41.9 Stilfragen

Stilfragen

Autor: Goetz
Quelle: Mein EEP-Forum

Mit meinem letzten Kapitel bin ich nicht wirklich glücklich.

Ich wollte zu viel auf einmal und habe ein Skript geschrieben, dass ich selbst sehr hässlich finde.

Ganz besonders stört mich, dass ich in der Funktion EEPSetSignal() berechne, welches Signal umgeschaltet wird. Das funktioniert zwar und Lua ist es völlig egal, ob ich die Berechnung im Funktionsaufruf durchführe oder zuvor. Aber den Code finde ich sehr unleserlich.

Meines Erachtens ist es übersichtlicher, wenn ich erst die Berechnung durchführe und in einer Variablen ablege. Beispielsweise so:

```
x = TaktGeber % 5 + 1
EEPSetSignal(AusfahrSignal[x],Halt)
```

Dann steht die Formel nämlich frei und ist leichter zu erkennen.

Alternativ könnte ich in der Formel auch gleich die Nummer des Ausfahrsignals bestimmen:

```
x = AusfahrSignal[TaktGeber % 5 + 1]
EEPSetSignal(x,Halt)
```

Im Ergebnis unterscheiden sich diese Varianten nicht. Es ist nur eine Stilfrage, wie man das schreibt. Aber guter Stil ist wirklich wichtig. Weil stilvolle Schreibweise einen lesbaren Code ergibt. Über Stilfragen können sich Programmierer prächtig streiten. Denn letztlich bleiben sie Geschmacksache. In schlechtem Stil geschriebener Code kann von Lua ebenso gut ausgeführt werden wie stilvoller Code. Aber die Lesbarkeit ist immens wichtig. Für einen selbst bei der Fehlersuche. Und erst recht für andere, die den Code nachvollziehen möchten.

Ich bin selber Anfänger und mache daher gerade in Stilfragen ärgerliche Fehler. Die stören Lua zwar nicht, führen aber zu solch unleserlichen Code-Beispielen wie dem aus dem letzten Kapitel.

Eine weitere Stilfrage wäre beispielsweise, ob ich die if … then Bedingung nutze um zu unterscheiden, wann ich meine Signale auf Halt oder Fahrt schalte. Ich könnte nämlich ebenso gut abwechselnd aus dem TaktGeber den Wert 1 oder 2 errechnen und diese Zahl in der Funktion EEPSetSignal() verwenden:

```
y = math.floor(TaktGeber/5) % 2 + 1
```

Wenn ich den Wert aus dem TaktGeber durch 5 teile, dann bekomme ich für die ersten Werte 0 bis 4 als Ergebnis eine Zahl, die kleiner als 1 ist. Die in Lua definierte Funktion math.floor() schneidet die Nachkommastellen einfach ab. Übrig bleibt 0. Bei den Werten 5 bis 9 erhalte ich als Ergebnis einen Wert zwischen 1 und 2. Durch math.floor() bleibt 1 übrig, weil wieder die Nachkommastelle abgeschnitten wird. Ich bekomme also immer eine ganze Zahl, die nur jedes fünfte Mal um 1 größer wird. Mit dem Modulo Operator sorge ich dafür, dass immer abwechselnd eine 0 oder 1 heraus kommt. Denn aus der 2 wird wieder 0, aus der 3 wird 1 aus der 4 wird wieder 0 und so weiter. Weil ich für das Signalbild aber die Werte 1 und 2 benötige, zähle ich zum Schluss noch 1 dazu.

So sähe die alternative Version der Funktion gebeHandzeichen() aus:

Quellcode

```
function gebeHandzeichen()
    x = TaktGeber % 5 + 1
    y = math.floor(TaktGeber/5) % 2 + 1
    EEPSetSignal(AusfahrSignal[x],y)

end
```

Meines Erachtens ist das viel übersichtlicher.

Apropos Stil:
Ich bin ein Freund lesbarer Namen für Variablen. Deshalb vermeide ich beispielsweise das
I=0 und I=I+1
aus dem Standard Skript und schreibe lieber
TaktGeber=0 und TaktGeber=TaktGeber+1

Trotzdem habe ich im obigen Beispiel als Variablennamen x und y verwendet. Der Grund dafür ist, dass ich diese Variablen nur für einen kurzen Moment brauche. Die sind sozusagen für den "Sofortverzehr".

Das kann ich noch weiter verbessern, indem ich diese Variablen im Skript als lokale Variablen definiere. Denn lokale Variablen existieren nur innerhalb der Funktion, in der sie definiert werden. Anschließend werden sie wieder "vergessen". Also gelöscht. Das spart Resourcen.

Um x und y zu lokalen Variablen zu machen, muss bei der ersten Verwendung das Schlüsselwort local vorangestellt werden. Also:
local x = TaktGeber % 5 + 1
und
local y = math.floor(TaktGeber/5) % 2 + 1

Komplett sieht das dann so aus:

Quellcode

```
function gebeHandzeichen()
    local x = TaktGeber % 5 + 1
    local y = math.floor(TaktGeber/5) % 2 + 1
   EEPSetSignal(AusfahrSignal[x],y)

end
```

Weil ich das x und y gleich an Ort und Stelle benutze, kann ich auch leicht nachvollziehen, wo es herkommt, was drin steht und wie es benutzt wird. In diesem Zusammenhang ist ein einfaches x oder y für mich lesbarer als ein langer Namen.

Bei globalen Variablen sieht es anders aus. Die benutzt man oft an ganz unterschiedlichen Stellen im Skript. Und dann wird es schwer im Kopf zu behalten, welche Variablen für welchen Zweck gedacht waren wenn man sie nur x, y oder z nennt.

Und zum Schluss noch ein letzter Satz zur Stilfrage:
Mit Leerzeichen kann man in Lua recht frei umgehen. Beispielsweise bei Berechnungen.
Ich kann wahlweise
TaktGeber=TaktGeber+1
oder
TaktGeber = TaktGeber + 1
schreiben.
Lua ist das egal.

Und ich gestehe, dass ich Mühe habe, mich zu entscheiden. Wenn ich hier eine einzelne Berechnung erläutere, dann verwende ich gerne die zusätzlichen Leerzeichen, weil ich denke, dass es der Lesbarkeit dient. Im Code selber verzichte ich eher darauf.

Bei den Parametern in Funktionsaufrufen gilt das selbe:
EEPSetSignal(29,1) und EEPSetSignal(29, 1)
sind völlig gleichwertig. In der zweiten Variante habe ich hinter dem Komma ein Leerzeichen eingefügt, um die 1 etwas abzurücken. Meines Erachtens ist so deutlicher zu erkennen, dass es sich bei 29 und 1 um zwei separate Zahlen handelt.