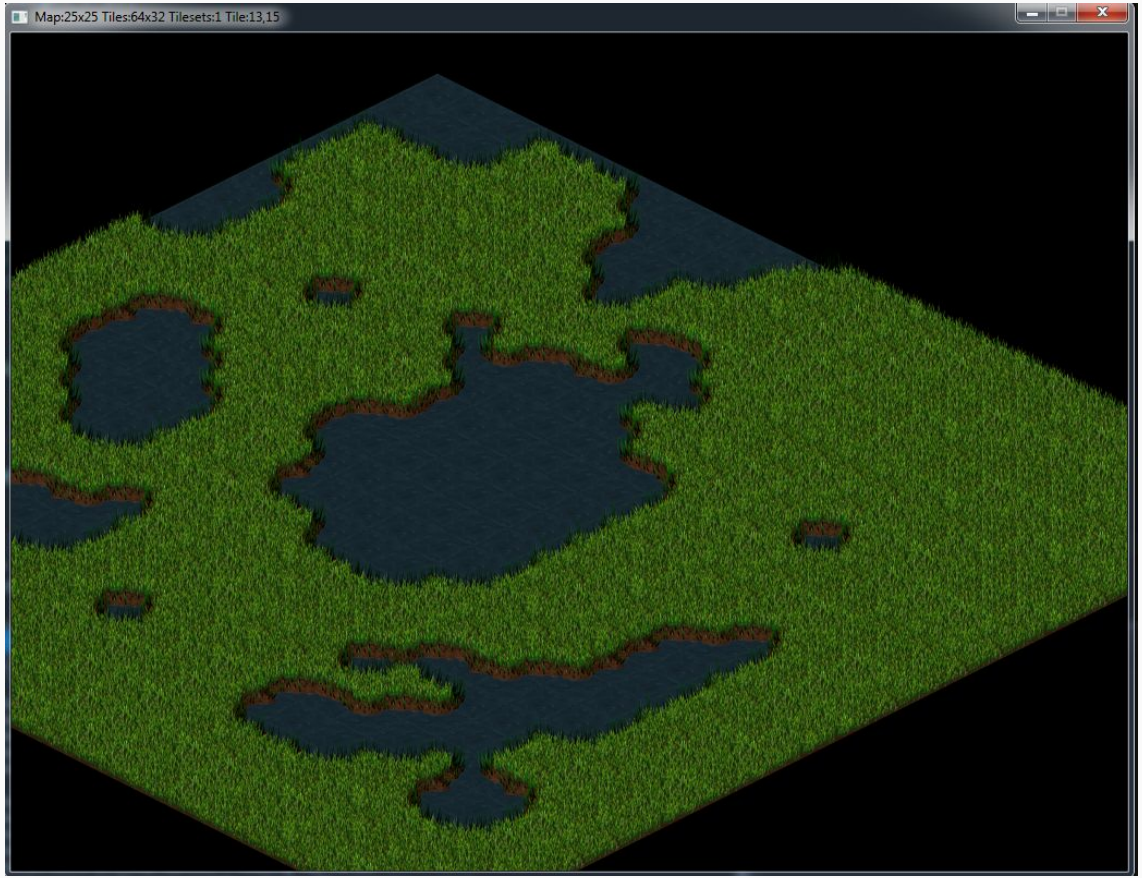
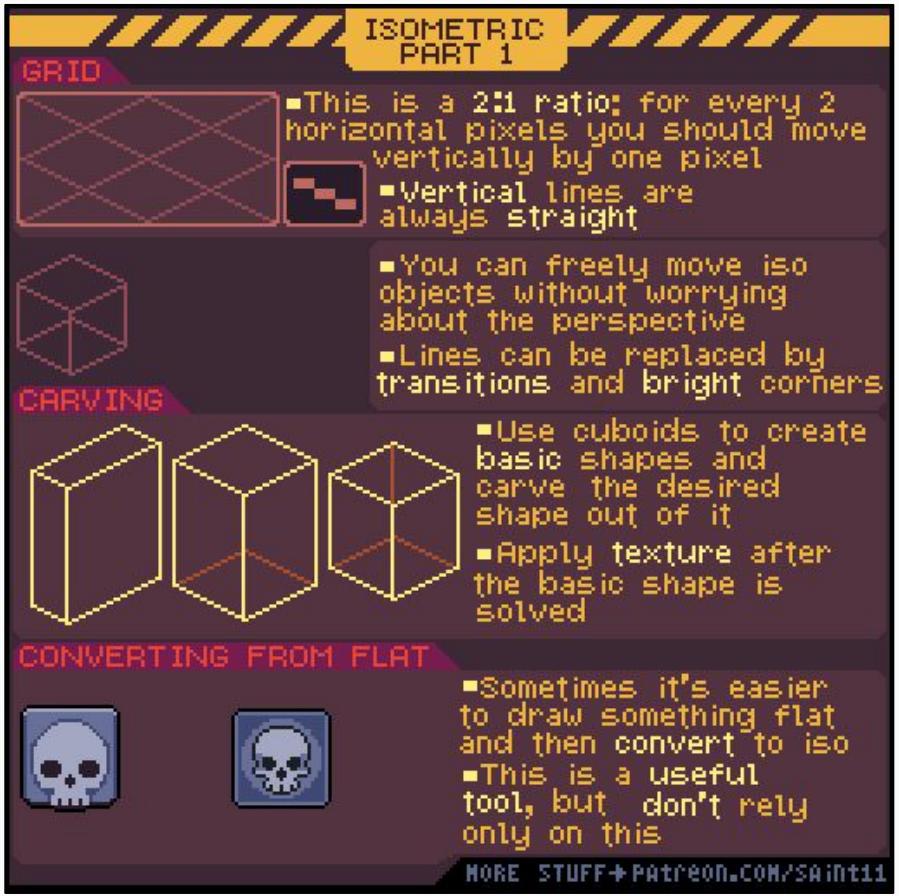
Game Development

Isometric Draw

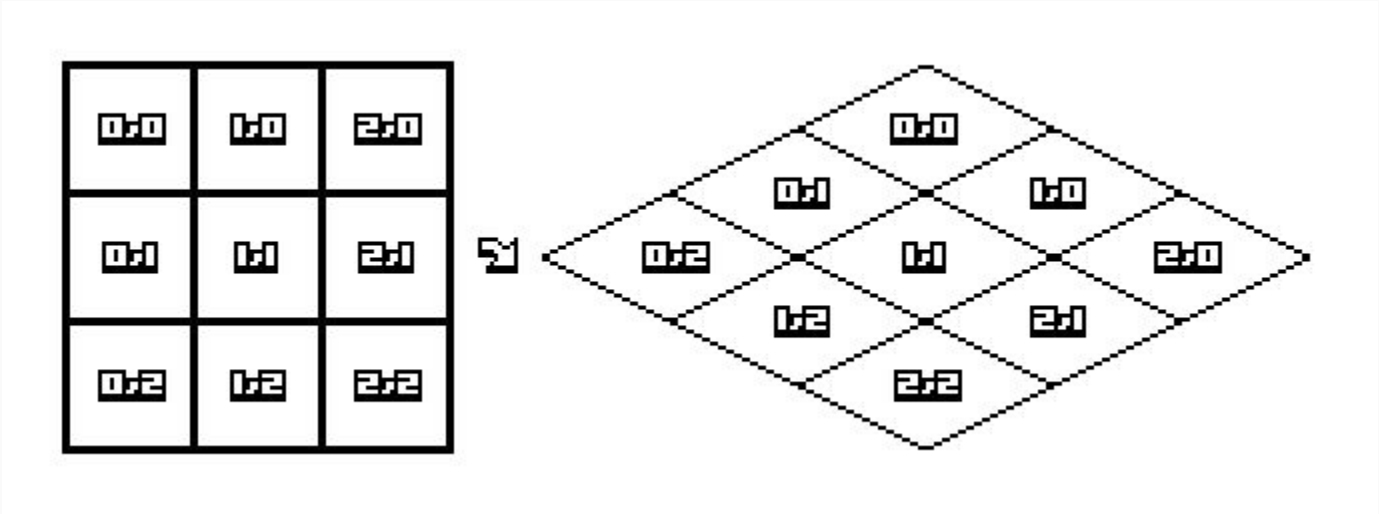
Let’s draw isometric maps



We will use the traditional 2:1 ratio

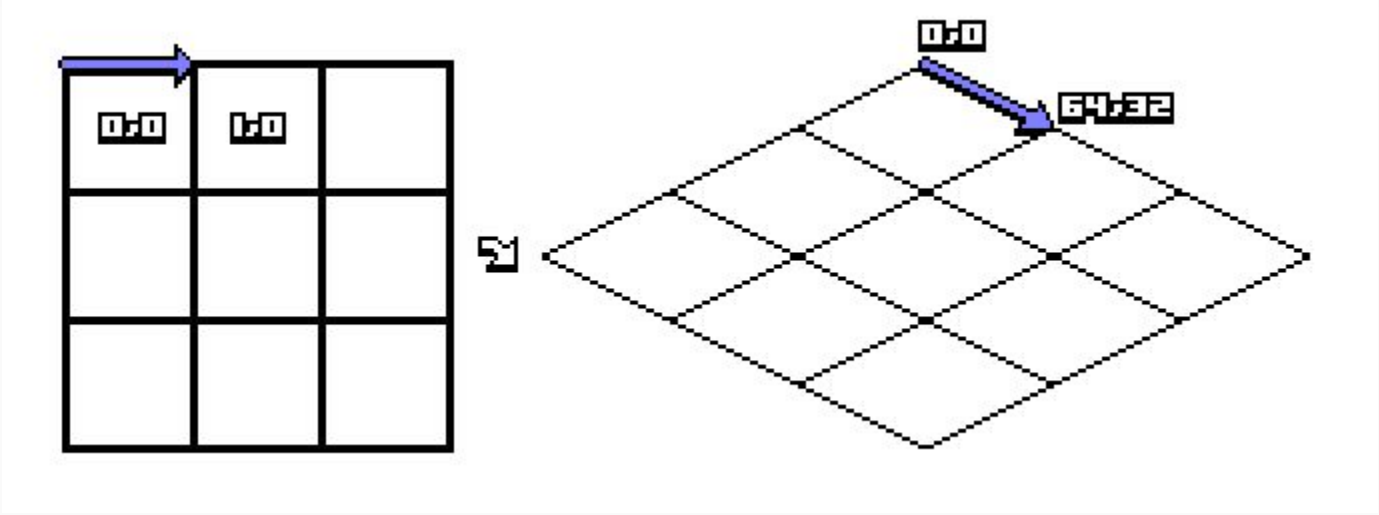


Isometric Projection (one of many)



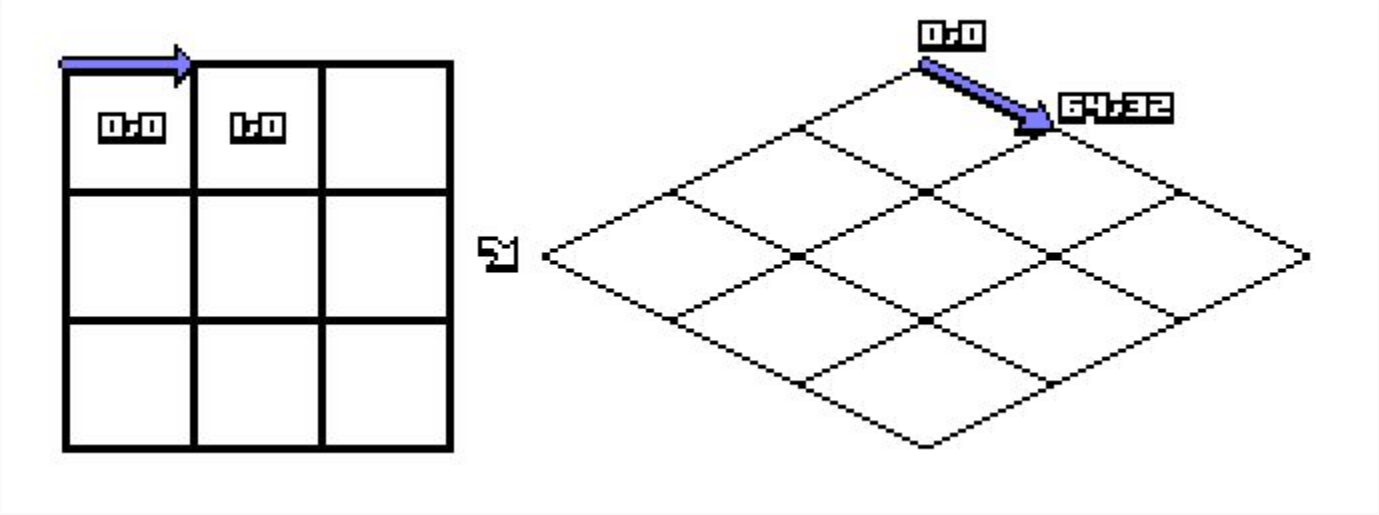
With pen & paper write down all screen coordinates (each projected tile is 128x64)

Map to World



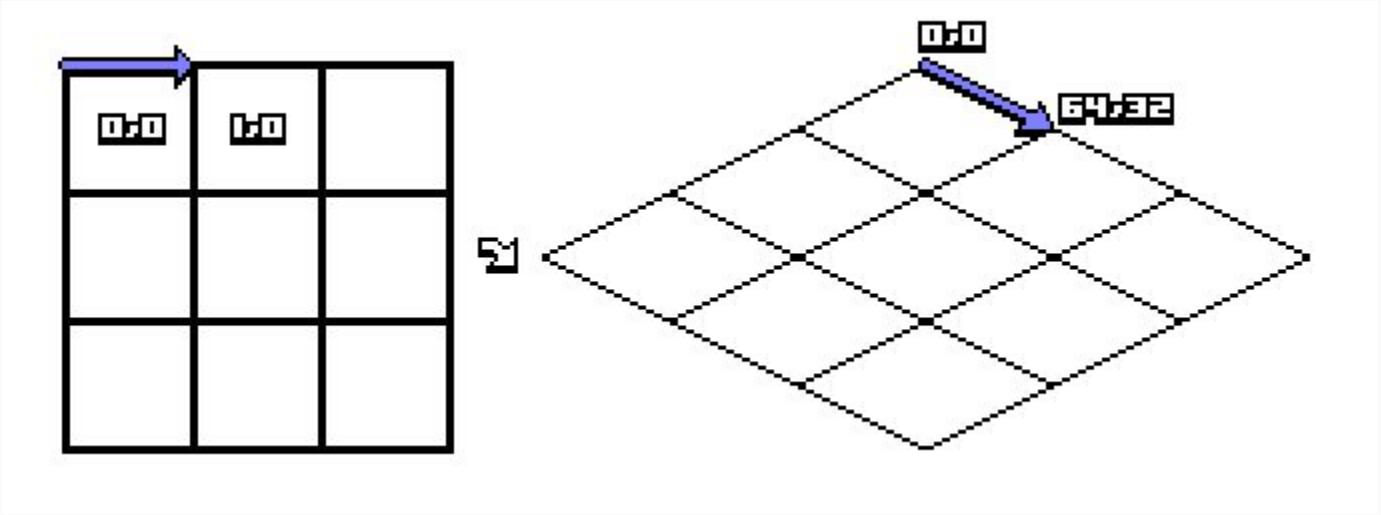
To go from 0,0 to 1,0 we increase **x** by 64 **(half tileset width)** and **y** by 32 **(half tileset height)**

Map to World



What happens if we want to go to 0,1 ? We decrease the **x** by half tile width and increase **y** by half tile height to get **(-64,32)**

Map to World



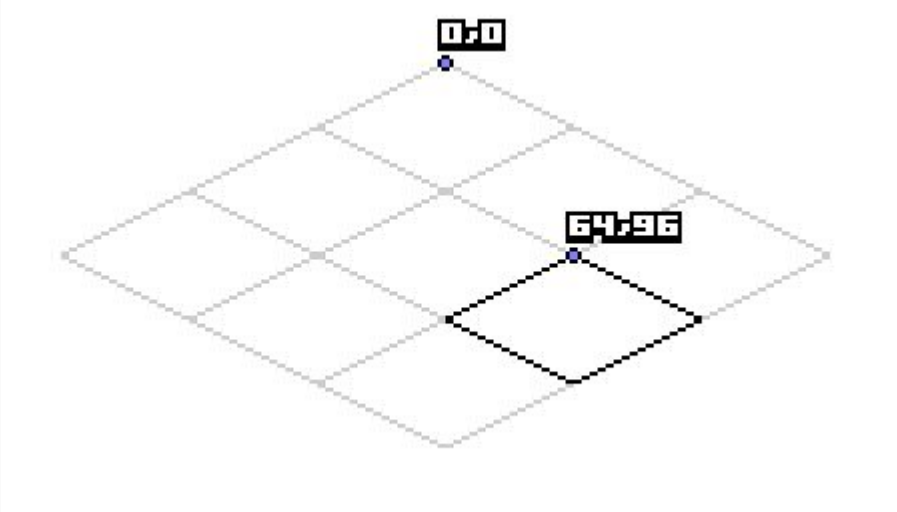
screen.x = map.x * HALF_TILE_WIDTH - map.y * HALF_TILE_WIDTH;

screen.y = map.x * HALF_TILE_HEIGHT + map.y * HALF_TILE_HEIGHT;

Map to World

What would be the screen coordinates to tile 2,1 ?

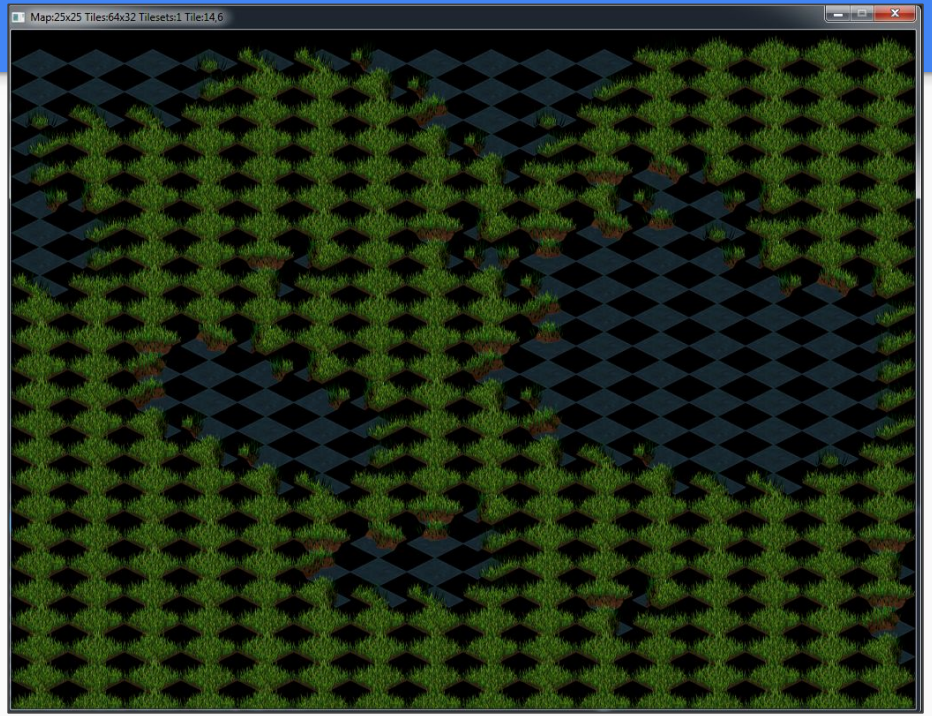
Would be pixel **64,96** !



Map to World

- Orthogonal is:
  - *ret.x = x * data.tile_width;*
  - *ret.y = y * data.tile_height;*

- Isometric changes to:
  - *ret.x = (x - y) * (data.tile_width * 0.5f);*
  - *ret.y = (x + y) * (data.tile_height * 0.5f);*



TODO 1

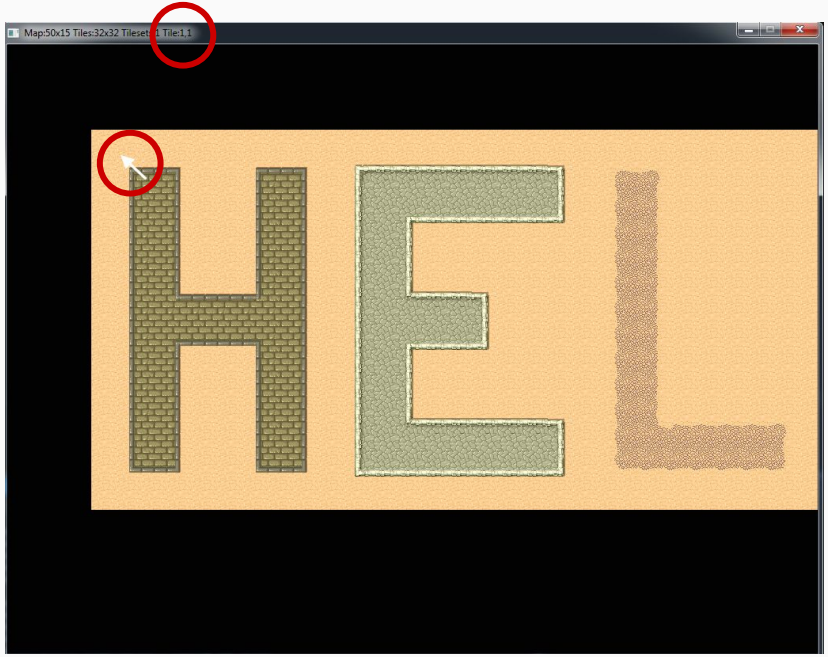
“Add isometric map to world coordinates”

- Add a case in *MapToWorld* function for isometric maps
- Try loading **iso.tmx**

TODO 2

“Add orthographic world to map coordinates”

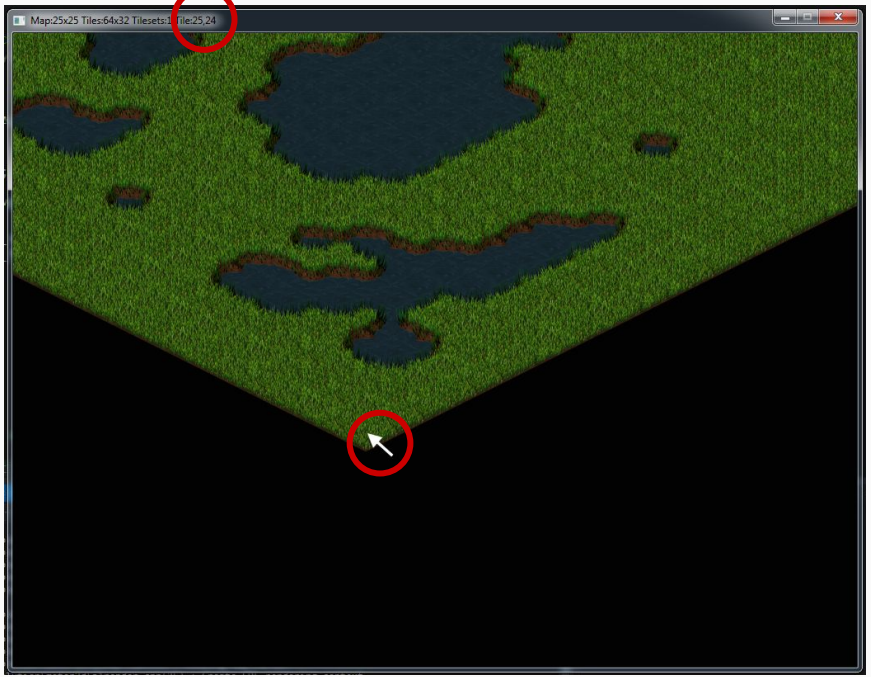
- It is just the inverse math from MapToWorld
- Check that you receive something that makes sense



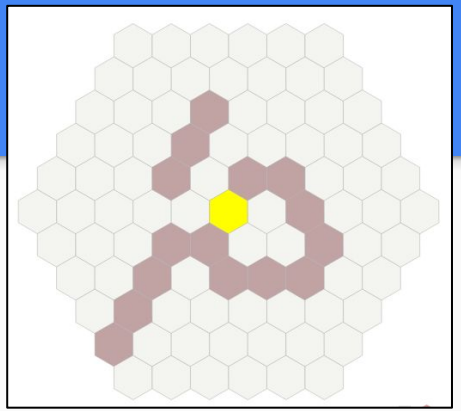
TODO 3

“Add the case for isometric maps to WorldToMap”

- Again, you have to make some math!
- Check the result



Documentation



- If in doubt, read this article about isometric math
- If curious, check how to deal with hexagonal maps