Zielanzeigen

Zielanzeigen in Transport Fever 2

Die Definition einer solchen Anzeige findet in der mdl-Datei des jeweiligen Fahrzeugs statt. Diese liegt gewöhnlich irgendwo im Ordner res/models/model/vehicle/...

Ein typischer Eintrag sieht so aus:

Code: aboag.mdl (Bus aus Vanilla-Spiel)

```
1. labelList = {
2. labels = {
3. {
4. type = "LINE_NAME",
5. transf = {
6. 0, 1, 0, 0,
7. 0, 0, 1, 0,
8. 1, 0, 0, 0,
9. 2.8254, -0.403 , 2.7219, 1,
10. },
11. size = { 0.807 , 0.333 },
12. color = {16 / 255, 16 / 255, 16 / 255},
13. fitting = "SCALE",
14. alignment = "CENTER",
15. verticalAlignment = "CENTER",
16. renderMode = "STD",
17. childId = "RootNode",
18. },
19. },
20. },
```

Display More

Es kann folgende Felder geben (Liste basiert auf meinem aktuellen Wissensstand und kann gerne erweitert werden!)

alignment (horizontale Zentrierung des Textes innerhalb des definierten Raumes)
- CENTER
- LEFT
- RIGHT

verticalAlignment (vertikale Zentrierung des Textes innerhalb des definierten Raumes)
- CENTER
- TOP
- BOTTOM

childId
"RootNode" bedeuted das es an den Nullpunkt der MDL angehangen wird
Sinnvoller ist es an dein Mesh zu hängen da es sich dann in den Kurven mit bewegt.
Dazu muss man einem passenden Mesh einen Namen verpassen und diesen dann bei childid eintragen

Code

```
1. {
2. materials = {"vehicle/train/mc_br442_1.mtl", },
3. mesh = "vehicle/train/mc_br442/mc_br442_lod_0_body.msh",
4. name = "mc_br442_lod_0_body",
5. transf = {1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1,},
6. },
```

color (Farbe des Textes)

Basiert auf RGB (Rot-Grün-Blau). Habe verschiedene Schreibweisen gesehen, grundsätzlich ist es aber immer so, dass die erste Zahl für Rot, die zweite für Grün und die dritte für Blau steht.
- Schreibweise 1: color = { 1.0, 1.0, 1.0, 0.1, } - 1.0 ist 100% der Farbe, 0.0 ist 0% der Farbe. Wenn ich also ein reines Rot will, muss also bei der ersten Zahl auf 1.0 gehen und die anderen Zahlen auf 0 setzen).
- Schreibweise 2: color = {16 / 255, 16 / 255, 16 / 255} - Farbe aussuchen und Werte eintragen: https://www.rapidtables.com/web/color/RGB_Color.html
Es gibt auch noch weitere Schreibweisen mit * drin, ich habe aber keine Ahnung, was das bedeutet. Auch ist mir unklar, was die vierte Zahl macht (die ist auch nicht immer dabei). Im Webdesign wäre diese zuständig für die Transparenz einer Farbe, das konnte ich bei meinen Tests aber nicht beobachten.

filter (es werden nur bestimmte Inhalte angezeigt)
- NUMBER - aus dem angelieferten Text (z.B. Linienname) werden nur die ersten zusammenhängenden Zahlen angezeigt, alle anderen Zeichen aber weggelassen. Das ist z.B. sinnvoll, wenn es eine eigene Anzeige nur für die Liniennummer gibt, auf der aber kein Ziel steht. Sieht man z.B. oft im Heck von Bussen.
Beispiel: Heißt die Linie z.B. "23 Betriebshof 5A" dann würde letztendlich nur "23" dort stehen, weil quasi nur die ersten zusammenstehenden Zahlen dargestellt werden, die 5 weiter hinten ist dadurch auch schon irrelavant.
- NONE - habe ich bei einer Vanilla-Straßenbahn gesehen, dann könnte man aber vermutlich auch die ganze Zeile einfach weglassen
- CUSTOM - damit lassen sich eigene Filter erstellen. Diese werden über "params" (siehe unten) erstellt

fitting (was passiert mit Text, der in der Standardschriftgröße nicht reinpasst)

- SCALE - Text wird verkleinert, damit alles reinpasst
- CUT- Text wird abgeschnitten

font
Man kann entweder keine angeben, oder eine der Schriftarten aus res/fonts/

nlines (
Number of Lines - Gibt an wieviele Zeilen die Anzeige haben kann.

params (hiermit lassen sich wohl diverse Anpassungen vornehmen, bin aber noch nicht ganz durch das System gestiegen)

@mediziner nutzt z.B. in der MGT6D-Tram folgendes:
params = {offset = 2,}, - das führt dazu, dass im Cockpit auch die darauffolgende Haltestelle (= 2) angezeigt wird. Das kann man auch weiter hochzählen, "3" wäre dann die überübernächste Haltestelle usw.
params = { expr = "([^0-9.]*)", replace = "\\1" }, --die Zahlen 0-9 werden nicht in der Anzeige angezeigt, nur Buchstaben und Sonderzeichen (danke @trunky)
params = {relative = false, offset = 0}, - hiermit kann man die erste Haltestelle der Linie anzeigen lassen (danke @mediziner & @Marcolino26)

renderMode (leuchtet der Text der Zielanzeige oder nicht)
- EMISSIVE - sie leuchtet
- STD - sie leuchtet nicht

size (die Größe der Anzeigefläche)
Hier wird angegeben wie breit und hoch die Fläche ist die zur Verfügung steht (die Werte kann man in Blender abgreifen)

transf (die Position der Anzeigefläche)
(um das ganze in einem Eintrag zu haben, danke an Marcolino26)

Vordere ZZA in Fahrtrichtung:
0, 1, 0, 0,
0, 0, 1, 0,
1, 0, 0, 0,
x, y, z, 1

Hintere ZZA in Fahrtrichtung:
0, -1, 0, 0,

0, 0, 1, 0,
1, 0, 0, 0,
x, y, z, 1

In Fahrtrichtung auf der rechten Seite:
1, 0, 0, 0,
0, 0, 1, 0,
0, -1, 0, 0,
x, y, z, 1

In Fahrtrichtung auf der linken Seite:
-1, 0, 0, 0,
0, 0, 1, 0,
0, -1, 0, 0,
x, y, z, 1

x,y,z ist dabei der linke untere Punkt der Anzeige.

type (was wird überhaupt angezeigt)

- COMPANY_NAME (Name der Firma des Spielers)
- LINE_NAME (Name der Linie)
- NAME (Name des Fahrzeugs)
- NEXT_STOP (Name der gerade angefahrenen Haltestelle)
- STATION_NAME (Name einer der Stationen der Linie, in Verbindung mit Params - wenn keine Params angegeben werden, dann die Station, die gerade angefahren wird)

Ich würde mich sehr freuen, wenn wir die Liste gemeinsam vervollständigen würden - alles, was ich hier eingetragen habe, basiert auf dem Lesen von Code der Vanilla- und Mod-Fahrzeuge und auf eigenem Ausprobieren. Das ist sicher keine vollständige Aufzählung bisher.