map ag

Issue 2: MINI GAMES

Cover Picture by @CrushedPixel. Logo design by @GamerGuppy.

About the Magazine

This project is a community driven and contributed magazine. By publishing we seek to develop the wonderful craft of Minecraft Map Making. All content remains the property of the respective author and is used with permission. All trademarks referenced in this publication remain the property of the respective trademark holder.

About the Team

This issue has been carefully crafted by the talents of:

 @abrightmoore - Editor in Chief. http://www.brightmoore.net

 @mwthecool - an avid fan of video games, such as Minecraft, and their communities. He is currently working on several maps that will be released soon!

 @cocoamix86 - made a modpack! 106 mods total, will probably need 2GB+ ram to play smoothly. Have a gander!

 @gamerguppy - Reinvents simple things to be overcomplicated for absolutely no reason. www.youtube.com/gamerguppy

 @skylinerw - Command advocate.

 @lemoesh - http://moesh.ca/ - Moesh is an unofficial community manager who has led a number of collaborative projects focussed on improving the tools and talent in the map making community. His goal is to reimage Minecraft as a game development engine. His other projects include Limited Engagement, a Map Maker podcast by Map Makers for Map Makers, and testfor[dev], a hub for Map Making advice: http://testfordev.com/

 @Gnasp - a programmer that likes to play Minecraft. He's working on a few maps, has created the map making tool Smelt, and writes about all of this on www.gnasp.com

 @realplagiatus - small Mapmaker, recently published Negentropy, active feedback-giver on /r/realms

 @StealthyExpert - a console Map Maker who is bringing the rich world of PC creations to XBOX

 @immersivemind - Global Education Consultant - EdTech & Games-Based Learning. Makerspace creator. www.immersiveminds.com

@... your name could be here - write an article or provide art for future editions! See submission guidelines in The Lobby.

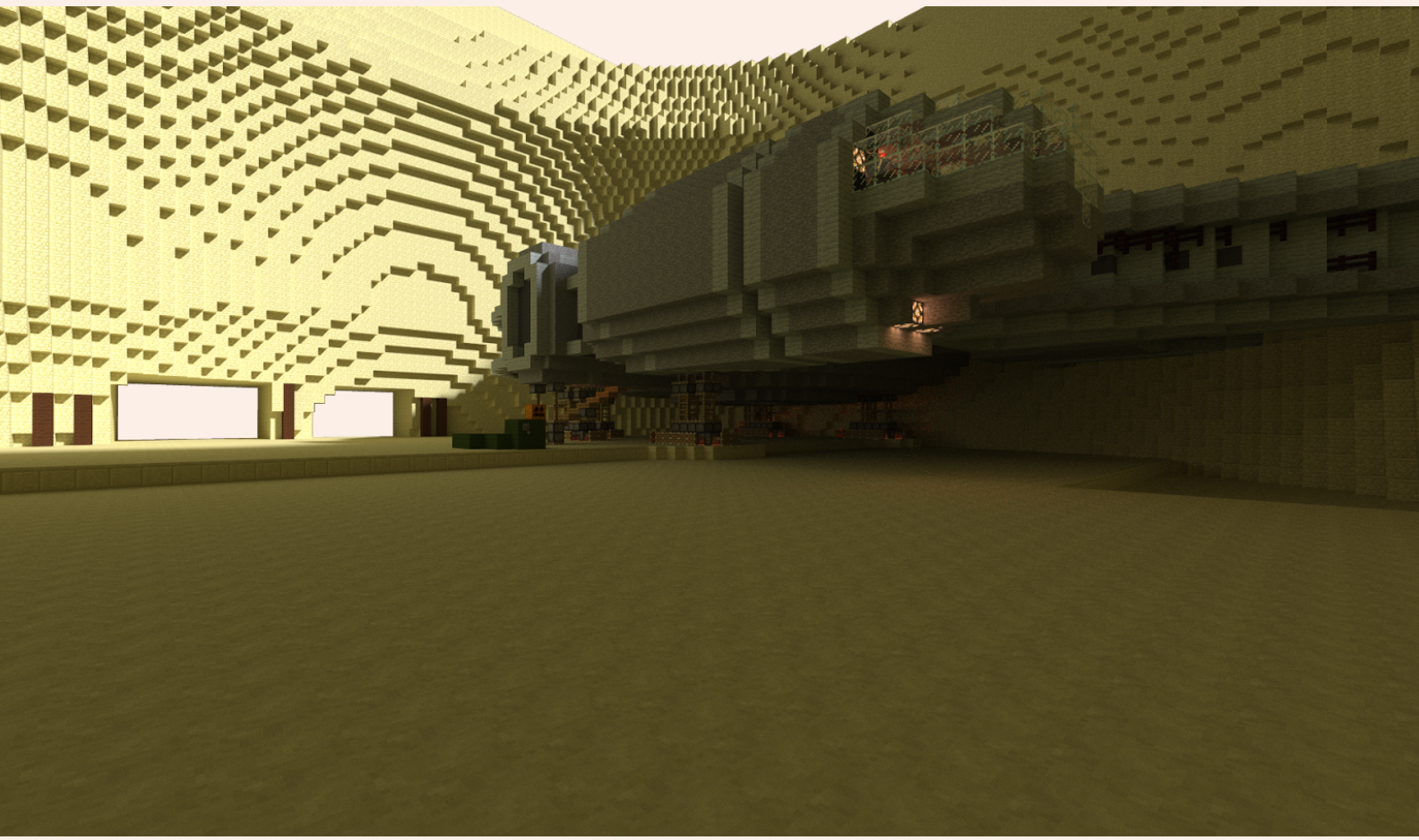
Production Assistance:

This publication is a community effort and this issue has been compiled with input from:

@qwertyuiopthepie and the Minecraft Map Making community - Review and guidance

Map[Mag] is supported by donations from: @immersivemind and @cocoamix86

Contents



The Lobby

Welcome back to the Minecraft Map Making Scene!

We must be doing something right because your response and feedback to our first issue of **Map**[Mag] is overwhelmingly positive and supportive. Thank you!

Our **first issue** (from http://testfordev.com/mapmag) introduced you to the Minecraft Map Making scene and breaking news. There was something for everyone, including exciting new maps and tools. We looked at how to choose the map type you will build, and what it takes to create a polished product.

We kicked off our map making project by deciding on a theme and a game type. If you missed it, and you would like to join in, simply open up issue 1 again and read The End. Each issue will include a new step in the process so that we all are working toward a working and fun map that we can all play and discuss.

This **second issue** is looking at the type of map called a 'Mini-Game'. @CocoaMix86 shares thinking on why these maps are the right size for most projects. We take a look at how you go about efficiently creating a working level in our article on *Project Management*. We also look at how to use one of the most important map mechanics that Mojang has given us: the *ArmorStand*. We also explore some essential tools and websites that Map Makers use to make their lives easier. In our Map Making review feature we take a close look at the team combat Capture-the-Flag (CTF) game *Nuked* by @MC_Labs15 and @CocoaMix86.

Our ace interviewer @mwthecool has tracked down innovative Minigame "Shattered" author, and long time mapmaking force-for-good, @Panguino who shares an approach to minigame design, development, and release.

The latest news and releases of significant interest to the community can be found in our Tech News section.

While the digital ink has hardly dried on this issue, we are already planning what issue 3 will look like and have a huge number of great ideas about what you want to see. But we can't always predict what you are thinking, so make sure you get in touch via Twitter @MapMakingMag or email MapMakingMag@gmail.com to tell us what you want in future issues of **Map**[Mag].

Map[Mag] is free and we can certainly use some funds to improve the design and production of each issue. **Feel free to donate via PayPal** using **abrightmoore@yahoo.com.au** as the account. Mark your donation with a note explaining the funds are to improve **Map**[Mag]. We will update you on who is donating and what we are using the donations for. If you want your donation to be anonymous, include this in your donation message. We are currently exploring design and publishing improvements.

We are all excited about the amazing new things people are doing with Minecraft and vow to continue to bring you all the news and articles we can fit into each issue for as long as our computers have power (RIP @Jigarbov's PC). Enjoy Issue 2, the Mini Game edition. We enjoyed preparing it for you!

- Adrian Brightmoore, Editor
  Twitter: **@abrightmoore**

Submission Guidelines

We are interested in what YOU have to say. Content you make for **Map**[Mag] can be sent to:
mapmakingmag@gmail.com.

The best letters, articles, art, and other work may be selected for inclusion in **Map**[Mag] editions or on affiliate websites and other communication channels. Because **Map**[Mag] is made by the community for the community, **Map**[Mag] is free for readers and we don't pay you for anything. We ask for permission to include your work in the magazine.

Any content you submit must be your own work, or work that you have the right to submit. By sending us your work you agree that we may edit it for readability or make changes we think are necessary for the magazine. If we decide to include your work you acknowledge that you have granted us the right to publish your work in **Map**[Mag] and you understand that your work may be quoted or discussed on the internet by anyone in the world without limitation.

All other rights to your work remain with you. You own your work. We are allowed to use it for **Map**[Mag]. It is that simple.

We will credit you by real name, game name, social media account, or another method that you prefer and that we mutually agree. We will not share your email address without your express permission. If you do not tell us how to credit you for your work then you will not be published in **Map**[Mag].

If we refer to you or your work in **Map**[Mag] you acknowledge that we do so in good will and our intention is not to damage or harm.

DISPUTES

Writing about what you enjoy and hearing from other people with similar interests can be great fun. When people are excited about what they are doing sometimes things can get a little heated in a large community. If you have any concerns over what **Map**[Mag] is doing or how we are doing it then please contact us describing your concern. This will allow us to understand how we can do better. We can be reached at **mapmakingmag@gmail.com**.

By reading this magazine you agree that the Contributors, Production Team, and anyone associated with this activity are not liable for any damages to the fullest extent permitted under law. You agree that any dispute arising from this publication is governed by the laws of New South Wales, Australia.

Message

Map[Mag] is created by Map Makers for Map Makers. We make something we want to read in the hope you will enjoy it too. Community is about connections, so we welcome your feedback and ideas on how we are doing. Send your letters to us at mapmakingmag@gmail.com or on Twitter @MapMakingMag. We may publish your letter in future editions, and your feedback may be edited for clarity.

Congratulations, you have over 200 followers.

Issue 1 really arrived with a bang. Over 200 people joined us on Twitter @MapMakingMag in the first 72 hours. Here are some of your thoughts and comments:

Jaziel wrote:
I'm a graphic designer and also an amateur map maker, your work has helped me a lot when it comes to creating maps. I love the idea for a Map making Magazine.

Thank your for your kind words Jaziel and we're always glad to help. Stick with us and we'll make great maps together!

Austin writes:
Keep up the good work!

Thanks Austin, that's the plan!

Stephen says:
Brilliant magazine. Love the idea and execution. Keep it up.

Aw shucks, Stephen. You are clearly are a person of wonderful taste. I hope this issue is up to standard!

Joel says:
Great resource and a really good read. Recommend this to all my followers into #minecraft. Good work.

Torchwhisperer writes:
Well done! Thanks for making this and for including the pocket community.

*You are welcome, we are having heaps of fun with **Map**[Mag]. We promise to go wherever the Map Makers are!*

Simon says:
Awesome resource & a great read. Don't be daunted if you are starting your #MinecraftEDU journey. Well worth a read

Stay tuned - we will explore the amazing range of Minecraft worlds used for educating and engaging kids in outcome-based learning.

Poseidon says:
This magazine is amazing! It is so good. I want more to come out.

You are in luck.

N:CS Computing suggests:
You should include stuff for @Raspberry_Pi too

Great suggestion, we'll look into it.

MC Architect reminds us:
Map making is so easy. Only a wasteful 5 months! :D

Only 5 months? You got off easy…

ZloYxp shares this feedback:
You might know a whole lot about map making, but you might want to get someone who knows a bit about design and fonts and magazine covers to work with you. With that said, the content itself is great. Just wish it was presented a bit... fancier?... cleaner? something amongst those lines.

Thanks for your kind words. Design is a professional skill, and for us to get the right input we need to select and work with a Designer on our vision for the magazine. Once we can afford it, via your kind community donations, we will take the right steps to make the improvements you are asking for!

Clava offers:
Really nice idea! I love the game, I love to see new maps and I have always wondered what it is like behind-the-scenes, i.e. How does the Hypixel server present the football game? Do they use arm stands and command blocks? Is the ball a ghast-ball just covered with a skin? How do they know I kicked the ball at a certain angle? And so on.. if you will cover this kind of stuff it will be great, but everything map-making related has my interest, even if I won't do one myself.

A lot of server games are built as code (plugins and mods) extending the Minecraft server. We will look at similar maps to those found on server networks. Most features can be created in the PC game now!

Microgames are Useful

By @CocoaMix86

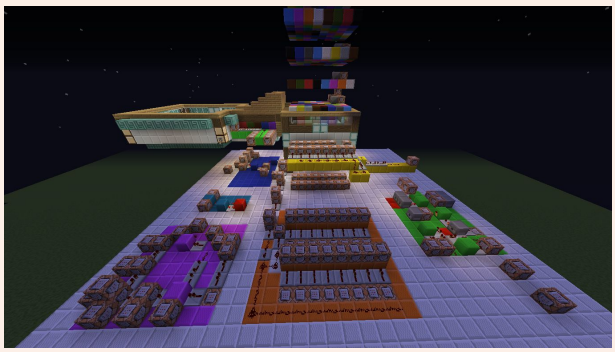
“Confusion is the welcome mat at the door of creativity.” - Michael J. Gelb

Most map makers all dream of building that one map that is so awesome, it will get a crazy amount of attention and they will become famous. I am subject to this. Lots of times, they go too big and the project falls apart. So here I am to say to you all:

“Don’t go too big”

Well, I should rephrase that: Start small and work your way up. This is the reason I have started my newest series of projects called “microgames”. These are a series of really tiny games made within a few days that demonstrate a few bits and ends of what command blocks can do, even in a small amount. Microgames don’t take much effort either and are easy to make, even for beginner *mechanics**.

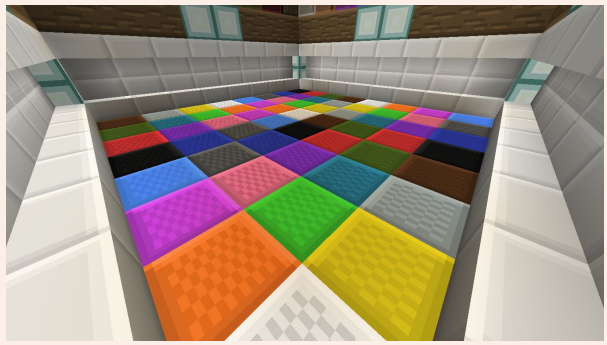
Take Quick Colors for example:



This is literally all the redstone. This was built within 7 days and all it took was a basic knowledge of setblock, fill, and scoreboard commands. It was really easy to execute. (it was also built in the 1.8 snapshots, and now with 1.9 and 1.10 can be vastly condensed and improved)

- *”Mechanic” is a term used by some in place of “command blocker”.*

Another great feature about microgames is that the arena tends to be super small, which makes for quick games that last only a few seconds up to a few minutes. It is a great game to play within your spare time.



*Quick Colors arena is only 8x8 (64 blocks!)**

To me, microgames are a great way to slowly build up understanding and knowledge of command blocks and what they can do. Since the game is really small, you can mess around a lot and still keep track of everything going on. Plus, if something breaks, it does not take very long to find the error.

So, if you are having trouble with your game that takes 20,000 command blocks to operate, I suggest you tone back, relax, and try building a microgame. Let your creative juices flow and try something new. And hey! Maybe while building one, you will get inspired to expand on the idea and incorporate it into your bigger game or you will realize that certain areas could actually be incredibly simplified.

About the Author

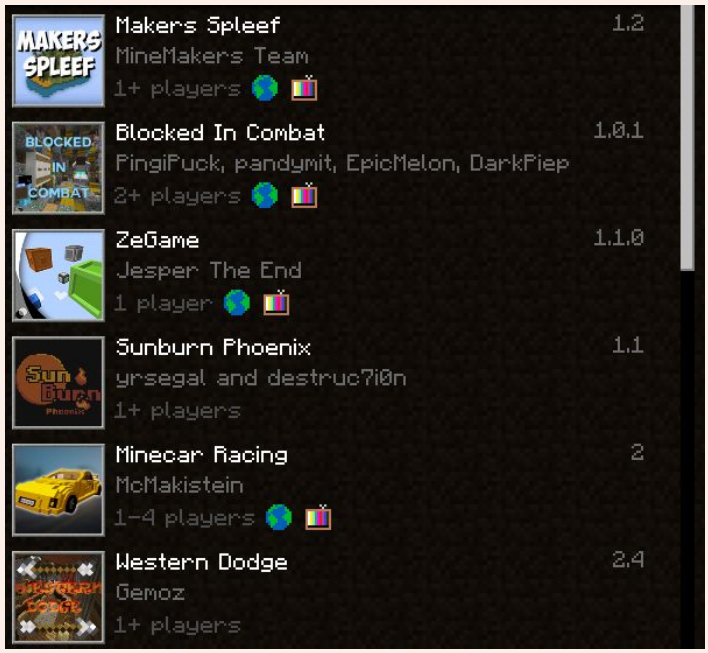
CocoaMix is a Canadian map maker who specializes in minigames and “quirky” mechanics of command blocks. He has worked on games such as Nuked, Quick Colors, and Stack Masters, as well as the Easter Egg hunt module showcased by Logdotzip. In his spare Minecraft time, he likes to mess around with command blocks and try new things, resulting in the discoveries of a couple quirks with command blocks, such as Reverse Order Activation and Toggleable Conditional Clocks.

Mini games and Minecraft PC Realms

In an issue about Minigames, we have to mention Mojang's Realms. Minecraft Realms is Mojang's own version of server hosting; with a few additional features not available on standard servers. One of these features is the Minigames.

Minigames on Realms gives Realm owners the ability to temporarily replace their world with a Mini Game, and later return to their original world. The part of this that is of interest to Map Makers is where the mini games come from. Marc Watson (@Marc_IRL on twitter) is the "Minecraft Realms Content Manager" responsible to selecting the mini games from the community of Map Makers themselves! None of the current mini games available are developed by Mojang.

Submitting your mini game for consideration to be included on Realms may be one way to get your map played. But not all mini games submitted get used: Mojang has a very high standard for what they'll accept. We don't say this to put you off. Quite the opposite. If you want to make a "good" mini game, then considering the requirements set out by Watson is a great way to benchmark your game against others.



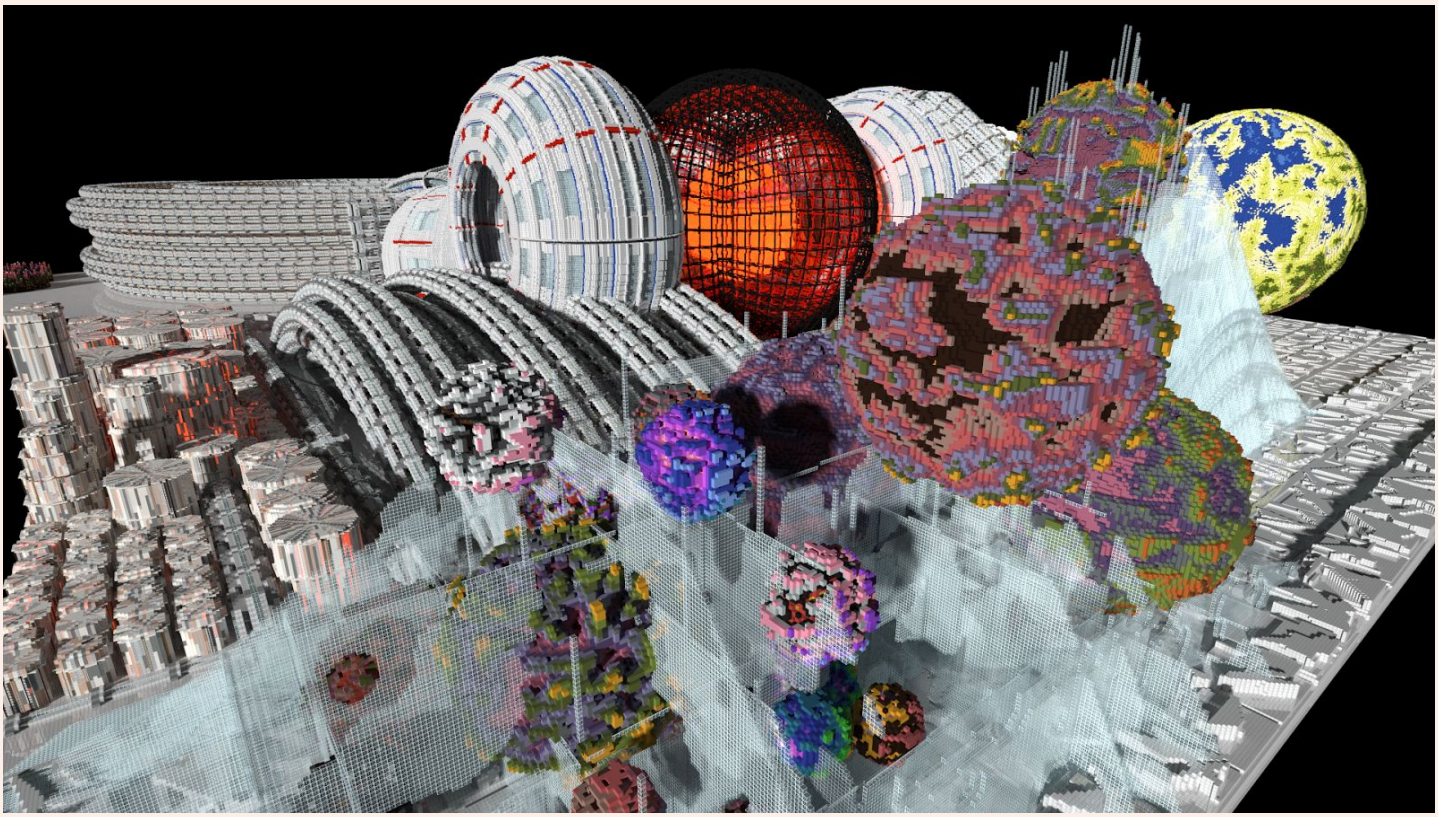
Watson accepts submissions via the /r/realms Reddit sub-reddit, and there is a pinned post about how to submit your map, but Mojang's Atlas website is where you'll find the guidelines. Map makers can also submit adventure maps and survival spawn maps to Realms. Some of these guidelines also apply to mini games in general, whether you plan on submitting your map to Realms or not. For example; "Your lobby should be clean and clear, and have a way to start the game quickly. Additional options are fine, but shouldn't slow down the ability to jump into the game too much." Good lobby design applies to all maps, but in mini games the ability to get started and playing quickly is an especially important feature. Nobody wants to spend 5 minutes figuring out the settings to play a 4 minute game.

Have a read through the Realms guidelines; it may just help you make your game better!

- Gnasp (www.gnasp.com)

Making a Map: Managing the Project

Every Map Making project starts with an idea and, if all goes well, results in a finished product. You can control how quickly this happens, and how closely the finished product is to the original idea. You do this by choosing the right tools and processes to manage the flow of work. As you run a project you will learn more about tools and your skills. Over time you will find ways to improve how you work if you pay attention to the available options and select the best one for the task at hand.



Plan, Task, Owner, Status, Duration

If you are assembling a team to create a map then you should consider what approach you may use to manage the construction of your map. A ***Plan*** shared between the team members can be broken into ***Tasks*** and these tasks can be assigned to ***Owners***. As the construction of the map proceeds the Tasks can change ***Status*** from ***Planned*** to ***Underway*** to ***Complete***. Each Task can also be discussed with the owner of the task to understand how long it should take, and you can both track progress against this estimate. This is called the ***Duration*** and it can be measured in terms of effort, hours, or by a due date. If it looks like the Task may be taking longer than you originally thought then this gives you an opportunity to ask why: has something gone wrong? Are there bugs that need to be addressed? Did we not understand clearly what was required? Is someone overloaded with too much work?

Whether you are a one-man Map Maker or a team of talented people, software tools can make tracking to Plan easier than keeping it all in your head. Some current tools you may find useful include:

- Asana (https://asana.com/) for Task Management
- Jira (https://www.atlassian.com/software/jira) is used by Mojang for bug tracking and assigning people to fix bugs. You can also use it to manage an entire set of projects all the way through to test and packaging of your map.
- Trello (https://trello.com/) provides collaboration tools with some very impressive interfaces for visually managing work.
- Spreadsheets can be used as flexible databases and planning tools
- Conference tools like Skype, Discord, Twitter messaging, and Google Hangouts (allows screen sharing) are useful to connect the team and stakeholders throughout the project:

That covers HOW you will run your project, though as you read more about “Project Management” you will find there are many approaches and frameworks that can be used.

Stakeholder Requirements

Many Minecraft map projects start with a ***Requirement***. This can be as simple or detailed as you like, though a vague requirement can leave a lot of wiggle-room when it comes to deciding whether the map has been built 'correctly'. There is usually one person, or group, who is driving the requirement and it may not be you! This person is your key **Stakeholder** in delivering the map and so it is important that you listen to what they have to say, and help them through the process by keeping them updated on progress and on what they can expect in terms of timing and result. Sometimes a too-detailed requirement can leave you with some very difficult decisions at implementation time when the game is not behaving as expected. It is good to strike a balance and to describe a requirement that explains the outcome you expect. An example may be:

> **Requirement:** I want a Player-vs-Player Arena map with custom weapons that you can buy in-game using gems you earn by camping on capture points.

There are a number of details missing here which can be essential to the gameplay. For instance, how big should the play area be? Is there a certain style for the map, such as 'tunnels' or 'ruined city'? What abilities are there for each of the weapons, how many weapons can the players use when the game starts, and what is the cost of each of the upgrades the players can buy during each round of the game? How long should each round take? Do the players select a kit in the lobby before starting? These questions can be addressed in ***Design***.

Design

The Design process for a game can be as fluid as you like. You can return to expand on details over time, though it is important to remember that you cannot really plan your build at the Task level unless you have a design to work with.

You can work out what pieces of the map need to be built, and in what order, using the design and a few people who have a good understanding of the technology. A Command Block person may give you really good advice on how long the game mechanics will take to develop, based on how they interpret the design.

Using pictures and diagrams can help explain the design. Other ways to explain the design include using YouTube videos of other games and maps or even using ideas from movies! Legendary Star Wars IV Director George Lucas initially placed footage of World War 2 airplane battles in the movie, before the special effects team finished, to explain his vision for the Death Star dogfights.

When you break the Design down into pieces of work you are building the Plan we discussed earlier. You can start to think about what tools and technology will be the best way to accomplish each task, and who is the best person to work on each task. In this issue we share details and download information for some of the best free map making tools, so remember to read up on them.

Some commercial projects may require commercial licences for software and so it is important that you understand any cost options in your plan and communicate them with your stakeholders before you make final decisions on the plan.

Sometimes you may give a task to a person who does not yet have all the skills yet to complete the task. This can be an important way to improve team performance, and so you may also need to consider how you make sure they have enough help to be successful in their task.

Late delivery can be a real problem for a project and so you need to keep in mind what risks there are to delivery timing and quality and how you will address them.

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W
1	ARISE! Custom Minecraft Adventure Map	http://minecra	Landscape	Spawn	Village	Waterfall	Mountain	Beach	Coal Mine	Caves	Spider Lair	Tunnel	Dragon Roost	Tower - Gatehouse and hall	Tower - Lab	Tower - Dungeon	Tower - Residences	Tower - Barracks	Tower - Observatory	Tower - Vault	Tower - Conservatory	Tower - Library	Prologue
2		http://www.rec																					
3	Build Environment		100%	100%	80%	50%	20%	80%	0%	0%	80%	80%	0%	80%	80%	80%	80%	80%	80%	NA	80%	NA	0%
4	NPCs		NA	80%	10%	NA	0%	50%	0%	0%	80%	0%	0%	50%	0%	10%	0%	0%	0%	NA	0%	NA	0%
5	Events		0%	80%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	NA	0%	NA	0%
6	Mobs		NA	60%	0%	0%	0%	0%	0%	0%	80%	0%	0%	50%	0%	0%	0%	80%	0%	NA	0%	NA	0%
7	Puzzles		NA	0%	0%	0%	0%	20%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	NA	0%	NA	0%
8	Flairs		0%	50%	10%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	NA	0%	NA	0%
9	**Playability:**	17%	33%	62%	17%	10%	3%	25%	0%	0%	40%	13%	0%	30%	13%	15%	13%	27%	13%		13%		0%
10	Bugs			Set Nogard's trades Bucket	NPCs saved from tower								More dragons Volcano finale	Floor design / visuals		Save NPCs							

Designing your map by component can lead directly into a development plan and detailed progress tracker

Build and Test

The best plans deliver as much as possible early enough to allow for feedback and course correction if changes need to be made. Wherever possible your plan should allow for lots of things to be worked on at the same time by different people. This is not always possible. For example, a script may be needed before anyone can start their work building the environment, game mechanics, models, and custom sounds. In this case you would make sure you don't start any of these other tasks until the script has been approved, otherwise you may waste time, energy, and risk introducing bugs while making changes to something that should have been built once correctly.

The earlier you can assemble a working prototype the better. Not everything has to be in place for the map to be play tested. Valuable feedback is possible while there is still time to act on it. Feedback that comes late, or after all the work has been done, may delay the project or result in a low quality map.

If you have people involved in your project from more than one country you will need to consider time zones and whether everyone understands what you mean when scheduling calls, conferences, and due dates. There are some services which allow for automatic time zone conversion. Try http://www.timeanddate.com/countdown/create.

Testing should be done as often as practical. To achieve this you need to be able to quickly build the map into a runnable version. This may require you to make sure project components are maintained in a version control system, like GitHub, or even in a shared location such as Google Drive or DropBox. Another advantage of using a central repository is the ability to recover from computer failure or other resource disaster without loss of information.

Release

When you deliver a working map you need to consider hosting options, installation instructions, marketing the map, and ongoing support for niggly bugs that have made it into the map.

Map hosting of a public release needs to cater for anyone on the Internet accessing the map. Hosting can be simply through a file hosting site like Google Drive, or Dropbox. Dedicated community sites exist that let you build a page for your map. These include www.Planetminecraft.com, www.Curse.com , and www.MinecraftMaps.com. Another hosting option is to host your map on Mojang Realms. Realms submissions are through http://www.reddit.com/r/Realms.

Statistics are important to collect so you can understand what people are doing with your map. You may need to discuss how effective your work has been to the person who requested the map. Some sites show download statistics, others do not. A bit.ly link may give you access to statistics you may otherwise not have access to. Be careful though - the person creating the link is the only one with access to statistics so you may create a dependency on one of your team. Also some people do not like to click URL redirects and so the download performance of your map may be impacted. This can also be avoided using a site such as PlanetMinecraft or Curse, both of which track downloads automatically and publicly.

Installation of maps is reasonably standard and does not need to change from map to map. You may need to explain some basic concepts of working with a file system however if your audience is young (or old!) so consider creating or linking to a YouTube video to make this easier for your audience, such as this one: https://www.youtube.com/watch?v=M0yaUYrXlbA

Marketing your map can be hit and miss unless you have specialist skills or access to them. A polished trailer for your map showing the exciting gameplay and linking to a hosting page can entice people to check it out. Having someone popular in the community play your map can help drive uptake of your map, if that is important to you. Many maps achieve fame through their innovative gameplay and emerge from the technical map community who recognise the work as unique. Whatever method you follow, try to keep the information on your map clear and uncluttered. Focus on gameplay and make sure you have tested thoroughly to make playing the game as simple as possible.

Sometimes you or your audience will have problems with your map shortly after release. It is important to be able to respond quickly, which may mean you need your team on hand to review and respond. Remember to increase the version number if you re-release a new version of the map so you can be clear about which copy of the map someone is using. Check the comments where you have posted the map and don't take bug reports as a criticism. It is important to maintain a calm attitude when dealing with people you don’t know.

You may feel justified in snapping back at someone who has found a problem with your map because you spent so long working on it. You certainly didn't plan for it to have a problem. But things happen. People don't have to play your map and so they won't necessarily understand your position. They can easily move on to the next map so your goal needs to be to keep them engaged and interested in your work. Stay calm. Deal with the issue and move on. Every person reaching out to you for help with a problem is an opportunity for you to develop a loyal follower of your work.

Feedback on your creative work comes in many forms. It is always great to get positive notes from the community. Sometimes the best advice can be hard to listen to. When someone offers you suggestions on what could be improved think about how you might do things differently next time for a better result. Even when you are at the peak of your craft there will be new audiences and changing markets for you to pursue. Creativity is a shared journey when it is done well.

> *However you manage your map project, the most important thing is to **finish the project** and to keep learning about what can be done better next time. Let **Map**^{Mag} know what you do to make sure your map projects run smoothly via Twitter **@MapMakingMag**.*

Map Making Tools

Map Makers are supported by a massive number of tools built by the community to allow extensive customisation of the game. Here are a few essentials:

The Minecraft Wiki

Minecraft does not come with an instruction manual, and so the community has stepped up over the years to write its own. There are a number of Wikis that describe various aspects of the Minecraft game and one stands out for current information on the sorts of things that Map Makers need to use: http://minecraft.gamepedia.com/Chunk_format

Minecraft game objects are described by different data structures depending on the object types, and command formats can be inconsistent with various issues introduced over time for historical reasons. The Wiki is a web site that you can access and read when you are online.

@theqmagnet's Test Map for Resource Packs and Map Makers

Minecraft provides a rich feature set for making your own games, however not all of the features can be selected easily from a palette of objects in game. @theqmagnet has created a master map of all the in game blocks, items, and other features like sounds and particles. This map is available from: https://mods.curse.com/worlds/minecraft/qmagnets-test-map-for-resource-packs-and-map

This map is like a laboratory of all the intricate and advanced tricks that are possible in the game. Once you load it up you then walk through the map to use it, pressing the occasional button to change options and displays. There are multiple areas in the world to explore including a wall of all the ingame blocks that you can use to quickly check how any new textures you create in a custom resource pack work together.

You can also enter the Debug world in your Minecraft client to test resource packs with all block states. See http://minecraft.gamepedia.com/Debug_mode for instructions.

World Edit

World Edit is an open source in-game tool, like a paint program, that provides enhanced tools to work with regions of space and areas of blocks while you are running around within the world. It supports external scripting, giving access to extensions that implement procedural algorithms to warp and change the world around the player. The tools supports importing and exporting model files called Minecraft Schematics. The program is implemented as a Mod. You can download WorldEdit and review the feature set in more detail here: http://wiki.sk89q.com/wiki/WorldEdit

WorldPainter

WorldPainter is an open source dedicated tool, like a paint tool, that is used to sculpt the world and populate it with ores, trees, and other structures. You work outside the game using brushes and selecting generation options. You can download WorldPainter here: http://www.worldpainter.net/

MCEdit

MCEdit is an open source dedicated tool that loosely resembles a 3D editing Computer Aided Design package. It provides an interface that is similar to your in-game experience for flying through the world you are editing. The tool supports importing and exporting model files called Minecraft Schematics. It includes tools to manage Entities and Block Entity data. Custom extensions are provided by the community through an interface known as a "Filter". MCEdit is used for both large and small scale operations and can be successful when other tools struggle for resources to complete a task. MCEdit can be downloaded through: http://www.mecedit.net

Version 1 of MCEdit is maintained by the community as "MCEdit Unified" and is currently recommended for all tasks. Version 2 is in early development by @CodeWarrior0 who values your feedback.

NBT Explorer

NBTExplorer, by Justin Aquadro, provides a collapsable-tree way to explore and edit the data structures that make up Tile Entities and Entities in a Minecraft World Save. It pre-dates the /entitydata and /blockdata commands, and still remains useful because of its powerful search function and the ease with which you can navigate to investigate a desired structure. It may be downloaded here: http://www.minecraftforum.net/forums/mapping-and-modding/minecraft-tools/1262665-nbtexplorer-nbt-editor-for-windows-and-mac

Map Making Websites

MCStacker

mcstacker.bimbimma.com *(toggelable 1.8/1.9+)*

A powerful website that provides a generator for a lot of commands with very in-depth customisation possibilities as well as the possibility to save and modify your created commands.

Most notably is the /summon command, which is probably the origin of the sites name, since you are able to stack (custom) mobs on top of each other, with easy management of the order, the only limit being the amount of characters a command can be at maximum.

MinecraftJSON

minecraftjson.com *(uses strict JSON)*
A website that is dedicated to generate anything JSON related, from colorful, clickable `/tellraw` commands with hovering Text to Books and Signs with similar awesome attributes.

Command Combiner

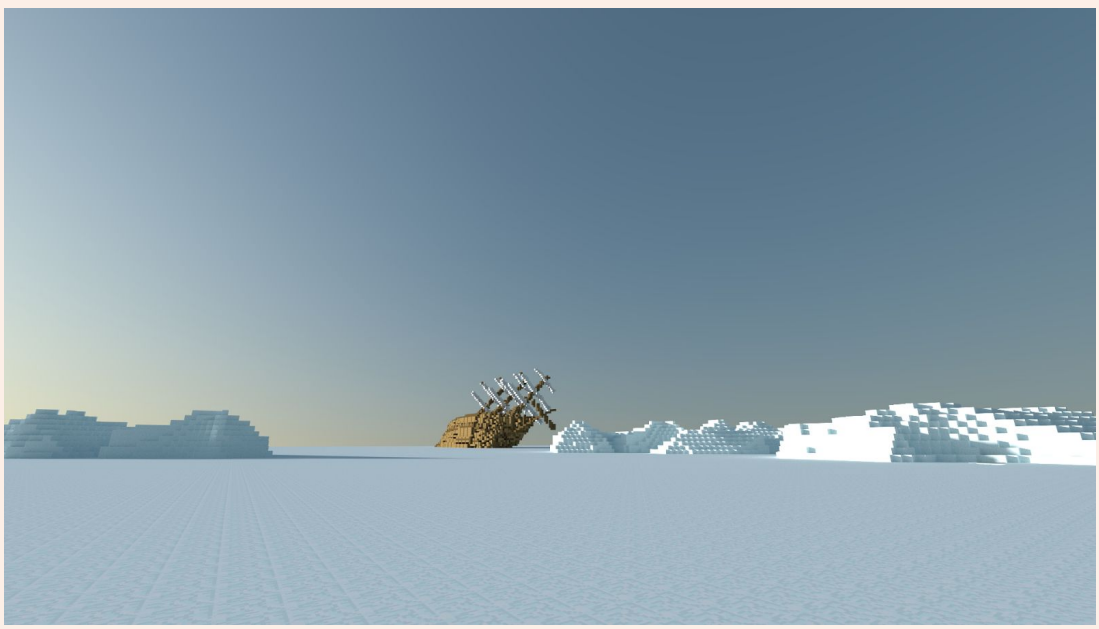
MrGaretto's (@JustMrGarretto) Command-Combiner https://mrgarretto.com/commanddump *(1.9+, a 1.8 version also exists on his website)*
This website allows you to paste your commands in a textbox and it will generate a one-command for you. It has several options to choose from, including boxing in your CommandBlocks, selecting the size of the box, adding clickable signs as well as multiple customizable methods for the commands and some prebuilt functions.

Cubical.xyz

http://cubical.xyz/ is an online Minecraft schematic editing suite created by @inhaze. It is similar to MCEdit Unified and with some additional features including terrain generation through a Perlin noise based function and text generation. No client installation is required as the tool runs within your browser. Schematics you build in cubical.xyz can be loaded into all other tools where the .schematic file format is recognised.

BlockModels.com

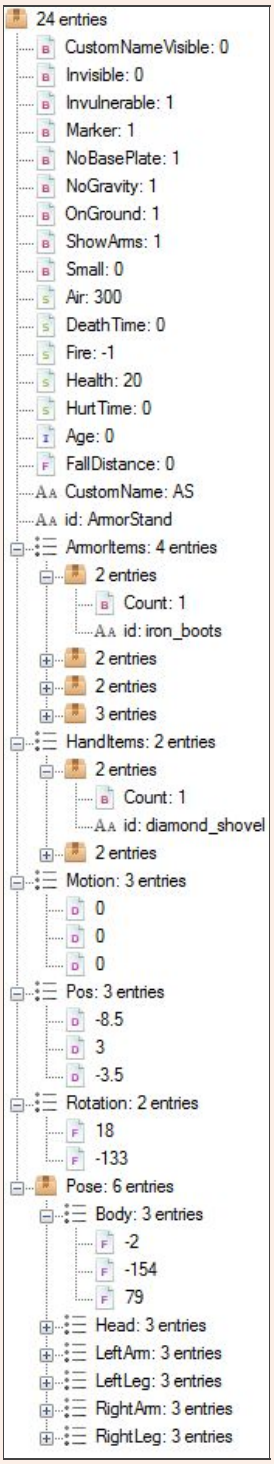
https://blockmodels.com/ is a custom model hosting site that includes support for online viewing of models in the library. Models are organised by category and are provided by the community. At the time of writing there are 56 unique models across six categories, including Food, Weapons, Decor Furniture and Animals. This month @TheDestruc7i0n has updated https://blockmodels.com/ to include a viewer for animated models and models with multiple textures.



In Depth: ArmorStands

Minecraft is often described as a game about placing and destroying blocks. That description falls short of the rich platform that is available to players of the game. As well as blocks, Minecraft also includes *Entities*, which are objects that can move around off the block grid. For the older readers, Entities are 'like' game Sprites in that they can be controlled as a unit. You can create and position an Entity, assign properties and behaviours, and destroy them when you are done. Examples of Entities include the ever-present mobs like Zombies and Creepers, the various Minecarts, and also the very weird and useful ArmorStand.

An ArmorStand is an object that can hold and display Items. Items include things like weapons and armour, which explains the name. In Survival you can create an ArmorStand and place it in your base to show off your best equipment, keeping it safe so it is not lost when you fall in the lava. Right clicking the right spot on an ArmorStand while holding something will transfer the item to the ArmorStand, and repeating the action retrieves the item.



Underneath the 3D in-world ArmorStand Entity is a complex structure with very interesting tags that you can use to control the behaviour of the ArmorStand. The information in the picture on the left is a copy of an ArmorStand's NBT tags captured using a tool called NBTExplorer *(see our article on Map Maker Tools this issue - Ed)*. ArmorStands include useful properties that Map Makers can use to do some pretty neat things. The tags are discussed in detail on the Wiki *(see the same article, no really, it's that good! - Ed)* and we will take a closer look at a few of them now.

Making a Marker

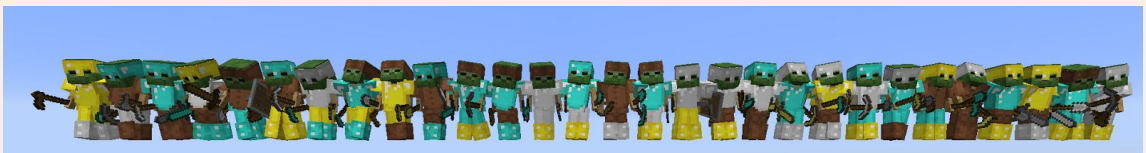
ArmorStands can be used as placeholders on your map. This lets you detect players moving through particular regions of the map, or you can spawn items at the ArmorStand without knowing where it is by coordinates. You can move the stand around during development and in the game, and the game will allow you to target the ArmorStand in Command selectors.

To create a Marker ArmorStand at the current location of the player:

```
/summon ArmorStand ~ ~ ~ {Marker:1,Invisible:1,NoGravity:1,CustomName:myMarker}
```

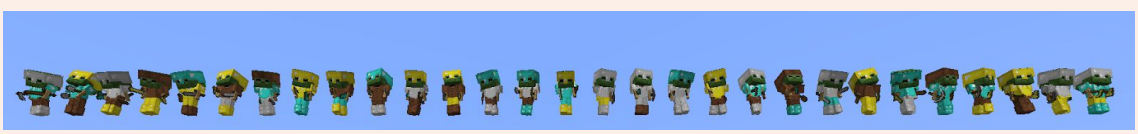
- The *Marker* tag tells Minecraft to reduce the Hitbox of the ArmorStand to a point. This reduces the complexity of interactions of the ArmorStand with the player and other Entities in the game and so reduces the computational load on the game. If the *CustomNameVisible* tag is **1** then the name will be shown at the feet of the ArmorStand.
- The *Invisible* tag tells Minecraft only to render the ArmorStand to a player in Spectator mode.
- The *NoGravity* tag tells Minecraft to stop checking to see if the ArmorStand needs to fall.
- The *CustomName* tag tells Minecraft what to label this newly created ArmorStand. You can then use this CustomName in selectors to target this instance of a Stand you have spawned in and named. An example is: `/execute @e[name=myMarker] ~ ~ ~ setblock ~ ~ ~ stone 2`

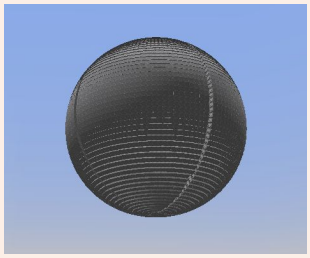
Non-Player Characters (NPCs)



ArmorStands can be very effective static NPCs in a map, or can be animated using the **/entitydata** command to adjust the pose of the ArmorStand in real time. The characters above are each made from two stands at the same spot and pose.

- The *Small* Tag controls whether the ArmorStand is Player height (when set to 0) or Baby height (when set to 1). For example: `/entitydata @e[type=ArmorStand] {Small:1}` toggles all ArmorStands to Small size.
- The *ShowArms* Tag displays the ArmorStand arms as sticks on the model or hides them, which is the default. NPCs benefit from arms so set this to 1. For example: `/entitydata @r[type=ArmorStand] {ShowArms:1}` toggles a randomly selected ArmorStand to show the arms.
- The base plate may be disabled as follows: `/entitydata @e[type=ArmorStand,r=10] {NoBasePlate:1}` which hides the Base Plate for any ArmorStand within 10 blocks of the Player executing the command.





The key element to an effective NPC is the equipment it carries. The equipment and pose can make a convincing character or ruin the experience, so consider your options. Equipment is specified in the NBT using the following format:

- The Tags *HandItems* and *ArmorItems* are used to specify items in particular inventory slots on the ArmorStand. In the following example an ArmorStand is created with a Diamond Sword in each hand, Diamond Armour, and a Command Block on the head:
  `/summon ArmorStand ~ ~1 ~ {NoGravity:1, NoBasePlate:1, ShowArms:1, HandItems:[{id:diamond_sword,Count:1}, {id:diamond_sword,Count:1}], ArmorItems:[{id:diamond_boots,Count:1}, {id:diamond_leggings,Count:1}, {id:diamond_chestplate,Count:1}, {id:command_block,Count:1}]}`
- The Pose is specified using a set of rotations around the pivot joints on the ArmorStand:
  `/entitydata @e[type=ArmorStand,r=10] {Pose: {RightArm: [10f,20f,30f], Head:[10f,10f,10f], LeftArm:[10f,10f,0f], Body:[10f,10f,0f], LeftLeg:[10f,10f,0f]}}`
- Blocks can be held by the ArmorStand. This modifies the block size to be proportional to the ArmorStand size. In this way you can create builds using 'micro blocks'. The sphere shown to the left is made from Invisible ArmorStands wearing Blocks on their heads, and each block is rotated to the right place to smooth out the surface as much as possible.

Twitter user @Wergat has discovered ArmorStands can be located within a Villager mob so that you can make it look as though the Villager is carrying items! The picture shown to the right is provided by @Wergat and shows what can be achieved. Start a conversation with @Wergat to find out more!

Performance

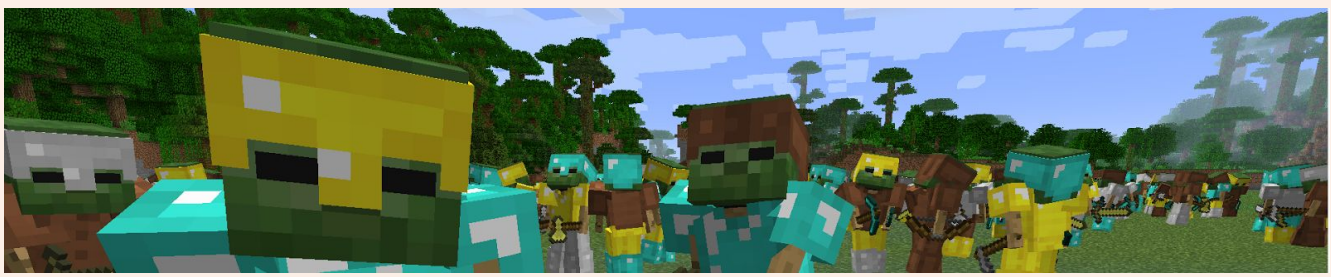
ArmorStands (AS) can slow the game if used in high numbers. @qwertyuiopthepie and @skylinerw point out that there is a higher performance marker Entity called an AreaEffectCloud (AEC) that is suitable for large scale deployments. The following information summarises the key considerations when selecting between the two Entities based on your map goals:

Downsides for ArmorStand:	**Downsides for AreaEffectCloud:**
1. Cannot hide them from spectators.	1. Requires maintenance to prevent them from despawning.
2. Targeting issues while `Marker` is set to 1.	2. Cannot hold equipment.
3. Heavy impact on performance.	3. Cannot be affected by gravity.

For example, you would not use invisible armor stands to mark locations if spectators can be in that area. You also wouldn't use an AEC when you wanted to present a piece of equipment to the player. You wouldn't use armor stands in large amounts as markers on a playfield due to the client lag players would suffer.

AreaEffectCloud

To create an AreaEffectCloud at your location in-game, issue this command: `/summon AreaEffectCloud ~ ~ ~ {Duration:2147483647}` Like most Entities the *Age* on an AreaEffectCloud increases over time and it will disappear when it reaches the *Duration*. You may choose to reset the Age from time to time using `/entitydata @e[type=AreaEffectCloud] {Duration:2147483647}` or by killing and recreating your AreaEffectClouds.



Map Exam: Nuked

> “When everything is overpowered
> Nothing is!”
> -@CrushedPixel

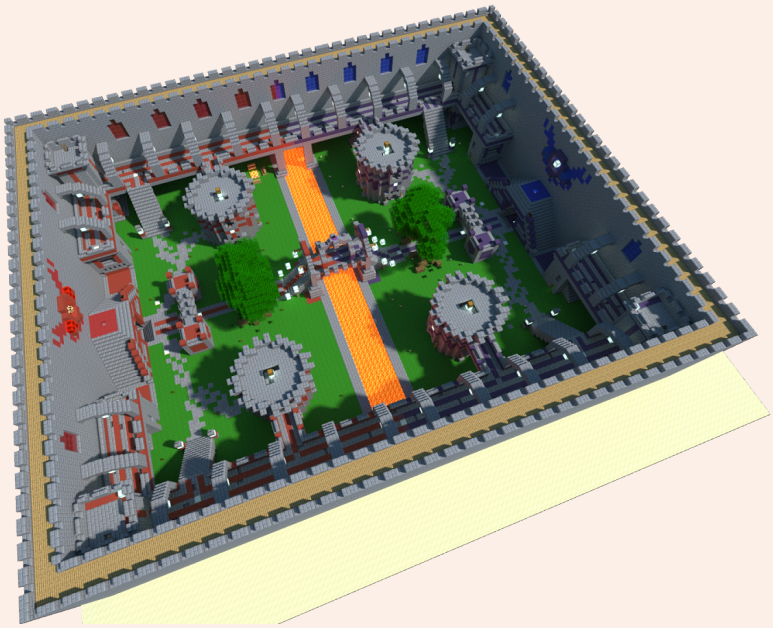
NUKED is a fast paced v1.8 compatible chaotic combat game for two teams where each team has to capture and hold the other team’s base while being blown up with a variety of creative custom weaponry. It is a type of “Capture the Point” game. The player can place traps around the map which are triggered by proximity. This allows for complex (and fun) combat. Air support, in the form of Ghasts, can be triggered, as well of a range of explosive mines, and the ultimate nuclear surprise.

The capture points are above the defending team’s spawn point. This makes for a frantic end game and balances the defending team’s new spawn equipment kits against the attacking team’s superior inventory.

The Build

The map is symmetrical, mirrored around a fiery moat of lava that bisects the battle zone evenly. Cover is provided by a small fort slightly ahead of spawn and a sentinel tree. In the middle of the map a bridge forms both a choke point and a timer-based power up location. There are raised paths that cross the lava along the walls that run the length of the arena. On these raised paths there are ribs of stone that provide partial cover from arrow and airstrikes, but camping is discouraged by the comparatively open access from all parts of the map.

A game lasts 10 to 30 minutes, depending on team size and the rate at which teams accumulate points.



The Map Maker Explains:

When McLabs15 and I created **NUKED**, we knew it wasn't going to be an overly strategic game. We created it with fun and chaos in mind. We wanted to create an environment where even beginners felt that they were awesome. We purposefully created the game to be non-competitive, like no complicated mechanics, next to no choke points or areas to hide. By not including these types of things, we broadened our player field to include everyone that wanted to experience the chaos that is **NUKED**. Our vision was to create a game where people could have a blast (pun may be intended) no matter their skill level, and that's exactly what we ended up with.

We hope you have fun playing our game!


~CocoaMix86

Layout

The map is designed with the redstone and command blocks located at world spawn, and the play area outside the render distance for this zone. Map makers separate the game command blocks from the play area to minimise lag. Minecraft sends packets of data across the network to clients based on changes to the world blocks. A rapidly clocking set of command blocks can cause massive amounts of traffic in certain circumstances, but not if the affected chunks are out of the way of all players.

On joining the world each player is detected within a spawn box, then is teleported to the game lobby near the battle zone where team selection takes place and the game is then started.

Teams

The map uses 'teams' to sort players and take action on them based on which team they are placed in. The teams are:

1. **lobby -** any new player is added to this team. Players between game sessions are added to this team: **/scoreboard teams join lobby @a[score_age=3]**
2. **red** or **blue** - these are the combat teams.

Speed Pads

There are speed pads in the map and these are triggered if a player passes over them by giving a speed effect for a short period (**execute @a ~ ~ ~ detect ~ ~ ~ heavy_weighted_pressure_plate -1 execute @p ~ ~ ~ detect ~ ~-1 ~ gold_block -1 effect @p 1 3 8 true**)

Traps

The key unique mechanic in the map is the proximity detection for custom traps. This is implemented by sending execute commands from entities marked with scores (**scoreboard players add @e[type=Enderman,score_age_min=1,score_age=1] explode 5**) to be proximity mines to the player, and then back to be blown up. Depending on the mine, the radius of detection changes in size. A normal mine detection usually happens in 2 steps: Execute an execute command at a player within the detection radius, and then execute a summon command back at the entity as the origin of the explosion. The detection and explosion happens within the same tick, so timing is key is making it look real and not lag.

Organising the Code

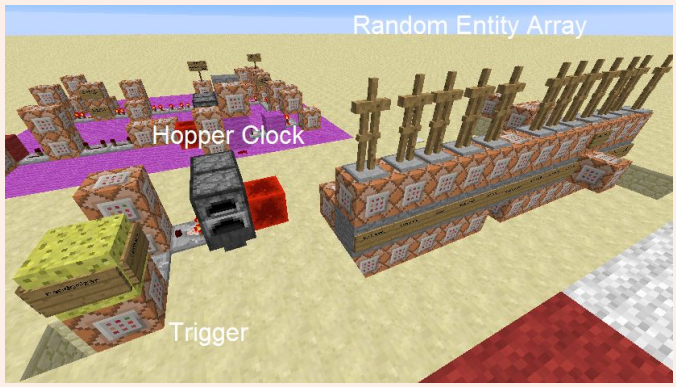
To better manage the code, Command Blocks are grouped by function using a wool-and-carpet system. Wool covered with a sign provides the key to understand which collection of blocks implement which feature.

The command blocks responsible for each feature have Carpet blocks on top or wool blocks of the appropriate colour.

In this way the development team can quickly locate where in the sea of orange blocks they need to look when implementing or debugging a particular part of the map mechanics.

Randomisation

Generating random events is important in gaming. This game uses a randomiser based on the @r selector and a number of pre-created ArmorStand entities. A Hopper clock timer is controlled using the TransferCooldown method (**blockdata ~2 ~ ~ {TransferCooldown:300}**). This clock randomly chooses an ArmorStand of a certain type and creates a redstone block offset from it to trigger a set of commands to fire (**execute @r[type=ArmorStand,name=random] ~ ~ ~ setblock ~ ~-2 ~ redstone_block**). In this way the power ups are introduced into the map at intervals during each battle. The clock is enabled when the game starts, and disabled otherwise, by passing a redstone signal to one of the Hoppers.



Another method of randomisation that was available in 1.8 was to dispense a random command block which would then execute. This feature has been removed in later versions of Minecraft as it was considered a security vulnerability to have Dispensers place executable Command Blocks. Now Map Makers must use other methods of triggering random events.

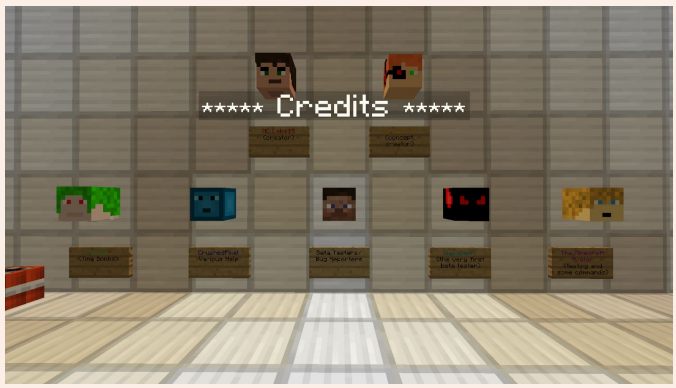
On a related note, it is quite common for Map Makers to implement a random event system using the *spreadplayers* command. This command can be executed against a non-Player entity which is landing on pressure-plates over Command Blocks which are weakly powered by the plates. Spreadplayers has the added advantage of reliably forcing a chunk to be loaded in some versions of the game.

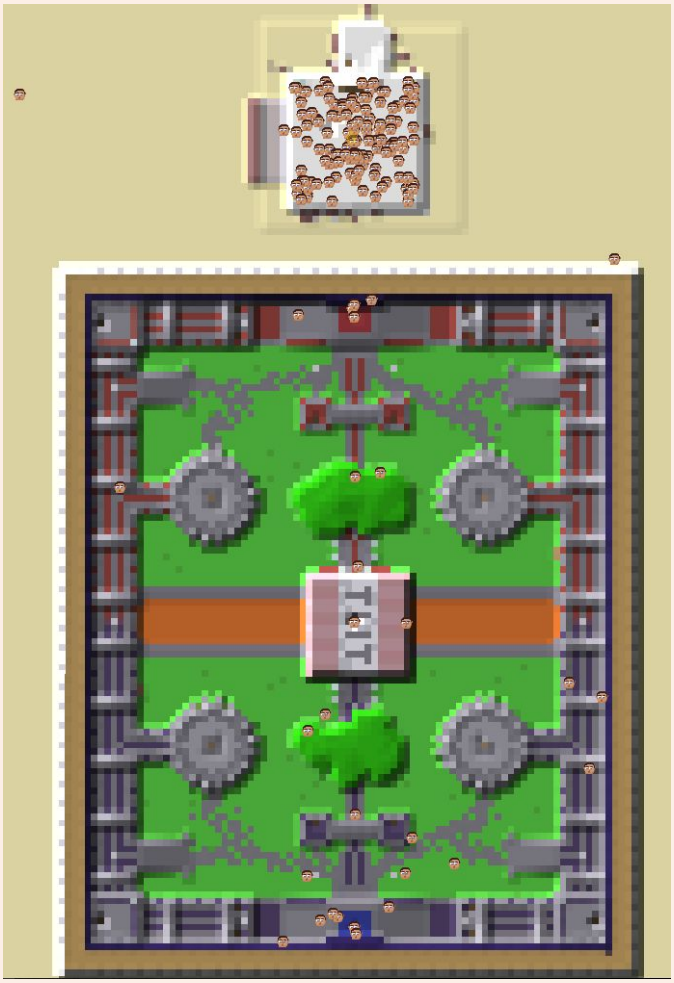
Credits

A wall in the lobby displays the team members who created and supported the map through testing, or supplying ideas and help. You can include player heads on your own map by using a command similar to this one:

- **/give @p minecraft:skull 1 3 {SkullOwner:abrightmoore}**

Swap the name ‘abrightmoore’ in that command to some other in-game name for a copy of that player’s head.





Testing

The picture to the left shows the last logout location of players on the map. Each head is a player. The high volume of players shows how extensively this map was tested before release. Testing goals include ensuring the weaponry is balanced, the map design meets the gameplay objectives, and that there is an awesome trailer to post once the map is complete. You can check out the **Nuked** trailer here and download the map using the link in the description of the video: https://www.youtube.com/watch?v=n-a7Jp1XKB0

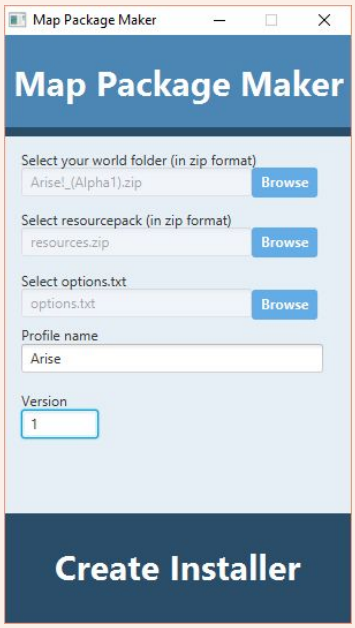
Nuked is just one of many hundreds of high quality mini-games built in Minecraft and available for download. See if you can find a map type you like, and then spend some time taking a closer look at how the map author has constructed the redstone and command blocks that power the game!

Did you miss Issue One?

Never fear! **Map**[Mag] is available now as a free digital download from http://testfordev.com/mapmag/

Tech News

We round up all the neat news from map makers around the world into a digest you can use to make better projects.



MapPackageMaker by MrJVS

This is a tool under development by MrJVS. MapPackageMaker compiles your map into an installable package with options, version and Resource Pack support. It is intended for use with the Minecraft client. You need to have Java installed on your system to use this tool. You can use this method when you prepare your Map for distribution. When you run the tool it asks you to locate the level.dat for your package, select the resource pack to be included, include the options.txt, set up the client profile name and Minecraft client version for the map to work with.

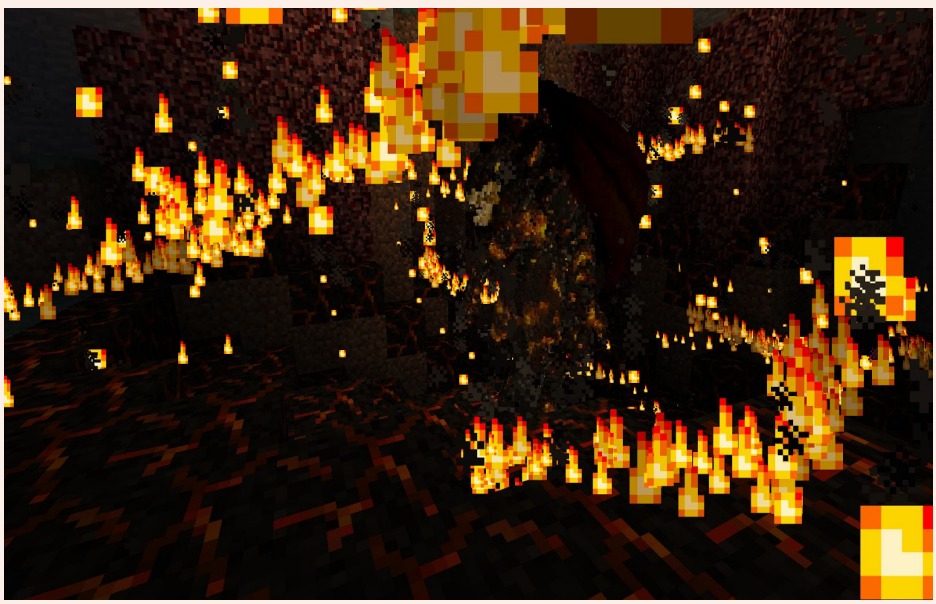
Once you have created the installer a .jar file is produced. You can then share your map through a file sharing site. Your audience can then run the installer (using their java installation) to place all the map components in the right place and generate a new Minecraft client profile. Be careful naming your map as some characters are invalid characters in file names on some systems and this may cause problems with the installer.

You can download MrJVS' MapPackageMaker tool here: https://github.com/mrjvs/MapPackageMaker/releases

On a related note, Reddit User DavidRO99 has also recently created an Apple Script based Mac map installer: https://redd.it/4vcgak

Balrog Boss by @official_sqored

Custom mobs are fun. Simple tweaks to zombies can go a long way to making your map special in the eyes of a Player. Though could it be possible to go… too far? With apologies for spoilers: In J.R.R.Tolkein's great popular work The Lord of the Rings, the Fellowship tries to take a shortcut to their destination by travelling through what they think are the abandoned halls of the ancient Dwarven kingdom of Moria.



Through an unfortunate mistake, the group are attacked by Goblins and are forced to attempt escape. They discover, to their peril, that the threat is worse than mere Goblins as a great darkness chases them through the labyrinth, eventually forcing Gandalf to sacrifice himself to buy the rest of the heroes time to scramble back to the surface. This monster, called a Balrog, has been recreated by @official_sqored in Minecraft.

Some of the features of this monster include:

- Custom Models
- Special AI and attack modes
- Particle and environmental effects

You can experience this beast, and explore the inner workings of him, by downloading the resource pack and copying the two "one-commands" from the announcement video: http://youtu.be/FMfjnbrxQVc?a

Star Wars Vehicle Models by @official_sqored

Sqored is some sort of over-achiever this **Map**[Mag] Issue with the simultaneous release of models for X-Wing, Darth Vader's Tie Fighter, R2D2 and a Speeder: https://www.youtube.com/watch?v=WAou4qBTAIw

Smooth Falling Ladders by @jragon014

@jragon014 has been actively building useful command block contraptions under his Map Mechanic series for Map Makers for some time now. Smooth Falling Ladders are an in-game animation for creating climbable ladders in your map. They can be triggered on an event in your map to allow access to new areas in the game following completion of a puzzle or an ingame event. You can view the tutorial and access the commands from Jragon's video here: https://www.youtube.com/watch?v=25kdoZH7PHg

Changing biome colour Kopasz7

Veteran Minecrafter Kopasz7 has pulled apart the grass.png file to work out which parts of this picture influence the biome colours in-game. If you are interested in customising the world biome colours check out his work here:
https://www.reddit.com/r/Minecraft/comments/4vcnpe/biome_color_changing_with_resource_pack/

Excalibur Resource Pack by /u/Maffew

Excalibur is new from the creator of the classic Chivalry Pack, /u/Maffew. It has clean lines and a fantasy feel: https://redd.it/4v85qi

Block Sampler image for Excalibur using @theqmagnet's Resource Pack Test Map, described in detail in this issue

CommandStudio by /u/Marcantouf

http://commandstudio.github.io/commandstudio/ is a Command editor that supports creating Minecraft game logic and compiling the results into a one-command installer. The result is an executable command block structure in your world which creates a library of game logic when it is run. The editor supports syntax highlighting, procedure calls, list iteration, and importing library files. The instructions are provided in an easy to read manual here: http://commandstudio.github.io/commandstudio/manual/ It is by /u/Marcantouf (YouTube: https://www.youtube.com/user/LeMarcantouf) who explains the project in this Reddit post: https://redd.it/4vzznj

MCEdit Filter to Convert Data Values to Block States by @Onnowhere_

Mojang has made a fundamental change to how block data is stored and recovered in Minecraft Maps. Soon, in v1.11 PC, ID and Data Value pairs are expected to stop working. To future-proof your maps @Onnowhere_ has created an MCEdit filter. https://redd.it/4x7dyk
Other filters are in development. Watch out for @JustMrGarretto's work over the coming weeks.

New PC Maps

MrGaretto’s Destructive Worms

Navigate your Worm through a cut-away world, chomping everything in your path! This is a fun mini game for v1.10 of Minecraft with multiple levels and challenges. Get it here: https://www.youtube.com/watch?v=cwNts5rOVxM

Hybrids by /u/Random_llama1

Hybrids is an Escape Adventure map involving gene-spliced mobs and puzzles based on this custom mechanic. A download link for the map is on the Reddit post https://redd.it/4uxjxs and the Imgur album http://imgur.com/a/54LxC

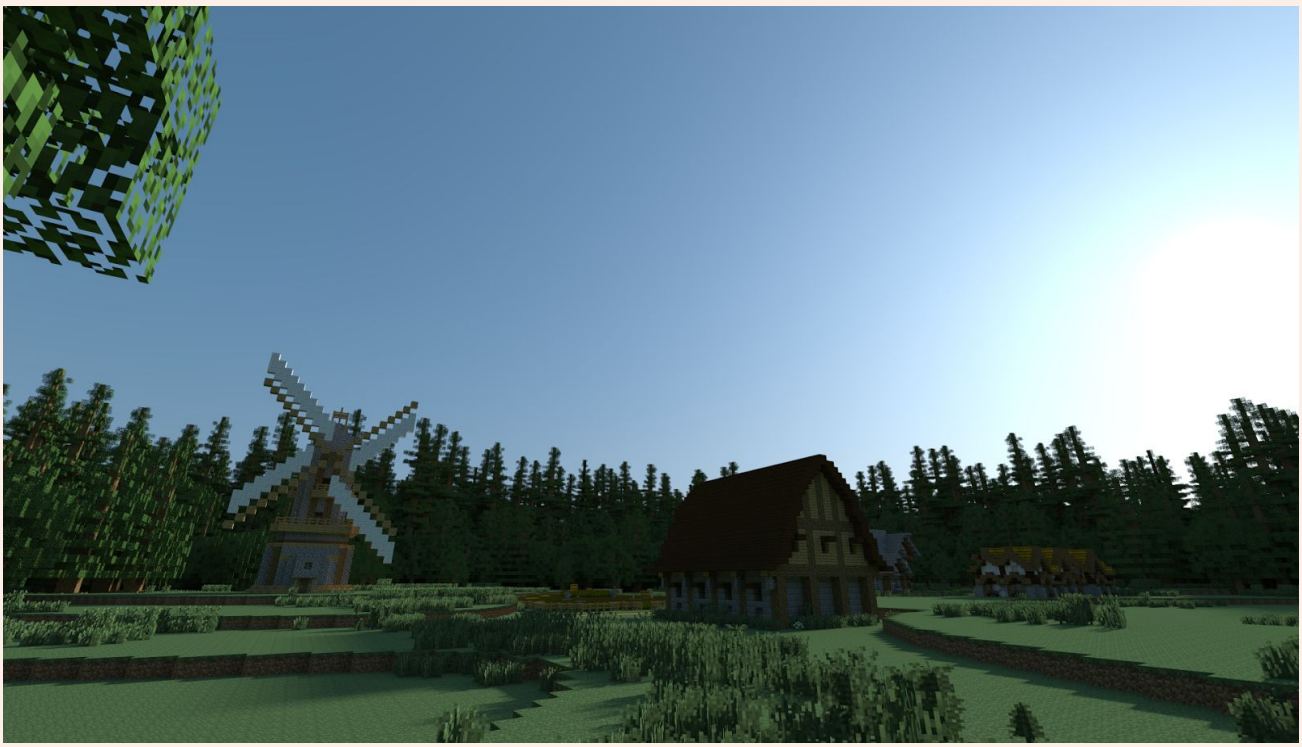
If you enjoy this map, check out the novel “The Island of Doctor Moreau” by H.G.Wells, which has similar themes.

Dream of Omniscient by Ronin Build Team

This gorgeous map is a detailed diorama in Minecraft as an homage to Minecraft builds. One intrepid player suggests playing Hide-and-Seek within it. Get it here: http://www.planetminecraft.com/project/dream-of-omniscient/

Chasing Time by Vu6

Vu6 (YouTube: https://www.youtube.com/c/vufour) has shared a Zombie virus infected Adventure map where you must travel back in time to prevent the outbreak. Get the map here: http://www.minecraftmaps.com/adventure-maps/chasing-time



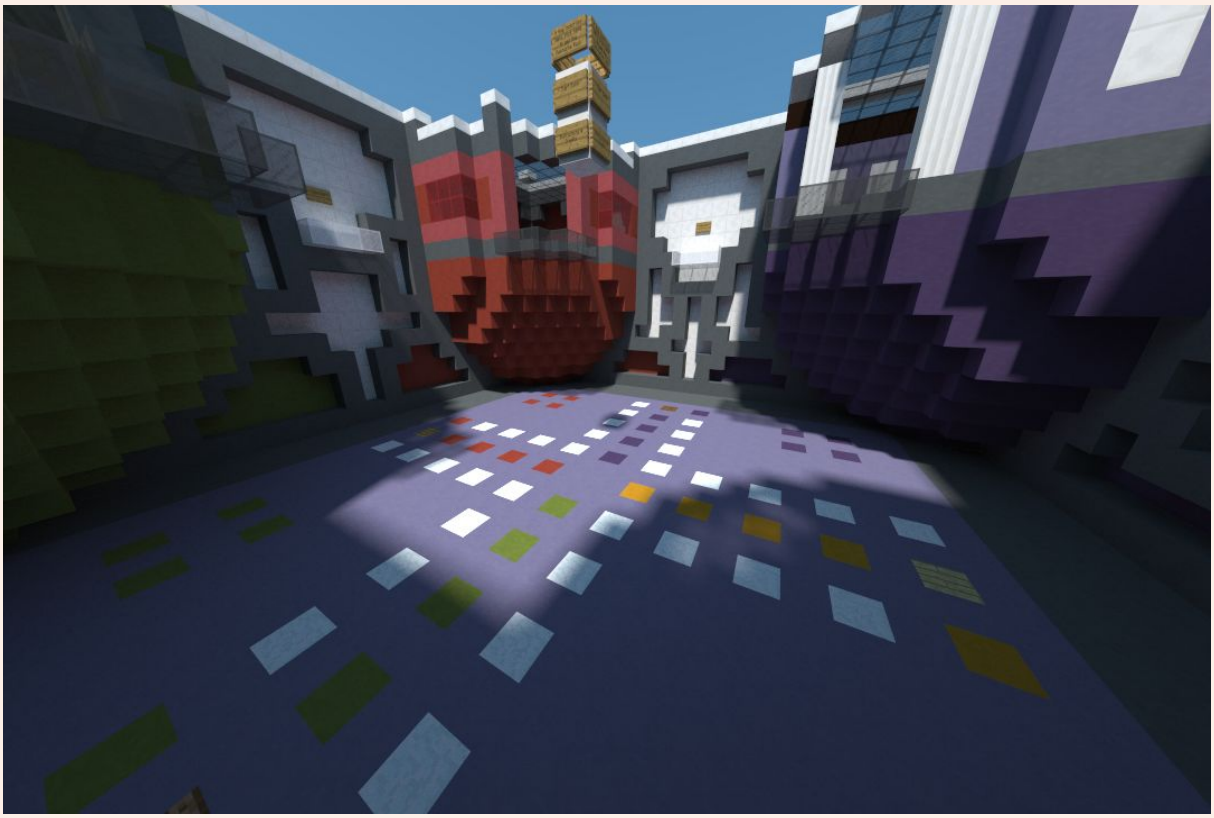
Earth by 4chan /int/

Reddit user /u/randomthrowaway385 has shared a map of the Earth surface that has been recreated at approximately 1:1400 scale.

Trees, rivers and hill features provide a fascinating explorable world. Be aware this map was originally hosted on a server so some of the content may be unpredictable. Get it from https://redd.it/4v1gqg

Ludo (Parcheesi) by Plagiatus

@realplagiatus has recreated Ludo in Minecraft using custom models and innovative gameplay elements to handle dice rolling and piece movement.



Trailer and map download: https://www.youtube.com/watch?v=s7j3wsGgAqU

Minecraft Story Mode - Redstonia by @Onnowhere_

@Onnowhere_ has recreated a key scene from the TellTale Games episodic interactive fiction game “Minecraft Story Mode”. It is an impressive technical achievement using only the Minecraft game itself.
Check out the trailer and the link to the map download: https://www.youtube.com/watch?v=On-S5eGi6nE

New Console and PE Maps

Villains by Mojang

Mojang have released a playable map, possibly to promote new downloadable content (DLC). DLC is a revenue stream for Mojang/Microsoft based on players on non-PC platform players who want pre-built ‘official’ cosmetic upgrade skins. Villains is a multiplayer cooperative map where only one survives.

Download the map from Mojang’s announcement page: http://www.mojang.com/2016/08/evil-live-our-devs-let-out-their-inner-villains-on-stream/
Watch Mojang play through the map (@_tomcc, @shoghicp, @darngeek, and @Marc_IRL): https://www.youtube.com/watch?v=_IfIMZMpRy4

- **Plagiatus**

Map Making Methods for Non-PC Platforms by RedstonerLabs

@StealthyExpert has been making a range of customised maps and concepts for XBOX and shares some of his ideas with us here. This Map Maker often uses an NBT Editor tool called MCC Tool Chest (http://mcctoolchest.weebly.com/) by @cynodonta for the creations:

> Minecraft is full of so many ideas just waiting to be born!.
> **Force Fields:** which will kill players instantly when walking through them using Charged Creepers for effect, and Zombies for the function, special areas that are just Endermite Entities that have -999999 LifeTime and have NoAI set to 1 and look great in maps where you want redstone contraptions or command contraptions to be of interest to the player
>
> 
>
> Who knows what map makers might come up with next and it’s up to you to make more concepts and invent crazier ideas to keep map making alive!
>
> - **ReadstonerLabs aka @StealthyExpert**

All in One by @EaterComputer

When @EaterComputer was thinking about what map to make all the ideas went into one project. All in One has 50 challenges ranging from minigames to jump puzzles.

Download this map from here: http://mcpedl.com/all-in-one-map/

Have you got a hot map tip for us? Contact us on Twitter @MapMakingMag or email MapMakingMag@gmail.com

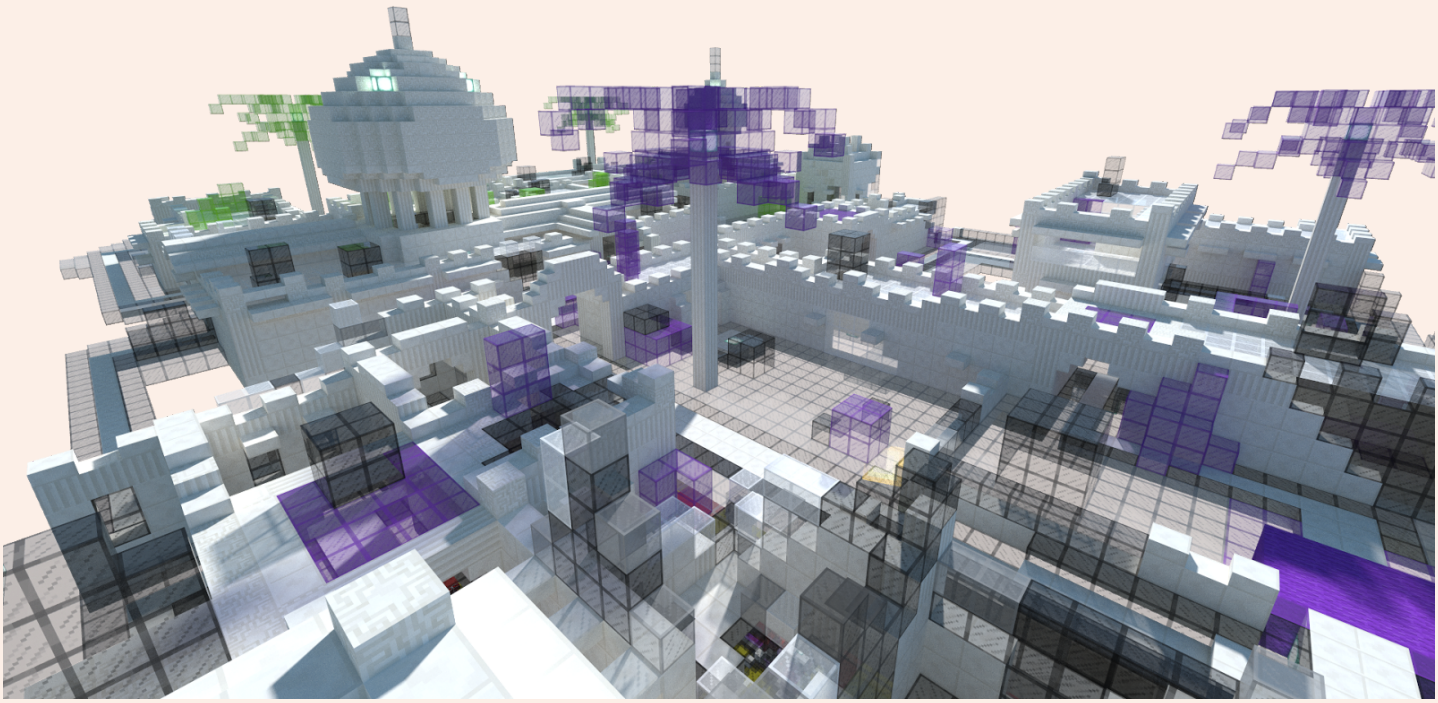
/Summon Panguino

@mwthecool talks to @Panguino who Shattered the Minecraft MiniGame scene with his first complete Minecraft Map!

Thanks for joining us. Talk us through your Minecraft experience.
Panguino: *I built a map called Shattered and I helped the Minecraft community understand the whole chunk border 64 block fill rule. There's just a couple of really technical command block things I helped debug and then work through back when that actually mattered. Now there's new command blocks, there's structure blocks, there's all that kind of stuff, so all the work I did has kind of been undone and it doesn't necessarily apply anymore but it was important at the time.*

You talked about Shattered. What exactly is that?
Panguino: *It's a multiplayer mini game based on shooting arrows. When the arrows hit glass blocks they degrade and eventually break, so it's sort of like a bow multiplayer map. The environment is very destructable, and then at the end of each game it resets and rebuilds the map and you can start over. There are different maps. A lot of people liked it because it was very different from a lot of the map they had played or seen before.*

What version was Shattered made for?
Panguino: *Version 1.8. I updated it for 1.9 and now i'm working on 1.10. There's been a lot of changes in how things are being done so I've had to sort of rework stuff over and over the past few patches.*



What do you think was the hardest mechanic to implement?
Panguino: *I think the hardest mechanic to implement is when I actually wanted to keep track of whose arrow was hitting. One of the things you can do in it is the mode called Fruit Buster in my map. You are shooting fruit. You are trying to shoot fruit faster than other people to get a score. If you kill them that knocks them down. Kind of king of the hill, I guess, but keeping track of who's arrow belongs to who is not something that's natively in Minecraft. Handling those that process of "okay this arrow belongs to this player" was a pretty big mechanic to overcome.*

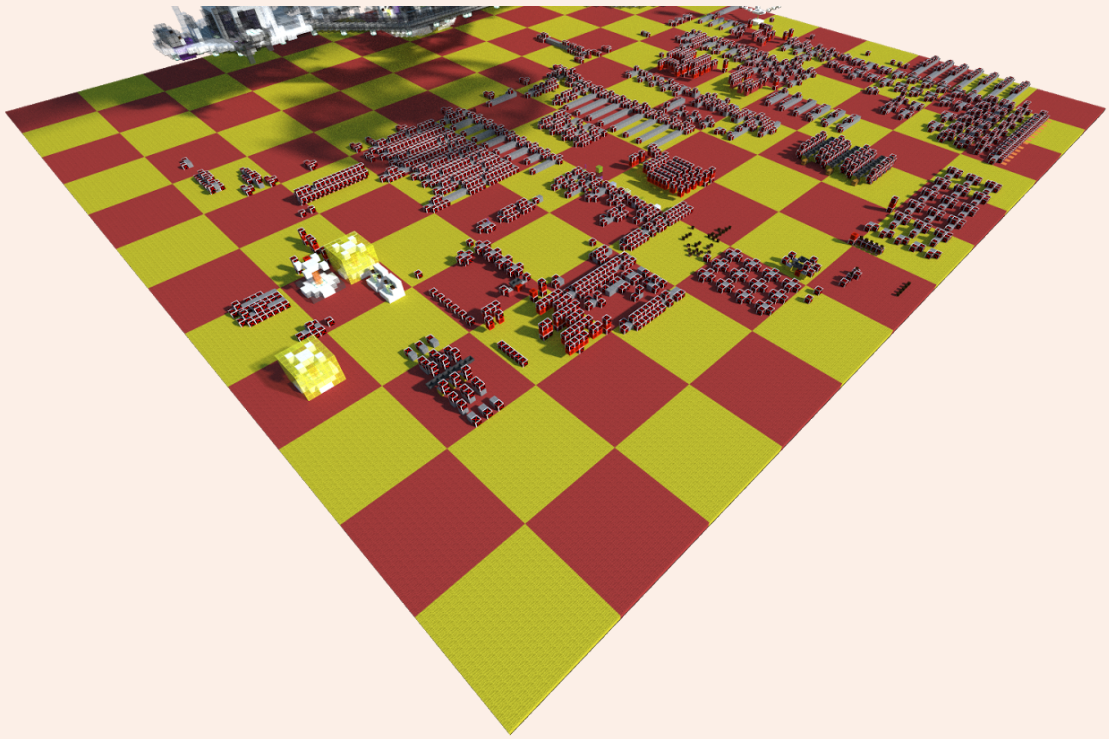
I do scoreboards but now you can use tags and maybe people understand tags better. As an arrow leaves the bow it will have no tag because it's just spawned out of your bow and so you're going to have to give it a tag and you're going to get some sort of ID. The way you detect a new arrow is it's not going to have a tag so you to find all the arrows that aren't tagged and then you're going to tag it with the same scoreboard as the player near it. Right as the players enters your server each player has to have a unique score and that's pretty easy to set up. You just have to squirt an increment as any player joins so then the arrow gets the same score. Now you have all these arrows flying through the air potentially and you have players with scores. All the arrows will have the same scoreboard score as the player.

This player needs to be rewarded once the arrow hit something so I need to go through and i need to subtract the player score from the arrow score and then figure out which ones are 0. There's a lot of sort of scoreboard math that has to go into play. Once you've sorted that out then you can actually say okay this player's team gets rewarded. He gets rewarded points because his arrow hit something that it should have our shouldn't have. You could go both ways. If you hit something you weren't supposed to you can take points away or or you can give them a negative effect. The arrows can all hit at the same time.

It was tough doing it. If you have one player playing is easy. Doing multiplayer when you have three arrows that can hit three different things at the same time in the same tick, you have to be able to handle it. Going through those scoreboard objectives three different ways … it is hard to visualize in your head when you're just writing code into blocks in a sequence.

Was the jump from 1.8 to 1.9 quickly stressful and hard for you?
Panguino: *No not really. It allowed me to simplify a lot. I liked the new command block stuff. It's hard to rework something, to redo it, but sometimes you just have to break it down and redo it because you can optimize it and make it better than it was before. A lot of times you'll do that anyway if you're building a map. That update forced me to do it but I think the map still worked. It just wasn't optimized and there's better way to do it, you know?*

What do you think was one of your biggest challenges originally making the map?
Panguino: *Our biggest challenge was I never really worked with anybody or knew how to do a lot of the stuff. My teacher was basically YouTube and Reddit. You come across a problem, like I came across that 64 block rule, and I just got sidetracked by "okay this doesn't work why doesn't it work?". When you look at it it's like "this should work" so there's a lot of sort of gotchas and things that will catch people up. It's not like Minecraft command blocks have documentation or a lot of examples like other programming languages have. If you have a problem with jquery or a website you can probably go on google and find the exact problem you've had, and people have answers to, it but Minecraft isn't like that. Finding out that there's a map making community and there's actually skype channels and now discord channels, and there's there's a community that you can ask questions to and people actually help you out helped me a lot. Originally I didn't know anybody. I just kind of got into it and it was pretty tough.*

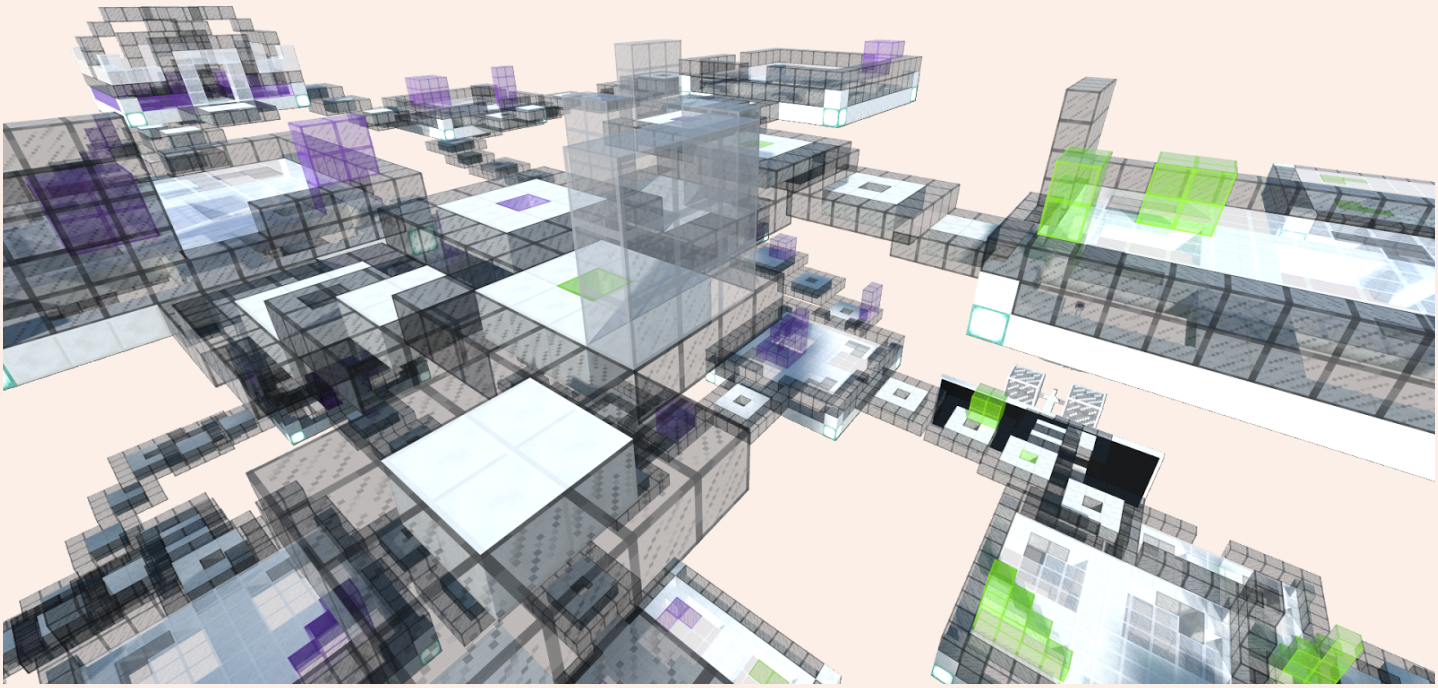


Have you made any the other many games besides Shattered?
Panguino: *Shattered really the only minigame I made by myself. I contributed to some of the voxel box stuff. I contributed to Muk-Luk Lodge (https://www.youtube.com/watch?v=fhpnmftrfn4). I worked on something they did for a museum Canaletto (https://www.youtube.com/watch?v=5XILTp9VqM8) where I worked on the stuff behind the scenes. They wanted like boat tours so I had basically had to make horses into boats. I'm not going to get into the super nitty-gritty of it but they had to do a lot of weird things and so I set those systems up.*

Was that your break into the Voxel Box?
Panguino: *Yes, because I had done some of the help with solving some of those weird issues that a lot of people were having. I got to meet Moesh and I got to meet a lot of other people in the community. My niche was making systems that you could reuse, like jQuery makes other things easy. I made tools for the Voxel box and I made systems that people can use when they make maps and command blocks so I designed the system that allowed them to do cooler stuff easier.*

You're like the guy in the background with the puppet strings working everything!
Panguino: *One of the things I helped develop is called the Voxel Board. It was big like a dev board where you put all your command blocks and stuff. There were a lot of different systems I made and helped to make for time, like timers and delay timers and different things you reuse a lot. There's a lot of things where you have to have some loop, or there's a lot of things we have, to something run every two seconds. There's just all these little systems i made in a process where if you follow these rules and use the different tags it made your life a lot easier. @CDFDMAN did some really cool lighting show effects and he had to write a big sequence so I made a tool for him where he could sequence stuff and add delays and pauses really easy and it made his work a lot easier to do something pretty incredible.*

How is making a minigame different for making like an adventure map or a story-based map?
Panguino: *When you're doing a mini game and you have to support multiplayer. There's a lot more that you have to think through programming wise from the very start. If you're expecting their only be one player the map it's kind of simple.When you do multiplayer it it makes it a lot more complex to handle certain events in certain circumstances and it can make it laggy. You have to think about performance. Single-player: you've got the one person working on their computer. When it's multiplayer if you're updating lots of chunks and doing lots of stuff, you've got ten people in the game, then everyone has to like download those things so testing performance and stuff becomes a lot more complex. I think the performance side was one of the big hang-ups for me when I worked on my map. I enjoyed making it perform well, because if it doesn't perform well I don't think it's going to be fun for a lot of people.*



What is it like being a part of a team and working together?
Panguino: *I think working together is very hard for a number of reasons, and it's very rewarding, and it's a better way to do it. When you're working in a team, especially in an environment like this, it's very creative. It's hard to agree upon things. What theme should our map be? What's the story? What color should stuff be? What textures? There's so much creativity and creative aspect to what Minecraft can be that getting a team of five or six people to agree on that is not easy. Especially if you're doing a side project or project for yourself. In a lot of the stuff we did for Voxel Box someone else had an idea and they wanted us to execute it so it took that that burden of having to be creative in a lot of ways out of our hands and it made that easier for us because then we didn't have to make those choices.*

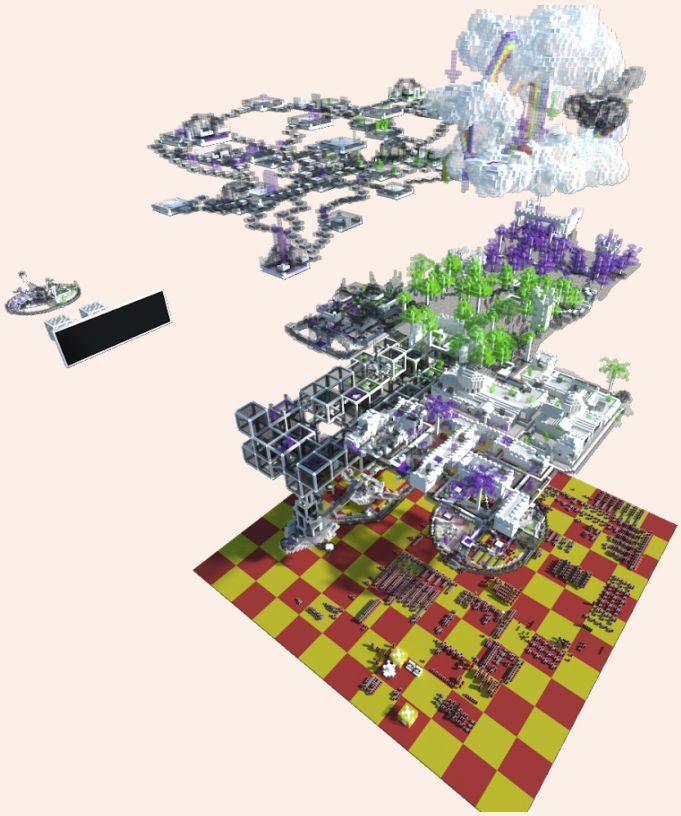
Working together can be rewarding because things get done faster. You can see the product at the end. Minecraft stuff can take a long time. There's people that worked on maps for forever, like years, and then they finally get done. You can get it done faster if you have a team of people working on it and you have specialists. I specialized in making some processes for people to use. Other people were really going to resource packs. Other people are the builders, you know, the guys that just built the terrains and made them look pretty. We were the mechanics team that just worked on the command block stuff. When you have specialized people that can focus their efforts things get done a lot more faster instead of having one guy having to know everything. When one person has to do all that it's very overwhelming you know

Are you working on anything right now?
Panguino: *I've been working on Shattered 2.0 and also some side stuff. I know Moesh has been working hard on taking what I started with making a system that people can reuse when they make maps. He's done some updates so now we've made a system where you can basically just work on code*

in a file in sublime to basically copy that whole code your clipboard and it creates that one command block (Ed: See Tech News, Issue 1). You paste that one command in the game and it will generate an array of command blocks. The benefit to that is if you make one little change in a line of command blocks in your system you don't have to go and copy and paste out 15 or 20 more blocks to put your commands all back. You can just compile it, copy paste into one block, run at interval, reset your system. It takes a lot of the copy and paste pain out of it. From what I started back just making a couple little systems is now turned into a much bigger system that a lot of people might be able to use and benefit from.

What else would you like to share with Map Makers?

Everyone has their idea of like what makes a good game. My reason has always been replay value: a game that you can just play over and over. There's different ways that a game can have good replay value. People go back and play Zelda. There's games you go back and you replay over and over that either you grew up with or used to play, and they're still good! If you can tap into that replay value I think that's what can make it successful because the marketing is very hard to do. You're competing against a lot of people doing minecraft stuff and there's no Steam where people can connect and see new games that come out, and servers that are already set up doing a multiplayer map, or even an adventure map, with the exception of Realms. It's hard to get in front of people. If you're a content creator like me that made a map you want people to play (that's really the goal, you want it to become popular you want people to know it's there) I think the easiest way to do that is making a map that's got good replay value and it's fun to play over and over, and different. That's what I did with shattered, or tried to do. Each time they played it was a little different. The map would destroy differently. There's some randomization to it so I guess that's what I would leave people with is: if you can add that replay value



Download the Shattered map: http://shatteredpvp.com/

/kill @p[name=Panguino]

End...?

Issue 2 has been a blast to prepare. We hope you enjoyed our feature on Mini Games. You might choose a Mini Game for your next map!

And so we return to our project: creating a map from scratch. In our last issue we decided on a game type, which for the **Map**[Mag] crew is Survival/Arena. Now it is time to develop the game design. Hints and ideas on how to approach this step are in the article on Project Management for Minecraft Projects earlier in this issue.

Our story will be that the Player works in the maintenance area of a spaceship and over time there are all sorts of challenging combat situations that occur as creatures enter the play area through the airlock. The purpose of the game will be to survive the challenges and progress through the levels.

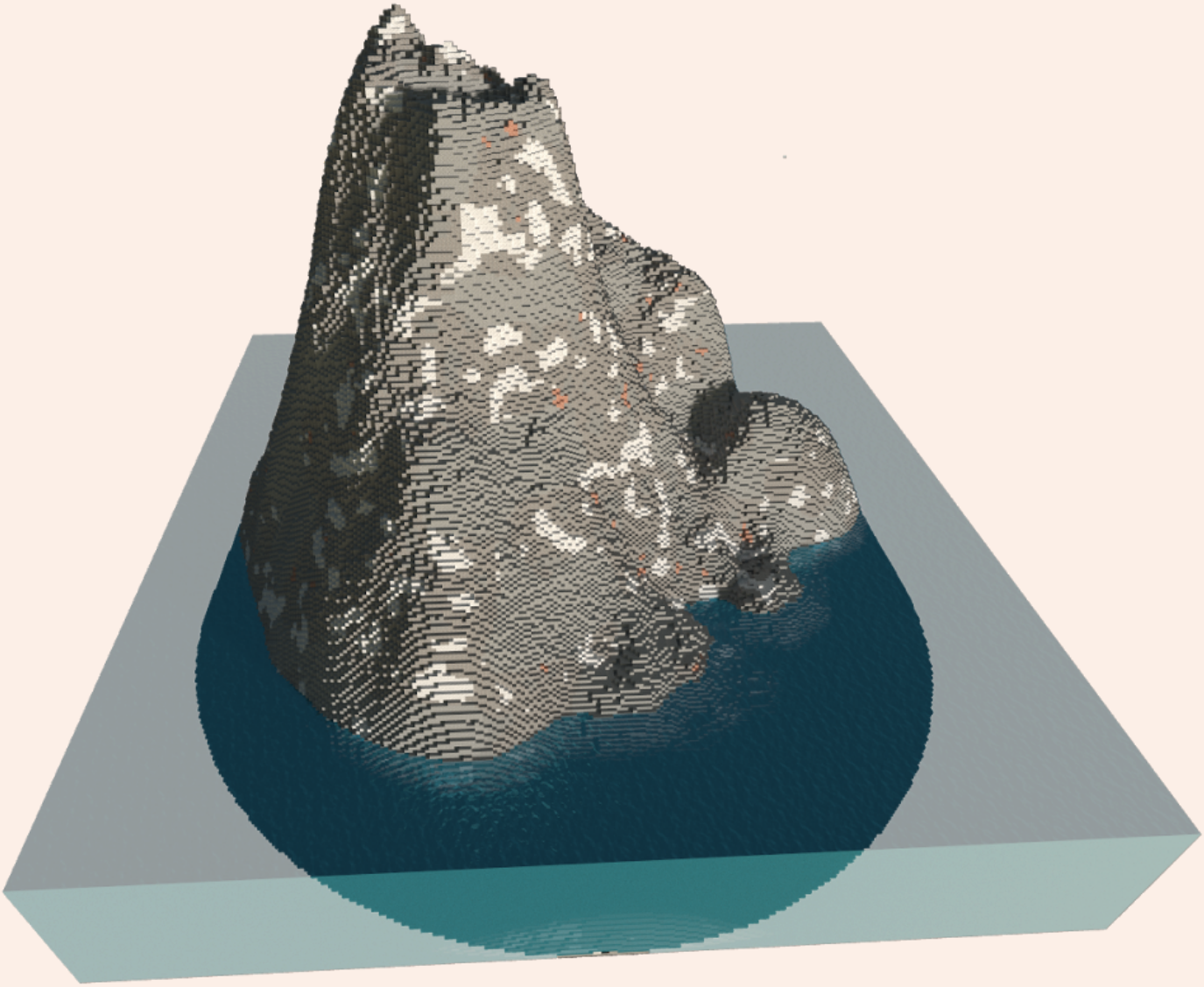
With a game like this the player's interest needs to be maintained and so we will need to evolve the challenges in ways that develop the story. Our challenges will involve a range of problems from nuisance alien bugs sneaking in, to full-blown assault teams that need to be repelled.

A good place to start, however, is by building a prototype of the play environment. This can give rise to design thinking around how the gameplay will feel, and even start you thinking about what sort of mechanics might work well. We will need some custom weaponry suitable for ship-board combat.

Our map will need a way of controlling each of the encounter events, possibly on a timer or according to a schedule. See our article on NUKED for ideas. A simple method of resource management will also be required so that the player has control over a technology tree of equipment that can be used to solve each challenge. In this issue we discovered ArmorStands and AreaEffectClouds. Both of these will be useful tools in implementing these ideas.

But first, let's think about building our section of the spaceship where the action takes place, including the airlock mechanism. In our next issue we will explore some of the tools and approaches you can take to construct fun playable environments.

Until then, continue the discussion on Reddit at http://www.reddit.com/r/mapmag or Twitter: @MapMakingMag or email: MapMakingMag@gmail.com



@MapMakingMag | MapMakingMag@gmail.com

Picture Credit:
Design input by RazeNave+J. Weaponised Villager by @Wergat. Safe Fire and Force Field by @StealthyExpert. NUKED ingame picture by @CrushedPixel (see https://www.youtube.com/watch?v=l3lOC36LsjQ) . All other images by @abrightmoore.

The Cover Font type was selected by @GamerGuppy and is Lato (http://www.latofonts.com/lato-free-fonts/) and is used under the SIL Open Font License.
We use **Chunky** by Jesper Öqvist and the community (http://chunky.llbit.se/) for renders.
We use **MCEDIT** by @Codewarrior0 and the community (http://www.mcedit.net) in the preparation of **Map**$^{\text{Mag}}$