

MANUAL BARRIER

für den Landwirtschafts Simulator 2019



Inhaltsverzeichnis

2	Allgemeine Beschreibung
3	Erklärung der User Attribute
10	Erklärung der modDesc.xml Einträge

Beschreibung zur Version:

1.9.0.7 vom 09.02.2020

Änderungen gegenüber LS17 Version:

- fieldId (Integer) wurde ersetzt durch farmLandRestricted (Boolean) zur Anpassung an die MP-Farmen

Änderungen gegenüber Versionen < 1.9.0.7:

- Loop wenn öffnen, Loop wenn geschlossen und spiele volle Animation für animierte Objekte hinzugefügt

Allgemeine Beschreibung

ManualBarrier ist für Karten-Ersteller gedacht die viele unterschiedliche Objekte auf ihrer Karte entweder manuell oder automatisch in Bewegung versetzen wollen.

Eigentlich täuscht der Name denn dieses Paket kann weitaus mehr als nur eine Schranke manuell zu bedienen.

Egal ob nun ein Schiebetor, eine Schwingtür, eine Schranke oder gar ein komplex animiertes Objekt in Bewegung versetzt werden muss, manuell oder automatisch spielt dabei keine Rolle, oder ob es sich um einschaltbare Lichtquellen oder sonstiges handelt. Für all dies kann ManualBarrier verwendet werden.

Im Grunde kann ManualBarrier für so gut wie jedes Objekt verwendet werden, naja auf jeden Fall für die meisten 😊

Und dabei ist es sehr vielfältig, alles kann miteinander kombiniert werden, ob Rotation und Bewegung oder Animation und Licht, oder auch alles zusammen, ob man nun eine alte Leuchtstoffröhre mit dem typischen Einschaltflackern haben möchte oder ein Rundumlicht bei Schiebetoröffnung. Man will den Zugang zu einem Ort nur zu einer bestimmten Uhrzeit gewähren? Kein Problem, auch das kann ManualBarrier. Automatisches Hoflicht ab 20 Uhr? Warum nicht, ManualBarrier macht auch dies einfach.

Selbstverständlich ist ManualBarrier auch für den Multiplayer-Modus von LS19 geeignet, jede Funktion wird dabei voll unterstützt ebenso wie die Zuordnung zu den Farmen.

Für den genauen Einbau folgen eine Erklärung der User Attribute, ein Beispielaufbau im I3D-Editor Scenegraphen sowie die nötigen Einträge in der modDesc.xml.

Grundsätzlich setze ich jedoch Kenntnisse mit dem Giants-Editor sowie der XML-Bearbeitung voraus damit ManualBarrier Verwendung finden kann.

Erklärung der User Attribute

allowTraffic

- Werte-Typ: Boolean
- Standardwert: FALSE
- Verarbeitungsvoraussetzung: **CollisionMask** des Triggers Korrekt eingestellt
- ist der Wert gesetzt reagiert der Trigger auf KI-Fahrzeuge

animationClip

- Werte-Typ: String
- Standardwert: nicht vergeben
- Verarbeitungsvoraussetzung: **typeAnimated** = TRUE
- gibt den Namen des Clips an welcher für die Animation festgelegt wurde
- ist der Name falsch angegeben wird keine Animation aufgerufen

animatorIndex

- Werte-Typ: String
- Standardwert: nicht vergeben
- Verarbeitungsvoraussetzung: **typeAnimated** = TRUE
- gibt den Index zu einem animierten Objekt an
- sind mehrere Objekte in einem **animationClip** animiert ist es egal welcher von den Objekten über diesen Index angesprochen wird

animationPlayFull

- Werte-Typ: Boolean
- Standardwert: FALSE
- Verarbeitungsvoraussetzung: **typeAnimated** = TRUE
- besagt das die Animation in jedem Fall voll abgespielt wird unabhängig davon ob der Offen/Geschlossen-Status sich geändert hat
- Statusänderung der Animation erst nach vollem abspielen der Animation

animationLoopOnOpen

- Werte-Typ: Boolean
- Standardwert: FALSE
- Verarbeitungsvoraussetzung: **typeAnimated** = TRUE
- die Animation wiederholt sich wenn das Objekt den OFFEN-Status hat

animationLoopOnOpen

- Werte-Typ: Boolean
- Standardwert: FALSE
- Verarbeitungsvoraussetzung: **typeAnimated** = TRUE
- die Animation wiederholt sich wenn das Objekt den GESCHLOSSEN-Status hat

Name	Value	Checkbox
allowTraffic	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
animationClip	rollgate_rul	<input checked="" type="checkbox"/>
animatorIndex	0>2 0	<input checked="" type="checkbox"/>
audioIndex	0>3	<input checked="" type="checkbox"/>
automaticClose	18	<input checked="" type="checkbox"/>
automaticMode	<input type="checkbox"/>	<input checked="" type="checkbox"/>
automaticOpen	7	<input checked="" type="checkbox"/>
automaticStrict	<input type="checkbox"/>	<input checked="" type="checkbox"/>
fieldId	10	<input checked="" type="checkbox"/>
flickerMode	<input type="checkbox"/>	<input checked="" type="checkbox"/>
lightIndex	0>2 0 0	<input checked="" type="checkbox"/>
lightOnMove	<input type="checkbox"/>	<input checked="" type="checkbox"/>
manualBarrierId	1	<input checked="" type="checkbox"/>
manualOpen	<input type="checkbox"/>	<input checked="" type="checkbox"/>
maxX	0	<input checked="" type="checkbox"/>
maxXAngle	0	<input checked="" type="checkbox"/>
maxY	0	<input checked="" type="checkbox"/>
maxYAngle	0	<input checked="" type="checkbox"/>
maxZ	0	<input checked="" type="checkbox"/>
maxZAngle	0	<input checked="" type="checkbox"/>
minX	0	<input checked="" type="checkbox"/>
minXAngle	0	<input checked="" type="checkbox"/>
minY	0	<input checked="" type="checkbox"/>
minYAngle	0	<input checked="" type="checkbox"/>
minZ	0	<input checked="" type="checkbox"/>
minZAngle	0	<input checked="" type="checkbox"/>
onCreate	modOnCreate.ManualBarrier	<input checked="" type="checkbox"/>
speedScale	60	<input checked="" type="checkbox"/>
string_Close	string_CLOSE	<input checked="" type="checkbox"/>
string_Name	string_DEFAULT	<input checked="" type="checkbox"/>
string_Open	string_OPEN	<input checked="" type="checkbox"/>
triggerIndex	0>0	<input checked="" type="checkbox"/>
typeAnimated	<input type="checkbox"/>	<input checked="" type="checkbox"/>
typeBarrier	<input type="checkbox"/>	<input checked="" type="checkbox"/>
typeGate	<input type="checkbox"/>	<input checked="" type="checkbox"/>
typeLight	<input type="checkbox"/>	<input checked="" type="checkbox"/>

audioIndex

- Werte-Typ: String
- Standardwert: nicht vergeben
- gibt den Index zu einer AudioSource an welche bei der ausgeführten Bewegung abgespielt werden soll
- wenn angegeben wird der AudioClip abgespielt sobald sich das Objekt (Barrier|Gate|AnimatedObject) bewegt

audioLoopOnClose, audioLoopOnOpen

- Werte-Typ: Boolean
- Standardwert: false
- wenn das Objekt den entsprechenden Status angenommen hat (Close, Open) wird der angegebene Sound in einer Schleife abgespielt

automaticClose

- Werte-Typ: String
- Standardwert: 18
- Wertebereich: 0 bis 23.9999
- Verarbeitungsvoraussetzung: **automaticMode** = TRUE
- gibt die Uhrzeiten an zu der das Objekt den geschlossenen Zustand annehmen soll (z.B: „12.5 20“)
! mehrere Uhrzeiten werden mit einem Leerzeichen getrennt !

automaticMode

- Werte-Type: Boolean
- Standardwert: FALSE
- dient zum Setzen von automatischen Öffnungszeiten
- ist **fieldId** angegeben reagiert die automatische Funktion erst nach Kauf des unter **fieldId** angegebenen Objektes
- wenn Objekt NICHT auf manuelle Öffnung eingestellt ist wird das Objekt automatisch zur den **automaticOpen** Uhrzeiten geöffnet und verbleibt bis zur den **automaticClose** Uhrzeiten im offenen Zustand
- ist das Objekt auf manuelle Öffnung eingestellt so gelten folgende Regeln (Halbautomatischer Betrieb)
 - + das Objekt kann zu jeder Zeit geöffnet werden
 - + das Objekt wird automatisch den geschlossenen Zustand annehmen wenn die **automaticClose** Uhrzeit erreicht ist
 - + wird das Objekt während in der Zeit zwischen **automaticClose** und **automaticOpen** geöffnet und der Trigger im Anschluss wieder verlassen so schließt sich das Objekt selbstständig

automaticOpen

- Werte-Typ: String
- Standardwert: 7
- Wertebereich: 0 bis 23.9999
- Verarbeitungsvoraussetzung: **automaticMode** = TRUE
- gibt die Uhrzeiten an zu der das Objekt den geöffneten Zustand annehmen soll (z.B: „6.5 15“)
! mehrere Uhrzeiten werden mit einem Leerzeichen getrennt !
 - sind mehrere Uhrzeiten angegeben und dazu passend keine **automaticClose** Uhrzeit dann wird die **automaticClose** Uhrzeit auf **automaticOpen** + 1 Stunde gesetzt

automaticStrict

- Werte-Typ: Boolean
- Standardwert: FALSE
- Verarbeitungsvoraussetzung: **automaticMode** = TRUE
manualOpen = TRUE
- greift nur wenn das Objekt mit manueller Öffnung versehen ist
- es gelten ähnliche Regeln wie beim Halbautomatischem Modus (siehe **automaticMode**) bis auf folgende Änderungen
+ ist eine der **automaticClose** Uhrzeiten erreicht wird sich das Objekt schließen und kann bis zum eintreten einer **automaticOpen** Uhrzeit NICHT wieder geöffnet werden

flickerMode

- Werte-Typ: Boolean
- Standardwert: FALSE
- Verarbeitungsvoraussetzung: **typeLight** = TRUE
- das unter **lightIndex** angegebene Licht wird mit einem flackern (ähnlich einer alten Leuchtstoffröhre) gestartet

farmLandRestricted

- Werte-Typ: Boolean
- Standardwert: FALSE
- besitzt die Farm das Land auf welches das Objekt steht können Spieler dieser Farm das Objekt bedienen

lightIndex

- Werte-Typ: String
- Standardwert: nicht vergeben
- gibt den Index zu einem Objekt an welches eingeblendet werden soll

lightOnMove

- Werte-Typ: Boolean
- Standardwert: FALSE
- gibt an ob ein Objekt auch dann noch eingeblendet bleiben soll
wenn sich das Objekt nach Verlassen des Triggers schließt
- ist dieser Wert nicht angegeben (= FALSE) dann wird das unter **lightIndex** angegebene Objekt nach Verlassen des Triggers direkt ausgeblendet, unabhängig von der Bewegung anderer Objekte des Triggers

lightOnlyNight

- Werte-Typ: Boolean
- Standardwert: FALSE
- gibt an ob das Objekt welches unter **lightIndex** angegeben wurde nur Nachts aktiv werden soll
- kann mit **lightAutoNightOn** und **lightAutoNightOff** eingestellt werden, andernfalls wird der Sonnenstatus abgefragt

lightAutoNight

- Werte-Typ: Boolean
- Standardwert: FALSE
- gibt an ob das Objekt welches unter **lightIndex** angegeben wurde automatisch Nachts aktiv geschaltet soll
- kann mit **lightAutoNightOn** und **lightAutoNightOff** eingestellt werden, andernfalls wird der Sonnenstatus abgefragt

lightsAutoNightOn

- Werte-Typ: Float
- Standardwert: -1 (off)
- Verarbeitungsvoraussetzung: **lightAutoNight** oder **lightOnlyNight** = TRUE
- gibt an ab wann das Objekt welches unter **lightIndex** angegeben wurde Nachts aktiv geschaltet werden kann

lightsAutoNightOff

- Werte-Typ: Float
- Standardwert: -1 (off)
- Verarbeitungsvoraussetzung: **lightAutoNight** oder **lightOnlyNight** = TRUE
- gibt an ab wann das Objekt welches unter **lightIndex** angegeben wurde Nachts inaktiv geschaltet werden soll

greenLightIndex

- Werte-Typ: String (Objektzuordnung)
- Standardwert: nicht vergeben
- wenn angegeben wird das Objekt ab 80% Öffnungszustand aktiv geschaltet

redLightIndex

- Werte-Typ: String (Objektzuordnung)
- Standardwert: nicht vergeben
- wenn angegeben wird das Objekt bei unter 80% Öffnungszustand aktiv geschaltet

manualBarrierId

- Werte-Typ: Integer
- Standardwert: 1 (!!!)
- Wertebereich: 1 bis {Spielinterne Integerbegrenzung}
- gibt die fortlaufende Nummer zu den mit Manual Barrier Scripten gesteuerten Objekten an
- darf NICHT mehrfach verwendet werden da andernfalls lediglich das letzte Objekt welches die gleiche ID hat korrekt arbeiten kann

TIP: weise die `nodeId` aus dem GIANTS Editor als *manualBarrierId* zu, so umgehst du doppelte ID Zuweisungen

closeSymbolIndex

- Werte-Typ: String (Objektzuordnung)
- Standardwert: nicht vergeben
- wenn angegeben wird es bei Schließung des Objektes eingeblendet und bei Öffnung ausgeblendet

openSymbolIndex

- Werte-Typ: String (Objektzuordnung)
- Standardwert: nicht vergeben
- wenn angegeben wird es bei Öffnung des Objektes eingeblendet und bei Schließung ausgeblendet

rotationObjectIndex

- Werte-Typ: String (Objektzuordnung)
- Standardwert: nicht vergeben
- wenn angegeben wird das Objekt während des Öffnungs- und Schließvorganges rotiert um die unter **rotationObjectAxis** angegebene Achse mit der unter **rotationObjectSpeed** angegebenen Geschwindigkeit
- die Rotation stoppt sobald der Anfangs- bzw Endzustand des zu öffnenden Objektes erreicht wurde

rotationObjectSpeed

- Werte-Typ: Float
- Standardwert: 10
- Verarbeitungsvoraussetzung: **rotationObjectIndex** zugewiesen
- gibt die Rotationsgeschwindigkeit des unter **rotationObjectIndex** angegebenen Objektes an

rotationObjectAxis

- Werte-Typ: Integer
- Standardwert: 1
- Wertebereich: 1 bis 3
- Wertebereichzuordnung: 1 = X-Achse ; 2 = Y-Achse ; 3 = Z-Achse
- Verarbeitungsvoraussetzung: **rotationObjectIndex** zugewiesen
- gibt die zu rotierende Achse des unter **rotationObjectIndex** zugewiesenen Objektes an

manualOpen

- Werte-Typ: Boolean
- Standardwert: FALSE
- gibt an dass das Objekt manuell zu steuern ist
- ist der Wert TRUE lässt sich das Objekt komplett manuell öffnen/anschalten und schließen/abschalten

maxX, minX, maxY, minY, maxZ, minZ

- Werte-Type: Float
- Standardwert: 0.0
- Wertebereich: Spielinterne Float Begrenzung (+ und -)
- Verarbeitungsvoraussetzung: **typeGate** = TRUE
- **min#** gibt die Ausgangsposition für zu verschiebende Objekte an, definieren gleichzeitig die geschlossene Position
- **max#** gibt die maximale Offen-Position an
- es müssen nicht alle Achsen angegeben werden, lediglich die Achse welche bewegt werden soll, andere nicht angegebene Achsen verbleiben in ihrer Ausgangsposition

maxXAngle, minXAngle (für X, Y und Z)

- Werte-Typ: Float
- Standardwert: 0.0
- Wertebereich: -65535 bis 65535
- Verarbeitungsvoraussetzung: **typeBarrier** = TRUE
- **min#Angle** gibt die minimale Rotation in Grad der entspr. Achse an, definiert gleichzeitig die geschlossene Position
- **max#Angle** gibt die maximale Offen-Rotation in Grad an
- es müssen nicht alle Achsen angegeben werden, lediglich die Achse welche rotiert werden soll, andere nicht angegebene Achsen verbleiben in ihrer Ausgangsrotation

onCreate

- Werte-Typ: ScriptCallback
- Standardwert: nicht vergeben, für diese Funktion notwendig ist ***modOnCreate.ManualBarrier***
- legt den Aufruf für das Script fest
- ohne diesen Aufruf wird sich das Objekt nicht bewegen oder auf Trigger-Kollisionen reagieren

speedScale

- Werte-Typ: Float
- Standardwert: 60
- Wertebereich: 1 bis 65535
- legt die Geschwindigkeit der Bewegung/Rotation/Animation fest
- niedrige Werte verlangsamen, höhere Werte beschleunigen die Bewegung/Rotation/Animation

closelyRaining

- Werte-Typ: Boolean
- Standardwert: FALSE
- ist aktiv sobald es regnet und schließt das Objekt
- schaltet das Objekt frei sobald der Regen aufhört
- überschreibt die manuelle Steuerung wenn es regnet, Objekt läßt sich also nicht mehr öffnen wenn es regnet

string_Close

- Werte-Typ: String
- Standardwert: string_CLOSE (= schließen)
- setzt die Textvariable für den Schließvorgang für die Anzeige im Hilfefenster des Spiels
- Werte werden in der ***modDesc.xml*** der Karte definiert

string_Name

- Werte-Typ: String
- Standardwert: string_DEFAULT (= Tor/Schranke)
- setzt die Textvariable für den Namen des Objektes für die Anzeige im Hilfefenster des Spiels
- Werte werden in der ***modDesc.xml*** der Karte definiert

string_Open

- Werte-Typ: String
- Standardwert: string_OPEN (= öffnen)
- setzt die Textvariable für den Öffnungsvorgang für die Anzeige im Hilfefenster des Spiels
- Werte werden in der ***modDesc.xml*** der Karte definiert

triggerIndex

- Werte-Typ: String
- Standardwert: nicht vergeben
- gibt den Index zum Trigger an
- wird kein Trigger angegeben kann das Objekt nur funktionieren wenn **manualOpen** = FALSE und **automaticMode** = TRUE ist, andernfalls wird das Objekt keine Funktion darstellen

typeAnimated

- Werte-Typ: Boolean
- Standardwert: FALSE
- gibt an ob das Objekt ein animiertes Objekt ist und demnach eine Animation enthält
- kann mit allen anderen Typen kombiniert werden
- wird deaktiviert wenn keine Animation zugewiesen und/oder keine animierter Index zugewiesen wird

typeBarrier

- Werte-Typ: Boolean
- Standardwert: FALSE
- gibt an ob das Objekt ein zu rotierendes Objekt ist
- kann mit allen anderen Typen kombiniert werden

typeGate

- Werte-Typ: Boolean
- Standardwert: FALSE
- gibt an ob das Objekt ein zu bewegendes (verschiebendes) Objekt ist
- kann mit allen anderen Typen kombiniert werden

typeLight

- Werte-Typ: Boolean
- Standardwert: FALSE
- gibt an ob das Objekt ein/ausgeblendet werden soll
- kann mit allen anderen Typen kombiniert werden

randomClose

- Werte-Typ: Boolean
- Standardwert: FALSE
- gibt an ob das Objekt zufälligen Schließzeiten unterliegen soll
- kann mit allen Typen kombiniert werden

randomRotate

- Werte-Typ: Boolean
- Standardwert: FALSE
- Verarbeitungsvoraussetzung: **rotationObjectIndex** zugewiesen
randomClose = TRUE
- gibt an ob das Objekt welches unter **rotationObjectIndex** zugewiesen wurde bei zufälligen Schließzeiten dauerhaft rotieren soll während der Schließung

randomLight

- Werte-Typ: Boolean
- Standardwert: FALSE
- Verarbeitungsvoraussetzung: **typeLight** = TRUE
randomClose = TRUE
- gibt an ob das Objekt welches unter **lightIndex** zugewiesen wurde bei zufälligen Schließzeiten dauerhaft leuchten soll während der Schließung

randomChance

- Werte-Typ: Float
- Standardwert: 25
- Verarbeitungsvoraussetzung: **randomClose** = TRUE
- gibt die Wahrscheinlichkeit einer zufälligen Schließung des Objektes in Prozent an

randomText1

randomText2

randomText3

- Werte-Typ: String
- Standardwert: default_noentry_msg
- Verarbeitungsvoraussetzung: **randomClose** = TRUE
- gibt mögliche Gründe für zufällige Schließungen an welche bei betreten/befahren des Triggers angezeigt werden

controlKey

- Werte-Typ: String
- Standardwert: OPEN_GATE
- gibt die Tastenkombination für das Bedienen des Objektes an
- zugehörige Tastenwerte und Bezeichnungen müssen in der modDesc.xml definiert werden

Erklärung der MODDESC.XML Einträge

1.1

Folgende Zeilen müssen in die modDesc.xml aufgenommen werden:

```
<extraSourceFiles>
    <sourceFile filename="scripts/ManualBarrier.lua" />
</extraSourceFiles>
```

Sollte bereits eine <extraSourceFiles>-Sektion in der modDesc.xml vorhanden sein sind lediglich die beiden <sourceFile>-Einträge zu übernehmen damit ManualBarrier funktioniert.

WICHTIG !

Der Pfad zum Script muß eventuell an eure Karte angepasst werden!

1.1.1

Wer die Scripte nicht in der modDesc.xml eintragen möchte, sie jedoch dennoch nutzbar machen will auf seiner Karte kann auch die entsprechende LUA der Karte (Beispiel: SampleModMap.lua) mit dem Eintrag versehen:

```
source(Utils.getFilename("scripts/ManualBarrier.lua", baseDirectory));
```

Diese Einträge sind in der :new(baseDirectory, customMt) Funktion des Kartenscriptes einzutragen und zwar unmittelbar VOR dem return self der Funktion.

Beispiel:

```
function SampleModMap:new(baseDirectory, customMt)
    local mt = customMt;
    if mt == nil then
        mt = SampleModMap_mt;
    end;
    local self = SampleModMap:superClass():new(baseDirectory, mt);

    source(Utils.getFilename("scripts/ManualBarrier.lua", baseDirectory));

    return self;
end;
```

1.2

Für das manuelle Bedienen ist eine Tastenkonfiguration sowie Aktion in der modDesc.xml anzugeben:

```
<inputBinding>
    <actionBinding action="OPEN_GATE">
        <binding device="KB_MOUSE_DEFAULT" input="KEY_o" />
    </actionBinding>
</inputBinding>
<actions>
    <action name="OPEN_GATE" />
</actions>
```

Der Name "OPEN_GATE" ist dabei unbedingt beizubehalten da dieser vom Script explizit abgefragt wird und es andernfalls zu Fehlermeldungen in der Log-Datei kommt.

Auch hier gilt: sollte bereits eine <inputBinding>-Sektion oder eine <actions>-Sektion in der modDesc.xml bestehen lediglich den <actionBinding>-Eintrag bzw den <action>-Eintrag hinzufügen.

Standard Texteinträge für die mitgelieferten Beispielobjekte und Standard-Textzuweisungen des Scriptes sind in den L10N-Einträgen in der modDesc.xml hinzuzufügen:

```
</l10n>
    <text name="string_OPEN"><en>open</en><de>öffnen</de></text>
    <text name="string_CLOSE"><en>close</en><de>schließen</de></text>
    <text name="string_ON"><en>switch on</en><de>anschalten</de></text>
    <text name="string_OFF"><en>switch off</en><de>ausschalten</de></text>
    <text name="string_BARRIER"><en>barrier</en><de>Schranke</de></text>
    <text name="string_GATE"><en>gate</en><de>Tor</de></text>
    <text name="string_LIGHT"><en>light</en><de>Licht</de></text>
    <text name="string_WICKET"><en>wicket</en><de>Gatter</de></text>
    <text name="string_DEFAULT"><en>gate/barrier</en><de>Tor/Schranke</de></text>
    <text name="OPEN_GATE"><en>Open/close gate/door</en><de>Tor/Schranke öffnen/schließen</de></text>
    <text name="input_OPEN_GATE"><en>Open/close object</en><de>Objekt öffnen/schließen</de></text>
    <text name="default_noentry_msg"><en><![CDATA[We are closed due to a disturbance at the moment. We expect
to open again at ]]></en><de><![CDATA[Momentan haben wir Aufgrund einer Störung geschlossen. Wir öffnen
voraussichtlich wieder gegen ]]></de></text>
    <text name="default_noentry_clock"><en><![CDATA[o Clock ]]></en><de><![CDATA[ Uhr.]]></de></text>
</l10n>
```

Diese Werte sind die Standardwerte, eigene Einträge können selbstverständlich vorgenommen werden, wenn man zum Beispiel explizit ein Holztor angezeigt bekommen will kann man den Eintrag selbst hinzufügen:

```
<text name="string_WOODGATE"><en>wooden gate</en><de>Holztor</de></text>
```

Nachfolgend noch im I3D-Editor in den User Attributen den **string_Name** Eintrag anpassen zu: string_WOODGATE (Standard ist: string_DEFAULT) und schon bekommt man im Spiel anstelle von „Tor/Schranke öffnen“ den Hilfeintrag „Holztor öffnen“ angezeigt.

Auch hier gilt: sollte es bereits eine <l10n>-Sektion in der modDesc.xml bestehen lediglich die <text>-Eintrag hinzufügen.

Viel Spaß wünscht euch Blacky_BPG