

# Software Coding Specs

---



**Team members:**

Alper Akbal (developer)  
Farid Harhad (developer & team lead)  
Sean O'Connell (developer)  
Andrew Staley (developer)  
Tarik Teksen Tural (developer)

**Instructors:**

Dr. Jason Leigh ([spiff@uic.edu](mailto:spiff@uic.edu))  
Dr. Gabrielle Allen ([gallen@cct.lsu.edu](mailto:gallen@cct.lsu.edu))

## Contents

Pathfinding.....	3
Sound .....	<b>Error! Bookmark not defined.</b>
Midterm Project Goals.....	5
Game Code Architecture Summary .....	6

## Pathfinding

(Alp)

### References:

<http://theory.stanford.edu/~amitp/GameProgramming/>

### Class Declarations and Function Prototypes

```
struct PathNode
{
    PathNode() { // default constructor
        row = column = 0;
        next = prev = 0;
    }

    int row, column; // location in 2d map
    PathNode* next; // pointer to next PathNode in calculated path
    PathNode* prev; // pointer to previous PathNode in path
};

PathNode* FindPath(int* map,
                  int mapRows,
                  int mapCols,
                  int startRow,
                  int startCol,
                  int destRow,
                  int destCol);
```

### How To Index Map Array:

The map array is a 1-dimensional array containing all of the data for the 2D map. Hence you have to index it correctly like so:

```
for (int row = 0; row < mapRows; row++)
{
    for (int col = 0; col < mapCols; col++)
    {
        int index = row * mapCols + col;
        int value = map[index];
    }
}
```

### Example Use Of The FindPath Function

```
int* map = CreateMapArrayFromTerrain(heightmap);

mapCols = 512;
mapRows = 512;
startCol = 0;
startRow = 0;
destCol = 511;
```

```
destRow = 511;

PathNode* start = FindPath(map, mapRows, mapCols, startRow, startCol, destRow, destCol);

PathNode* curNode = start;
while (curNode != 0)
{
    // do stuff with curNode
    curNode = curNode->next;
}
```

The FindPath function should return a linked-list of PathNodes where the returned PathNode object is the start location. Then each successive node linked to it in the list should be from the start position to the dest position.

## Sound

(Terik)

### References

<http://www.ambiera.com/irrklang/>

### Goals for Game

We would like to be able to arbitrarily 'submit' sounds to some class object for playing. We shouldn't have to worry about whether it is actually played, or if it is mixed, ect...All we want is a simple interface to allow us to do:

```
SomeClassObject->PlaySound( effect1 );
```

Or

```
SomeClassObject->LoopSound ( backgroundMusic );
```

Assuming our sound card supports only 8 slots to mix simultaneous sounds. We would want to be able to arbitrarily submit 30 sounds and not worry about which ones will actually be played.

```
PlaySound (sound1);
```

```
PlaySound (sound2);
```

```
PlaySound (sound3);
```

```
...
```

```
PlaySound(sound30);
```

We also want to be able to specify the 3D world position of the sound to be played, hence PlaySound will have multiple overloaded versions of itself, possibly like so:

```
void PlaySound (Sound sound, int volume);
```

```
void PlaySound (Sound sound, int volume, float x, float y, float z);
```

All of this depends on the features IrrKlang supports. After briefly reading through its feature list, it may already take care of a lot of this for us. We just need to learn how to use it. Below is a complete bulleted feature list of what we would need for the game.

- Play and mix simultaneous sounds
- Play background music (and be able to specify nothing should EVER stop the music from being played)
- Possibly specify the 'priority' of sounds being playing so some take precedence over others
- Specify 3D world position of sound
- Specify volume of sound

I wish I could be more specific on some of these things, but I do not have a lot of experience working with sounds libraries. But overall, the IrrKlang library looks like it will definitely meet all of our requirements, you will just need to figure out how to use them and then possibly design an easy to use interface for use in the game.

## Midterm Project Goals

- Have a complete working game prototype with the following gameplay features:
  - 1 working playable level
  - Character selection screen complete
  - Characters standing on wall with movable aimer sprites
  - Moving enemy AI bots that travel towards base and attack players
  - Ability to shoot the AI bots and kill them
  - A score summary screen calculating stats for each player when level ends
  - Sound effects and background music
- Coding wise, we need the following done:
  - Loading of game world from IrrEdit world editor
  - Flocking code integrated with pathfinding code to move bugs across the terrain correctly
  - Correct vector math in place to compute where each character is shooting on the screen
  - Simple collision detection for bullets and bugs when they get shot
  - Simple classes representing each Player
    - Avatar selected
    - Player Name
    - Health
    - Weapon
    - Ammo
    - Score
  - Simple classes representing each AI bug
    - Speed
    - Health
    - Attack Damage

- Intelligence? (makes better decisions when flocking towards players?)

## **Game Code Architecture Summary**

*To be filled in later*