

## Lua Scripting Funktion (Function) erstellen LUA

### LUA Erstellen einer Funktion (Function)

Was ist eine Funktion in LUA?

Da Lua eine Programmiersprache ist, besteht diese aus programmierbarem Code.

In eine Funktion können wir einen Codeblock zusammen mit den neuen TEN Syntax Befehlen setzen, den wir dann so oft wir möchten ausführen können. Wir können damit also quasi unsere eigenen "TRNG Action, Fliepeffect und Condition Trigger" schreiben, nur viel einfacher und komfortabler.

In diesem Tutorial werde ich nicht darauf eingehen können, wie man in LUA grundsätzlich programmiert. Kubsy hat darüber ein [Basis Tutorial](#) geschrieben. Die neuen TEN Syntax Befehle kann man in der [Dokumentation](#) von Squidshire nachlesen.

Es gibt zwei Arten von Funktionen:

- Funktionen, die wir innerhalb eines LUA Scriptes setzen und mit .ON Funktionen aktivieren können.
- Funktionen, die wir in die neuen Volume Trigger des Tomb Editors setzen können und somit im Level durch Lara, Gegner, Kameras usw. aktiviert werden.

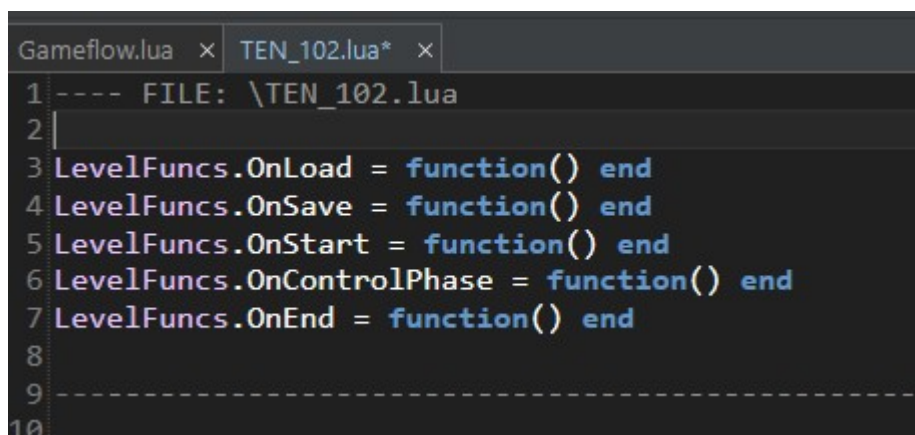
#### Anmerkung:

Diese Funktionen, die wir in Volume Trigger setzen, erkläre ich in einem anderen Tutorial.

In welcher LUA Datei schreiben wir diese Funktionen?

Beim Erstellen eines neuen Levels haben wir auch eine Datei mit dem Namen erstellt, die wir unter "Level1.scriptfile=" vergeben haben.

In diese Datei schreiben wir alle unsere Funktionen. Wir verwalten hier also unsere komplette Sammlung von selbstgeschriebenen "Triggern", egal ob wir diese später innerhalb des Scriptes oder in einem Volume Trigger aktivieren möchten.



```
Gameflow.lua x TEN_102.lua* x
1 ---- FILE: \TEN_102.lua
2 |
3 LevelFuncs.OnLoad = function() end
4 LevelFuncs.OnSave = function() end
5 LevelFuncs.OnStart = function() end
6 LevelFuncs.OnControlPhase = function() end
7 LevelFuncs.OnEnd = function() end
8
9 -----
10
```

ellen eines neuen Levels vorher

Was bedeuten diese Standardfunktionen?

Ich vergleiche sie mit den GlobalTriggern der TRNG Engine, da wir hier Funktionen z.B. gleich beim Starten eines Levels aktivieren können.

Hier mal die einzelnen Erklärungen:

**.OnLoad** = Funktionen, die wir hier hineinsetzen, werden beim Laden eines Savegames ausgeführt.

**.OnSave** = Funktionen, die wir hier hineinsetzen, werden beim Speichern eines Savegames ausgeführt.

**.OnStart** = Funktionen, die wir hier hineinsetzen, werden beim Starten eines Levels ausgeführt.

**.OnControlPhase** = Funktionen, die wir hier hineinsetzen werden 30-mal in der Sekunde überprüft (30 FPS)

**.OnEnd** = Funktionen, die wir hier hineinsetzen, werden beim Verlassen des Levels ausgeführt ( z.B. durch Laden, Levelwechsel oder zur Titel).

An der Schreibweise "LevelFuncs." erkennen wir, dass es sich hierbei um Funktionen handelt, die man direkt im Level aktivieren kann.

Als erstes praktisches Beispiel lösen wir einen Baddy1 sofort aus, wenn das Level startet.

Code

```
local baddy = GetMoveableByName( "firstbaddy" )  
baddy:Enable( )
```

Was bedeutet dieser Code?

**local baddy =**

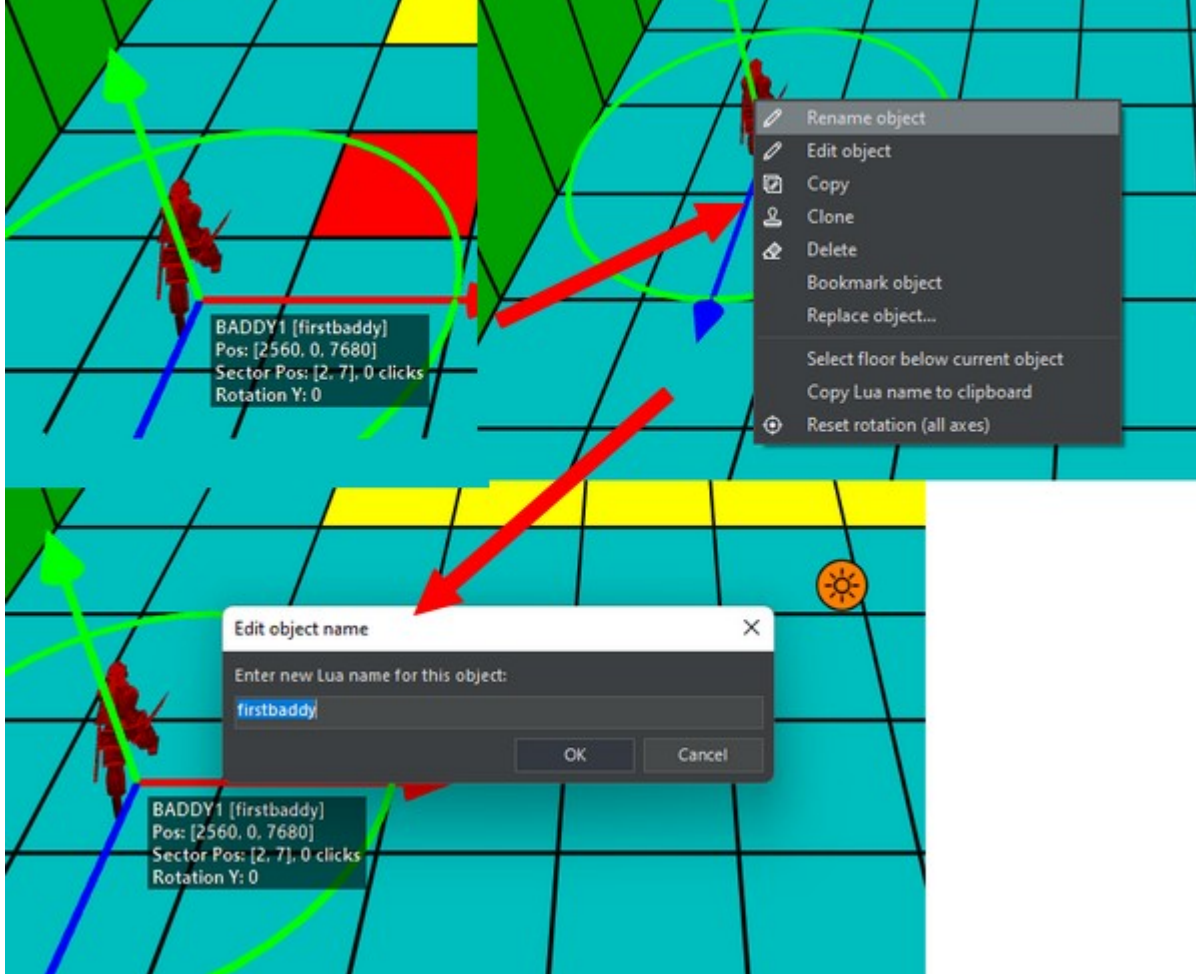
hier vergeben wir eine lokale Variabel und geben ihr den Namen baddy.

**GetMoveablebyName("firstbaddy")**

In diese Variabel **baddy** setzen wir einen Baddy1 mit dem Lua Namen "firstbaddy"

Anmerkung:

Diesen Namen finden wir, wenn wir das Objekt im TE anklicken. Ändern können wir diesen Namen indem wir da Objekt rechts anklicken und "Rename Object" anklicken.



### **baddy: Enable()**

Hier sagen wir, dass das Objekt, welches sich in der Variabel "baddy" befindet sofort aktiviert bzw. getriggert werden soll.

Wie schreiben wir nun diesen Code in die Funktionen?

Es gibt zwei Möglichkeiten:

1. Wir können den Code direkt zwischen "function()" und "end" reinschreiben.

Code

```
LevelFuncs.OnStart                                     = function()
local baddy                                             baddy= GetMoveableByName("firstbaddy")
    baddy:Enable()
end
```

```
LevelFuncs.OnStart = function()

local baddy= GetMoveableByName("firstbaddy")
    baddy:Enable()

end
```

2. Besser ist es allerdings, den Code in eine weitere Funktion zu schreiben und auf diese dann zu verweisen:

Wir erstellen also eine leere Funktion und vergeben einen Funktionsnamen (hier als Beispiel baddy1):

Code

```
function                                baddy1                                ( )
end
```

In diese Funktion setzen wir nun unseren Code:

Code

```
function                                baddy1                                ( )
local                                  baddy=                                GetMoveableByName("firstbaddy")
    baddy:Enable( )
end
```

Diese Funktion setzen wir weiter unten in unsere LUA Level Datei, nun müssen wir in der ON Funktion noch auf sie verweisen.

Dazu schreiben wir ein baddy1 () dazwischen, das sieht dann so aus:

Code

```
LevelFuncs.OnStart                    =                                function()
baddy1                                ( )
end
```

### Anmerkung:

Dadurch, dass wir unseren Codeblock in eigene Funktionen schreiben, bleibt das Script lesbarer, wenn wir nachher weitere Codeblöcke schreiben und hinzufügen. Ausserdem können wir durch den Funktionsnamen, den wir vergeben haben, immer wieder von neuem auf den Codeblock verweisen und ihn auch in andere .ON Funktionen setzen.

In unserer Level Lua Datei sieht das ganze dann in der Praxis so aus:

```

2
3
4
5
6 LevelFuncs.OnLoad = function() end
7 LevelFuncs.OnSave = function() end
8 LevelFuncs.OnStart = function()
9
10 baddy1 ()
11
12 end
13 LevelFuncs.OnControlPhase = function()end
14 LevelFuncs.OnEnd = function() end
15
16 -----
17
18 function baddy1 ()
19     local baddy= GetMoveableByName("firstbaddy")
20     baddy:Enable()
21 end
22
23
24
25
26

```

Somit haben wir also unseren ersten Code geschrieben und diesen in eine Funktion gesetzt, die wir dann durch die .ON Funktionen im Level auslösen.

Wie wir Funktionen erstellen, die wir in die neuen Volume Trigger setzen, erkläre ich im nächsten Tutorial.