

# D4 System

## Adventure Editor Manuel

---

### Vorwort:

Tja, normalerweise ist ein Vorwort ziemlich trocken geschrieben und meist gar nicht zum Lesen gedacht. Ich muss jedoch bei dieser vertrauten Regel eine Ausnahme machen und einige wichtige Dinge zu dem von dir erworbenen Programm erläutern. Nun, du bist mit dem Editor und dem Player in der Lage, ein Adventure zu erstellen, das mit jedem kommerziellen Programm mithalten kann. Du solltest jedoch diese Anleitung auf jeden Fall durchlesen, denn sonst garantiere ich dafür, das du mit deinem Adventure nicht zufrieden sein wirst. Ein Adventure ist mit viel Arbeit verbunden, und du verbringst dafür schlaflose Nächte, in denen du neue Gags und Ideen ausdenkst. Lies diese Anleitung bitte erst einmal durch, dann befolge sie Schritt für Schritt, um zu einem brauchbaren Adventure zu gelangen. Die Vorbereitungen sind am schwierigsten, jedoch kann man sie nicht weglassen. Hat man jedoch eine gewisse Vorarbeit geleistet, wird die Arbeit am Adventure immer einfacher - steigt sogar bis in den "Freude"-Bereich an. Hast du dein Adventure als "fertig" erklärt, bist du so motiviert, dass du gleich dein zweites Adventure beginnen willst. Wenn du schon mal ein eigenes Game gemacht hast, dann kennst du diese Motivationskurve (Sucht), andernfalls bitte ich dich, mir zu vertrauen, denn ich mache das schliesslich nicht zum ersten Mal - habe ich dir doch mit diesem Kit den Hauptteil der Arbeit schon abgenommen. Ich hoffe, dass die zweijährige Entwicklungszeit dieses Systems Bände spricht und über Makel wie den ersten Editor hinwegsehen lässt. Meine derzeitige Adresse findest du sicher irgendwo in diesem Heft, und ich freue mich immer über Kreationen, die mit meinen Systemen erstellt worden sind.

Dr. Zoom (Tobias Erbsland)

---

### Die Hardware und Software Voraussetzungen:

Theoretisch läuft das System mit nahezu allen 64'er Anlagen, jedoch gibt es einige nützliche Erweiterungen, die dann auch unterstützt werden. Klar, du besitzt sicher einen C64 oder einen C128, dazu mindestens ein Floppylaufwerk. Der Editor unterstützt bis zu vier Floppylaufwerke (auch eine Festplatte..), was bei grösseren Adventures ziemlich nützlich

sein kann. Die Files sind allsamt nicht allzu gross, jedoch ist ein Speedermodul von grossem Nutzen, weil man immer zwischen Spiel und Editor wechselt. Erkennt und unterstützt wird das Modul "Action Replay". Erkennt das Programm selbiges Modul, wird der Lader des Moduls automatisch eingeschaltet. Andere Speedermodule werden unter Umständen erkannt, aber ein Speeder muss schon selbst eingeschaltet werden. Grundsätzlich kann auch ein Softwarespeeder installiert werden, wobei ich bissher keinen lauffähigen (mit dem Editor) gefunden habe... Die erste bugfreie (fehlerfreie) Version des Players ist der "D4-Player V5.8xx"; hast du eine tiefere Version, dann hat dich jemand beschummelt. Grundsätzlich ist es egal, welchen Player und welchen Editor Du verwendest. Es kann höchstens passieren, dass der eine Editor Funktionen unterstützt, die der Player nicht ausführen kann oder umgekehrt. Solche Unstimmigkeiten wirst Du jedoch schon beim ersten Spielen feststellen und dann notfalls auf diese speziellen Funktionen verzichten müssen. Auch mit den Grundfunktionen läuft ein Adventure ordentlich (siehe Demoadventure).

Neben all den technischen Voraussetzungen braucht man sicher einmal einen guten Grafiker/gute Grafikerin, falls man die Bilder nicht selber malt. Dazu sollte man sich einen Testspieler suchen, der nicht gleich das Adventure weiterkopiert. Ein Testspieler/eine Testspielerin ist sehr wichtig, da dadurch meistens die Fehler gefunden werden, die man selbst nicht entdeckt hat. Ich habe die Erfahrung gemacht, dass die Zusammenarbeit mit einem Testspieler gegenpolaren Geschlechts (ist man Frau also Mann oder andersrum) sehr harmonisch verläuft. Auch in diesem Punkt solltest du mir vertrauen, denn dies ist keinen Schikane sondern basiert auf Erfahrung. Eine gute Kollegin/Kollege (Freund/Freundin) als Testspieler ist auch eine gute Idee. Die erste Entwicklungsphase schafft man zu zweit schneller als nur mit einem Hirn. Zum Adventure gehört weiter auch eine Musik ab \$1000. Am besten sucht man sich einen guten Composer, der die Musikstücke für das Adventure schreibt (komponiert). Findet man keinen Composer, der diese Arbeit übernimmt, so kann auch einmal ein Song aus dem Demoadventure verwendet werden. Soll das Ganze jedoch vermarktet werden, so darf man den Punkt Musik auf keinen Fall vergessen. Die Musik darf maximal den Bereich \$1000-\$2000 belegen. Wer dies nicht versteht oder meint, dass sei eine Frage des Dollars, der sollte diesen Satz und die weiteren Informationen abschreiben und dem entsprechenden Composer zukommen lassen, der die Musikstücke für das Spiel generiert. Auch das der Initvektor bei \$1000 und der Playaufruf bei \$1003 sollte man mit dem Composer besprechen. Ein bisschen sollte man das System schon kennen - es ist jedoch nicht Voraussetzung. Ich gehe einmal davon aus, dass du gerade mal Basic programmieren kannst und dich mit laden, speichern, löschen und kopieren von Files auskennst. Wie das funktioniert, steht aber auch im 64'er Handbuch (hoffe ich). Wer sich mit solchen Operationen nicht auskennt, sollte dies sofort nachholen und erst dann wieder weiterlesen. Ich hoffe aber, dass die Anleitung auch für Laien durchaus verständlich ist und keine allzugrossen Unklarheiten bleiben werden. Konstruktive Kritik an meine Adresse, pure Kritik solltest du jedoch in Anbetracht der zweijährigen Entwicklungszeit lieber lassen.

## Es geht los...

Grundsätzlich beginnt man jetzt erst ein kleines Adventure. Das erste Adventure, welches man erstellt, sollte auf keinen Fall mehr als 15 Räume beinhalten. Schon ein kleines Adventure kann ganz schön komplex sein. Ich habe aus diesem Grund auch erst das kleine Demoventure gemacht, das jedoch garantiert einige Stunden an den Bildschirm fesselt. Zudem macht ein kleines Adventure Lust auf mehr. Vielleicht gibt es sogar schon eine bessere Version des Editors und des Players zu kaufen nachdem du das kleine Adventure gemacht hast.

Mach einen Tisch frei, und besorge dir einen mittleren Stapel Papier - mit Vorteil kariertes. Dazu einige Stifte, Bleistifte, Radiergummi und auf jeden Fall einen feinen farbigen Stift (Rot, Blau, Grün). Ein Notizblock wäre auch ganz gut. Diesen kannst du auch unauffällig zur Arbeit mitnehmen und dort noch einige Notizen machen. Nun, wenn du alles bereitliegen hast, dann kann es weitergehen. Auf der unteren Illustration siehst du die linke obere Ecke des Raumbereichs. Dazu solltest du wissen, dass das ganze Adventure auf verschiedenen Räumen basiert, die in einer ganz bestimmten Matrix angeordnet sind. Die Maximalzahl der Räume ist 127 (ohne Introraum). Dies ist eine dreistellige Zahl, ich arbeite aber im ganzen System nur mit Hexadezimalen Zahlen. Das Hexadezimale Zahlensystem erweitert die normale Zahlenfolge 0-9 noch mit a-f. So entspricht der Hexadezimale Wert \$10 dem dezimalen Wert 16, der hexadezimale Wert \$7f dem dezimalen Wert 127. Hier erkennst du auch die Bedeutung des '\$' Zeichens. Ich setze es immer vor eine Hexadezimale Zahl, wenn diese nicht klar von einer dezimalen Zahl unterschieden werden kann (z.B. \$10). Nun, auf jeden Fall ist die Raummatrix 16 Räume breit und 8 Räume hoch. Nun verstehst du auch, warum der Anfang der Matrix immer "gerade" Zahlen enthält. Du erkennst so auch an der Zahl, wo sich der Raum befindet. Die "zehner"-Stelle gibt die Zeile an, die zweite Ziffer kennzeichnet die Spalte in der Matrix. Grundsätzlich ist es egal, welche Räume verwendet werden, wichtig ist die Verbindung zwischen den Räumen. Am besten beginnt man aber in die linken oberen Ecke.

Der Raum 00 bildet standardmässig das Introbild und den Introtex. Wie das genau funktioniert, erfährst Du später. Es bleiben für das Adventure aber alle restlichen Räume von \$01 - \$7f. Am besten du zeichnest diese Matrix einmal auf ein freies Blatt Papier, möglichst gross und vor allem richtig numeriert. Die Numerierung sollte in den Feldern nicht allzuviel Platz wegnehmen, weil hier Platz für Notizen und ähnliches frei sein muss. Dies ist vergleichbar mit einem Kalender.

00	01	02	03	04
10	11	12	13	
20	21	22	23	
30	31	32	33	
40				

Erstmals brauchst Du eine Story. Diese spielt in einer ganz bestimmten Gegend, die man dann in verschiedene Felder unterteilen kann. Im Demoventure ist dies Maulbronn, eine kleine Stadt in Deutschland. Auf der nächsten Abbildung erkennst du, wie ich jetzt

diese Stadt in einige wichtigen Schauplätze aufgeteilt habe und jedem dieser Schauplätze einen Namen zugeordnet habe. Dies ist der wichtigste Schritt, denn jede Szene/ jeder Raum ist eine kleine Story für sich, die mit den anderen Räumen zusammen einen Sinn bzw. ein Adventure ergibt; eine Art Puzzlespiel für Computerfreax. Die Räume an sich müssen interessant sein, denn ein langer Feldweg oder eine freie Grasfläche hat nur einen gewissen Reiz, wenn man da noch andere Dinge machen könnte, als in einen andere Richtung wieder davonzulaufen. Die wichtigsten Schauplätze im kleinen Demoadventure sind also hier z.B. die Kneipe auf Feld 21, der 'Tante Emma' Laden auf Feld 10, die Post auf Feld 01, das Kloster mit dem Weinkeller auf den Feldern 22 und 12. Alle diese Schauplätze werden vom Mittelfeld - dem Platz vor der Post - verbunden. Der Platz an sich ist nutzlos, hat aber in Verbindung mit den vier Hauptschauplätzen eine wichtige Funktion. Zerlege jetzt also deine Story in einige Hauptschauplätze, die man betreten kann. Dazu kommen dann später vielleicht noch einige Nebenräume, in die man nur unter bestimmten Voraussetzungen kommt. Lies am besten erst weiter, wenn du das gemacht hast.

Jetzt sollten alle wichtigen Schauplätze des Adventures sauber auf die Räume verteilt sein. Es ist egal, wenn sich dabei Lücken bilden - Hauptsache, die Räume schliessen alle aneinander an und können so auch begangen werden. Vergiss nicht, dass der Raum 00 tabu, bzw. später für das Intro verwendet wird. Diesen Raum kannst du also nicht mit einem Schauplatz belegen. Ich hoffe, dass alles auf deiner Karte (Map) stimmt und du jetzt noch einige Ergänzungen anbringen kannst. Wie du hoffentlich auf der letzten Abbildung erkennst, sind einige Linien dicker gezeichnet als andere. Das selbe solltest du auch auf deiner Karte machen, jedoch wenn möglich mit einer gut erkennbaren Farbe (z.B. Rot). Mit diesen Linien markierst du die "Wände" im Adventure. Über diese stark gezeichneten Linien kann man nicht gehen. Bei einem solchen Versuch erscheint "In diese Richtung kann ich nicht gehen!". Mit diesen Linie grenzt du die Schauplätze nach aussen ab, verhinderst das Betreten eines nicht existierenden Raumes. Diese Begrenzungen können, wie fast alle solchen "Standarts", mit einem Befehl umgangen werden. Diese Grenzen nehmen dir nur die grösste Arbeit ab, lassen dir mehr Zeit für das Wesentliche. Später kannst Du immer noch mit diesen Grenzen spielen, im kleinen Adventure solltest du aber die wichtigsten Dinge des Systems kennenlernen.

Jetzt wird es Zeit für die Bilder des Adventures zu sorgen. Die Bilder dürfen maximal 12 Charzeilen vom oberen Rand gross sein, das entspricht 96 Pixelzeilen, und sollten im Amica-Paint Format vorliegen. Notfalls konvertierst du erst die Bilder mit einem Konverter ins Amica-Paint Format. Die Bilder sind ziemlich wichtig, auch das System baut auf ihnen auf. Theoretisch kann man auch schon ohne Bilder mit der weiteren Arbeit beginnen, jedoch sind die Bil-

00 Intro	01 In Post	02 Wein- Keller	03	04
10 Laden	11 Vor Post	12 Vor Kloster	13	
20	21 Kneipe	22	23	
30	31	32	33	
40				

der eine gute optische Hilfe beim Testen und beim Ausarbeiten des Adventures. Nach Möglichkeit sollte es vermieden werden, Personen auf den Bildern zu zeichnen. Befindet sich keine Person auf dem Bild, kann man diese schnell einmal verschwinden lassen und dann wieder ins Bild setzen. Eine Person, die auf dem Bild gezeichnet ist, die muss logischerweise auch als Objekt vorhanden sein - in jedem Fall. Im Demo-Adventure sind fast alle Personen auf dem Bild eingezeichnet - Unflexibel. Auf keinen Fall sollte man jedoch Objekte im Bild einzeichnen, die man später verwenden kann und die dann logischerweise auch vom Bild verschwunden sein sollten. Dazu bieten sich dann 48 Sprites an, die man ins Bild setzen kann und die dann auch verschwinden, wenn man sie entfernt.

Lies jetzt am besten die Anleitung zum neuen D4-Editor durch, danach kannst du an dieser Stelle weiterlesen. Nach dem du jetzt die Pläne zu deinem Adventure gezeichnet und die Sache durchdacht hast ist es nun an der Zeit das Adventure aufzubauen. Dazu benötigst du eine leere Disk auf die du als erstes File den D4-Player kopierst. Jetzt suchst Du noch einen beliebigen Song, den du unter "M0" direkt nach dem D4-Player auf die Disk kopierst. Wenn das alles geschafft ist, lädst du den D4-Editor und startest diesen.

Gleich nach dem Start gehst du ins Diskmenü und speicherst "Alle Datenfiles" auf die Disk. Jetzt erzeugst Du nacheinander alle benötigten Räume mit der Funktion "Speichern unter..." auf der Disk. Du stellst einfach vor jedem Speichervorgang eine neue Raumnummer ein. Vergiss dabei nicht Raum 00 der als Intro dient. In meinem kleinen Beispiel speicherst du z.B. die Räume 00, 01, 02, 10, 11, 12 und 21. Auf der so erzeugten Disk sollten nun folgende Files zu finden sein:

- D4-Player
- Musikfile M0
- Gegenstandskarte NW
- Allgemeine Bedingungen BA
- Objektdatei1 O1
- Objektdatei2 O2
- Raumfiles 00 ... ??

Alles klar soweit? Jetzt kommen die Bilder ins Spiel. Wenn Du alle Bilder zu den verschiedenen Räumen gezeichnet hast, konvertierst du diese auf einer anderen Diskette ins "PIC"-Format. Klar benötigst du ein Intro-Bild das auch mitkonvertiert wird. Wenn alle Bilder konvertiert sind, steigst du wieder in den D4-Editor ein. Nach dem Start des Editors liest du im Diskmenü erst "Alle Datenfiles" ein, danach lädst du den ersten Raum. Lade jetzt in jeden Raum das richtige Bild ein und speichere den Raum mit "Speichern" wieder auf Disk.

Nachdem alle Räume mit Bildern versehen sind bearbeitest du die Bildtexte. Die Bildtexte editierst du in der Sparte "Resteditor" unter "Maintext editieren". Vermeide dabei in diesen

Texte Gegenstände oder Personen zu erwähnen die Flexibel eingebaut sind. Im Introraum ist der Bildtext überflüssig und kann weggelassen werden. Mit dem Bildtext zusammen kannst du auch gleich die Sperren in das Adventure einbauen um am Anfang keine Fehler zu verursachen.

Nachdem du diese ersten Arbeiten erledigt hast solltest du das Adventure gleich lauffähig machen. Gehe in den Introraum 00 und editiere die Raumbedingung 00. Die Bedingungen kannst du ignorieren, editiere gleich die Befehlsliste. An erste Stelle setzt du den Befehl "Sprung zu Raum" und nach dem Befehl gibst du den Startraum des Adventures an. In dieser ersten Befehlsliste kannst du auch beliebige Dinge dem Spieler direkt nach dem Start auf den "Weg" geben. Füge dazu vor den "Sprung zu Raum" verschiedene "Gegenstand in Besitz" ein.

Das Adventure ist nun voll lauffähig. Du kannst das Spiel schon jetzt mit dem Player starten und Spielen. Ich liste hier noch die restlichen Arbeiten an dem Adventure in der "richtigen" Reihenfolge auf:

1. Erstelle alle sichtbaren und unsichtbaren Objekte mit dem Objekteditor und einem Spriteeditor. Danach verteilst du die verschiedenen Objekte in die Räume und Bilder. Vergiss dabei nicht die Gegenstandskarte zu speichern.
2. Erstelle den grundlegenden *Plot* deines Adventures, dies ist der grobe Ablauf der Story. Erstelle dazu die notwendigen Bedingungen und teste das Adventure dann. Vergiss nicht eine Endsequenz (SQ) zu erzeugen, oder einen einfachen Textschluss einzubauen.
3. Verfeinere die Story mit Gag's, Geheimräumen, zweiten Wegen, Sackgassen, Fallen etc. Setzte jetzt die verschiedenen Musikstücke ein. Vergiss dabei nicht, dass die Musik bei einem Wechsel immer nachgeladen werden muss.
4. Mach die Endversion des Spiels und teste mit verschiedenen Leuten zusammen das Spiel bis es Perfekt ist. Benütze den Trackgeneraor um eine schnelle Version zu erzeugen. Style das Directory.
5. Das eigene Adventure ist nun fertig...

---

## Jetzt noch einen Stapel Theorie...

---

Ja, du ahnst es schon - diese Anleitung ist nicht ein Heft, das sich flüssig lesen lässt. Zuerst musst du dich mit dem System vertraut machen. Ich erläutere in den nächsten Zeilen das komplette System und dessen Aufbau. So genau, dass du sicher danach einen eigenen Editor programmieren könntest, sofern du eine Ass in der Kunst des Programmierens bist. Ich rate dir jedoch, falls du ein Hobbyprogrammierer bist, schon jetzt von dem Vorhaben ab, denn du würdest schon auf halber Strecke das Vorhaben wieder fallen lassen. Ich beginne hier mit den Grundlegenden Aufbau der Daten auf der Disk. Zudem hoffe ich, dass meine Erklärungen auch für ein Anwender mit weniger Erfahrung verständlich sind. Sind Stellen in dieser Anleitung, welche beim besten Willen unverständlich erklärt wurden, so bitte ich dich, mit deinem Händler in Kontakt zu treten. Dann wird die Sache an mich weitergeleitet und ich werde die Anleitung erneut überarbeiten.

Das ganze Adventure baut auf verschiedenen Räumen auf, die nach einem starren Prinzip angeordnet sind. Jeder Raum ist in sich ein eigenes kleines Adventure, welches zusammen mit den anderen Räumen ein grosses Adventure ergibt - ähnlich wie ein Puzzlespiel. Jeder Raum enthält ein Bild bis zu acht Texte und 32 Bedingungen, die frei Editiert werden können. In jedem Raum ist auch ein sogenannter Maintext (Grundtext) gespeichert, welcher beim Betreten des Raumes angezeigt wird. Zudem enthält jeder Raum die Koordinaten der bis zu acht Gegenstände, welche aber nicht in der Raumdatei gespeichert sind. Kodiert ist auch angegeben, welche Richtungen möglich und welche gesperrt sind. Eine Zahl gibt auch die Musiknummer im Raum an. Diese Daten befinden sich gepackt in einem Raumfile, das sich mit dem Namen \$00-\$7f auf der Disk befindet. Hier noch eine klare Auflistung, was sich im gepackten Raumfile befindet:

- 1 Multicolor 16 Farben Bild
- 8 Meldungen (je 6 Zeilen)
- 32 Bedingungen
- 1 Maintext (Bildtext/Grundtext)
- 8 Positionen von Gegenständen auf dem Bild.

Zusätzlich müssen sich verschiedene andere Files auf der Disk befinden. So enthält das File "OJ", das sich auf der Disk befindet, 56 verschiedene Sprites. Dazu auch die 56 Bezeichnungen der Sprites und die 56 Erkennungsbits. Das ganze ist wie folgt aufgebaut: Am Anfang des Files "OJ" befinden sich 56 Sprites, die je \$40 (Dez. 64) Bytes benötigen. Nach



den Spritedaten folgen die Namen der Objekte (jedes Sprite gehört zu einem 'Objekt'), die jeweils 16 Bytes benötigen. 15 Bytes werden für den Namen des Objekts verwendet. Dies entspricht 15 Zeichen für den Namen. Das 16. Byte/Zeichen wird als Steuerregister für das Objekt verwendet. Je nachdem welche Bits in diesem Byte gesetzt oder gelöscht sind, verhält sich das Objekt anders. Die Bits dieses Registers sind wie folgt belegt:

- |          |   |
|----------|---|
| Bit 0    | Ist dieses Bit gesetzt, so kann dieses Objekt nicht 'genommen' werden. Es erscheint die Meldung "Ich kann das nicht Nehmen". Dieses Bit ist eingebaut für Dinge wie Türen, Fenster oder ähnlich unverrückbare Dinge.  |
| Bit 1    | Wenn dieses Bit gesetzt ist, dann stellt dieses Objekt eine Person dar. Mit einer Person kann man Reden, man kann sie aber nicht 'einschalten' oder ähnlich sinnloses Zeug mit ihm machen. Der Parser fängt solche fehlerangaben mit entsprechenden Meldungen ab. |
| Bit 2    | Dieses Bit setzt das Sprite in den Multicolormodus. Ist das Bit gelöscht, so erscheint das Sprite Hires.  |
| Bit 3    | Dieses Bit ist für spätere Anwendungen und Erweiterungen reserviert.  |
| Bits 4-7 | Hier wird die individuelle Spritefarbe angegeben (0-15).  |

Vielleicht hast du schon gemerkt, dass hier bisher noch nicht erwähnt wurde, wo die beiden Farben: Multicolor1 und Multicolor2 gespeichert werden. Diese Farben werden auch in der Datei "OJ" gespeichert. Da nirgends sonst Platz dafür ist, befinden sich die Farben in obengenannter Reihenfolge in den Bytes \$3f und \$7f vom Anfang der Datei an gerechnet. Dies sind zwei letzte Bytes der ersten Sprites, das letzte Byte eines Sprite wird nicht dargestellt. Daher wird in zweien derer die Multicolorfarbe gespeichert.

Ich habe während der Arbeit mit dem System festgestellt, dass 56 Objekte unter Umständen nicht genug sind, daher habe ich noch das File "O2" eingeführt. Du findest in diesem File weitere 64 Objekte, zu denen leider keine Sprites angezeigt werden können. Diese Objekte werden nur als reine "Textobjekte" verwendet. Die reinen Textobjekte tragen die Objektnummern von \$40 bis \$7F. Du kannst sie also aufsammeln und wieder verlieren, ohne dass ein Sprite auf dem Bildschirm sichtbar oder unsichtbar wird. Aus diesem Grund solltest du die Objekte in zwei Bereiche aufteilen. Der erste Bereich sind Objekten gewidmet, welche am Anfang in den Bildern sichtbar sind, aber eingesammelt werden können. Der zweite Bereich gehört Objekten welche nie sichtbar sind, mit denen nur "gehandelt" werden. Das File "O2" besteht nur aus Bezeichnungen von 16 Zeichen, welche genau wie im File "OJ" aufgebaut sind.

Neben den Objekten und den Raumdateien bildet das File "NW" eine zentrale Funktion im Adventure. In diesem File findest du die **Verteilung** der Objekte im Adventure. Plazierst du im Raum 01 ein Objekt, so wird dies in diesem Bereich eingetragen. Wenn du also im Editor die Objekte neu platzierst und in Räume verteilst, danach aber vergisst, das File "NW" zu speichern, so sind alle neuen Objekte wieder verschwunden.



Grundsätzlich können in jedem Raum maximal 8 Objekte platziert werden. Der Fileaufbau ist daher ziemlich einfach gelöst: Die ersten 128 Bytes zeigen die Verteilung des ersten Objekts über die 128 Räume (\$00-\$7F). Dabei steht ein \$FF in einem dieser Bytes für kein Objekt. Diese Tabelle wiederholt sich noch sieben Mal, bis alle acht Objekte gespeichert sind.

Während des Spiels wird diese Tabelle oft umgestellt, wenn der Spieler z.B. ein Objekt nimmt, oder wieder verliert. Interessant dabei ist die Funktion des 8. Bits in diesen Tabellen. Ist das 8. Bit gesetzt, so erscheint das Objekt nicht als Sprite auf dem Bild, sondern nur in der Objektliste des Raums. Ein Sprite kann also nur vom Bild gelöscht werden, nicht aber wieder erscheinen. Das liegt vorallem daran, dass die Position der Sprites als Festwerte in den Raum-Files gespeichert sind. Verliert man so ein Objekt in einem anderen Raum, erscheint es unter Umständen an einer völlig unsinnigen Position im Raum.

Nicht minder wichtig ist das File "BA" auf der Disk. In diesem File sind alle allgemeinen Daten enthalten. Das sind zum einen 64 Bedingungen, die genau wie die Raumtexte behandelt werden, aber auch 16 Meldungstexte. Was aber sind genau die allgemeinen Daten? Grundsätzlich werden die "Allgemeinen-Daten" bei jedem Befehl des Spielers abgefragt - unabhängig von Raum und Situation. So ist es z.B. möglich, dass man an jeder beliebigen Stelle im Adventure eine Flasche Wein austrinken kann, danach nur noch eine leere Flasche in seinem Besitz hat. Mit diesen Bedingungen sollte man natürlich sehr vorsichtig umgehen, da sie ja für das ganze Adventure gelten. Grundsätzlich unterscheidet sich aber das Format nicht von dem der Raumbedingungen.

## Der Parser, dein Spielpartner

Der Player mit dem eingebauten Parser ist wohl der wichtigste Teil des Adventures. Nur wenn du ihn kennst, weißt du auch wie er sich verhält. Grundsätzlich vertraut der Parser immer auf den Spielersteller - also dich. Deine Kommandos haben immer Vorrang und umgehen dabei die Standartfunktionen des Parsers. Das ganze System ist in drei Stufen aufgeteilt:

- Stufe 1: Trifft Kommando auf eine Raumbedingung zu?
- Stufe 2: Trifft Kommando auf eine Allgemeinbedingung zu?
- Stufe 3: Auswertung auf mögliche Fehleingaben durch den Parser.

Da die Raumbedingungen und die Allgemeinbedingungen von dir erschaffen werden, gehe ich hier nur noch auf die Standartantworten des Parsers ein, welche bei Stufe 3 gemeldet werden. Diese Standartantworten können natürlich von den hier aufgeführten Beispielen abweichen, bleiben jedoch immer sinnmässig dieselben. In der nachfolgenden Tabelle findest Du am Anfang den Befehl, danach die interne Meldungsnummer bei einem normalen Objekt, bei einer Person und bei einem unverrückbaren Objekt:

Befehl	Objekt normal	Objekt Person	Objekt gesperrt
--------	---------------	---------------	-----------------

öffne	1	2	1
verschiebe	1	2	1
ein	1	2	1
untersuche	13	13	13
benutze	1	2	1
kombiniere	1	2	1
gebe	3 od. 10	2	3 od. 10
nimm	4, 9 od. 12	2 od. 12	5 od. 12
verliere	6, 10 od. 11	6, 10 od. 11	6, 10 od. 11
rede mit	7	8	7

- 1) "Das scheint nicht zu funktionieren!"
- 2) "Seit wann kann man den das?"
- 3) "Das will scheinbar niemand!"
- 4) "Ich hab's genommen!" (Danach befindet sich Objekt in Besitz)
- 5) "Das kann ich nicht nehmen!"
- 6) "Und weg isses!" (Danach befindet sich das Objekt wieder in Bild)
- 7) "Mach doch mal ne Pause..."
- 8) "Die Person gibt keine Antwort!"
- 9) "Ich kann nur 8 Dinge tragen!" (Wenn schon 8 Objekte im Besitz)
- 10) "Das besitze ich gar nicht!" (Wenn Objekt auf Bild angewählt)
- 11) "Hier ist kein Platz dafür!" (Wenn in Raum schon 8 Objekte sind)
- 12) "Das habe ich schon genommen!" (Wenn Objekt aus Besitz angewählt)
- 13) "Ich entdecke nichts Ungewöhnliches!"

Bei den vier Himmelsrichtungen hängt die Antwort von der Sperrung einzelner Richtungen ab. Dabei kann die Meldung: "Ich kann nicht in diese Richtung gehen!" erscheinen, falls diese Richtung gesperrt ist. Wie du vielleicht erkennst, funktioniert das System schon völlig unabhängig von deiner Story, die du schreibst. Deine Aufgabe ist es nur, durch gezieltes Sperren und Freigeben einzelner Optionen einen komplexen Spielablauf zu generieren, welcher einen Spieler an den Bildschirm fesselt.

## Der neue Dimension 4 Editor

Der neue Dimension 4 Editor lässt sich kinderleicht bedienen und erfordert nur wenig Kenntnisse über das System und dessen Datenformat. Trotzdem solltest du wissen, wie die Daten gespeichert werden und in welchem Format. Nach dem Starten des Editors erscheint eine Box mit Angaben zum Writerecht und der Versionsnummer an welcher man erkennt wie weit der Editor verbessert wurde. Die kleinste lauffähige Versionsnummer ist: m2151.59. Die Ziffern "21" stehen für den Entwicklungsstand des Windows-Systems. Entstehen also Änderungen am Windows-System, so erhöht sich diese Nummer um eine Stelle. Die Ziffern "51" symbolisieren das Grundsystem, welches die grundlegenden Abläufe

fe des Systems steuert. Die "59" zeigt den Standart des überwachungssystems, welches für die Uhr oder andere IRQ-gesteuerte Vorgänge verantwortlich ist.

Um mit dem D4-Editor arbeiten zu können, brauchst du einen Joystick oder eine Maus. Mit der Maus lässt sich das Programm wesentlich leichter bedienen - ein Joystick genügt aber völlig.

---

## Steuerung über Windows

---

Auf dem Bildschirm befindet sich der sogenannte Mauszeiger (☞). Man kann diesen Zeiger mit dem Joystick oder der Maus über den Bildschirm bewegen. Bewegt man den Mauszeiger über Textzeilen oder Kommandos, so leuchten diese auf. Drückt man den Feuerknopf (Button), so wird der Befehl ausgeführt, welcher hell angezeigt wurde.

Fast alle Funktionen können abgebrochen werden, indem man **neben** dem aktuellen Fenster den Button betätigt. Funktioniert dies nicht, so muss die Funktion über einen speziellen Punkt im Fenster abgebrochen werden. Wird ein Vorgang **abgebrochen**, so wird in den meisten Fällen auch die Eingabe **rückgängig** gemacht.

Erscheint statt dem Mauszeiger (☞) die Uhr (⌚), so ist der Computer beschäftigt und man kann keine Eingaben über den Mauszeiger machen. Die Uhr erscheint auch, wenn man Eingaben über die Tastatur machen muss. Diese Tastatureingaben kann man nur mit einer **Leereingabe** abbrechen.

Am oberen Rand des Bildschirms befindet sich eine Linie, auf der alle verschiedenen Editorsparten aufgeführt sind. Mit dieser Linie schaltet man zwischen den verschiedenen Editoren und Menüs um. Dieses Umschalten funktioniert aber nur, wenn man sich in der untersten Menüebene befindet. Man stellt sich das so vor: Jedes Fenster, welches neu geöffnet wird, wird **über** das letzte gelegt. Erst wenn keine Fenster mehr über dem untersten Fenster liegen, kann man in eine neue Sparte wechseln.

Einfacher ist es jedoch wenn man die Funktionen ausprobiert und so erkennt, was sie bewirken. Obwohl in keinem Fall etwas zerstört werden kann, sollte bei Experimenten im Disk-Menü keine Editordisk im Laufwerk sein. Im Editor sind so viele Funktionen eingebaut, dass ich nicht jede genau beschreiben kann. Hier auf jeden Fall die Beschreibungen zu den verschiedenen Sparten: Die allgemeinen Bedingungen unterscheiden sich nicht von den Raumbedingungen (natürlich nur optisch), so dass ich mich nur mit letzteren befasse.

## Das Diskmenü

Das Diskmenü ist die Ein/Ausgabe Schnittstelle des Editors. Hier kann man jede nur erdenkliche Art von Daten laden, speichern und löschen. Im Grundmenü befinden sich folgende einfache Funktionen:

### Raum speichern:

Der aktuelle Raum wird gepackt und gespeichert. Nach dem Start des Editors ist der aktuelle Raum auf Nummer "00" eingestellt. Wird ein anderer Raum von Disk gelesen, so wird die Raumnummer automatisch angepasst. Befindet sich der aktuelle Raum schon auf der eingelegten Diskette, so erscheint die Meldung "File Exists". Nach einem Druck auf den Button wird der Raum auf Disk mit dem neuen Raum überschrieben.

-+	Disk Menü	+-
-----		
File	Laden	
File	Speichern	
Raum	Speichern	
Raum	Speichern unter	
File	Löschen	
Disk	befehl Senden	
Laufwerk	Wechseln	

### Raum speichern unter:

Nach dem Anwählen dieser Funktion öffnet sich eine weitere Box, in der die Raumnummer eingestellt werden kann. Dies wird mit den horizontalen und vertikalen Zahlenreihen gemacht. Wählt man die Ziffer in der linken oberen Ecke, so wird der aktuelle Raum unter der eingestellten und angewählten Nummer gespeichert. Der aktuelle Raum wird auf die angewählte Nummer gesetzt.

### Diskbefehl senden:

Hier kann ein Befehl an die Floppy gesendet werden. Abgebrochen wird die Funktion, wenn man keine Eingabe macht, bzw. im leeren Feld "Return" drückt. Um Räume zu löschen verwendet man z.B. den Befehl "S:12", dabei gibt 12 die Raumnummer an.

### Laufwerk wechseln:

Es erscheint eine weitere Box, in der eines der aktiven Laufwerke angewählt werden kann. Die als -Aus- gekennzeichneten Laufwerke können nicht angewählt werden. Tritt die Situation auf, dass keines der vier Laufwerke eingeschaltet ist, so bricht man die Funktion einfach ab.

•	Laufwerk 8	: Aktiv
	Laufwerk 9	: -Aus-
	Laufwerk 10:	-Aus-
	Laufwerk 11:	-Aus-

### File laden, speichern, löschen:

Nachdem einer dieser drei Punkte angewählt wurde, erscheint eine weitere Box mit jeweils den selben Funktionen. Ich fasse daher diese Funktionen in einem Textabschnitt zusammen. Alle Funktionen beziehen sich auf Laden, Speichern und Löschen.

- Alle Datenfiles

Alle Datenfiles, die unter nachfolgenden Punkten (Gegenstandskarte, allgemeine Daten, Objekte) aufgeführt sind, werden nacheinander auf Disk gespeichert. Treten Fehler auf,

wird der Vorgang nach einem Druck auf den Button fortgesetzt. Vor dem Speichern der Datenfiles sollten diese unter löschen gelöscht werden. Bei jedem Löschvorgang erscheint noch eine Sicherheitsmeldung, die mit "Yes" oder "No" beantwortet werden muss.

#### - Gegenstands Karte

Mit diesem Punkt wird die Gegenstands-Karte (NW) auf die Disk gespeichert.

#### - Allgemeine Daten

Die Datei (BA) wird auf der Disk erzeugt.

#### - Objekte

Bei diesem Punkt werden die beiden Objekt-Files (OJ,O2) auf der Disk erzeugt. Tritt beim ersten File ein Fehler auf, so wird der Vorgang nach einem Druck auf den Button mit dem nächsten File fortgesetzt.

#### - PIC-Bilddatei

Hier kann ein Bildfile in einen Raum eingeladen werden. Das Bildformat entsteht nach dem Konvertieren mit dem Bildkonverter. Ein Bildfile kann auch wieder auf Disk gespeichert werden.

#### - Sprite/Sprites

Wählt man diesen Punkt, erscheint beim Laden oder Speichern die Objektauswahl-Box, in der das zu speichernde Sprite oder das Startsprite zum Laden angegeben werden muss. Zugelassen sind natürlich nur die sogenannten "Sprite-Objekte" mit Nummern kleiner als 540. Beim Speichern wird immer nur ein Sprite gespeichert, laden kann man dagegen ganze Spritesätze.

```

-   FILE LADEN   -
-----
Alle Datenfiles
Gegenstands Karte
Allgemeine Daten
Objekte
PIC-Bilddatei
Sprite/Sprites
-----
Directory zeigen !

```

## Die Raumbedingungen

Wählt man die Sparte Raumbedingungen (oder allgemeine-Bedingungen), so erscheint eine grössere Box, in der verschiedenste Parameter verändert werden können. Sämtliche Hexadezimal-Zahlen können angewählt, "manuel" verstellt und auf vielleicht unmögliche Werte eingestellt werden. Diese Methode hat jedoch nur Sinn, wenn man mit einem speziell modifizierten Player arbeitet. Die manuelle Einstellung erlaubt es aber auch, die spezielle "flag" Abfrage in die Bedingung einzubauen. Dazu später mehr.

Am rechten Rand erkennt man Pfeile nach oben und nach unten. Mit diesen bewegt man sich in der Liste nach oben und unten, wobei immer nur die mittlere Bedingung editiert werden kann.

Wählt man den Befehl an oder die am Anfang eingestellte Linie, so erscheint eine Liste mit allen Befehlen des Players. Hier kann einer der Befehle angewählt werden, um die Bedingung zu verändern. Bricht man die Funktion ab, so wird wieder der alte Zustand hergestellt.

03 nimm	E:Hammer	01
/\	B:#Keine Abfrage#	FF
	R:#Keine Abfrage#	FF
-----+		
04 öffne	E:Tür	41
\	B:#Keine Abfrage#	FF
/\	R:?Flag 0 gesetzt?	90
-----+		
05 -----	E:#Keine Abfrage#	FF
	B:#Keine Abfrage#	FF
\	R:#Keine Abfrage#	FF

Dasselbe funktioniert mit den Werten nach dem Befehl mit den Gegenständen auf der rechten Seite. Wählt man einen Gegenstand an, so erscheint die Gegenstandsliste, in der man einen Gegenstand anwählen kann oder auch "Keine Abfrage". Diese Funktion erscheint, wenn man in der Gegenstandsliste ganz nach oben scrollt mit den Pfeilen.

Im mittleren Abschnitt lässt sich auch die **Nummer** der Bedingung anwählen. Es erscheint dann eine weitere kleine Box in der man verschiedene Punkte anwählen kann. Mit *Insert One* wird an dieser Stelle eine Bedingung eingefügt. Alle weiteren Bedingungen verschieben sich damit eine Position gegen das Ende der Liste. Oft ist die Reihenfolge der Bedingungen wichtig, da diese ja sequentiell abgefragt werden. Die letzte Bedingung der Liste geht dabei verloren, dass ist bei den Raumbedingungen die Nummer \$1F und bei den allgemeinen Bedingungen die Nummer \$3F. Ein weiterer Punkt ist *Delete One*. Wählt man diesen an, so wird die aktuelle Bedingung gelöscht und alle weiteren um eine Position nachgerückt. Am Ende der Liste wird eine leere Bedingung eingefügt. Der dritte Punkt ist *Copy to ??*. Wählt man diesen Punkt, so erscheint eine weitere Box in der man die Zielnummer einstellen kann. Danach wird die aktuelle Bedingung kopiert, dabei wird eine evt. an dieser Stelle vorhandene Bedingung überschrieben.

Diese erste Aufstellung zeigt die **Bedingung** an welche erst erfüllt werden muss um gewisse Ereignisse auszulösen. Basic-Programmierer können diese Abfrage mit einem If, Then Befehl vergleichen - wobei dieser erste Teil den Parameter zwischen dem If und dem Then darstellt. Eine solche Abfrage besteht in jedem Fall aus einem Befehl, welcher aus diesem Grund auch den Gegenständen vorangestellt ist. Steht an der Stelle des Befehls eine Linie aus Minuszeichen, so wird diese Bedingung **nicht** abgefragt. Dabei spielen die Parameter dahinter keine Rolle. Um diese Zusammenhänge etwas verständlicher zu gestalten, betrachtet man das Ganze besser vom Spieler aus, welcher in dem Adventure Befehle gibt:

Der Spieler wählt erst den Befehl "öffne" an. Danach erscheinen die beiden Listen, die eine mit dem Besitz, die andere mit den Raumgegenständen. Er wählt nun aus der Besitzliste den Gegenstand "Flasche". Jetzt wird der Parser aktiv und sucht die Raumbedingungen nach dem Befehl "öffne" ab. Bei Bedingung vier findet der Parser den Befehl

"öffne" und untersucht nun die Parameter der vierten Bedingung. Der Vergleich ist negativ, in der vierten Bedingung wird eine Tür abgefragt, keine Flasche. Die Suche geht weiter und am Ende der Raumbedingungen wechselt der Parser zu dem allgemeinen Bedingungen. Auch bei den allgemeinen Bedingungen hat der Parser kein Erfolg. So untersucht er die Eingabe auf Fehler, z.B. ob "Flasche" eine Person ist. Nach dieser riesigen Aktion erscheint auf dem Textfeld "Das scheint nicht zu funktionieren!". Von dieser Aktion merkt der Spieler natürlich nichts.

Trifft der Befehl also zu, nach welchen Eigenschaften werden die Gegenstände verglichen? Das ist nicht einmal so einfach wie es scheint, trotzdem kein Problem. Vor den drei Gegenständen stehen die drei Buchstaben "E", "B" und "R". Sie bedeuten soviel wie "Egal wo", "Besitz" und "Raum". Abhängig davon, hinter welches Zeichen der Gegenstand gesetzt wird, wird die Bedingung anders überprüft. Steht z.B. Hammer hinter dem "E", so darf dieser in der Besitzliste wie auch bei den Raumgegenständen angewählt werden. In jedem Fall trifft die Bedingung zu. Ein Gegenstand hinter dem Buchstaben "B", darf nur aus der Besitzliste gewählt werden. Das Selbe natürlich mit dem Buchstaben "R", der Gegenstand muss dabei im Raum zu finden sein.

Hinter jedem Befehl darf nur immer ein Gegenstand stehen, mit Ausnahme des Befehls "kombiniere", bei dem zwei Gegenstände angegeben werden müssen. Zudem kann eines der 16 Flag's, welche während des Adventures verändert werden können, auf einen bestimmten Zustand abgefragt werden. Diese Bedingung lässt sich leider nicht über ein fertiges Untermenü einstellen. Diese Abfrage der Flag's kann bei dem letzten Gegenstand, hinter dem Zeichen "R" eingestellt werden. Dazu verändert man die Zahl hinter dem Zeichen nach folgendem Schema:

\$80-\$8F = Gesetztes Flag 0-F  
 \$90-\$9F = Gelöschtes Flag 0-F

Ist eine dieser Zahlen eingestellt, so wird diese Abfrage entschlüsselt dargestellt. Statt einem Gegenstand erscheint: "Flag x gesetzt ?" oder "Flag x gelöscht ?". Ist dieser Code eingestellt, so bedeutet das eine zusätzliche Abfrage. Nur wenn diese zusätzliche Abfrage zutrifft, also das entsprechende Flag gelöscht oder gesetzt ist, so gilt die Bedingung als erfüllt und die Liste der Befehle wird ausgeführt.

## Die Befehlsliste

Trifft eine Bedingung zu, so muss im Adventure irgendwie darauf reagiert werden. Wählt man am linken Rand der Bedingungsliste das Plusymbol (+) so erscheint die Befehlsliste. Die Liste ist unterteilt in zwei Spalten: Die Befehlsspalte auf der linken Seite und die Parameterspalte rechts. Wie schon bei den Bedingungen kann jeder Wert manuel verändert werden. Eine durchgehende Linie mit dem Minuszeichen bedeutet soviel wie kein Befehl. Die Befehle werden von oben nach unten ausgeführt, dabei werden leere



Linien einfach übersprungen. Wählt man eine leere Linie oder ein Befehl an so erscheint eine Liste mit allen möglichen Befehlen. Je nachdem, welchen Befehl ausgewählt wurde

Löschen in Bes.	05	Hammer	05
Löschen in Bes.	05	Nagel	03
Gegenstand in B.	02	Bilderrahmen	09
Meldung anzeigen	01	Text \$01	01
-----	00		00
-----	00		00

verändert sich auch der Parameterbereich in der zweiten Spalte hinter dem Befehl. Ich gehe nur auf jeden Befehl im einzelnen ein und auf dessen Parameter.

## Die Befehle des D4-Systems

### Gegenstand in Besitz

Der Parameter dieses Befehls ist ein beliebiger Gegenstand. Trifft der Parser auf diesen Befehl, wird der nachfolgend aufgeführte Gegenstand in die Inventarliste des Spielers übernommen. Ist die Inventarliste des Spielers voll, so wird dieser Befehl ignoriert. Beim Ersetzen von Gegenständen muss daher immer erst der alte Gegenstand gelöscht, erst dann der neue in die Liste übernommen werden.

### Löschen in Besitz

Als Parameter benötigt dieser Befehl auch wieder einen beliebigen Gegenstand. Wird dieser Befehl ausgeführt sucht der Parser in der Inventarliste nach dem als Parameter aufgeführten Gegenstand. Existiert ein solcher Gegenstand in der Inventarliste, löscht der Parser diesen. Findet der Parser keinen solchen Gegenstand wird dieser Befehl ignoriert. Beim Ersetzen von Gegenständen muss immer erst der alte Gegenstand gelöscht, erst dann der neue in die Liste übernommen werden.

### Löschen in Raum

Der Parameter dieses Befehls ist ein beliebiger Gegenstand. Wie auch bei *Löschen in Besitz* sucht der Parser in dem aktuellen Raum nach dem angegebenen Gegenstand. Wird dieser im aktuellen Raum gefunden so löscht der Parser den Gegenstand. Dabei wird auch ein evt. Sprite vom Bild entfernt. Beim Ersetzen von Gegenständen muss immer erst der alte Gegenstand gelöscht, erst dann der neue in die Liste übernommen werden.

### Flag setzen

Als Parameter benötigt dieser Befehl eine Flagnummer zwischen '0' und 'F'. Das so angegebene Flag wird gesetzt. Grundsätzlich ist dieser Befehl auf \$7F Flag's ausgelegt. In einer späteren erweiterten Sprache können diese zusätzlichen Flag's auch abgefragt werden.

### Flag löschen

Wie auch bei *Flag setzen* benötigt dieser Befehl als Parameter eine Flagnummer zwischen '0' und 'F'. Das angegebene Flag wird gelöscht. Grundsätzlich ist dieser Befehl auf \$7F Flag's ausgelegt. In einer späteren erweiterten Sprache können diese zusätzlichen Flag's auch abgefragt werden.

## Meldung anzeigen

Hinter diesem Befehl wird als Parameter eine Textnummer zwischen '0' und '7' angegeben. Trifft der Parser auf diesen Befehl wird die angegebene Raummeldung im Textdisplay angezeigt. Hat man diesen Befehl angewählt, erscheint in der Parameterspalte die Meldung "Text \$00". Wählt man diese Meldung an, so erscheint das Textmenü in dem die verschiedenen Raummeldungen editiert werden können. Die Texteingabe im Textmenü wird mit der Taste 'F1' beendet. Die Meldung wird bis zur letzten beschriebenen Zeile angezeigt, leere Linien am Schluss werden ignoriert. Im *Raum Bedingungen* Editor sollte immer der Befehl 'Meldung anzeigen' verwendet werden, bei den *allgemeinen Bedingungen* immer der Befehl 'A-Meldung anzeigen'. Die beiden Befehle unterscheiden sich nur dadurch, dass die Daten an verschiedenen Orten im Speicher abgelegt werden. Befindet sich unmittelbar nach diesem Befehl einer folgender Befehle: *Meldung anzeigen*, *A-Meldung anzeigen*, *Sprung zu Raum*, *Endsequenz laden* erscheint neben dem angezeigten Text die Meldung 'Press Button'. Dabei wird die Textausgabe solange angehalten, bis der Spieler der Feuerknopf oder den Button der Maus betätigt.

## Allgemein Meldung anzeigen

Hinter diesem Befehl wird als Parameter eine Textnummer zwischen '0' und 'F' angegeben. Sonst ist der Befehl, bis auf oben genannte Punkte, identisch mit *Meldung anzeigen*.

## Sprung zu Raum

Der Parameter dieses Befehls stellt eine Raumnummer zwischen '01' und '7F' dar. Trifft der Parser auf diesen Befehl, springt er zu dem angegebenen Raum. Dabei werden alle Sperren umgangen. Der angegebene Raum muss in jedem Fall existieren, andernfalls bricht der Parser mit einem Ladefehler das Spiel ab. Die Befehle nach *Sprung zu Raum* werden ignoriert.

## Game Over

Grundsätzlich benötigt dieser Befehl keinen Parameter. Stellt man jedoch manuell einen Wert zwischen '0' und '3' ein, wählt man zwischen vier verschiedenen Game Over Meldungen. Trifft der Parser auf diesen Befehl wird das laufende Spiel abgebrochen, eine 'Game Over' Meldung ausgegeben und die Intro-Sequenz nachgeladen. Die Befehle nach *Game Over* werden ignoriert.

## Endsequenz laden

Dieser Befehl benötigt keine weiteren Parameter. Trifft der Parser auf diesen Befehl, wird die Meldung 'Press Button' ausgegeben. Drückt der Spieler auf den Fireknopf, oder betätigt den Button der Maus, erscheint in der ersten Zeile des Bildschirms die Meldung 'Einen Moment Geduld', währenddessen wird die Endsequenz 'SQ' nachgeladen. Nach dem Ladevorgang wird die Musik ausgeblendet und die Sequenz gestartet. Eine Rückkehr nach diesem Befehl ist nur mit einem Speziellen Programm möglich, grundsätzlich bildet jedoch die Endsequenz das Ende des Adventures.

## Warten auf Taste

Trifft der Parser auf diesen Befehl wird die Meldung 'Press Button' angezeigt und das Spiel solange unterbrochen, bis der Spieler den Button betätigt. Dieser Befehl wird in den meisten Fällen jedoch ausreichend simuliert.

## Gegenstand in Raum

Als Parameter benötigt dieser Befehl einen Beliebigen Gegenstand. Trifft der Parser auf diesen Befehl wird der angegebene Gegenstand in den aktuellen Raum gesetzt. Dabei wird ein Sprite-Gegenstand **nicht** angezeigt. Enthält der aktuelle Raum schon acht Gegenstände wird dieser Befehl ignoriert. Beim Ersetzen von Gegenständen muss immer erst der alte Gegenstand gelöscht, erst dann der neue in den Raum gesetzt werden.

## Der Objekt Editor

Eine weitere wichtige Spalte stellt der Objekt Editor dar. In diesem werden die verschiedenen Objekte mit Namen versehen, alle Farben eingestellt und Objekte auf die Räume verteilt sowie dessen Sprites in den Bildern platziert. Wählt man diese Spalte aus, erscheint in der Mitte des Bildschirms das Mainfenster, welches die wichtigsten Editierfunktionen enthält.

In der Spalte ganz links kann man ein Rechteck erkennen. In diesem erscheint das Sprite zu dem mit einem (■) markierten Objekt. Direkt unter dem Sprite stehen folgende Punkte mit denen die verschiedenen Farben eingestellt werden können:

		↑
Bild...	00 ■ Hammer 01 Einsensäge 02 Tante Emma 03 Roter Schlüssel	↓
Backg.	Unbewegliches Obj.	■ ■
Multi1	Person	
Multi2	Multicolor	■ ■
Objcol	Positionierung !	

- Backg. Ändert die Hintergrundfarbe, also das grosse Rechteck hinter dem Sprite. Teilweise ist dies sinnvoll, auch wenn die Hintergrundfarbe keinen Einfluss auf das spätere Spiel nimmt.
- Multi1 Setzt die erste Multicolorfarbe für das komplette Spiel. Diese Farbe bleibt bei allen Sprites gleich. Die Farbe wird in der Objektdatei geändert. Bei einer Farbänderung muss auch die Objektdatei auf der Disk verändert werden.
- Multi2 Wie bei Multi1, doch wird hier die zweite Multicolorfarbe für das komplette Spiel eingestellt.
- Objcol Stellt die Farbe für das aktuelle Objekt/Sprite ein. Jedes Sprite kann eine eigene Objektfarbe haben.

Rechts neben der Farbeinstellung findet man vier weitere Punkte die allsamt angewählt werden können. Direkt rechts von den Punkten, in der Spalte ganz rechts, wird der Zustand der Option angezeigt. Ist der Punkt aktiv, so erscheinen zwei (■ ■) in dieser

Spalte. Die Bedeutung von "Unbewegliches Objekt" und "Person" wurden schon in einem anderen Kapitel behandelt, ich gehe daher nicht weiter darauf ein. Der dritte Punkt "Multicolor" zeigt an, ob das Sprite in Hires oder in Multicolor angezeigt wird. Der letzte Punkt "Positionierung" lässt ein Untermenü erscheinen, in dem Objekte auf das Bild gesetzt werden können. Dazu später mehr.

Mit den beiden Pfeilen am rechten Rand kann die Objektanzeige gescrollt werden. Dabei beziehen sich alle Operationen auf das Objekt, welches mit dem (■) markiert ist. Will man den Namen des Objekts editieren, so kann dieses Objekt angewählt werden. Es erscheint ein Eingabefenster in dem die Objektbezeichnung editiert werden kann. Abgebrochen wird diese Funktion durch eine Leereingabe, dabei bleibt die alte Bezeichnung bestehen. Will man jedoch eine Bezeichnung löschen, so macht man dies indem man nach der Eingabeaufforderung ein Space-Zeichen eingibt, danach die Eingabe mit Return abschliesst.

Wählt man den Punkt Positionierung, so erscheint ein weiteres Fenster. In diesem sind alle Objekte aufgeführt die sich im aktuellen Raum befinden. Der Wert \$FF ist ein Platzfüller und stellt kein Objekt dar. Wählt man eine dieser acht Zeilen an, so erscheint ein Auswahlfenster in dem das aktuelle Objekt verändert werden kann. Eine solche Änderung des Objekts wird in der **Gegenstandskarte** gemacht, danach muss diese auf der Disk aktualisiert werden. Die hier angegebene Liste von Objekten stellt die Startverteilung aller Objekte auf verschiedene Räume dar. Diese Liste ist abhängig von dem aktuellen Raum der bearbeitet wird.

pos	00	Hammer
pos	ff	-keine Abfrage-
pos	ff	-keine Abfrage-
pos	ff	-keine Abfrage-
pos	ff	-keine Abfrage-
pos	ff	-keine Abfrage-
pos	ff	-keine Abfrage-
pos	ff	-keine Abfrage-

Um ein Spriteobjekt auf dem Bild zu platzieren wählt man neben dem Spriteobjekt den Punkt "pos". Es erscheint daraufhin das Raumbild und der Mauszeiger verwandelt sich in das betreffende Objekt. Dieses kann nun an einer beliebigen Stelle im Bild platziert werden. Setzt man das Sprite unterhalb des Bildes in den schwarzen Bereich wird das Sprite unsichtbar platziert. Es erscheint also nicht auf dem Bild. Die Positionierung der Spriteobjekte wird in den Raumdaten gespeichert, eine Veränderung muss also mit "**Raum Speichern**" auf der Disk aktualisiert werden.

## Der Resteditor

Der Resteditor stellt die letzte Sparte des Editors dar. In dieser Sparte werden die letzten, aber auch grundlegendsten Einstellungen gemacht. Wählt man diese Sparte, so erscheint das Grundmenü mit sieben Punkten die ich hier erklären möchte:

### Edit Maintext

Wählt man diesen Punkt, so erscheint ein Fenster in dem der Grundtext editiert werden kann. Dieser Grundtext erscheint am Anfang jedes Raums den man betritt. Es sollte vermieden werden Gegenstände oder Personen im Raum zu erwähnen, welche verändert oder verschoben werden können. Abgeschlossen wird die Texteingabe mit der Taste "F1".

### Change Soundnr.

Hinter diesem Menüpunkt erscheint immer die aktuelle Einstellung der Soundnummer. Grundsätzlich können im gesamten Adventure bis zu 16 verschiedene Musikstücke gespielt werden. Diese Option legt die Soundnummer für den aktuellen Raum fest.

### Norden, Süden, Westen und Osten

Nach den vier Himmelsrichtungen erkennt man die Anzeigen "Frei" oder "Blockiert". Sie können durch anwählen der vier Himmelsrichtungen verändert werden. Die Bedeutung dieser Sperren wurde schon in einem anderen Kapitel beschrieben.

### Bild betrachten

Wählt man diesen Punkt erscheint das aktuelle Raumbild. Mit einem weiteren Druck auf den Button wechselt man wieder zur Menüebene.

## Hinweise zum Programm

---

Folgende wichtige Dinge sind mir beim Testen des Editors aufgefallen. Dieses kleine Kapitel enthält auch einige Tips und Vorschläge:

**Die erste Allgemeinbedingung 00** sollte übersprungen werden. Leider fällt diese Bedingung auf die Adresse \$A000 welche aus unfindbaren Gründen öfters verändert wird. Die Bedingung sollte also auf unmögliche Werte eingestellt werden. Dazu setzt man einfach drei verschiedene Objekte hinter dem Befehl ein.

Es existieren verschiedene **billig Versionen der Floppylaufwerken 1541, 1570 und 1571**. Der Hardwareaufbau scheint bei diesen Floppytypen anders zu sein. Der Irq-Loader funktioniert aus diesem Grund nicht mit diesen Laufwerken zusammen.

Die Zusatzkarte **Flash 8 funktioniert noch nicht** mit dem aktuellen Player zusammen. Ein kompatibler Player ist aber in nächster Zukunft geplant.

Das **Freezermodul Final Cadridge III erzeugt Fehler** beim Diskhandling im D4-Editor und D4-Player. Das Modul sollte daher nach dem Laden, vor dem Starten, mit dem Befehl KILL ausgeschaltet werden.

### Noch einige Vorschläge und Ideen zu deinem eigenen Adventure

**Stört der Befehl "Verliere"** kann dieser mit einer einfachen Allegemeinbedingung ausgeschaltet werden. Dazu wird einfach eine "leere" Bedingung mit dem Befehl "Verliere" erzeugt.

Kennst du dich mit Assembler aus, so kannst du auch **deine eigene Endsequenz "SQ"** erzeugen. Dazu muss dein File ab \$4000 bis \$C000 liegen. Gestartet wird das File bei der Adresse \$4000.

### Kontaktadressen

## **Independent Softworks**

Markenhofstr. 22  
79199 Burg am Wald

Tel. 076 61 - 931 11 - Fax. 931 12

# TECTRON

Dr. Zoom  
Föhrenweg 6  
CH-4514 Lommiswil  
Schweiz