Creating custom file lists and sub-mods for the dummy mod

Table Of Contents

1 File lists

If you encounter an error message, indicating that the game cannot find a certain file and there is no sub-mod available for this file, the easiest way to get your savegame working, is a custom file list.
This list has to be stored in text-file named `userList.lua`, either directly in the main folder of the mod (where the mod.lua is located) or in the subfolder `res/scripts` (it possible to use different files in both locations if you like). The content of the file should look like this:
Code: userList.lua

```
1. local modelList = {
2. {
3. fileName = "res/models/model/vehicle/train/br_120_0.mdl",
4. dummyModel = {name = "trainEngine", length = 19.2, seats = 0}
5. },
6. }
7. return modelList
```

You should only change the lines 1 and 6-7 if you know what they do. The block from lines 2-5 can be used multiple times (one for each file you want to create, all of them before line 6 and 7).

Line 3 defines the name of the missing file. You can find it in the error message:
[box]File: res/models/model/vehicle/train/br_120_0.mdl

cannot open res/models/model/vehicle/train/br_120_0.mdl: No such file or directory

This error is usually caused by modding. The Syntax of some game resources is not correct.[/box]
or, after the game has been closed, at the end of the stdout.txt:
[box]`error in file res/models/model/vehicle/train/br_120_0.mdl: cannot open res/models/model/vehicle/train/br_120_0.mdl: No such file or directory`[/box]
You'll have to make sure that the folder where the file is stored already exists in the mod. Unfortunately it's not possible to create folders with a Lua script. In this example the mod should have a subfolder `res` with another subfolder `models` and so on, to the folder `train` (these specific folders are already present in the mod, but for more "exotic" objects, some folders may be missing).

Line 4 supplies some informations about the type of the missing file. Unfortunately there is no easy way to find them (since the file where you could find them no longer exists), but they are necessary in order to make the object work properly.

`name` decides which object to use for the file, the following values are valid (all other values will be ignored):

trainEngine	Train Engine (also EMUs and DMUs)
trainWagonPerson	Passenger wagons
trainWagonCargo	Freight wagons
tram	Trams
tramCargo	Crargotrams
bus	Busses
truck	Trucks
ship	Passenger ships
shipCargo	Cargo ships
person	Persons
car	AI Cars
misc	All other models (signals, depots, assets, etc.)
construction	Constructions

`length` defines the length of the model. This entry is optional and only relevant for trains, because train models that are to short or to long can also cause crashes. Therefore I recommend using values that are as accurate as possible. I'm aware that this might be difficult, if the object was modelled after a real vehicle, you could use its length.

You can use `seats` to define the amount of seats for crew and passengers. This entry is optional too and should only be used, if you know the exact amount or if the standard value is to low.

In case you need neither `length` nor `seats`, the line can be simplified to `dummyModel = "<name>"` (e.g. `dummyModel = "bus"` or `dummyModel = "shipCargo"`).

If the error message persists after you saved the file and added the mod to your savegame, check if the `userList.lua` is located in the correct folder and all folders for the missing files exist.

2 Sub-mods

You're having a backup of a deleted mod and are searching for a legal way to provide others with the necessary files, so they can still use their savegames? A custom sub-mod might be the solution.
The sub-mod follows the same rules as any other mod, therefore you will need a folder named like `<modname>_1` (<modname> may contain additional underscores). You could use the names of the existing sub-mods as a template: `dummies_<name of the original mod>_1` or use something completely different.
Inside this folder you have to create a file named `mod.lua` and the folders for all files to be created.

The content of the mod.lua should look like this:

Code: mod.lua

```
 1. local modelList = {
 2. {
 3. fileName = "<filename>",
 4. possiblePaths = {
 5. "mods/<mod_folder>/",
 6. "../../workshop/content/446800/<workshop_id>/"
 7. },
 8. dummyModel = {name = "misc", length = 0, seats = 0},
 9. },
10. }
11. local info = debug.getinfo(1,'S')
12. local modPath = string.sub(info.source, 2, string.len(info.source)-7)
13. function data()
14. return {
15. info = {},
16. runFn = function (settings)
17. if dummyModels then
18. dummyModels.util.createModels(modPath, modelList)
19. else
20. print("Could not create dummy models. It seems the main mod is missing or not loaded at this point.")
21. end
22. end
23. }
24. end
```

Display More

The block from line 2 to 9 has to be repeated for each file. `fileName` and `dummyModel` work the same way as in the custom file lists. The only new part is `possiblePaths`, where all file paths, where the file might be stored, are listed. They are used so that the dummy mod knows where to look for these files, in order to decide if they have to be created or not. Line 5 shows an example for a mod from the mod folder, line 6 for a mod from the workshop. Of course you don't necessarily need these two paths, you can also use more ore less.

The entry in line 15 contains basically the same informations as the info.lua from Train Fever. If you want to learn more about it, i suggest reading this article: [Making mods work with the in-game mod manager]

If you want to share your own sub-mod, you can do that under these conditions:

Display Spoiler