Doom Games

Some games that I tinkered with

Gerph, May 2024

Doom Games

Some games that I tinkered with

Gerph, May 2024

0. Introduction

0. Introduction

Good evening.
This will hopefully be an interesting presentation about the games I ported back in the late 90s, and have recently been updating.

Introduction

How I'll do this talk

I'll be talking about Doom, Heretic and Hexen. There are 7 parts to the talk...

1. Leaked Doom
2. Doom+
3. Heretic
4. Hexen
5. One off pressie
6. Hexen (2024)
7. Heretic (2024)

Introduction

How I'll do this talk

I'm going to talk about the games Doom, Heretic and Hexen. I'll try to make it interesting, without too much technical detail.

Please feel free to use the chat to ask questions as we go. I will try to pay attention to it and answer anything that comes up.

At the end of the presentation, these slides and a bunch of links to resources will be made available.

Introduction

Who am I?

- I'm Charles, but known as Gerph in most things online.
- I worked on these games, numerous patches and applications back in the day.
- I worked at RISCOS Ltd, and produced RISC OS Select.
- I've written the only other implementation of RISC OS - RISC OS Pyromaniac.
- Began working on Doom in 1997.

Introduction

Who am I?

First though, let me introduce myself. I'm Charles, previously known as Justin, better known as Gerph. I've done work on RISC OS since the early 90s, patching and porting things, developing RISC OS itself, and most recently creating a new implementation of the OS - RISC OS Pyromaniac.
In many respects, my later work with RISCOS Ltd is directly tied to having done work on the Doom games. It all goes way back to University around 1997.

1. Doom

Early 1997

1. Doom

Early 1997

Doom

What is Doom?

- Doom is a first person shooter, by id Software.

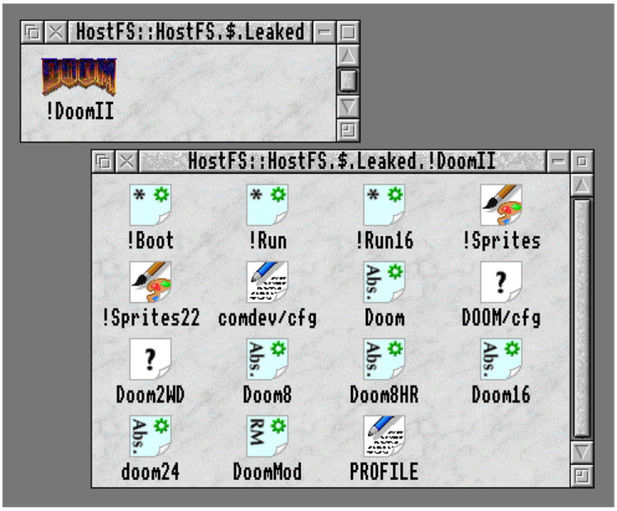
Doom

What is Doom?

Doom is a game - a 3D first person shooter - written by id Software. The game was released in 1993 for DOS and it changed the way that games were viewed on the PC. id Software's prior game, Wolfenstein 3D, had been ported to RISC OS by Eddie Edwards. It was a success in the RISC OS world, and Eddie obtained code for Doom, but the license costs were high, so it didn't get released.
This was back in 1997, whilst I was at University. My friend Chris and I got a leaked copy of Doom. This was a version from Eddie Edwards port - it wasn't ready for release and was really in the middle of development.

Doom

Leaked RISC OS Doom



Doom

Leaked RISC OS Doom

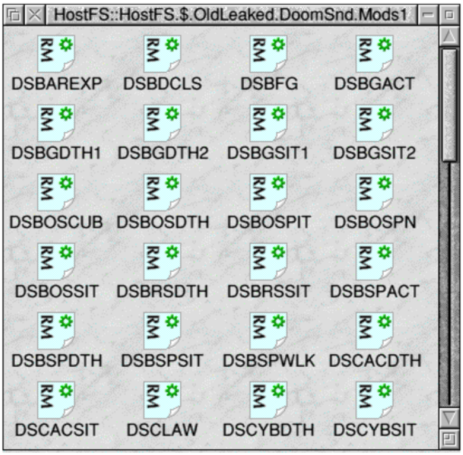
It worked on the RiscPC and A5000, but only had sound on the RiscPC. It wasn't too bad speed-wise, either. I only had an A5000, but Chris had a RiscPC. The difference between them was quite marked - and the sound made a big difference. I wasn't one to take such things as they were, and began investigating.

I worked through the game binary and began to understand how the sound was meant to work and what the code did to trigger effects. With a few small patches - and quite a bit of data extraction from Doom's data files - I created a version that played sound on the A5000.

Doom

Sound samples

- I created many sound sample modules to play the game's sound.



HostFS::HostFS.$.OldLeaked.DoomSnd.Mods1

DSBAREXP	DSBDCLS	DSBFG	DSBGACT
DSBGDTH1	DSBGDTH2	DSBGSIT1	DSBGSIT2
DSBOSCUB	DSBOSDTH	DSBOSPIT	DSBOSPN
DSBOSSIT	DSBRSDTH	DSBRSSIT	DSBSPACT
DSBSPDTH	DSBSPSIT	DSBSPWLK	DSCACDTH
DSCACSIT	DSCLAW	DSCYBDTH	DSCYBSIT

Doom

Sound samples

Because I wasn't very familiar with the sound system, I used standard RISC OS sound voices modules - one module for each sample that could be played. The RISC OS sound system only supports having 32 sound voices loaded at once - and in any case, the memory is quite pushed on a 4MB A5000 so you couldn't load too many. So my patches would load the modules on demand, and kill off any that weren't needed any more as the system become full.
I might not be able to do the 16bit colour that Chris' RiscPC could do, but my A5000 now had sound.

Doom

Networking

- Networking required a separate module to do the communication.
- Initially networking was Econet based.
- Serial and UDP based modules were also created.
- A management tool that could negotiate with other players was created.
- At home, my brothers and I played.
- It was also tested with friends - we organised games over IRC.

Doom

Networking

Looking through the code, the networking was intriguing. It appeared to be able to do networking if you just passed the data in and out properly. What 'properly' was, however, was more difficult - id Software had made available code for network drivers but for DOS only.

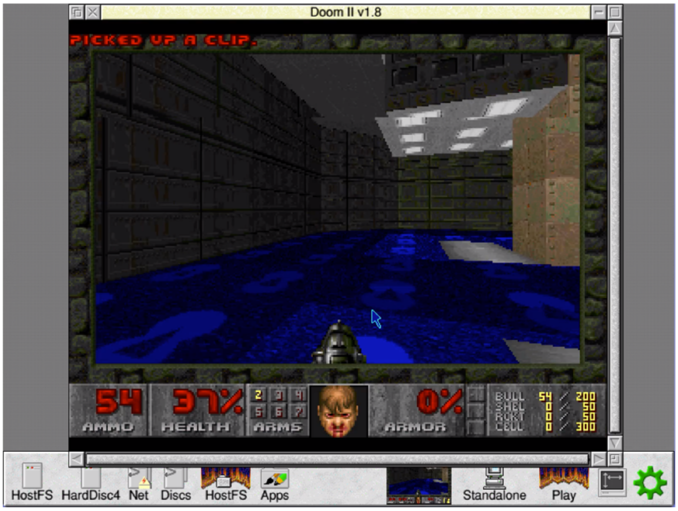
Using this information, I managed to patch in some networking using Econet. With two copies of Doom, one on each A5000, you could connect via the Econet network and play multiplayer games against one another. My brothers and I had a lot of fun with that.

The networking was later expanded to support playing over a serial port and over IP, using UDP. Using IP networking on an A5000 was something of a strain, but was just about doable. You have to also remember that this was in the days of dialup networking, so it was even more of a challenge.

Fortunately, around this time I bought a RiscPC, which began to make things a lot easier to work with.

Doom

DoomWads launcher



Doom

DoomWads launcher

Having got networking working, I continued to tinker with the code, and added support to allow the game - which had been exclusively full screen - run within the Desktop. Not something that was very practical on an A5000, but was playable on a RiscPC. And you could even play the game on the iconbar - a tiny sprite could be used instead of the window.

I wrapped all these bits together with a launcher application which could manage the extension 'PWADs', and integrated the network selector application as well.

<indicate the parts as they are described>

You can see here my DoomWads launcher and the game window. The computer icon on the iconbar would control the network settings. The tiny iconbar icon here reflects the game as you're playing.

I passed copies of this to a few people, and the attention that the game had been gathering (not only from my work), meant that R-Comp got interested in properly releasing the game. Eddie and R-Comp released Doom to the RISC OS world not long after that.

Doom

R-Comp releases Doom

- R-Comp organised things with Eddie for a proper release, fully licensed.
- I was asked to write proper network drivers - after my final year exams finished.
- I updated the launcher to give it better support for PWADs.

Doom

RComp releases Doom

Having got a proper release, I was asked to write some network drivers for the game, as my patches weren't really re-usable. Eddie had done a quite nice job of making the network drivers modular, so I could reuse some of the code that I had created previously. I remember at the time being repeatedly asked if I'd looked at the network drivers, but I was trying to focus on final year exams. In a rare case of will power, I managed to put off working on them until after the exams had finished.
Returning home after University, I'd not worked out what I wanted to do for a job. Speaking to Andrew, he let me have the source to the front end to address some bugs, as Eddie had moved on to other things. I added some of the features of the DoomWads launcher like the patch WAD manager and text viewer. This was quite successful, so I took on enhancing Doom itself.
This was what would become Doom+.

2. Doom+

July 1998 - May 1999

2. Doom+

July 1998 - May 1999

11 / 53

Doom+

What was Doom+? (1)

The goal of Doom+ was to add cool things to the game. But also...

- It got faster.
- It got better PWAD support.
- It got smoothed walls.
- It got an in-game map.
- It got better networking.
- It got fixes.

Doom+

What was Doom+? (1)

What was Doom+ ?
The remit for Doom+ was just to make things cooler, really. Fixing some bugs, removing limitations, making the game faster, and adding new features were all on the cards.
There's a lot that I've already written about Doom+ in my Rambles, but there's a couple of little things that are interesting about it. At the time, CD ROMs were far more common than MIDI interfaces, and I often listened to other music whilst testing the game. So I added in CD controls to the game as well.
Not only could you control the CD in game, but if you were running in the desktop, it would communicate with the !CDDB application and find out what the tracks were called, and provide a notice of the new track name when they changed.
There's a comment in my notes:
...

Doom+

What was Doom+? (1)

- Doom+ was meant to add cool things to the game.
- It got faster.
- It got better PWAD support.
- It got smoothed walls.
- It got an-in game map.
- It got better networking.
- It got fixes.
- It got a CD juke box.

In my notes:

> *Doom, one of the few Acorn games with a built in CD jukebox.*

...

> *Doom, one of the few Acorn games with a built in CD jukebox.*

With the CD music, I also tried to make the game a little more immersive by making 'atmospheric music'. This was done by bumping the volume as you came to the action. It only kinda worked, but it was an interesting experiment.

After adding the smoothed textures I began to notice something odd about parts of the display. Blue pixels would shimmer sometimes within the smooth texture. I couldn't work out what on earth was going on. I spent some time reworking the code, adding in diagnostics and saving out screenshots. Nothing useful came of them. After a few weeks of looking at this on and off, I tried the same game on a different machine... and the problem vanished. I assume it was some odd timing effect on my machine, but I wasted a whole bunch of time on that.

I really wish I'd had some tests to exercise the code somehow around there. If I'd had tests to show that things were working, I'd have trusted the code and not spent so much time looking for the problem.

Doom+

What was Doom+? (2)

- In-game clock.
- Deathmatch counters for kills.
- Named players in network games.
- Monster and secret counts on screen.
- Translucent fireballs and other effects.
- Bouncing menu titles.

Doom+

What was Doom+? (2)

There were lots of other things as well in Doom+. The in-game clock, improved network presentation, better map information, transparent fireballs, and even silly effects that bounced text in the menus. The development was quite long, and there's a comment in my notes from what was going on at Acorn...
...

Doom+

What was Doom+? (2)

- In-game clock.
- Deathmatch counters for kills.
- Named players in network games.
- Monster and secret counts on screen.
- Translucent fireballs and other effects.
- Bouncing menu titles.

A contemporary note about Acorn...

> *Lethargy is setting in at the moment. It's so hard to concentrate when you are always wondering what the point is. I know I am continuing on this, but since Acorn isn't Acorn anymore. Well... you get the idea.*

...

> *Lethargy is setting in at the moment. It's so hard to concentrate when you are always wondering what the point is. I know I am continuing on this, but since Acorn isn't Acorn anymore. Well... you get the idea.*

This was not long after the September 17th announcement of the closure of the workstations division.

Doom+

Release

- Doom+ came second in the Acorn New Service awards for 1999.
- Acorn Arcade's review gave the Doom+ release a rating of 83%.

Doom+

Release

Despite the Acorn announcement, and the extra work leading up to RISCOS Ltd's formation, a couple of releases were made.
Doom+ even came second in the Acorn New Service awards for 1999.
Acorn Arcade's review gave Doom+ a rating of 83%.

3. Heretic

October 1998 - February 1999

3. Heretic

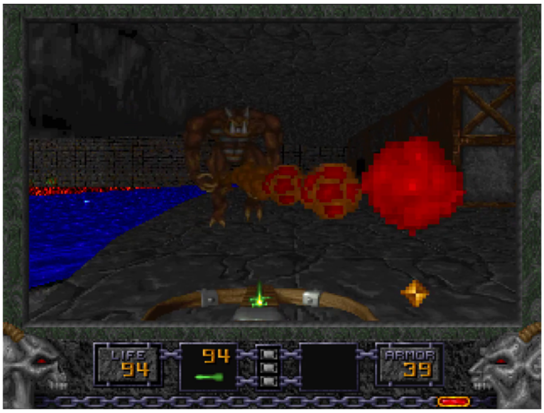
October 1998 - February 1999

And so we come to Heretic...

Heretic

What is Heretic?

- Doom, in a fantasy setting.
- Licensed from Raven software by R-Comp.



Heretic

What is Heretic?

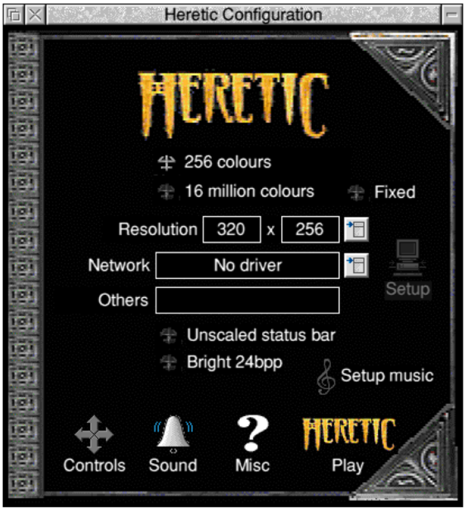
Heretic wasn't a game I knew. It's essentially the Doom engine, but with fantasy elements instead of science fiction and hell-based themes. Whilst I'd been working on Doom+, Andrew had been in negotiations with Raven Software who created the Heretic and Hexen games. When the game was finally licensed, we got the source code and I could start looking at how to port it.

After a false start, I managed to get a rudimentary game up and running in a few hours. It didn't have any input, but it ran.

There followed a couple of months more of development to get the input, rendering and sound working properly in the game. And mostly it went quite well - even with the distractions getting Doom+ finished as well.

Heretic

Launcher



Heretic Configuration

HERETIC

256 colours

16 million colours

Fixed

Resolution 320 x 256

Network No driver

Others

Setup

Unscaled status bar

Bright 24bpp

Setup music

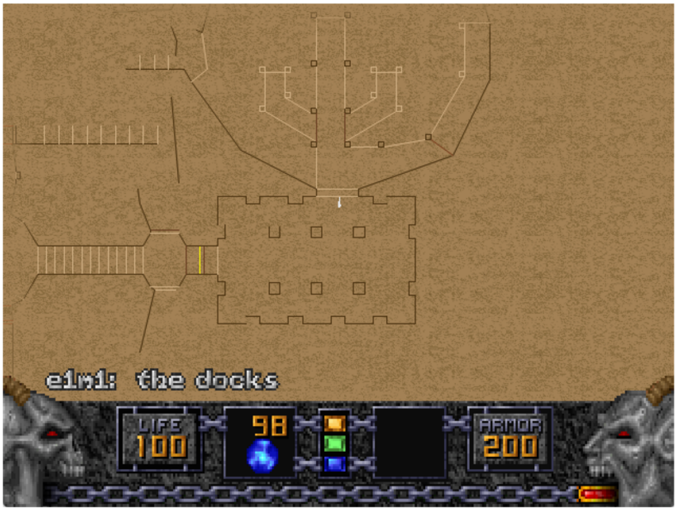
Heretic

Launcher

By the end of December, the game had most of the rendering integrated from Doom, with the Heretic enhancements, and a cut down and modified version of the Doom+ launcher had been integrated.
That screenshot is actually from the old version of !Heretic running on RISC OS Select. It looks a lot better on RISC OS Select because the shaded icons blend into the window backgrounds.
<point to the 24bpp option as we get there>
My notes from Heretic lament how the game is too dark some of the time. This became a constant problem in Heretic - and later Hexen. I tried a number of ways to improve the lighting, resulting in a 'bright 24bpp' option, which would bump the lighting to twice its expected level. However, because of how it worked it didn't address the real issue. The lighting issue wasn't actually fixed until this year.
There were interesting features in Heretic, that Doom had missed out on.

Heretic

New features



Heretic

New features

Heretic had anti-aliased smoothing of the map lines, and even faded out at the edges of the map. Seeing this, I integrated the same code into Doom's map and the overlay map became feasible - it could fade at the edges to make the overlay less jarring to the user.

Heretic had fun game features that Doom didn't have, particularly for the water. There were water currents that pushed you around, and when you were in the water, it would obscure the character's feet, making it seem like it had actual depth.

As Heretic was being development whilst I was finishing Doom+, many of the networking fixes were passed through to the drivers for Heretic as well.

Heretic was released in February, with some updates and fixes in March and May. However, Heretic was ported alongside Hexen...

4. Hexen

October 1998 - February 1999

4. Hexen

October 1998 - February 1999

Hexen

What is Hexen? (1)

- So much more than Heretic.
- Licensed from Raven Software.

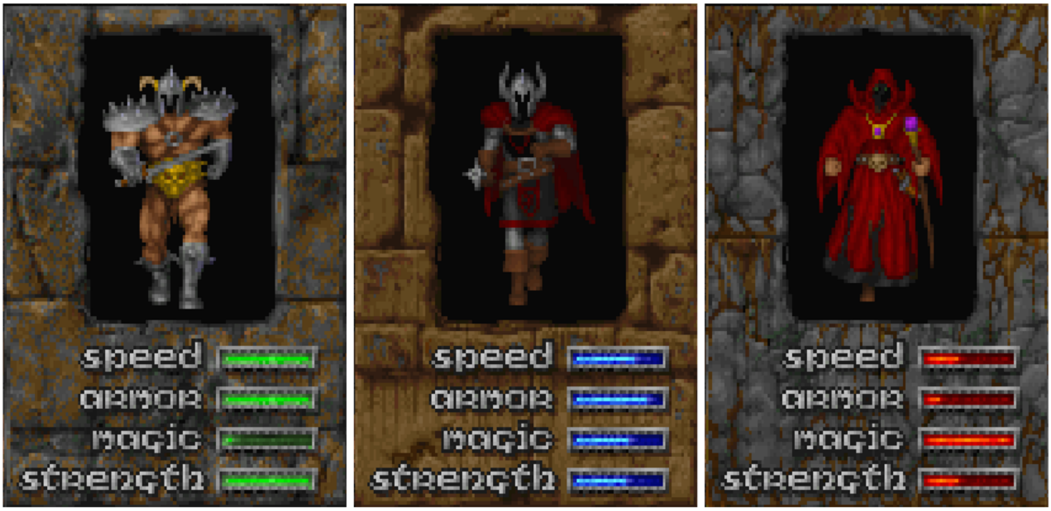
Hexen

What is Hexen? (1)

For background, Hexen is a continuation of the fantasy elements that were started in Heretic, but expanded even more. Instead of just having simple linear levels, there were puzzles in different areas that could be completed in different orders.

Hexen

What is Hexen? (2)



Hexen

What is Hexen? (2)

And as a player you can choose one of three 'classes' of character to play - a fighter, a cleric and a mage. They have different weapons and strengths.

Hexen

Development

- Initial work was done at the start of Heretic development - proving it worked.
- Focused on Heretic before returning to Hexen.
- Initial version only got 20 frames per second at 320x200.
- Very slow startup.
- Having learnt a lot from Heretic, Hexen then became easier.
- Lots of optimisations to apply from Doom and Heretic.

Hexen

Development

The Hexen source code arrived at the same time as Heretic, and much of the development was done in parallel. Although Heretic was the primary area I was work in, as it was a simpler game, what I learnt from Heretic could go into Hexen as well.

At the start of Heretic development, I got things working in just 6 hours. So I spent a day with Hexen, and I got Hexen to a similar point as Heretic. It ran the demo and the game was effectively working - just not very fast. Then I focused on Heretic - I didn't come back to Hexen until over a month later.

Hexen was slow.

My notes say that on my StrongARM machine I was getting 20 frames per second at 320x200 - and it took 20 seconds just to load data on startup. The startup was easy to fix by integrating some of the cached loading from Doom, and added the improved memory debugging from Heretic.

Hexen

New Features

> *WTF is a PolyObject ?!*

Hexen

New Features

There's a comment in my notes...

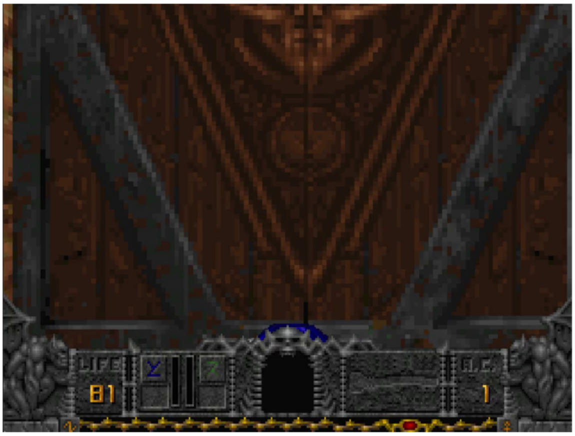
> *WTF is a PolyObject ?!*

Heretic had a lot of things that were the same as Doom. Hexen was more complex, in many different ways, and this was my introduction to one of the map objects which was far more advanced.
...

Hexen

New Features

> *WTF is a PolyObject ?!*



...

A `PolyObject` is a part of the map which is able to have effects applied to it - they could slide or rotate in ways that were not supported by the typical Doom engine. This made for doors that could swing open, or rotating barriers.

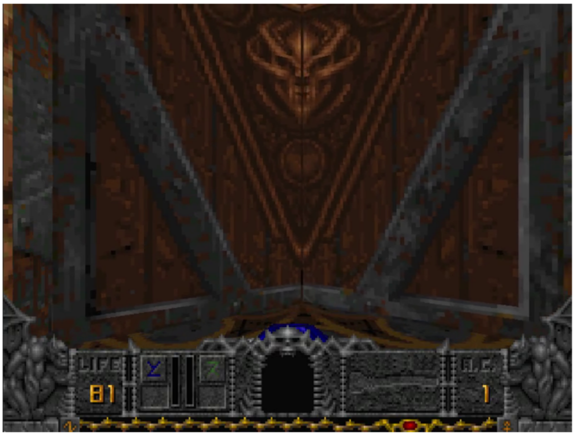
Here, there's a door that we can open. Unlike in Doom, the door will swing open when pushed.

...

Hexen

New Features

> *WTF is a PolyObject ?!*



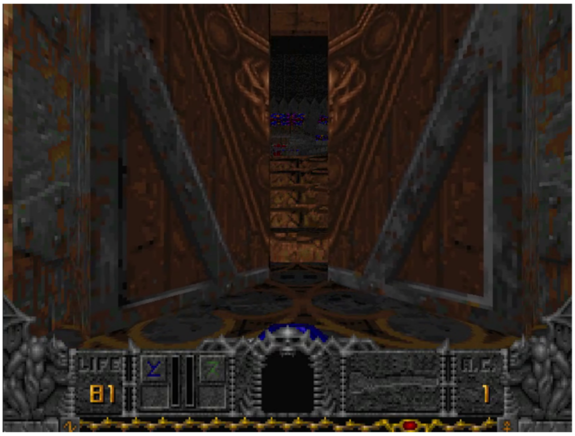
…

…

Hexen

New Features

> *WTF is a PolyObject ?!*



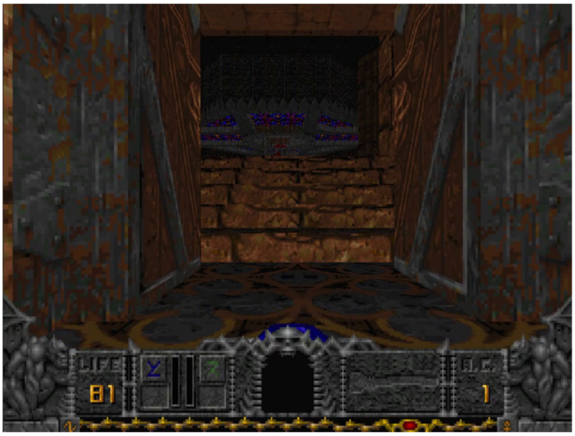
...

...

Hexen

New Features

> *WTF is a PolyObject ?!*



...

However, this feature was made even more powerful by another Hexen specific enhancement - scripting. Different events, like hitting things, pressing buttons or even randomly delayed events, could trigger a script defined in the level. For example, this meant that walls could be pulled away in sequence, regions could rotate, monsters could be spawned or doors opened all in a scripted manner.

When you hit the windows in Hexen the glass will shatter and fragments will fall to the ground. Trees can have leaves blow off them and drift across the landscape.

This made the maps in Hexen both more complex, and more interesting. It made the game a little slower as a result. Some parts of this code could be optimised, but it didn't have the big gains that the main rendering system got by porting from Heretic.

Heretic and Hexen

Release

- My notes say a StrongARM got 15.4 FPS at 640x480 in 256 colours.
- Heretic and Hexen were released together in a pack.
- An Acorn Gaming review back in '99, they were given a 5 out of 5 rating.
- An Acorn Arcade review gave the pack an 80% rating.

Heretic and Hexen

Release

The final timing I've got in my notes for Hexen is that on a StrongARM in 8 bit I was getting 15.4 frames per second for 640x480. Honestly, that's a little low, but for the complexity and size of the display, that's not too bad.
Heretic and Hexen were released together in a pack. In an Acorn Gaming review back in '99, they were given a 5 out of 5 rating.
An Acorn Arcade review gave the pack an 80% rating.

5. One-off pressie

December 1999

5. One-off pressie

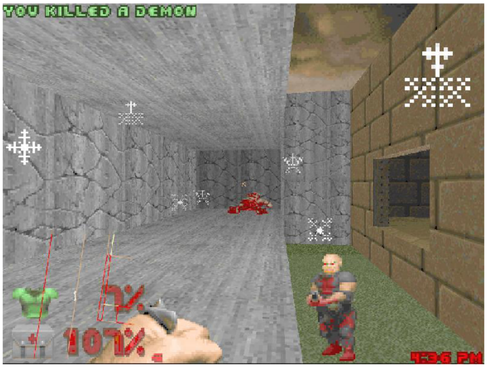
December 1999

The was also a one-off present that I created at the end of the December 1999.

One-off pressie

Christmas release

- Only went to a few testers.
- Had falling snowflakes.
- Introduced the jump key.



One-off pressie

December 1999

On Christmas day 1999 I sent around a special release of Doom+ to friends and testers. This was the 'Pressie' release. It had a number of updates that hadn't been seen in public yet.

There were some new launcher options, a 'jump' option, looking up and down, extra wipe effects... and there was snow.

I don't believe those features ever saw a wider release at the time.

Before we move on to the modern developments does anyone have any questions about the old games?

6. Hexen (2024)

January 2024 - February 2024

6. Hexen (2024)

January 2024 - February 2024

We jump forward now to the start of 2024...

Hexen (2024)

Why go back to Hexen?

- Returning to things that I've enjoyed is good for my mood.
- A different sort of challenge.
- I actually get to see something concrete that I've achieved!

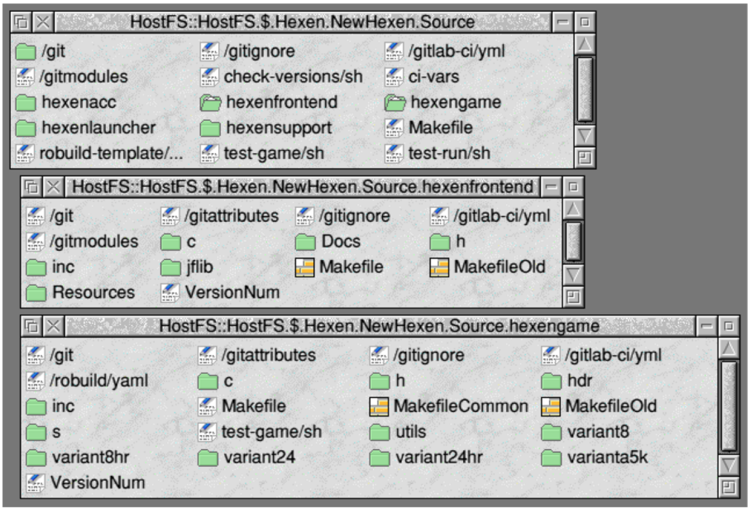
Hexen (2024)

Why go back to Hexen?

At the start of January, my fiancee suggested that I try doing something else that I enjoy. I'd been feeling pretty down for a while and she knew what would help. So I dusted off the Hexen sources - the only ones that were under Git source control - and started updating them for a more modern build and to run as a 32bit application.

Hexen (2024)

Anatomy of a game



HostFS::HostFS.$.Hexen.NewHexen.Source

/git /gitignore /gitlab-ci/yml
/gitmodules check-versions/sh ci-vars
hexenacc hexenfrontend hexengame
hexenlauncher hexensupport Makefile
robuild-template/... test-game/sh test-run/sh

HostFS::HostFS.$.Hexen.NewHexen.Source.hexenfrontend

/git /gitattributes /gitignore /gitlab-ci/yml
/gitmodules c Docs h
inc jflib Makefile MakefileOld
Resources VersionNum

HostFS::HostFS.$.Hexen.NewHexen.Source.hexengame

/git /gitattributes /gitignore /gitlab-ci/yml
/robuild/yaml c h hdr
inc Makefile MakefileCommon MakefileOld
s test-game/sh utils variant8
variant8hr variant24 variant24hr varianta5k
VersionNum

Hexen (2024)

Anatomy of a game

There's a number of parts that go into making the games work.
<point to bits as you go - xc for the cursor toggle>
Here you can see part of the structure of Hexen - this is the top level component that actually builds the release application that I send on to RComp. It's made up of a few bits - there's the game binary itself, the support module that provides sound and input, the desktop launcher, and there are other tools that we want to release.
And there are other parts that make up the testing - scripts to run the game and report whether things worked or not.
Each of these parts - like the front end here - has sources in it. It can be build and used independantly of the rest of the game's code.
And there's the game itself, here. This is slightly more complex because the same source code builds 5 different variations of the game - 24bit, 8bit, high resolution, fixed resolution and the A5000 version. Or actually 10 different versions, because it can be built as debug or release.

Hexen (2024)

Converting to 32bit (1)

- For C code, add `-apcs 3/32` and link with StubsG.
- For assembler, also add `-apcs 3/32` but then the code needs updating.

Instead of:

```
LDMFD   fp, {v1-v2, fp, sp, pc}^
LDMFD   sp!, {r0-r4, pc}^
MOVS    pc, lr
```

You use:

```
LDMFD   fp, {v1-v2, fp, sp, pc}
LDMFD   sp!, {r4-r5, pc}
MOV     pc, lr
```

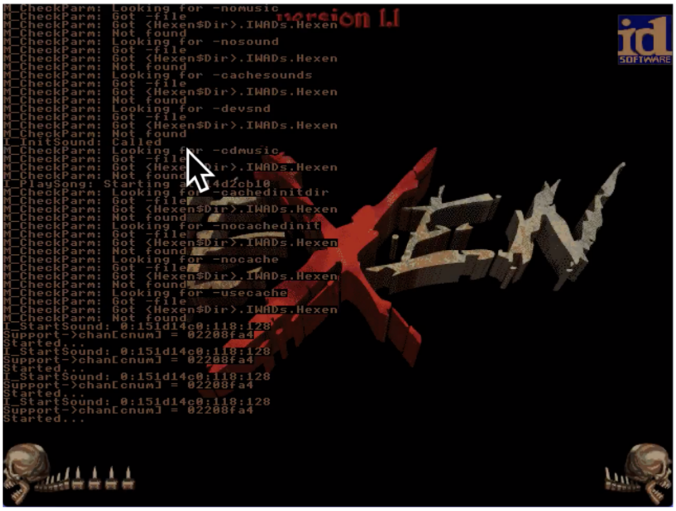
Hexen (2024)

Converting to 32bit (1)

The first thing to do do was to convert was the game itself.
32bit conversions are not very complicated if the code is written in C - you just add `-apcs 3/32` to the compiler, and tell it to link against the 32bit version of the C library.
However, there are a few assembly files which are faster than compiler would produce. Fortunately, the difference are pretty minor between 26bit and 32bit for application code. You need to not restore flags on return from functions.
Yup, that's just removing an up-tick or an `s` from the instructions. Not especially hard at all. I took the opportunity to make the code clearer by adding macros for the entry and exit sequences - the same process as I'd done for the 32 bit RISC OS, many years ago.
The assembler code took a couple of hours to convert safely and get it all to link together. I could then try the game out.
Which, of course, didn't work.
I was testing things under RISC OS Pyromaniac and there's a few differences with how it works and how RISC OS Classic works. Some of which you don't find out until they go wrong.
Much of the game was dependant on having the support module to provide key input and the screen bank changing. I hadn't converted the module to be 32bit yet, so I changed all the code that relied on the module to just ignore it if it wasn't there. That way I could get the game starting, even if I didn't have any input.
This was largely successful and the game got as far as trying to put up the startup screen. At which point it crashed because it was trying to write directly to screen memory.
RISC OS Pyromaniac doesn't have screen memory. It expects you to only access the display through the OS interfaces, not direct memory access.
Hexen's video system was updated so that if the requested mode existed and it could access the screen, it would do so. If either of those weren't possible, it will switch to using sprite rendering.

Hexen (2024)

Converting to 32bit (2)



```
M_CheckParm: Looking for -nomusic
M_CheckParm: Got -file
M_CheckParm: Got <Hexen$Dir>.IWADs.Hexen
M_CheckParm: Not found
M_CheckParm: Looking for -nosound
M_CheckParm: Got -file
M_CheckParm: Got <Hexen$Dir>.IWADs.Hexen
M_CheckParm: Not found
M_CheckParm: Looking for -cachesounds
M_CheckParm: Got -file
M_CheckParm: Got <Hexen$Dir>.IWADs.Hexen
M_CheckParm: Not found
M_CheckParm: Looking for -devsnd
M_CheckParm: Got -file
M_CheckParm: Got <Hexen$Dir>.IWADs.Hexen
M_CheckParm: Not found
I_InitSound: Called
M_CheckParm: Looking for -cdmusic
M_CheckParm: Got -file
M_CheckParm: Got <Hexen$Dir>.IWADs.Hexen
M_CheckParm: Not found
I_PlaySong: Starting at 14d2cb10
M_CheckParm: Looking for -cachedinitdir
M_CheckParm: Got -file
M_CheckParm: Got <Hexen$Dir>.IWADs.Hexen
M_CheckParm: Not found
M_CheckParm: Looking for -nocachedinit
M_CheckParm: Got -file
M_CheckParm: Got <Hexen$Dir>.IWADs.Hexen
M_CheckParm: Not found
M_CheckParm: Looking for -nocache
M_CheckParm: Got -file
M_CheckParm: Got <Hexen$Dir>.IWADs.Hexen
M_CheckParm: Not found
M_CheckParm: Looking for -usecache
M_CheckParm: Got -file
M_CheckParm: Got <Hexen$Dir>.IWADs.Hexen
M_CheckParm: Not found
I_StartSound: 0:151d14c0:118:128
Support->chan[cnum] = 02208fa4
Started...
I_StartSound: 0:151d14c0:118:128
Support->chan[cnum] = 02208fa4
Started...
I_StartSound: 0:151d14c0:118:128
Support->chan[cnum] = 02208fa4
Started...
I_StartSound: 0:151d14c0:118:128
Support->chan[cnum] = 02208fa4
Started...
```

Hexen (2024)

Converting to 32bit (2)

Using sprite rendering like this is going to be slower than directly accessing the screen, but it allows a lot more flexibility about the type of modes we can use. We can use a large mode but with the game running at a low resolution, and having the sprite scaled up when rendering.

Why does that matter? Well, some of the modern hardware doesn't support the lower resolution modes. If the user wanted to use 320x200 - the original resolution - they wouldn't have been able to. But by using the sprite and rendering that way, it would still work. It would be slower, but it gives a lot more flexibility.

This image shows the early startup screen, with a bunch of debug enabled.

A day or so later, though, I had a game that ran. Not well, because there was debug everywhere and it wasn't fast. From running the game to the menu appearing took about 70 seconds - that's the cost of having lots of the Pyromaniac instrumentation enabled so that I know what had gone wrong.

The demo started up as well, and we get a shocking 2 and a half seconds per frame. That's seconds per frame, not frames per second. That's ok because it was actually running.

Spin forward a week or so and I had converted the FrontEnd and support module so we have sound and input, and the game starts up pretty well. Pyromaniac gets about 3.5 frames per second now, and it's running well on RPCEmu.

Hexen (2024)

Video limitations

- Low resolution modes might not be available - using sprites allows them to work.
- Low colour depths (like 256 colours) might not be available - using ColourTrans allows them to work.
- VSync events may not work reliably - using the OS_Byte 19 to wait for the VSync allows that.
- Multiple screen banks might not be available - we can use single buffering.
- Higher resolutions than before are available - Hexen supports up to 8192x8192.
- Widescreen modes (16:9 or 16:10) are possible - Hexen moves weapons around to make them better fit these modes.

Hexen (2024)

Video limitations

The original Hexen that I released in 2000 only supported up to 1280x1280, but I knew that most modern systems would support higher resolutions. I decided to double the limits - that would make it support 2560x2560, which would include the common 1920x1080 resolution. Internally most of the rendering works fine as you increase the resolution but some parts need special attention.

The most recent release supports up to 8192x8192, which should be enough for a while!

And with it working on RISC OS Pyromaniac and RPCEmu, I needed to try more things, so I send a copy to Chris to try out - he had a PineBook Pro.

It immediately failed.

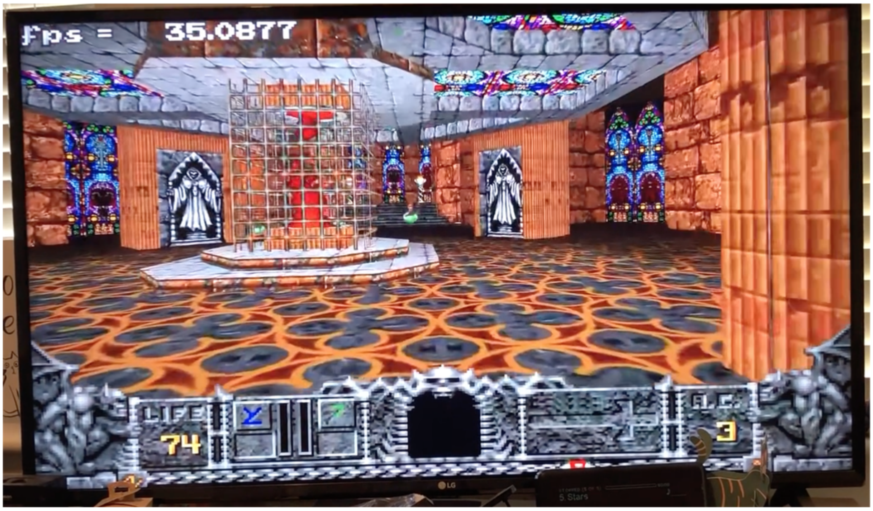
Why? There is only one screen mode resolution available on the PineBook - 1920x1080 - and for all my expectations about how to support different resolutions, I'd not considered that you'd only have one resolution. The startup screen doesn't support that. So the startup had to be reworked to allow it to run in other screen modes, and with that we had a game that runs on the PineBook.

I dropped a copy to Andrew, who was very surprised at this out-of-the-blue message! There were a few iterations - and Andrew sent me a Rock 4C+ board to try it on, with all the bits I needed.

Many thanks to him for the little board!

Hexen (2024)

Rock 4 C+ at 1920x1080



Hexen (2024)

Rock 4 C+ at 1920x1080

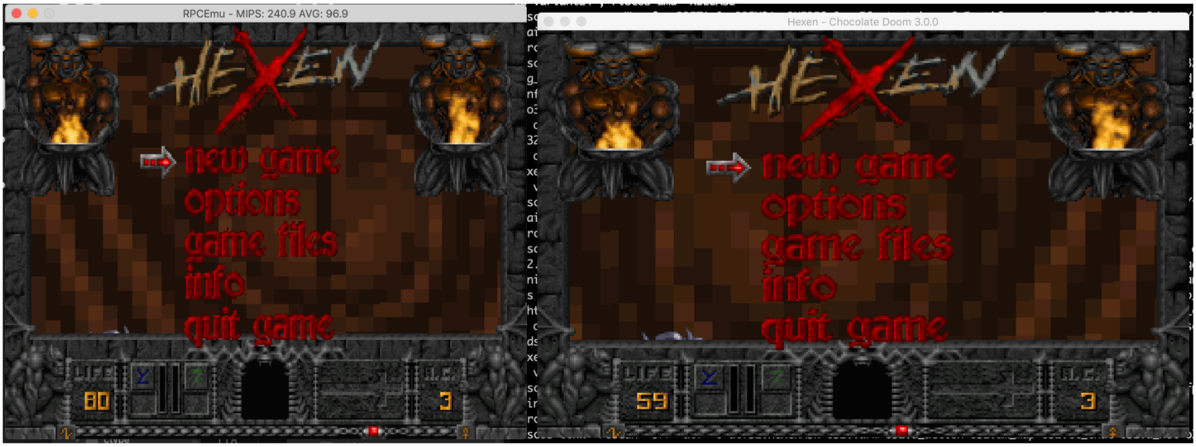
Here we see Hexen running on the Rock 4C+ at 1920x1080 on my telly. It's looking pretty good, and it's got a solid 35 frames per second.

Meanwhile I'm looking into the lighting problems. The lighting in Doom had been a problem for a while if you remember. The 'brighten' option that I had added wasn't very effective. With the larger resolutions, the problem was even worse.

After much searching around I found two bugs in how the lighting was determined, which had been there since the original conversion.

Hexen (2024)

Fixing the lighting



Hexen (2024)

Fixing the lighting

And with that, the colours matched up whatever resolution you were using.
Here you can see RISC OS Hexen on the left, and Chocolate Doom running Hexen for reference on the right.
The door that we're looking at has the same brightness of brown.
Andrew also asked if I could add MP3 music to the game, as CD music wasn't really a thing any more and nobody had MIDI hardware. Fortunately, the code to do that was actually really simple. It probably took longer to add the launcher option to select the MP3 choice!

Hexen (2024)

Automated testing

- Builds the whole release on macOS (Front end, game * 5, module, ACC tool) in about 24 seconds.
- Builds on Linux (two sets of the above release) when pushed to source control in about 2 minutes.
- Tests on Linux by sending to the build service with a WAD file and running like:

```
WimpSlot 4800K
echo Run Hexen with the WAD file
/<Hexen$Dir>.Bin.Hexen8HR -file <Here$Dir>.Hexen/WAD -x 320 -y 256
  -startmessages -fastdemostart -fpsmeter -fpsticker -warp 1 -timer 10s
echo Exited ok
```

- Tests through the build service in different combinations take around 7 minutes.
- Emails are sent to notify me whether it worked or not.

Hexen (2024)

Automated testing

During the development I was making changes and checking that they were working manually, but also had automated builds happening on my server. Every time I pushed changes, they were built to make sure that I hadn't forgotten to commit anything - building both the debug and release versions. As the game started working well, I added the ability to automatically run the game code as well.

After the game was built on Linux it would be sent over to the RISC OS Build service with a set of templated parameters to run the game, something like the commands you see here.

That starts the game automatically and runs it for 10 seconds, then exits. So if the game were to crash, or access bad memory during that time, it would fail the tests and I'd know not to release it.

Then after Andrew complained that I'd left some debug enabled, I added some extra checks to make sure that the release didn't write out any debug either.

These checks mean that - at the barest minimum - the game runs. Which is useful as it wastes everyone's time if I send out a version that doesn't even start.

Hexen was released at the 2024 South West show, and an update went out in early April.

Hexen Demo!

Hexen Demo!

Ok, let's see if we can do a demo of Hexen.
<launch RPCEmu, switch the screen sharing and run a short demo of the game>
<restore the screen sharing to the slides>
Before I move on to Heretic, does anyone have any questions about Hexen and the development process?

8. Heretic (2024)

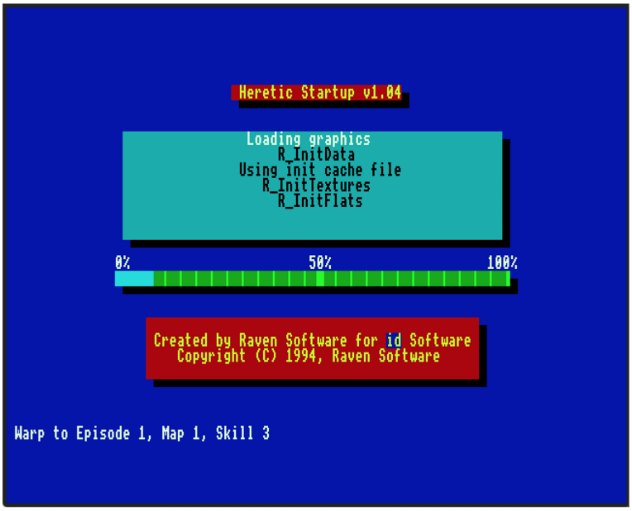
February 2024 - April 2024

7. Heretic (2024)

February 2024 - April 2024

Heretic (2024)

Converting to 32bit



Heretic Startup v1.04

Loading graphics
R_InitData
Using init cache file
R_InitTextures
R_InitFlats

Created by Raven Software for id Software
Copyright (C) 1994, Raven Software

Warp to Episode 1, Map 1, Skill 3

Heretic (2024)

Converting to 32bit

Heretic was a little easier to get started with - everything that I'd learnt from Hexen could be largely applied directly to Heretic. It only took about 4 hours to get a working version that had all the new mode selection code to handle modern machines.

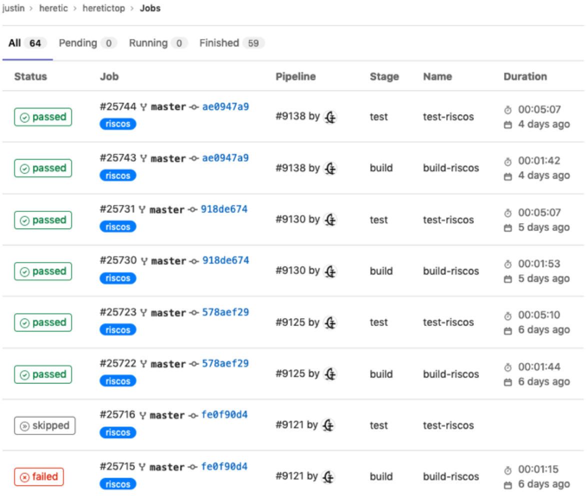
I spent a lot of time trying to get the startup screen to work sensibly - it had assumed that an 80x25 text screen mode in 16 colours would be available, like mode 14. Of course, that's not true of modern hardware - few systems have 16 colours, and few have such a low resolution.

It's now working fine in whatever mode you have available, but it starts up so fast that you almost never even see the startup screen. I feel good about the work I did, but it's a little sad that it doesn't get seen!

Backporting the sound, lighting fixes and the MP3 music playback from Hexen was really easy, and at the same time the code for the launchers was made much more similar to one another. That meant that making improvements to one game could be passed back to the other.

Heretic (2024)

More testing (1)



justin › heretic › heretictop › **Jobs**

All 64 · Pending 0 · Running 0 · Finished 59

Status	Job	Pipeline	Stage	Name	Duration
✓ passed	#25744 master ae0947a9 riscos	#9138 by	test	test-riscos	00:05:07 4 days ago
✓ passed	#25743 master ae0947a9 riscos	#9138 by	build	build-riscos	00:01:42 4 days ago
✓ passed	#25731 master 918de674 riscos	#9130 by	test	test-riscos	00:05:07 5 days ago
✓ passed	#25730 master 918de674 riscos	#9130 by	build	build-riscos	00:01:53 5 days ago
✓ passed	#25723 master 578aef29 riscos	#9125 by	test	test-riscos	00:05:10 6 days ago
✓ passed	#25722 master 578aef29 riscos	#9125 by	build	build-riscos	00:01:44 6 days ago
skipped	#25716 master fe0f90d4 riscos	#9121 by	test	test-riscos	
✗ failed	#25715 master fe0f90d4 riscos	#9121 by	build	build-riscos	00:01:15 6 days ago

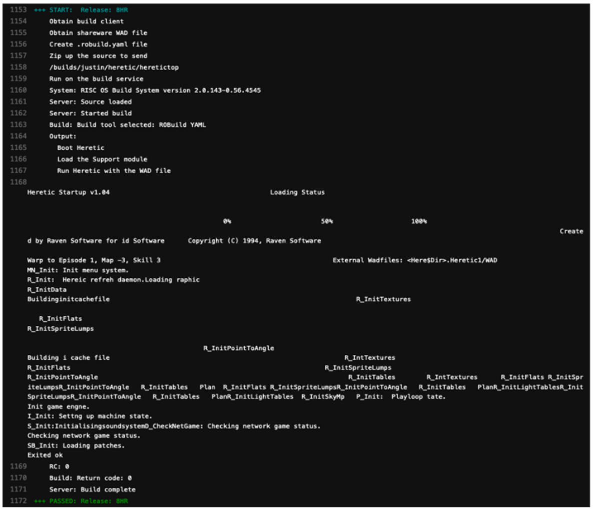
Heretic (2024)

More testing (1)

Like Hexen, the game is tested automatically on every push to source control.
It's not very exciting to see that this works, but it does. Each push gets a build job and then a test job. It takes about 7 minutes, and you rarely even look at the output.

Heretic (2024)

More testing (2)



```
+++ START:  Release: 8HR
    Obtain build client
    Obtain shareware WAD file
    Create .robuild.yaml file
    Zip up the source to send
    /builds/justin/heretic/heretictop
    Run on the build service
    System: RISC OS Build System version 2.0.143-0.56.4545
    Server: Source loaded
    Server: Started build
    Build: Build tool selected: ROBuild YAML
    Output:
      Boot Heretic
      Load the Support module
      Run Heretic with the WAD file

 Heretic Startup v1.04                              Loading Status


                                          0%                50%               100%
                                                                                                Create
 d by Raven Software for id Software     Copyright (C) 1994, Raven Software

 Warp to Episode 1, Map -3, Skill 3                                 External Wadfiles: <Here$Dir>.Heretic1/WAD
 MN_Init: Init menu system.
 R_Init:  Hereic refreh daemon.Loading raphic
 R_InitData
 Buildinginitcachefile                                                  R_InitTextures

   R_InitFlats
 R_InitSpriteLumps

                                    R_InitPointToAngle
 Building i cache file                                         R_IntTextures
 R_InitFlats                                               R_InitSpriteLumps
 R_InitPointToAngle                                            R_InitTables        R_IntTextures      R_InitFlats R_InitSpr
 iteLumpsR_InitPointToAngle   R_InitTables   Plan  R_InitFlats R_InitSpriteLumpsR_InitPointToAngle   R_InitTables   PlanR_InitLightTablesR_Init
 SpriteLumpsR_InitPointToAngle   R_InitTables   PlanR_InitLightTables  R_InitSkyMp   P_Init:  Playloop tate.
 Init game engne.
 I_Init: Settng up machine state.
 S_Init:InitialisingsoundsystemD_CheckNetGame: Checking network game status.
 Checking network game status.
 SB_Init: Loading patches.
 Exited ok
    RC: 0
    Build: Return code: 0
    Server: Build complete
+++ PASSED: Release: 8HR
```

Heretic (2024)

More testing (2)

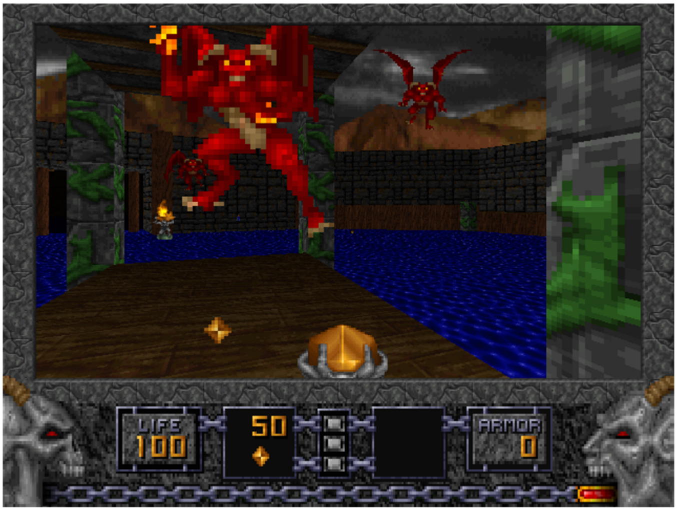
When you do look at the test runs, you see things that look like this. This it looks like when it works...

- There's a bit of setup where we obtain tools and build archives.
- Then we send it to the build service, and it starts to run.
- We can see the startup text being printed.
- And finally we exit successfully if nothing went wrong.

Obviously we don't test the gameplay, but we at least know the game starts.

Heretic (2024)

Release



Heretic (2024)

Release

Andrew asked me, towards the end of April, whether I was happy with the release going out, and I had to do a quick last minute fix. After I pushed the change I got an email telling me that it'd failed its tests!
It turned out I'd left another bit of debug turned on and that caused the failure. There are many reasons to have the tests - but knowing that you've not made stupid mistakes is one of the best!
Heretic was finally released at the Wakefield show at the end of April.

Heretic Demo!

Heretic Demo!

Time for a little demo of Heretic as well.
<launch RPCEmu, switch the screen sharing and run a short demo of the game>
<restore the screen sharing to the slides>

7. Conclusion

7. Conclusion

Let's wrap this up.

Conclusion

How did it go?

- It's fun doing the old games.
- Testing with RISC OS Pyromaniac has been very useful.
- I've had a chance to play with modern hardware.

Where to now?

- Not sure - it's beginning to feel a lot like work.
- Probably get Doom+ finished at some point.

Conclusion

How did it go?

It's been fun to get back to doing the old games - and using RISC OS Pyromaniac to test things, in the way that it was meant to, was really good.
I've have had a chance to play with some modern hardware too, which was interesting.
Mostly it's been pretty relaxing - with the exception of the maths and confusing code that I had to understand again.
Am I going to be doing some more? Maybe. I've got a working version of Doom+ again, but it's all beginning to feel a lot like work, so that might take a little while to get out properly.

Conclusion

Finally...

> *How does it feel to be nearly 50 and be revisiting the games you did over 20 years ago?*

Conclusion

Finally...

My fiancee asked me... "How does it feel to be nearly 50 and be revisiting the games you did over 20 years ago?"
...

Conclusion

Finally...

> *How does it feel to be nearly 50 and be revisiting the games you did over 20 years ago?*

- A bit old.
- Some things have faded through disuse.
- Definitely don't have the same energy from back then.
- But in others, experience really helps.
- Things were still very familiar.
- Still fun to do.

...

Yeah, it makes me feel old.

Some things were much harder, like the maths has faded through disuse. And I don't have the same energy that I had back then.

But it's also kinda nice to see how quickly I returned to other things. Some of the changes were a lot easier now than they ever would have been back in the day - experience really does pay off.

At the end of the day it was all very familiar, but I was able to see some of it with new eyes.

And it was still fun.

Questions

I'll take any questions that people have.
Slides and Info: **http://presentation.riscos.online/doom-games/**

Questions

I hope that has been interesting. I've put some resources up on my presentation site about the development as well.
I'll take any questions that people have now.