Data

- path.data
- aipolicy.data
- domain.data
- dyn_tasks.data
- dynamicobjects.data
- elements.data
- gshop.data
- screentasks.data
- task_npc.data
- tasks.data

path.data

`path.data` is fairly simple. It contains a mapping of ID's to asset paths.

The file starts with 4 bytes representing a `timestamp` in unix epoch and 4 bytes representing the total number of paths in the file.

Each path consists of a varying number of bytes. The first 4 bytes of a path are the ID, the next 4 bytes are the length of the path in bytes. Reading bytes equivalent to the length will give you a `GB2312` encoded string representing the path.

Example snippet of a `path.data` export.

```
1,Models\npcs\□□\□□□\□□\□□.ecm
2,Models\npcs\□□\□□□\□□□□□\□□□□□.ecm
3,Models\npcs\□□\□□□\□□□\□□□.ecm
4,Models\npcs\□□\□□□\□□□□□\□□□□□.ecm
5,Models\npcs\□□\□□□\□□□□\□□□□.ecm
6,Models\npcs\□□\□□□\□□□\□□□.ecm
7,Models\npcs\□□\□□□\□□□\□□□.ecm
```

An example C# class representing the data.

```
public class PathData
{
□public Int32 Timestamp { get; set; }
□public Int32 TotalPaths { get; set; }
□public ESOPath[] ESOPaths { get; set; }
}

public class ESOPath : PathData
{
□public Int32 ID { get; set; }
□public Int32 PathLength {get; set;}
□public String? FilePath { get; set; }
}
```

aipolicy.data

domain.data

dyn_tasks.data

dynamicobjects.data

elements.data

gshop.data

screentasks.data

task_npc.data

tasks.data