Videospielgeschichten

Persönliche Geschichten über Videospiele

Ghost Town für den C16 – Showdown in JavaScript

Ingo Hinterding · Dienstag den 1. März 2016

1986

199 Mark. Soviel habe ich für meinen ersten Computer bezahlt, das war 1986, der Computer ein Commodore C16 samt Datasette und der Verkäufer der große Bruder von Stefan aus meiner Klasse. Es ist Nachmittag und wir stehen im Wohnzimmer seiner Eltern. Die Flügeltür einer erdrückenden Schrankwand schwingt auf und aus dem Familienfernseher erstrahlt schon bald ein blau-weißes Rechteck mit dem Hinweis:

```
COMMODORE BASIC V3.5 12277 BYTES FREE
READY.
```

Auf dem Teppichboden unter dem Bildschirm liegt in einem Gewirr aus Kabeln das, was schon bald mein liebstes Hobby werden sollte. Vor meinen staunenden Augen tippt Stefans Bruder etwas auf der Tastatur:

```
10 PRINT "WIE HEISST DU";
20 INPUT A$
30 PRINT "HEISST DU WIRKLICH "+ A$;
40 INPUT A$
50 PRINT "HAESSLICHER NAME!"
```

Fünf Zeilen BASIC, die meinen weiteren Lebensweg bestimmten. Und dann diese kleine schwarze Kiste, die tut, was man ihr sagt! So aufregend und ganz anders als unser erziehungsresistenter Rauhhaardackel oder „Husten-Markus“ zwei Häuser weiter, der, wie der Name schon sagt, meistens krank war und nur mit einem spielen wollte, wenn er auch garantiert alles bestimmen durfte, und zwar immer und alles. Und immer. Einen Tag und ein irgendwie-dann-doch gemeistertes Problemgespräch mit den Eltern später stand er dann bei mir im Kinderzimmer: mein erster Computer!

Der C16 unterschied sich vor allem in zwei Dingen vom drei Jahre älteren C64: ein hervorragendes BASIC 3.5 und eine nahezu nicht vorhandene Raubkopierszene. Wenn also ein Freund ein neues Spiel erstanden hatte, dann wurden zwei Kassettenrekorder zusammengesteckt und Leerkassetten bespielt. Wer ein Doppelkassettendeck hatte, war natürlich der König – aber Vorsicht vor „High Speed Dubbing“ – das klappte nur selten.

Gute Spiele waren also Mangelware und so vertrieb ich mir die Zeit vor allem mit selbst geschriebenen Programmen und Animationen in Zeichensatzgrafik. Der Computer war die Bühne, der Programmcode die Schauspieler und ich der Regisseur. Die Möglichkeiten schienen unendlich und das Handbuch verbarg auf jeder Seite neue Allmachtsphantasien in acht Bit. Andere Hobbies verloren zeitgleich jeglichen Reiz, getreu dem Motto: „Sommer ist, wenn man in kurzen Hosen vor’m Computer sitzt.“

Einen weiteren Vorteil des C16 gegenüber dem C64 konnte ich leider nie geltend machen. Mit 121 unterschiedlichen Farben war der C16 dem C64 mit seinen 16 Farben deutlich überlegen. Doch einige Wochen zuvor hatte ich mich von meiner vier Jahre älteren Schwester über’s Ohr hauen lassen: ich bekam den kleinen Schwarzweißfernseher und die Micky Maus Sitztonne, sie dafür den dicken Farbfernseher von Oma, die kurz zuvor zu den Ahnen abgeritten war. So ganz verarbeitet habe ich das wohl nie. Also, das mit dem Schwarzweißfernseher.

Ab und zu gab’s dann doch ganz gute Spiele, Winter Olympiade zum Beispiel war „megageil“ (1986 war schließlich das Jahr von Bruce & Bongos Nummer eins Hit), und die deutsche Firma Kingsoft nahm sich mit günstigen Spielen auf Datasette zwischen 9 und 39 Mark den unterversorgten Besitzern der 264er Serie (C16, C116 und Plus/4) an.

Ghost Town

Der Puls rast und das Herz pocht ganz oben im Hals, als ich auf dem Heimweg von der Schule im Schaufenster des Brillenladens ein neues Spiel für den C16 entdecke: Ghost Town. Das gesparte Geld reicht, das wusste ich, man ist als Zwölfjähriger auf solche Extremsituationen bestens vorbereitet. Da waren die fünf Mark vom letzten Kaffeetrinken bei Oma und mindestens nochmal soviel im Porzellansparschwein mit dem praktischen Gummipfropfen an der Unterseite – passt! Nach Hause radeln, schnell Mittag essen, Geld zusammensuchen, zurück zum Optiker (Mist! Der macht ja erst um 15 Uhr wieder auf!) und dann direkt weiter zu Hendrik. Der hatte einen Plus/4 mit 64kb Speicher und einen eigenen Fernseher (in Farbe!) im Hobbykeller – unser boy cave!

Computer eingeschaltet, Kassette in die Datasette eingelegt, los ging’s.

```
LOAD
PRESS PLAY ON TAPE
OK
SEARCHING
GHOST TOWN C16
LOADING
```

Der Fliesentisch, der zum Kleiderständer umfunktionierte Heimtrainer und die ausgediente Schreibmaschine verschwanden. Die farbigen Pixel des Fernsehers erfassten den ganzen Raum und zogen uns hinein in die geheimnisvolle Ghost Town.

```
„Suchen Sie die Schatztruhe der Geisterstadt und oeffnen Sie di
ese !
Toeten Sie Belegro, den Zauberer und weichen Sie vielen anderen
 Wesen geschickt aus.
Bedienen Sie sich an den vielen Gegenstaenden, welche sich in d
en 19 Bildern befinden.
Viel Spass !“
```

Über das Spiel selbst wurde schon oft und gut geschrieben, auch hier bei Videospielgeschichten. Wichtig ist vor allem eines: es stammt aus einer Zeit als das Wort „Perma-Death“ noch ein Pleonasmus war, denn jeder Tod brachte das unwiederbringliche Ende mit sich. Auch Ghost Town verzeiht keine Fehler, ein unbedachter Schritt, ein in der falschen Reihenfolge aufgenommenes Objekt und nach dem „blue screen of death“ erscheint wieder das Titelbild mit der höhnischen Aufforderung: „Play it again“.

Doch gerade diese Verlustangst mit dem ständigen Bildschirmtod vor Augen übt einen beeindruckenden Sog aus, eine Intensität, wie man sie heute vielleicht nur noch von Resident Evil kennen mag, welches den Spieler mit ständiger Munitionsknappheit ins Unbekannte schubst.

Für uns war Ghost Townein fast real erlebtes Abenteuer, auf das man sich am besten in Begleitung begab. Der eine hielt den Joystick fest im Griff (Tennisschweissbänder waren da schon fast Pflicht), der andere studierte die Notizen und Zeichnungen, die Fortschritt und Scheitern dokumentierten, gab Tipps und Hinweise, um dann gemeinsam auf die nächste Falle zu treffen und die beste Vorgehensweise zu besprechen, bevor man dann, durch die neueste Todesursache in Schockstarre versetzt, die Reise von Neuem antrat. Joystickübergabe, schnell noch fünf Salzstangen quer zwischen die Backen geschoben – und weiter.

Ich glaube wir haben das Geheimnis der Ghost Town nie gelüftet, Hendrik und ich. Irgendwann

kamen wir nicht weiter, der Joystick hatte ne Macke bekommen und das nächste Spiel wartete darauf, erobert zu werden. Das Rätsel um die Geisterstadt blieb uns verborgen.

„Wir sehen uns wieder Belegro! Und wenn es 30 Jahre dauert!“ drohte ich noch. Was natürlich nicht stimmt. Aber schön in die Geschichte passt.

2016

Ich könnte jetzt schreiben, ich wäre zu Besuch bei den Eltern gewesen, hätte den Dachboden entrümpelt und in der alten Kinderzimmerkiste meine Notizen über Ghost Town gefunden, mit dem Schwur and mein zukünftiges Selbst, “der Geisterstadt ihre Rätsel zu entlocken, und wenn es 30 Jahre dauern würde!”. Weil’s eben schön in die Geschichte passt. Aber auch, weil die Wahrheit manchmal etwas unglamourös daherkommt.

Inzwischen ist Softwareentwicklung mein Beruf und wenn ich neue Technologien verstehen möchte, suche ich mir ein kleines Projekt, welches genug Motivation mitbringt, die üblichen Durststrecken und WTF-Momente zu überstehen. Eine gute Gelegenheit, meinem Erzfeind Belegro einen erneuten Besuch abzustatten, beschließe ich. Man trifft mich eben immer zweimal.

Ghost Town JS

Jetzt wird’s ein wenig technisch. Das Ziel ist ein Remake des Spieles für den Browser, aus genannten Gründen kommen pixi.js als Engine und CoffeeScript als Scriptsprache zum Einsatz. Wichtigstes Tool neben einem Texteditor für den Code ist der Emulator (Yape soll besser sein als Vice, den gibt’s aber nicht für den Mac), um das Spiel zu analysieren und Daten zu extrahieren.

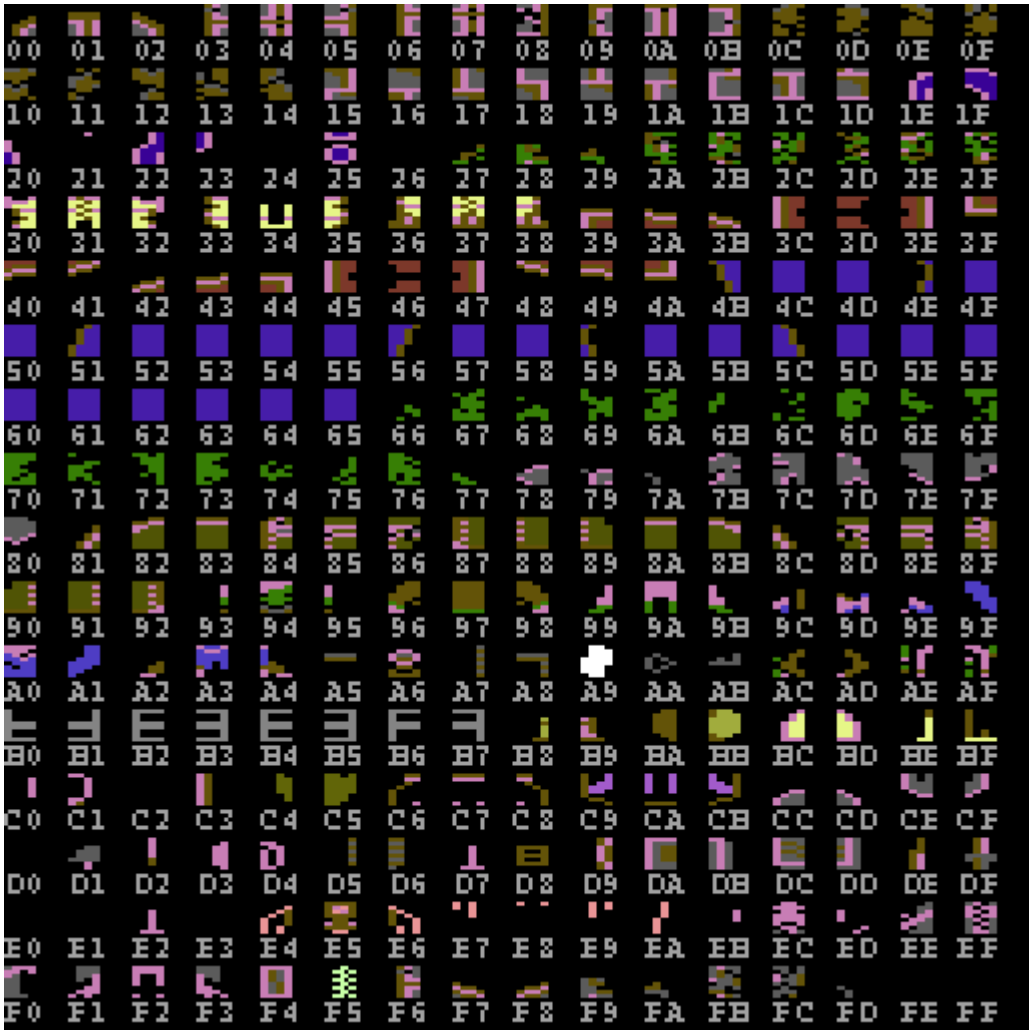
Ein paar Tutorials später stehen die Grundvoraussetzungen, nun geht es dem Originalspiel an den Kragen: wir betreten die Geisterstadt durch den Lieferanteneingang.

Die Welt von Ghost Town ist wie ein Puzzle aus 256 unterschiedlichen Zeichen zusammengesetzt, die im „Character RAM“ abgelegt sind. Das „Color RAM“ definiert, welche Farben an welcher Stelle des Bildschirms dargestellt werden. Das „Screen RAM“ zeigt, welche Zeichen an welcher Position angezeigt werden. Alle drei kombiniert ergeben das, was der Spieler auf dem Bildschirm sieht.

Im oberen Bereich des Spieles sieht man alle verfügbaren Zeichen von 00 bis FF. (Bild: Ingo Hinterding)

Ich entscheide, Character RAM (den Font) und Color RAM (die Farbe) mit einer Klappe zu erschlagen und zusammenzuführen, indem ich eine bereits colorierte Spritemap erstelle. Dazu extrahiere ich den Font, indem ich alle Zeichen des Spieles im Emulator darstelle, einen Screenshot mit 1:1 Pixeln (320×200) abspeichere und die Farbinformationen in Photoshop hinzufüge. Das Ergebnis ist eine Spritemap mit allen verfügbaren Zeichen, in der richtigen Reihenfolge sortiert von 00 bis FF (das wird später noch wichtig).

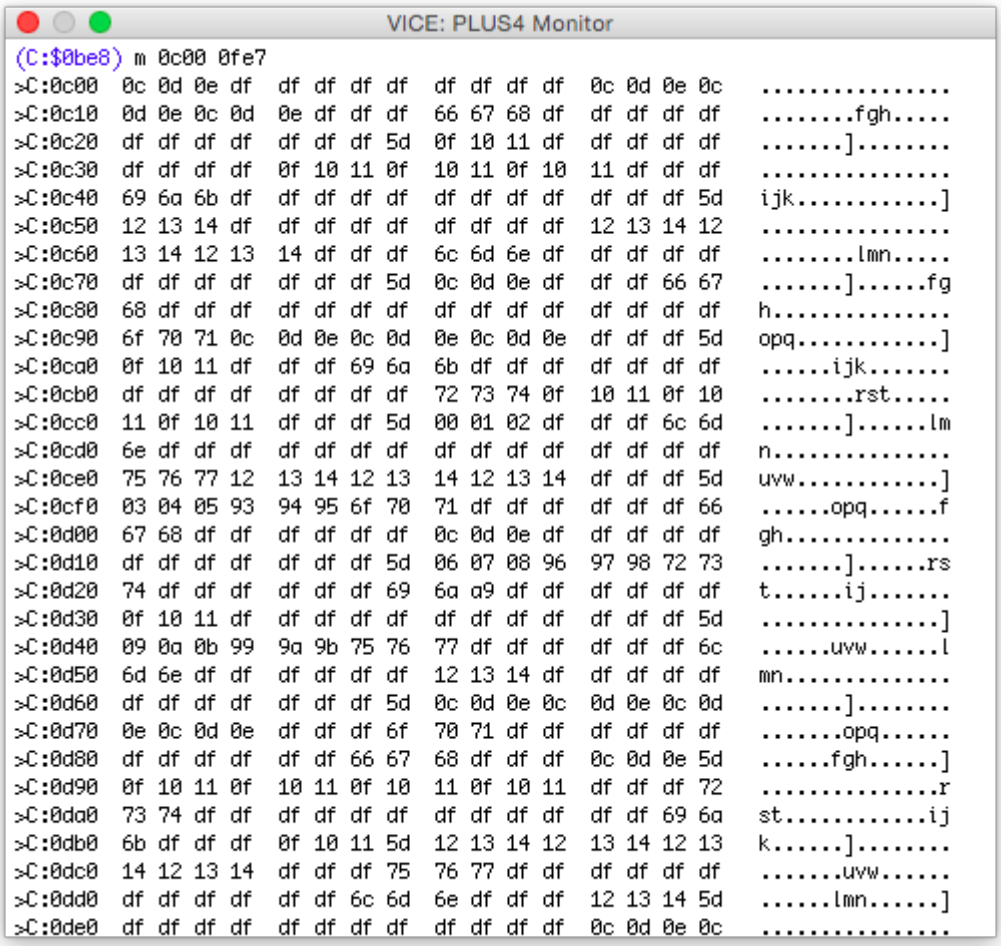
Copyright © 2024 Videospielgeschichten



Alle verfügbaren Zeichen fertig coloriert und geordnet in einem Bild. (Bild: Ingo Hinterding)

Einmal die rote Pille bitte

Als nächstes werden die Leveldaten extrahiert. Dazu kann man den Emulator einfach an der Stelle pausieren, die man auslesen will und lässt sich das Screen RAM (beim C16 ist das $0c00-$0fe7) anzeigen. Jeder zweistellige Hexadezimalwert steht für ein definiertes Zeichen. Wir haben uns also für die rote Pille entschieden und blicken nun wie Neo durch eine Matrix aus Code auf die Welt im Computer. Wir können an jede beliebige Stelle springen, den Bauplan ändern und der ahnungslosen Spielfigur das Leben schwerer oder leichter machen. Ist hier eine Wand im Weg? Flink „0c 0d 0e“ durch „df df df“ ersetzt und schon haben wir freie Bahn. Da sind sie wieder, die Allmachtsphantasien.

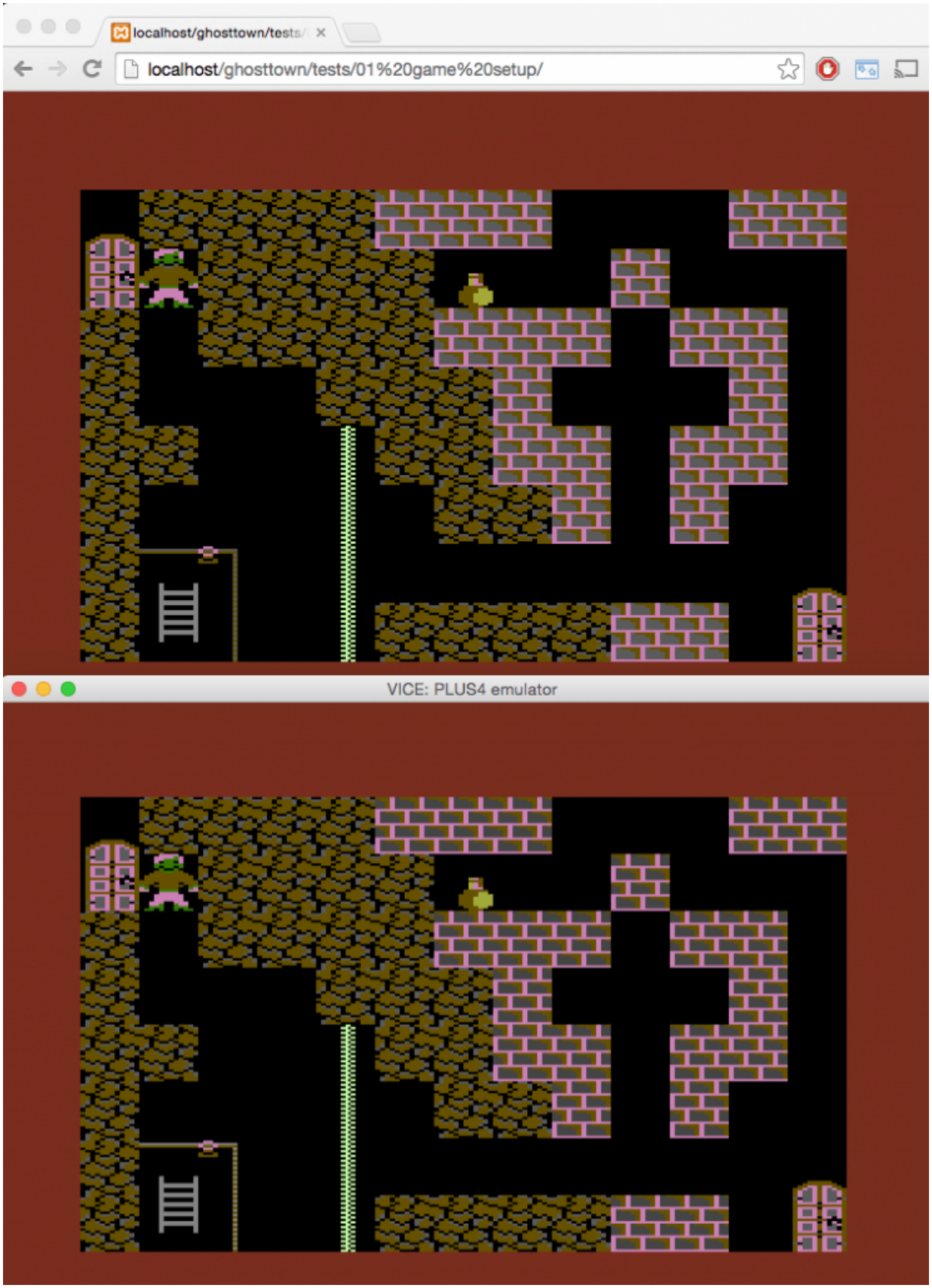


Der erste Raum des Spieles in der Hexcode Ansicht. (Bild: Ingo Hinterding)

Wir wollen aber das unveränderte Spiel umsetzen, also speichern wir jeden Screen mittels save „/Users/name/ghosttown/level-xx.bin“ 0 0c00 0fe7 ab. In Summe kommen wir auf 19 Räume, 30 Nachrichten auf deutsch, 30 auf englisch, Einleitung und Siegesscreen auf deutsch und englisch sowie ein Titelbild, insgesamt also 84 unterschiedliche Bildschirme. Die Verwendung der Binärdaten birgt Vor- und Nachteile. Zum einen müssen diese nur sauber eingelesen, verarbeitet und angezeigt werden, um eine originalgetreue Darstellung zu garantieren. Kein Pixel kann sich so am falschen Platz befinden. Der Nachteil der Methode ist, falls man etwas übersehen hat, dann existiert es auch im Remake nicht.

Die Darstellung der Levels ist nun recht simpel: Binärdaten einlesen und Zeile für Zeile das entsprechende Zeichen aus der Spritemap auf dem Bildschirm anzeigen. Wenn alles funktioniert, hat man eine exakte Kopie der Spielegrafik.

Copyright © 2024 Videospielgeschichten



Im Vergleich: oben die Ansicht im Browser, unten im Emulator. (Bild: Ingo Hinterding)

Jetzt kommt Leben in die Bude

Da die Spielfigur selbst auch nur aus Zeichensatzgrafik zusammengesetzt ist, können wir sie

einfach an die gewünschte Stelle in die Leveldaten kopieren und verschieben, je nachdem, in welche Richtung sich der Spieler bewegen möchte. Wird etwa die rechte Pfeiltaste gedrückt, schauen wir, welche Zeichen sich rechts neben der Spielfigur befinden. „df“ steht etwa für ein leeres Feld und die Spielfigur kann an die neue Postion kopiert werden. Befindet sich ein Hindernis im Weg (Busch, Mauer, Wasser usw.) geht es erstmal nicht weiter.

Dieses einfache Prinzip ist der Schlüssel für die gesamte Logik des Spieles. Der Ablauf ist immer gleich und funktioniert in etwa so:

```
1. Der Spieler drückt eine Richtungstaste
2. Ist das Feld in dieser Richtung frei?
3. Ja: Spielfigur bewegen
4. Nein: Zeichencode prüfen und je nach Raum die passende Aktio
n auslösen
```

Das Geheimnis der Eierpflanze

Jetzt bauen wir nach gleichem Muster die Rätsel im Spiel auf. *Ghost Town*lässt sich nur wenig Information entlocken und daher wussten damals die wenigsten Spieler, was denn dieses weiße Ding im ersten Raum darstellte. Die handgezeichnete Komplettlösung von Kingsoft selbst verrät es (Achtung Spoiler!): es handelt sich um einen Handschuh, den man benötigt, um unverletzt an die Heckenschere zu gelangen. An den Handschuh wiederum gelangt man nur mit der Leiter, für die wiederum… usw. usw.

Die Logik für den Handschuh sieht also grob so aus:

```
1. Der Spieler bewegt sich zum Handschuh
2. Befindet sich die Leiter im Inventar?
```

```
3. Ja: Handschuh aufnehmen, Leiter wegwerfen
```

Der tatsächliche Code liest sich fast genauso einfach:

```
if "a9" in new_position
  if "ladder" in player.inventory
    player.add "gloves"
    player.remove "ladder"
```

Daß es sich beim Handschuh um das Zeichen „A9“ handelt, kann man leicht an der Spritemap ablesen. Nach diesem Prinzip ist jeder Raum des Spieles aufgebaut.

Das wahre Geheimnis der Ghost Town

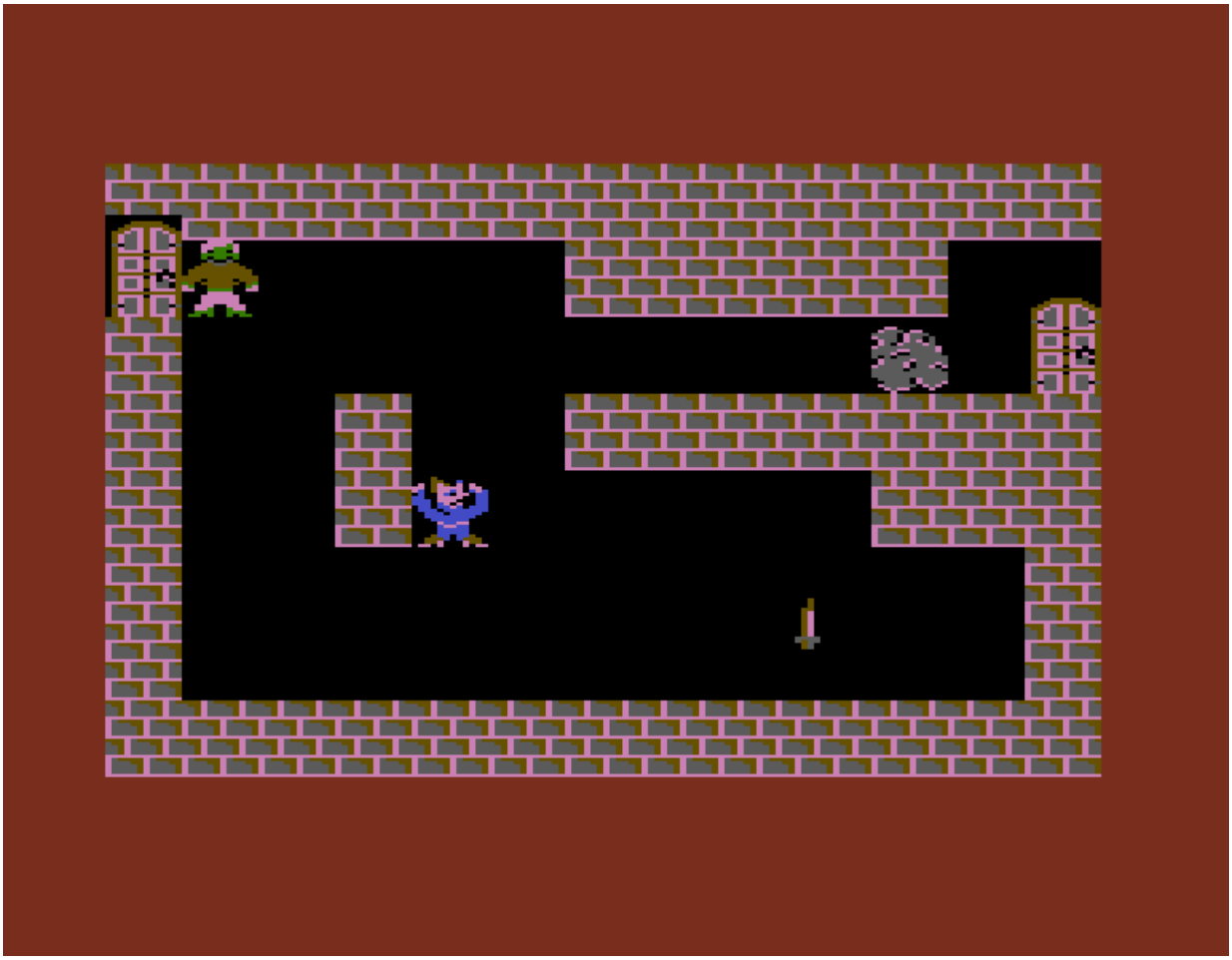
Ich merkte bald, daß Udo Gertz, der Entwickler des Originals, genauso vorgegangen sein musste wie ich, denn überall im Hexcode verstreut fand ich kleine hilfreiche „Geheimzeichen“: so ist etwa eine bestimmte Tür im Spiel die sichere Passage in den nächsten Raum, äußerlich jedoch nicht von drei weiteren Türen zu unterscheiden, die den sofortigen Tod bedeuten. Schaut man jedoch in den Zeichencode, entdeckt man, daß diese eine Tür ein anderes Zeichen benutzt, welches für den normalen Spieler identisch aussieht.

Da war ich also fündig geworden und entdeckte 30 Jahre später die wahren Geheimnisse der Ghost Town. Vor mir hat sie schon Udo Gertz durchstreift und überall versteckte Kreidezeichen hinterlassen, die mich nun an die Hand nehmen, vor Gefahren warnen und Lösungen anbieten, die so einfach wie genial sind.

So folge ich ihm durch seine Welt und meistere Raum für Raum, lerne, verstehe. Die Herausforderungen steigen, doch sie sind nicht mehr unüberwindbar, bis ich schließlich wieder auf den letzten Gegner treffe, der sich mir schon als Kind in den Weg stellte:

Belegro

Am Ende der Reise wartet Belegro, der böse Zauberer, auf den zittrigen Helden. Ihn zu besiegen, erfordert exaktes Timing und Fingerspitzengefühl, denn eine falsche Bewegung und der Magier verfolgt den Spieler, holt ihn ein und beendet das Abenteuer auf der Zielgeraden.



Showdown. (Bild: Ingo Hinterding)

Nun stehe ich wieder am Ort des Geschehens und Udo tritt einen Schritt zurück in den Hintergrund. Für die letzte Prüfung hat er keine hilfreichen Zeichen hinterlegt. Ich muß es allein schaffen.

Damit das Spiel möglichst nah am Original bleibt, sollte sich der Endgegener auch genau so verhalten. Ist er zu langsam, ist das Ende zu einfach. Ist er zu schnell, ist es unschaffbar. Belegros Bewegungslogik lässt nur eine Spieltaktik zu, und eben diese muß auch im Remake genau so funktionieren. Ich beobachte Belegros Verhalten und leite daraus Regeln ab, mit dem flauen Gefühl, an dieser Stelle erneut zu scheitern.

Doch dann wird mir klar: für jede Hürde im Spiel gibt es eine einfache Lösung, egal ob vor oder hinter der Bühne. Also erkläre ich „meinem“ Belegro:

```
1. Gehe in die Richtung des Spielers, wenn der Weg frei ist
2. Ist der Weg versperrt, warte bis der Spieler sich bewegt
3. Weiter bei 1.
```

Ich betrete den Raum und treffe auf den gleichen bösen Zauberer, der mich schon als Kind verfolgte. Er hat sich kein bisschen verändert, fast unheimlich, aber ich habe dazu gelernt, kämpfe, triumphiere und verlasse – **lebend!**– die *Ghost Town*.

Vielen Dank Udo Gertz für dieses großartige Erlebnis!

GhostTownJS kann hier gespielt werden: http://www.awsm.de/ghosttown

PS: Wer gerne mehr über die technische Umsetzung des Spieles erfahren möchte, kann sich den kommentierten Sourcecode auf Github herunterladen: GhostTownJS auf Github.

Ingo Hinterding // Twitter: @awsm9000

Dieser Beitrag wurde publiziert am Dienstag den 1. März 2016 um 18:22 in der Kategorie: Spielebesprechungen. Kommentare können über den Kommentar (RSS) Feed verfolgt werden. Du kannst zum Ende springen und ein Kommentar abgeben. Pingen ist momentan nicht erlaubt.