Schöner programmieren Teil 1 – Disziplin

Description

Wer mit der Programmierung beginnt, egal ob Scriptsprache, Hochsprache oder einfache Programmiersprache, konzentriert sich erst einmal darauf, Sprache, Syntax, Befehle und Routinen zu verstehen. Man beginnt mit simplen Beispielen und stellt sich in der Regel immer größeren Problemen. Doch je mehr man programmiert, je mehr Projekte man angeht und je mehr Leute daran beteiligt sind stellt man fest, dass es damit noch lange nicht getan ist.

Mit ansteigender Komplexität und Personen wird es immer schwieriger, den eigenen Code zu warten und, wenn man ihn länger nicht gesehen hat, zu verstehen. In dieser Tutorial-Serie geht es darum, möglichst übersichtlichen Code zu produzieren, mit dem man auch noch Monate oder gar Jahre nach der Erstellung etwas anfangen kann.

Eines der wichtigsten Regeln bei der Programmierung ist folgende:

“Man arbeitet immer im Team, auch wenn man alleine arbeitet.”

Was nach einem Widerspruch klingt, hat tatsächlich einen tieferen Sinn. Selbst wenn man ganz alleine arbeitet, kann es immer mal vorkommen, dass man Hilfe braucht. In der Regel geht man in ein Forum, beschreibt kurz sein Problem und kopiert den Code hinein, womit meistens die Schwierigkeiten beginnen. Verstehen die Leute in einem englischen Forum den Code, wenn die Variablen auf Deutsch sind? Ist der Code gut kommentiert? Ist er so aufgebaut, dass man ihn gut lesen und verstehen kann? Je besser man gearbeitet hat, umso eher ist jemand bereit und auch in der Lage zu helfen.

Aber selbst wenn man alleine arbeitet und keine Hilfe braucht, kommt irgendwann der Tag, an dem man sich seinen eigenen, älteren Code anschaut. Spätestens dann versteht man, was mit Teamarbeit gemeint ist, obwohl man alleine arbeitet. Man muss mit seinem früheren Ich zusammen arbeiten und je nach Codequalität ist das nicht einfach, manchmal sogar fast unmöglich.

Ich bin zwar kein Programmier-Guru, aber ich habe viele Erfahrungen, vor allem Projekt-Erfahrungen und einiges davon würde ich hier gerne weiter geben. Es richtet sich vorwiegend an Anfänger, aber einige Dinge könnten auch für Fortgeschrittene interessant sein.

Neben vielen Tipps sind natürlich auch subjektive Meinungen dabei und natürlich würden mich eure Erfahrungen und Meinungen interessieren.

Programmierung ist vor allem logisches Denken, je nach Aufgabe auch Mathematik, aber sie sollte immer aus Disziplin bestehen. Jeder der mal schnell einige Zeilen Code geschrieben hat und sich diesen Code nach einigen Tage, Wochen oder Monaten ansieht, weiß worauf ich hinaus will. Ohne die nötige Disziplin versteht man irgendwann seinen eigenen Code nicht mehr und im schlimmsten Fall versucht man an einer Stelle einen Fehler zu beseitigen, die man nicht mehr nachvollziehen kann. Wenn alle Stricke reißen, programmiert man die Stelle neu und stellt am Ende fest, dass dieser Bereich überhaupt keinen Fehler hatte.

Wie solche Dinge passieren können? Es beginnt mit fragwürdigen Variablen, Hilfsvariablen, schlechten oder gar falschen Kommentaren, chaotische Einrückung und mündet dann in einer Sackgasse voll mit wirren Gedanken. Dabei ist es schon schwierig den eigenen Code zu verstehen, spätestens wenn zwei Programmierer oder mehrere zusammenarbeiten, hat Disziplin oberste Priorität. Die Grundregel lautet dabei: **Der Code muss nicht funktionieren, aber er muss lesbar sein!**

Wer nach dieser scheinbar einfachen Regel programmiert, erspart sich viel Ärger. Da es aber sehr schwierig ist, völlig diszipliniert zu programmieren, erfordert es Regeln und viel Übung. Selbst wenn ein Abschnitt gut aussieht und funktioniert, sollte man noch einmal die Disziplin aufbringen und schauen, was man verbessern kann.

Viele Programmierer leben nach dem Motto: Guten Code muss man nicht kommentieren! Ich habe da eine ganz andere Meinung und vertrete die Auffassung, dass jeder Code zumindest einen kleinen Kommentar benötigt der sagt, was der Programmierer damit beabsichtigt. Selbst wenn der Code wirklich toll ist, ist ein Kommentar manchmal kürzer und somit bei der Suche nach Fehlern sehr hilfreich wenn es darum geht, ob man an der richtigen Stelle sucht. Um dies zu realisieren und die nachfolgenden Ratschläge zu befolgen, braucht es sehr viel Disziplin.

Im übrigen bezieht sich das Kommentiergebot vor allem auf Abschnitte, also Funktionen, Methoden, Scripte. Im GameMaker hat man den Vorteil, dass die Events bereits viel erahnen lassen. Im **Draw-Event** muss man nicht kommentieren, dass hier gezeichnet wird. Wenn der Code aber eine Hand voll Zeilen überschreitet, sollte zumindest kurz beschrieben werden, was da alles dargestellt wird.

Wer nicht nur in GML programmiert sondern sich der großen Welt der Webprogrammierung öffnet, wird hier irgendwann auch darauf stoßen, dass man mit HTML, JavaScript und PHP drei Sprachen mischt. Vor allem bei Umgang mit HTML und PHP braucht es viel Disziplin, wenn es um die Trennung geht. Eine wichtige Grundregel ist, auf die Trennung von Ausgabecode und Seitenlogik zu achten. Im Code, der die eigentliche Seitenlogik darstellt, darf kein HTML auftauchen. Umgekehrt heißt das aber nicht, dass man im Ausgabecode kein PHP verwenden darf. Am Anfang klingt das unbequem und man drückt sich gerne davor. Spätestens, wenn ein Projekt eine gewisse Komplexität erreicht hat, wünscht man sich aber, man hätte diesen Rat befolgt und einige Zeit investiert, um eine elegantere, besser wartbare Version zu schreiben.

Wie so oft muss man auch die nötige Disziplin erlernen und trainieren. Deshalb ist es ratsam, erst viele kleine Projekte zu realisieren um sich Stück für Stück zu verbessern. Besonders wenn man Anfänger ist, sollte man nicht einfach drauf loslegen sondern sich vorher Gedanken machen, was man dabei

lernen will. Das ist ein Fehler, den ich selber zu oft gemacht habe. Ich habe immer gleich ein mehr oder minder großes Projekt begonnen und anhand dessen gelernt. Das Ergebnis war nicht nur ein Haufen Code, den ich selbst nicht mehr verstand, sondern das ich lediglich die Probleme lösen konnte, die sich im Projekt stellten. Wenn ich keine Arrays gebraucht habe, konnte ich das auch nicht.

Außerdem, wenn wir bereits bei Lernmethoden sind, sollte man besonders als Anfänger ein großes Projekt in kleinere Aufdröseln und lernen, diese zu meistern. Ein großes Spiel mit umfangreichen Effekten, Netzwerkcode, einer Online-Bestenliste und vieles mehr ist eine feine Sache, aber es stellen sich so viele Probleme, dass man sie als Anfänger unmöglich überblicken kann. Deswegen sollte man möglichst viele Komponente in einzelne, kleine Projekte auslagern, üben, lernen und am Ende überlegen, wie man das alles zu einem großen Projekt vereint bekommt. Während man immer weiter programmiert, verbessert man sich auch und man stellt fest, dass der Code vom letzten Projekt eine viel höhere Qualität hat, als vom ersten. Dies sollte dann beim großen Projekt immer berücksichtigt werden.

Im zweiten Teil gehe ich dann auf Programmiersprachen und Variablen ein.

Date Created
19. September 2016
Author
sven