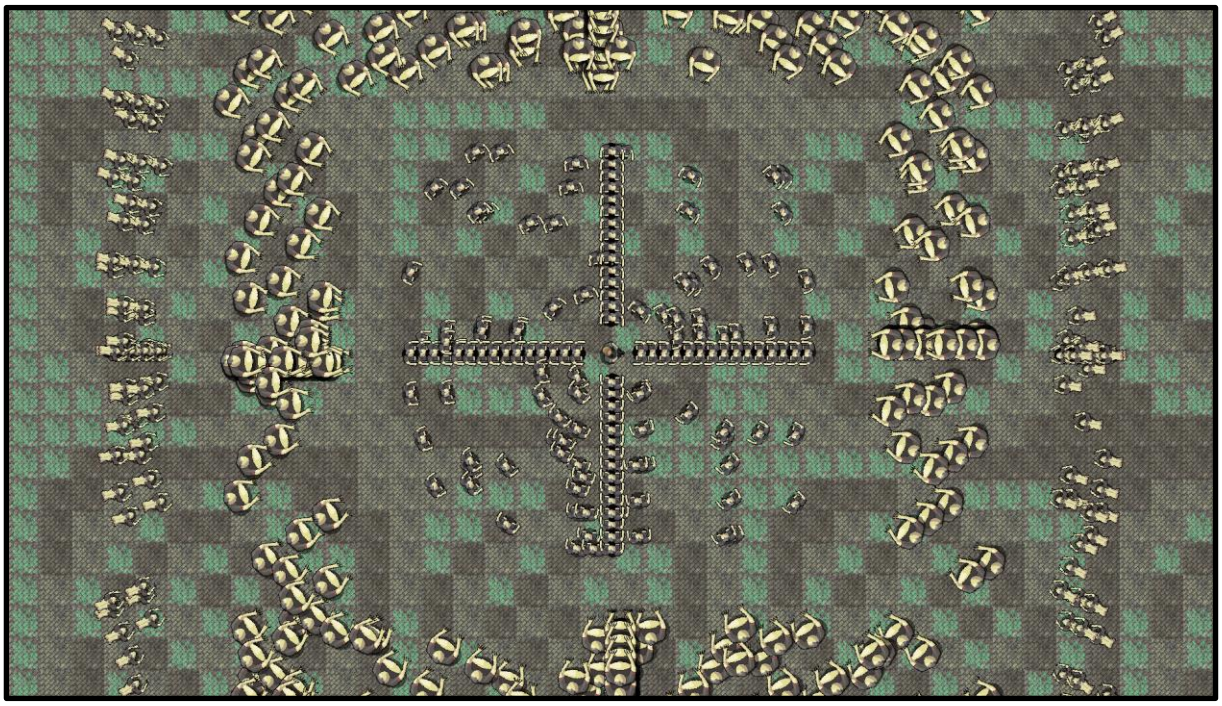


Chapter 9: C++ References, Sprite Sheets, and Vertex Arrays

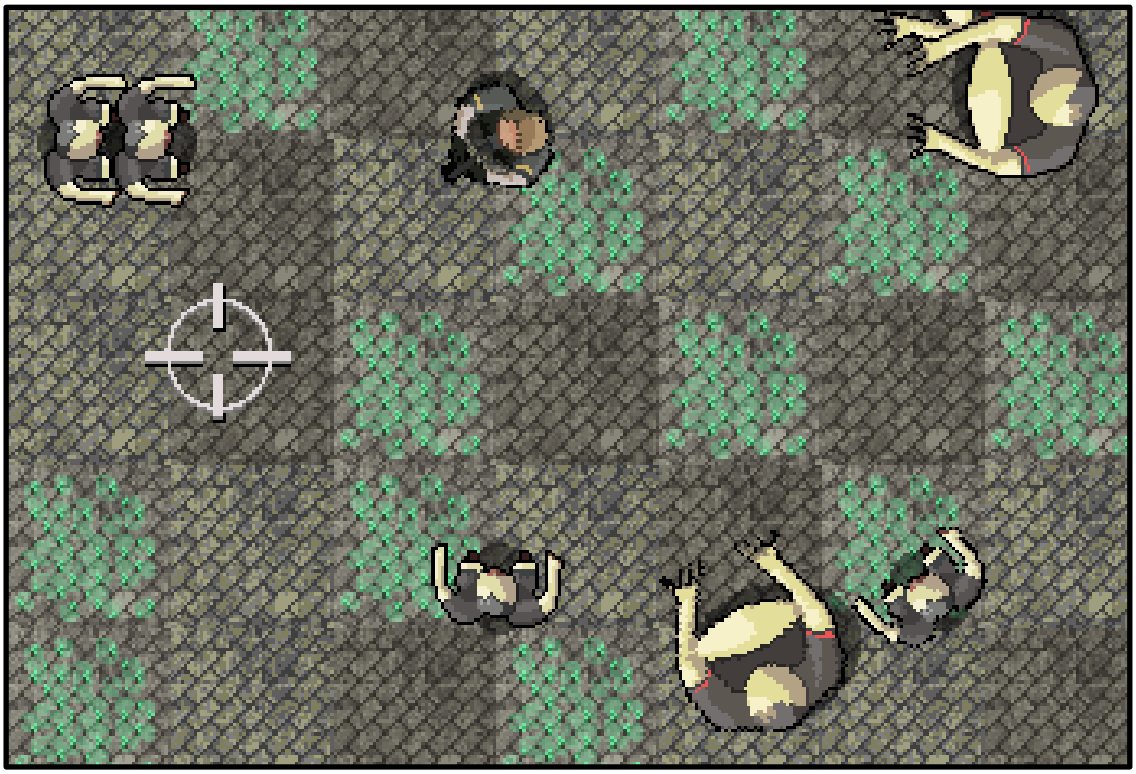


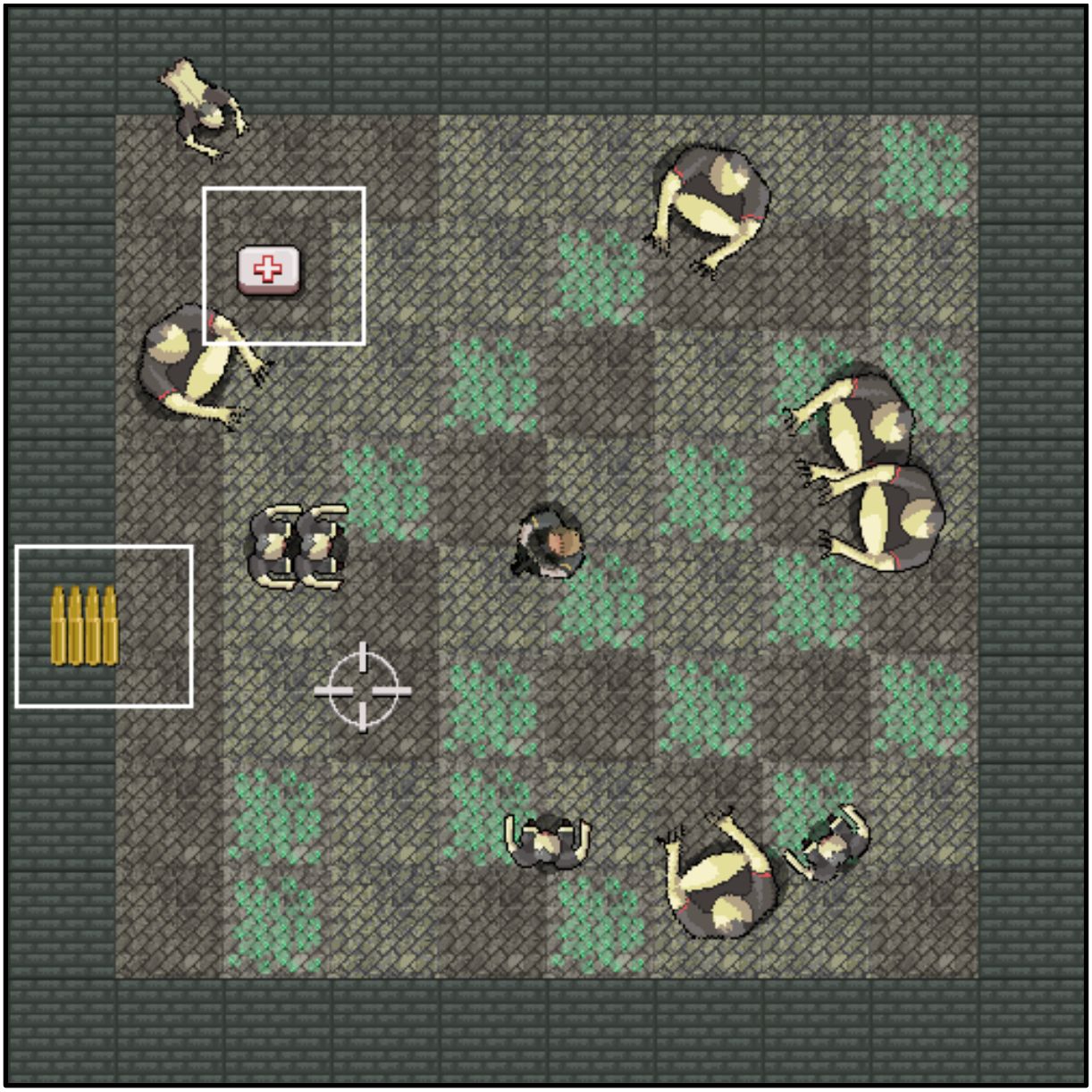


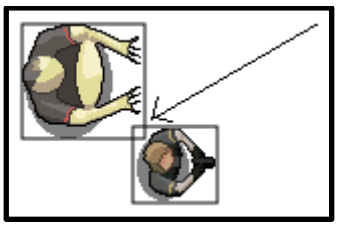
Chapter 10: Pointers, the Standard Template Library, and Texture Management



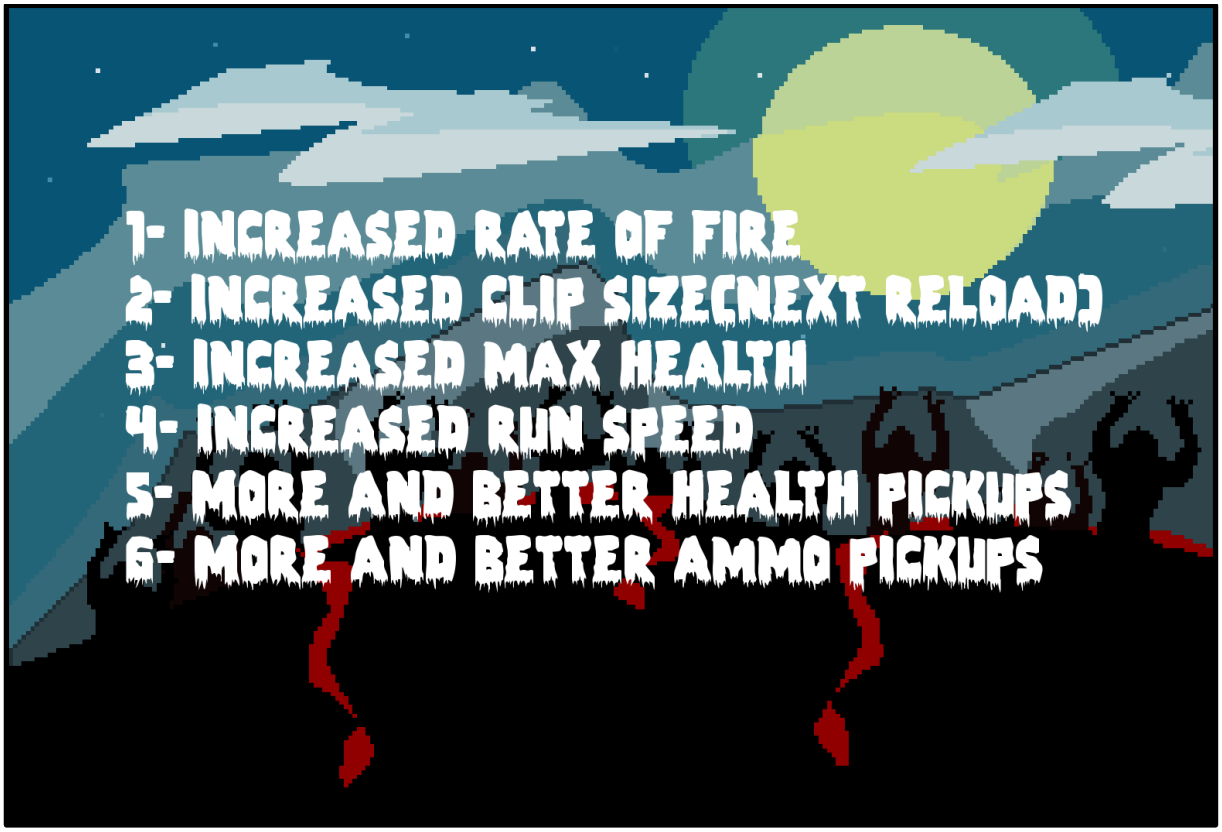
Chapter 11: Collision Detection, Pickups, and Bullets



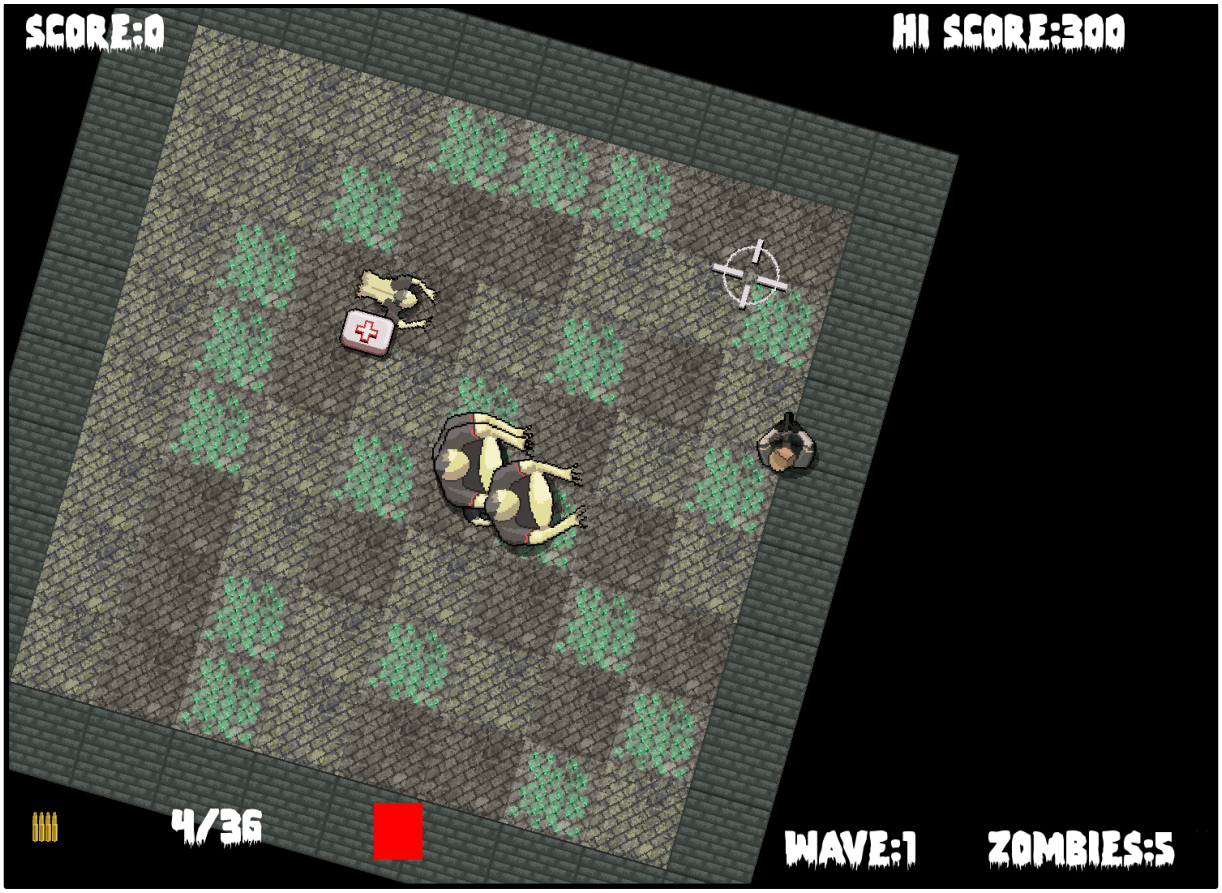




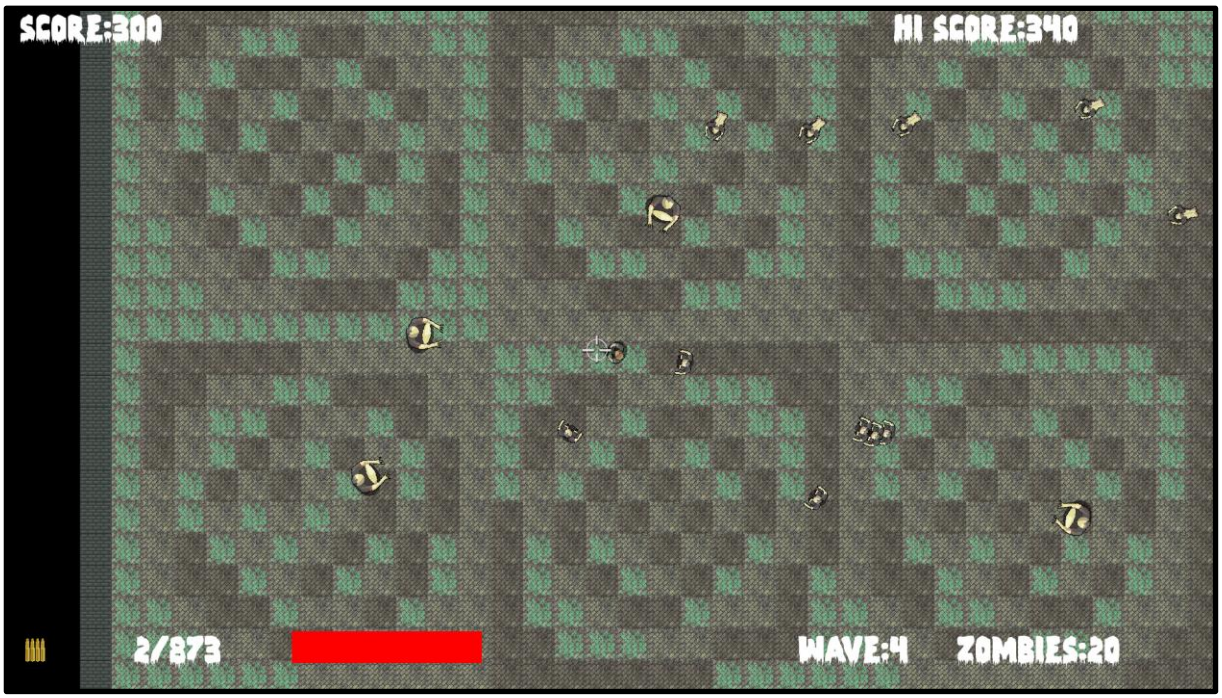
Chapter 12: Layering Views and Implementing the HUD


1- INCREASED RATE OF FIRE
2- INCREASED CLIP SIZE(NEXT RELOAD)
3- INCREASED MAX HEALTH
4- INCREASED RUN SPEED
5- MORE AND BETTER HEALTH PICKUPS
6- MORE AND BETTER AMMO PICKUPS

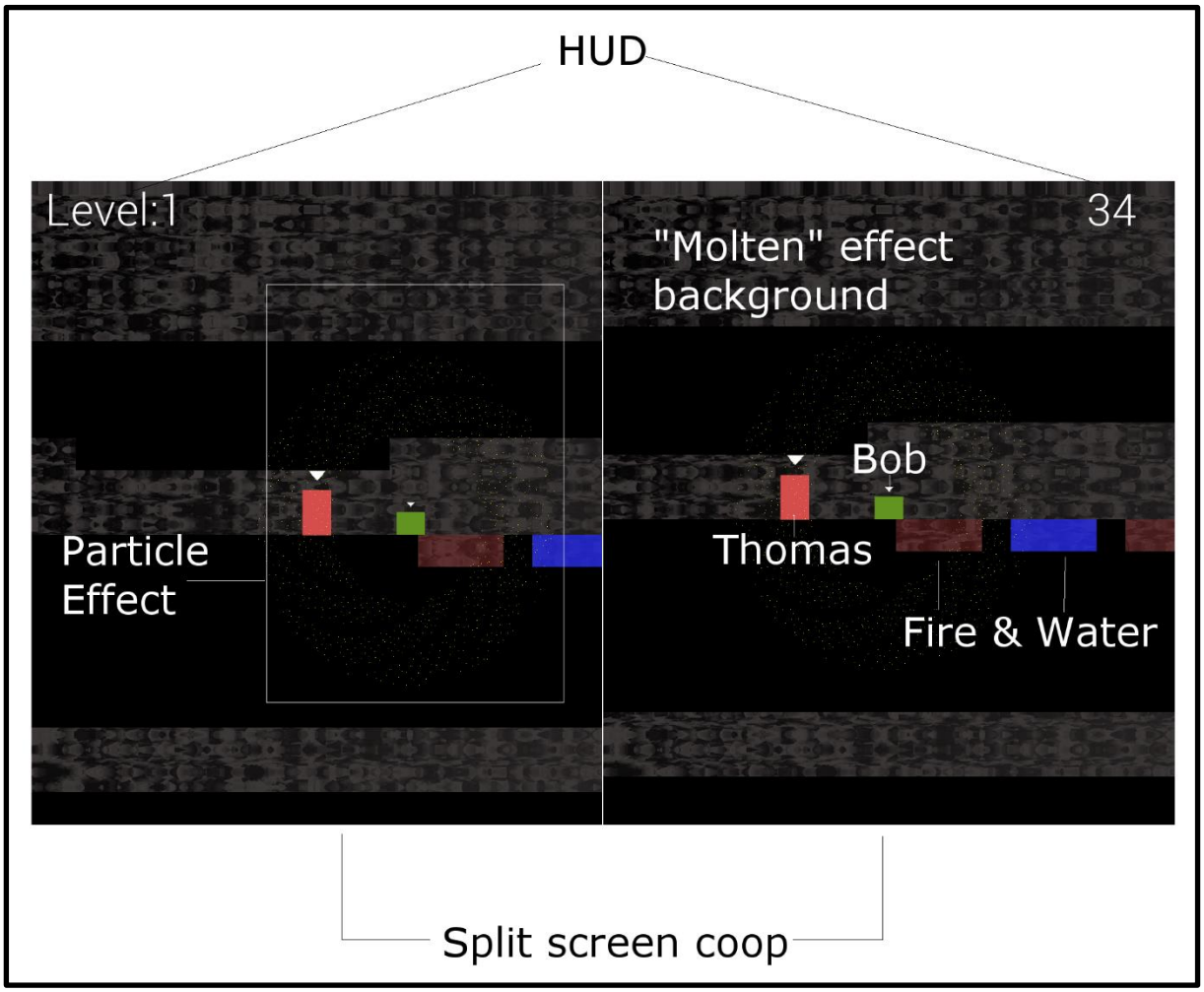

HI SCORE:0
PRESS ENTER
TO CONTINUE
WAVE: 0
ZOMBIES: 100

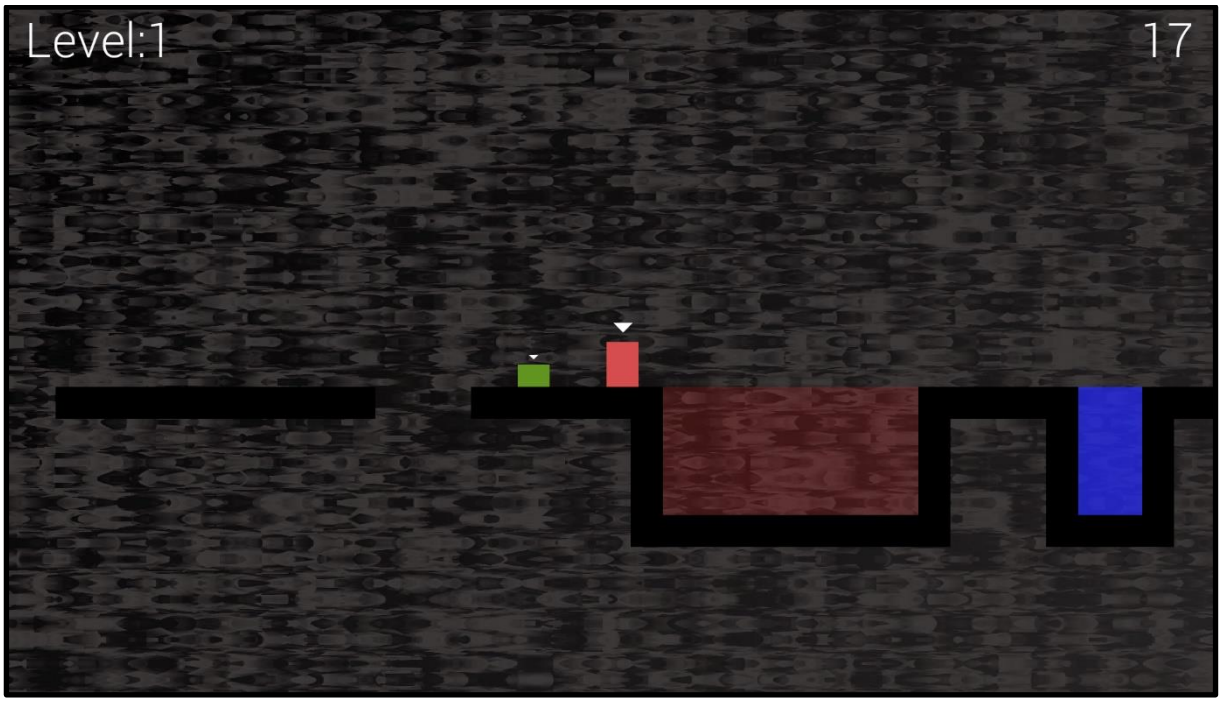


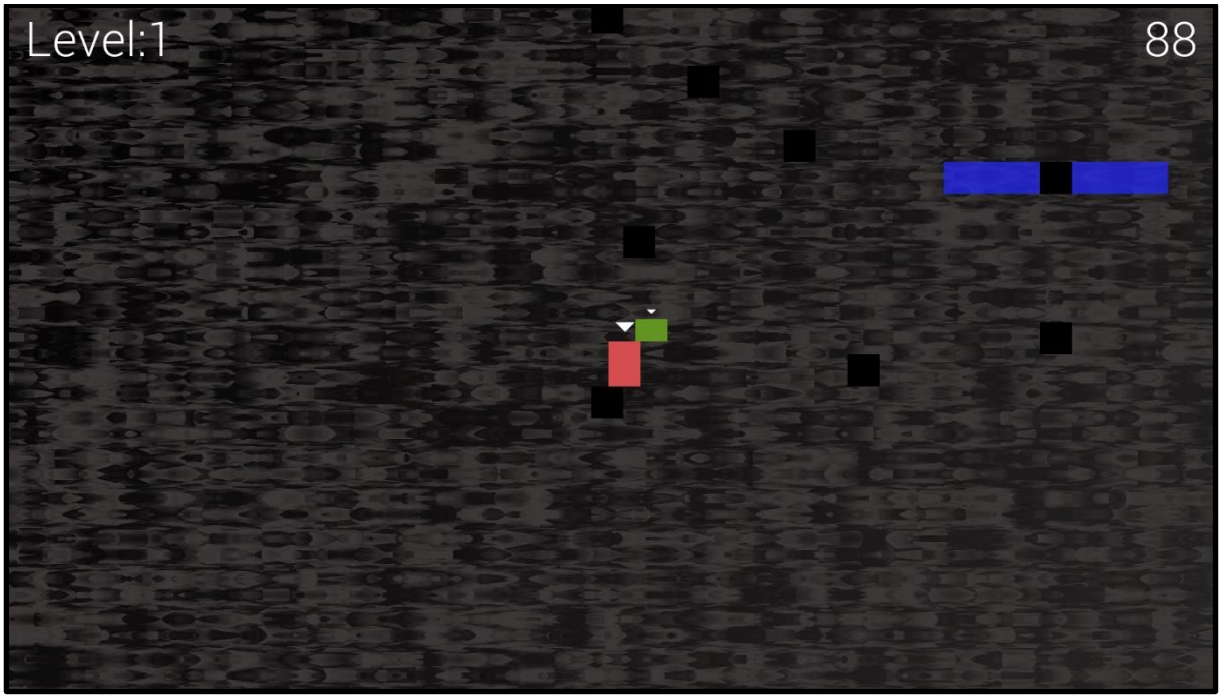
Chapter 13: Sound Effects, File I/O, and Finishing the Game

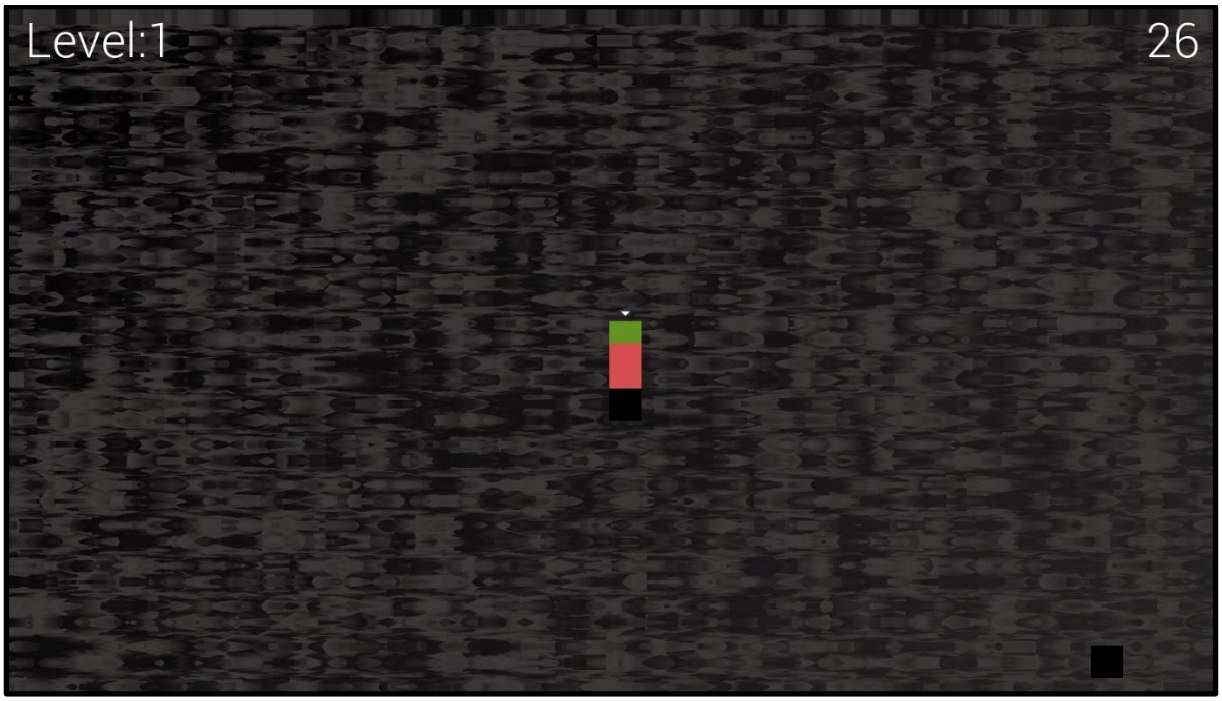


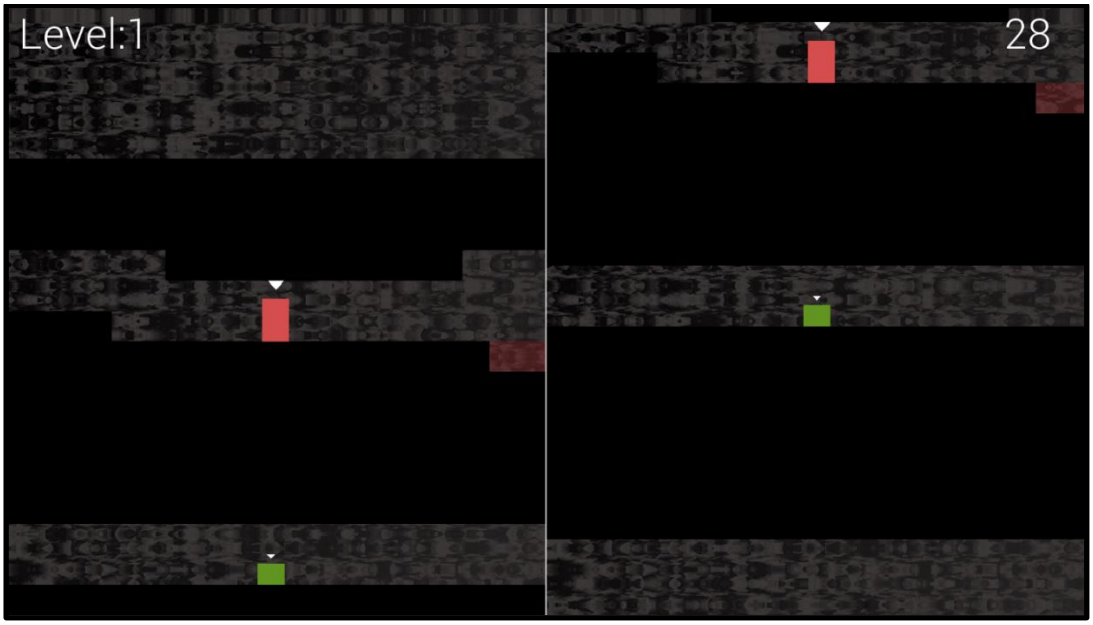
Chapter 14: Abstraction and Code Management – Making Better Use of OOP

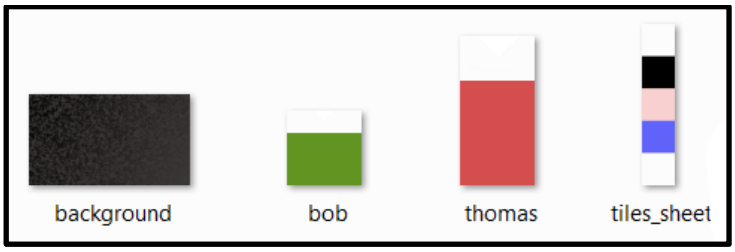


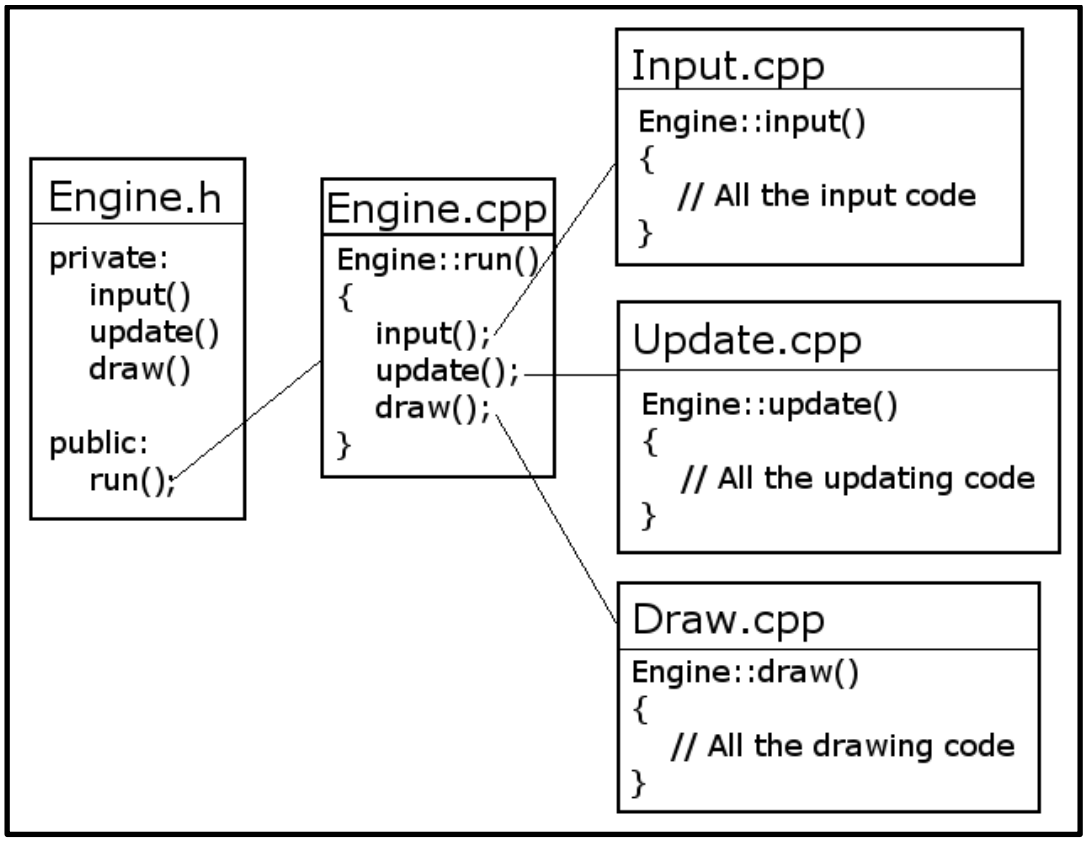



Level:1
88

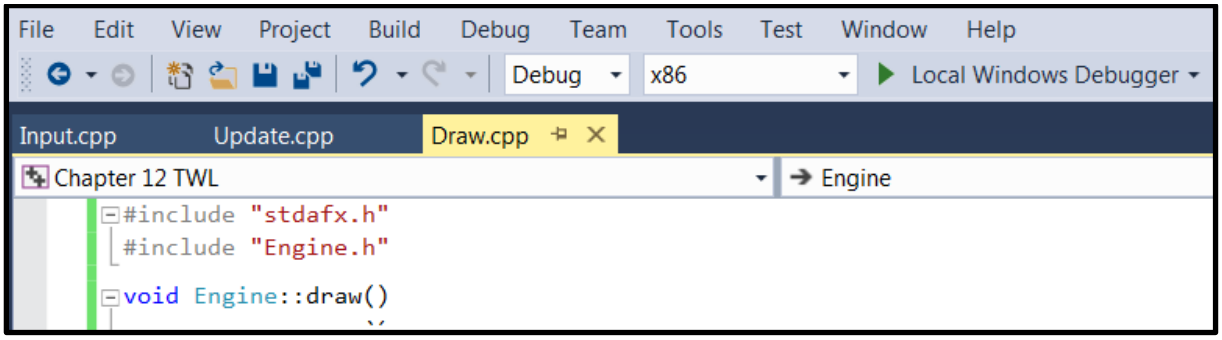

Level:1
26

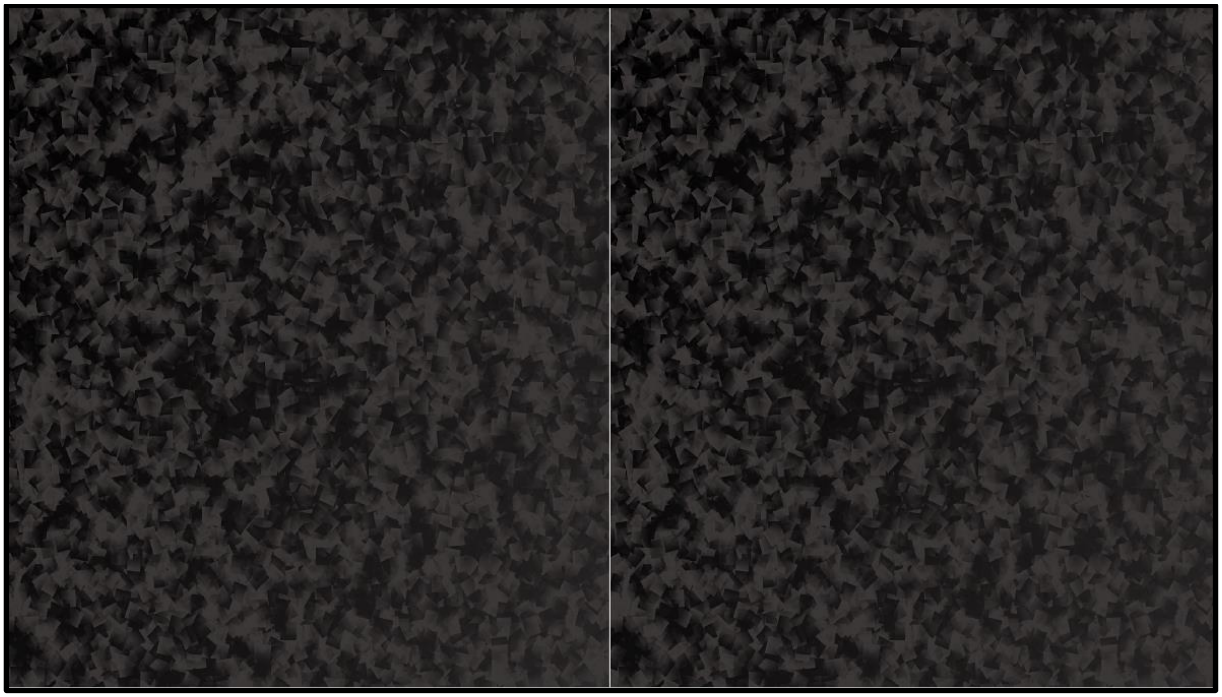

Level:1
28


background
bob
thomas
tiles_sheet

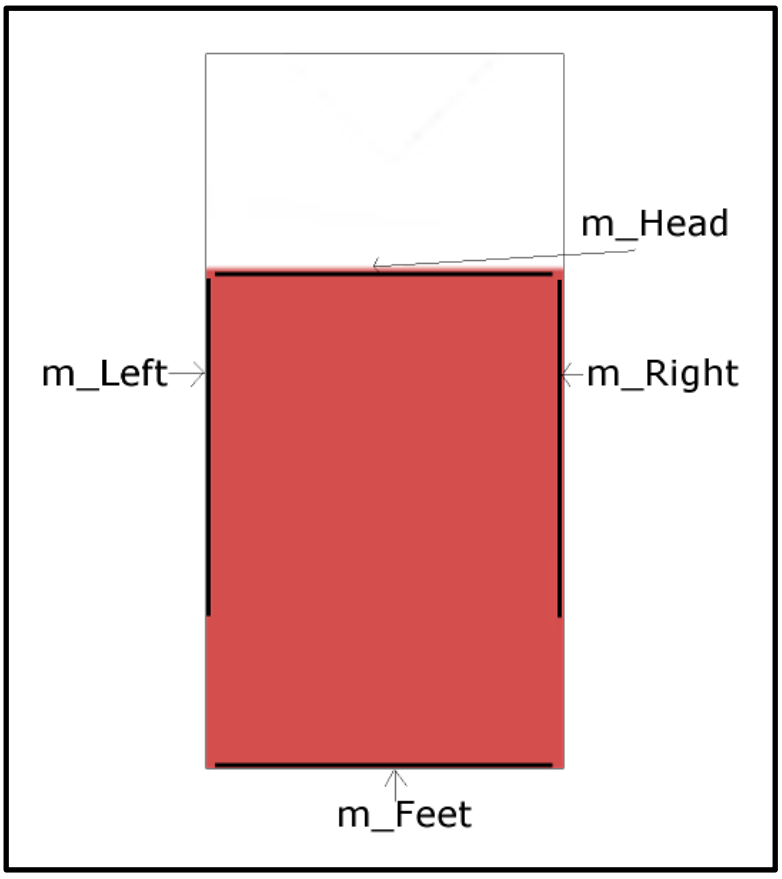

Engine.h
private:
input()
update()
draw()
public:
run();
Engine.cpp
Engine::run()
{
input();
update();
draw();
}
Input.cpp
Engine::input()
{
// All the input code
}
Update.cpp
Engine::update()
{
// All the updating code
}
Draw.cpp
Engine::draw()
{
// All the drawing code
}

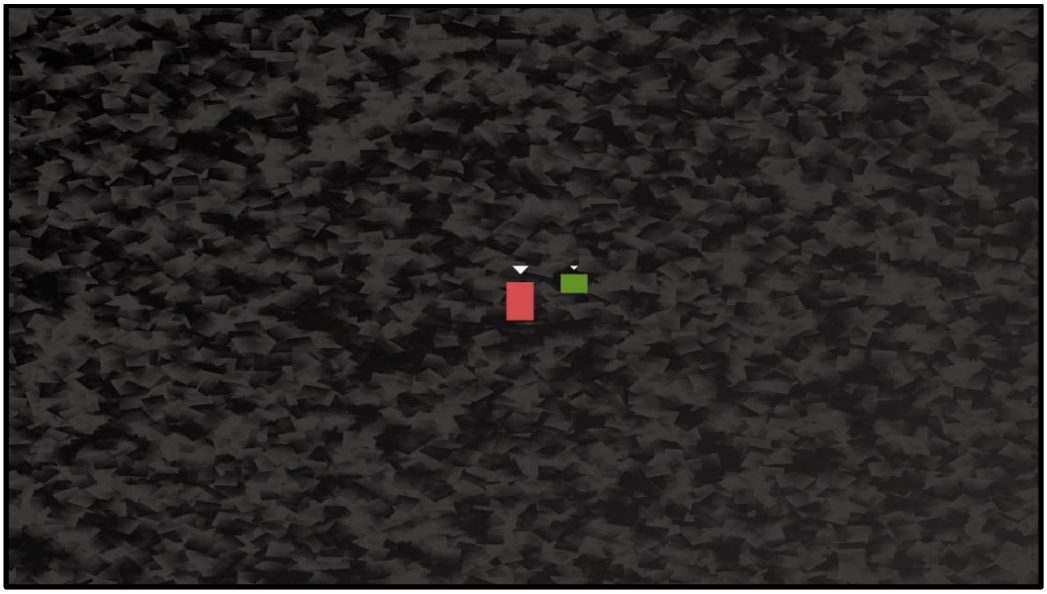
m_LeftView on top of m_BGLeftView	m_RightView on top of m_BGRightView

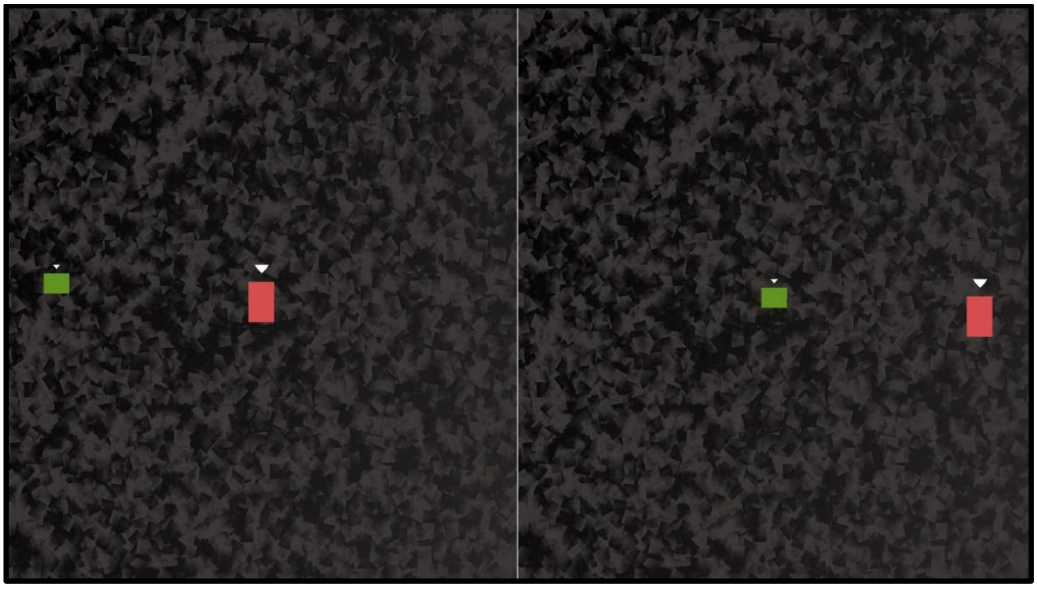


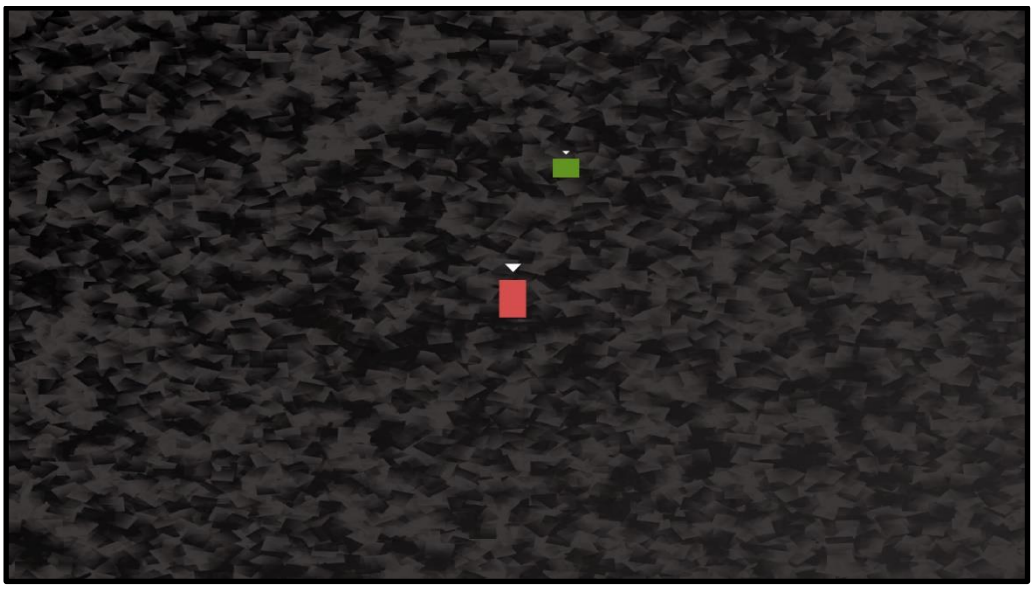


Chapter 15: Advanced OOP – Inheritance and Polymorphism

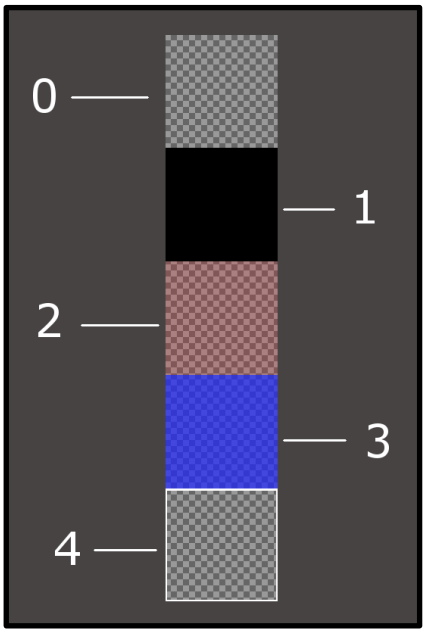


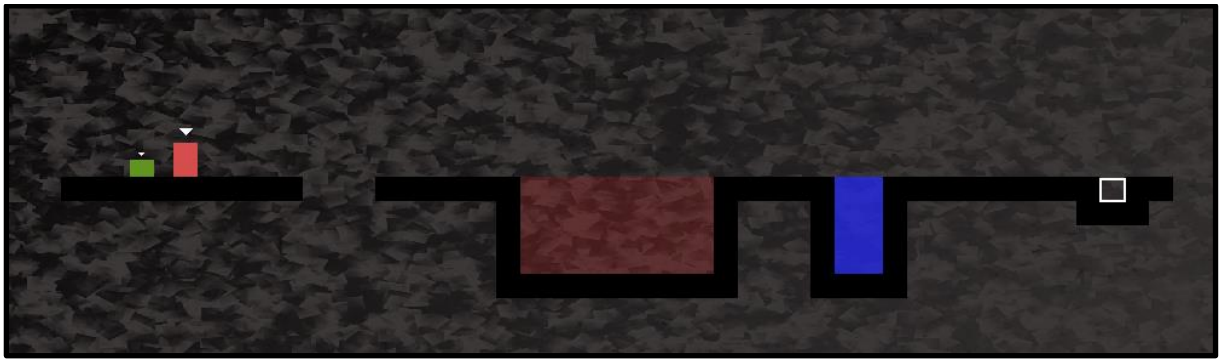


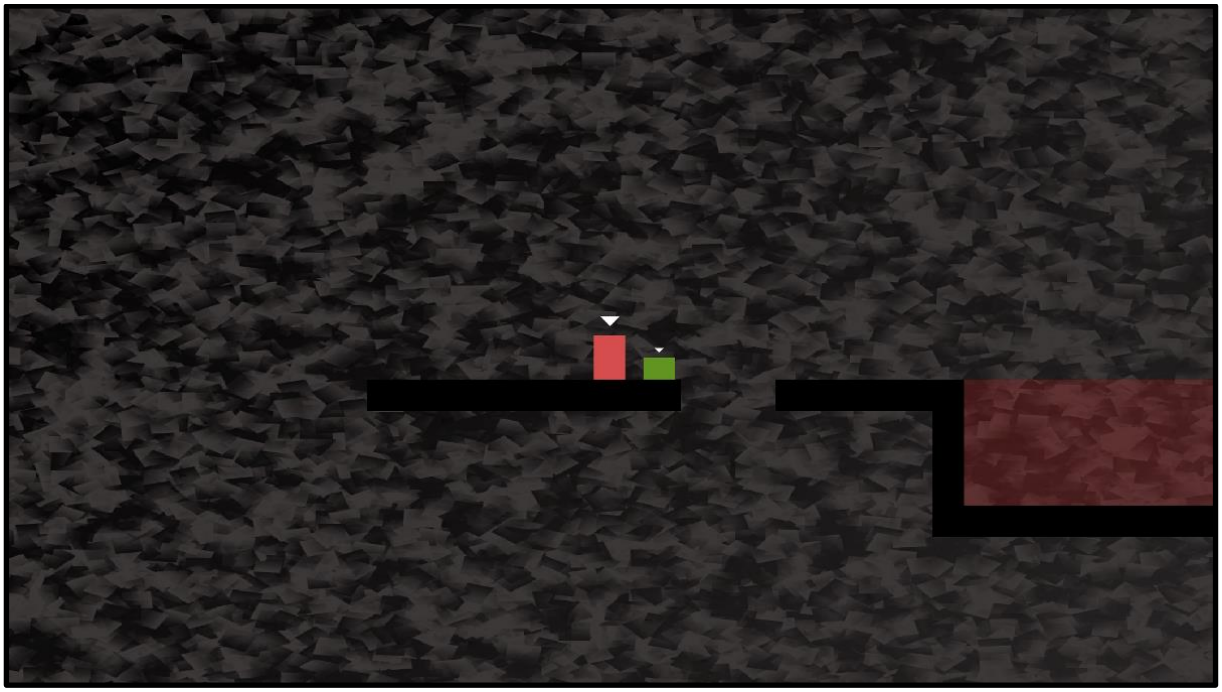


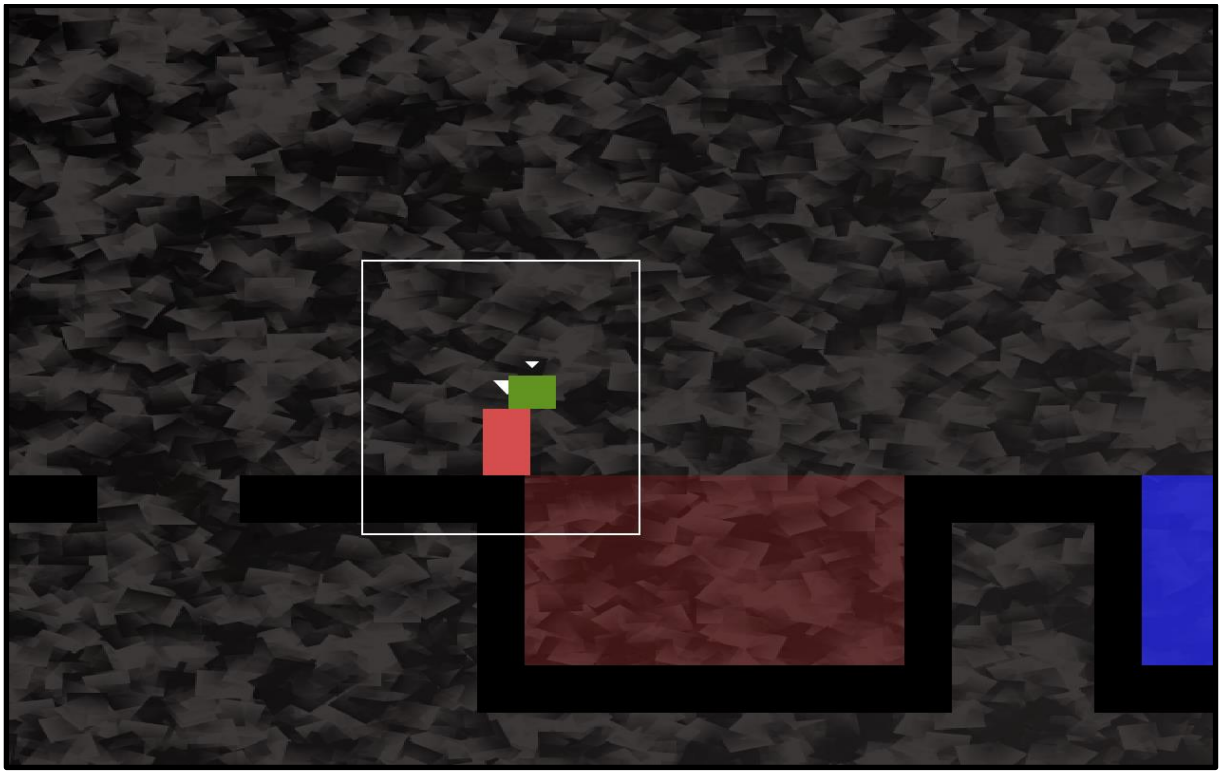


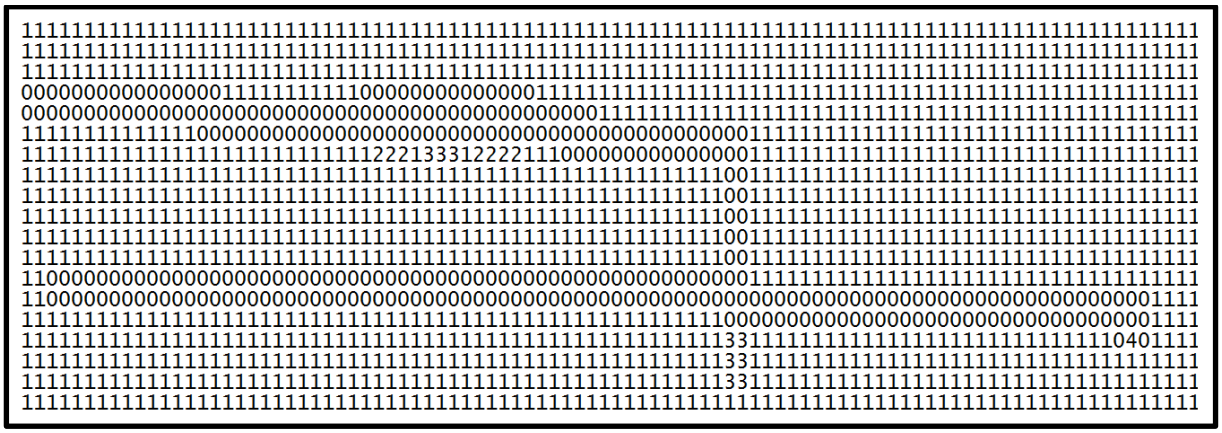
Chapter 16: Building Playable Levels and Collision Detection

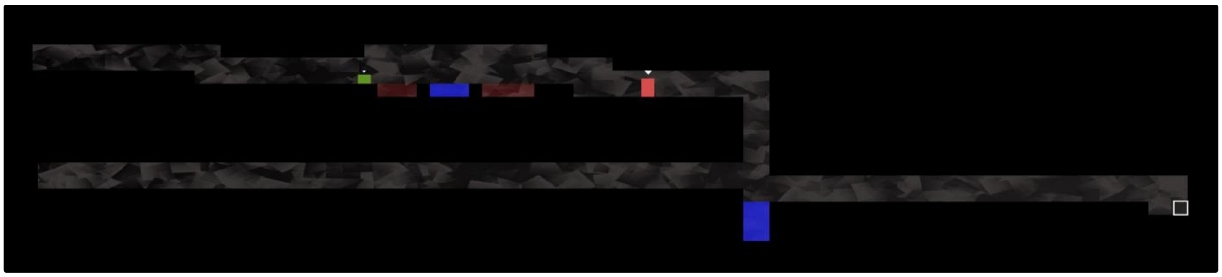


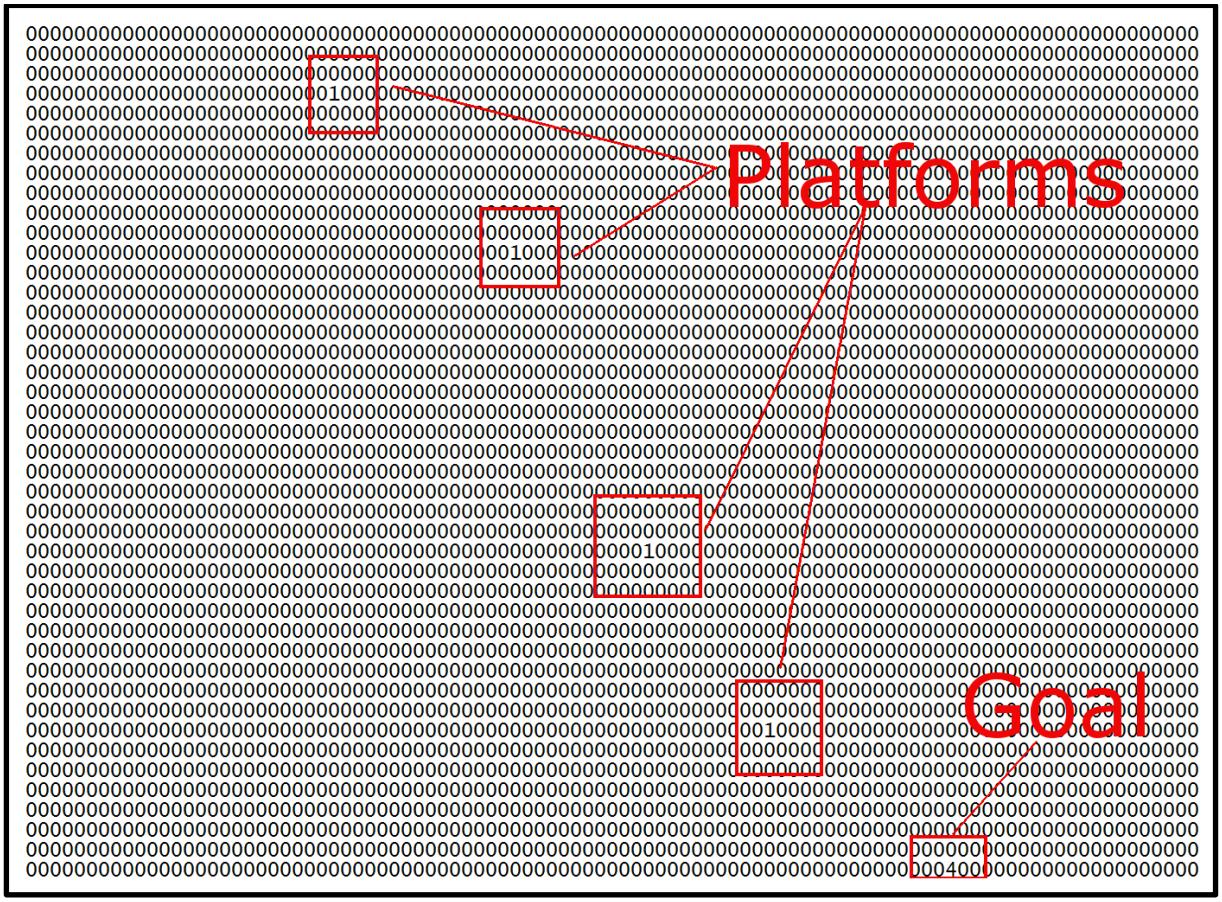


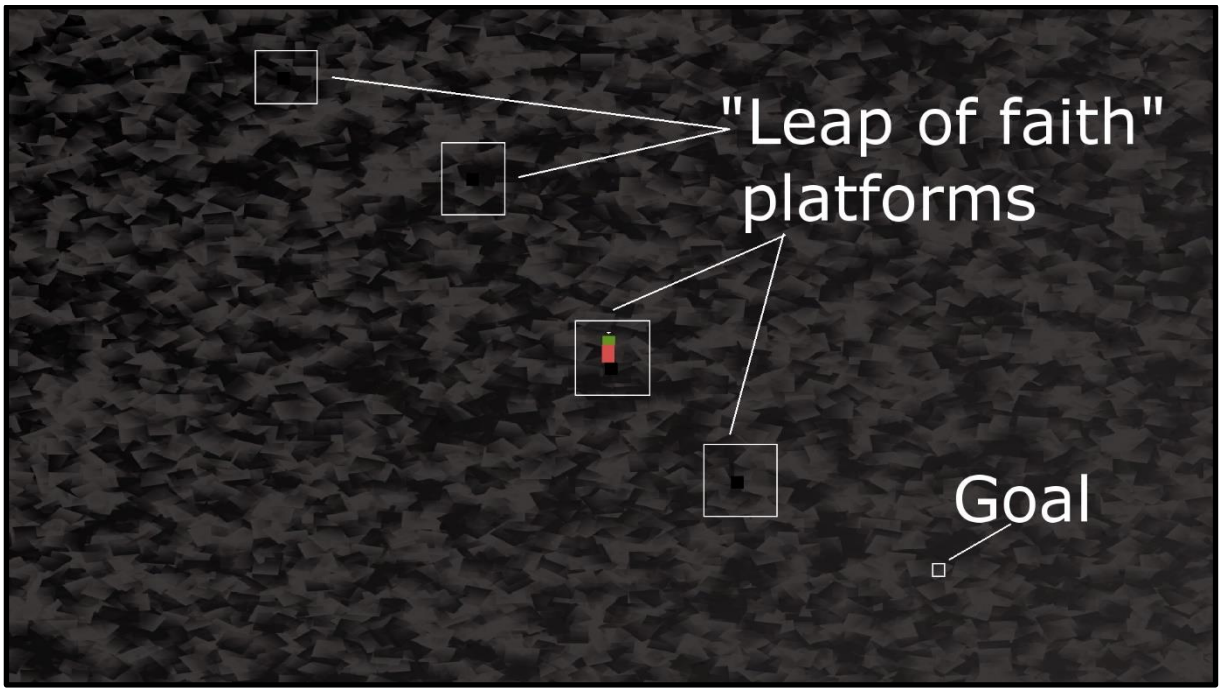


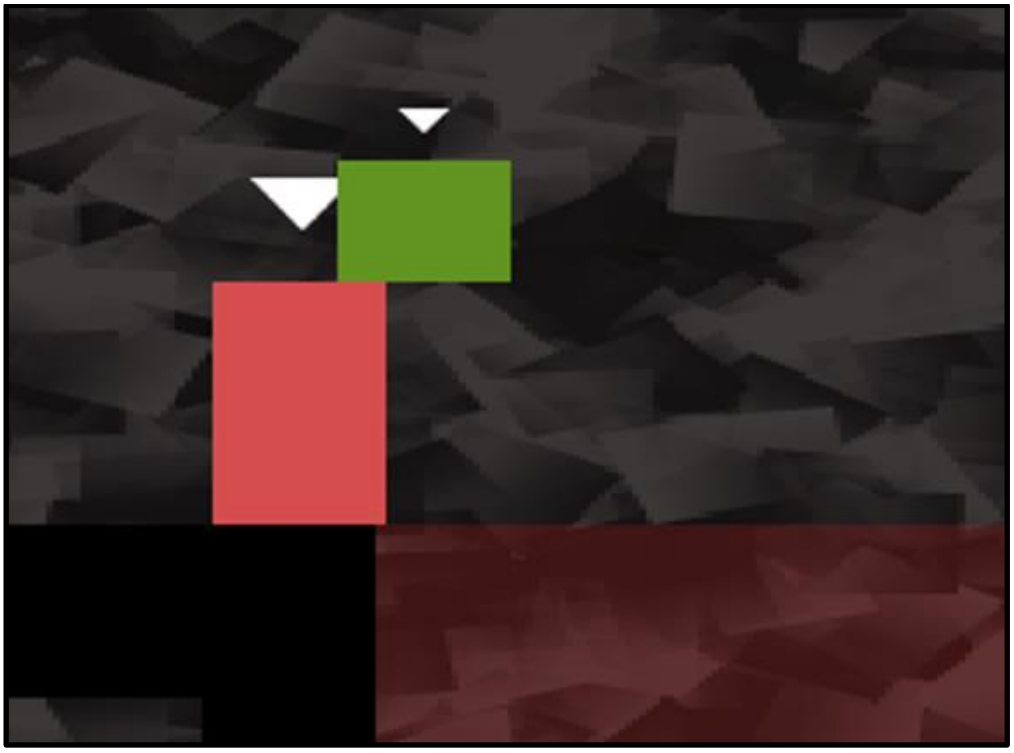




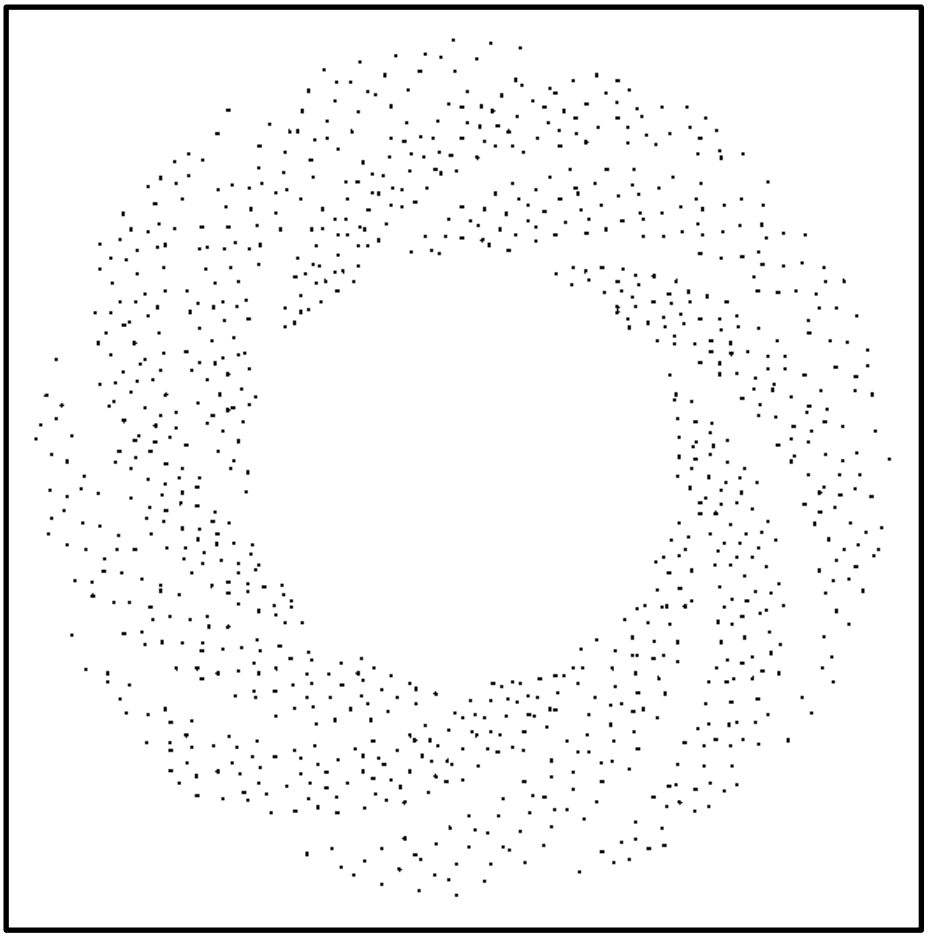


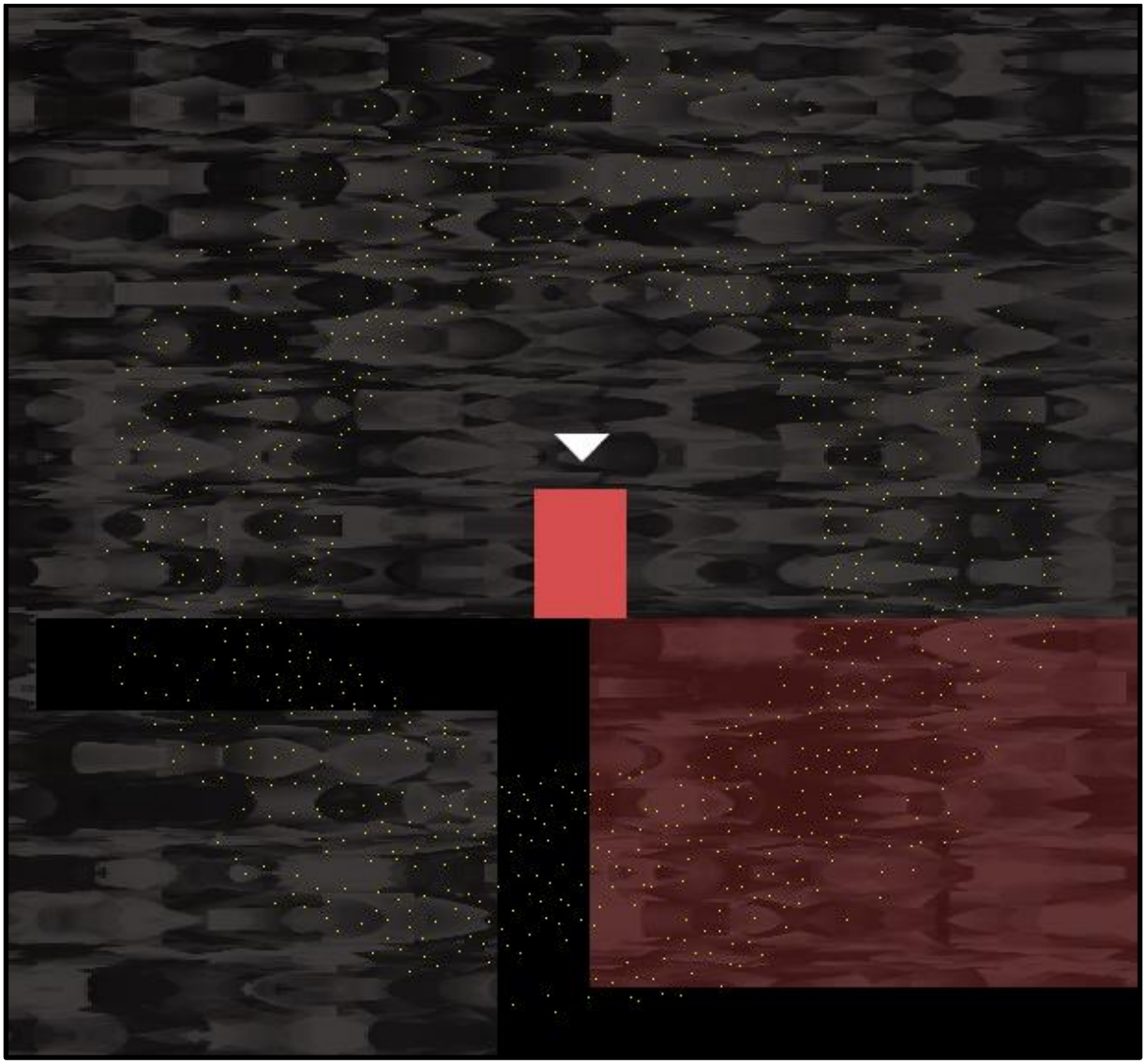

Platforms
Goal

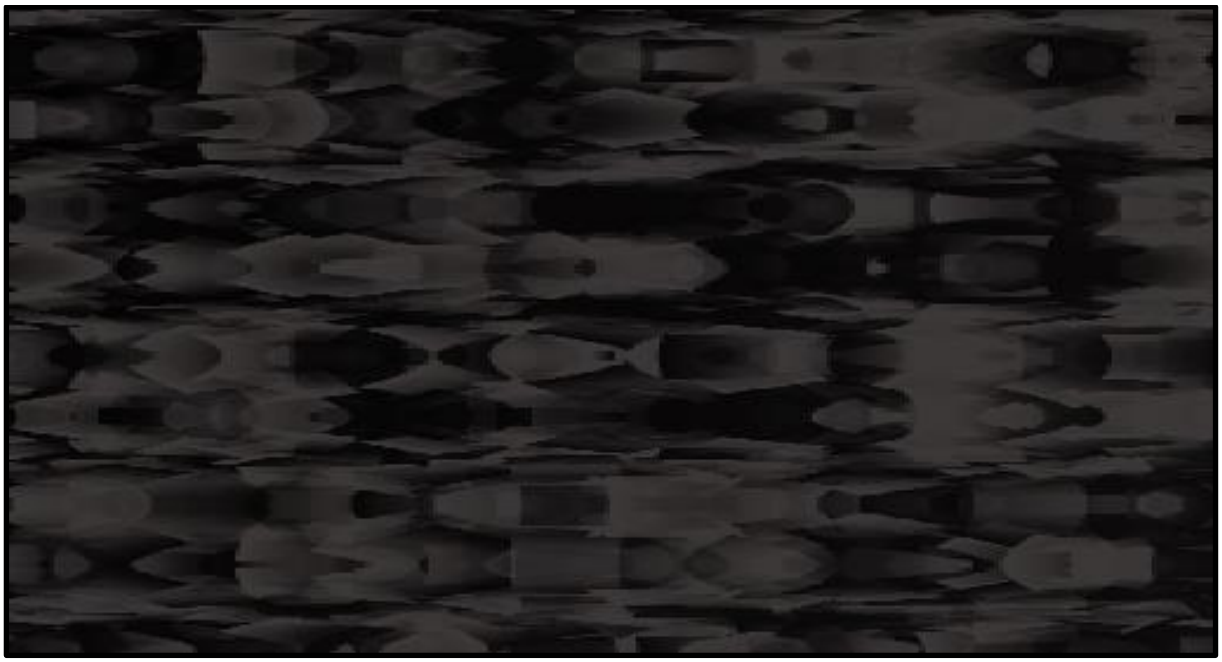

"Leap of faith"
platforms
Goal



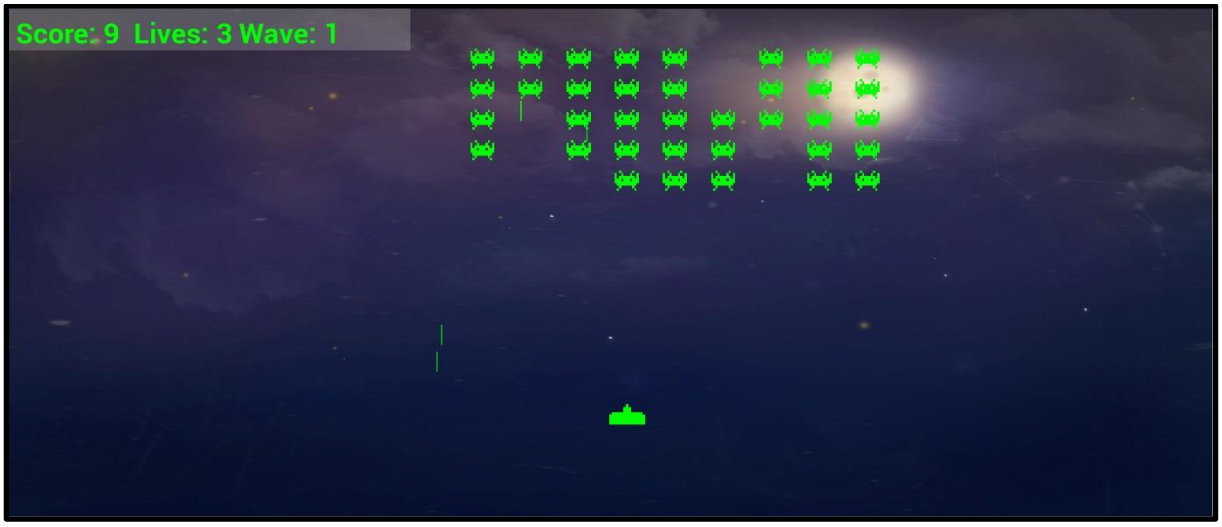
Chapter 18: Particle Systems and Shaders

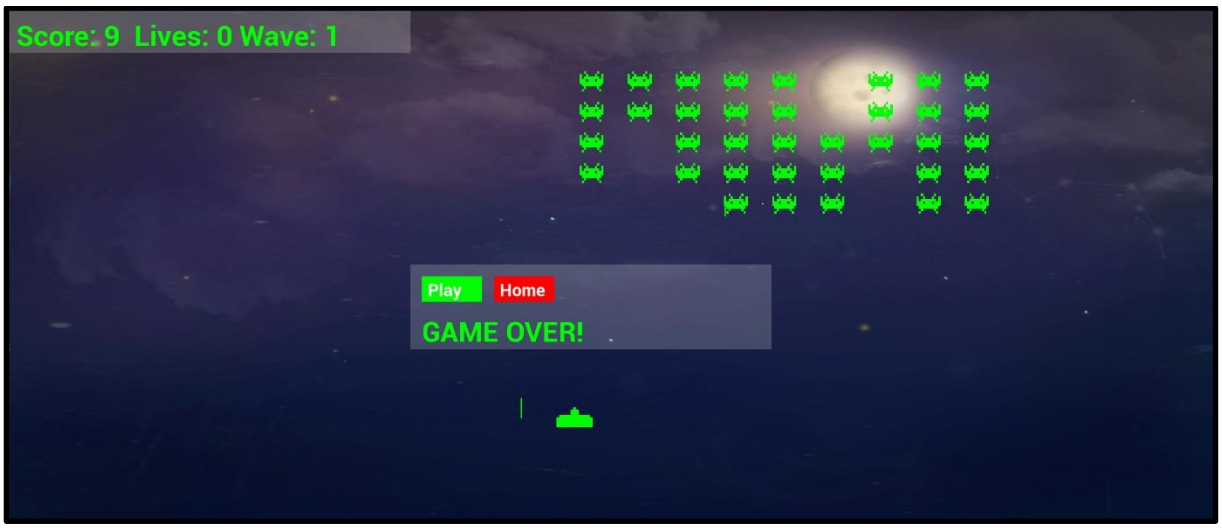


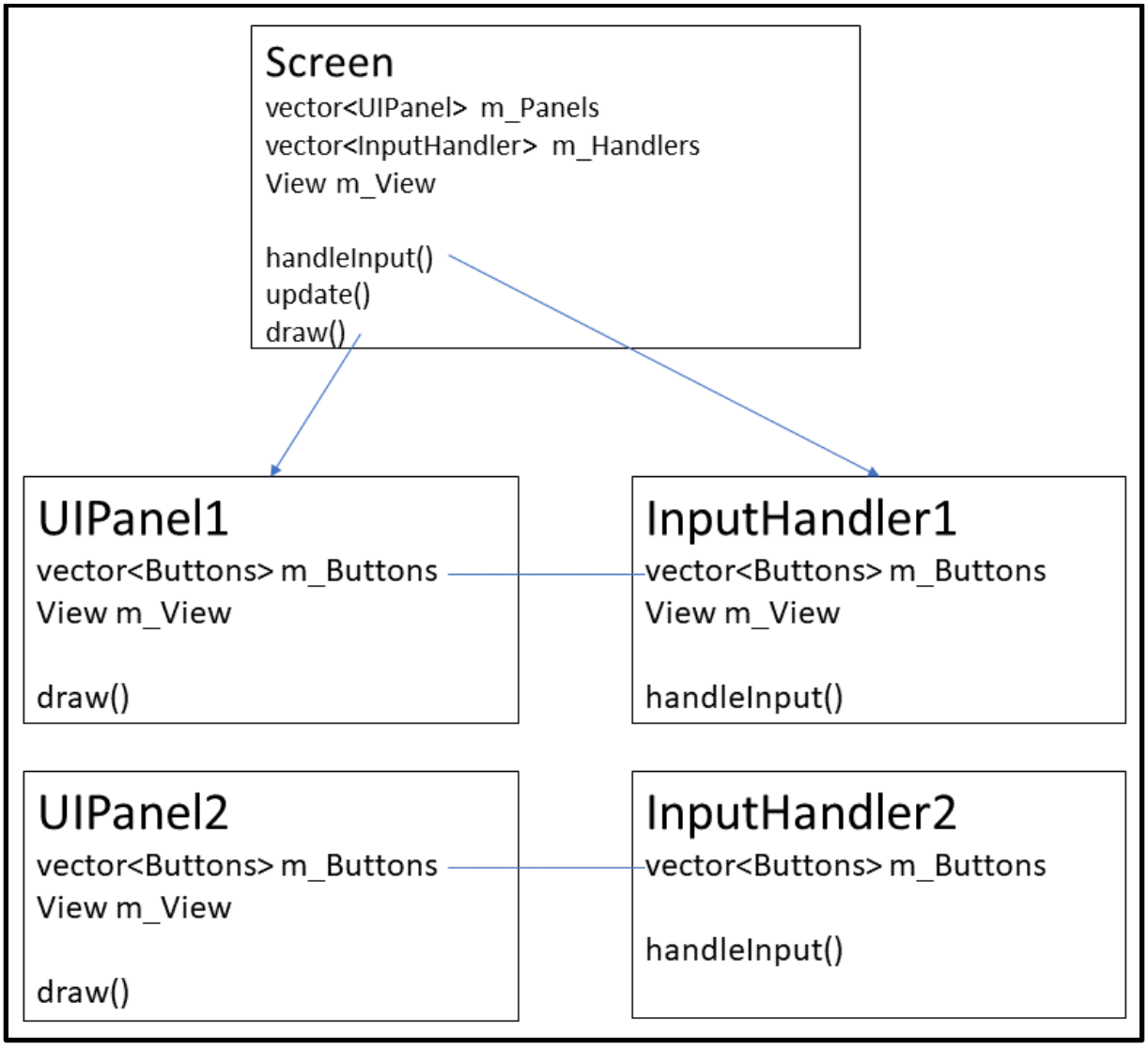


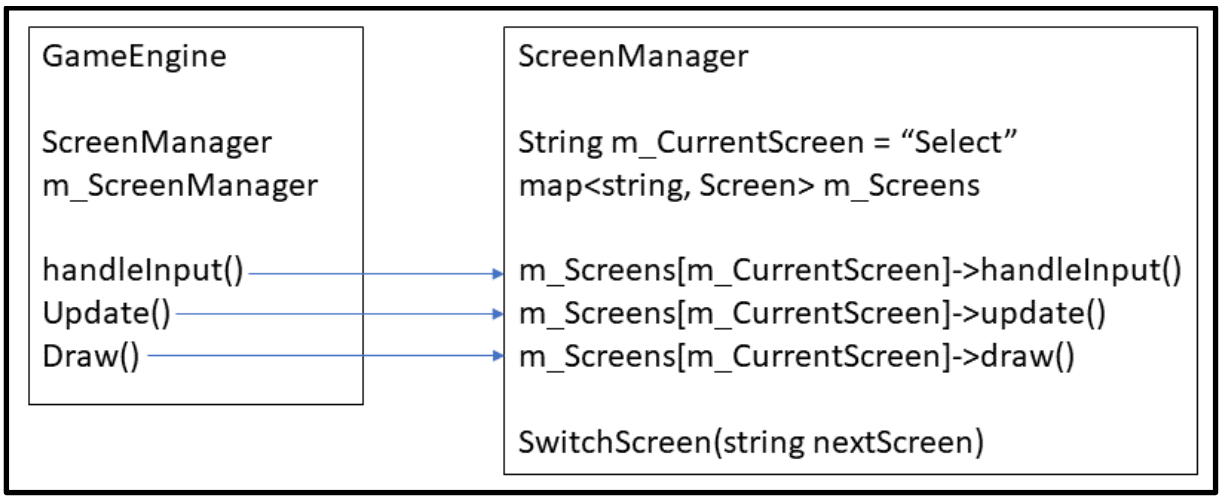


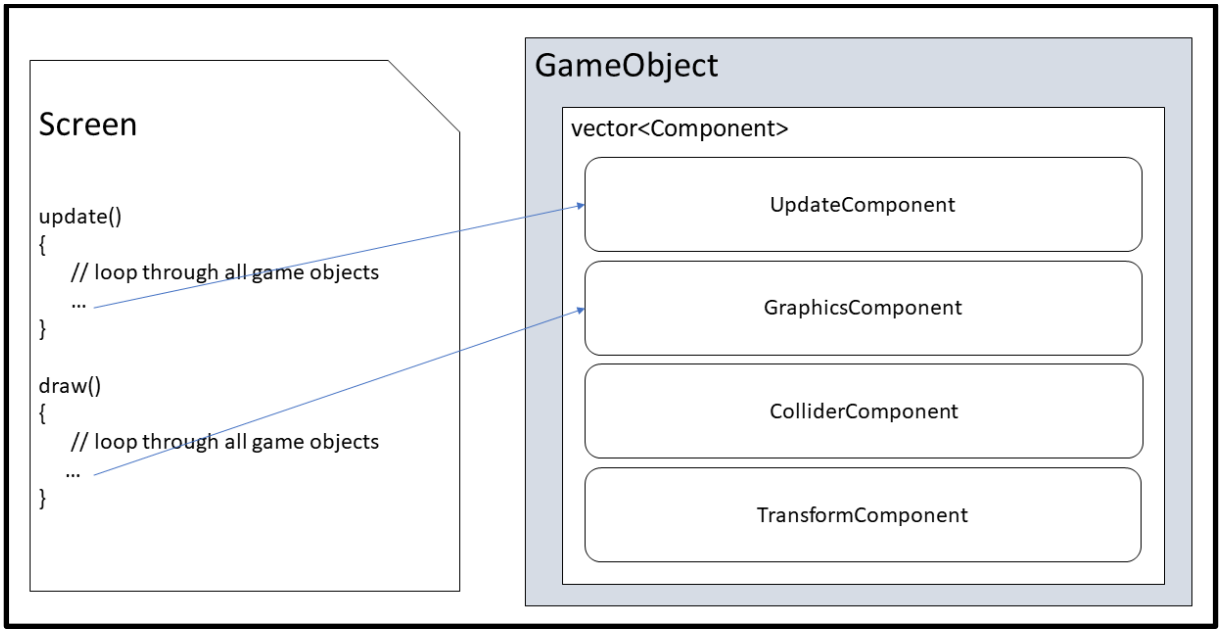
Chapter 19: Game Programming Design Patterns – Starting the Space Invaders ++ Game

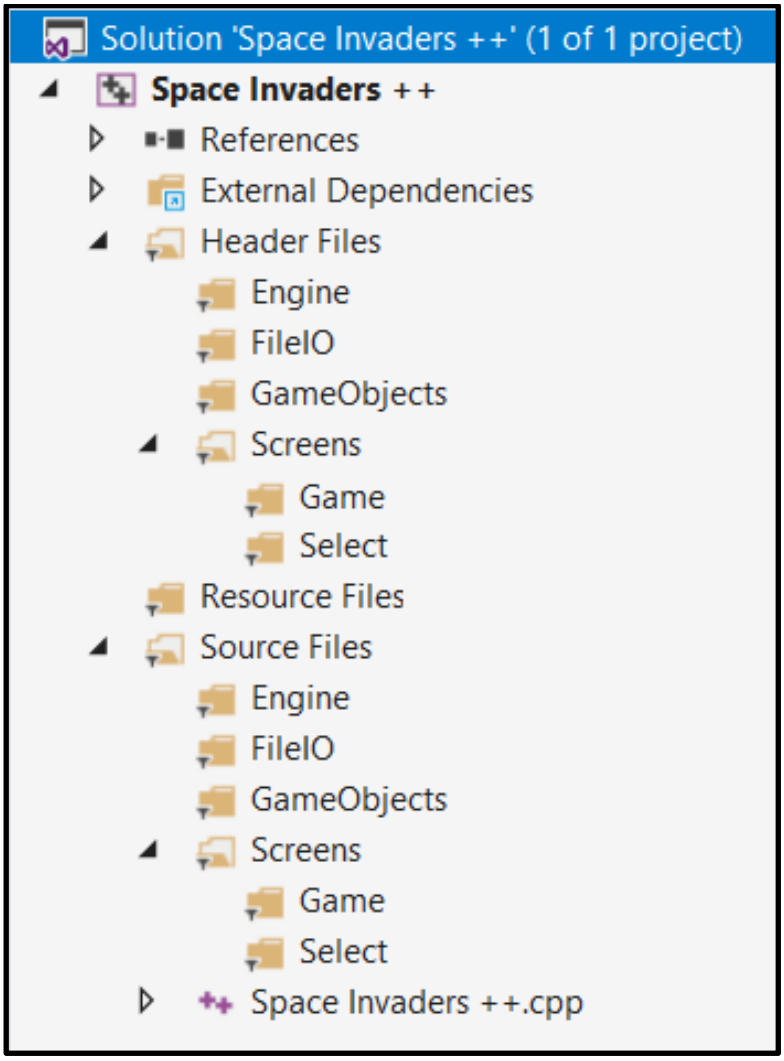





Screen
vector<UIPanel> m_Panels
vector<InputHandler> m_Handlers
View m_View
handleInput()
update()
draw()
UIPanel1
vector<Buttons> m_Buttons
View m_View
draw()
InputHandler1
vector<Buttons> m_Buttons
View m_View
handleInput()
UIPanel2
vector<Buttons> m_Buttons
View m_View
draw()
InputHandler2
vector<Buttons> m_Buttons
handleInput()

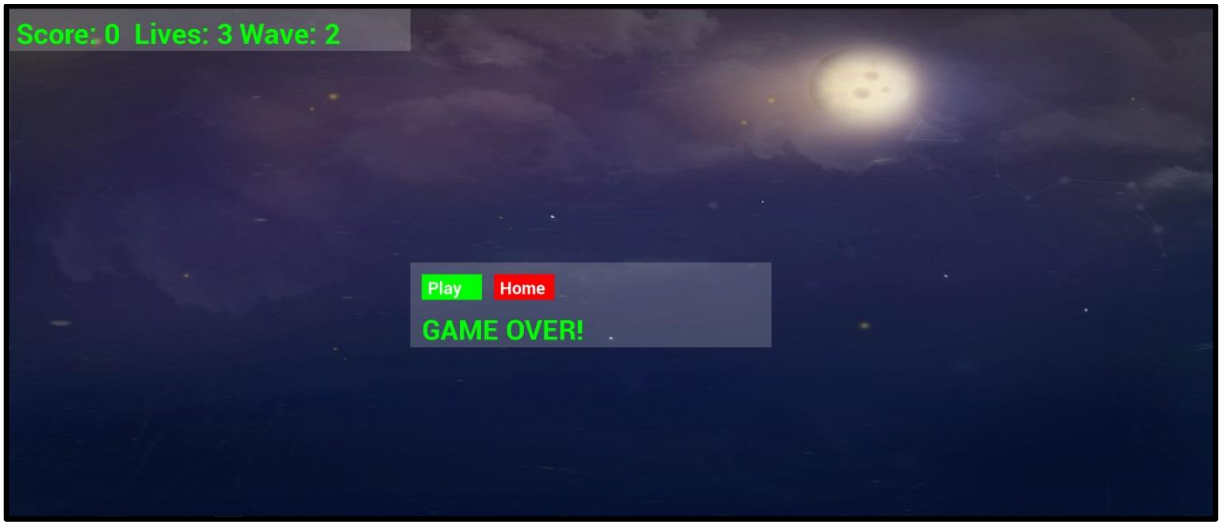

GameEngine
ScreenManager
m_ScreenManager
handleInput()
Update()
Draw()
ScreenManager
String m_CurrentScreen = "Select"
map<string, Screen> m_Screens
m_Screens[m_CurrentScreen]->handleInput()
m_Screens[m_CurrentScreen]->update()
m_Screens[m_CurrentScreen]->draw()
SwitchScreen(string nextScreen)


Screen
update()
{
// loop through all game objects
...
}
draw()
{
// loop through all game objects
...
}
GameObject
vector<Component>
UpdateComponent
GraphicsComponent
ColliderComponent
TransformComponent

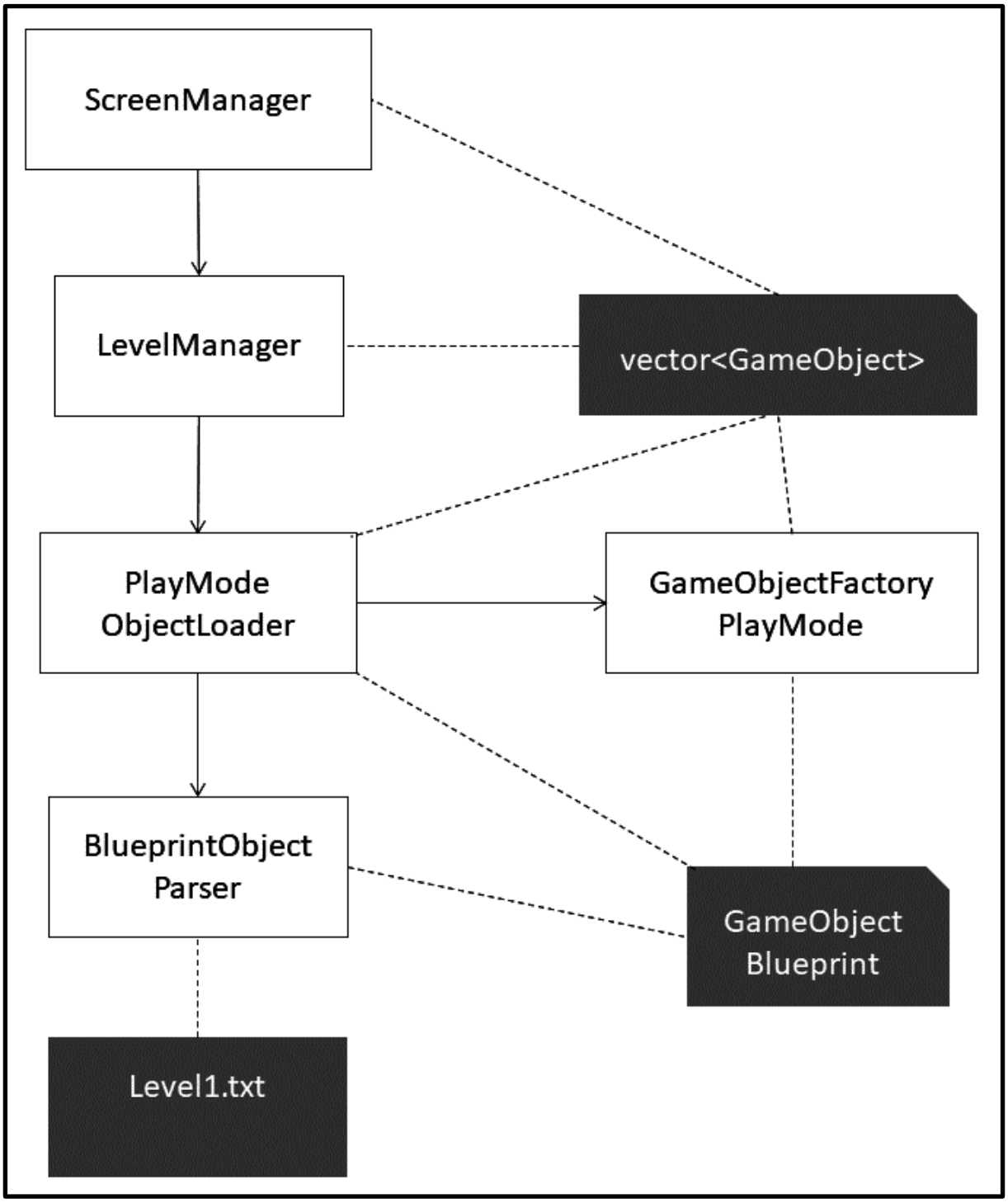

Solution 'Space Invaders ++' (1 of 1 project)
Space Invaders ++
References
External Dependencies
Header Files
Engine
FileIO
GameObjects
Screens
Game
Select
Resource Files
Source Files
Engine
FileIO
GameObjects
Screens
Game
Select
Space Invaders ++.cpp


Play
Quit
SPACE INVADERS ++

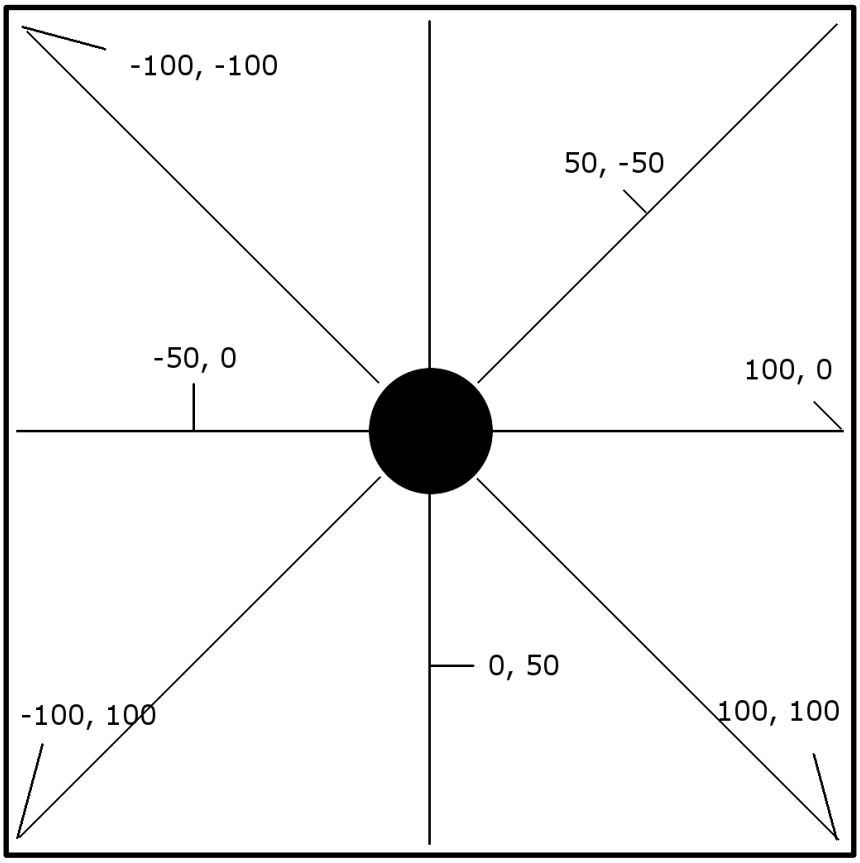

Score: 0 Lives: 3 Wave: 7847


Score: 0 Lives: 3 Wave: 2
Play
Home
GAME OVER!

Chapter 21: File I/O and the Game Object Factory



Chapter 22: Game Objects and Building a Game



Space Invaders ++.cpp

Space Invaders ++ 2

```
1    #include "GameEngine.h"
2
3    int main()
4    {
5        GameEngine m_GameEngine;
6        m_GameEngine.run();
7        return 0;
8    }
```

```
5        GameEngine m_GameEngine;
6        m_GameEngine.run();
7        return 0;
```

```
m_GameEngine;
```

m_GameEngine	{m_Clock={m_startTime={m_microseconds=-3689348814741910324 } } m_DT={m_microseconds=-3689348814741910324 } ...}
m_Clock	{m_startTime={m_microseconds=-3689348814741910324 } }
m_DT	{m_microseconds=-3689348814741910324 }
m_Window	{...}
m_ScreenManager	unique_ptr {m_Screens={ size=??? } m_LevelManager={m_GameObjects={ size=??? } WORLD_FOLDER={...} SLASH={...} } ...}
m_FPS	-107374176.
m_Resolution	{x=-107374176. y=-107374176. }
m_SoundEngine	{m_ShootBuffer={m_buffer=3435973836 m_samples={ size=0 } m_duration={m_microseconds=-3689348814741910324 } ...} ...}