41.10 LUA-Grundlagen - if- Bedeutung näher beleuchtet

Autor: Götz
Quelle: Mein EEP-Forum

if

Mir hat lange die Inspiration gefehlt.
Aber heute verspüre ich Lust, die if Bedingung genauer zu studieren.

if benutzt man, wenn etwas nur unter bestimmten Umständen passieren soll.

Falls das Hilfssignal "Fahrt" zeigt, gib den Weg für den Zug frei.

Quellcode

```
1    NotAus = 1
2    taktGeber = 0
3    hilfsSignal = 101
4    stellungFAHRT = 2 -- für aktuelle Signale, z.B. Schattenbahnhof Set / GBS etc.
5
6    function EEPmain()
7      taktGeber = taktGeber + 1
8      return NotAus
9    end
10
11   if EEPGetSignal(hilfsSignal) == stellungFAHRT then
12    -- stelleWeichen()
13    -- oeffneSignal()
14   end
```

Die Bedingung kann aus einem komplizierten Term bestehen, deshalb muss zur Kennzeichnung am Ende das Schlüsseleort then, zu Deutsch "dann" stehen.

In einer if Verzweigung können mehrere Anweisungen stehen. Daher weiß Lua nicht, welche der nachfolgenden Zeilen nur dann ausgeführt werden sollen, wenn die Bedingung erfüllt ist. Deshalb muss das Ende des if Blocks immer mit dem Schlüsselwort end gekennzeichnet werden. Auch dann, wenn nur eine einzige Anweisung direkt hinter das Schlüsselwort then geschrieben wird.

Quellcode

```
1    if EEPGetSignal(hilfsSignal) == stellungFAHRT then EEPSetSignal(1, 2)
2    end
```

Außerdem ist es bei jeder if Verzweigung sinnvoll, Überlegungen anzustellen was ist, wenn die Bedingung nicht erfüllt ist.

Falls das Hilfssignal "Fahrt" zeigt, gib den Weg für den Zug frei.
Ansonsten merk dir, dass ein Zug wartet.

Das englische Wort für "ansonsten" ist else

Quellcode

```
1    if EEPGetSignal(hilfsSignal) == stellungFAHRT then
2      -- stelleWeichen()
3      -- oeffneSignal()
4    else
5      -- setzeMerker()
6    end
```

Genau genommen muss mein HilfsSignal "Halt" zeigen, wenn der Merker gesetzt werden soll.
Ich will also nicht "ansonsten", sondern "ansonsten, falls ..."

Dafür benutzt man statt else das elseif

Quellcode

```
1  if EEPGetSignal(hilfsSignal) == stellungFAHRT then
2    -- stelleWeichen()
3  -- oeffneSignal()
4   elseif EEPGetSignal(hilfsSignal) = stellungHALT then
5    --setzeMerker()
6   end
```

Aber was machen wir in den Fällen, in denen das Hilfssignal weder Fahrt noch Halt zeigt?
Den Fall gibt es nicht?
Sehr gut! Das sollten wir ausnutzen:

Quellcode

```
1   NotAus = 1
2   taktGeber = 0
3  hilfsSignal = 101
4  stellungFAHRT = 2 -- für aktuelle Signale, z.B. Schattenbahnhof Set / GBS etc.
5  stellungHALT = 1
6
7  function EEPmain()
8  taktGeber = taktGeber + 1
9  return NotAus
10 end
11
12  if EEPGetSignal(hilfsSignal) == stellungFAHRT then
13  -- stelleWeichen()
14  -- oeffneSignal()
15  elseif EEPGetSignal(hilfsSignal) == stellungHALT then
16   -- setzeMerker
17  else
18  print("Ein Fehler ist passiert!")
19  NotAus = 0
20  end
```

Ich schreibe also eine Fehlermeldung in das Ereignisfenster und stoppe dann die Ausführung des Lua Skripts.
Und wenn ich die Fehlermeldung detaillierter schreibe, dann weiß ich später auch genau, wo welcher Fehler passiert ist.

Das Schema muss man nicht zwingend befolgen. Es soll nur zeigen, wie man sich Dinge zunutze machen.

Ich habe zudem manches in diesem Bespiel vereinfacht. Ich habe bewusst darauf verzichtet die if Verzweigung in eine Funktion einzubetten. Das muss man aber tun, wenn man diese Verzweigung im richtigen Augenblick verwenden will. Zum Beispiel dann, wenn ein Kontaktpunkt überfahren wird. Funktionen sind sozusagen das, was in Basic das GOTO ist. Eine Sprungadresse. Nur, dass Funktionen viel mehr können. Denn eine Funktion kehrt am Ende dorthin zurück, wo sie aufgerufen wurde. Und liefert wahlweise auch gleich die gewünschten Attribute mit.

Aber das ist ein anderes Thema.