# World of Nenn Contribution Guide

## Adding a new NPC type

### Step 1 – Editing the code

In the file “EntityType.h” you can find the table that holds all the IDs for entity types. Each type of Entity must have a unique integer value. The ID that is assigned to the entity must never change. The 16-bit ID means there can be up to 65,536 different kinds of entities.

enum EntityType : u16

{

ET\_ADMIN = 0,

ET\_PLAYER = 1,

ET\_RAT = 2,

ET\_PIG = 3,

ET\_COUNT

};

Leave ET\_COUNT at the end and do not assign it a value.

### Step 2 – Adding a sprite

Then you have to add a new sprite for the EntityType. Just make a 16x16 sprite with a transparent background that looks like the npc and place it in “assets/graphics/entities/”. Since rats are entity type 2, their sprite is named “entity\_2.png. The engine will find it automatically.

### Step 3 – Editing EntityData

Open the data file “assets/data/EntityData.csv”. This holds the text that the player sees in game. The description is printed in the chatbox when you choose to inspect the entity.

|  |  |  |
| --- | --- | --- |
| Type | Name-en | Description-en |
| 0 | Admin | A god among all men. |
| 1 | Player | A human. |
| 2 | Rat | A rat. |
| 3 | Pig | A pig. |

## Placing an NPC

You can use the file “assets/data/EntityInstanceData.csv” to place NPCs in the world. For reference, players always spawn around (115,115).

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Id | Type | Level | BoundsX | BoundsY | BoundsW | BoundsH | RespawnTime | Hostile | Note |
| 0 | 2 | 1 | 105 | 110 | 3 | 3 | 15 | 0 | Rat1 |
| 1 | 2 | 1 | 107 | 112 | 3 | 3 | 15 | 0 | Rat2 |
| 2 | 3 | 3 | 110 | 116 | 3 | 3 | 25 | 1 | Pig1 |
| 3 | 3 | 3 | 113 | 119 | 3 | 4 | 25 | 1 | Pig2 |

Type is the EntityType. You must use the integer value from the table. Level is the level of the npc. Bounds X, Y, W, H are the position and size of the rectangle the npc will wander in. RespawnTime is the time in ticks that it takes for the entity to respawn. There are 6 ticks every second. If Hostile is 1, the npc will attack nearby players that are less than 10 combat levels above their own level. Note is just notes…

# Adding a new item

### Part 1 – Editing the code

Adding an item is very similar to adding a new entity. You can find the table in “ItemType.h”.

enum ItemType : u32

{

ITEM\_BONES = 0,

ITEM\_COINS = 1,

ITEM\_RAW\_RAT\_MEAT = 2,

ITEM\_COOKED\_RAT\_MEAT = 3,

ITEM\_COUNT

};

Leave ITEM\_COUNT at the end and do not assign it a value.

### Part 2 – Editing ItemData

“assets/data/ItemData.h” contains a lot of important data that must be added.

|  |  |  |  |
| --- | --- | --- | --- |
| Item | MaxStack | Name | Description |
| 0 | 1 | Bones | A pile of bones. |
| 1 | 0 | Coins | Some coins. |
| 2 | 1 | Raw Rat Meat | Cooking it will barely help. |
| 3 | 1 | Rat Meat | You probably shouldn’t eat this. |

Item is the unique ID. MaxStack is the max quantity of the item that can be held in a single slot. 1 means the Item can’t stack and 0 means that there is no limit. Any other value is just the max stack size.

### Part 2 – Adding a Sprite

The same process for adding an entity sprite. For example the sprite for coins is “assets/graphics/items/item\_1.png”.

# Editing drop tables

For now, the code to edit drop tables is in C++, so it might be a little difficult for non-programmers to pick it up. Just try to stick to the patterns and a programmer can always tidy it up later. The code for this is in “LootTable.cpp”

If you want to add an item that always drops, use addAlwaysDrop().

If you want varying drops, add a drop table. Drop tables have a list of items and their

probabilities of being dropped. Only one item from the drop table is selected. Make sure

add the drops with a higher drop rate first. The arguments for LootEntry are:

LootEntry(item type, drop rate, quantity, range)

The quantity of loot dropped is random from quantity to quantity + range. If you want to drop between 10 and 15 gold you can use:

LootEntry(ItemType::IT\_COINS, DR\_QUARTER, 10, 5)

You can use these values for drop rates or pick your own by hand

#define DR\_ALWAYS 8192

#define DR\_HALF 4096

#define DR\_QUARTER 2048

#define DR\_EIGTH 1024

#define DR\_COMMON (DR\_ALWAYS / 16)

#define DR\_UNCOMMON (DR\_ALWAYS / 32)

#define DR\_RARE (DR\_ALWAYS / 128)

#define DR\_SUPER\_RARE (DR\_ALWAYS / 256)

#define DR\_MEGA\_RARE (DR\_ALWAYS / 2048)

#define DR\_MIN 1

For example, of you want a rat to have a 50% chance to drop bones and a 25% chance to drop meat, the code would look like this...

addNewLootTable(EntityType::ET\_RAT, new LootTable({

LootEntry(ItemType::ITEM\_BONES, DR\_HALF, 1, 0),

LootEntry(ItemType::ITEM\_RAW\_RAT\_MEAT, DR\_QUARTER, 1, 0),

));

There also premade loot tables and loot entries in LootTableValues.h that you can use and add to.