

Andrew Yoder

Level Design Blog

Making Your First Map in Quake (Part 3)

MAY 18, 2023MAY 18, 2023 / MCLOGENOG

For the rest of the series, see [Preface](https://andrewyoderdesign.blog/2019/06/08/so-you-want-to-learn-3d-level-design/) (<https://andrewyoderdesign.blog/2019/06/08/so-you-want-to-learn-3d-level-design/>), [Part 1](https://andrewyoderdesign.blog/2019/06/08/making-your-first-map-in-quake-part-1/) (<https://andrewyoderdesign.blog/2019/06/08/making-your-first-map-in-quake-part-1/>), and [Part 2](https://andrewyoderdesign.blog/2019/06/16/making-your-first-map-in-quake-part-2/) (<https://andrewyoderdesign.blog/2019/06/16/making-your-first-map-in-quake-part-2/>)

Welcome back for the conclusion to this tutorial series! At this point, you should know your way around the Trenchbroom level editor and the process of compiling maps. If you have those basics down but aren't sure how to make the jump into your first map, you are in the right place!

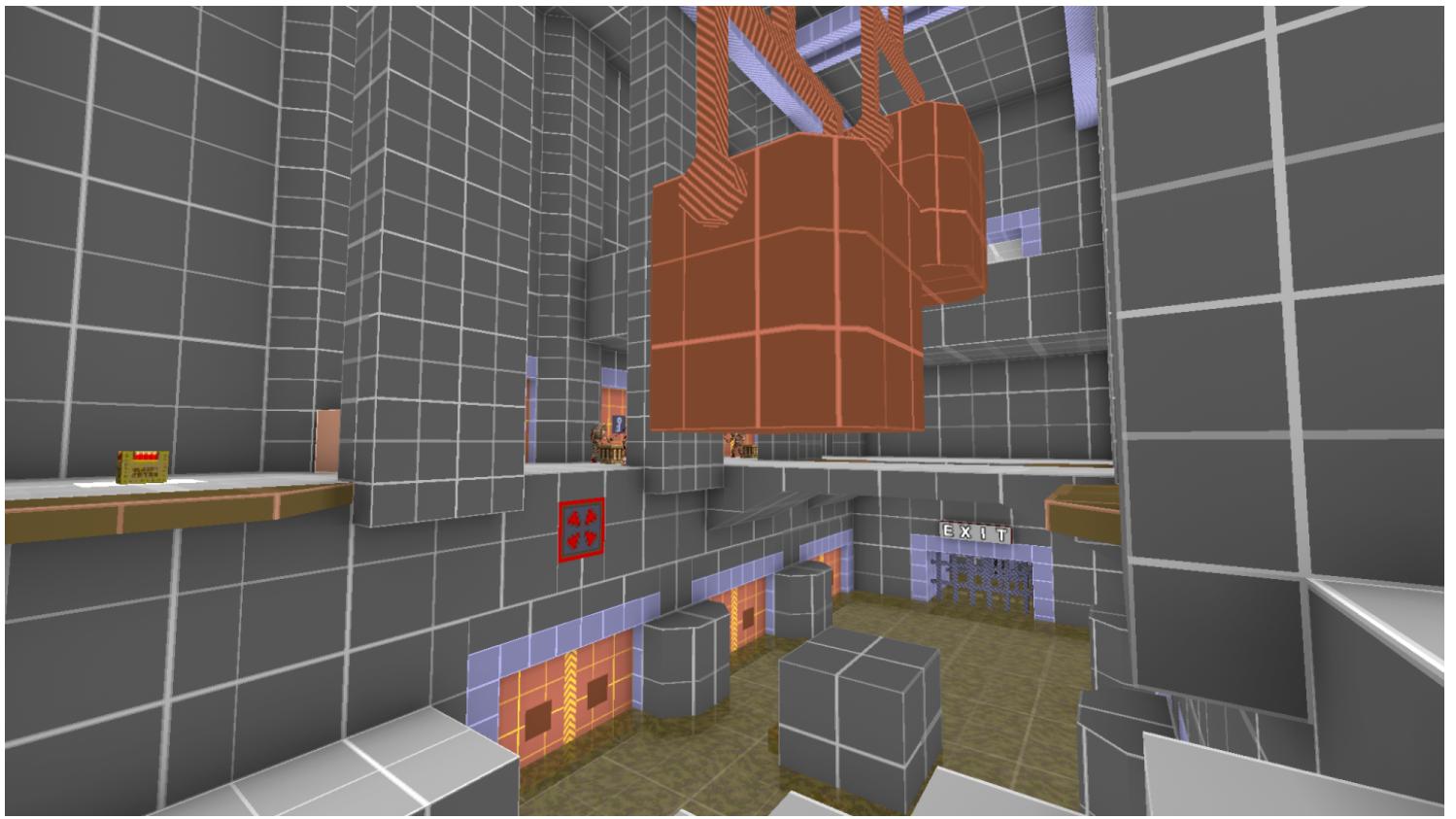
Compared to learning tools, the jump into design is the fun part, but it can also feel overwhelming. My goal here is to provide a variety of starting points and prompts to help you overcome the blank page. Each of these starting points comes with tradeoffs, but none of them are a wrong or bad way of starting. Whichever option excites you the most is a great way of starting!

Method 1: Gameplay First

Think about the fun mechanics available in the game and then imagine how your level could serve those mechanics. This could be the weapons, powerups, enemies, or even the stranger mechanics like doors, trains, and slime. Alternatively, think about a mechanic you don't normally like, try to imagine a way to make it fun. I find it helps to think about this for each encounter or gameplay-beat instead of trying to solve the whole level at once.

A few examples:

- An encounter featuring the rocket launcher. First empowering the player to show off the rocket launcher's strengths, then ramping up the intensity to challenge the player.
- A showcase for the Quad Damage powerup. Give the player a whole bunch of enemies to chew through. If you want to increase the challenge, make the player feel like they're in a race to eliminate the enemies before the powerup runs out.
- An encounter making use of crushers! Evade the crushers to proceed deeper into the level, or lure monsters out and under the crushers for a satisfying crunch.
- Play with the monsters! A jumpscare with a spawn, a hasty retreat from a shambler, or a dancefloor filled with knights!



(https://andrewyoderdesign.blog/wp-content/uploads/2023/05/ayry_v2_2023-05-18_18-52-48.png)

Above is a gameplay-first blockout of an unreleased map.

If you find that you are still struggling to imagine a gameplay encounter, I recommend doing some play-research. Look into other maps and note the moments where you feel a strong reaction: can you identify ways to improve that encounter and modify it to your own work? Are there moments you'd like to study and attempt to emulate?

Once you have an idea, experiment! The best part about the gameplay-first method is that you can quickly playtest and iterate until you are happy with the results.

A word of caution though: even when you are solving for gameplay in abstraction, you are setting up constraints for your art. You might be painting yourself into a corner. Or, if you intend to collaborate on the map with an artist, you might be passing along a bunch of hard problems and making a bad time for your artist. As you take more maps through the whole process, you will get a better sense of where you are creating problems, but expect a few surprises along the way if this is your first map.

Method 2: Art First

Imagine some striking compositions that would make a gorgeous screenshot, or imagine some spaces that are enjoyable to move around in game. Imagine the theme, the architecture and its building materials, the mood and lighting.

If you are struggling to imagine anything, go image searching for architecture, photography, and art. As you discover things that excite you, start collecting them into a reference sheet or mood board. Reference gathering is a skill in itself, and you'll find you have a better understanding of your own tastes as you practice.

Once you have some ideas in mind, start experimenting! These experiments should help you answer

some questions about how to realize your ideas within the limits of Quake.

A few specific questions to think on for your experiments:

1. What geometry and what textures best realize the ideas?
2. How do these ideas combine to create a consistent visual language and style?
3. How is the level lit? Will there be light fixtures?

One experiment that can help answer these questions is to build a “beautiful corner”. This is a practice in AAA environment art where you take a small part of a map that represents a meaningful sample, and solve the art to nearly final quality. From this experiment, you should solve many of the problems for the level’s geometry, textures, lighting, and overall visual language.

If you aren’t happy with the result of your experiment, you can try again. You can also go back to reference searching, informed now by the new questions your experiment prompted.

Once you are happy with a few compelling environments, you can start populating the level with monsters and items. You may find that you need to reevaluate some art decisions as you adjust for the gameplay, but Quake’s gameplay is remarkably supportive for art-first levels.



(https://andrewyoderdesign.blog/wp-content/uploads/2023/05/sj2_yoder_2023-05-18_19-16-11.png)

Above is my contribution to sewerjam 2. This map started from the art, and solved gameplay along the way.

As with the gameplay-first method, this art-first method comes at a cost: what works well for art may not work well for gameplay! You may find it difficult to add satisfying gameplay on top of your arted map. If you are building in a collaboration and are handing the level off for someone to add monsters and items, they may be unhappy with the constraints the art imposes.

Method 3: Fantasy First / Experience First

This method is all about the player experience, the fantasy that the level delivers as a product of both its art and design.¹ Imagine an exciting player story, the kind of memorable experience a player might later tell friends about.

For example: “There’s this snow level with a spooky castle. The front gate was closed, so I had to find another way inside. I found a frozen aqueduct that led through a cistern full of zombies, but I made it inside and blew the front gate open!”

Or, “I thought I’d made it to the end of the dungeon… A horde of monsters was on my tail, but the end was in sight down the hall. I was feet away from the exit portal when a portcullis slammed down in front of me, blocking the exit! I had no choice but to turn and face the horde”.

In both of these examples, the level geometry has an identity, and the player actions are loaded with richer meaning. The level is a *snow level*, a *dungeon*, or a *castle*, not just abstract geometry that facilitates the gameplay. The player is *infiltrating*, not merely *entering*. Giving the level geometry identity helps the player remember it and make a story of it. Even with the gameplay focus of Nintendo’s Mario series, levels and mechanics are heavily themed to create strong identities that serve memorable player experiences.

If you are struggling to think of a player story, replay one of the original episodes of Quake. After completing each map, write a short description of how you experienced your favorite moment in the map. Think about how you would heighten the story to excite a listener, and then consider how you would apply that into improvements for the level.

Once you have a player story, build it out and playtest! From here, the work is similar to the gameplay-first method above: playtest and iterate!

The strength of this method is how it aligns art and design. “Storm a castle” is a different player fantasy than “storm the warehouse” or “storm the spaceport” even though each may share similar design. Mood has a role here, architectural identity and visual language matter, but there are also gameplay hooks to design around from the start.

The weakness of this method is that it requires you to be comfortable solving art problems *and* design problems simultaneously. This is a lot to ask of anyone, especially on a first map, so I recommend trying the art-first and design-first methods if you find this one overwhelming.

Another weakness of this method is a bias toward setpieces. Most Quake maps rely on “meat and potato” patterns to make up most of a level’s experience and only offer one or two setpieces to spice up the pacing and variety. “Meat and potato” gameplay has an important role as satisfying filler, but the fantasy-first method doesn’t offer clear guidance here.

Method 4: Pattern Language²

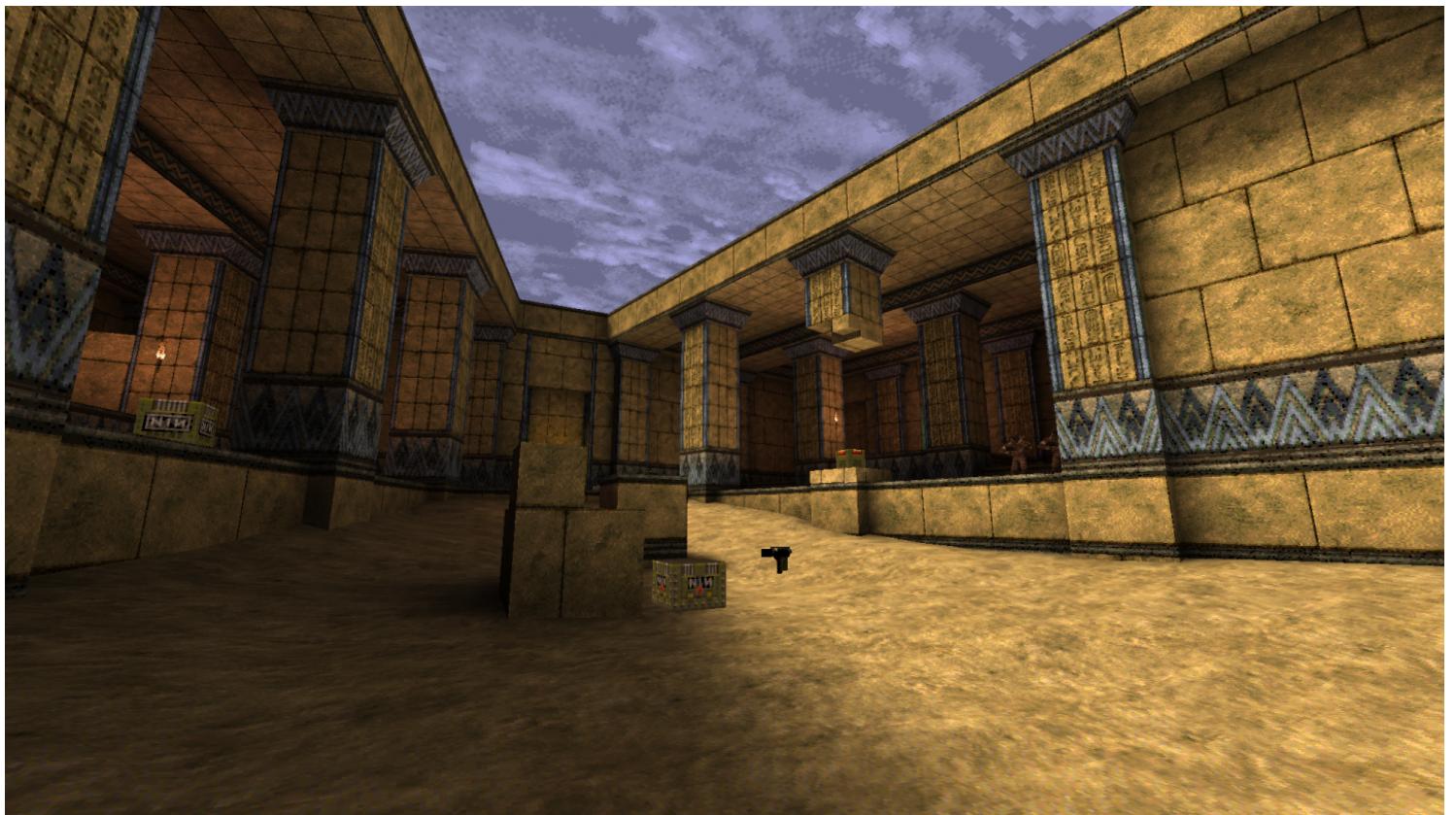
This method uses repeatable patterns as the building block for levels. Patterns can be any repeatable chunk of level geometry or entity arrangement that reliably creates good player experiences. Patterns can also be more abstract ideas, such as a method of structuring pacing to encourage an emotional journey.

A few examples of patterns:

- A “loop” is a pattern of level geometry where the player or enemy movement creates a circular

shape. In a loop, the middle of the circle is either filled with cover that the player can duck behind and kite enemies around, or the middle is a hazard with open sightlines across it; these are different sub-patterns with their own tradeoffs. If the player is overwhelmed by numerous foes, the loop pattern can fall apart as the player is trapped.

- A “figure 8” is a pattern created by combining two loops. This gives the player more path options than the loop pattern, which makes it more suitable for higher enemy counts. As with a loop becoming a figure 8, a figure 8 can scale to create other higher-order patterns with their own tradeoffs.
- There are a variety of “stair” patterns that produce interesting one-way loops with dropdowns and ledges to leash enemies in place. Sock’s levels in Arcane Dimensions use this pattern extensively to good effect.³
- A “Romero lift” is a slow elevator with waves of enemies dropping in while the player is trapped waiting for the elevator to arrive at its destination. This pattern first appeared in E2M6 “The Dismal Oubliette” and is a reliable setpiece. This pattern is a little more abstract, allowing for many possible variations and reinterpretations.
- A “reversal” is an abstract pattern focused on pacing and the emotional journey the player experiences. A typical reversal starts the player in a vulnerable positions against some obstacle, and once the player overcomes the obstacle, they can turn that same threat against a new wave of monsters.
 - In E1M3, there is a room where ogres rain grenades down on the player in a pit. Due to the height difference and the nature of bouncing grenades, the player is at a disadvantage. Once the player defeats the ogres, the player can climb up the ledge to where the ogres were, then press a button to release some zombies back down in the pit below. Now the player gets to be the one raining grenades from on high!
 - Turret setpieces in mid ’00s FPS games frequently follow the reversal pattern. The player faces an enemy turret and has to zigzag from cover to cover to eventually reach their flank and defeat them. Then, once the player grabs the turret, new enemies arrive and its the player’s turn to put the turret to use.



(https://andrewyoderdesign.blog/wp-content/uploads/2023/05/e1_yoder3_2023-05-18_18-53-45-1.png)
Above is my map e1m10 from *Coffee Quake 2* (<https://www.slipseer.com/index.php?resources%2Fcoffee-quake-2-double-shot.213%2F>). I used several stair patterns and a partial romero-lift pattern in the design.

There are also patterns that reliably produce *bad* experience. These bad patterns are sometimes called “anti-patterns”. In UX there are “dark patterns”, but I feel this term is best left for patterns that have malicious intent or seek to manipulate the users. Here I’ll simply call them “bad patterns”.

- A “door problem”, where the player fights from a threshold into a combat space instead of fighting in the arena, is a bad pattern that reliably produces boring, tedious gameplay.
- “mazes”, where the player makes a series of blind choices to discover if they are or aren’t facing a dead end.

There are always exceptions though. Plutonia’s map11 “Hunted” takes a maze pattern and some archviles to produce a highly memorable, and terrifying, experience. Meanwhile, Kaizo Mario level design often celebrates “bad” patterns as a form of joke and challenge; some of these levels are brilliant in the ways they defy convention.⁴

A strength of a pattern language is modularity. You can plan out a whole level quickly with a few patterns, or even build out the level geometry without a plan by improvising on a few familiar patterns along the way. Many Quake speedmaps are built with this simple repetition of patterns as building blocks, and the Quake modding community has built up a vast range of patterns—some named, most not—in the 25+ years since the game released.

The weakness of pattern languages is that you need to build up a vocabulary to make the most of it. This takes practice trying out different patterns or stumbling into others to see what works and doesn’t. If your vocabulary isn’t rich, you may find yourself repeating the same few patterns in a way that ruins the player experience. Developing a rich vocabulary is the work of a career—it takes time. But where you find your vocabulary lacking, you can always fall back on the other methods described above, and seek

new patterns when playtesting the work of others.

Conclusion

This post has been a long time coming. When I started the tutorial series in 2019, I'd only been making Quake maps for a little over a year, and I frankly was not equipped to write the above. At the time, I would have advocated strongly for a gameplay-first approach and left it at that, but this would have been an expression of my own tastes, biases, and blindspots. My experience with modding in the 4 years since then, and my continuing work in AAA, has given me opportunities to try many different methods of working, with many different struggles as a result. It is from these struggles that I believe I've arrived at a fair assessment of methods for mapping.

Above all, I've held off from concluding the tutorial due to a worry that I would create a dogma around how a mapper ought to go about mapping. There is already enough level design dogma from AAA designers (frequently with books to sell), but modding is not AAA. Even within the industry, methods vary from team to team. The best practices are the ones that can adapt to local needs.

My recommendation: experiment with different methods, adapt them as you go, think critically on ways to improve, and—if you are working with a team—make time to talk it out!

If you want people to talk with about Quake modding, I recommend heading over to [Slipseer](https://www.slipseer.com/index.php) (<https://www.slipseer.com/index.php>), or over to the Quake Mapping Discord (<https://discord.gg/xDuxZsfpsf>). These are welcoming communities that will help you get started and provide playtesting as you experiment. They also frequently organize community mapping events, which are a great way to jump in.

Thanks for reading, now Go Map!

-Andrew

Notes:

1. Most players don't experience a level's design and art as separate things. When a player likes or dislikes a level, it is because of the experience that art and design provide together. This is extremely evident in youtuber commentary where players often say things like "I like the design of this level, it's so bright and sunny", or "this is some bad design, the window model doesn't match with the inside of the house". A fantasy-first or experience-first method aims for a more organic process that aligns art and design toward a shared goal. This method also rejects the modern western AAA practice where art and design are separate disciplines. Other worlds are possible, and modding is a safe environment to explore them!
2. The specific terminology of Pattern Languages originates with Christopher Alexander's *The Timeless Way of Building* (1979), and *A Pattern Language* (1977) cowritten with Sara Ishikawa and Murray Silverstein. Unlike the subject, some of the writing is far from timeless, but the way the books give praise to vernacular architecture is worth a look. The metaphor of pattern languages have also grown beyond the original books, which is closer to how I use it here.
3. Sock has been kind to share his patterns on twitter, archived now here <https://book.leveldesignbook.com/process/combat/encounter#arena-design> (<https://book.leveldesignbook.com/process/combat/encounter#arena-design>), as well as on Slipseer <https://www.slipseer.com/index.php?threads/top-10-mapper-hints.186/#post-1126> (<https://www.slipseer.com/index.php?threads/top-10-mapper-hints.186/#post-1126>).
4. See for example "Grand Poo World 2" by BarbarousKing, <https://youtu.be/GY0HnqnYMO8?t=241>

(<https://youtu.be/GY0HnqnYMO8?t=241>), timestamped where a hidden block requires foreknowledge or the player will have to restart from the last checkpoint. BarbarousKing says it best himself: “a really mean invisible block that has killed a lot of people, and caused a lot of laughter for me”.

Further Reading:

- Check out Robert Yang’s site <https://book.leveldesignbook.com/> (<https://book.leveldesignbook.com/>), which is a fantastic collection of level design theory and history.
- Seth Marinello gave a great talk on the struggles with the blank page, and how to overcome them: <https://youtu.be/R75g3elj7y4> (<https://youtu.be/R75g3elj7y4>)
- JP LeBreton’s Oblique Strategies for Doom Level Designers (http://vectorpoem.com/doom_oblique/), inspired by Brian Eno’s “Oblique Strategies” (1975), is a fun example of pattern languages as an ideation tool

Categories: [Uncategorized](#)

[BLOG AT WORDPRESS.COM.](#)