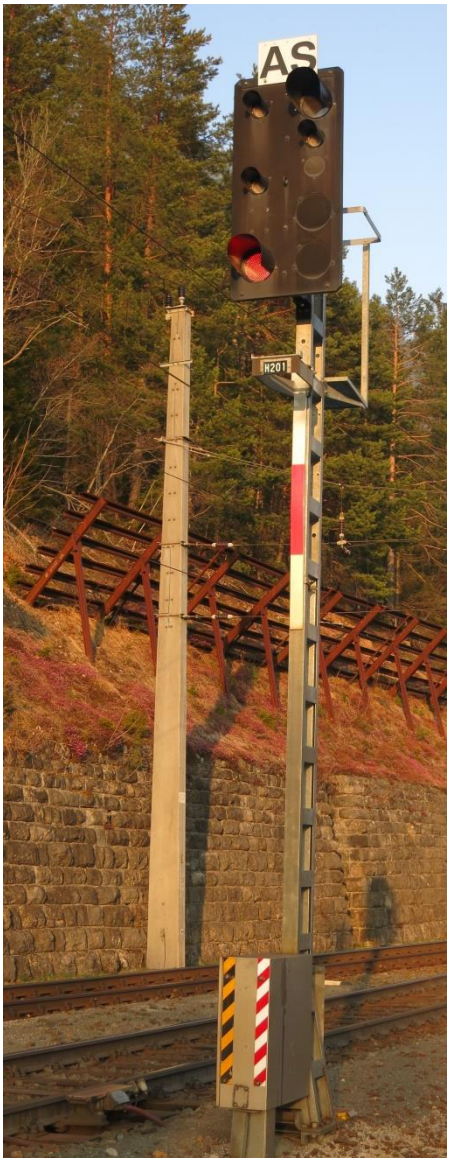
Skripte für OEBB-Signale im Train Simulator (Version 9.0)

Am 1. Juni 1980 trat die umfangreichste Änderung der Betriebsvorschriften seit dem Zweiten Weltkrieg in Kraft. Neben der neu aufgelegten Signalvorschrift V2 wurden auch zahlreiche andere Vorschriften geändert oder völlig neu gestaltet.

Der „Halt"-Begriff des Hauptsignals gilt nun auch für Verschubfahrten (in Deutschland-West wurde dieser Schritt erst 2006 gemacht).
Bei Verschubsignalen am Standort eines Hauptsignals wurden daher die Lampen für den Begriff „Verschubverbot" abgeschaltet und im Lauf der Zeit entfernt. Beim neuen Standardsignal wurde dann das Verschubsignal in den Signalschirm integriert.
Quelle: Wikipedia



Die Skripte aus diesem Paket sind für die Signalisierung nach der Signalvorschrift V2 von 1980 geeignet.

Sie können ihre selbst gebauten OEBB-Signale verschiedenster Bauformen unter Einsatz dieser Skripte auf Freewarestrecken im Train Simulator verwenden.
Beachten Sie die Lizenzbedingungen unter Punkt 1.2.

Dieses Paket beinhaltet die Anleitung und Skripte, um selbst erstellte OEBB-Signale mit den Skript-Modulen aus dem Paket „**Freeware Skript-Module und Signal-Trigger"** zu einem funktionierenden Signalsystem zu verbinden.

Wenn Sie lediglich fertige Signale nutzen wollen, die für die **Schuster/Freeware - Skript-Module** erstellt wurden, benötigen Sie dieses Paket nicht, sondern laden Sie sich von Rail-Sim.de das Paket **„Freeware Skript-Module und Signal-Trigger"** herunter und installieren dieses mit der Utilities.exe.

Bestandteile des Paketes: Skriptpaket_OEBB-Signale_V9.0.zip

- Anleitung Skriptpaket_OEBB-Signale_V9.0.pdf
- Excel-Tabelle Code-Berechnung_OEBB-Signale.xlsx
- Skripte für Source Skripte_OEBB-Signale_V9.0.zip

Indem Sie dieses Paket nutzen, akzeptieren Sie auch die Lizenzbestimmungen.

Bei Problemen/Anregungen bitte über das Forum www.Rail-Sim.de anschreiben.

Viel Erfolg beim Einsatz der Skripte für die OEBB-Signale wünscht
Mathias Gundlach (Schuster at Rail-Sim)

Inhaltsverzeichnis

1. Einleitung

1.1. Einige Vorworte

Wer sich mit dem Bau von OEBB-Signalen beschäftigt, und diese Skripte verwenden möchte, sollte sich grundsätzlich mit andern Signalpaketen vom SignalTeam auseinandersetzen, um die Sonderfunktionen der Signalsysteme zu verstehen.

Bezüglich der Sonderfunktionen im Zusammenhang mit Triggern und Einstellungen im Signal-Flyout unterscheiden sich diese Skripte nicht von denen, die unter „**Schuster/SignalTeam**“ veröffentlicht wurden. Hierbei ist jedoch der jeweilige Versionsstand zu beachten.

Trigger sind nicht Bestandteil dieses Skriptpaketes. Diese werden separat unter „**Schuster/Freeware**“ veröffentlicht und können dann für alle Signalsysteme dieser Veröffentlichungsreihe gemeinsam genutzt werden. Gleichfalls können auch alle bisher veröffentlichten Trigger aus anderen Signalpaketen, soweit sie dem aktuellen Stand entsprechen, genutzt werden.

1.2. Lizenzbestimmungen

Das Paket wird als Freeware auf Rail-Sim (www.Rail-Sim.de) angeboten und darf nicht auf weiteren Plattformen ohne meine Erlaubnis angeboten werden.

Die Skripte und Module dürfen ausschließlich auf **Freeware-Strecken** verwendet werden. Es ist nicht gestattet, die Skripte oder Module für kommerzielle Strecken zu verwenden.

Die Module dürfen nicht geändert, angepasst oder in anderen Provider-/Produktordnern gespeichert und dort heraus geladen werden. Updates stelle ich ausschließlich selbst zur Verfügung.

Die Module dürfen nicht als Bestandteil von Strecken oder Signalpaketen verteilt werden. Sie dürfen nur per Link aus ihrer ursprünglichen Downloadquelle unter Rail-Sim angeboten werden.

Die im Klartext gelieferten Skripte dürfen angepasst, kopiert, exportiert und im eigenen Source-Ordner verwendet werden. Sie dürfen dann auch im OUT-Format innerhalb der eigenen Streckenpakete verpackt und verteilt werden.

Sollte Bedarf bestehen, dass diese Module in **Payware-Projekten** verwendet werden, so bitte ich um Kontaktaufnahme per E-Mail an Railworks@mgundlach.de . Die Module werden dann durch mich an den entsprechenden Provider- und Produktordner angepasst und eine Lizenz zur Verwendung vergeben.

1.3. Installation

Das Skriptpaket ist eine ZIP-Datei und muss mit einem geeigneten Programm entpackt werden. Die Skripte werden dann in den Source-Ordner kopiert, in dem später auch die eigenen Signale abgelegt werden.

1.4. Neuerungen gegenüber der vorherigen Version

Version 9.0

- Erste veröffentlichte Version des Skriptpaketes
- Gegenüber den bisher veröffentlichten Freeware-Paketen ist es bei diesem Signalpaket auch möglich bei Hauptsignalschirmen einzelne Signaloptiken per Code auszublenden.

2. Ordnerstruktur

- Hauptordner
    - Assets\Schuster\Freeware\RailNetwork\Signals\OEBB

  In diesem Ordner befindet sich lediglich die Optionsdatei. Diese muss dort unbedingt gespeichert bleiben und dient der Einstellung von einigen Optionen für das Signalsystem.

- Module

  Der Ordner Module im Bereich **Assets** ist ein Unterordner von „OEBB“ und enthält alle Skript-Module im OUT-Dateiformat, die aus den Skripten heraus nachgeladen werden. Diese Skriptmodule werden in keinem BluePrint direkt angegeben!

- Skripte

  Im Bereich **Source** liegen in einem Unterorder der sinnvoller Weise „Skripte“ lauten sollte, unterhalb ihres eigenen Signalverzeichnisses alle Skripte mit der Dateiendung „Lua“. Diese Skripte werden dann im **BluePrintEditor** jeweils angegeben. In diesen Skripten befindet sich dann auch der Verweis („require….“) auf das entsprechende Modul. Der Modulaufruf darf nicht geändert werden.

3. Vorgaben für 3D-Modelle

Damit die Signale mit den Skripten funktionieren, müssen verschiedene Bedingungen erfüllt sein:

- Die Namen der Licht-Nodes im 3D-Modell müssen mit denen im Skript übereinstimmen
- Die Child-Namen der Signalschirme, die im BluePrintEditor angegeben werden, müssen mit denen im Skript übereinstimmen
- Für jede im BluePrintEditor zusammengebaute Kombination aus Signalschirmen und Zusatzanzeigern, muss ein passender Signalskript erstellt und im BluePrintEditor angegeben werden.

3.1. Allgemeines

Um für dieses und andere Signalsysteme möglichst wenig unterschiedliche 3D-Modelle erstellen zu müssen, sollten die Bezeichnungen der Licht-Nodes einem System folgen. Einmal vorgegeben, dürfen diese auch nicht mehr geändert werden.

Die in den Skripten stehenden Verweise auf die Module dürfen nicht geändert werden und passen für das entsprechende Signal oder Zusatzanzeiger bei Einhaltung aller Vorgaben.

3.1.1. Regeln für die Benennung der Licht-Nodes

Die Namen der Licht-Nodes werden vorgegeben, da diese Namen im Skript festgelegt sind.

Die Namen der Licht-Nodes sind abweichend von denen der Deutschen Signale und auch für jeden Signaltyp unterschiedlich. Beachten Sie also bitte die nachfolgenden Tabellen.

In späteren Abschnitten folgen also Tabellen mit den vorgegebenen Licht-Node-Namen und den notwendigen Code-Werten zur Berechnung der Code-Summe. Diese Code-Summe wird dann im Skript eingetragen und muss mit den verbauten Licht-Nodes im Signal übereinstimmen.

3.1.2. Anmerkungen zu den Child-Namen

Für den 3D-Bau der Signalschirme werden außer den festgelegten Licht-Node-Namen keine weiteren Vorgaben benötigt. Erst die Verwendung des **BluePrintEditors** erfordert einige Einstellungen, von denen zum Beispiel die Child-Namen ganz wichtig für die Funktion der Signale sind. Diese Child-Namen sind entsprechend des Signaltyps unterschiedlich.
In den nachfolgenden Tabellen werden die jeweils zu verwendenden Child-Namen in der ersten Spalte aufgeführt. Bei diesen Child-Namen muss unbedingt auf Groß- und Kleinschreibung geachtet werden.

3.1.3. HP_CODE und VR_CODE

Die Vergangenheit hat gezeigt, dass es eine Unmenge an Sonderbauformen bei den Signalschirmen gibt. Gerade die Verwendung dieser Sonderbauformen macht eine Strecke so interessant. Leider war der Aufwand bei der Skriptgestaltung durch diese Sonderbauformen bisher sehr hoch.

Aus diesem Grunde habe ich für diese Skripte einen ganz neuen Lösungsansatz entwickelt:

- Es werden einheitliche Child-Namen auch bei unterschiedlichen Signalschirmen verwendet.
- Jede Signaloptik erhält einen eindeutigen Node-Namen, der auch bei unterschiedlichen Signalschirmen entsprechend der Funktion gleich lautet.
- Jedem dieser Node-Namen wird ein eindeutiger Zahlencode zugeordnet.
- Im Skript wird dann nur noch die Summe aller Zahlencodes je Signalschirm übermittelt und daraus alle gültigen Signaloptiken errechnet.
- Hierfür wurde im Skriptpaket eine Excel-Tabelle zur automatischen Berechnung der Code-Summe mitgeliefert. Auswahlen, die sich widersprechen erzeugen eine rote Einfärbung.

Wer kein Excel zur Verfügung hat muss sich diese Code-Summe anhand der Tabelle manuell errechnen.

Berechnungstabelle für OEBB-Hauptsignale			
Node-Name	**Bezeichnung**	**Hauptsignale**	**Basis-Code**
HSred01	Rot		2
HSgreen01	Grün oben		16
HSgreen02	Grün Mitte oder unten		64
HSyellow01	Gelb unten		128
HSwhite04	Abfahrsignal (grün)		256
HSwhite02	Verschubsignal einzeln links tiefer		512
HSwhite03	Verschubsignal einzeln rechts höher		1024
HSwhite23	Verschubsignal beide weißen Signallichter		2048
HSwhite01	Kennlicht / Zusatzlicht		4096
ZS1	Ersatzsignal (weiß)		8192
ZS7	Vorsichtssignal (rot)		16384
		Code-Summe berechnet:	

Die Signallichter für ZS1 und Zs7 lassen sich gleichzeitig übereinander verbauen und später per OBJ_CODE wahlweise ausblenden. Es wird nur eine fest eingebaute Signallaterne verwendet.

Berechnungstabelle für OEBB-Vorsignale am Hauptsignalmast			
Node-Name	**Bezeichnung**	**Vorsignale**	**Basis-Code**
VSgreen01	Grün links unten		16
VSyellow01	Gelb links oben		32
VSgreen02	Grün rechts Mitte		64
VSyellow02	Gelb rechts oben		128
		Code-Summe berechnet:	

Node-Namen für OEBB-Vorsignale		
Node-Name	**Bezeichnung**	**Vorsignale**
VSgreen01	Grün links unten	
VSyellow01	Gelb links oben	
VSgreen02	Grün rechts Mitte	
VSyellow02	Gelb rechts oben	
		Es wird keine Code-Summe verwendet

Die Code-Summe wird bei einzelnen Vorsignalen nicht benötigt. Es sind immer alle Nodes aktiv.

Berechnungstabelle für OEBB- Vorsignalnachahmer			
Node-Name	**Bezeichnung**	**Vorsignalnachahmer**	**Basis-Code**
Snawhite01	Balken senkrecht		2
Snawhite02	Balken aufsteigend		4
Snawhite03	Balken waagerecht		8
Snawhite04	Balken absteigend		16
Snawhite05	Ersatzsignalankündigung		32
		Code-Summe berechnet:	

Berechnungstabelle für OEBB-Schutzsignale			
Node-Name	**Bezeichnung**	**Schutzsignale**	**Basis-Code**
HSred01	Rot		2
HSSchutz01	Weiß links oben		16
HSSchutz02	Weiß links Mitte		64
HSwhite04	Abfahrsignal (grün)		256
HSwhite02	Verschubsignal einzeln links tiefer		512
HSwhite03	Verschubsignal einzeln rechts höher		1024
HSwhite23	Verschubsignal beide weißen Signallichter		2048
HSwhite01	Kennlicht / Zusatzlicht		4096
ZS1	Ersatzsignal (weiß)		8192
ZS7	Vorsichtssignal (rot)		16384
		Code-Summe berechnet:	

Hinweis:
Die Signaloptik HSSchutz02 ersetzt die Funktion der Signaloptik HSwithe02 vollständig. Wenn also eine HSSchutz02 vorhanden ist, so leuchtet diese auch bei Verschubfreigabe.

Node-Namen für OEBB-Verschubsignale		
Node-Name	**Bezeichnung**	**Verschubsignale**
ZwSWhite01	Weiß rechts oben	
ZwSWhite02	Weiß links Mitte	
ZwSWhite03	Weiß rechts Mitte	
ZwSWhite04	Weiß links unten	
		Es wird keine Code-Summe verwendet

Node-Namen und Basis-Codes, die in anderen Signalsystemen möglich sind, aber in dieser Tabelle nicht aufgeführt wurden, dürfen nicht verwendet werden!

3.1.4. Zusatzanzeiger

An einem Hauptsignal oder Hauptsignal mit Vorsignal bzw. Vorsignal können verschiedene Geschwindigkeitsanzeiger angefügt werden. Dies sind dann separate 3D-Modelle, die in der Signal-BIN über den Child-Namen angesprochen werden.

Alle möglichen Zusatzanzeiger sind im Skript als Kommentarzeile bereits aufgeführt und können durch entfernen der Kommentarstriche aktiviert werden. In den Skripten, in denen keine Zusatzanzeiger als aktivierbare Variable aufgeführt wurden, sind diese auch nicht möglich. In diesem Fall sind die Zusatzanzeiger als separates Signalobjekt zu erstellen.

3.2. Sonderfunktionen

3.2.1. Ausblenden von Signaloptiken

Um weniger unterschiedliche Signalschirme bauen zu müssen, ist es mit diesen Skripten möglich, vorhandene Signallaternen auszublenden und die vorhanden Licht-Nodes dauerhaft dunkel zu schalten.

Damit Signaloptiken ausgeblendet werden können, müssen 2 Bedingungen erfüllt werden:

1. Das 3D Modell wurde so gebaut, dass eine komplette Signallaterne über einen Node-Namen wie eine Signaloptik angesprochen werden kann
2. Der Skript muss beim Laden der Strecke genau diesen Node-Namen ansprechen und ähnlich wie ein Signallicht dunkel geschaltet wird, die Signaloptik ausblenden.

Um diese Funktionalität möglichst einfach zu gestalten, wird eine zusätzliche Code-Variable (`OBJ_CODE`) eingeführt, die die Summe aller Signaloptik-Codes enthält die ausgeblendet werden sollen. Da die Signaloptik dann existiert, **muss die Signaloptik auch im HP_CODE** mit eingerechnet werden.
Die Node-Namen der Signaloptiken sind zu den Licht-Nodes analog und werden nur die Bezeichnung OBJ_ vorangestellt. Soll nun zum Beispiel die beide Signaloptiken vom Verschubsignal ausblendbar gebaut werden, gilt folgendes:

Im Signalschirm vorhandene Signaloptiken: HSred01, HSgreen01, HSwhite02, HSwhite03

Name der Licht-Node:	HSwhite02, HSwhite03
Name der Signallaterne:	OBJ_ HSwhite02, OBJ_HSwhite03
Code im Skript:	HP_CODE = 1554
	OBJ_CODE = 1536

Da nun HSwhite02 und HSwhite03 ausgeblendet wurde, wird nun auch kein Sh1 mehr signalisiert.

3.2.2. Ausblenden von Mastvarianten bei Haupt-, Schutz- und Vorsignalen

Um weniger unterschiedliche Signalschirme bauen zu müssen, ist es mit diesen Skripten möglich, ein Signalmodell zu erstellen, welches 2 Mastvarianten beinhaltet. Einer der beiden Masten wird beim Laden der Strecke automatisch ausgeblendet.

Damit Mastvarianten ausgeblendet werden können, müssen 4 Bedingungen erfüllt werden:

1. Der Child-Name des Hauptsignals enthält im Namen eine 2
   z.B. HP_SIGNAL_HEAD_NAME = „SchildHS2“
   oder HP_SIGNAL_HEAD_NAME = „2SchildHS“
2. Die beiden Mastvarianten werden in der Signal-BIN als Childs hinzugefügt. Die Child-Namen sind festgelegt! (siehe weiter unten)
3. Die Node, also der Mast, welcher später ausgeblendet werden soll muss einen bestimmten Namen haben.
4. Der Skript muss beim Laden der Strecke über den Child-Namen genau diesen Node-Namen ansprechen und ähnlich wie ein Signallicht dunkel geschaltet wird, den Mast ausblenden.

Soll der alternative Mast ausgeblendet werden, so muss im ID-Feld des Signals das Sonderzeichen ‘ (Hochkomma) eingetragen werden.

Die Client-Namen der beiden Masten müssen folgendermaßen lauten:

MastLinks	wird standardmäßig ausgeblendet
MastRechts	wird standardmäßig angezeigt

Die Node des Mastes muss „Mast“ heißen. Diese wird dann ggf. ausgeblendet.

3.2.3. Weißer Rahmen um Signalschirme

Signalschirme sind beim Vorbild teilweise mit einem weißen Rahmen ausgestattet.

Um weniger unterschiedliche Signalschirme bauen zu müssen, ist es mit diesen Skripten möglich, ein Signalmodell zu erstellen, welches diesen weißen Rahmen bereits schaltbar beinhaltet. Der Rahmen wird später per Skript standardmäßig ausgeblendet.
Dieser Rahmen kann per Sonderzeichen **[** (eckige Klammer auf) im ID-Feld eingeblendet werden.

Hierzu muss der Rahmen natürlich als Node existieren. Der Namen der Node ist frei wählbar und wird im Skript mit der Variable **gRahmen** vereinbart. Beispiel für einen Node-Namen „vsRahmen“:

gRahmen = “vsRahmen“

Wird in der Variable der Node-Name des Rahmens angegeben, wird der Rahmen beim Laden der Strecke standardmäßig durch den einen Skriptbefehl ausgeblendet. Mit dem Sonderzeichen **[** (eckige Klammer auf) an einer beliebigen Stelle im ID-Feld wird der Rahmen dann beim entsprechenden Vorsignal nicht ausgeblendet, sondern bleibt eingeblendet.

3.3. Vorsignale

Separate Vorsignale verwenden im Skript keinen VR_CODE.

Allerdings muss in der Variable VR_SIGNAL_HEAD_NAME der Child-Name des Vorsignalschirmes angegeben werden. Dieser Wert ist in der Regel auch im Skript direkt vorgegeben.

3.3.1. Vorsignal

Beschreibung		Child-Name
Vorsignal	(Child-Name muss diesen Namen enthalten!)	SchildVS

Node-Name	Beschreibung	Leuchtet bei Signalbegriff
VSgreen01	Grün links unten	Hauptsignal frei, Hauptsignal frei mit 60 km/h, Hauptsignal frei mit 40 km/h
VSyellow01	Gelb links oben	Vorsicht, Hauptsignal frei mit 60 km/h, Hauptsignal frei mit 40 km/h
VSgreen02	Grün rechts Mitte	Hauptsignal frei, Hauptsignal frei mit 60 km/h
VSyellow02	Gelb rechts oben	Vorsicht, Hauptsignal frei mit 40 km/h

3.3.2. Vorsignal mit Geschwindigkeitsvoranzeiger

Beschreibung	Child-Name
Vorsignal mit Geschwindigkeitsanzeiger (Child-Name muss diesen Namen enthalten!)	SchildVS

Node-Name	Beschreibung	Leuchtet bei Signalbegriff
VSgreen01	Grün links unten	Hauptsignal frei, Hauptsignal frei mit 60 km/h, Hauptsignal frei mit 40 km/h
VSyellow01	Gelb links oben	Vorsicht, Hauptsignal frei mit 60 km/h, Hauptsignal frei mit 40 km/h
VSgreen02	Grün rechts Mitte	Hauptsignal frei, Hauptsignal frei mit 60 km/h
VSyellow02	Gelb rechts oben	Vorsicht, Hauptsignal frei mit 40 km/h
Im Skript: ZS3V_SIGNAL_HEAD_NAME = „ZS3v Signal Head“		

3.3.3. Vorsignal zum Schutzsignal

Beschreibung	Child-Name
Vorsignal mit Geschwindigkeitsanzeiger zeigt die Geschwindigkeiten vom nachfolgenden Schutzsignal an. (Child-Name muss diesen Namen enthalten!)	SchildVSH

Node-Name	Beschreibung	Leuchtet bei Signalbegriff
VSgreen01	Grün links unten	Hauptsignal frei, Hauptsignal frei mit 60 km/h, Hauptsignal frei mit 40 km/h
VSyellow01	Gelb links oben	Vorsicht, Hauptsignal frei mit 60 km/h, Hauptsignal frei mit 40 km/h
VSgreen02	Grün rechts Mitte	Hauptsignal frei, Hauptsignal frei mit 60 km/h
VSyellow02	Gelb rechts oben	Vorsicht, Hauptsignal frei mit 40 km/h
Im Skript möglich: ZS3V_SIGNAL_HEAD_NAME = „ZS3v Signal Head“		

3.3.4. Vorsignalnachahmer

Vorsignalnachahmer benötigen immer einen VR_CODE sowie den vorgegebenen Child-Namen.

Beschreibung		Child-Name
Vorsignalnachahmer	(Child-Name muss diesen Namen enthalten!)	SchildSNA

Node-Name	Beschreibung	Leuchtet bei Signalbegriff	VR_CODE
Snawhite02	Balken aufsteigend	Hauptsignal zeigt frei	4
Snawhite03	Balken waagerecht	Hauptsignal zeigt Halt	8
Snawhite04	Balken absteigend	Hauptsignal zeigt frei mit Geschwindigkeitsbeschränkung	16
Snawhite05	Ersatzsignalankündigung	Hauptsignal zeigt Ersatzsignal	32
		Code-Summe:	60

In diesem Beispiel wurde der senkrechte Balken nicht über den Skript angesteuert, da es ohnehin immer leuchten wird.

Wird der Vorsignalnachahmer mit einem Gleismagnet für die Zugbeeinflussung ausgerüstet, so wird das gleiche Skriptmodul wie beim Vorsignal verwendet.
Aus diesem Grunde gibt es zwei unterschiedliche Skripte („OEBB SNA.lua“ und „OEBB SNA 1000.lua“).

OEBB SNA.lua	-> Ohne Ansteuerung eines Gleismagneten	Modul SNA
OEBB SNA 1000.lua	-> Mit Ansteuerung eines Gleismagneten	Modul VS

Weiterhin kann der Vorsignalnachahmer mit einem Schutzsignal kombiniert werden.

OEBB SHSNA.lua	-> Schutzsignal mit Vorsignalnachahmer	Modul SH

3.4. Hauptsignale (einige Beispiele)

Auf Grund der Vielzahl an Möglichkeiten folgen nur einige Beispiele für die Hauptsignale. Es folgt nochmals der Hinweis, dass der Child-Name bei den Hauptsignalschirmen immer gleich ist und „HS“ lautet. Lediglich die Code-Summe wird für unterschiedliche Kombinationen an verwendeten Signaloptiken geändert.

Bei allen Hauptsignalen muss im Skript in der Variable HP_CODE der Wert für die Code-Summe der verwendeten Signaloptiken angegeben werden. Weiterhin muss in der Variable HP_SIGNAL_HEAD_NAME der Child-Name des Hauptsignalschirmes angegeben werden.

Zusätzlich muss bei Hauptsignalen mit Vorsignalschirm an einem Mast im Skript noch die Variable VR_SIGNAL_HEAD_NAME = ““
aktiviert werden. Ein spezieller Wert wird nicht angegeben.

3.4.1. Hauptsignal

1. Beispiel	Child-Name
Hauptsignal mit Ersatzsignal ohne Verschubsignal	SchildHS

Node-Name	Beschreibung	Leuchtet bei Signalbegriff	HP_CODE
HSred01	rotes Signallicht	Halt, Ersatzsignal	2
HSgreen01	grünes Signallicht	Fahrt frei	16
ZS1	Kleines weißes Signallicht	Ersatzsignal	8192
		Code-Summe:	8210

2. Beispiel	Child-Name
Hauptsignal mit Vorsichtssignal ohne Verschubsignal	SchildHS

Node-Name	Beschreibung	Leuchtet bei Signalbegriff	HP_CODE
HSred01	rotes Signallicht	Halt, Ersatzsignal, Vorsichtsignal	2
HSgreen01	grünes Signallicht	Fahrt frei	16
ZS7	Kleines rotes Signallicht	Vorsichtsignal	16384
		Code-Summe:	16402

3. Beispiel	Child-Name
Hauptsignal mit Verschubsignal	SchildHS

Node-Name	Beschreibung	Leuchtet bei Signalbegriff	HP_CODE
HSred01	rotes Signallicht	Halt, Verschubsignal	2
HSgreen01	grünes Signallicht	Fahrt frei	16
HSyellow01	gelbes unteres Signallicht	Fahrt frei mit 40 km/h	128
HSwhite02	Verschubsignal einzeln links tiefer	Verschubsignal	512
HSwhite03	Verschubsignal einzeln rechts höher	Verschubsignal	1024
		Code-Summe:	1682

3.4.2. Hauptsignal mit Geschwindigkeitsanzeiger

1. Beispiel	Child-Name
Hauptsignal mit Vorsignal, Ersatzsignal, Verschubsignal und Geschwindigkeitsanzeiger / -voranzeiger	SchildHS SchildVS

Node-Name	Beschreibung	Leuchtet bei Signalbegriff	HP_CODE
HSred01	rotes Signallicht	Halt, Verschubsignal, Ersatzsignal	2
HSgreen01	grünes Signallicht	Fahrt frei	16
HSwhite02	Verschubsignal einzeln links tiefer	Verschubsignal	512
HSwhite03	Verschubsignal einzeln rechts höher	Verschubsignal	1024
ZS1	Ersatzsignal (weiß)	Ersatzsignal	8192
Im Skript:	ZS3_SIGNAL_HEAD_NAME = „ZS3 Signal Head“		
		Code-Summe:	9746

Node-Name	Bezeichnung		VR_CODE
VSgreen01	Grün links unten		16
VSyellow01	Gelb links oben		32
VSgreen02	Grün rechts Mitte		64
VSyellow02	Gelb rechts oben		128
Im Skript:	ZS3v_SIGNAL_HEAD_NAME = „ZS3v Signal Head“		
		Code-Summe:	240

Werden feste Zusatzanzeiger verwendet, so müssen die Variablen der Zusatzanzeiger angepasst werden:

Im Skript: ZS3_SIGNAL_HEAD_NAME = „ZS3 Form Head“ ZS3v_SIGNAL_HEAD_NAME = „ZS3v Form Head“	

Die Zusatzanzeiger können in beliebigen Kombinationen mit Aktivierung oder Deaktivierung verwendet werden.

2. Beispiel	Child-Name
Hauptsignal (Rot, Grün, Grün, Gelb) mit Abfahrsignal, sowie Zusatzanzeiger	SchildHS

Node-Name	Beschreibung	Leuchtet bei Signalbegriff	HP_CODE
HSred01	rotes Signallicht	Halt, Verschubsignal, Ersatzsignal	2
HSgreen01	grünes Signallicht	Fahrt frei	16
HSgreen02	grünes unteres Signallicht	Fahrt frei mit 60 km/h	64
HSyellow01	gelbes unteres Signallicht	Fahrt frei mit 40 km/h	128
HSwhite04	Abfahrsignal	Abfahrt	256
Im Skript:	ZS3_SIGNAL_HEAD_NAME = „ZS3 Signal Head“		
		Code-Summe:	466

3.5. Verschubsignal

Verschubsignale werden ohne Child gebaut. In der Mastgeometrie sitzen somit alle Licht-Nodes und werden direkt angesteuert.

Beispiel
Verschubsignal

Node-Name	Beschreibung	Leuchtet bei Signalbegriff
ZwSWhite01	Weiß rechts oben	Verschubfahrt erlaubt
ZwSWhite02	Weiß links Mitte	Verschubfahrt verboten
ZwSWhite03	Weiß rechts Mitte	Verschubfahrt verboten
ZwSWhite04	Weiß links unten	Verschubfahrt erlaubt

3.6. Zusatzsignale und Zusatzanzeiger

Sämtliche Zusatzsignale und Zusatzanzeiger schalten ausschließlich Texturen. Deshalb gibt es hier keine Licht-Nodes, sondern es muss lediglich auf die richtige Verwendung des Child-Namens geachtet werden.

Alle folgenden Zusatzanzeiger sind separate Signalobjekte als eigenes BluePrint und mit eigenen Links, die später in vorgesehener Weise in der Nähe der Hauptsignale platziert werden können.

Es gibt immer 2 Varianten. Eine normale mit Basisgeometrie und einem Child und eine Variante ohne Child nur mit einer Basisgeometrie.

3.6.1. Richtungsanzeiger

Child-Name	Beschreibung
ZS2v Signal Head	Licht - Richtungsvoranzeiger
ZS2v	Licht - Richtungsvoranzeiger (ohne Child im BluePrint)
ZS2 Signal Head	Licht - Richtungsanzeiger
ZS2	Licht - Richtungsanzeiger (ohne Child im BluePrint)

Die Texturen werden über die Eintragungen im Bereich PrimaryNamedTextureSet geladen.
Die Buchstaben für die Texturen können frei gewählt werden.

3.6.2. Geschwindigkeitsanzeiger

Child-Name	Beschreibung
Zs3v Signal Head	Licht - Geschwindigkeitsvoranzeiger
Zs3v	Licht - Geschwindigkeitsvoranzeiger (ohne Child im BluePrint)
Zs3 Signal Head	Licht - Geschwindigkeitsanzeiger
Zs3	Licht - Geschwindigkeitsanzeiger (ohne Child im BluePrint)
Zs3v Form Head	Form - Geschwindigkeitsvoranzeiger
Zs3v	Form - Geschwindigkeitsvoranzeiger (ohne Child im BluePrint)
Zs3 Form Head	Form - Geschwindigkeitsanzeiger
Zs3	Form - Geschwindigkeitsanzeiger (ohne Child im BluePrint)

Die Texturen werden über die Eintragungen im Bereich PrimaryNamedTextureSet geladen.

Geschwindigkeitsanzeiger und Geschwindigkeitsvoranzeiger können in ein Signal-BluePrint von einem Hauptsignal integriert werden. Hierzu muss im BluePrint das Child mit eingetragen und im Skript die Variable des Child-Namens aktiviert werden.

Form - Geschwindigkeitsanzeiger bzw. Form - Geschwindigkeitsvoranzeiger besitzen in der Regel keinen separaten Kopf (Variante ohne Child im BluePrint). Die Texturen für die Zahlen werden dennoch im Bereich PrimaryNamedTextureSet vereinbart.

3.6.3. Geschwindigkeitsanzeiger für Schutzsignale

Da Schutzsignale auf ihrem Signalschirm keine Geschwindigkeiten signalisieren, wie es bei einen Hauptsignal für 40 bzw. 60 km/h möglich ist, benötigen Schutzsignale ein spezielles Modul zur Verwendung als separater Geschwindigkeitsanzeiger.

Im Skript des Geschwindigkeitsanzeigers oder Geschwindigkeitsvoranzeigers wird dann das Modul ZsSH geladen.

4. Skripte

Für jede Kombination aus Signalschirmen und Zusatzanzeigern muss genau ein Skript im Verzeichnis Skripte unter Ihrem Source-Verzeichnis existieren.
Es wurden ja nur Beispielkombinationen mitgeliefert. Nun liegt es an Ihnen, nach persönlichem Geschmack Skriptnamen festzulegen. Durch die Verwendung des BluePrint-Editors wird beim Export ein gleichnamiger Skript mit der Dateiendung „out“ im entsprechenden Ordner unter Assets erstellt. Dieser Skript benötigt dann die Module und die Optionsdatei im Unterordner bei Schuster/Freeware, damit das Signal funktionieren kann.

Es können also auch die mitgelieferten Skriptnamen geändert werden. Lediglich der Inhalt der Skripte folgt den speziellen Vorgaben, die weiter oben genannt wurden. So darf auch das unter „require“ zugeordnete Modul inklusive der Pfadangabe keinesfalls geändert werden.

Im Skript wird je nach Signalschirm der errechnete HP_CODE und / oder der VR_CODE hinter der entsprechenden Variablen angegeben.

Auf den nächsten Seiten folgen Tabellen mit den im Skriptpaket vorhandenen Skripten.

Bemerkungen:

Kasten = Es wird ein 3D-Objekt ohne Licht-Nodes in Form eines Gehäuse benötigt
Anbau = Spezielles Signal zur Realisierung eines Hauptsignals an einer DKW

Signaltyp	Skript-Name	Child-Name				Bemer-kung
		Haupt-signal	Vor-signal	Zusatzanzeiger HS	Zusatzanzeiger VS	
Vorsignale						
Vorsignal	OEBB VS.lua		SchildVS			
Vorsignal mit Geschwindigkeitsvoranzeiger	OEBB VS_GAV.lua		SchildVS		ZS3v Signal Head oder ZS3v Form Head	
Vorsignal für Schutsignale	OEBB VSH.lua		SchildVSH			
Vorsignal für Schutsignale mit Geschwindigkeitsvoranzeiger	OEBB VSH_GAV.lua		SchildVSH		ZS3v Signal Head oder ZS3v Form Head	
Vorsignalnachahmer						
Vorsignalnachahmer	OEBB SNA.lua		SchildSNA			
Vorsignalnachahmer mit Zugbeeinflussung	OEBB SNA 1000.lua		SchildSNA			
Hauptsignale						
Blocksignal	OEBB BS.lua	SchildBS				
Hauptsignal	OEBB HS.lua	SchildHS				
Hauptsignal mit Geschwindigkeitsanzeiger	OEBB HS Speed.lua	SchildHS		ZS3 Signal Head oder ZS3 Form Head		
Hauptsignal mit Vorsignal	OEBB HS VS.lua	SchildHS	SchildVS			
Hauptsignal mit Vorsignal und Geschwindigkeitsanzeiger	OEBB HS VS Speed.lua	SchildHS	SchildVS	ZS3 Signal Head oder ZS3 Form Head	ZS3v Signal Head oder ZS3v Form Head	Zusatzanzeiger in jeder Kombi-nation möglich
Schutzsignale						
Schutzsignal	OEBB SH.lua	SchildHS				
Schutzsignal mit Vorsignalnachahmer	OEBB SHSNA.lua	SchildHS	SchildSNA			

Verschubsignale						
Verschubsignal	OEBB ZwS.lua					ohne Child
Sonderbauformen						
Vorsignal – Dummy	OEBB VS Dummy.lua					Kasten
Vorsignalnachahmer – Dummy	OEBB SNA Dummy.lua					Kasten
Hauptsignal – Dummy	OEBB HS Dummy.lua					Kasten
Hauptsignal – Ohne Signalmalst	OEBB HS OSS.lua					Kasten
Blocksignal als Anbauvariante (speziell für DKW-Bereiche)	OEBB BS Anbau.lua	SchildBS				Anbau
Hauptsignal als Anbauvariante (speziell für DKW-Bereiche)	OEBB HS Anbau.lua	SchildHS				Anbau
Hauptsignal mit Vorsignal als Anbauvariante (speziell für DKW-Bereiche)	OEBB HSVS Anbau.lua	SchildHS	SchildVS			Anbau
Verschubsignal als Anbauvariante (speziell für DKW-Bereiche)	OEBB ZwS Anbau.lua	„“				Anbau
Zusatzsignale						
Fahrtanzeiger (einfach oder doppelt)	OEBB FAZ.lua			FAZ Signal Head		
Fahrtanzeiger (einfach oder doppelt)	OEBB FAZ.lua			FAZ		ohne Child
Richtungsanzeiger	OEBB Zs2.lua			ZS2 Signal Head		
Richtungsanzeiger	OEBB Zs2.lua			ZS2		ohne Child
Richtungsvoranzeiger	OEBB Zs2v.lua				ZS2v Signal Head	
Richtungsvoranzeiger	OEBB Zs2v.lua				ZS2v	ohne Child
Geschwindigkeitsanzeiger	OEBB Zs3.lua			ZS3 Signal Head		
Geschwindigkeitsanzeiger	OEBB Zs3.lua			ZS3		ohne Child
Geschwindigkeitsvoranzeiger	OEBB Zs3v.lua				ZS3v Signal Head	
Geschwindigkeitsvoranzeiger	OEBB Zs3v.lua				ZS3v	ohne Child

5. BluePrintEditor

Um das 3D-Modell mit den Skript zu verbinden, ist der BluePrintEditor erforderlich. Hier werden einige Einstellungen für das Signal vorgenommen. Aus den erstellten Signalschirmen und Zusatzanzeigern kann nun unter Auswahl des richtigen Signalskriptes ein funktionierendes Signal erstellt werden.

Dieses Thema kann nur angerissen und nicht erschöpfend beschrieben werden. Deshalb gibt es hier nur einige Hinweise, die bei speziellen Bauarten von Signalschirmen auch noch abweichen können.

5.1. Grundsätzliches

Kenntnisse zur Benutzung des BluePrintEditors setze ich voraus. Ich werde nur auf die relevanten Einstellungen für die Funktion des 3D-Modells mit dem Skript eingehen.

5.2. Köpfe

Für alle Signalschirme und Zusatzanzeiger muss vorab ein AnimSceneryBluePrint erstellt werden.

Später werden diese Signalschirme und Zusatzanzeiger in einem SignalBluePrint zusammengefügt.

Die 3D-Modelle der Signalschirme bzw. Zusatzanzeiger-Objekte werden zuerst als ein separater „Kopf“ im BluePrintEditor definiert. Hierfür wird in der Regel das AnimSceneryBluePrint verwendet.

Im BluePrint wird im Bereich BrowseInformation unbedingt die **Category** „**eExcludeFromBrowserList**“ angegeben, damit der Kopf später im Editor nicht auswählbar ist, sondern verborgen bleibt.
Ein Name muss bei allen BluePrints angegeben werden. Jedoch DisplayName und Beschreibung können entfallen.

Ein weiterer wichtiger Eintrag ist die **Geometrie** im Bereich **RenderComponent**, in der der Signalschirm bzw. der Zusatzanzeiger angegeben wird.

Lediglich bei den Form-Geschwindigkeitsanzeigern gibt es keinen separaten Kopf, damit zusätzlich Zahlen (im ID-Feld eingetragen) direkt im Schild angezeigt werden können.

5.3. Einstellungen für Signale und Zusatzanzeiger

Zuerst wird ein neues BluePrint geöffnet. Wir wählen ein „**Signal BluePrint“** und nehmen sorgfältig alle notwendigen Eintragungen vor:

- ⓘ **Name**
  - **Name**

Ein eindeutiger Name muss bei allen BluePrints angegeben werden. Dieser wird in der Tracks.bin vermerkt und erscheint auch bei Fehlermeldungen.

- ⓘ **BrowseInformation**
  - **DisplayName**
  - **Description**

In dem Bereich DisplayName ist grundsätzlich die Sprache English und German mit einer passenden Bezeichnung auszufüllen. Die restlichen Sprachen werden wohl auf English verwiesen, wenn nichts ausgefüllt ist. Man sollte sich vorher ein gutes System überlegen, um die Signale später eindeutig identifizieren zu können.

Keinesfalls darf sich die Bezeichnung mit anderen Editorobjekten überschneiden, da bei gleichzeitiger Freischaltung ungewollte Ergebnisse entstehen.

Auch ein Eintrag im Feld Description kann später bei der Auswahl im Welteditor hilfreich sein, da diese Information als Tool-Tipp unterhalb des Mauszeigers angezeigt wird.

ⓘ **BrowseInformation**
- **Category**

Der Eintrag Category legt fest, in welcher Gruppe später im Welteditor das Signal gelistet werden soll. Wir verwenden für alle Signale die Auswahl: eTrackInfrastrukture

ⓘ _object
- **PrimaryNamedTextureSet**
- **SecondaryTextureSet**

Im Bereich PrimaryNamedTextureSet und SecondaryTextureSet werden die Textursets für die erste und zweite Zeile der Mastnummer, soweit vorhanden, angegeben. Ist keine Mastnummer vorhanden, können diese Angaben entfallen. Auch wenn diese Bereiche frei bleiben, können später die im ID-Feld eingetragenen Bezeichnungen zum Debug verwendet werden.

Bei allen **Zusatzanzeigern** wird im PrimaryNamedTextureSet das zu verwendende Texturset für die darzustellenden Zeichen angegeben.

ⓘ **_object**
- **GeometryID**

Im Eintrag „GeometryID“ wird die Mastgeometrie angegeben. Hierbei handelt es sich **nicht** um die Signalschirme. Diese werden später im Bereich _container als Children definiert.

ⓘ **_signal**
- **NumberOfTrackLinks**

Im Bereich _**signal** werden nun weitere Signaleigenschaften definiert. Dort findet sich als erster Eintrag NumberOfTrackLinks, welcher die Anzahl der Ziellinks zuzüglich des Link 0 beinhaltet.
Für ein Blocksignal (0T), alle Vorsignale und bei allen Zusatzanzeigern bis auf die „T“-Varianten für Zs2 und Zs6 wird hier der Wert 1 eingetragen. Bei allen Hauptsignalen, die Fahrstraßen über Weichen haben, zählen wir für jedes Ziel den Wert 1 hinzu. Liegt hinter dem Hauptsignal zum Beispiel eine Weiche, dann wird der Wert 3 in das Feld NumberOfTrackLinks eingetragen.

ⓘ **_signal**
- **Stopping**

In das Kästchen bei Stopping wird bei allen Hauptsignalen und Sperrsignalen ein Häkchen gesetzt. Hierdurch wird später ein neuer Blockabschnitt, beginnend am Link 0, gebildet.

Dies ist die einzige Eigenschaft, die KI-Züge vom Signal kennen und darauf auch reagieren.

ⓘ **_signal**
- **JunctionSignalType**

Bei der Auswahlliste JunctionSignalType sollte bei allen Signalen der Wert **eJunctionTypSpeed** ausgewählt werden, um später im Signal-Flyout die Eingabemöglichkeit der Richtungsangabe und der Geschwindigkeit zu haben.

ⓘ **_signal**
- **ControlMode**

Die Auswahl im Bereich ControlMode ist für die normale Funktion weniger von Bedeutung. Wird jedoch beabsichtigt, das Signalsystem gemeinsam mit der von mir erstellen 2DMapPro zu nutzen, dann sollte folgende Einstellung vorgenommen werden:

Hauptsignale ohne Vorsignalfunktion:	eControlModeControlled
Mehrabschnittsignale:	eControlModeControlledCallOn
Alle anderen Signale:	eControlModeAutomatic

Nur mit diesen Einstellungen werden dann die passenden Mastsymbole in der 2DMapPro angezeigt.

ⓘ **_script**

Nun kommt im Bereich _script die wichtige Entscheidung, welcher Skript zu verwenden ist. Wie schon weiter oben geschrieben, gibt es in der Regel für eine bestimmte Kombination aus Signalschirmen immer nur einen einzigen Skript, der zu verwenden ist. Schauen Sie also genau in der unter Punkt 4. aufgeführten Tabelle nach, welcher Skript zu ihrem Signal passt. Ist dieser nicht vorhanden, dann kopieren Sie am besten einen vorhandenen Skript und passen dessen Einstellungen durch Aktivierung von Variablen an und tragen den jeweils gültigen CODE für den oder die Signalschirme ein.

Der dort einzutragende Skript wird mit dem Verzeichnisnamen oberhalb von Provider- Produktnamen und ohne Dateiendung angegeben.

Ändern Sie keinesfalls den Verweis auf das dazugehörige Modul!

Beispiel für ein einfaches Hauptsignal: RailNetwork\Signals\German OEBB\Skripte\OEBB HS

Der Skript muss als Datei im Verzeichnis Source existieren, sonst schlägt der Export fehl. Im oben genannten Beispiel also genau diese Datei:

Source\[Provider]\[Produkt]\RailNetwork\Signals\German OEBB\Skripte\OEBB HS.lua

Die Werte für [Provider] und [Produkt] entsprechen Ihren Verzeichnisnamen.

ⓘ **_container**
- **Children**

Erst jetzt, im Bereich _container -> Children werden die Einstellungen, die die einzelnen Signalschirme und Zusatzanzeiger betreffen, festgelegt.
Für jeden Signalschirm bzw. Zusatzanzeiger muss ein Child hinzugefügt werden (Add Element…).
Der Übersicht halber beginnen wir immer mit dem Hauptsignal. Als 2. Child dann das Vorsignal (wenn vorhanden) und zum Schluss die Zusatzanzeiger (wenn vorhanden).

Im Feld ChildName wird der ChildName aus der Tabelle (Punkt 3.3) für den entsprechenden Signalschirm eingetragen. Hierbei ist auf die korrekte Schreibweise zu achten.

Es kann immer nur ein Zs3 und/oder Zs3v Zusatzanzeiger hinzugefügt werden. Die gleichzeitige Verwendung von **Licht- und Form-** Geschwindigkeits- bzw. voranzeigern ist nicht möglich.

ⓘ **_container**
- **Children**
  - **BlueprintID**

An dieser Stelle wird die XML-Datei vom Signalschirm eingetragen.
Die Einstellungen hierzu sind bei den Signalschirmen sehr unterschiedlich und werden deshalb nicht näher aufgeführt. Anzumerken ist jedoch, dass bei den Signalschirmen die Category**: eExcludeFromBrowserList** auszuwählen ist und in diesem BluePrint kein Skript angegeben wird.

ⓘ **_container**
- **Children**
  - **Matrix**

Die Matrix ermöglicht es, den Signalschirm am Mast entsprechend zu positionieren.

5.4. Export des BluePrints

Im unteren Bereich sollte man sich unbedingt den Bereich Output aufklappen [+], damit man die Meldungen verfolgen kann.
Wenn alles richtig eingegeben wurde, wird das BluePrint mit der Funktionstaste F7 nach Assets exportiert.
Nur wenn am Ende des Protokolls „Export Succeded“ steht, ist der Export erfolgreich abgeschlossen.
Anderen falls muss man die Einstellungen korrigieren.

Anschließend sollte das Signal im Welteditor des Train Simulators zur Verfügung stehen.

5.5. Einsatz der Signale in einer Train Simulator Strecke

Wenn Sie das Signal in eine Strecke einbauen, muss das Signal nach dem Neustart der Strecke das richtige Signalbild, zum Beispiel Hp0 anzeigen. Leuchten auf dem Signalschirm dann noch alle Signallichter, so gibt es ein Problem, welches an falschen Einstellungen liegen kann.

Prüfen Sie dann folgende Dinge:

- Starten Sie LogMate mit dem TS und kontrollieren Sie, ob dort Fehler angezeigt werden
- Passt der Skript auch zu den verwendeten Signalschirmen, die im BluePrint verwendet wurden?
- Sind alle Child-Namen richtig geschrieben?
- Wurde die richtige Code-Summe berechnet und im Skript eingetragen?
- Prüfen Sie das 3D-Modell, ob die Licht-Node-Namen stimmen

Ich hoffe, dass ich das Thema hinreichend ausführlich und richtig beschrieben habe.
Sollte es Probleme geben, bitte ich darum, im Forum bei einem geeigneten Thema nachzufragen.
So können Fragen schnell geklärt werden und die Lösungen stehen Vielen zur Verfügung.
Danke.

Ich wünsche viel Erfolg beim Einsatz der Skripte
Mathias Gundlach
