Einen eigenen Trigger mit Funktion erstellen

Hier einmal ein Lösungsvorschlag für einen Trigger der in beide Richtungen ausgelöst werden soll.

Die config.txt muss für ein Objekt der Kategorie ?Trackside? sein (kind ?scenery? mit Eintrag: ?trackside?). Außerdem muss für die Trigger-Funktion der Eintrag ?trigger 1?, sowie ?surveyor-only 1?, eingefügt werden. Wie genau das aussieht wird gleich gezeigt. Näher darauf eingehen werde ich nicht. Ich setze beim Scripting voraus, daß man in der Lage ist eine config.txt anzulegen. Eine Hilfe kann dabei der CCP (Content Creator Plus) sein.

Als Vorlage für die config.txt habe ich eine vor mir genommen. Die Einträge müssen natürlich individuell angepasst werden.

Vorbereitung der config.txt

Code: config.txt

```
kuid
username
trackside
trainz-build
kind
category-era
category-region
author
organisation
contact-email
contact-website
category-class
script
class
trigger
surveyor-only
mesh-table
{
    default
   {


    }
}
thumbnails
{
    0
    {



    }
}
```

Alles anzeigen

Das Scripting
Nun wäre es fast schon an der Zeit zu überlegen, wie wir das ganze realisieren können.

Ich denke, wir schreiben uns vorher aber unser obligatorisches Rahmenscript:

Dateiname (wie in der config.txt angegeben): trigger_zweiseitig.gs

Code: trigger_zweiseitig.gs

```
include                                                          "Trackside.gs"

class               C2SidedTrigger               isclass               Trackside
{
                                                                 public                 voi
    {
        inherited(pAsset);
    }
};
```

Wie kann man nun feststellen, daß ein Zug einen Trigger auslöst?
Die einzige Möglichkeit dies zu bewerkstelligen ist es, die Nachrichten abzufangen, die die Objekte in Trainz sich diesbezüglich schicken. Aber welche sind das?

Ein Blick auf ?Class Trigger – TrainzOnline? verrät uns, daß wir die Nachricht ?Object, Enter? im Trigger abfangen müssen, um tätig werden zu können.

Bauen wir doch diesen sog. Message-Handler ein. Ein Message-Handler ist wie ein Männelein, daß im Computer sitzt. Immer, wenn die Nachricht, die es abfangen soll, kommt, löst dieses Männelein eine Funktion aus, die wir selbst definieren. Die Funktion muss als einzigen Parameter ein Objekt des Typs ?Message? haben. Damit können wir nicht nur abfragen, ob unser Trigger ausgelöst worden ist, sondern auch noch:

- Wer hat den Trigger ausgelöst
- Welcher Trigger wurde ausgelöst (Wird später noch wichtig!)

Schreiben wir nun unser Script dementsprechend um. Um eine Nachricht abfangen zu können brauchen wir nur einen Handler in die Init-Methode zu setzen, der die Informationen enthält welche Nachricht abgefangen werden soll. Das ist die Funktion ?AddHandler?.

Schauen wir uns das einfach mal an:

Code: trigger_zweiseitig.gs

```
include                                                                 "Trackside.gs"
class               C2SidedTrigger              isclass             Trackside
{
                                                                        void

                                                                public                  voi
    {
         inherited(pAsset);


                                                  AddHandler(me,     "Object",     "Enter",
     }


     {
     }
 };
```

Alles anzeigen

Dieses Script macht nichts anderes, als daß es auf die Nachricht: ?Object, Enter? reagiert.
Bekommt das Script eine solche Nachricht, wird die Funktion ?TriggerMessageHandler? aufgerufen.
(siehe AddHandler, Zeile 012). Der erste Parameter ?me? ist ein Verweis auf das aktuelle Objekt, bzw. auf die aktuelle Instanz der Klasse ?C2SidedTrigger?. Das sagt der AddHandler-Methode, dass der MessageHandler für das aktuelle Objekt gesetzt wird. Wir können so auch Nachrichten für andere Objekte abfangen. Das soll aber hier nicht Thema sein

Die Funktion ?TriggerMessageHandler? ist noch leer. Das ändern wir jetzt. Wichtig ist nämlich, daß wir noch einmal abfragen, ob die Nachricht für den aktuellen Trigger ist. Das ?me? in AddHandler sagt nämlich nicht aus, daß nur Nachrichten abgefangen werden, die an das aktuelle Objekt geschickt werden.
Außerdem gibt es die Möglichkeit eine solche Nachricht an alle Objekte auf der Strecke zu senden (broadcasting). Das ist aber auch nicht Thema. Wichtig ist eben nur, daß wir diese Abfrage einbauen.

Damit der Code übersichtlich bleibt, ist es von Vorteil für die Prozesse, die wir durchführen, wenn wir erkannt haben, daß ein Zug den Trigger ausgelöst hat, eine eigene Methode zu schreiben, die heißen könnte: ?TriggerEvent?. Als Parameter könnten wir ihr ein GameObject geben, das das auslösende Objekt enthält. Also ändern wir unser Script wie folgt:

Code: trigger_zweiseitig.gs

```
include                                                                 "Trackside.gs"
class               C2SidedTrigger              isclass                 Trackside
{
                                                                                void
                                                                    void                TriggerE
                                                                       public                   voi
    {
        inherited(pAsset);

                                                AddHandler(me,      "Object",      "Enter",
    }
                                                                                  void
    {
                                    //    Abfragen,    ob    die    Nachricht    für    dieses    Ob
                                        //    Wenn    nicht,    die    Funktion    mit    "retur

                                                    //      Die      Funktion      wurde      noch
                                       //    Ein    Zug    hat    tatsächlich    diesen    Trigge
                                                                  //       Dann       die       Me
        // Der Member "src" aus der Message-Klasse enthält einen Verweis auf das
  // GameObject des auslösenden Objekts. Also genau passend für unser TriggerEvent
        TriggerEvent(pMsg.src);
    }

                                                                          void                Trigge
    {
              //  TODO:  Hier  kannst  du  deine  gewünschten  Funktionen  einbauen,  die
                      //   abgearbeitet   werden,   wenn   ein   Zug   diesen   Trigger   ausl
  // Ich habe mich für eine einfache Nachricht ins Nachrichtenfenster entschieden,
//die uns mitteilt, welche Identifikationsnummer das GameObject des auslösenden Zuges hat
                                                    Interface.Print("Train-ID:      "      +
    }
};
```

Alles anzeigen

Die Kommentare im Quellcode erklären eigentlich ziemlich alles genau.
Darum hier nochmal eine Übersicht über den Ablauf des Scripts:

Init: Das Objekt wird initialisiert, es wird ein Message-Handler mit AddHandeler eingerichtet.
Die Nachricht: ?Object, Enter? wird geschickt, sobald ein Objekt in den Trigger-Radius einfährt.
Danach wird sofort überprüft, ob der es sich überhaupt um den jeweiligen Trigger handelt, wenn ja, wird die Funktion ?TriggerEvent? aufgerufen, die dann den Code abarbeitet, der die gewünschte Funktion nach dem Auslösen bringt.

Damit wäre der beidseitige Trigger fertig.
Dieses Beispiel gibt es hier einmal als fertiges Objekt: