

Reading Text Files

Knox Game Design

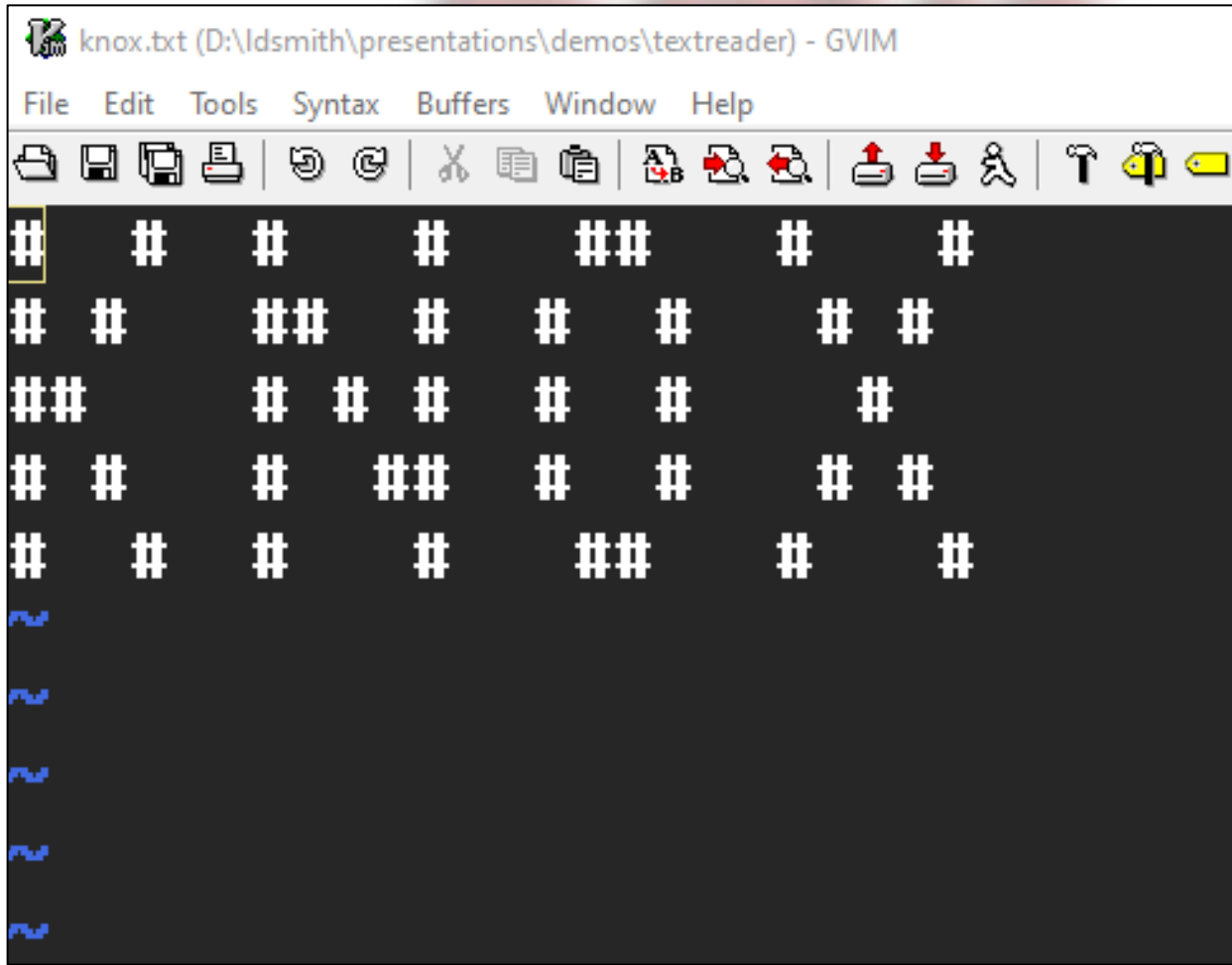
September 2022

Levi D. Smith

Basics

- Loop through each line, separated by newline '\n' character
 - Increment row number for each line
 - Reset column number to zero at the start of each line
- Loop through each character on the line
 - Increment column number for each character
 - Stop at line length
- If the character matches '#' then create object (example: wall)
- X position is the column number
- Y position is the row number
 - If using Y-up coordinate system, take the total rows and subtract the row number

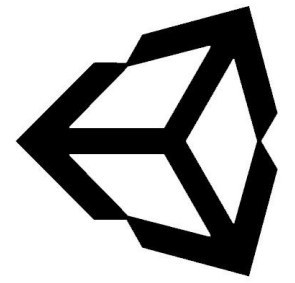
Text file example



The screenshot shows a GVIM window titled 'knox.txt (D:\ldsmith\presentations\demos\textreader) - GVIM'. The menu bar includes 'File', 'Edit', 'Tools', 'Syntax', 'Buffers', 'Window', and 'Help'. The toolbar contains various icons for file operations and editing. The main text area has a black background with white '#' characters forming a pattern:

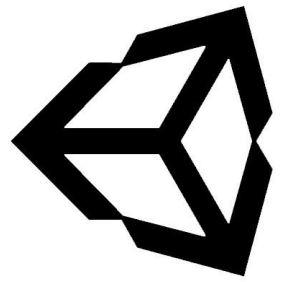
```
# # # # ## # #  
# # ## # # # #  
## # # # # # #  
# # # ## # # # #  
# # # # ## # #  
#  
#  
#  
#  
#
```

KNOX
GAME
DESIGN



Unity

- Copy text file to *Assets/Resources* folder
- Assign to **TextAsset** object in C# script
- Get contents as string with **TextAsset.text**
- Create string array with *Split('\n')*
- Loop through characters with *foreach(char c in <string>)*
- Instantiate wall prefab when character matches

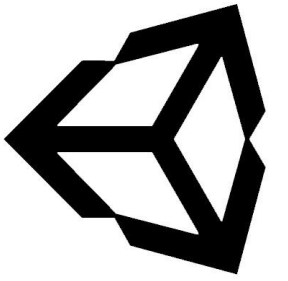
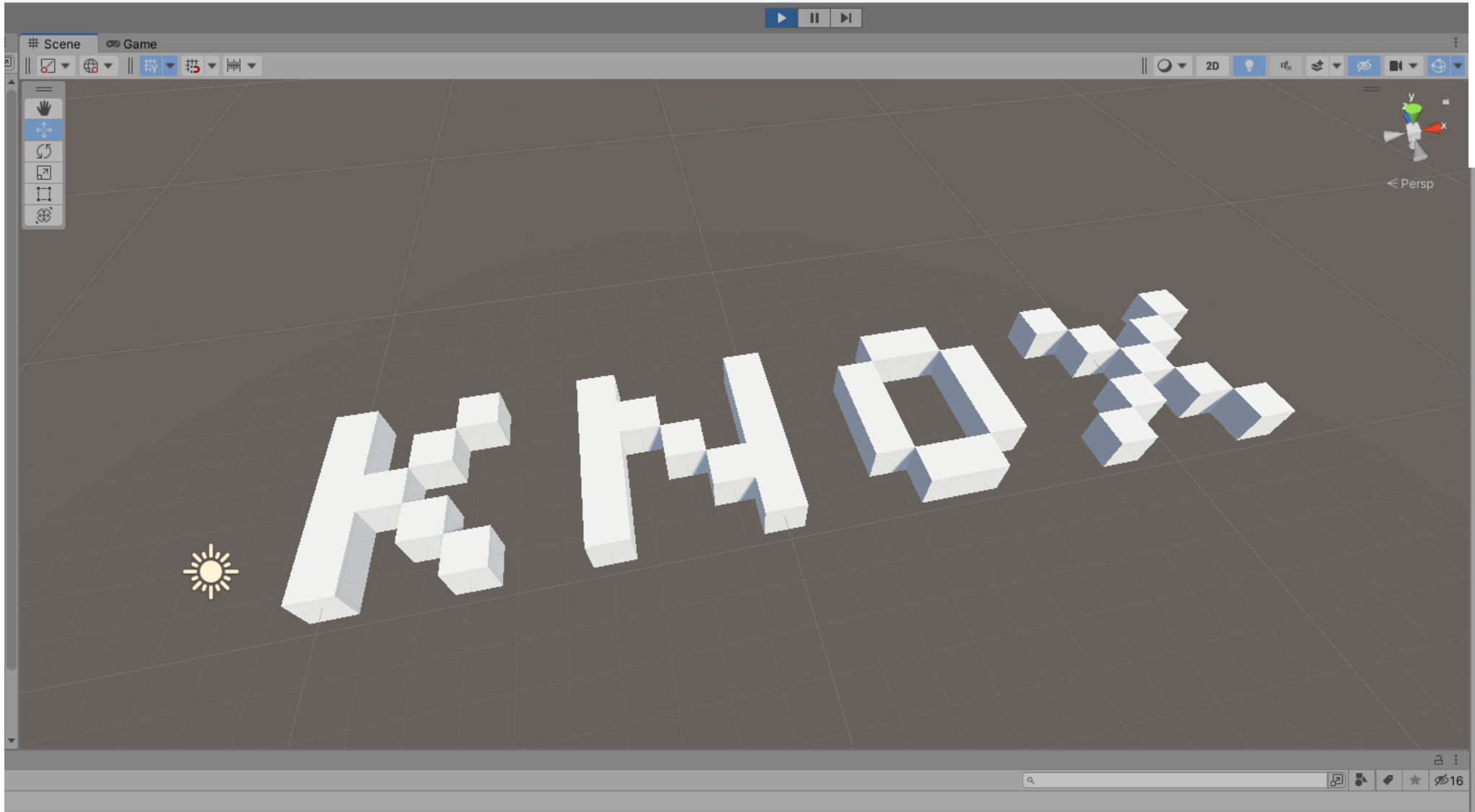


```
string strData = textData.text;
int iRow;
int iCol;

iRow = 0;
string[] strRowData = strData.Split('\n');

foreach(string strLine in strRowData) {
    iCol = 0;
    foreach (char c in strLine) {
        if (c == '#') {
            Vector3 pos = new Vector3(iCol, 0f, strRowData.Length - iRow);
            Instantiate(objWall, pos, Quaternion.identity);
        }
        iCol++;
    }
    iRow++;
}
```

KNOX
GAME
DESIGN



KNOX
GAME
DESIGN



GameMaker

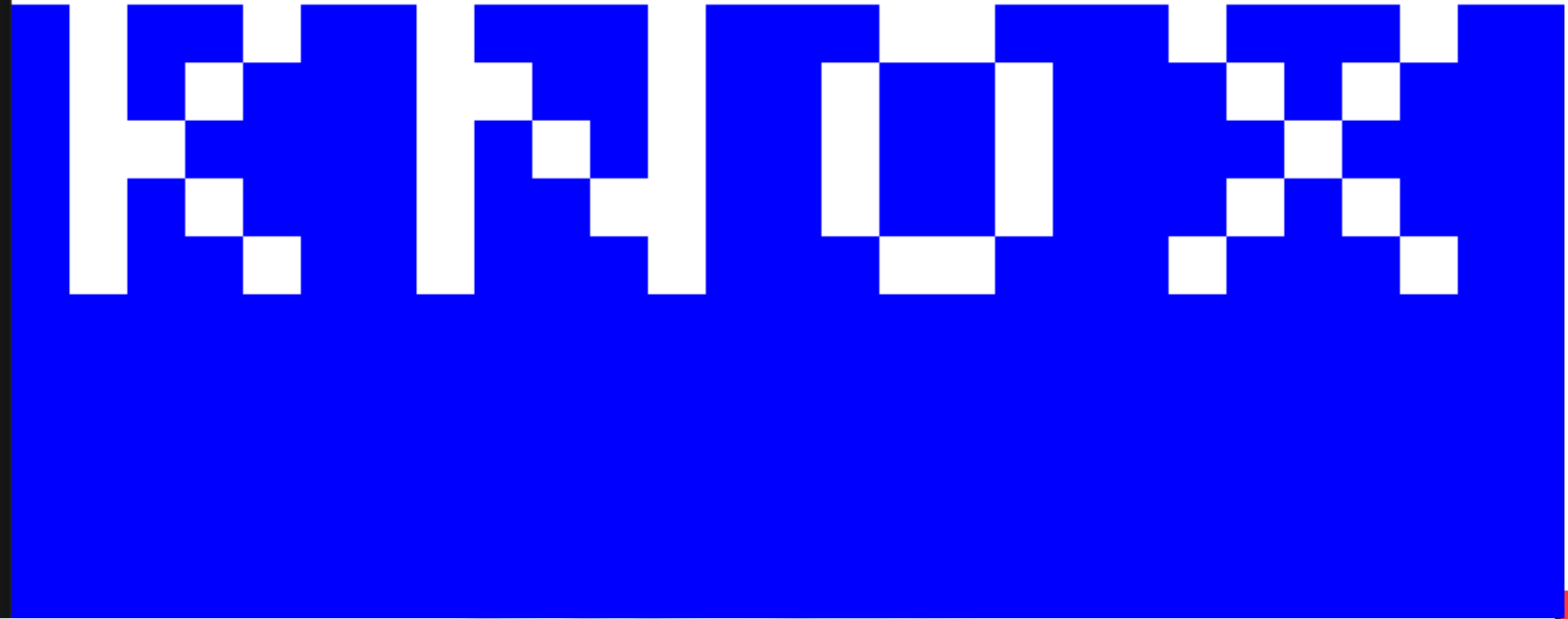
- Set file in *Included Files* (or copy to **datafiles** subdirectory)
- Create object for reading file. Remember to put object in room
- Open with *file_text_open_read(<filename>)* in Create
- Get line data with *file_text_read_string(<file>)*
- Create object with *instance_create_layer(x, y, layer, obj)*



```
5 file = file_text_open_read("knox.txt");
6
7 strData = "";
8
9 iRow = 0
10 while(!file_text_eof(file)) {
11     strData = file_text_readln(file)
12
13     iCol = 1
14     while(iCol < string_length(strData) + 1) {
15         if (string_char_at(strData, iCol) = "#") {
16             instance_create_layer(iCol * 32, iRow * 32, 0, objWall)
17         }
18         iCol += 1
19     }
20     iRow += 1
21
22 }
23 file_text_close(file)
24
```

KNOX
GAME
DESIGN

Created with GameMaker



KNOX
GAME
DESIGN



MonoGame

- Add file as *Existing Item*
- Set Copy to Output Directory to **Copy Always**
- Import System.IO
- Read each line with `StreamReader.ReadLine()`
- Read characters from string with *foreach (char c in <string>)*

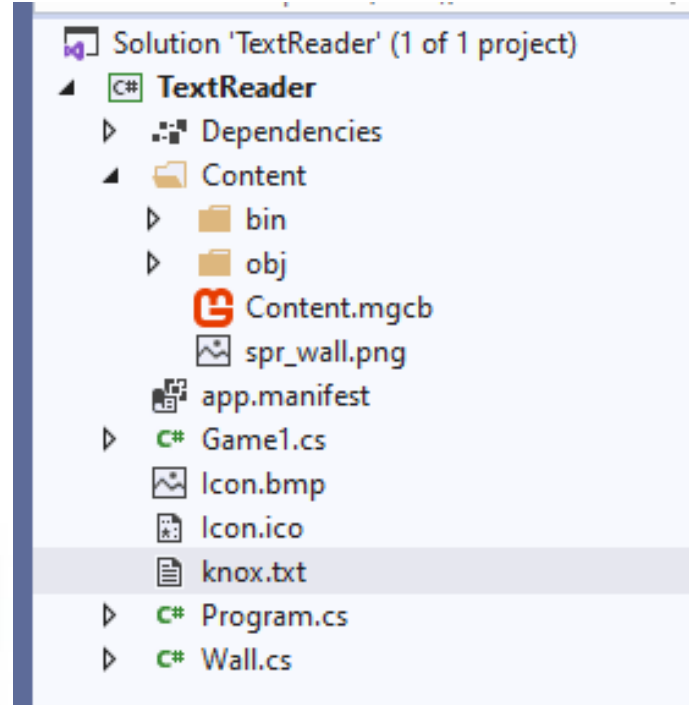


```
walls = new List<Wall>();

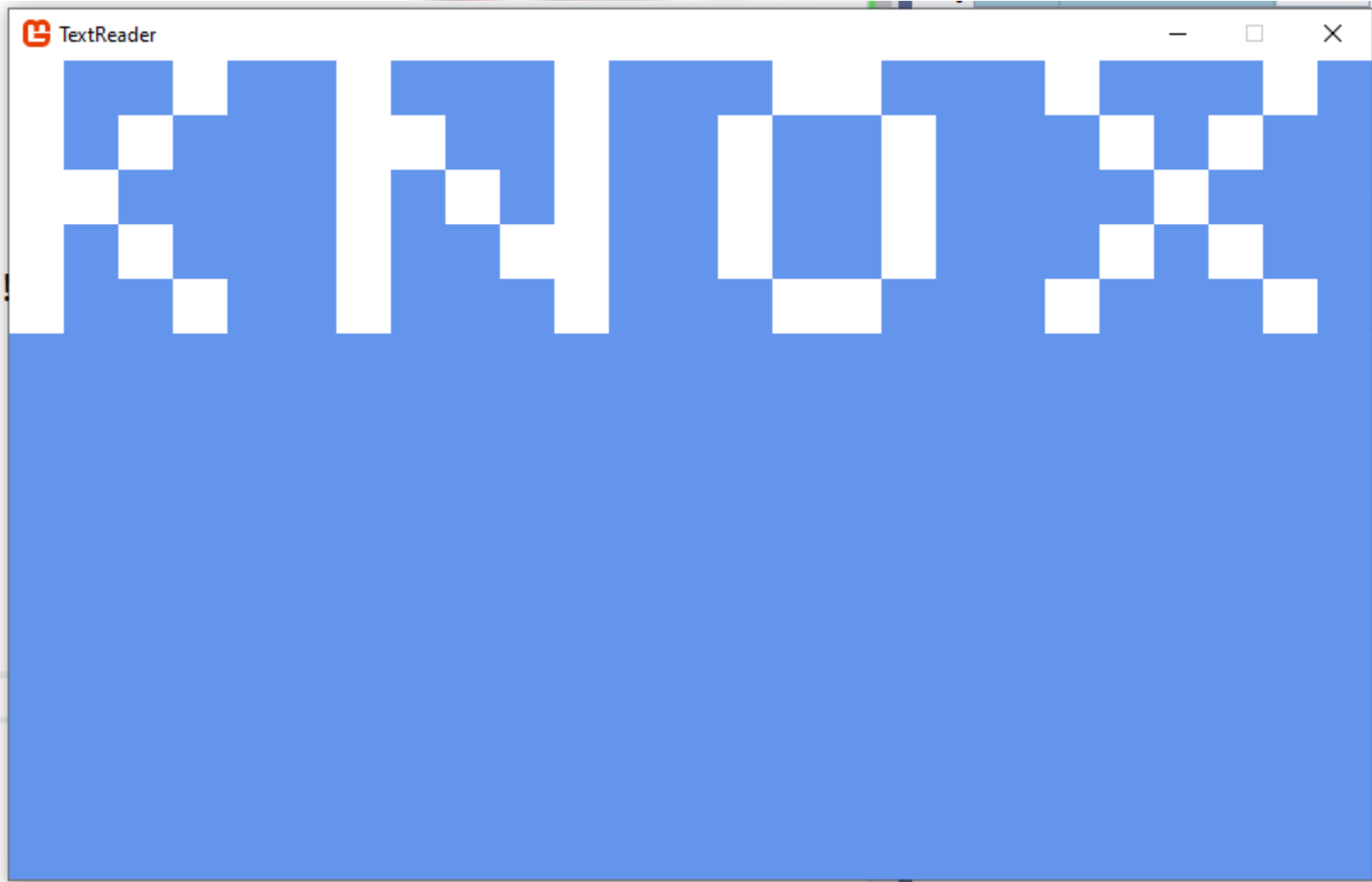
using (Stream stream = TitleContainer.OpenStream("knox.txt")) {
    using (StreamReader reader = new StreamReader(stream)) {
        string strLine;
        int iRow;
        int iCol;

        iRow = 0;
        while ((strLine = reader.ReadLine()) != null) {
            iCol = 0;

            foreach(char c in strLine) {
                if (c == '#') {
                    Wall wall = new Wall(iCol * 32, iRow * 32);
                    walls.Add(wall);
                }
                iCol++;
            }
            iRow++;
        }
    }
}
```



KNOX
GAME
DESIGN

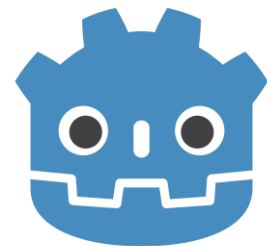


KNOX
GAME
DESIGN



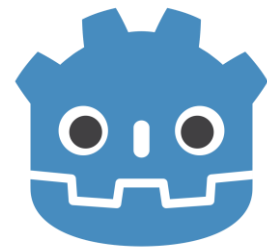
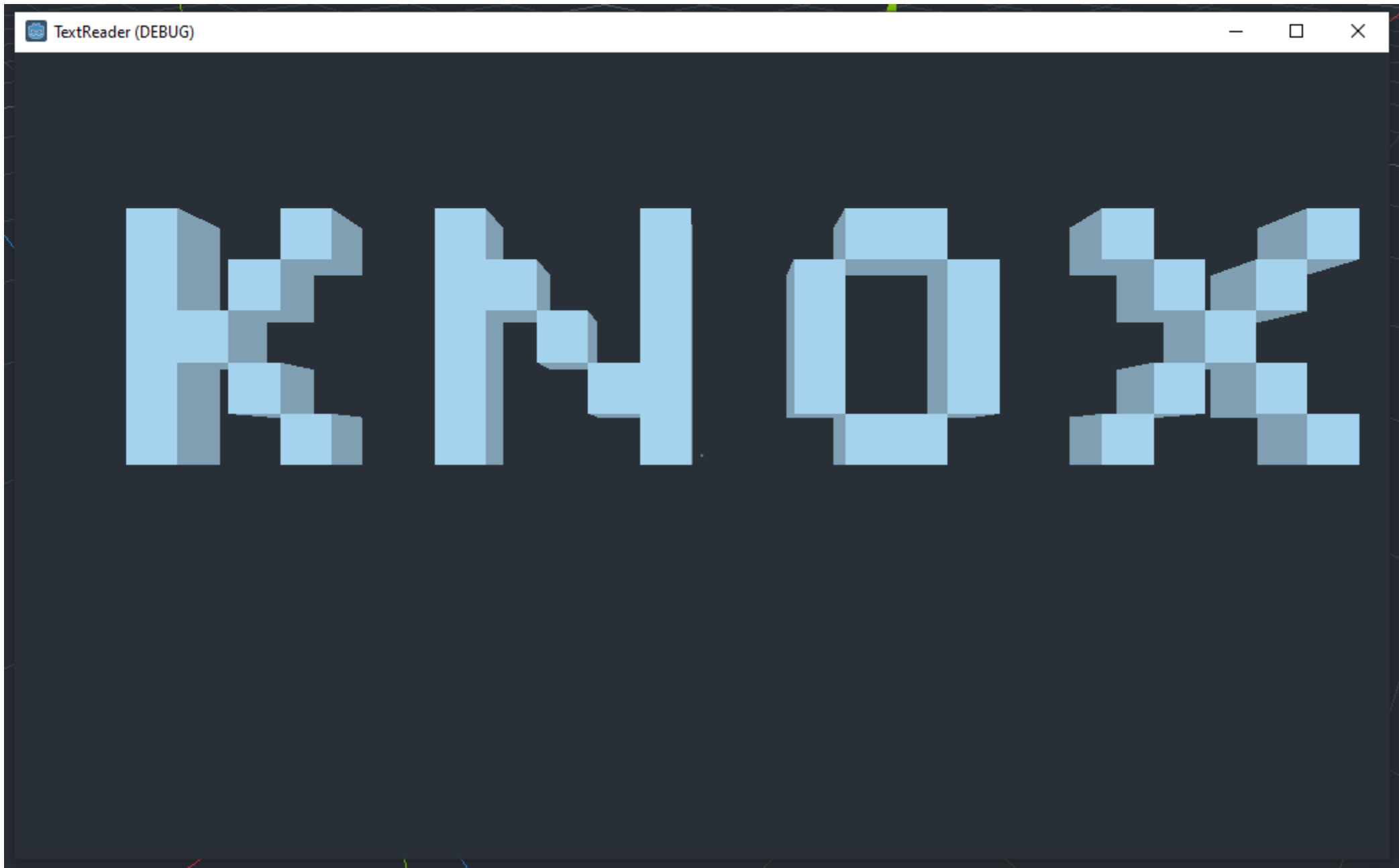
Godot

- Add the text file to the project folder
 - Note - it won't display in the `res://` FileSystem display
- Create a new *File* object using the '`res://knox.txt`' filename in *File.READ* mode
- Read lines with `f.get_line()`



```
>| var scene_wall = load("res://box.tscn")
>| var wall>|
>|
>| var strFile = 'res://knox.txt'
>| var f = File.new()
>| f.open(strFile, File.READ)
>| var iRow = 0
>| while not f.eof_reached():
>| >| var line = f.get_line()
>| >| line += " "
>| >| var iCol = 0
>| >| while (iCol < line.length()):
>| >| >| if (line[iCol] == '#'):
>| >| >| >| wall = scene_wall.instance()
>| >| >| >| wall.translation = Vector3(iCol, 0, iRow)
>| >| >| >| add_child(wall)
>| >| >| iCol += 1
>| >| >|
>| >| iRow += 1
>| f.close()
```

KNOX
GAME
DESIGN



KNOX
GAME
DESIGN

SDL



- Use *FILE *file=fopen(<filename>, "r")* to open file
- Read lines with *fgets(<line>, <chars to read>, file)*
- Loop through lines until *fgets* equals *NULL*
- Loop through chars in char array using bracket notation
- Ensure that enough characters are read to reach end of line
- Use *fclose(file)* to cleanup after file has been read
- Use *fgetc* if length of line is not known

KNOX
GAME
DESIGN

M /d/ldsmith/presentations/demos/textreader/sdl

```
#include <stdio.h>
#include <SDL2/SDL.h>
#define CHARS_PER_LINE 32
#define WALL_WIDTH 32
#define WALL_HEIGHT 32

SDL_Window *window = NULL;
SDL_Surface *screenSurface = NULL;

void parse_file() {
    SDL_Surface *sprWall;
    SDL_Rect rect;
    sprWall = SDL_LoadBMP("spr_wall.bmp");
    rect.w = WALL_WIDTH;
    rect.h = WALL_HEIGHT;

    FILE *file = fopen("knox.txt", "r");
    char strLine[CHARS_PER_LINE];

    int iRow = 0;
    int iCol = 0;
    while (fgets(strLine, CHARS_PER_LINE, file) != NULL) {
        while (iCol < CHARS_PER_LINE) {
            if (strLine[iCol] == '#') {
                rect.x = iCol * WALL_WIDTH;
                rect.y = iRow * WALL_HEIGHT;
                SDL_BlitSurface(sprWall, NULL, screenSurface, &rect);
            }
            iCol++;
        }
        iRow++;
        iCol = 0;
    }

    fclose(file);
    SDL_FreeSurface(sprWall);
}
```



```
int main(int argc, char *args[]) {
    SDL_Init(SDL_INIT_VIDEO);

    window = SDL_CreateWindow("SDL Text Reader", SDL_WINDOWPOS_UNDEFINED,
                              SDL_WINDOWPOS_UNDEFINED, 1024, 768, SDL_WINDOW_SHOWN);

    screenSurface = SDL_GetWindowSurface(window);
    SDL_FillRect(screenSurface, NULL, SDL_MapRGB(screenSurface->format,
                                                0x00, 0x00, 0xFF));

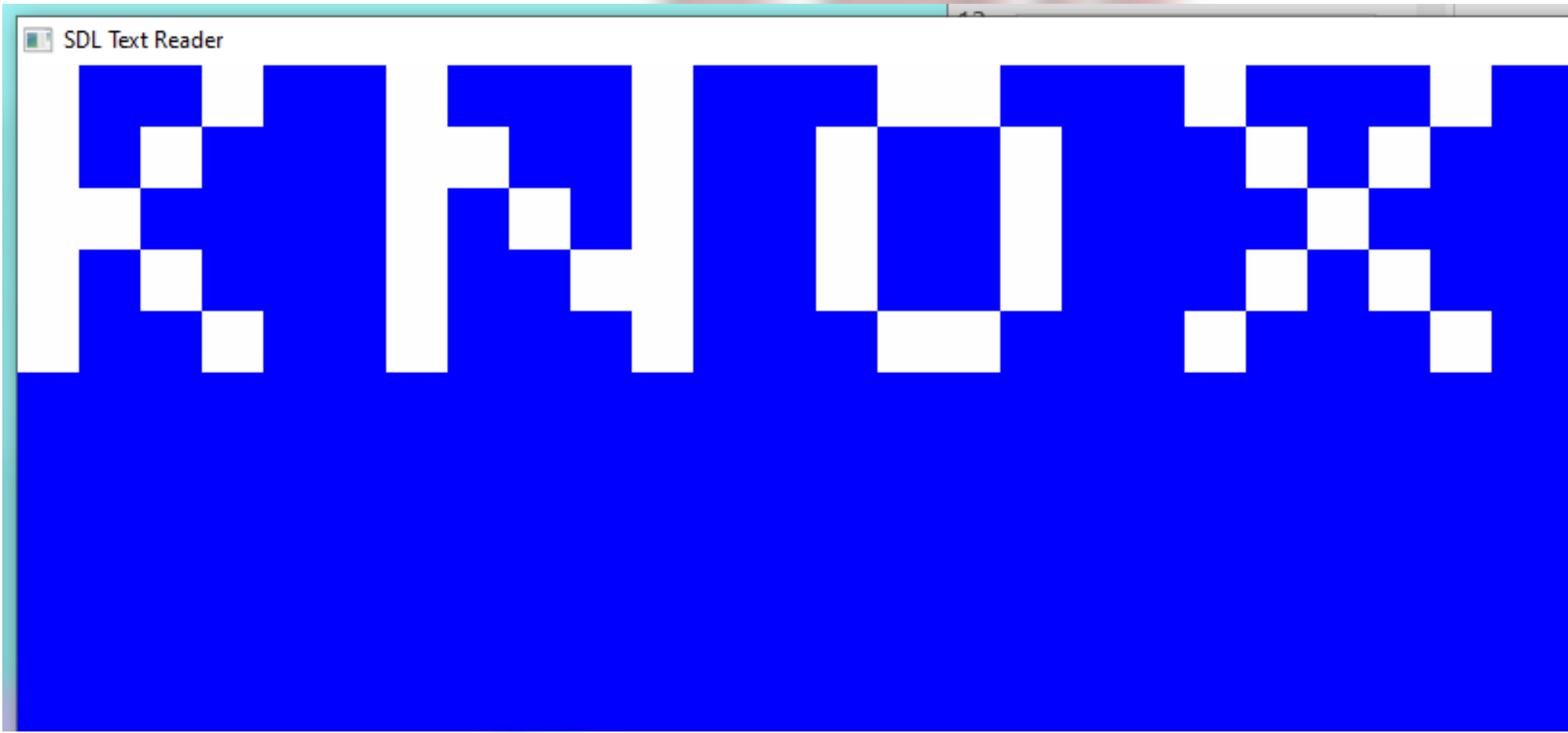
    parse_file();

    SDL_UpdateWindowSurface(window);
    SDL_Delay(2000);
    SDL_DestroyWindow(window);
    SDL_Quit();
    return 0;
}
```

```
$ gcc textreader.c -o textreader `sdl2-config --cflags --libs`
```

KNOX
GAME
DESIGN

KNOX



KNOX
GAME
DESIGN