How to input data to create maps

List of Enums used

Many of the data used to classified elements in maps are defined by enums.

Here are the list of enums used (some are place more appropriately placed close to the specific part relating to that enum):

```
class EnvironmentObjectFactory // Singleton Factory
{
    public:
    enum EnvironmentObjectType
    {
        WARP_PIPE, // 0
        WARP_PIPE_SHORT, // 1
        WARP_PIPE_TINY, // 2
        BRICK, // so on & so on
        HARD_BLOCK,
        BLUE_BRICK,
        LEFT_GRASS_PLATFORM,
        MID_GRASS_PLATFORM,
        RIGHT_GRASS_PLATFORM,
        BLUE_HARD_BLOCK
    };

class EnvironmentInteractiveObjectFactory // Singleton Factory
{
    public:
    enum EnvironmentInteractiveObjectType
    {
        QUESTION_BLOCK,
        BREAKABLE_BRICK
    };

class DrawableObjectFactory
{
    public:
    enum DrawableObjectType
    {
        GRASS,
        CLOUD,
        MOUNTAIN,
        CASTLE,
        DIRT
    };

class EnemyFactory
```

```
{
    public:
        enum EnemyType {
            GOOMBA,
            KOOPA_TROOPA,
            PIRANHA_PLANT,
            INVERSE_PIRANHA_PLANT,
            LAKITU,
            SHY_GUY,
            PROJECTILE
        };

enum Itemtype {
    COIN,
    MUSHROOM,
    FIREFLOWER,
    STARMAN,
    IDLECOIN
};
```

Format of the map file

All map files should be .txt files, with their name corresponding with the enum specified.

```
class LevelFactory
{
    public:
        enum LevelType
        {
            LEVEL_101 = 0,
            HIDDEN_LEVEL_101,
            LEVEL_102,
            HIDDEN_LEVEL_102,
            HIDDEN_LEVEL_112,
            LEVEL_103,
            HIDDEN_LEVEL_103,

        };
```

Files should be organised into list of many types of entities.

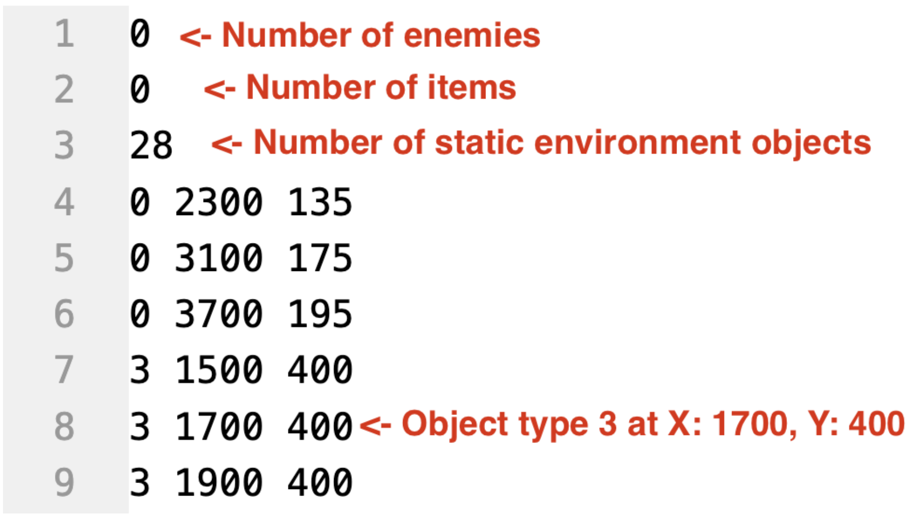
For example, the first number in the file is the number of enemies, followed by the list of enemies. It is the same for all distince type of entities (items, drawables, etc.).

Format of an entity in a list

Every enity is represented in one line.

The first number is the type of that entity. The type number is relevant with the type of the list (please use enums for reference). Following this number, there should be a pair of integer (**not float**) specifying the location.

Example:



```
0
0
28
0 2300 135
0 3100 175
0 3700 195
3 1500 400
3 1700 400
3 1900 400
```

<- Number of enemies
<- Number of items
<- Number of static environment objects
<- Object type 3 at X: 1700, Y: 400

Explanation: - This map doesn't have any enemy or item - There are 28 static environment object

Note: Normally, the location of an entity is relative to the origin of the window (which is its top left corner). However, some entity (like the warp pipe), the Y position is relative to the ground (decreasing from the 750 at the ground).

Order of entities types (list order)

You have to input lists in a specified order (shown after this part). Each list contain similar-type entites.

Order of types:

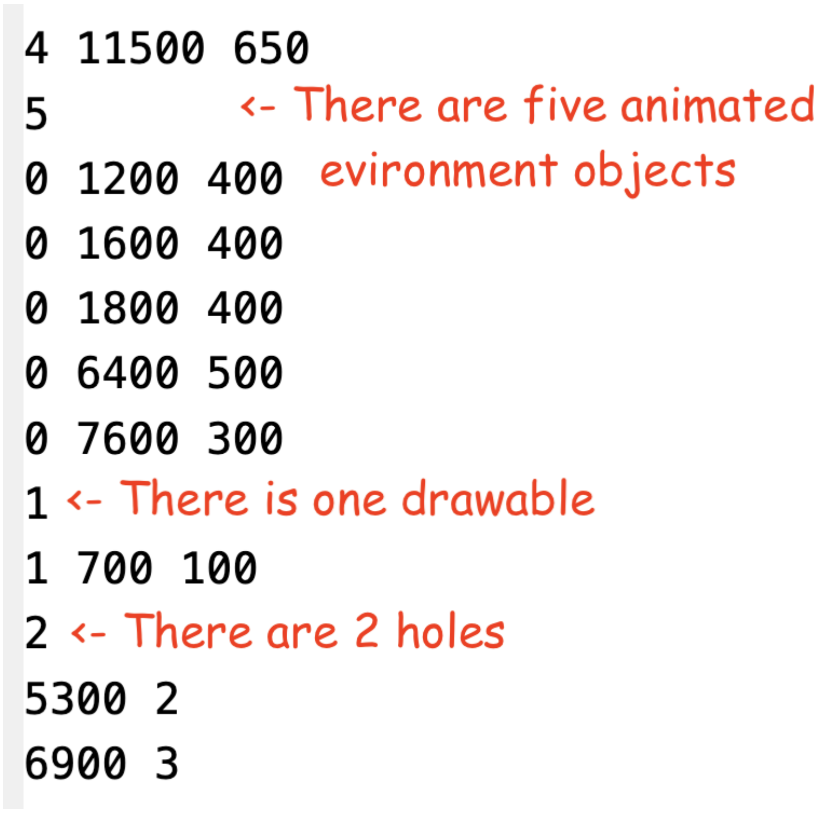
0. Start Position
1. Enemy
2. Item
3. Static Environment Object
4. Animated Evironment Object
5. Drawable
6. Hole
7. Lift
8. End Pipe

9. Type of Level
10. End Flag

Start Position

Simply input the XY coordinate where you want Mario to spawn at.

Example:



```
4 11500 650
5         <- There are five animated
0 1200 400  evironment objects
0 1600 400
0 1800 400
0 6400 500
0 7600 300
1 <- There is one drawable
1 700 100
2 <- There are 2 holes
5300 2
6900 3
```

Figure 1: Example_2

Explanation: - There are 5 animated environment objects - All of them are type 0 (QUESTION_BLOCK) - There is 1 drawble (the type is CLOUD) - There are 2 holes

Note: Holes don't need any type; you just need to specify the X position and the width (e.g., 2 is 2 blocks, which is 2 100 pixels wide). Lifts only need X and Y coordinate.*

Animated Environment Object

This is the question blocks or the breakable block in the map.

Explanation for question blocks: 0 1200 400 0. This means that the object is type 0 (question block), followed by its coordinate. After that it's the enum of the item type (0 for coin, 1 for magic mushroom).

For breakable block, there is no item so you should input number 0 after the coordninate.

Lift

With lifts, you do not need to input the type, just the coordinate

End Pipe

```
enum EndPipeType
{
    TOP,
    SIDE
};
```

Type of level

After the list of entities, you should input the type of level.

```
class Level
{
    enum WorldType
    {
        OVERWORLD,
        UNDERGROUND
    };
```

End Flag

If there is a flag, input 1, followed by the X coordinate. Else, input 0. You can only have maximum end flag per level. The flag is always set to be on ground level.

Example:

```
1 10000
```

Explanation: The flag is at X coordinate 10000.

Example:

```
0
```

Explanation: There is no end flag.