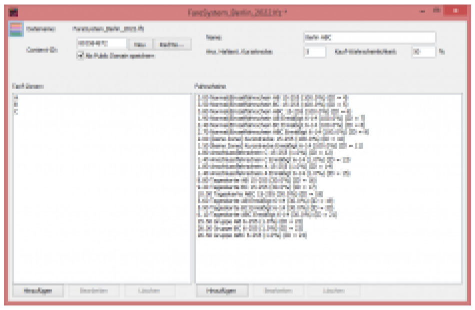
Ticket sales

Inhaltsverzeichnis

1 Creating a fare system

New fare systems are created in the ContentTool. To do this, click on "Additional Map Content" and then on "Fare Systems".

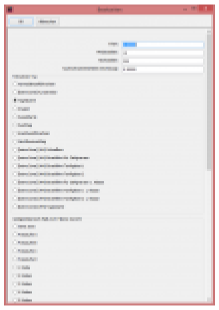


First, a few general settings can be made at the top: A plain name, the number of stops allowed using short distance, and how likely passengers are in general to want to buy tickets.

Next, both the available fare zones and the available tickets can be defined.

Fare zones currently only have the property "Name".

Tickets have the following properties:



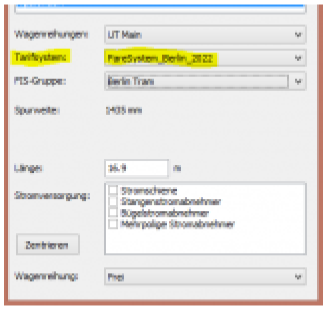
- Value: This is how much this ticket costs
- Minimum and maximum age
- Purchase probability: This is a weighting, i.e. it is not a probability percentage, but the probability of purchasing the ticket is calculated relative to the weightings of the other (available) tickets.
- Ticket type/validity zone/addition: Validity "character" of the ticket. Here, "no zone" means that the ticket type is independent of a fare zone, e.g. for a short trip. The selection under "Validity area" is therefore irrelevant.

2 Configuration of the map

2.1 Entrypoints

In order for your own vehicle to be equipped with the correct fare system (available tickets, etc.), you must set for each entry point which fare system is to be selected for a vehicle placed there. This determines which tickets are available and how, for example, the printer is to be repainted.

For this purpose, the corresponding entry point and then one of the installed/created fare systems must be selected on the left under "General configuration" and "Entrypoints":

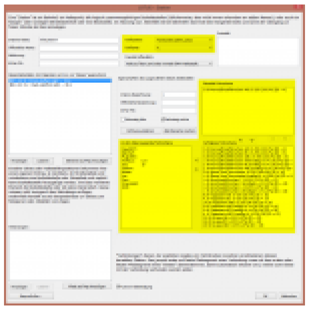


Wagenreihungen: UT Main
Tarifsystem: FareSystem_Berlin_2022
FIS-Gruppe: Berlin Tram
Spurweite: 1435 mm
Länge: 16.9 m
Stromversorgung:
- [ ] Stromschiene
- [ ] Stangenstromabnehmer
- [ ] Bügelstromabnehmer
- [ ] Mehrpolige Stromabnehmer

Wagenreihung: Frei

2.2 Stations

In order for passengers to know that they should buy tickets and which tickets they should buy, the stations must be configured accordingly. To do this, call up the properties window of the station to be configured:



- First, the station must be specified to which fare system it belongs (above).
- In addition, the station should be assigned to a fare zone. This will be needed later for any concrete calculations of the appropriate ticket orders. At the moment, however, this setting is not yet relevant.
- Now all tracks/stops are to be selected one after the other and set which tickets are purchased there on the part of the passengers:
    - At the bottom right is a list of all tickets in the fare system. Using the arrow buttons, tickets can be added to or removed from the lists "Plausible tickets" and "Tickets matching destinations".
    - Plausible tickets are those tickets that are also purchased by passengers there, e.g. the location of the stop and the fare zone must match. With "C" and "V" the plausible tickets can be copied and pasted at other stations/stops/tracks.
    - In the list "Tickets matching destinations", one or more tickets can be assigned to each of the "standard destinations". In LOTUS, passengers can then also order tickets by ordering such a "standard destination" ("A ticket to the airport, please!").

3 Vehicle

3.1 Configure vehicle

For ticket sales, two module slots must be created with predefined data, namely one each for the money given by the passenger and one for the money given by the driver.

The properties must be as follows:

- The name of the module class must be "MONEY_GIVEN" or "MONEY_CHANGE" respectively.
- The position corresponds to the center of the area where the money should be. The Z-coordinate should be slightly raised (approx. 5mm)
- The Checkbox "invisible" should be set
- Set light space and animation as usual
- do not select a standard module!

In the object properties there is also the possibility to set the variance in each case, so that the coins do not always fall exactly on the same spot.

In addition, at the corresponding path point, where the passengers should stand during the ticket sale, the check mark for "Ticket sale takes place here" must be set in the path properties.

4 Script

A large proportion of equipping a vehicle with a vending facility is "unfortunately" done by script, although it should be noted that the script blocks are primarily located in the modules for the printer and the changer.

4.1 Ticket Code

In this context, the meaning of the "Ticket Code" is important: here, the three ticket criteria ticket type/validity area/addition are coded in a number, the ticket code: `ZTTGGG`, where the individual letters stand for the respective criteria. Here, the list index of the respective criterion is to be used (as seen in the ContentTool, zero-based counted). Thus, a discounted day ticket for 7 combs has the coding 1002011.

4.2 Ticket return

In order for any module or the actual vehicle to tell LOTUS which ticket is to be "presented" (so that the user can take it), the module/vehicle sends the broadcast "SetTicket" in the BusId "TICKET" with the desired ticket code.

For its part, LOTUS sends the broadcast "RemoveTicket" in the same BusId when the passenger has taken the ticket, so that the ticket can also be visually removed from the vehicle/module.

The following script procedures are also available for printer operation:

- `procedure FareSystemGetContentID(self: integer; var user, sub: integer);')` - Determines the ContentID of the fare system that was assigned to the vehicle when it was placed (via the entry point properties).
- `function FareGetValue(self: integer; fareCode: integer): single;` - get the value of the ticket coded with "fareCode", e.g. to show the amount of money on the display when you have selected a ticket.

4.3 Money

To "give" the passenger (change) money, send a single broadcast also on BusId "TICKET" with the name "AddChangeMoney", sending the value of the coin as the value.

For its part, LOTUS sends the following integer broadcasts, also on the BusId "TICKET", which can be used in particular to play the corresponding sounds:

- "MoneyGiven": The passenger put his money on the pay table
- "RemoveGivenCoin": The user has removed a coin put down by the passenger by clicking on it (e.g. in the changer).
- "RemoveChangeCoin": The user has taken back a coin that was ejected by the user by clicking on it.

4.4 Individualisation of the ticket printer

By means of a few new options, the ticket printer can be highly adapted to the fare system. In the case of the Almex printer, there are two crucial points:

- The assignment and labelling of the buttons with specific tickets and
- the ticket names as they should appear in the display.

The procedure is explained here using the Almex printer and the "Berlin 2022" tariff system as an example. For this purpose, the printer is first delivered with a blank texture: Drucker_Almex_Blanco.bmp & Drucker_Almex_Blanco_alpha.bmp. In addition, the corresponding individualised texture is imported and the ContentID is noted (1000:186513005): Drucker_Almex.bmp.

Now a text file is created in the export folder which contains the individual configuration data for the special connection "Almex printer" + "Berlin 2022". This is given the file name "#4000#FareForAlmex_Berlin_2022.txt". It is important to note that the beginning must be correct - 4000 is the ContentSubID of this text file (but this is not important later) - but otherwise it does not matter how it is named. It is then imported using the function "Text File" in the ContentTool.

It is structured like this:

Code

```
1000:893584872#ALMEX
1000
186513005
Kurzstrecke
1000
Kurzstr. Erm.
1001000
BWC 48 AB
-1
Normal AB
30
Ermäßigt AB
1000030
CTC 48 AB
-1
Anschluss Fs.
6000
Normal BC
31
Ermäßigt BC
1000031
Kurzstr. OVG
-1
Normal ABC
32
Ermäßigt ABC
1000030
2 Waben Norm.
6
2 Waben Erm.
1000006
Tagesk. AB
2030
```

37. Tk Erm. AB
38. 10002030
39. Klein GK AB
40. 3030
42. Tagesk. BC
44. 2031
45. Tk Erm. BC
46. 1002031
47. Klein GK BC
48. 3031
49. Tagesk. ABC
50. 2032
51. Tk Erm. ABC
52. 1002032
53. Klein GK ABC
54. 3032
57. BWC 48 ABC
58. -1
59. CTC 48 ABC
60. -1
61. BWC 72 AB
62. -1
63. Potsd. Nor AB
64. -1
65. CTC 72 ABC
66. -1
67. CTC 72 AB
68. -1
69. BWC 72 ABC
70. -1
71. Großgem. Norm.
72. -1
73. Potsdam Kurz.
74. -1
75. Kurz OVG Erm.
76. -1
77. Großgem. Erm.
78. -1
79. BBT
80. -1

Alles anzeigen

Important: The structure of the text file can be defined completely arbitrarily by the ticket printer developer! Although the first line plays a special role, this "freedom" applies to all lines!

In the case of the Almex printer, the first line is structured like this: [UserID:SubID#ALMEX], where UserID and SubID refer to the tariff system to which the vehicle is currently configured. This first line is very important because it allows the script to find such a text file. The script contains the following lines:

Code

```
FareSystemGetContentID(Self, user, sub);
TextFileGetContentIDByTag(IntToStr(user) + ':' + IntToStr(sub) + '#ALMEX', FContentIDUserFareText, FContentIDSubFareText);
```

The first step is to determine which tariff system is currently assigned to the vehicle and write the values in user and sub.

Then the ContentID of the text file is determined, which "fits" here. For this purpose, the first line is "assembled" and the user and subID are determined with the command. LOTUS uses this assembled string and compares it with the first line of all existing text files (don't worry, not all text files are loaded separately for this! 😉).

If the "Berlin 2022" tariff system is selected in the example, then user = 1000 and sub = 893584872 are determined via the first line. From this, the string "1000:893584872#ALMEX" is assembled - and voilà: When searching, it will find the very text file we just created and write the two parts of the ContentID.

Once this has been prepared, the content of the text file can be read line by line. The following function is available for this purpose:

```
function TextFileGetLine(contentUserID, contentSubID: integer; line: integer): string;
```

We now have the possibility to access any text file via script depending on the set fare system, whereby this can be designed individually for the device as well as the fare system.

The following structure is (also) arbitrarily chosen; in this respect, the structure of the file is only interesting insofar as one either wants to adapt the ticket printer to another fare system, or simply as inspiration as to how the structure could look.

The first line is followed by the UserID and the SubID of the exchange texture, so that the buttons are labelled accordingly. This is followed by a blank line and then, always in pairs, the display text and ticket code of the respective key. Since three levels can be switched through by means of "Shift", there are three blocks with twelve double lines each, separated by two blank lines each.