Game Dev: Orthogonal Draw

Ricard Pillosu - UPC

Drawing Orthogonal Maps (check solution.exe)

TODO 1

“Create a struct for the map layer”

- Create all vars to store layer data
- For the core of the data, just use *pointer to unsigned int* for now

```
<layer name="name" width="50" height="15">
  <data>
   <tile gid="30"/>
   ...
   <tile gid="0"/>
   </data>
 </layer>
```

```
<layer name="name" width="50" height="15">
  <data encoding="csv">
   30,30,31,2,1,30,...
  </data>
 </layer>
```

TODO 2

“Add a list/array of layers to the map!” + “clean up all layer data”

- Just add a list/array of layers to your map structure
- As with the tilesets, make sure all memory is deleted
- OPTION: to delete the memory for tile data, you could use a destructor in the layer struct

TODO 3

“Create the definition for a function that loads a single layer”

- First load all header data (name, width, height)
- Once the array is allocated, use **memset** to fill it with zeroes

```
void * memset ( void * ptr, int value, size_t num );
```

- Then iterate all tile gid and store them in an array
- What will be the size of that array ? … how many do you have in the TMX file ?

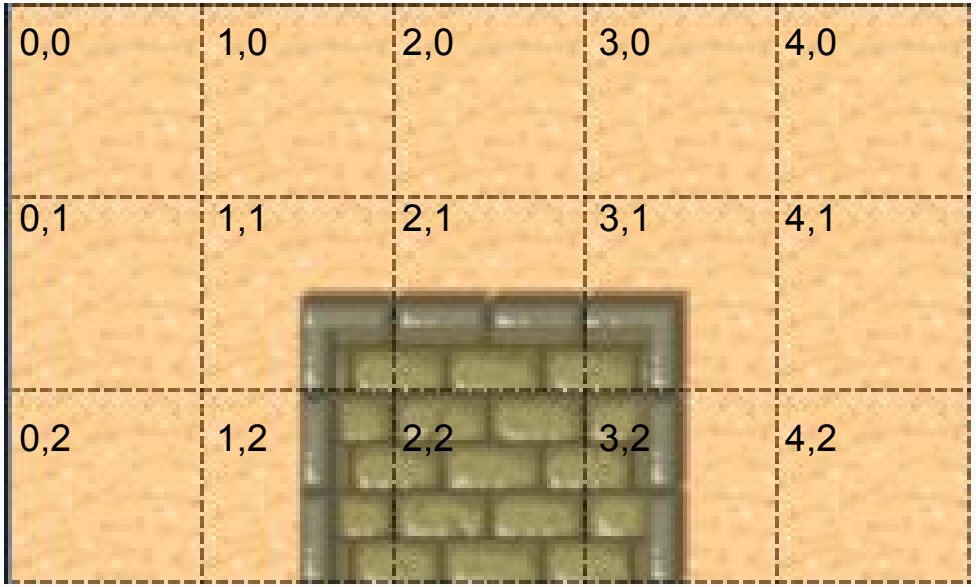
TODO 4

“Iterate all layers and load each of them”

- Very similar to tileset loading just above the TODO

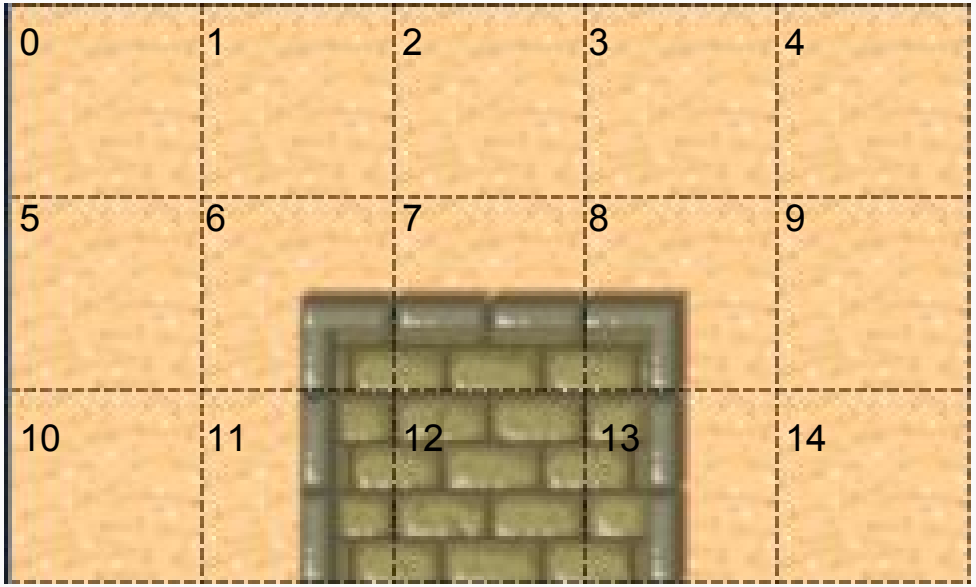
Drawing the map

- We draw 1 tile each time
- Each have a x,y



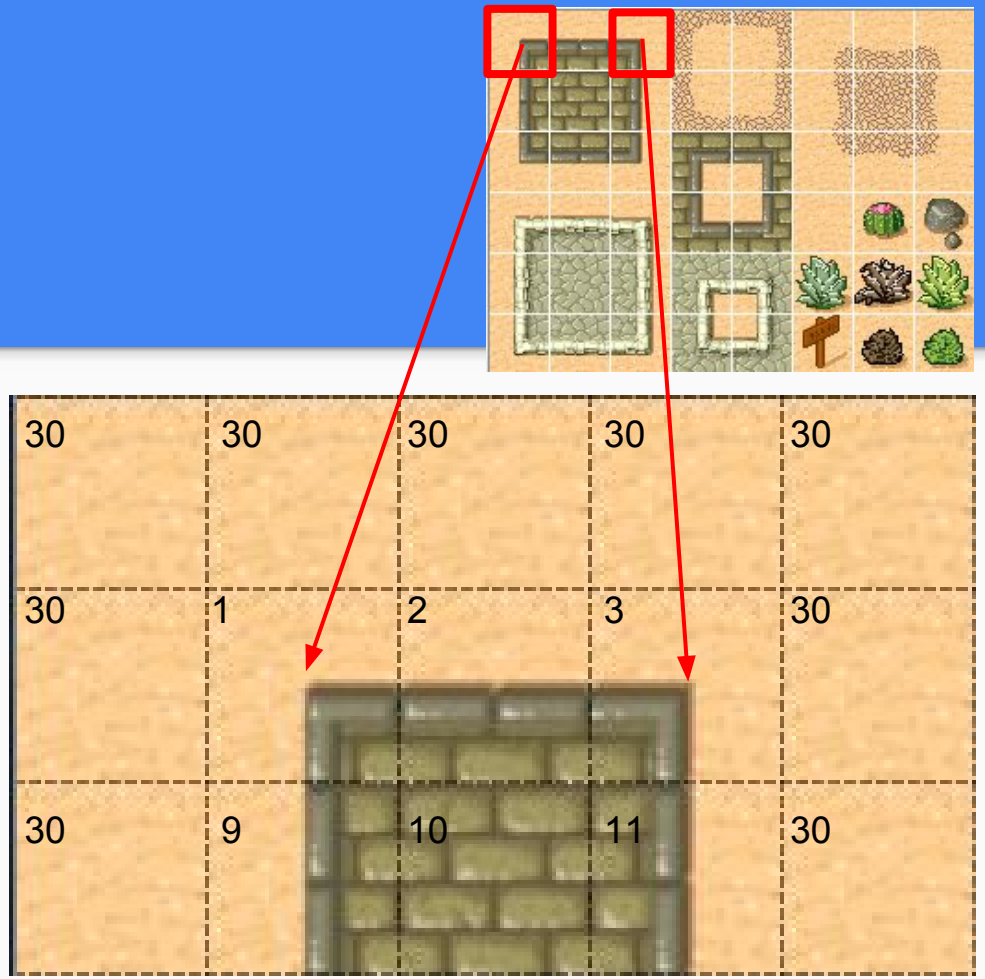
Drawing the map

- We draw 1 tile each time
- Each have a x,y
- We transform to 1 dimension



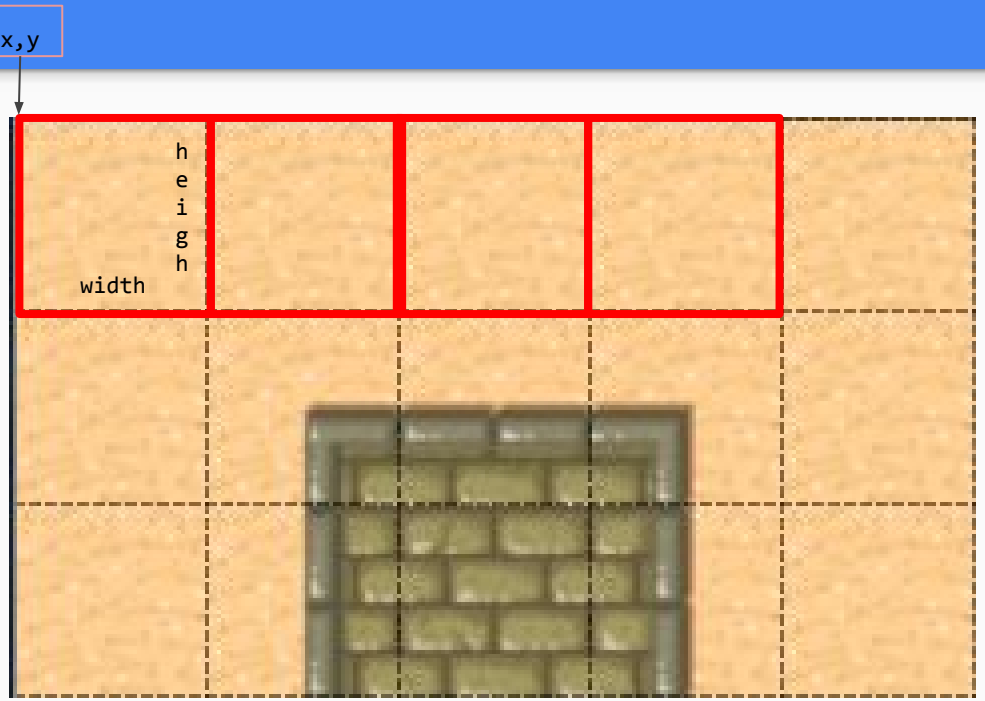
Drawing the map

- We draw 1 tile each time
- Each have a x,y
- We transform to 1 dimension
- Later we read tileset **sprite id**
- Then find out their tileset **Rect**



Drawing the map

- We draw 1 tile each time
- Each have a x,y
- We transform to 1 dimension
- Later we read tileset **sprite id**
- Then find out their tileset **Rect**
- Now find tile place in the **world**
- Now we can Blit!



TODO 5

“Prepare the loop to draw all tilesets + Blit”

- We want to iterate all tilesets in order
- That means **two nested** for()
- Then a call to render->Blit()

TODO 6

“Short function to get the value of x, y out of a vector”

- How do we access a 1 dimension array as a 2 dimension (x,y) ?
- Make the method inline
- Add this to the main draw loop, and be sure to ignore tiles of id == 0

```
inline uint Get(int x, int y) const
```

TODO 7

“Create a method that receives a tile id and returns it's Rect”

- On each tileset, gid start from “*firstgid*” value found in the tileset header
- Take in account the margin when calculating the Rect
- The width and height are easy and fixed
- For the x and y you will have to take in account the width and height of the whole tileset

TODO 8

“Create a method that translates x, y coordinates from map positions to world positions”

- It’s very simple!
- You only really need the size of the tiles

TODO 9

“Complete the draw function”

- Now we have everything to properly Blit every tile:
  1. Find which tile id is on x,y coordinates
  2. Find out that Tile’s Rect inside the tileset Image
  3. Find out where in the World (screen) we have to draw
  4. Blit!

Homework

- Define in config.xml which is the map that should be loaded
- Try loading and drawing all TMX inside the examples directory of Tiled
- Add code to be able to load the rest of the TMX correctly
- **Not** the isometric maps, we will work on those later