Schöner programmieren Teil 2 – Sprachen und Variablen

Description

Schon beim Titel werden viele an Programmiersprachen denken, doch das ist im zweiten Teil nicht gemeint. Diese Tutorial-Serie ist sehr allgemein gehalten, womit es egal ist, ob in C, C++, GML, Basic oder VBA Programmiert wird.

Mit Sprache meine ich, in welcher Sprache man Variablen und Kommentare verfasst. Wie im ersten Teil bereits beschrieben, können die ersten Probleme auftauchen, wenn man seinen Code in ein Forum stellt. Natürlich kann nun jeder von sich behaupten, dass man ohnehin alles auf Deutsch macht und nur in Deutschen Foren unterwegs ist und auch so alles alleine macht, aber dann kommt dieser Tag X, wenn einem im deutschsprachigen Forum keiner helfen kann oder will. Oder es kommt Tag Y, an dem sich ein Russe meldet, der Interesse daran hat beim Projekt zu helfen, allerdings kann der nur Russisch und Englisch.

Wenn man der deutschen Sprache mächtig ist, hat man folgende Möglichkeiten: man kann alles in Deutsch schreiben und hoffen, dass weder Tag X noch Tag Y kommen. Allerdings kann auch Tag Z kommen, wenn man ein neues, größeres Projekt startet und man einige Bestandteile des kleineren Projekts mit einbauen will, etwa den Umgang mit INIs, Savegames und andere universelle Dinge. Und weil man das Projekt bereits mit dem Russen und dem Inder startet, kann man diese Passagen gleich überarbeiten.

Der zweite Weg ist ein Kompromiss, den viele machen. Code komplett in Englisch, Kommentare auf Deutsch. Das hat den Vorteil, dann an den Tagen X, Y und Z nur der Kommentar neu gemacht werden muss und somit die Funktionalität des Codes nicht beeinträchtigt wird.

Der dritte Weg ist, man macht alles auf Englisch. Das fällt vielen (auch mir) schwer, aber vor allem bei größeren Projekten ist das nützlich, erst recht wenn man mit mehrsprachiger Software arbeitet. Für Übersetzter sollte es eine Basis geben und das ist nun einmal die englische Sprache. Wenn Variablen und/oder Kommentare mal Deutsch und mal Englisch sind, gibt es ein mords Chaos.

Eine Ausnahme können Tutorials und Beispiele sein, die für ein spezielles Forum oder diese Seite verfasst werden. Da ist es sogar von Vorteil, wenn die jeweilige Landessprache verwendet wird.

Wichtig: Vor Projektbeginn sollte man sich festlegen und diesen Standard konsequent und diszipliniert durchziehen. Das fällt oft bei Variablen schwer, wenn einem das deutsche Wort eher einfällt als das englische, aber hier sollten Geschwindigkeit und Bequemlichkeit keinen Vorrang erhalten.

Noch kurz ein Beispiel für die Leute, die meinen, dass so ein paar Worte auf Deutsch doch kein Problem sein sollten. Angenommen, euer russische Teamkollege sendet euch einen älteren Code für das neue, gemeinsame Projekt. Der Code sieht dann so aus:

```
// ??? ?????????? ???????? ??????

if (kot3 > c?b?k?2)
{
        ???n = pbiq? * kot3 + c?b?k?2;
}else{
        ???n = kot3 - pbiq?;
}
```

Super formatiert, aber ansonsten: Bahnhof!

Variablen

Das Thema Variablen ist ein sehr großes Gebiet. Damit es nicht zu sehr ausartet, versuche ich, meine Kernaussagen etwas zu stutzen. Grundsätzlich kann ich jedem, der auch nur wenige Zeilen Code schreibt, dazu raten, sich über das Thema intensiver Gedanken zu machen.

Wie ich später noch erwähnen werde, ist GML eine relativ einfache Programmiersprache, die aber dadurch auch eine gewisse Schlamperei fördert. Dazu kommt noch der Umstand, dass GM einen nur in gravierenden Fällen vor Fehlern warnt, der Rest taucht erst beim spielen auf. Ein Klassiker sind Variablen, die man nicht deklariert, aber irgendwo im Spiel abfragt.

Die eben erwähnten sprachlichen Probleme schieben wir einfach mal beiseite und beschränken uns auf die übrigen Probleme mit Variablen.

Das erste Problem ist die Deklaration. Eigentlich ist es eine bequeme Sache, dass man nicht gezwungen ist, eine Variable erst einzurichten. Die Sache hat allerdings auch einen Hacken, über den ich immer mal wieder gestolpert bin: man verwendet Variablennamen mehrfach. Das Problem taucht bevorzugt dann auf, wenn man den Variablen keinen gescheiten Namen gibt (siehe weiter unten) und auch ganz gerne dann, wenn man nach einer bestimmten Zeit den Code erweitern will, aber nicht mehr den ganzen Code liest. Mehrfach ist mir das im GM beim Draw-Event passiert, wo man gerne, beispielsweise für die GUI, viele Dinge auf eine Seite programmiert (auch so eine Sache die man nicht machen sollte). Wenn man jede Variable artig oben oder beim Create deklariert, gibt es hier weniger Probleme. Bevor man den Code erweitert, schaut man erst, welche Variablennamen bereits existieren und denkt sich für die neuen Aufgaben auch neue Variablenamen aus.

Die zweite Sache ist etwas, worüber man streiten kann, ich will es aber dennoch als Anregung

erwähnen, auch wenn ich zugeben muss, dass ich das in GM auch nicht so mache. In GML haben Variablen keine Vorzeichen wie beispielsweise in PHP das $-Zeichen. Manchmal kann es etwas stören, wenn man nicht sofort sieht, wo die Variablen im Code sind. Manche machen folgendes: jede Variable bekommt einen bis drei Buchstaben vorangestellt, also Beispielsweise ein v oder ein var. Aus der Variable ObjectCounter wird also varObjectCounter. Das ist zwar etwas Zusatzarbeit, kann aber dennoch hilfreich sein, vor allem bei Diskussionen im Teamchat, damit man gleich weiß, dass man von einer Variable spricht.

Nun aber zum wichtigsten Thema, welches auch in Tutorials und Fachbüchern gerne vernachlässigt wird: Namen für Variablen. Einige Programmierer finden es besonders witzig und cool, wenn die Variablen kryptisch, sehr kurz oder lustig sind. Das mag für einen Gag in der Mittagspause gut sein, aber auf Dauer sind solche Variablen unnütz und ein blanker Horror. Dabei gilt es zwei Dinge zu beachten: eine Variable muss einen klaren, individuellen Namen haben und man muss den Namen auch lesen können. Hier eine kleine Liste mit schlechten Namen:

- zahl1
- flt
- zaehlerfuerallesundjenes
- isdoublescore
- mU_ha_ha
- essen
- meinevariable

Im Prinzip ist es sehr nützlich, wenn eine Variable aus mehreren Wörtern besteht, weil man damit besser beschreiben kann, wozu sie da ist, allerdings kommt es dann auf die Schreibweise an. Ich empfehle die Wörter mit Großbuchstaben voneinander zu trennen. Also isDoubleScore statt isdoublescore. Eine andere Möglichkeit ist die Verwendung von Unterstrichen, etwa bei einer Variable mit dem Namen lang_imp_sektions_start_date. Auch auf solche Standards sollte man sich vor Projektbeginn einigen. Wenn man sich nicht gleich einigen kann, sollte man schauen was in der jeweiligen Programmiersprache meistens verwendet wird.

Ressourcen trenne ich in GM immer mit einem Unterstrich und setzte einen entsprechenden Präfix davor, zum Beispiel obj_player. Auch wenn ich die Variable grundsätzlich nicht so nennen würde, hätte sie als Variable die Bezeichnung objPlayer. So kann man sofort sehen, was gemeint ist. Wer eine Variable für den Player braucht, sollte diese aber player nennen, ohne das obj_ davor.

Tipp: Man sollte nicht stur auf sein beliebtes System pochen, nur weil man zu faul ist sich anzupassen. Sofern es gut lesbar ist, sind die Standards egal, sie sollten nur nicht zu kompliziert sein.

Beispiel für übertriebenen Standard: var_is_Double_Score

Ich behaupte, dass sich Anfänger über nichts so wenig Gedanken machen wie über die Namen ihrer Variablen. Dabei sollte man sich dessen bewusst sein, dass es mühsam ist Code zu lesen, nicht nur fremden sondern auch eigenen. Selbst wenn man den Code gut dokumentiert und beschreibt, was die einzelnen Variablen bedeuten, kann es im Code dennoch sehr mühsam sein zu verstehen was passiert, wenn man als Variablenamen nur zahl1, zahl2, zahl3 etc. hat oder gar kryptische, unübliche Abkürzungen. Das Thema Abkürzungen wird im nächsten Teil noch eines sein, aber an dieser Stelle möchte ich darauf hinweisen, dass man möglichst auf Abkürzungen verzichten sollte und nur dann

welche nimmt, wenn diese üblich und für alle klar sind.

Bevor man also einfach darauf los programmiert, sollte man sich für einige Sekunden die Zeit nehmen und überlegen, welcher Name für die Variable am besten wäre. Wenn einem selbst nach einigen Minuten nichts einfällt, kann man auch mal ein Provisorium nehmen, sollte das aber im Code deutlich vermerken, damit man sich der Sache später annehmen kann.

Im Übrigen: Manchen ist vielleicht aufgefallen, dass es sich bei der Variable isDoubleScore möglicherweise um eine Boolesche Variable handelt, was man am is am Anfang des Namens erkennen kann. Schon mit solchen Kleinigkeiten kann man viele Missverständnisse im Code vermeiden.

Es mag Leute geben, die an manchen Stellen des Tutorials wild den Kopf schütteln. Schließlich zeigt einem der Editor i. d. R. an, was eine Variable ist und was nicht. Wozu es dann so überdeutlich benennen? Es mag sein, dass es in der einen oder anderen Entwicklungsumgebung nicht so wichtig ist, aber Umstände können sich ändern. Man steigt auf eine andere Version um und muss sich erst an die neuen Farben gewöhnen oder man verbreitet seinen Code per Mail, Foren oder Blog. Es ist nicht klar, wo die Zeilen einmal landen werden und ob am Ziel auch alles schön farbig ist. Die Arbeit der Farbgebung einer Umgebung zu überlassen, ist eine ziemlich laue Ausrede für undisziplinierte Arbeiten.

Im dritten Teil der Serie geht es um Ressourcenbezeichnungen und Codekommentare. Um die Einrückung geht es dann im vierten Teil.

Date Created
29. September 2016
Author
sven