Quake III Arena - GtkRadiant [~] 07/10/2023

Introduction

In this particular lesson, I'm going to cover:
- [GtkRadiant]: A level editor that is used to make game levels for various [idTech] based games
- [<I3FG20K>'s Shopping Maul]: A mapping website that I used to manage on [PlanetQuake.com]

Index	Name	Title	Build	Age
0	bfgdm1	Crossfire	05/28/2000 0907	23y 1m 12d 4h 23m 23s
1	bfgdm2	Breakthru	08/20/2000 1442	22y 10m 18d 22h 48m 23s
2	bfgdm3	Space Station 1138 (Original)	04/06/2001 1801	22y 3m 2d 19h 29m 23s
3	bfgdm4	Suspended Animation	05/04/2001 1837	22y 2m 5d 18h 53m 23s
4	bfgdm3a	Space Station 1138 (Color)	05/05/2001 1543	22y 2m 4d 21h 47m 23s
5	20kdm1	Tempered Graveyard	07/20/2001 2352	21y 11m 19d 13h 38m 23s
6	hellra3map1	Dude, You Can Go To Hell	07/26/2001 1513	21y 11m 13d 22h 17m 23s
7	20kdm2	Return to Castle: Quake	02/01/2002 2316	21y 5m 6d 14h 14m 23s
8	20kctf1	Out of My Head	03/08/2003 0512	20y 4m 2d 8h 18m 23s
9	20kdm3	Insane Products	09/09/2005 0017	17y 9m 30d 13h 13m 23s

Script

Here's the script for the information table up above.

```
Enum MapName
{
    _bfgdm1
    _bfgdm2
    _bfgdm3
    _bfgdm4
    _bfgdm3a
    _20kdm1
    _hellra3map1
    _20kdm2
    _20kctf1
    _20kdm3
}

Class MapFile
{
    [UInt32]         $Index
    [String]          $Name
    [String]         $Title
    Hidden [DateTime] $Date
    [String]         $Build
    [TimeSpan]        $Time
    [String]          $Age
    MapFile([UInt32]$Index,[String]$Name)
    {
        $This.Index = [UInt32][MapName]::$Name
        $This.Name  = $Name.TrimStart("_")
    }
    Set([String]$Title,[String]$Date)
    {
        $This.Title = $Title
        $This.Date  = $Date
        $This.Build = $This.Date.ToString("MM/dd/yyyy HHmm")
        $This.Time  = [TimeSpan]([DateTime]::Now-$This.Date)
        $This.GetAge()
    }
    GetAge()
```

```
    {
        # Actual floating point value of a [year] in [days]
        $Year     = 365.2425

        # Actual floating point value of a [month] in [days]
        $Month    = 30.436875

        $Years    = $Null
        $Months   = $Null
        $Days     = $Null

        # Year -> Returns remainder
        $RemYear  = $This.Time.Days % $Year

        # Year -> Removes remainder, then divides
        $Years    = ($This.Time.Days-$RemYear)/$Year

        # Month -> Returns remainder
        $RemMonth = $RemYear % $Month

        If ($RemMonth -match "NaN")
        {
            $Months = 0
        }
        Else
        {
            $Months = ($RemYear-$RemMonth)/$Month
            $Days   = [Math]::Round(($RemYear-($Months*$Month)))
        }

        $This.Age = "{0}y {1}m {2}d {3}h {4}m {5}s" -f $Years,
                                                       $Months,
                                                       $Days,
                                                       $This.Time.Hours,
                                                       $This.Time.Minutes,
                                                       $This.Time.Seconds
    }
    [String] ToString()
    {
        Return "{0}/{1}" -f $This.Name, $This.Title
    }
}

Class MapList
{
    [Object] $Output
    MapList()
    {
        $This.Refresh()
    }
    Clear()
    {
        $This.Output = @( )
    }
    [Object] MapFile([UInt32]$Index,[String]$Name)
    {
        Return [MapFile]::New($Index,$Name)
    }
    [Object] New([String]$Name)
    {
        Return $This.MapFile($This.Output.Count,$Name)
    }
    Refresh()
    {
        $This.Clear()

        ForEach ($Name in [System.Enum]::GetNames([MapName]))
        {
            $Item = $This.New($Name)
            Switch ($Item.Name)
            {
                bfgdm1      { $Item.Set("Crossfire",                    "05/28/2000 09:07") }
```

```
                bfgdm2      { $Item.Set("Breakthru",                  "08/20/2000 14:42") }
                bfgdm3      { $Item.Set("Space Station 1138 (Original)", "04/06/2001 18:01") }
                bfgdm4      { $Item.Set("Suspended Animation",        "05/04/2001 18:37") }
                bfgdm3a     { $Item.Set("Space Station 1138 (Color)",    "05/05/2001 15:43") }
                20kdm1      { $Item.Set("Tempered Graveyard",         "07/20/2001 23:52") }
                hellra3map1 { $Item.Set("Dude, You Can Go To Hell",      "07/26/2001 15:13") }
                20kdm2      { $Item.Set("Return to Castle: Quake",       "02/01/2002 23:16") }
                20kctf1     { $Item.Set("Out of My Head",             "03/08/2003 05:12") }
                20kdm3      { $Item.Set("Insane Products",            "09/09/2005 00:17") }
            }

            $This.Output += $Item
        }
    }
}
```

Script GtkRadiant

[GtkRadiant] can be found at this website: [https://icculus.org/gtkradiant/]
[GtkRadiant] is the level editor I used to make all of those maps at or about (20) years ago.

Prior to attending [Capital Region Career and Technical School] to study [Microsoft System Administration], and [Cisco Network Academy Program] under instructor [David J. Patzarian] in [September 2001], I made the first (7) maps there...

Index	Name	Title	Build	Age
0	bfgdm1	Crossfire	05/28/2000 0907	23y 1m 12d 4h 23m 23s
1	bfgdm2	Breakthru	08/20/2000 1442	22y 10m 18d 22h 48m 23s
2	bfgdm3	Space Station 1138 (Original)	04/06/2001 1801	22y 3m 2d 19h 29m 23s
3	bfgdm4	Suspended Animation	05/04/2001 1837	22y 2m 5d 18h 53m 23s
4	bfgdm3a	Space Station 1138 (Color)	05/05/2001 1543	22y 2m 4d 21h 47m 23s
5	20kdm1	Tempered Graveyard	07/20/2001 2352	21y 11m 19d 13h 38m 23s
6	hellra3map1	Dude, You Can Go To Hell	07/26/2001 1513	21y 11m 13d 22h 17m 23s

[GtkRadiant] had a bit of a learning curve to it, however I'm going to cover how to use it to make some of these maps which I will demonstrate in a multiple part series.

Many newer games have much more complex level editors, but the basics are generally the same.

The key is that you start with a void, where nothing exists...
...and then, you have to sculpt out the world or level using a variety of strategies and techniques, in order for the level to compile and allow the hardware to generate lightmaps and things of that nature.

GtkRadiant Map Demonstration

In order to understand how to make the levels, it is worth studying maps that are currently playable, and determining what maps work, and what maps don't.

Fact of the matter is, there are a LOT of really well built maps out there that probably took a lot of time to build, had a lot of thought put into them, and they may even LOOK really awesome.

But, even still, the end result is whether or not the map plays well.

Every single factor mentioned above, can all fall victim to whether or not the map flows and the [gameplay] allows the map to be:

fun	balanced	competitive	replayable

[Replayability] requires a well thought out perspective on the [gameplay].

If the [gameplay] is not good...?
Then really, [all of the time invested] turns out to be a bit of a [waste].

That's important.
It's not unlike listening to stock investors saying "buy stocks in THIS, not THAT..."

Here's a few videos, sort of warming up to [Quake III Arena] again after a long time of inactivity...

Name	Date	Resource
2023_0708-(Q3A Practice)	07/08/2023	https://youtu.be/RCkI2OFtCB4
2023_0710-(Q3A Practice (Custom Maps))	07/10/2023	https://youtu.be/bQ46Pvp0tOo
2023_0710-(Q3A Practice (Custom Maps))	07/10/2023	https://youtu.be/_siuaph1_vc

Map Demonstration

[bfgdm1/Crossfire]

In [bfgdm1], the map is symmetrical in shape, but the items are laid out in a way where either side of the map a player spawns on, they are able to ascertain:

weapon	ammo	health	armor

The [gameplay] does suffer from the [geometry] of the map. For instance, it is [really difficult] to deal [splash damage] from [rockets] on the floor in the central area.

Also, the bots tend to have difficulty dealing with [curved/beveled surfaces], so their navigation does suffer. Sometimes, a map has to be redesigned because bots have difficulty in navigating it, though that is not necessarily the case in this particular map.

Generally, on [Nightmare] difficulty in [Quake III Arena]... bots have [really good aim].
Typically, they have [strengths] with [certain weapons] over [others].

[Hunter] is incredibly nightmarish when using the [lightning gun], and it is extremely difficult for any human player to match or exceed her accuracy with it. She is also pretty good with [railgun], and [machine gun], and is generally proficient with all [other weapons].

As for the match in the video, [Hunter] maintained the lead for a while, but I was able to get an edge by using the [geometry] of the map to my advantage. This is a tactic that is generally required...
...especially if the opponent has an edge in [accuracy].

[Movement] is critical to [gameplay], and if the items are spaced out too much for a losing player, they may not be able to [regain the advantage], and thus, this is something that needs to be considered when placing items throughout the map, AKA [item placement].

[Timing] is also critical to [gameplay], because if an item is worth picking up, but it cannot be accounted for in a cycle, then it means that it COULD be risky to get that item.

As for this particular level, the [lighting] is rather lackluster.

Though, I was just turning (15) years old when I made it, and [YouTube] didn't exist.
I couldn't just do a quick [Google] search for lessons on how to build maps for [Q3A], or whatever, and find thousands of videos of people doing just that.

At the time, this map was pretty taxing on the system I was using. Though, the map runs pretty well today.

[bfgdm2/Breakthru]

This is actually a map that I originally made for [Quakeworld/Quake].
That's right, this is a [Quake] map that I made, and converted to [Quake III Arena].

I cannot find [bfgdm2a], which is a remake of this map that has better lighting, and more space in it.
This is actually a pretty decent tournament (1v1) map.

The items are placed in such a way, that either player has a way
to retreat from a fight and stock up on items, such as:

weapons	ammunition	health	armor

It does suffer from feeling claustrophobic in many places, however that was worked out when I rebuilt this level multiple times afterward.

Note: [20kdm1] wound up being very large version of this map, but with a lot of additional rooms and stuff and is best suited for [Clan Arena].

The teleporters work in a [3-way] configuration. The surface allows players to look into the portal where they're going before they step into it. This can provide an edge over opponents and give them an opportunity to telefrag the opponent, or simply to watch what they're doing.

Bots tend to whiz through them, and they rarely (if ever) collect the [red armor].

Generally speaking, one thing to keep in mind is the idea of layering additional places for the players to go, where they can still be involved with the areas above, below, inside, or outside of the gameplay area.

[Team Arena/2000] did this a lot when it came out with this newer method for building outdoor terrain...

I won't expand on that concept right now, but the team at [ID Software] developed a way to make terrain using a wireframe approach, not unlike how curves work in [GtkRadiant]. By using that approach, it allowed the levels to feel more [expanse] and less claustrophobic.

This level provides a lot of tense moments.

[bfgdm2/Breakthru]

[bfgdm2/Space Station 1138]

This map uses less than (100) brushes, and was built for a competition. I didn't think to use the colored lighting like I did in [bfgdm3a], but in my opinion, the [gameplay] in this level is pretty top notch.

There are plenty of ways to stay ahead of the enemy if the player is in the lead and in control... as there are ways to catch back up if the player is losing.

Arrangement of the level is a lot more [vertical], and the jump pad allows a player to go from bottom-to-top as quickly as top-to-bottom. And, that's important because there are plenty of items stacked in this map.

[Railgun] is great for great concise shots, and can cause knockback into the void.
[Shotgun] is not the best weapon in this arena, though in close proximity it is better than a machinegun.
[Plasma gun] is able to cover a lot of ground, and it isn't as draconian as the [lightning gun] would be in the same position...
...and the [rocket launcher] is the perfect weapon to deal damage and throw people around if needed.

To this day, I think this map is underrated, as it plays better than it looks.

The tower allows a lot of items to be accessible.
I could've probably thrown in another one somewhere else...
PERHAPS, that may be a core focus with [GtkRadiant], to rebuild this particular map and make improvements.

The bots have a hard time making strategic movements throughout the map.

[bfgdm2/Space Station 1138]

[bfgdm2/Suspended Animation]

This map is basically a lot like [Dead Simple] from [Doom].

It's a square. The players can run around and collect the items, or run into the middle to take shortcuts to other items across the map. There's also a quad damage in the middle of the map that is only accessible with a well-placed rocket jump.

Despite the age and simplicity of this map, the [aesthetics] and [geometry] in this map are pretty slick, as it still feels rather fresh.

This isn't really great for tournament (1v1), but it isn't exactly off the table.

There are a lot of improvements that could be made, but all things considered, the [aesthetics], [lighting], [geometry], and [item placement] were all well-executed. Though, the lighting is a tad dark in some places.

Fighting bots on nightmare difficulty won't be an easy challenge, unless a player is using draconian spawn strategies where they're (aiming + prefiring) at potential enemy spawn points. This map is still a bit too big to cover all potential spawn points from any single perspective.

[bfgdm2/Suspended Animation]

[20kdm1/Tempered Graveyard]

This map is pretty big, it is part of [hellra3map1] which I did not cover in the video, but it is effectively the fourth version of [bfgdm2/Breakthru].

It is better suited for [clan arena] and [free for all], much less [1v1].

Some of the key aspects of [1v1] maps are that they're typically [smaller], and can take a player anywhere from (10-20) seconds to get to any other point in the map, whereas a [larger] map may take (15-60).

If a map takes [longer] than a [minute] to get to any other point...?
That's probably a [Team Arena] map.

Anyway, this map has a large atrium area that acts as the central hub, not unlike [q3dm6/Campgrounds], though it is a bit bigger. The [aesthetics] and [geometry] are pretty slick, considering the evolution of the maps I had made since [bfgdm1]. The lighting is generally decent, but the very top of the map has too much brightness from the lightbox texture.

The lighting in [Quake III Arena] comes from (2) sources.
1) [light entities]
2) [textures/shaders]

[Light entities] are [entities] in the editor that aren't visible when the map compiles, though it's effects certainly are. [Textures/shaders] are basically scriptlets that contain properties that relate to lighting, animation, or opacity/transparency... but they're definitely not limited to just that.

In either case, the lighting is rather static and is rendered when the map is compiled.
However, in [Doom 3], the lighting was able to cast shadows and could move around, so it was far more dynamic. In some cases that actually worked against the enjoyment factor of the game, but I think it was STILL able to provide a strong [paradigm shift] in the way [lighting] was rendered within a game.

In [Q3A], the light entities have a luminosity property (*I believe*).
The textures that emit light ALSO have a luminosity property, but that is controlled by the shader.

When selecting textures in the [editor/GtkRadiant], it is impossible to know whether the [luminosity] of a particular texture is [too bright] or [too dark] without first compiling the map, and this requires having to compile the map for each iteration to figure that part out.

Newer games don't have this problem, as they can take advantage of hardware to instantly render lighting within the editor, and [Doom (2016)] is great case of that.

Still, we're talking (15) *full-cycle*, (365/366) *earth day*, [years] prior to [Doom (2016)], when I made this map. The system I used to build this map took a REALLY long time to compile it.

- [0.866 Ghz Pentium 3]
- [512MB memory]
- [NVidia GeForce 2 GTS]

Yeh, processor speed used to be measured in megahertz, not gigahertz.
Suffice to say, these metrics mean that I sorta had to mull over waiting another (1-2) days to try it again, or to submit it, and be done with it.

The map is better for [Clan Arena] than it is for deathmatch or et cetera.
That's why the item placement feels like it could've been more polished.

[20kdm2/Return to Castle: Quake]

This is not a level that I selected in the beginning of the video, but that's ok.

I made this level after the release of [Return to Castle Wolfenstein] on [11/20/2001].

I wanted to utilize some of the brick-based castle textures to build something that felt modular and tight, without feeling claustrophobic, and this was the result of that.

This map is pretty good for [tournament] based play, and even [free-for-all].
If I were to rebuild this level today, there are many other ideas or improvements I would make.

The bots are able to navigate the map relatively well.
Exchanging and dodging rockets and railgun shots is pretty fun.

Some of the geometry forces the players to have to use it correctly, in order to reach certain items.

There is a missing texture in the map that can be fixed by downloading the [evil7] texture set which
is described in the readme file.

[20kctf1/Out Of My Head]

So, this map is a symmetrical map that has (3) separate areas in the middle of the map, that connect each
side of each teams base.

This has a lot of standard textures in it, it also has the capability of being played in [free for all].

There's plenty of items scattered all throughout the map, and there's ways to trick jump certain spots
to get from point to point. There's some fog in the map in the lower areas, and there's a bunch of
jump pads that throw players around. There's a couple of teleporters that connect the farthest points
of the (2) outside areas in the middle of the map, and the actual middle area in the center of the map
is indoors, and has (2) tiers.

The bots seem to play this map pretty well, and that's about all that needs to be said in that regard.

The map itself, uses a style that I put together by playing a lot of [Threewave CTF] and [Team Arena].

[20kdm3/Insane Products]

So, this map is really, really [ambitious]. It has a LOT of different routes to collect items, it also
has basically [every weapon] in the game in it, it has a [secret area] with a powerup in it, it has a lot
of [vertical gameplay], [jump pads], [stairwells], [geometry], [aesthetics], [textures], [shaders], [trick jumps],
and overall [flow].

Though, all that said, it lacks main traffic trunks.

Meaning, I can expect that a player in [tournament mode] would go [path A] or [path B].
I don't really know if that's a bad thing necessarily, because the number of ways to get from one end
of the map to the other are rather staggering.

That was the idea behind this map.
I also had a bit of an issue getting the poly count to remain low, and I had to make a lot of design
changes in order to finish it.

[Some] of the ideas I implemented don't make a whole lot of sense...
Other ideas I implemented are rather [stylistic] and [well done].

All in all, I think that having other people play test the map would've yielded better results.

Bots don't particularly navigate the map all that well, and during development, I distinctly remember
that the map would leak all the time, and sometimes that'll be because [GtkRadiant] can sometimes generate
a bad brush. I think I lost the original map file, and then had to decompile it in order to rebuild it.

I may cover the concept of decompiling (*.*bsp/binary space partition*) files, to import in [GtkRadiant].

Aesthetically, this map is really pleasing.
The geometry is pretty good too. Lighting, also rather good.
Gameplay, isn't bad at all.

All that said, something about it feels unfinished and could benefit from more polish.

<I3FG20K>'s Shopping Maul

[<I3FG20K>'s Shopping Maul] is the website that I made that was hosted on PlanetQuake a very long time ago.
I can't remember when they actually shut down aspects of [PlanetQuake] and other "*Planet*" sites, or when
they merged with [IGN].

What I can say, is that I'm lucky enough to still have some of the maps featured on [..::LvL],
and I'm also lucky enough to have snapshots of the site on the [Wayback Machine]:

[http://web.archive.org/web/20010716191646/http://www.planetquake.com/bfg20k/]

There's a lot of snapshots of the site there.
To be honest, I used it as a blog of sorts long before [Facebook], [MySpace], or [YouTube] even existed.

The point of all of this, is to talk about [GtkRadiant], in order to talk about: [Game Design].
[Game Design] involves a lot of things, particularly:
- [computer science]
- [mathematics]
- [programming]
- [performance]
- [hardware]
- [philosophy]
- [instructional design]
- [graphic design]
- [many other things]

So, how does this apply to the every day person out there in the world...?
It might not, unless you play a GAME of some sort.

A lot of people do this. They play [games], of some sort, on a [computer].

```
                                                                                                  ____________________________/
\_________________________________________________________________________________________/ <I3FG20K>'s Shopping Maul
  [What is a computer] /-------------------------------------------------------------------------------------------------------\
/----------------------
```

Some people might laugh at me and say:

[Guy]: YEH RIGHT DUDE, NOBODY PLAYS GAMES ON A COMPUTER~!
[Me] : Oh, ok.
       Do you have an [Xbox]...?
[Guy]: Uh, yeh.
[Me] : Uh, that's a [computer].
[Guy]: No it isn't, it's a [console].
[Me] : A [console] is a [computer].
[Guy]: No it isn't, it's a [console].
[Me] : Trust me when I say, a [console] is a [computer], and you're absolutely wrong.
[Guy]: Whatever.
[Me] : Do you play games on a [smartphone] or a [tablet]...?
[Guy]: Yeah, I do.
       So what...?
[Me] : [Smartphones] and [tablets] are ALSO [computers].
[Guy]: Yeh yeh yeh, whatever dude.
[Me] : Do you have a TV that you use to stream movies from [YouTube], [Netflix], or whatever...?
[Guy]: Yeah.
[Me] : Your TV is a [computer], too.

And, there you have it.
When people go on [Facebook], [Reddit], [YouTube] or whatever, to [post], [read], or [watch] stuff...?
They're using a [computer] to do it.

They may think that their [smartphone] isn't a computer...
...but that's because they don't know that they're [wrong].

Basically, the [Internet of Things] is the age that we currently live in, and understanding some
of these concepts, will help little kids become smart and navigate the world with fewer problems
because some smart dude decided to make a bunch of [Quake III Arena] maps way back in the day, and
then he decided to tear apart the many different aspects of [game design], in order to interest
them in the topics that I mentioned.

Such as:
- [computer science]
- [mathematics]
- [programming]
- [performance]
- [hardware]
- [philosophy]
- [instructional design]
- [graphic design]
- [many other things]

```
                                                                                                  ____________________/
\_________________________________________________________________________________________/ [What is a computer]
```

GtkRadiant Cont'd

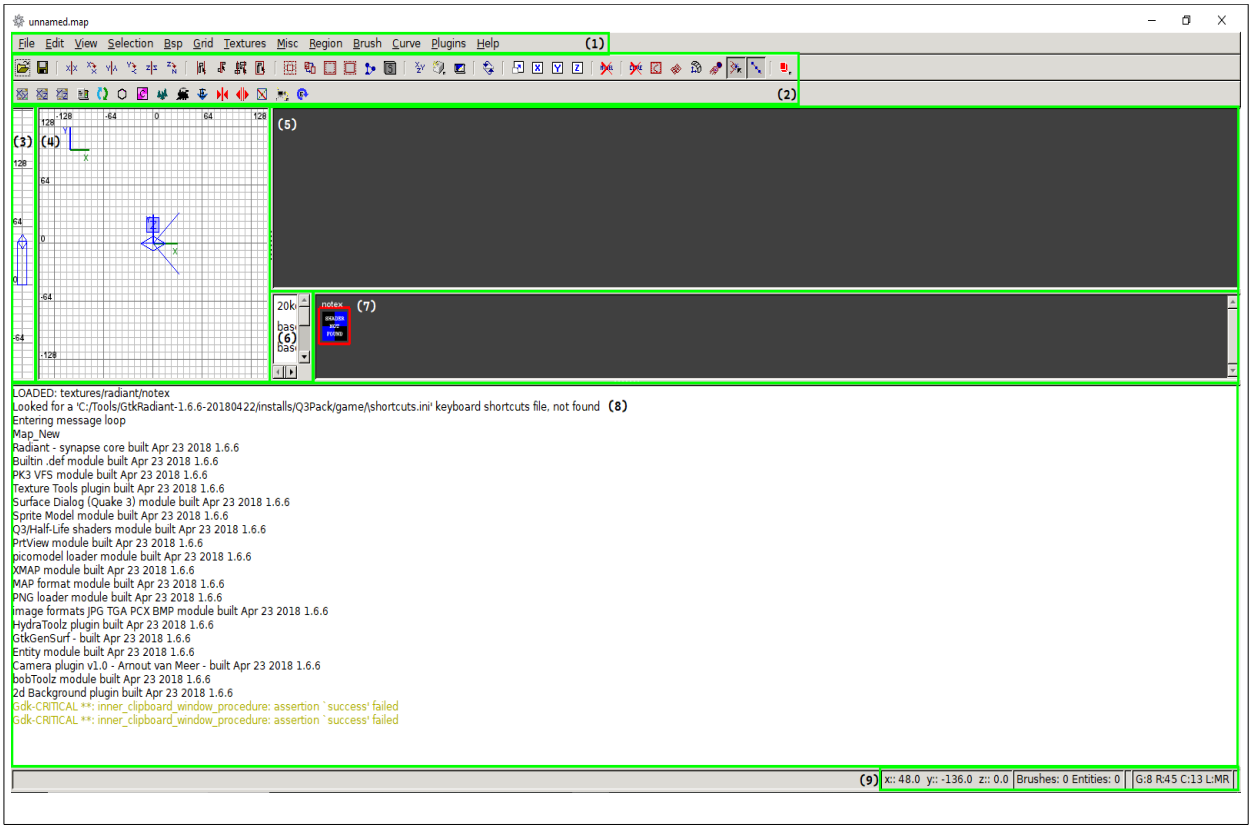
[GtkRadiant] forced the design artist to be really creative with [geometry], [aesthetics], and [lighting], in order to create an [immersive experience] that could exist about (20)+ years before the [Apple Vision Pro], or the [Meta Oculus Rift].

It's a daunting concept, really.

1) Draw stuff out on [graph paper]
2) Create brushes that align with the [graph paper]
3) Come up with a cool way to slap [textures] onto sides of [brushes]
4) Sculpt a realistic looking area using the wireframe approach
5) Do basically the same thing as [architects] and [engineers] that use [AutoCAD], [Maya], or [3DStudio Max]
6) Don't use (artificial intelligence/ChatGPT) when YOU have an idea of your own
7) Rinse, recycle, repeat...
8) Eventually, you can apply this to any current or modern game that uses 3d graphics
9) Develop this stuff called "*talent*" and "*skill*"

GtkRadiant Overview

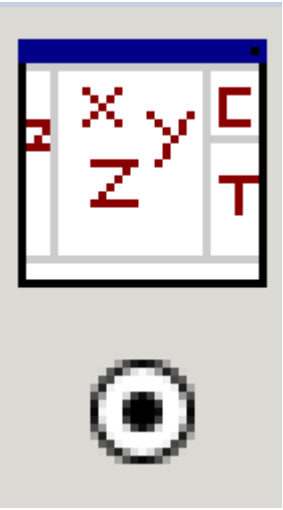
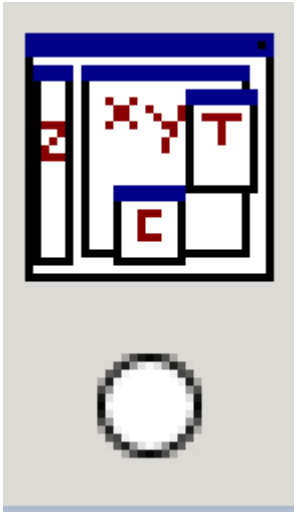
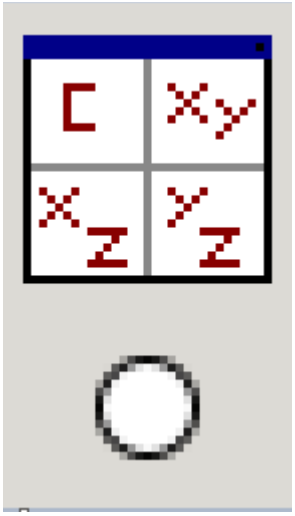
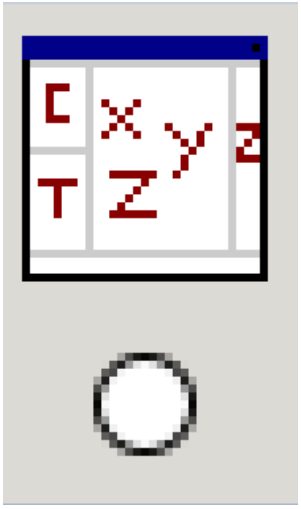
So, when launching [GtkRadiant], this is what the program looks like (minus the green boxes)



Now, to break down what is seen here before I start changing things about the editor...

(1) Menu : Provides menu selection for every aspect of the utility
(2) Toolbars : Contains useful actions and shortcuts
(3) Z-Axis : Used to move, stretch, or clip brushes
(4) X+Y-Axis : Does the same as [Z-Axis], but provides more granular control in (2) dimensions.
(5) Camera : Used to provide a look at how the map currently looks
(6) Sources : Provides a list of all available texture sources
(7) Textures : Shows all (*textures/shaders*) in a texture set, and allows selection
(8) Console : Prints messages for the user, or developer
(9) Status : Provides various status messages about the map in particular, and positioning

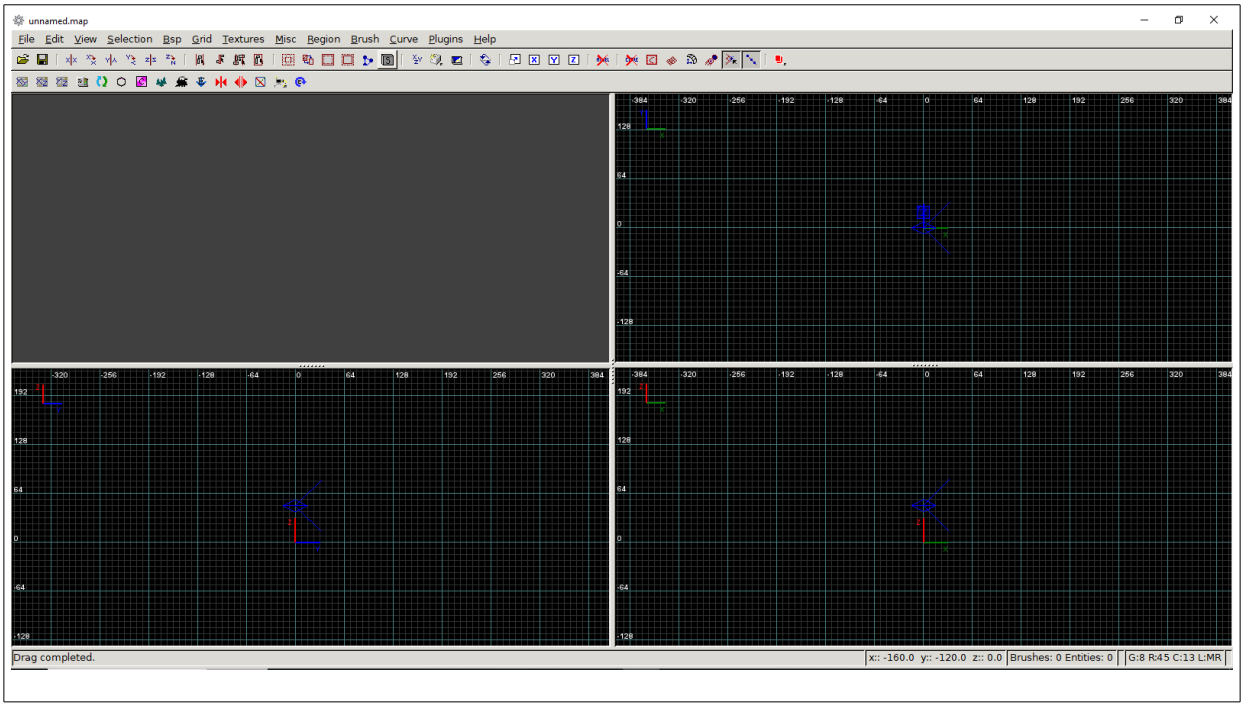
Now, this is the [Default] layout, and [I'm not a big fan of it].
Here are the list of editing layouts...

			
[Default], the Z is on the left, the (*X/Y/Z*) is in the middle, the camera is top right, and the textures are bottom right.	[Windowed], you move the windows whereever you feel is best, or whatever feels the most comfortable.	[Hammer*], this mode divides the camera layout, (*XY/XZ/YZ*) windows into a grid. Textures are handled via the pop out window.	[Inverted default], this works in nearly an identical way to the first option, except it is (*inverted/mirorred*).

If I have to come right out and say it, uh- I like the [Hammer] view, because it works the best in my opinion.
It's called the [Hammer] view, because it was a [similar editor]. It is the mode that I used to create this map...

Index	Name	Title	Build	Age
0	bfgdm1	Crossfire	05/28/2000 0907	23y 1m 12d 4h 23m 23s

...in addition to [all of the others], mind you, the build date says [05/28/2000].
Same editor, same guy, same game, totally different decade.
I also tend to change the [color theme]. So, allow me to change the editor around, and apply the theme...



...and then there ya have it.

_____/ GtkRadiant Overview

Rendering + Gameplay

So, in order to build maps and stuff, it's important to see what a good map might look like,
or how it might play.

[Gameplay] and [rendering dynamics] sort of go hand-in-hand.

If a map [renders well] but [plays like crap] → it won't get played
If a map [plays well], but [renders like crap] → it won't get played
If a map [renders well], and [plays well]   → it MIGHT get played

[Rendering] comes down to the [triangles] that the [game engine] has to render.

It also comes down to a lot of other things, but essentially, the most basic information that is processed
at [every level of detail], are the number of [triangles] in any given map, as well as their [portals/leafs].

I'm not going to get into the nitty gritty of [rendering dynamics], because that involves talking to a pretty
smart dude with a [PhD] in [Computer Science], to understand a lot of what goes into it.

However in [Q3A], the rendering dynamics that affect [map making] + [gameplay] are:
- [r_showtris] : 0 | (*Default*), does NOT show triagles
                 1 | Shows the triangles
- [r_speeds]   : 0 | (*Default*), does NOT show (*rendering speeds + information*)
                 1 | Shows the information being rendered

Here's a video where I load [/devmap 20kdm1] and then toggle the options for [r_showtris] and [r_speeds].

Name	Date	Resource
2023_0715-(Q3A + GtkRadiant)	07/15/23	https://youtu.be/aoS1HEDay4o

I really cannot get everything in these videos to be mega-ultra interesting all across
the board without unpacking and examining [gameplay characteristics], and [development].

[Map development] isn't that different from [3D model development].
[Map development] isn't that different from [engineering].
[Map development] isn't that different from [architecture].

Like, if I wanted to prove to a [CONSTRUCTION COMPANY], that I would not only read a tape measure,
but I could also engineer the blueprints for a residence that could cost upwards of [300,000 USD] or more...?

Uh- it doesn't hurt to show them that it is pretty easy to come up with blueprints when you understand the
software to make those blueprints. And even, the basics behind [architectural design].

[R_speeds] in [Q3A] are broken down like this:
AAA/BBB/CCC shaders/batches/surfs DDD leafs EEEEE verts FFFFF/GGGGG tris

A   : Shaders
B   : Batches
C   : Surfaces
D   : Leafs
E   : Vertices
F/G : Triangles

These numbers are the most important, (*F/G*).
F and G represent the [total number of triangles] that are being rendered to the screen.

Sometimes, it has to render stuff that isn't even able to be seen, because the artist
didn't have a handle on how to separate [leafs] + [nodes].

So basically, [shaders] are textures.
[Batches] are probably shaders after execution by the game engine.
[Surfaces] are the total number of surfaces seen.
[Leafs] are the number of areas that are broken down by binary space partitioning, which
is a technique that was used as far back as [Doom] and [Wolfenstein 3D].

[Vertices] are effectively the number of points that belong to any given [triangle].
[Triangles] are the objects that the [game engine] renders to the screen.

In order to build a level that's going to be played by people, it has to be simple enough
for the engine to render, and it also has to have a pretty good flow to it.

This requires an understanding of [item placement], [spacing], [timing], [rendering dynamics],
and things of that nature.

Take this into consideration.
[Q3DM0] is an extremely simple level design, and it isn't something that will get a lot of play.
Whereas, [Q3TOURNEY4] is a level that's been in (*tournament/1v1*) rotation for a long time.
Same thing with [ZTN3DM1].

Those (2) maps have really good gameplay and therefore, that is why they get a lot of play.
To get into the swing of things, I've been practicing in Q3A, and trying to beat the game in the least time.

Name	Date	Resource
2023_0716-(Q3A Practice)	07/16/23	https://youtu.be/aoS1HEDay4o

Now, apply some of these concepts to developing a map for [Q3A].

Rendering + Gameplay

New Map

So now what I'm going to do, is create a brand new level that I've never made before, in [GtkRadiant].

I'm going to apply some of the concepts that I've covered in other videos.
I'm also going to be heavily relying upon this video...

Name	Date	Resource
2023_0717-(GtkRadiant)	07/17/23	https://youtu.be/-tGdz6oxXZI

It's not going to be a very complicated map, because I want to showcase how to make something simple, first.

To start out, I create a floor.
I had selected the entire brush, and then selected a floor texture from the [20kdm3] texture set.

When I did, it made all (6) sides of that brush that same texture.

However, only (1) side of that brush will be seen by the player, and thus, it is good practice to change
the sides of a brush to a texture like [caulk], because then the polygon is not drawn to the screen after
the map is compiled and then played.

So now, what I've done is I've created walls that sort of cover the sides of the floor.
They also overlap each other. That will cause an issue in the game after the map is created.

What I will do now, is use a technique called [clipping], in order to miter the edges of the side brushes.

So, when using the clipper tool (X), you click a point either inside of the brush or outside of it, doesn't
particularly matter... and then you select another point that you want to (*clip/cut*). The YELLOW area is the
portion that will remain, while the RED area is the area that will be cut away.

You can use (CTRL+Z) to undo things that you do in the editor.
You can use (CTRL+C) to copy entire brushes that are selected, and they can be multiple brushes.
You can use (CTRL+V) to paste entire brushes that were copied to the clipboard.

And like I just did, you can use the rotation button to spin the selected brushes around by (90) degrees.

I'm going to make this whole floor area a bit larger so that the camera view can see what's going on.

So, now what I'm going to do is add some elements on the top like a skybox.
The skybox is a shader that renders a sky as well as shining light into the map from above.

Now I'm going to add in a rocket launcher, and a player spawn point.
Now that I've put all of this stuff together, I'm going to compile it.

New Map

Techniques

In the last few minutes, I tried to run the (Q3) engine to play against [Hunter] in a map I just compiled.
I'm not sure how long the video was skipping and such, but every once in a while I'll have to stop
what I'm doing and check the progress and whatnot.

In this specific instance, I just spent a fair amount of time building a brand new level from scratch.
There are some issues with the map thus far, but it did compile, and I did just fight a bot on
[nightmare] difficulty, and won.

One of the issues was the [jump pad].
Another issue is there's [no health].
Another issue is that it's really pretty simple, not much to it.

However, what I want to focus on is going to require me to change the codec so that I can show
in game footage a little bit better.

Before I start to dissect the video into multiple chunks...
What I'm gonna do is build the class that I talked about building beforehand...

```
                                                                                                                  ____________/
\_________________________________________________________________________________________________________________/ Techniques
  Class creation /-------------------------------------------------------------------------------------------------------------\
/----------------
```

Now, I COULD theoretically take all of the variables that I'm showcasing to launch:
- GtkRadiant
- q3map2
- quake3

But- I really want to focus on making maps for the editor.
[*However, it'll be something that I do at some later point*].

Here's what I did at some later point.

```
Class Q3AItemPath
{
    [String] $Name
    [String] $Fullname
    Q3AItemPath([String]$Name,[String]$Fullname)
    {
        $This.Name     = $Name
        $This.Fullname = $Fullname
    }
    [String] ToString()
    {
        Return $This.Fullname
    }
}

Class Q3AGtkRadiant
{
    [String] $Name
    [Object] $Path
    [Object] $File
    Q3AGtkRadiant()
    {
        $This.Name = "Q3A GtkRadiant"
        $This.Clear()
    }
    Clear()
    {
        $This.Path = @( )
        $This.File = @( )
    }
    SetGame([String]$Fullname)
    {
        $This.AddPath("Game",$Fullname)
        $This.AddPath("Base","$Fullname\baseq3")
        $This.AddPath("Maps","$Fullname\baseq3\maps")
    }
    SetRadiant([String]$Fullname)
    {
        $This.AddPath("Radiant",$Fullname)
    }
    SetWorkspace([String]$Fullname)
    {
```

```
        $This.AddPath("Workspace",$Fullname)
        $This.GetPath("Workspace")
        $This.AddPath
    }
    [Object] Q3AItemPath([String]$Name,[String]$Fullname)
    {
        Return [Q3AItemPath]::New($Name,$Fullname)
    }
    AddPath([String]$Name,[String]$Fullname)
    {
        $This.Path += $This.Q3AItemPath($Name,$Fullname)
    }
    AddFile([String]$Name,[String]$Fullname)
    {
        $This.File += $This.Q3AItemPath($Name,$Fullname)
    }
    [Object[]] GetPath([String]$Name)
    {
        Return $This.Path | ? Name -eq $Name | Get-ChildItem
    }
    [Object] GetFile([String]$Parent,[String]$Name)
    {
        Return $This.GetPath($Parent) | ? Name -eq $Name
    }
    [Object] PullFile([String]$Name)
    {
        Return $This.File | ? Name -eq $Name
    }
    [String] ToString()
    {
        Return $This.Name
    }
    StartProcess([String]$FilePath)
    {
        Start-Process -FilePath $FilePath
    }
    StartProcess([String]$FilePath,[String]$ArgumentList)
    {
        Start-Process -FilePath $FilePath -ArgumentList $ArgumentList
    }
}

$Ctrl = [Q3AGtkRadiant]::New()
$Ctrl.SetGame("C:\Program Files (x86)\Quake III Arena")
$Ctrl.SetRadiant("C:\Tools\GtkRadiant-1.6.6-20180422")
$Ctrl.SetWorkspace("C:\Workspace")

# Assign radiant.exe
$Ctrl.AddFile("Radiant Exe",$Ctrl.GetFile("Radiant","radiant.exe").Fullname)

# Start GtkRadiant
# $Ctrl.StartProcess($Ctrl.File[0].Fullname)

# Assign radiant.log
$Ctrl.AddFile("Radiant Log",$Ctrl.GetFile("Radiant","radiant.log").Fullname)

# radiant.log content
$Log        = [System.IO.File]::ReadAllLines($Ctrl.PullFile("Radiant Log").Fullname)

# Assign q3map2
$Ctrl.AddFile("q3map2",$Ctrl.GetFile("Radiant","q3map2.exe").Fullname)

$Target     = "C:\Workspace\20kdm3\maps\20kdm3.bsp"

# Converts a compiled (*.bsp) back into a (*.map) to open with [GtkRadiant]
Start-Process -FilePath $q3map2.Fullname -ArgumentList "-game quake3 -convert -format map $Target"

# Workspace stuff
$Workspace  = "C:\Workspace"

$Name       = "2023_0717-(testmap3)"
$RegName    = [Regex]::Escape($Name)
```

```
$Files      = Get-ChildItem $Workspace | ? Name -match $RegName

$Map        = $Files | ? Extension -eq .bsp
$AAS        = $Files | ? Extension -eq .aas

ForEach ($Item in $Map, $Aas)
{
    $Target     = "{0}\maps\{1}" -f $GameDir, $Item.Name
    If ([System.IO.File]::Exists($Target))
    {
        [System.IO.File]::Delete($Target)
    }
    [System.IO.File]::Copy($Item.Fullname,$Target)
}

$Game       = Split-Path $GameDir | Get-ChildItem -Filter "ioquake3.x86_64.exe"

Start-Process $Game.Fullname -ArgumentList "+seta sv_pure 0 +map $Name"
```

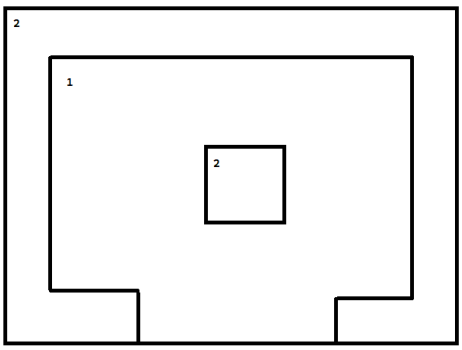
Map creation / Class creation

So, I'm going to come up with a more intricate [design philosophy] for the map.
Here are the criteria I want to establish...

1) multiple tiers
2) rocket launcher, lightning gun, armor, health
3) really simple layout that could be fun to play again
4) same textures from [20kdm3]

Here we go.

I'll start out with [MS paint], and come up with some shapes for the overall layout.
This map isn't going to be incredibly complicated by any means, but it WILL be somewhat
busy to create.



General shape of the map

One thing to keep in mind, is that there are various editors (*3DStudio Max*) out there that'll allow you to set
the background as an image, that way you can sculpt things out according to the drawing or what have you.

As the map gets larger and larger, then I can implement some nested details and such.

Part of the reason why I've been segmenting so many brushes and stuff in the manner that I have,
is because I know how the BSP process is going to subdivide all of the polygons that the game engine
will have to render, therefore...

If I know what the compiler is going to do to the map and the edges...? I can account for some of that work
ahead of time, and then make things easier on the game and the computer that would go to play the game.

Map creation

General Geometry

So, over the last several minutes I've been [modifying the geometry] because I [expanded] the scope of the map.

As time goes by, and I use textures that sort of provide a "*feel*" to the map, I start to rely on shaping the map around the textures and then the geometry sort of comes in second place to the textures.

Simply put, if the textures aren't applied with any visual theme or sense, people aren't going to play the map, nor will they really enjoy it.

Whereas, if the geometry sort of allows the item placement to be a bit of a no brainer, then what happens is that adding items provides a bit of context to the map and it gives a bit of incentive to expand around the items.

I'll elaborate more as I continue to build this.

I'm using the (H) key when I select certain brushes to hide them. They're not actually deleted.

Play Testing

I'm aware that the video is chopping out and the encoder gets overloaded when I go to play the map.

It's just what I'm working with for the time being until I splice stuff together.
Anyway, there's a few problems in the map.

Textures, a couple of them are wrong.
I was able to fix the lighting by changing the skybox.
By using a brighter skybox texture, it will illuminate the map even more.

I forgot to put the rocket launcher back into the map. I also ... that's basically it.

But, for a single day, this map is looking to be pretty well made.
I can continue to add to it and provide additional gameplay mechanics that utilize the top tier where the rocks are. As I was [playtesting] the map, I came up with an idea to use some [fog] where the dirt is, up top.

So, I'll actually add that now while I'm fixing the couple issues I found.

The fact that the [megahealth] and the [red armor] are so close to each other feels pretty ignorant.
Maybe that's not the right word...
There's gotta be a better way to place those items.

Finalizing the level

I won't be able to get into the finalization of the level in this particular video.

However, I will definitely be making many more changes to this map, as well as adding various elements to it that make the gameplay a lot more replayable and enjoyable.

In this particular video, I was able to cover many aspects of GtkRadiant.
[*It has been recorded in 900p, so I apologize for the resolution but I am not going to reencode this video... Maybe I will if I think it can be done with the hardware I have right now...*]

Disregard the above boxed in comment. This was all upsampled to [1080p/60] which is why the encoder was overloading when playing the map in the game.

[To be continued]...

Michael C. Cook Sr.
Security Engineer
Secure Digits Plus LLC