

# MANUAL DE USO DEL LOCOTOOL 0.4.3b

## Introducción:

Esta pequeña herramienta permite decodificar los archivos .dat que usa el Locomotion contenidos en el directorio ObjData.

El programa se usa desde una ventana de MSDos si queremos incluir algunos parámetros al decodificar el archivo.

## Parámetros que permite:

- p <num> : colores de empresa.
- d : crea un archivo .dat para la versión demo.
- o <path> : establece el nombre base para los archivos png.
- q : no informa de errores.
- s : coloca todos los sprites en un único archivo png.
- u : escribe todas las entradas <unknown> incluso si son 0.
- b : omite los bits que no se han establecido.

## Forma de uso:

Para usarlo desde la línea de comando tiene el siguiente formato.

Locotool [opciones] <.dat | .xml>

También permite usar el programa desde windows sin opciones. Para ello, si queremos decodificar un archivo .dat, sólo tenemos que cogerlo y arrastrarlo hasta el archivo locotool.exe y seguidamente lo soltamos. Esta acción generará un archivo .xml y un directorio que contiene todos los sprites usados por el objeto. El nombre del directorio y del xml que se han creado tienen el mismo nombre que el del archivo .dat.

Para modificar el objeto sólo tenemos que hacer clic con el botón derecho del ratón sobre el xml y elegir la opción de editar.

Una vez hecho las modificaciones, para codificar de nuevo el archivo .dat, arrastramos el archivo .xml hasta el exe como hicimos anteriormente para decodificarlo. Si el fichero .dat existe en el directorio, se creará un archivo .bak del original.

## Modificar un objeto conservando el original:

Para crear un objeto sin perder el original, debemos encontrar en el archivo todas las entradas donde hace referencia al nombre del objeto.

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<object class="0x97" subclass="0x118280" name="R470REG "><chunk compression="1">
  <variable name="class" size="1">0</variable>
  ...
  ...
  <bit name="copy">0</bit>
  <pngfile>C:\\Documents and Settings\\JJLOR\\Mis documentos\\Loco\\loco\\Trenes\\
R470REG\\R470REG\\000.png</pngfile>
  </sprite>
  <sprite id="1" xofs="-120" yofs="-120">
    <bit name="hasdata">1</bit>
```

...  
...

Para hacer esto de forma sencilla, desde el bloc de notas cuando editamos el xml, vamos al menú Edición/Reemplazar.

En el diálogo introducimos el nombre original tal y como aparece en el archivo original en el cuadro de texto superior y en el de abajo el nombre nuevo que le queremos poner.

**NOTA:** El nombre debe estar en mayúsculas y tener 8 caracteres justo, es decir, si tiene menos, añadiremos espacios al final. Sólo deben añadirse los espacios donde se haga referencia al nombre interno del objeto, no en la ruta de los sorites.

Por ultimo le damos al botón de reemplazar todo.

Es conveniente mirar que el nombre interno este correcto y tenga 8 caracteres exactos. De no ser así, al codificar de nuevo el .dar, dará un mensaje de error.

También es importante renombrar el nombre del fichero .xml y del directorio que contiene los sorites.

### Lista de variables conocidas:

#### Common

**auxdata[range]** - Unknown

**basecostfact/buildcostfact** - The cost of a bridge/track

**costfact** - Purchase cost factor (final cost = basecost[costind]\*costfact)

**costind** - Purchase cost index (index into a table of base costs), lower means higher final costs

**trackpieces** - What trackpieces this track, trackmod, roadmod or train station has

**description** - The description of the object in various languages

**designed** - Year object is designed, 0 if available from the beginning

**numaux01** - number of entries in aux\_0 and aux\_1

**numaux02ent** - number of aux\_2 array entries

**numaux03ent** - number of aux\_3 array entries

**numaux04ent** - number of aux\_4 array entries

**numaux05ent** - number of aux\_5 array entries

**nummods** - Number of mods required (vehicles) or buildable (for track)

**obsolete** - Year object becomes obsolete, 65535 if never

**roadpieces** - Same as trackpieces but ONLY has smallcurve, tightcurve, normalslope, steelslope and reverse (for tram), and only for roads, and road stations

**salecostfact/sellcostfact** - sale/sell cost factor of bridge/track (final cost = basecost[costind]\*salecostfact)

**sprite(range)** - Individual sprites

**stringtable(range)** - A range of strings, mainly used for descriptions of objects

**trackmod(array)** - Array of trackmods required for vehicle to run, or trackmods available in the case of tracks/roads

#### Class 00 - Interface

**field\_0[6]** - Game Object Tooltip - first line: Object name.

**field\_0[7]** - Game Object Tooltip - next lines: Object data/Action info.

**field\_0[8]** - UI Tooltip (button info)

**field\_0[9]** - Action popups, "Can't build/remove ...", "Screenshot saved ...".

**field\_0[10]** - Win-Tabs: **Player Items** - Build New Vehicles, Vehicles, Stations, Player Status, Messages.

**field\_0[11]** - Win-Top - most non-player related windows.

**field\_0[12]** - Win-Tab: **Map Objects** - Towns, Industries, Scenario Editor - Object Selection.  
**field\_0[13]** - Win-Tab: **Player Construction** - Build- Tracks/Roads, Docks/Airport.  
**field\_0[14]** - Win-Tab: **Map Construction** - Construction -land, Scenario-Generator/Challenge.  
**field\_0[15]** - Win-Tab - Overview Map.  
**field\_0[16]** - Win-Tab: **General** - Options, About.  
**field\_0[18]** - main buttons 1: File, Sound.  
**field\_0[19]** - main buttons 2: Zoom, Rotate, View.  
**field\_0[20]** - main buttons 3: Construction-Land, Tracks, Roads, Docks/Airport, Build Vehicles.  
**field\_0[21]** - main buttons 4: Vehicles, Stations, Towns, Industries.  
**field\_0[22]** - main buttons 5: Player Status, Performance.  
**field\_0[23]** - main buttons 6: Time, Messages, Map.

Values:

- Palette-numbers 0-30, those are the 31 internal used color gradient sets.
- Other values are excepted but have no real use, and can give strange side effects.
- Value/order of palette sets -> player-status color-picker(LR/UD)

Linked to PlayerColor 1:

- (Win-top) **Player Items & Player Construction**.

Not Editable:

- Scenario Editor: [Back to Previous] & [Forward to Next] navigation buttons.
- Intro screen: Big green buttons.
- Warning message windows: "Are you sure ...", "Vehicle can't ...", etc.
- File windows: Load, Save, New game.
- Two player connection window.

---

## Class 02 - Currencies

**shiftnum** - How many times to multiply pound value by 2

**zeroes** - How many zeros after pound value

---

## Class 05 - Water

Its as high as the Water-Level set in the map/scenario editor and absolutely flat in the beginning.

**field\_0[2]** = "2" (might be default water level)

**field\_0[4]** = "102" (might be ingame level changing price)

## Class 08 - Cargo types

**decay1days** - Duration of first decay rate

**decay1rate** - First decay rate

**decay2rate** - Second decay rate

flags

**bit\_2** - Unknown

**refitoption** - Does refitting a vehicle allow it to carry this cargo type?

**paymentfactor** - Payment factor

**paymentind** - Payment index

**peakdays** - Days at which payment remains at maximum

**unitweight** - Weight per unit in 1/256 tons

---

## Class 09 - Fences

**Field\_0[6]** is always "15"

**Field\_8** is "2" or "5" (might set the height in 1/16 units)

**Field\_9** is always "8"

---

## Class 0A - Signals

**field\_0[2]** - country signal built style + ?

**0** - build signal on left side of track (british style)

**1** - build signal on right side of track (american/european style)

**2** - ?

**4** - ?

**field\_0[5]** - number of signal frames (4,7,10) -> signal states (2,3,4), frames = (states\*3)-2

input value seems to be fixed, other values where not accepted (locomotion loading lockup)

signals have 8 directional images per frame group.

---

## Class 0C - Street Lights

**year[array]** - The year the street lights get build

---

## Class 0E - Bridges

**disabledtrackcfg** - Decides what types of corners and slopes the bridge can be used for

**heightcostfact** - Cost factor for bridge height

**maxheight** - Maximum height for the bridge, in feet

**maxspeed** - Maximum speed for the bridge, in mph

**norooft** - Does the bridge have a roof? (?)

**pillarspacing** - The space between the bridge pillars

**spanlength** - Length of a bridge element, the BrickBridge has 1, the Steel Arch 2, the Steel Girder 2, the WoodBridge 1, the Suspension 4 parts to have one complete bow.

**field\_9** - The space between the bridge columns (153 Steel Arch, 153 Steel Girder, 2550 Brick, 24 Suspension, 255 Wood Bridge )

## Class 0F - Train Stations

**auxdata[range]** - positions of cargo icons on stations

**field\_0[12]** - Bit 0, use company colored transparency effect (like Passenger Terminus)

auxdata: train stations (TRSTATx) (32 arrays \* 57 fields)

- positions of cargo/passenger on train stations, probably same use in road stations (RDSTATx/RDSTATLx)
- the arrays are grouped in sets of 4, every set covers one direction (rail build-direction),

- 1 array per station tile.
- station cargo arrays are linked to the on-screen orientation of the track build-directions.  
**aux\_0[0,1,2,3]** = BottomLeft -> TopRight (45')  
**aux\_0[4,5,6,7]** = TopLeft -> BottomRight (135')  
**aux\_0[8,9,10,11]** = TopRight -> BottomLeft (225')  
**aux\_0[12,13,14,15]** = LowerRight -> UpperLeft (315')  
**aux\_0[16;31]** = not sure, probably reserved for diagonal(map) stations (trains only)
- every array contains offset values for one station/tile  
array fields are grouped in sets of 4 fields, (57-1)/4=14 icons per station  
**aux\_x[0]** = general icon height offset for that array (screen +Y offset)  
**aux\_x[+1,+2]** = X + Y offset for cargo(non passenger) icon (offset is map orientated)  
**aux\_x[+3,+4]** = X + Y offset for passenger icon (offset is map orientated)

---

## Class 10 - Track mods

**isoverhead** - 1 for catenary like look, 0 for extra rail-like look

---

## Class 11 - Tracks

trackst.dat and trackng.dat (the object tree for \*.dat object files seems to start here, for a lot of objects being linked to in it and those link to their subobjects...)

**displayoffset** - Vertical offset of vehicles in list windows (e.g. train list, new vehicle list, ...)

**bridge[array]** - Array of bridges compatible for the track

**curvespeed** - Speed of vehicles in curves of track (not mph)

**numbridges** - Amount of bridge objects for the track

**numsignals** - Amount of signal objects for the track

**numstations** - Amount of stations objects for track

**signal[array]** - Array of signals available for the track

**station[array]** - Array of station objects for track

**stationtrackpieces** - What trackpieces supports a station

**tunnel** - Tunnel object for track

**tunnelcostfact** - Tunnel cost factor (final cost = basecost[costind]\*tunnelcostfact)

**field\_3[array]** - [0] has something to do with disallowing junctions

---

## Class 12 - Road stations

---

**Field\_B[0]** - Describes the function of the road station, per below.

##	Cargo	Where Built	Vehicles
--	-----	-----	-----
0	All	Any straight	(B)us, (T)rucks
1	All	Any straight	B,T
2	Passenger	Any straight	B,T, Trams**
3	Passenger	Any straight	B
4	Freight	Any straight	T
5	Freight	Any straight	T
6	X*		
7	X*		
8	All	Ends of roads	B,T
9	All	Ends of roads	B,T
10	Passenger	Ends of roads	B
11	Passenger	Ends of roads	B

```
12   Freight      Ends of roads      T
13   Freight      Ends of roads      T
14   X*
15   X*
* These values cause non-fatal crashes when loading scenario
** Possibly related to field_29 of trams?
```

## Class 14 - Roads

**Field\_0[2]** has is highest value in ROADRTRAM.dat "127" and is around "95" on all others.

**Field\_0[4]** unknown, similar to Field\_0[12] and Field\_0[13]

**Field\_0[6]** unknown, similar to Field\_0[12] and Field\_0[13] (inverted)

**Field\_0[7]** is always "255"

**Field\_0[8]** is "55" in general but "22" in ROARDONE.dat, its also used in Train Stations "249" and train Signals "253" (might set some directional behaviour).

**Field\_0[10]** is "1" in general, and "1" in all Train Stations and Train signals, too (this must be important).

**Field\_0[12]** might set the maximum speed or "friction", its "144" in general, "30" in ROADGH.dat and "200" in ROADTRAM.dat

**Field\_0[13]** is "1" in general and not set in ROADRGH.dat or ROADTRAM.dat (might set if the road is NOT upgradeable by a city)

**Field\_0[18]** is "220" or "221" in general but "2" in ROADTRAM (might set the ability/will to "overtake" a vehicle on the road infront of you)

**Field\_0[19]** is "1" in general and not set in ROADTRAM and ROADRGH , same as field\_0[13]

**Field\_1E[7]** is only used by ROADTRAM.dat and set to "1"

**Field\_15[7]** This indicates how many stations a road can have, all the current codes are 7 or 1.

**Field\_29[0]** is only used by ROADTRAM.dat and ROADUS1.dat and set to "2" , this smashed my dreams of unfolding the "secret of trams".

**Field\_29[5]** is "2" in every road but not set in ROADRGH.dat (might set if the road is an upgrade of ROADRGH.dat)

---

## Class 15 - Airports

**allowedplanetypes** - What plane types are allowed to land

**numspritesets** - Number of sprite sets (sets of four sprites not counting the UI icon) included in the file

**numtiles** - Number of tile definitions

**2x2tiles** - Bit mask of tiles that are 2x2 (note, bit numbers are in hex!)

**minx, miny, maxx, maxy** - Extent of airport grid

**numnodes** - Number of nodes (vertices) in aircraft movement pattern

**numedges** - Number of edges connecting nodes in aircraft movement pattern

**field\_B6** - Bit mask of edges from entrance node to terminals (see Aircraft Movement Graph)

**spriteheight** (auxdata) - defines height of all spritesets

**height[array]** - Height of the spritesets

**spriteanimframes** (auxdata) - number of animation frames of the spritesets

**frames[array]** - Number of frames, must be same for all frames of an animation

**tile[array]** (auxdata) - tile definitions for airport Layout

**spriteset[array]** - list of spritesets to use on this tile

**layout** (auxdata) - place tiles for airport Layout  
**node[array]** - node definitions for Aircraft Movement Graph  
**edge[array]** - edge definitions for Aircraft Movement Graph

---

## Class 17 - Vehicles

Alphabetical list of all vehicles (\*.dat file names)

**colourtype** - Which colour set to use if e.g. passenger vehicles have different colours

**cargo** - The cargo the vehicle can carry

**class** - What does it run on, 0=rail, 1=road, 2=air, 3=water

**compatible[array]** - Array of all compatible objects

**field\_0[16]** - The ID of the cargo, used in vehicles

**field\_10E** - Position of the Smoke or Steam (value sets height).

**field\_3** - Controls in which tab the vehicle is when you want to buy it

**0** - Trains

**1** - Busses

**2** - Trucks

**3** - Trams

**4** - Planes

**5** - Ships

**field\_24**

**length** - Length of the sprites (?)

**spriteind** - Index of the sprites (0-3 + [128 if reversed])

**field\_5** - Effect position (spark, exhaust plume, etc.)

**field\_B4[0]** - Something about the bogey (rotation?)

**field\_B4[1]** - Height of vehicle above track (not the bogey)

**flags**

**anytrack** - Can it run on any type of track? (used by buses and trucks)

**cancouple** - Specifies if the vehicle can couple

**dualhead** - If the vehicle requires two locomotives (for example, the TGV)

**noannounce** - If the news announcement disabled when this vehicle is invented

**rackrail** - Can the vehicle use a rack rail?

**refittable** - If the vehicle is refittable (mostly planes and passenger boats) (Refitting allows only mail, food, goods, and grapes. New capacity is passenger capacity divided by 2.5)

**numcompat** - Number of compatible objects, 0 if compatible with all

**numsnd** - Number of sound effects (the kinds are set in the required object list)

**power** - Amount of power the vehicle generates, in horsepower (doesn't seem to have any effect on planes or ships)

**rackspeed** - Top speed while on rack rail, or landing/holding pattern speed for planes

**reliability** - Reliability of the vehicle, higher values allow a longer operating lifetime

**runcostfact** - Running cost factor (final cost = basecost[runcostind]\*runcostfact)

**runcostind** - Running cost index (index into a table of base costs), lower means higher final costs

**soundeffect[array]** - Array of soundeffects of the vehicle

**speed** - Top speed, in mph

**sprites(array)** - Sprite properties

**startsnd** - The sound the vehicle makes when it starts

**startsndtype** - How to play the starting sound (what kind is set in the required object list)

**tracktype** - Track object required for the vehicle to run

**visfxtype** - How to draw the visual effect 'visualeffect'

**visualeffect** - The object used as visualeffect

**wakefxtype** - How to draw the wake effect 'wakeeffect' (doesn't exist in locotool 0.2)

**weight** - Weight of the vehicle, in tons

---

## Class 18 - Trees

**height** - Height of the tree in 1/16th height units

**field\_4[2]** - Number of tree viewpoint images (rotations) [1,2,4]. *n1*

**field\_4[3]** - Number of tree size images (growth) [1...8]. *n1*

**field\_4[4]** - (Bit-0) Tree has double season set - used for snow graphics (should realistically only go on coniferous trees)

**field\_4[56]** - (Bit-flag) Tree states, seasons + dying

Bit 0 - Autum

Bit 1 - Winter

Bit 2 - Spring

Bit 3 - Summer (default)

Bit 4 - Dying

Bit 5/7 - (not used/ignored)

*n1*) locked/protected/other values disable object.

---

## Class 1A - Climates

**firstseason** - The season that the year starts with, [0,1,2,3]. *n1*

**seasonlength[array]** - Number of days in the four seasons.

seasonlength[0] <- first season, set by firstseason.

seasonlength[3] -> after this one, its firstseason again until end of year.

**field\_7[0]** - Low snow level, starts at begin of Winter season. *n2*

**field\_7[1]** - High snow level, starts at begin of Spring season. *n2*

*n1*) locked/protected/other values disable object.

*n2*) High >= Low, or object will be disabled.

Notes:

- seasons: Autum(0) Winter(1) Spring(2) Summer(3), 3 -> Australia/Southern hemisphere climates.
- snow traveling speed: 4 days per map level.
- field\_7 value = (Level - 1) x 4 Where level is counting up from the lowest ground level (lowest ground is 0)

## Class 1B - Hill Shapes

**field\_0[2]** - number of hill height maps

**field\_0[3]** - number of mountain height maps

Notes:

- Last image is UI-icon for object-selection(SE)
- hill shapes will become a selectable option in object-selection(SE), if hill-dat files > 1



- max HF-image size 255x255. (ok, 256x256, but thats with some extra ribble effect)
  - field\_0[12] - bit 1 ... looks buggy to.
- 

## Class 1C - Buildings

**cargo[array]** - The cargo object it produces/accepts

flags

**bit\_0 = 1** Use 2x2 tile size

**bit\_1 = 1** Include under "Miscellaneous Buildings"

**ishq** Is company headquarters building

**clearcostfact** - Cost factor to clear the building

**numproduce[array]** - Amount of first and second cargo produced. Maximum value of 255 = approx. 12 units/mo. Exceeding 255 is converted to mod[256], so increasing this number beyond 255 does not seem to help. Also, numproduce[0] is the amount added to the city population.

**numaccept[array]** - Units of 1/8 cargo acceptance for the four cargo types

**useobject desc="cargo[array]" class="8">** - First two array elements are types of cargo generated. All four types are accepted.

**aux\_0, num=XX** XX=Number of .pngs / 4

**<unknown name="field\_0[0]" size="1">1</unknown>** not sure what this value means, but seems to increase in order of height?

**aux\_1**

**<unknown name="field\_0[X]" size="1">1</unknown>** same as aux\_0, except increments of x are 2x the aux\_0 values, and the value is always 1

**aux\_2[X]** X=Number of different appearences for this building

**<unknown name="field\_0[1]" size="1">1</unknown>** value is the number of png tile to use for level 0, as defined in aux\_0

**<unknown name="field\_0[2]" size="1">2</unknown>** value is the number of png tile to use for level 1, as defined in aux\_1

one entry for every storey of building height

---

## Class 1E - Industries

**firstyear** - First year it can appear

**lastyear** - Last year it can appear

**field\_0[188]** - Minimum number of buildings within industry area

**costind** - Index of foundingpricetable

**costfactor1** - How much costs a founding (must be more then 17 or loco.exe will crash)

flags

**bit\_10** - Industry can be founded by user

**needall** - If it needs all cargo types to produce

**canincreaseproduction** - If it can increase production

**candecreaseproduction** - If it can decrease production

**produces[array]** - The cargo object it produces

**accepts[array]** - The cargo object it accepts

**fence[array]** - The fence object to be placed around the industry

**numaux4ent** = number of different building base tiles available

**aux4** = groups of buildings/ 1, 23, 45, 56, etc. \*\*\*\*\* Unverified

**aux5** = range is aux4 variables, some sort of selection of aux4 to randomize industry appearance? \*\*\*\*\*  
Unverified

---

## **Class 20 - Companies**

**spritesets** - Bits 0..8 indicate to use those sprites, two each

**intelligence** - Intelligence

**aggressiveness** - Aggressiveness

**competitiveness** - Competitiveness

**NOTA:** Lista de variables extraída de [LocoWiki](#) .

### **Descargas:**

[LocoTool 0.4.3b](#)